goldprice-prediction

September 1, 2024

[1]:
```
import warnings
import itertools
import numpy as np
import matplotlib.pyplot as plt
warnings.filterwarnings("ignore")
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 15, 6
import pandas as pd


import statsmodels.api as sm
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.graphics.tsaplots import plot_pacf
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.ar_model import AR
from statsmodels.tsa.arima_model import ARMA, ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX


from math import sqrt

import matplotlib
matplotlib.rcParams['axes.labelsize'] = 14
matplotlib.rcParams['xtick.labelsize'] = 12
matplotlib.rcParams['ytick.labelsize'] = 12
matplotlib.rcParams['text.color'] = 'k'
import seaborn as sns

from random import random

from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error,␣
↪median_absolute_error, mean_squared_log_error
```

[2]:
```
goldprice = pd.read_csv("/content/drive/MyDrive/GoldPrice.csv")
```

[3]:
```python
headers = ['Date', 'Price', 'Open', 'High', 'Low', 'Chg%']
parse_dates = ['Date']
```

[4]:
```python
goldprice
```

[4]:
```
                Date    Price     Open    High      Low     Chg%
0     Sep 11, 2020  1957.35  1952.55  1963.3  1944.35 -0.0035
1     Sep 10, 2020  1964.30  1955.30  1975.2  1948.60  0.0048
2     Sep 09, 2020  1954.90  1939.40  1959.7  1926.30  0.0060
3     Sep 08, 2020  1943.20  1938.00  1948.3  1911.70  0.0031
4     Sep 07, 2020  1937.10  1940.70  1947.4  1930.45 -0.0018
…              …       …       …       …      …       …
2526  Jan 07, 2011  1368.50  1372.70  1377.2  1355.50 -0.0021
2527  Jan 06, 2011  1371.40  1374.80  1376.5  1368.90 -0.0015
2528  Jan 05, 2011  1373.40  1383.40  1384.0  1364.20 -0.0037
2529  Jan 04, 2011  1378.50  1409.60  1410.9  1375.80 -0.0310
2530  Jan 03, 2011  1422.60  1415.60  1423.9  1413.70  0.0011

[2531 rows x 6 columns]
```

[5]:
```python
goldprice.dtypes
```

[5]:
```
Date      object
Price    float64
Open     float64
High     float64
Low      float64
Chg%     float64
dtype: object
```

[6]:
```python
print ("Gold commodity has {} observations & {} features".format(*goldprice.
 ↪shape))
```

```
Gold commodity has 2531 observations & 6 features
```

[7]:
```python
def change_into_datetime(col):
    goldprice[col] = pd.to_datetime(goldprice[col])
```

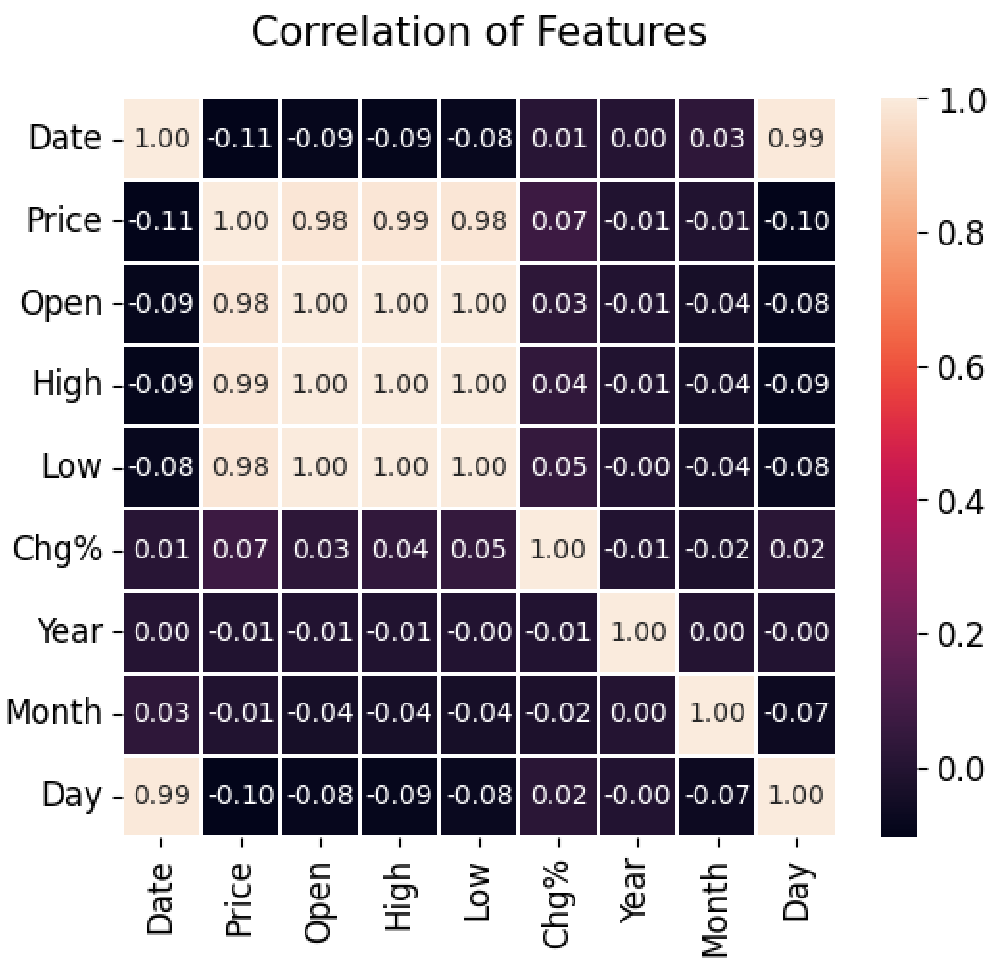
[8]:
```python
for i in ['Date']:
    change_into_datetime(i)
```

[9]:
```python
goldprice['Year'] = goldprice['Date'].dt.day
goldprice['Month'] = goldprice['Date'].dt.month
goldprice['Day'] = goldprice['Date'].dt.year
```

Visualization

[10]:
```
corr = goldprice.corr()
plt.figure(figsize = (6,5))
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values,
            annot=True,fmt='.2f',linewidths=0.30)
plt.title('Correlation of Features', y = 1.05, size=15)
```

[10]: Text(0.5, 1.05, 'Correlation of Features')



[11]:
```
df = pd.DataFrame(goldprice, columns = ['Date', 'Price'])
df.head()
```

[11]:

	Date	Price
0	2020-09-11	1957.35
1	2020-09-10	1964.30
2	2020-09-09	1954.90

```
3 2020-09-08  1943.20
4 2020-09-07  1937.10
```

[12]:
```
from IPython.display import display
display(df[:10].T)
```

```
                         0                    1                    2  \
Date   2020-09-11 00:00:00  2020-09-10 00:00:00  2020-09-09 00:00:00
Price              1957.35               1964.3               1954.9

                         3                    4                    5  \
Date   2020-09-08 00:00:00  2020-09-07 00:00:00  2020-09-06 00:00:00
Price               1943.2               1937.1              1940.65

                         6                    7                    8  \
Date   2020-09-04 00:00:00  2020-09-03 00:00:00  2020-09-02 00:00:00
Price               1934.3               1937.8               1944.7

                         9
Date   2020-09-01 00:00:00
Price               1978.9
```

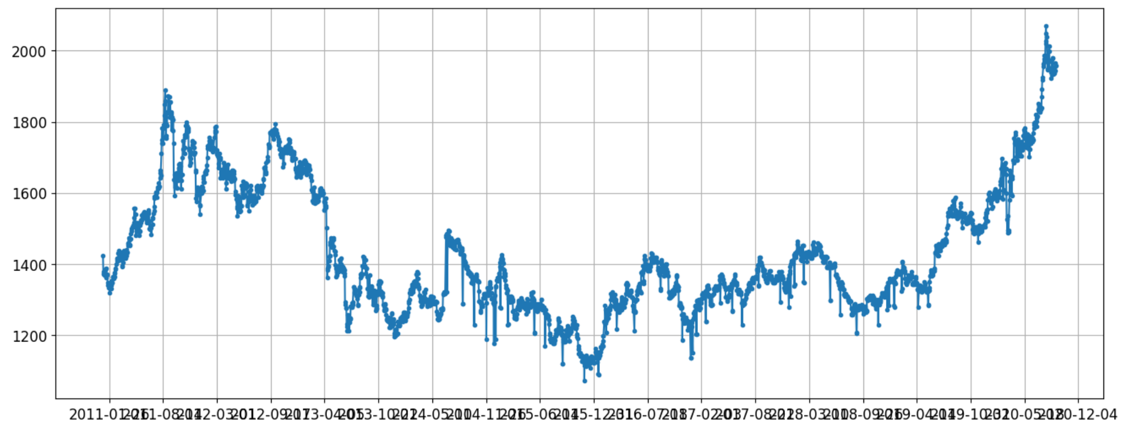
[13]:
```
y = df.set_index('Date')
```

[14]:
```
y.index
```

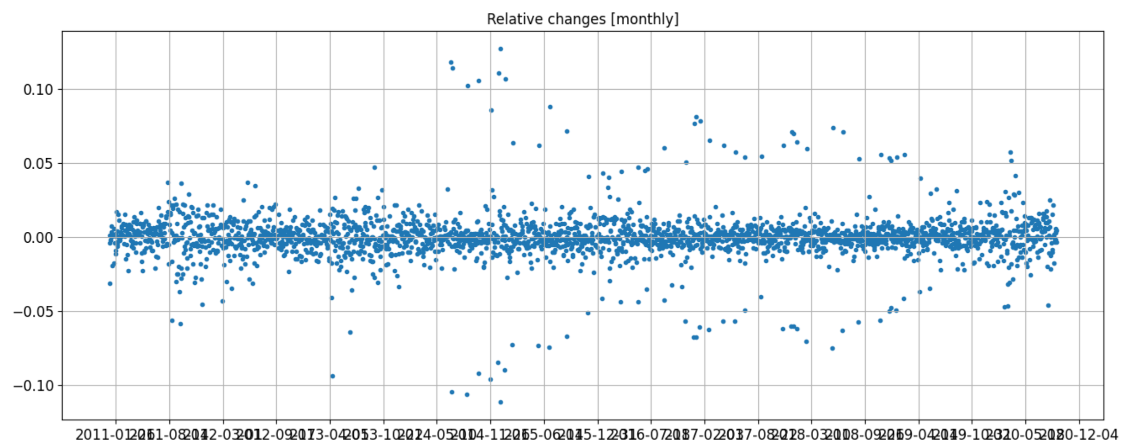
[14]:
```
DatetimeIndex(['2020-09-11', '2020-09-10', '2020-09-09', '2020-09-08',
               '2020-09-07', '2020-09-06', '2020-09-04', '2020-09-03',
               '2020-09-02', '2020-09-01',
               …
               '2011-01-14', '2011-01-13', '2011-01-12', '2011-01-11',
               '2011-01-10', '2011-01-07', '2011-01-06', '2011-01-05',
               '2011-01-04', '2011-01-03'],
              dtype='datetime64[ns]', name='Date', length=2531, freq=None)
```

Exploratory Data Analysis Explore price development

[15]:
```
fig, ax = plt.subplots(figsize=(16,6))
ax.plot(df.Date, df.Price, marker='.')
ax.xaxis.set_major_locator(plt.MaxNLocator(20)) # reduce number of x-labels
plt.grid()
plt.show()
```

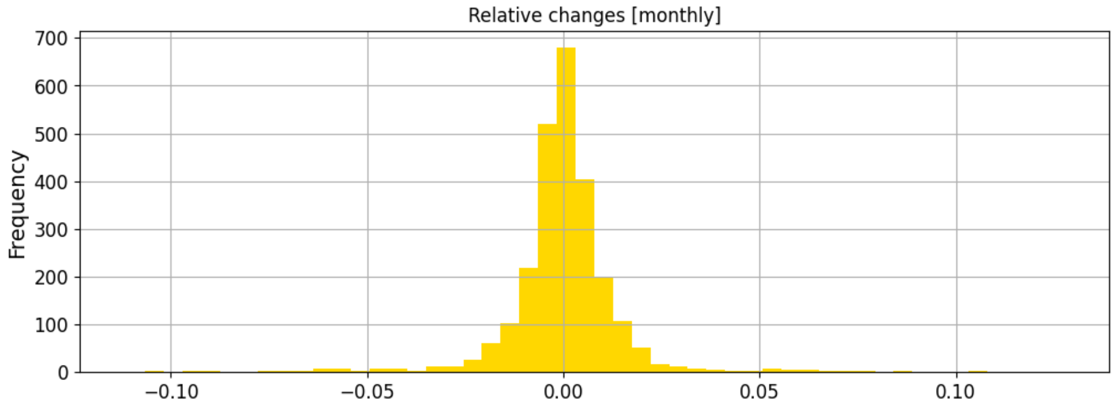



```
[16]: fig, ax = plt.subplots(figsize=(16,6))
      ax.scatter(df['Date'], goldprice['Chg%'], marker='.')
      ax.xaxis.set_major_locator(plt.MaxNLocator(20)) # reduce number of x-labels
      plt.title('Relative changes [monthly]')
      plt.grid()
      plt.show()
```



Relative changes distribution

```
[17]: plt.figure(figsize=(12,4))
      goldprice['Chg%'].plot(kind='hist', bins=50, color='gold')
      plt.title('Relative changes [monthly]')
      plt.grid()
      plt.show()
```



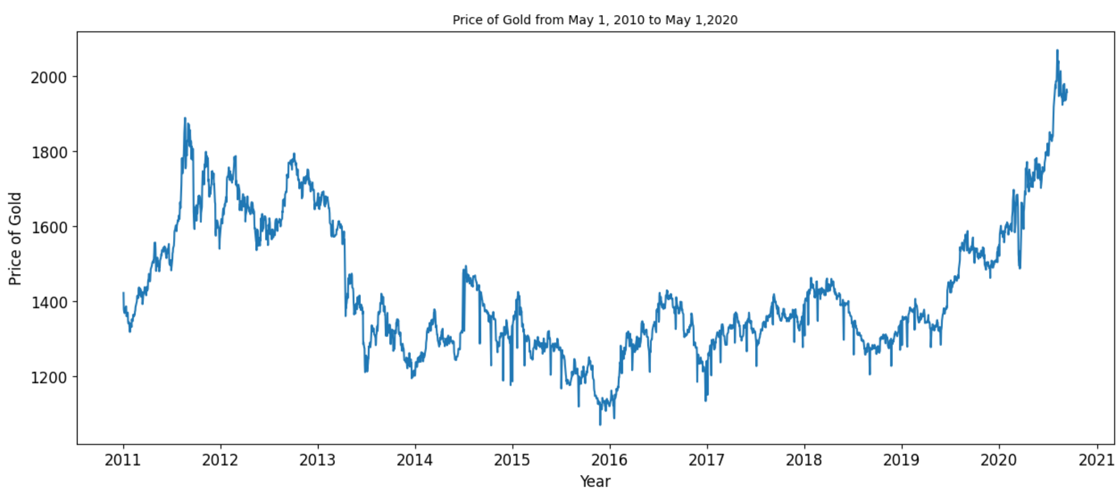
[18]:
```
y.isnull().sum()
```

[18]:
```
Price    0
dtype: int64
```

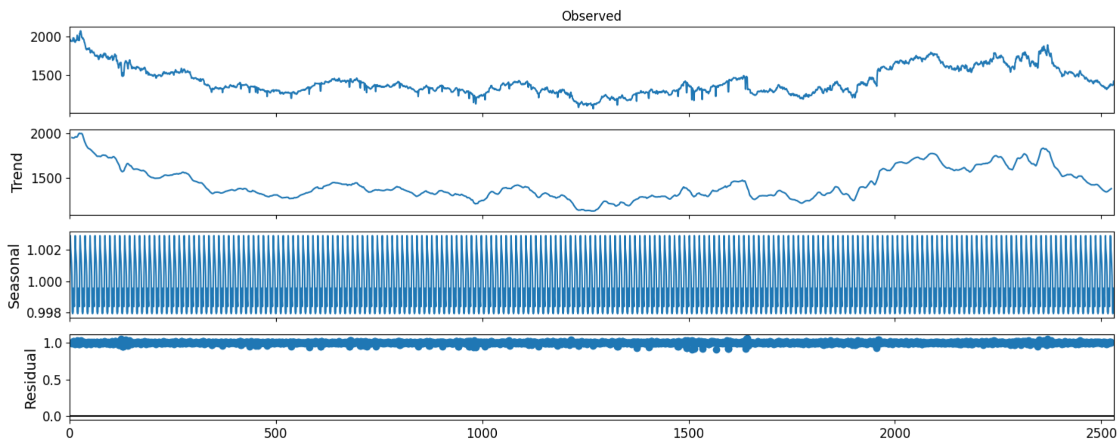
[19]:
```
stationary_check_gold = df.set_index(['Date'])

stationary_check_gold_price = stationary_check_gold['Price']
```

Non-Stationary time series of Gold

[20]:
```
plt.plot(stationary_check_gold_price)
plt.ylabel('Price of Gold', fontsize=12)
plt.xlabel('Year', fontsize=12)
plt.title('Price of Gold from May 1, 2010 to May 1,2020', fontsize=10)
plt.show()
```

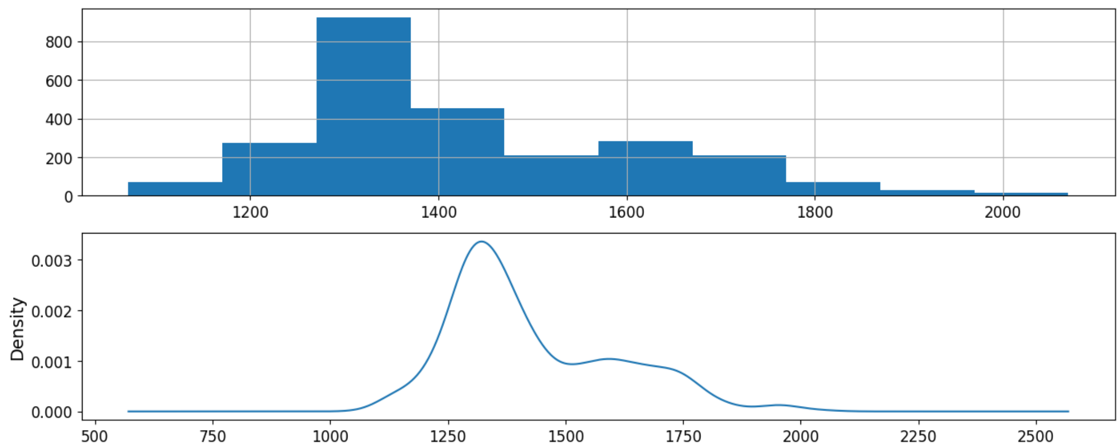


[25]:
```
import statsmodels.api as sm

# multiplicative
res = sm.tsa.seasonal_decompose(stationary_check_gold_price.values, period=12,␣
    ↪model="multiplicative")
fig = res.plot()
```



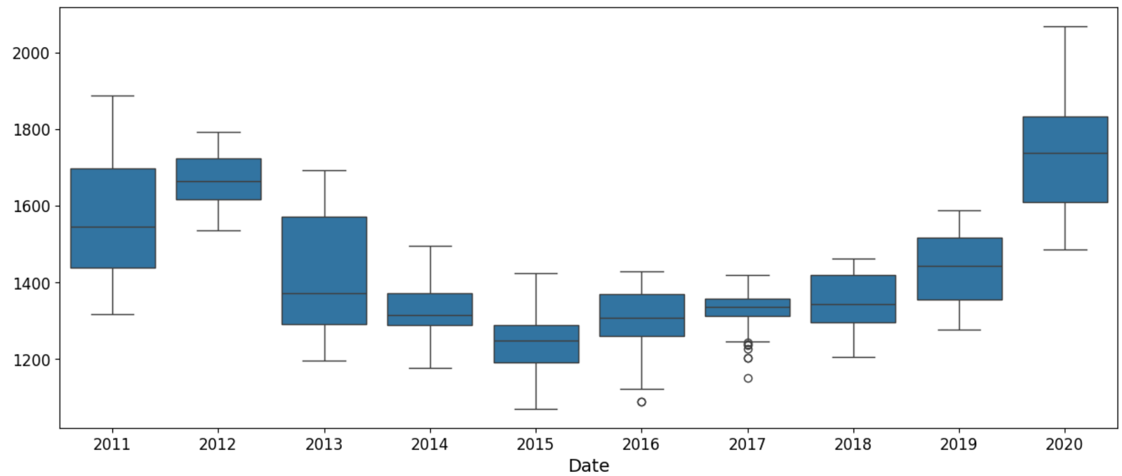
Box and Whisker Plots:

[26]:
```
from pandas import Series
from matplotlib import pyplot
pyplot.figure(1)
pyplot.subplot(211)
stationary_check_gold_price.hist()
pyplot.subplot(212)
stationary_check_gold_price.plot(kind='kde')
pyplot.show()
```



[28]:
```
import seaborn as sns
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize=(15, 6))
sns.boxplot(x=stationary_check_gold_price.index.year,␣
 ↪y=stationary_check_gold_price.values, ax=ax)
```

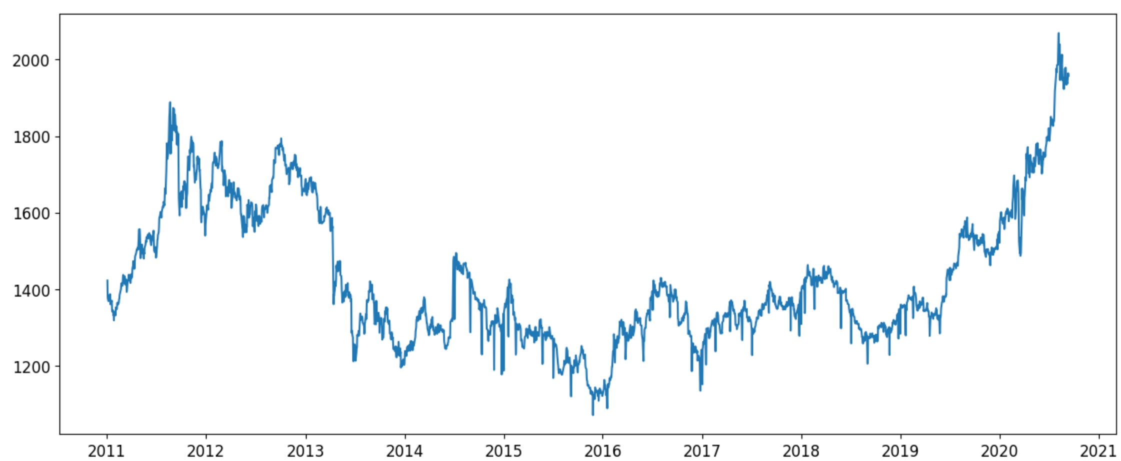
[28]:
```
<Axes: xlabel='Date'>
```



Plotting the Stationarity

[29]:
```
plt.plot(stationary_check_gold_price)
```

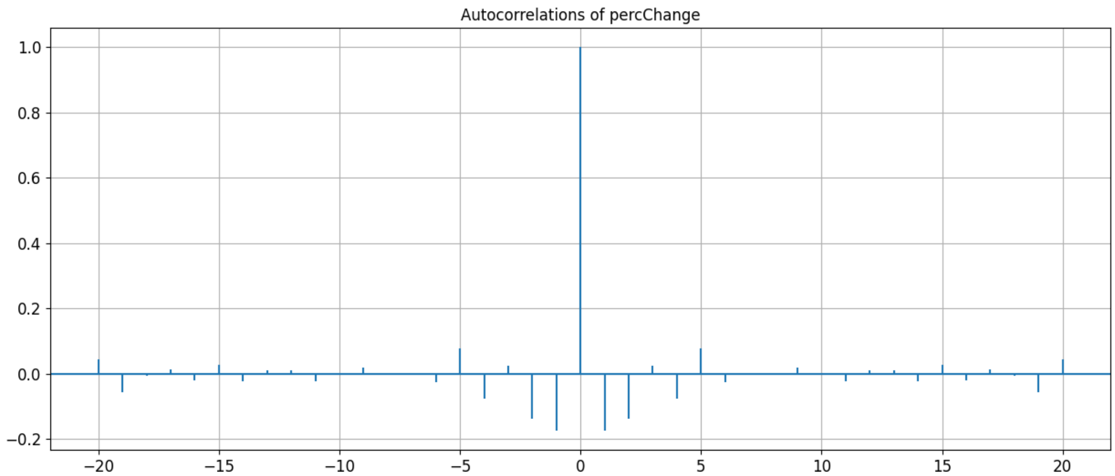
[29]:
```
[<matplotlib.lines.Line2D at 0x7d41d9ce2c80>]
```



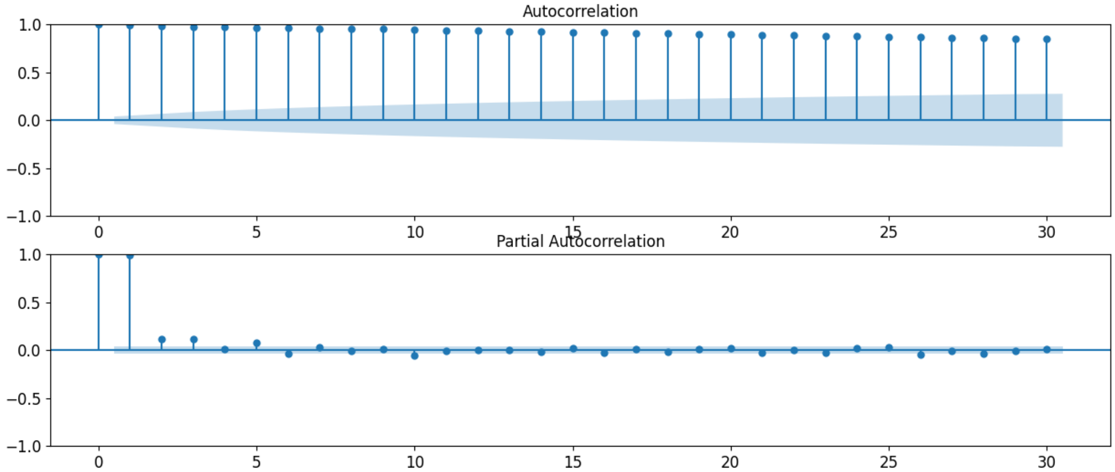
Analysing the Stationarity using Autocorrelation and Partial Autocorrelation functions

Autocorrelations

[30]:
```
plt.acorr(goldprice['Chg%'], maxlags=20)
plt.title('Autocorrelations of percChange')
plt.grid()
plt.show()
```



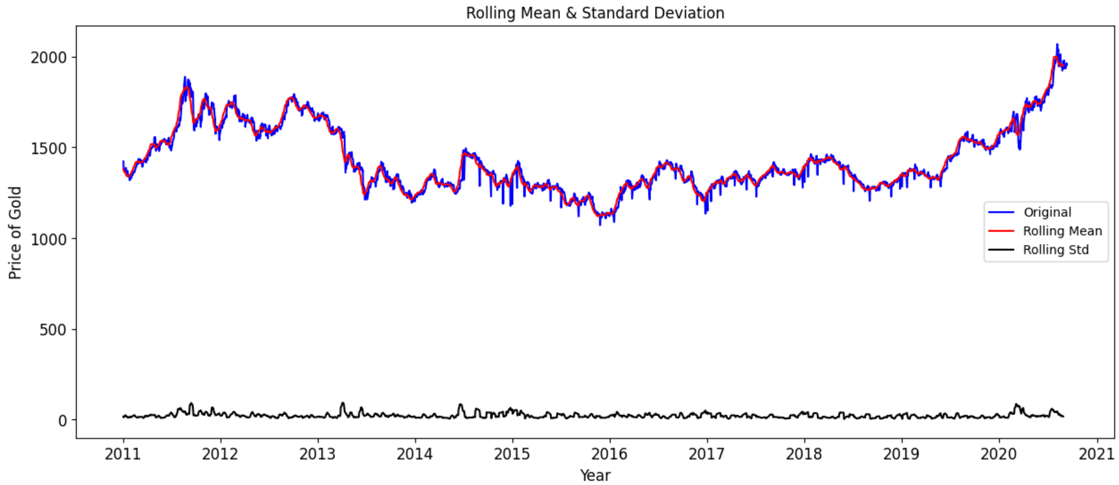
[31]:
```
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.graphics.tsaplots import plot_pacf

pyplot.figure()
pyplot.subplot(211)
plot_acf(stationary_check_gold_price, ax=pyplot.gca(), lags = 30)
pyplot.subplot(212)
plot_pacf(stationary_check_gold_price, ax=pyplot.gca(), lags = 30)
pyplot.show()
```



Plotting Rolling Statistics

[32]:
```
from statsmodels.tsa.stattools import adfuller
def test_stationarity(timeseries):

    #Determing rolling statistics
    #rolmean = pd.rolling_mean(timeseries, window=12)
    rolmean = timeseries.rolling(12).mean()
    rolstd = timeseries.rolling(12).std()

    #rolstd = pd.rolling_std(timeseries, window=12)

    #Plot rolling statistics:
    orig = plt.plot(timeseries, color='blue',label='Original')
    mean = plt.plot(rolmean, color='red', label='Rolling Mean')
    std = plt.plot(rolstd, color='black', label = 'Rolling Std')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.ylabel('Price of Gold', fontsize=12)
    plt.xlabel('Year', fontsize=12)
    plt.show(block=False)

    #Perform Dickey-Fuller test:
    print( 'Results of Dickey-Fuller Test:' )
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags␣
↪Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print (dfoutput)
```

```
test_stationarity(stationary_check_gold_price)
```




```
Results of Dickey-Fuller Test:
Test Statistic                   -3.030208
p-value                           0.032168
#Lags Used                        6.000000
Number of Observations Used    2524.000000
Critical Value (1%)              -3.432943
Critical Value (5%)              -2.862686
Critical Value (10%)             -2.567380
dtype: float64
```

Perfroming Augmented Dickey-Fuller Test

[33]:
```
from statsmodels.tsa.stattools import adfuller
print ('Results of Dickey-Fuller Test:')
dftest = adfuller(stationary_check_gold_price, autolag='AIC')
dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags␣
 ↪Used','Number of Observations Used'])
for key,value in dftest[4].items():
    dfoutput['Critical Value (%s)'%key] = value
print (dfoutput)
```

```
Results of Dickey-Fuller Test:
Test Statistic                   -3.030208
p-value                           0.032168
#Lags Used                        6.000000
Number of Observations Used    2524.000000
Critical Value (1%)              -3.432943
Critical Value (5%)              -2.862686
Critical Value (10%)             -2.567380
```

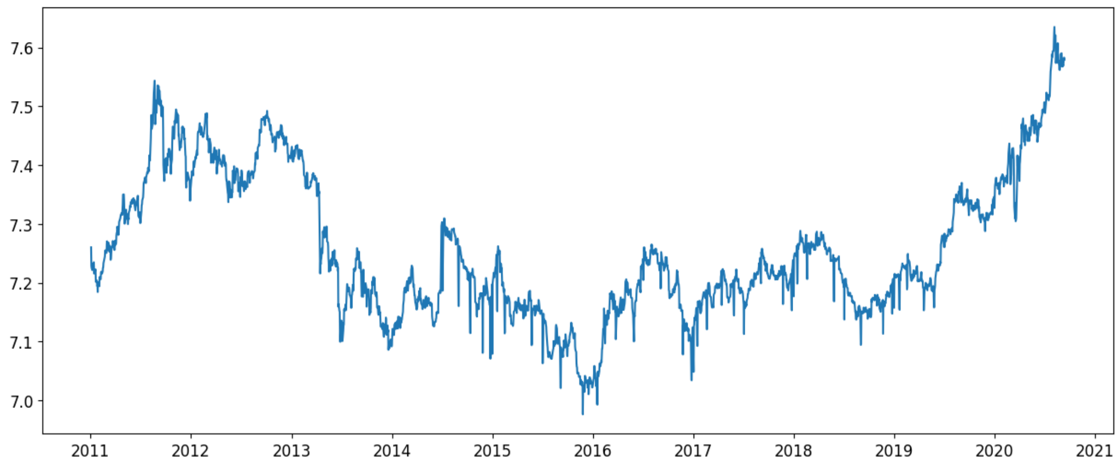
```
dtype: float64
```

[34]:
```python
def test_stationarity(timeseries):

    #Determing rolling statistics
    rolmean = timeseries.rolling(12).mean()
    rolstd = timeseries.rolling(12).std()

    #Plot rolling statistics:
    orig = plt.plot(timeseries, color='blue',label='Original')
    mean = plt.plot(rolmean, color='red', label='Rolling Mean')
    std = plt.plot(rolstd, color='black', label = 'Rolling Std')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.show(block=False)

    #Perform Dickey-Fuller test:
    print ('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags␣
↪Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print (dfoutput)
```

Making Time Series Stationary using Log Scale Transformation

[35]:
```python
ts_log = np.log(stationary_check_gold_price)
plt.plot(ts_log)
```

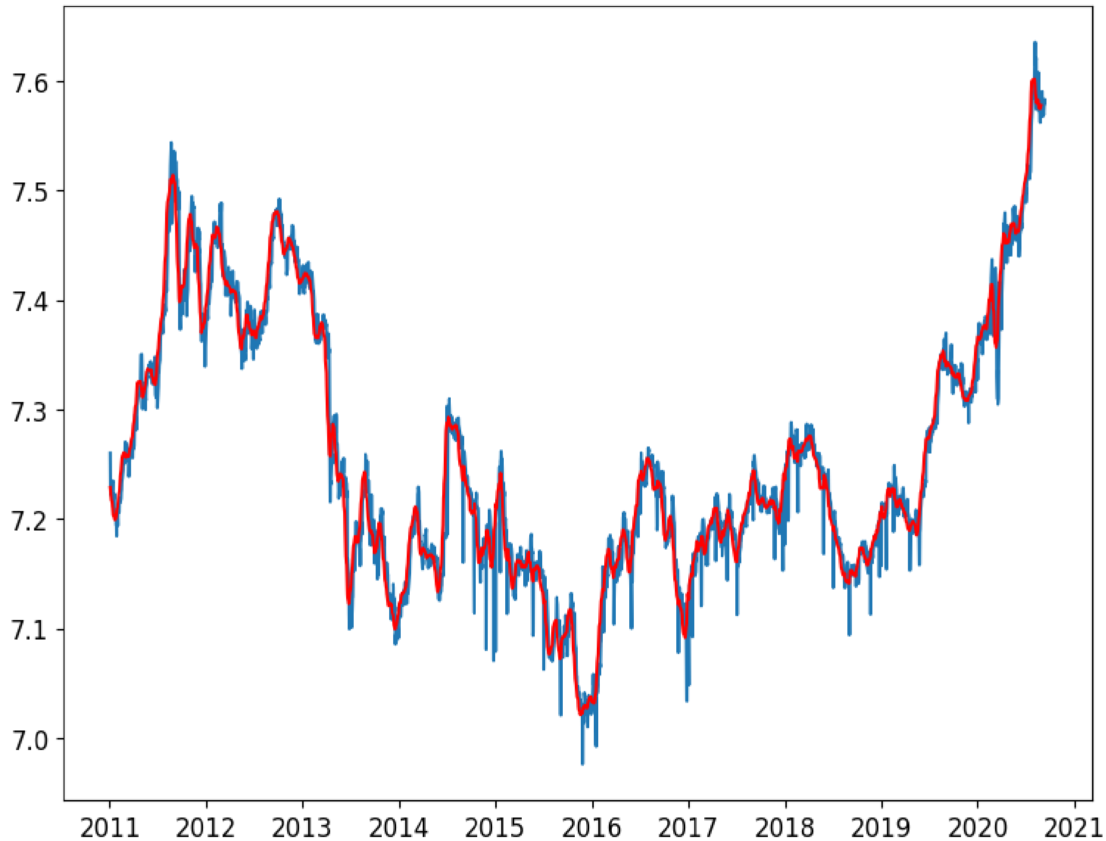
[35]:
```
[<matplotlib.lines.Line2D at 0x7d41d9add3f0>]
```



Using multiple techniques to remove Trend - Smoothing

Moving Average

[36]:
```
moving_avg = ts_log.rolling(window=12,center=False).mean()  # taking average of␣
↪LAST 2 years (36-12) values
plt.figure(figsize=(9,7))
plt.plot(ts_log)
plt.plot(moving_avg, color='red')
plt.show()
```

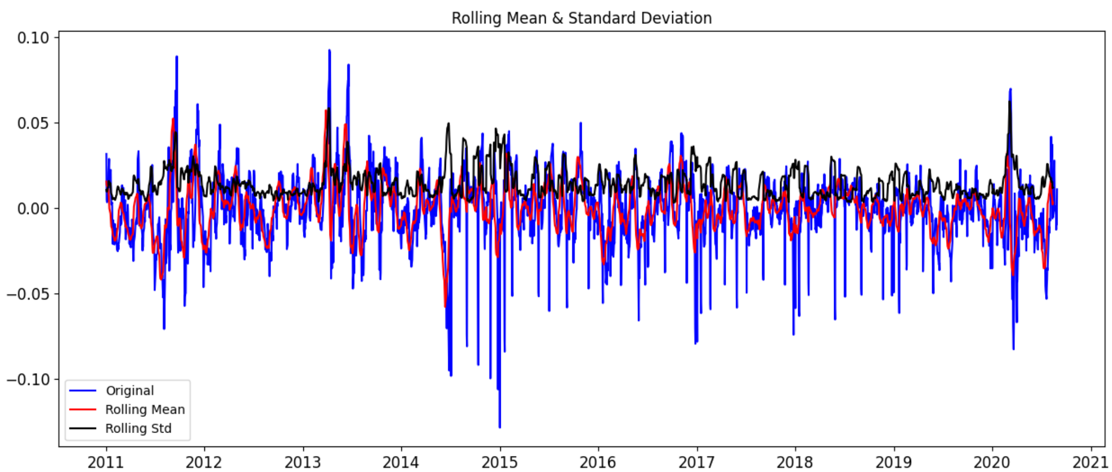


[37]:
```
ts_log_moving_avg_diff = ts_log - moving_avg
ts_log_moving_avg_diff.head(12)
```

[37]:
```
Date
2020-09-11          NaN
2020-09-10          NaN
2020-09-09          NaN
2020-09-08          NaN
2020-09-07          NaN
2020-09-06          NaN
```

```
2020-09-04          NaN
2020-09-03          NaN
2020-09-02          NaN
2020-09-01          NaN
2020-08-31          NaN
2020-08-28     0.010728
Name: Price, dtype: float64
```

[38]:
```
ts_log_moving_avg_diff.dropna(inplace=True)
test_stationarity(ts_log_moving_avg_diff)
```




```
Results of Dickey-Fuller Test:
Test Statistic                -1.194664e+01
p-value                        4.414553e-22
#Lags Used                     1.100000e+01
Number of Observations Used    2.508000e+03
Critical Value (1%)           -3.432960e+00
Critical Value (5%)           -2.862693e+00
Critical Value (10%)          -2.567384e+00
dtype: float64
```

Splitting the data

[39]:
```
X = goldprice.drop(['Date','Price'],axis=1)
X
```

[39]:

	Open	High	Low	Chg%	Year	Month	Day
0	1952.55	1963.3	1944.35	-0.0035	11	9	2020
1	1955.30	1975.2	1948.60	0.0048	10	9	2020
2	1939.40	1959.7	1926.30	0.0060	9	9	2020
3	1938.00	1948.3	1911.70	0.0031	8	9	2020

```
4     1940.70  1947.4  1930.45 -0.0018     7      9  2020
…        …       …        …       …     …     …    …
2526  1372.70  1377.2  1355.50 -0.0021     7      1  2011
2527  1374.80  1376.5  1368.90 -0.0015     6      1  2011
2528  1383.40  1384.0  1364.20 -0.0037     5      1  2011
2529  1409.60  1410.9  1375.80 -0.0310     4      1  2011
2530  1415.60  1423.9  1413.70  0.0011     3      1  2011

[2531 rows x 7 columns]
```

[40]:
```python
X.shape
```

[40]:
```
(2531, 7)
```

[41]:
```python
y = goldprice['Price']
y
```

[41]:
```
0       1957.35
1       1964.30
2       1954.90
3       1943.20
4       1937.10
          …
2526    1368.50
2527    1371.40
2528    1373.40
2529    1378.50
2530    1422.60
Name: Price, Length: 2531, dtype: float64
```

[42]:
```python
y.shape
```

[42]:
```
(2531,)
```

[43]:
```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
↪random_state=2)
```

Using Random Forest Regressor

[44]:
```python
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
```

[45]:
```python
def predict(algorithm):
    model = algorithm.fit(X_train,y_train)
    print('Training Score: {}'.format(model.score(X_train,y_train)))
```

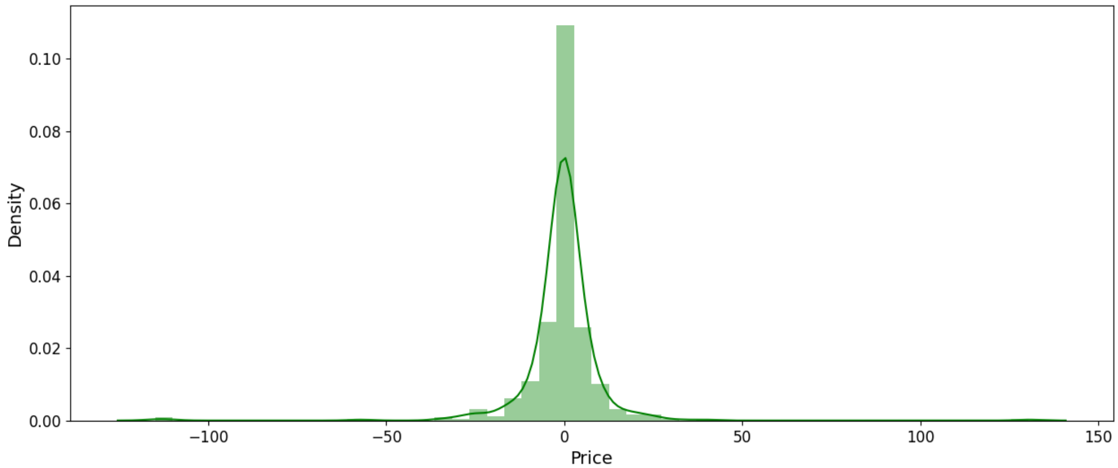
```
    preds = model.predict(X_test)
    print('Predictions are: {}'.format(preds))
    print('\n')

    r2_score = metrics.r2_score(y_test,preds)
    print('r2_score is:{}'.format(r2_score))

    print('MAE:',metrics.mean_absolute_error(y_test,preds))
    print('MSE:',metrics.mean_squared_error(y_test,preds))
    print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test,preds)))
    sns.distplot(y_test-preds,color='green')
```

[46]:
```
from sklearn.ensemble import RandomForestRegressor

predict(RandomForestRegressor())
```

```
Training Score: 0.9995326156264086
Predictions are: [1280.284  1373.936  1346.3088 1717.1    1333.9285 1290.06
1143.17
 1345.377  1265.7441 1304.66   1713.792  1344.5534 1360.933  1282.2395
 1571.057  1313.761  1332.64   1405.826  1524.6125 1335.533  1755.356
 1337.6092 1416.282  1265.653  1328.467  1539.271  1748.603  1202.341
 1593.175  1960.119  1360.125  1335.685  1678.566  1398.87   1322.862
 1361.148  1586.6135 1394.888  1650.088  1368.564  1195.2665 1362.657
 1427.808  1243.055  1312.154  1284.9925 1627.75   1367.933  1444.424
 1335.351  1279.1755 1595.448  1355.406  1543.276  1445.99   1304.636
 1272.199  1535.572  1265.2307 1520.025  1372.826  1755.281  1754.488
 1583.347  1347.636  1640.882  1247.071  1676.051  1551.4415 1496.874
 1629.197  1454.383  1580.9505 1678.369  1568.7385 1560.9535 1231.627
 1233.0525 1323.9325 1257.384  1358.045  1493.05   1350.447  1340.049
 1568.87   1596.266  1296.37   1446.255  1601.803  1573.794  1279.106
 1322.667  1411.915  1433.752  1609.277  1535.5935 1788.911  1342.066
 1280.928  1272.535  1788.894  1252.0235 1416.303  1198.4405 1367.042
 1287.844  1453.178  1496.375  1574.127  1414.075  1732.88   1429.726
 1388.8145 1939.9085 1676.575  1649.188  1306.045  1291.163  1762.267
 1427.701  1327.04   1403.615  1404.66   1424.048  1177.566  1240.883
 1218.034  1525.802  1413.176  1619.124  1553.8695 1404.57   1143.189
 1433.976  1518.7705 1216.396  1311.642  1215.813  1373.649  1400.146
 1292.576  1328.996  1369.152  1214.763  1939.7095 1610.31   1342.01
 1331.091  1742.428  1574.766  1317.85   1949.232  1739.853  1403.768
 1268.359  1236.638  1237.926  1409.819  1302.555  1300.666  1298.193
 1275.7    1424.89   1203.0595 1355.355  1287.6157 1740.     1368.851
 1264.0625 1284.9605 1301.397  1350.403  1428.395  1450.913  1729.255
 1141.49   1358.912  1610.87   1615.234  1574.395  1537.0635 1330.052
 1544.921  1503.674  1768.011  1239.482  1298.69   1602.084  1184.1695
 1238.318  1415.758  1737.4045 1367.201  1123.302  1274.473  1495.1585
 1367.199  1440.605  1325.55   1725.986  1282.6355 1282.6885 1368.193
```

```
 1339.5052 1419.001  1350.185  1150.9085 1662.059  1347.073  1321.826
 1418.196  1335.428  1297.544  1282.0895 1240.549  1774.699  1645.294
 1574.9245 1271.6479 1748.626  1385.377  1334.204  1299.69   1407.538
 1272.23   1591.833  1317.625  1613.852  1543.2455 1248.943  1657.308
 1280.894  1344.697  1360.569  1435.415  1343.1062 1589.079  1375.015
 1503.312  1698.624  1274.847  1659.156  1718.695  1467.636  1313.416
 1239.516  1134.082  1662.477  1293.686  1839.177  1512.1775 1350.0715
 1286.484  1299.321  1270.2062 1489.611  1283.1885 1759.891  1430.149
 1418.83   1286.7127 1570.464  1773.178  1291.8086 1280.677  1697.425
 1345.842  1753.009  1274.534  1276.766  1373.281  1311.873  1391.867
 1304.167  1324.705  1205.0805 1645.087  1772.837  1587.929  1414.664
 1228.0215 1434.986  1436.285  1553.641  1341.4    1787.464  1397.335
 1744.654  1668.396  1520.765  1765.879  1229.636  1396.591  1319.801
 1424.923  1335.923  1450.48   1338.542  1352.7698 1639.904  1335.383
 1412.53   1342.9614 1272.023  1812.103  1392.2735 1253.633  1387.261
 1887.0695 1205.182  1297.367  1665.399  1186.951  1324.705  1327.3945
 1506.48   1666.703  1359.257  1502.39   1207.059  1760.337  1230.338
 1237.975  1274.508  1298.797  1429.659  1252.33   1460.306  1284.9941
 1343.08   1515.962  1318.547  1331.87   1848.261  1669.387  1324.395
 1141.434  1381.712  1303.761  1300.923  1756.757  1618.135  1275.082
 1598.949  1314.045  1615.067  1272.839  1322.478  1374.277  1647.97
 1580.604  1318.431  1298.801  1848.6    1275.237  1604.483  1587.434
 1249.184  1383.004  1325.213  1284.6795 1544.0855 1528.17   1499.176
 1277.228  1371.278  1463.493  1278.792  1521.479  1505.7215 1580.3235
 1227.2035 1365.565  1716.457  1239.129  1453.881  1297.685  1525.288
 1417.523  1280.8465 1559.998  1342.1474 1213.264  1297.121  1355.454
 1378.231  1559.4215 1306.721  1120.411  1388.618  1334.164  1386.327
 1556.4525 1340.3798 1218.16   1578.8855 1587.168  1713.379  1293.8672
 1380.904  1740.501  1535.849  1306.731  1560.998  1282.239  1322.827
 1269.831  1378.18   1120.689  1342.7122 1396.835  1352.36   1619.37
 1402.718  1369.075  1640.97   1348.1452 1375.191  1329.921  1425.451
 1310.316  1317.034  1276.74   1497.257  1378.872  1447.622  1320.997
 1450.714  1291.6698 1137.2355 1544.703  1828.537  1243.111  1702.196
 1373.111  1288.4    1283.939  1288.241  1662.262  1434.051  1220.948
 1705.5985 1452.199  1677.187  1628.536  1288.9545 1288.1835 1269.4212
 1393.568  1864.5115 1225.286  1292.3362 1655.297  1329.163  1369.861
 1644.714  1787.389  1350.921  1499.595  1308.871  1615.984  1304.205
 1399.386  1670.848  1345.495  1335.02   1374.207  1360.832  1332.922
 1286.5175 1289.9915 1357.008  1637.414  1599.287  1371.674  1415.156
 1150.6645 1612.746  1313.577  1948.552  1306.125  1205.7555 1334.283
 1826.144  1575.948  1342.817  1241.407  1271.877  1656.205  1669.139
 1743.004  1743.756  1275.721  1290.9642 1310.141  1338.487  1251.379
 1255.367  1727.294  1644.599  1140.818  1893.63   1182.7365 1809.622
 1247.115  1574.4365 1134.0475]


r2_score is:0.9953972206753653
MAE: 5.248176923076879
```

```
MSE: 146.969150953944
RMSE: 12.123083393012852
```



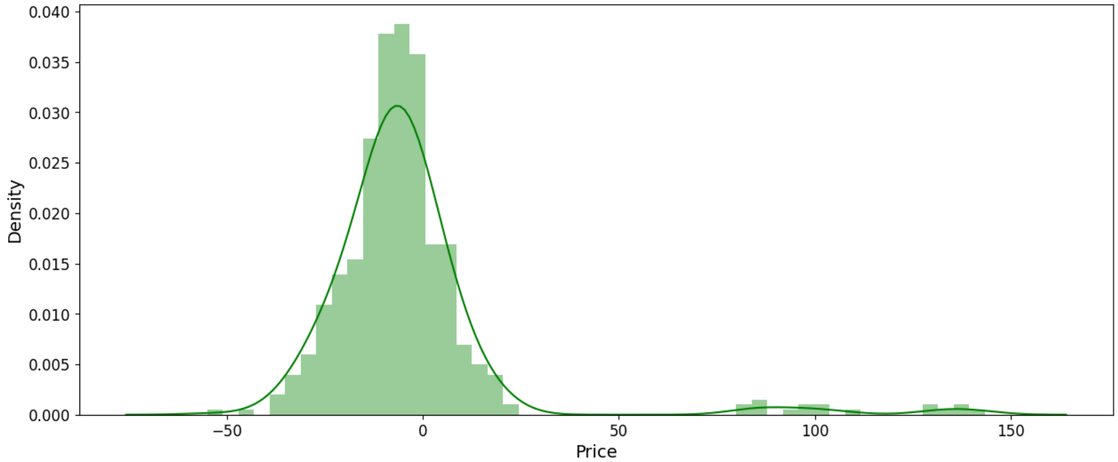
Using Linear Regression

[47]:
```python
from sklearn.linear_model import LinearRegression

predict(LinearRegression())
```

```
Training Score: 0.9720489762983339
Predictions are: [1287.46509401 1381.1335197  1349.02699003 1708.15351128
1337.6139623
 1305.2974777  1176.84026124 1348.79406241 1274.67616002 1316.54671617
 1722.3847627  1348.61530229 1370.70980134 1292.16250098 1569.34503439
 1316.63295296 1335.3827299  1411.79896563 1527.80814535 1330.94094103
 1739.072931   1341.43834851 1422.95738006 1270.20100171 1332.58185012
 1548.35263609 1740.92377919 1218.09105188 1585.98911985 1952.22431769
 1376.77944912 1345.10439649 1675.54758129 1278.10874728 1324.17910345
 1363.29713204 1602.85035255 1396.97035239 1646.77730068 1375.16890577
 1217.21910521 1392.55717245 1419.43210369 1265.89287443 1335.27249572
 1288.03513898 1624.16678634 1369.81739241 1439.40405146 1337.01326427
 1301.14760161 1597.38675689 1369.60254536 1544.02617924 1442.70147853
 1309.45200532 1285.29685425 1540.80458533 1272.27537545 1541.33150534
 1390.21604477 1749.5876601  1750.43403024 1610.78462962 1356.7009396
 1658.18878433 1275.83479596 1669.92450397 1552.3960604  1502.95088031
 1643.56574967 1457.81301702 1638.51558368 1673.63299606 1596.05803604
 1563.74676947 1250.81659824 1263.19587148 1329.23955484 1277.97650253
 1372.27202603 1503.39506273 1370.37387817 1261.63943477 1570.6992924
 1608.98195708 1305.92412612 1445.60970697 1610.27059107 1571.66893787
 1299.99729359 1327.84813092 1416.42169924 1444.7878836  1603.58268885
 1534.65638009 1784.98938864 1266.02103589 1303.99663792 1285.33871336
 1771.02421301 1277.44944975 1411.32748108 1217.36068073 1372.76800692
```

1298.52122991	1466.10247135	1503.91938831	1572.92900696	1412.83043475
1717.13018676	1436.22308454	1392.42030751	1910.45795641	1667.99553265
1644.70818267	1325.56365099	1294.62688288	1798.61507758	1419.90561608
1330.4122285	1406.27765129	1412.69240658	1418.8386838	1185.92900095
1266.1319676	1238.64226753	1529.20813842	1426.26816898	1620.88051596
1564.82578627	1400.1504797	1151.22123977	1446.05441156	1520.09768467
1232.11536487	1313.04908963	1241.38370612	1368.40385173	1408.17590513
1305.02092131	1337.61557807	1399.1602053	1234.06918365	1919.58256233
1618.39695183	1371.150752	1348.09955995	1739.54556113	1578.09020659
1321.96399992	1902.5397502	1730.6735642	1405.33436631	1272.8031996
1255.22415557	1258.2988033	1435.26168561	1224.69885287	1306.02084713
1301.76956418	1288.08728321	1432.98382815	1225.75970423	1355.49371261
1299.99761354	1730.15668795	1385.13874988	1282.92232737	1288.00050536
1318.64949269	1354.02320052	1423.48083159	1443.77622476	1743.24119545
1169.58331966	1372.42493958	1606.6949781	1616.18682199	1576.20106239
1550.39194028	1331.57234704	1551.16752295	1512.15753042	1767.68046735
1246.79928778	1295.58655892	1610.58892367	1200.75401012	1243.92641782
1412.31754624	1722.14086748	1406.91747098	1150.29637757	1281.60404293
1502.95846462	1374.42061468	1340.89091495	1328.26921456	1712.94464069
1293.35177788	1288.28766376	1407.93956664	1347.38267034	1295.54380588
1362.55148605	1163.14046017	1648.02161139	1359.51446545	1337.73441208
1416.60705489	1336.49624764	1310.51587107	1295.13203388	1261.68885031
1776.06444172	1638.42288073	1577.5275269	1278.72180676	1748.33330085
1392.0855222	1347.46875012	1306.57554187	1414.85853338	1278.03099581
1587.84246566	1336.99818957	1607.09873251	1559.49103418	1257.35869761
1647.23920508	1286.4973157	1366.8412935	1380.22684303	1431.91687984
1344.03580551	1589.15392876	1376.13502306	1515.48256805	1694.72458875
1297.55353969	1674.66256768	1706.62361585	1480.72650344	1331.43243241
1261.45972798	1161.61732396	1653.16269498	1310.33293738	1837.55834058
1518.18506851	1346.43911745	1293.96756556	1304.51815939	1281.91127042
1498.17681156	1290.54882577	1747.17771768	1308.56353	1433.67049426
1296.47693219	1571.78705856	1759.25277065	1293.44093606	1293.80906491
1700.60535502	1338.23396596	1747.59269154	1180.80263609	1293.4703229
1384.25500994	1326.65624637	1391.90538072	1311.89298289	1331.7967046
1224.56783748	1638.54087226	1762.05449326	1588.22862191	1418.87644313
1219.7537954	1434.08197186	1311.31251696	1555.14701907	1351.22312822
1792.63891426	1396.90499385	1731.91075696	1669.16238171	1521.80460372
1747.81929452	1245.73165685	1412.19588531	1325.39415497	1444.1193391
1339.87901284	1452.0936591	1237.22400613	1369.03539933	1656.52377725
1336.71244762	1405.07986171	1347.11415907	1286.25301305	1801.27338188
1405.68089557	1258.27400235	1403.67650738	1886.44428583	1227.64427079
1301.30119941	1666.43399234	1207.87316287	1336.1632604	1327.7261199
1508.33184814	1663.31075583	1359.77666185	1509.40532905	1225.05243547
1759.34405061	1251.7018554	1264.80417585	1280.44203047	1306.79671414
1434.79472061	1277.16870065	1466.97916163	1293.2176261	1329.78815323
1522.97033251	1340.92176317	1333.34373612	1886.43035693	1685.77567351
1330.52426664	1151.69413445	1392.08609275	1306.30724228	1310.55310213
1748.16272232	1619.55147659	1279.78658199	1583.04805687	1315.90298104

```
 1611.47715431 1282.20121863 1326.027709   1376.13725232 1635.48104763
 1564.08337818 1322.16315056 1308.36191485 1867.61493979 1285.60181573
 1611.85099697 1590.48008598 1257.72636953 1385.5362506  1331.33479306
 1291.24648573 1553.58364206 1534.14165858 1503.30410009 1290.9376268
 1364.34145479 1481.59313125 1291.87260287 1526.80528381 1508.1439951
 1585.37583444 1249.60421736 1380.34799167 1717.69247739 1264.20513121
 1465.0053516  1301.55437387 1534.15753871 1419.91119195 1293.23935245
 1562.89837332 1347.91887967 1236.30566905 1307.80687514 1366.37144897
 1373.35347981 1559.76419466 1322.08858101 1149.12124386 1385.79120035
 1338.3483722  1415.51643184 1549.1126367  1339.52254178 1237.82629479
 1582.70932325 1587.56890394 1721.77613724 1295.68586409 1391.23509481
 1742.75099888 1547.71943139 1221.29760294 1559.23390123 1290.61121356
 1337.24700852 1291.70309131 1390.59090909 1148.40531016 1341.90161322
 1398.09675603 1250.13837012 1641.17115666 1411.48789847 1376.337603
 1634.72249517 1355.35548185 1240.03367054 1346.24233453 1429.46493872
 1315.31704059 1326.3678542  1293.51848511 1502.48901322 1270.53595566
 1439.541187   1329.33016728 1451.30348623 1299.14637529 1147.37393937
 1550.2365856  1828.12042608 1257.92270216 1705.20488163 1384.99181691
 1295.52079387 1309.00943025 1296.58316289 1663.27783898 1436.39306119
 1247.80337747 1699.57524041 1452.26479374 1659.39387386 1639.98693335
 1211.72142142 1222.31827563 1283.25880484 1391.37038232 1861.46671587
 1231.37033378 1281.66721862 1648.28742997 1341.857217   1373.20000956
 1656.86175795 1788.96473117 1352.18643318 1511.93415134 1311.3715105
 1622.27331969 1320.12105943 1406.24762127 1661.26503945 1354.38988202
 1342.47237883 1235.00051984 1376.91170224 1336.80326171 1288.81145917
 1288.16843949 1254.96659406 1631.23167783 1591.98902816 1377.89058335
 1418.87734273 1163.94413931 1620.01209502 1346.61998646 1907.80191687
 1313.02927282 1222.96472577 1339.51746202 1821.08033387 1601.24734156
 1327.7588794  1270.96702179 1283.41498134 1653.11619369 1687.52754669
 1734.20885604 1780.59111262 1301.74684083 1296.6961092  1316.15165576
 1337.90929207 1287.0321436  1274.11160609 1728.87876084 1651.72926529
 1172.50716891 1894.26565442 1201.4708563  1830.93197818 1256.45089723
 1589.08210733 1163.08763982]

r2_score is:0.981472312450999
MAE: 14.06129297282004
MSE: 591.5987528758549
RMSE: 24.322803145933957
```

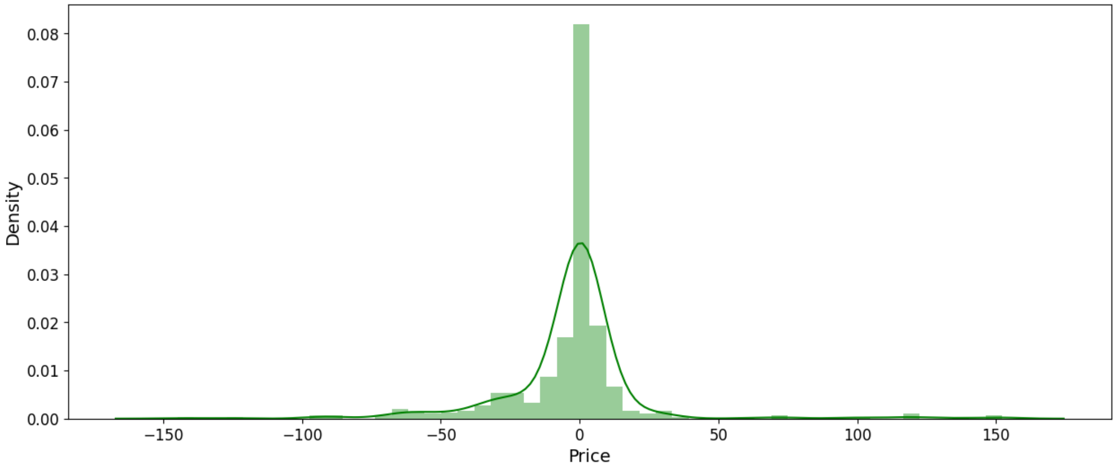


Using KNNs

[48]:
```
from sklearn.neighbors import KNeighborsRegressor

predict(KNeighborsRegressor())
```

```
Training Score: 0.9837444890953134
Predictions are: [1279.48 1374.9  1404.26 1724.9  1328.86 1323.04 1168.06
1345.26 1265.88
 1305.78 1714.38 1342.8  1361.6  1282.3  1570.3  1313.98 1333.76 1409.06
 1525.58 1391.14 1746.76 1335.34 1418.38 1263.28 1328.9  1535.54 1756.98
 1246.56 1592.02 1968.   1356.22 1330.64 1684.4  1319.74 1322.74 1362.52
 1587.44 1394.04 1650.4  1370.28 1221.7  1368.46 1424.7  1243.72 1346.72
 1287.26 1628.64 1367.04 1441.18 1334.74 1277.46 1600.12 1355.3  1540.3
 1445.76 1303.64 1272.5  1536.56 1265.18 1528.82 1372.6  1753.12 1757.42
 1595.9  1347.3  1630.96 1249.8  1670.56 1552.92 1494.9  1613.12 1454.44
 1590.62 1674.54 1566.66 1561.   1255.75 1259.1  1320.39 1255.08 1358.76
 1490.02 1347.24 1296.   1563.72 1591.84 1326.8  1443.58 1599.68 1575.54
 1330.8  1325.96 1413.24 1431.56 1607.38 1537.47 1788.48 1291.58 1308.74
 1272.38 1779.96 1252.67 1417.5  1222.62 1366.28 1289.78 1459.58 1495.3
 1576.26 1413.76 1736.14 1428.78 1387.48 1946.28 1678.92 1651.72 1305.
 1291.72 1770.02 1424.58 1322.96 1403.56 1405.02 1424.04 1174.26 1240.82
 1249.16 1522.2  1411.96 1619.36 1547.02 1403.22 1140.88 1430.52 1522.78
 1262.85 1306.44 1252.   1372.74 1399.62 1295.12 1322.76 1368.24 1266.46
 1942.17 1610.24 1338.34 1467.56 1739.94 1582.3  1346.12 1934.32 1737.96
 1406.66 1270.68 1238.98 1277.21 1408.68 1300.46 1300.26 1297.88 1276.04
 1421.36 1202.6  1355.32 1318.09 1736.18 1366.46 1256.   1285.58 1304.88
 1350.14 1430.7  1450.88 1727.04 1165.54 1360.6  1615.96 1614.04 1570.28
 1534.24 1329.78 1542.76 1510.74 1770.24 1241.78 1293.94 1606.98 1206.28
 1268.58 1416.62 1739.2  1374.44 1126.14 1273.32 1503.74 1365.86 1437.44
 1341.36 1724.6  1281.72 1281.34 1374.06 1338.8  1307.94 1350.02 1152.68
 1663.46 1346.62 1320.68 1419.   1334.14 1329.74 1281.94 1266.98 1773.6
```

```
 1640.86 1577.04 1329.06 1757.42 1385.52 1393.7  1298.82 1409.9  1271.9
 1593.66 1349.68 1612.58 1542.42 1248.36 1659.3  1279.98 1322.92 1357.28
 1436.08 1343.18 1579.84 1375.3  1501.62 1697.72 1276.1  1662.38 1725.56
 1469.28 1307.4  1324.8  1133.96 1659.02 1324.44 1838.04 1533.62 1359.24
 1287.55 1292.34 1271.6  1491.32 1285.84 1755.66 1330.04 1423.86 1317.01
 1570.5  1776.06 1288.48 1348.3  1700.5  1378.78 1753.02 1257.66 1276.26
 1372.1  1311.5  1391.74 1367.44 1321.95 1206.46 1651.5  1770.98 1591.48
 1414.54 1264.12 1436.08 1351.4  1551.22 1342.58 1785.12 1395.06 1748.08
 1665.44 1522.04 1766.48 1342.44 1393.64 1318.4  1422.3  1335.36 1450.6
 1297.84 1341.04 1639.02 1333.38 1411.54 1342.06 1278.8  1792.6  1389.27
 1246.34 1383.24 1860.2  1206.54 1321.28 1665.5  1187.9  1334.92 1327.7
 1511.14 1670.26 1357.16 1503.96 1206.54 1759.18 1277.9  1291.1  1275.3
 1299.96 1433.28 1248.52 1461.7  1313.93 1337.82 1517.04 1345.44 1330.82
 1850.42 1663.16 1321.09 1138.6  1382.32 1303.04 1301.22 1750.7  1615.66
 1271.98 1598.78 1314.44 1612.72 1268.5  1371.04 1381.42 1650.42 1580.18
 1318.86 1298.9  1850.   1272.7  1606.08 1581.42 1248.36 1381.64 1321.34
 1284.81 1542.94 1526.02 1499.26 1278.4  1371.78 1465.64 1279.16 1521.9
 1502.76 1578.3  1252.98 1366.02 1719.84 1259.9  1458.84 1295.56 1521.02
 1421.76 1282.1  1565.28 1342.8  1212.34 1296.54 1354.36 1378.74 1561.2
 1307.76 1121.34 1379.86 1332.68 1417.36 1550.46 1336.92 1312.16 1578.18
 1582.6  1710.98 1293.56 1380.2  1744.72 1535.96 1288.5  1565.28 1281.62
 1323.86 1322.96 1378.84 1122.12 1344.48 1394.88 1277.9  1626.54 1393.64
 1366.84 1635.88 1342.28 1247.56 1327.6  1423.   1311.64 1313.86 1276.88
 1497.88 1251.05 1448.42 1320.28 1450.6  1292.72 1136.86 1541.06 1827.72
 1277.68 1700.36 1376.56 1288.18 1286.68 1288.18 1662.86 1433.28 1277.83
 1712.74 1451.48 1681.38 1624.32 1186.37 1284.52 1304.5  1391.66 1876.36
 1285.36 1290.88 1655.46 1328.92 1372.24 1651.82 1792.96 1350.72 1504.58
 1399.84 1613.74 1305.32 1398.66 1676.7  1345.46 1335.48 1277.94 1361.96
 1333.42 1286.86 1310.96 1252.98 1639.72 1602.94 1375.06 1415.24 1150.14
 1609.12 1337.36 1946.28 1304.22 1207.09 1323.96 1831.26 1582.14 1350.16
 1286.6  1271.28 1651.78 1662.32 1751.32 1780.66 1275.   1292.24 1310.5
 1334.9  1254.44 1258.08 1726.8  1647.26 1137.88 1880.22 1206.28 1812.28
 1246.3  1575.68 1132.24]

r2_score is:0.9790869008282281
MAE: 12.115404339250498
MSE: 667.7661934911242
RMSE: 25.84117244807449
```

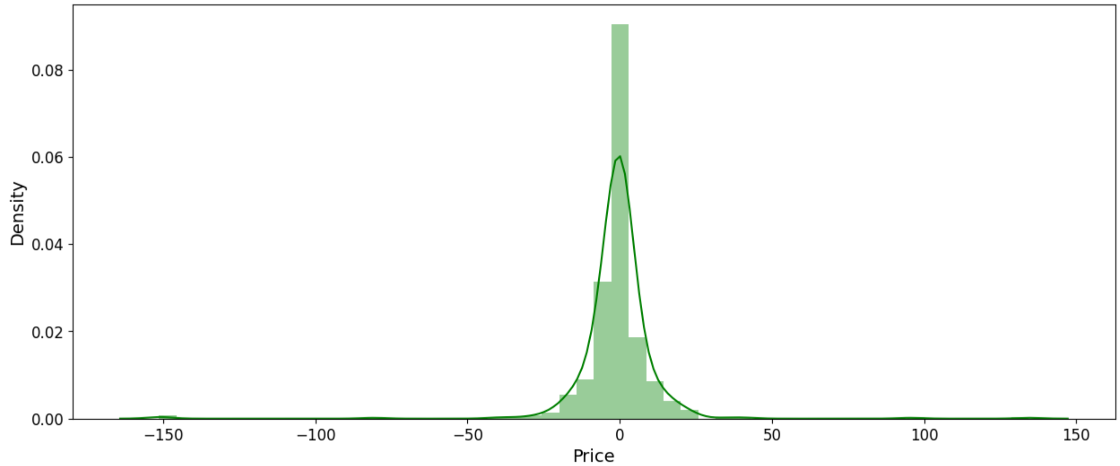


Using Decision Tree

[49]:
```python
from sklearn.tree import DecisionTreeRegressor

predict(DecisionTreeRegressor())
```

```
Training Score: 1.0
Predictions are: [1280.2  1374.5  1340.2  1714.1  1328.2  1293.8  1141.3  1345.6
1265.8
 1303.5  1708.8  1343.5  1363.8  1282.   1563.1  1313.7  1336.8  1408.
 1523.   1332.   1755.2  1336.2  1419.   1267.6  1328.5  1535.1  1750.4
 1203.5  1592.4  1946.5  1348.3  1328.2  1684.1  1439.   1322.5  1364.
 1564.5  1394.8  1652.2  1368.   1195.2  1373.6  1433.1  1242.4  1312.4
 1285.5  1625.7  1366.8  1434.1  1335.8  1276.4  1599.   1359.8  1545.4
 1447.8  1297.6  1272.4  1537.2  1264.4  1515.   1371.2  1762.   1768.6
 1592.5  1346.3  1638.9  1250.6  1683.3  1551.8  1499.1  1641.9  1456.
 1524.9  1683.3  1581.4  1561.9  1225.3  1221.7  1325.9  1257.3  1358.1
 1491.   1359.8  1366.6  1563.1  1586.2  1297.6  1446.3  1615.2  1562.9
 1268.   1324.2  1410.7  1429.4  1587.   1539.6  1794.1  1358.5  1280.1
 1271.7  1787.   1262.4  1416.2  1197.7  1368.   1287.6  1452.3  1493.2
 1574.8  1414.2  1730.6  1428.1  1385.5  1934.3  1671.4  1659.1  1306.4
 1291.9  1740.   1433.1  1323.1  1404.1  1406.   1424.   1178.7  1240.3
 1218.4  1532.4  1415.7  1619.7  1556.   1401.8  1145.4  1429.7  1517.3
 1214.1  1308.8  1198.   1373.2  1398.6  1294.6  1323.6  1373.6  1205.1
 1937.8  1609.6  1342.5  1317.9  1753.6  1564.5  1319.   1952.5  1744.7
 1404.1  1267.6  1234.5  1235.7  1415.7  1314.7  1300.6  1298.9  1275.7
 1432.7  1202.3  1356.2  1286.5  1739.   1368.5  1262.4  1285.5  1297.6
 1350.   1428.   1434.8  1731.2  1141.3  1356.6  1614.8  1615.2  1580.3
 1541.7  1329.9  1544.1  1509.7  1772.7  1239.   1295.8  1603.7  1183.4
 1240.3  1416.   1735.2  1386.8  1123.9  1275.9  1496.3  1373.6  1467.3
 1306.9  1725.9  1282.6  1281.   1373.4  1341.4  1440.6  1350.1  1151.9
```

```
 1663.7  1347.   1321.8  1418.   1336.2  1295.8  1282.05 1240.3  1790.9
 1650.3  1576.9  1267.2  1755.2  1385.5  1319.7  1299.8  1408.   1272.1
 1594.2  1314.5  1614.7  1542.1  1249.   1654.9  1280.9  1340.7  1373.6
 1436.2  1343.5  1587.05 1379.4  1500.1  1692.8  1273.4  1650.8  1723.5
 1465.5  1312.3  1241.7  1134.   1661.5  1296.6  1833.5  1574.5  1315.7
 1285.   1294.9  1268.7  1491.   1283.   1768.6  1449.   1428.1  1286.5
 1571.2  1762.1  1289.2  1280.6  1694.4  1314.6  1756.8  1281.3  1273.7
 1372.3  1307.4  1391.3  1303.5  1325.9  1205.1  1646.7  1787.   1594.2
 1414.   1231.7  1435.9  1472.   1543.3  1341.4  1790.3  1396.7  1742.7
 1677.3  1523.2  1773.4  1235.7  1392.6  1323.1  1416.   1337.4  1450.6
 1333.5  1337.7  1640.8  1335.8  1410.7  1341.2  1270.8  1816.1  1395.7
 1263.6  1392.8  1892.6  1205.1  1287.3  1664.   1185.8  1324.8  1327.6
 1501.   1664.   1357.4  1501.1  1207.9  1763.7  1228.2  1237.8  1274.7
 1298.8  1427.9  1260.4  1449.   1283.3  1310.6  1514.4  1324.8  1331.6
 1856.4  1669.6  1324.2  1141.3  1381.4  1302.8  1300.8  1768.4  1618.4
 1274.7  1599.6  1314.1  1624.   1268.   1308.3  1375.9  1648.7  1587.
 1314.6  1299.4  1856.4  1290.1  1604.   1589.1  1249.   1384.   1322.7
 1294.4  1545.   1528.1  1504.8  1276.8  1369.8  1450.6  1278.8  1515.3
 1508.8  1587.5  1231.7  1365.6  1720.9  1231.2  1454.9  1296.   1523.8
 1428.1  1280.9  1575.4  1341.6  1213.8  1296.7  1357.3  1378.6  1556.7
 1306.7  1120.6  1384.4  1333.7  1364.9  1543.3  1329.8  1218.8  1576.
 1590.6  1714.3  1293.4  1372.9  1742.7  1541.7  1314.4  1556.7  1282.05
 1323.1  1290.1  1378.7  1121.   1343.5  1395.95 1352.6  1625.7  1411.7
 1373.4  1659.1  1343.4  1387.8  1328.5  1424.   1301.4  1317.1  1276.4
 1493.4  1423.2  1434.8  1321.3  1450.6  1291.9  1137.1  1546.1  1826.
 1242.2  1702.6  1366.6  1288.2  1283.8  1288.2  1657.9  1438.3  1205.4
 1704.4  1448.8  1709.9  1639.9  1292.9  1291.   1267.2  1391.3  1848.9
 1214.1  1292.7  1654.8  1327.7  1373.7  1639.9  1776.4  1359.6  1494.
 1306.1  1617.9  1304.2  1399.2  1670.4  1345.6  1335.1  1376.4  1360.4
 1336.8  1285.8  1279.05 1358.8  1642.3  1604.6  1370.2  1416.   1151.9
 1612.9  1324.8  1952.5  1306.4  1204.4  1315.7  1823.5  1590.1  1324.7
 1236.1  1272.1  1659.1  1669.6  1740.2  1739.2  1273.4  1291.9  1308.8
 1333.1  1262.4  1255.1  1734.   1642.1  1141.3  1869.6  1183.4  1798.5
 1247.1  1562.9  1134.  ]


r2_score is:0.9931313724172174
MAE: 6.038165680473366
MSE: 219.3188708086784
RMSE: 14.809418314325463
```



Using XGBoost

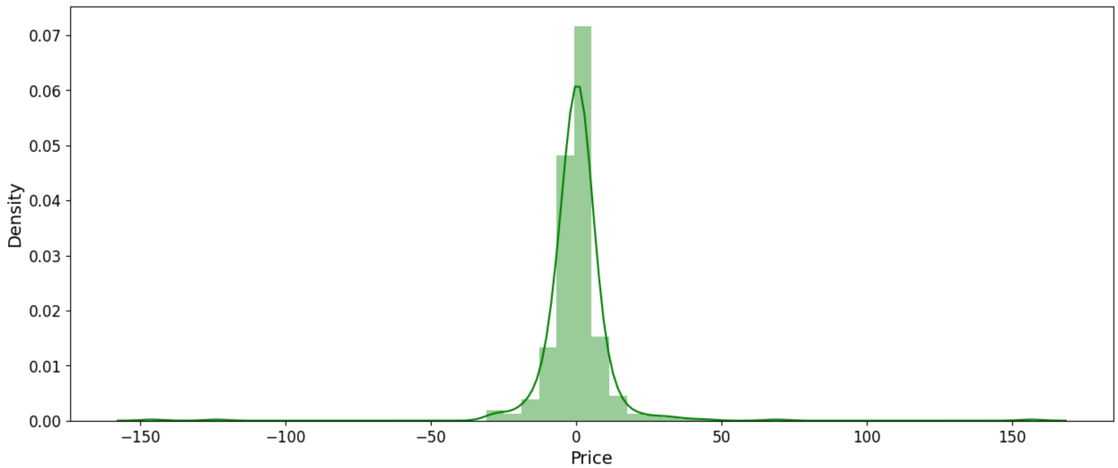
[51]:
```
from xgboost import XGBRegressor

predict(XGBRegressor())
```

```
Training Score: 0.9999229443115288
Predictions are: [1278.1401 1371.271  1355.9695 1723.461  1334.4335 1297.1885
1143.4845
 1345.4817 1261.3967 1301.5546 1712.7925 1346.2677 1361.0388 1280.0442
 1570.3239 1307.7097 1330.5502 1405.5698 1528.4703 1334.4691 1751.9741
 1339.1648 1416.658  1269.2885 1328.5906 1542.5172 1746.5859 1201.6
 1589.8477 1968.6177 1368.1272 1332.0349 1681.0333 1411.5604 1321.3562
 1368.2838 1587.745  1392.7683 1653.4155 1365.9961 1190.6746 1366.0255
 1425.8407 1248.3562 1309.78   1283.9656 1634.9989 1366.5922 1443.5984
 1332.3934 1309.3379 1592.4606 1358.125  1543.663  1444.7942 1301.2952
 1272.8468 1533.4299 1265.3164 1530.9901 1368.8342 1759.8955 1754.0508
 1556.21   1344.388  1647.2428 1247.8563 1674.9934 1552.3225 1503.549
 1619.5886 1457.717  1614.9015 1673.7311 1571.3754 1561.7765 1238.839
 1240.098  1323.3727 1261.1567 1357.4338 1501.3079 1351.5875 1360.682
 1565.3318 1599.4858 1296.2756 1444.8475 1604.5632 1580.2817 1298.9015
 1335.5979 1414.1458 1437.333  1595.7356 1532.4955 1787.0574 1361.242
 1253.5277 1262.0485 1787.2238 1234.8044 1415.709  1202.5835 1361.9067
 1287.6216 1462.7489 1496.2802 1575.358  1413.6388 1725.308  1430.6484
 1384.8676 1940.5999 1668.3163 1648.9375 1304.911  1284.184  1773.7411
 1426.2117 1333.1687 1403.8514 1402.8187 1422.5507 1168.6296 1239.81
 1214.3531 1526.9353 1412.6139 1612.4258 1554.7963 1406.0164 1139.9231
 1438.9379 1515.0935 1205.7859 1307.9661 1218.8116 1374.0381 1400.606
 1301.995  1335.3917 1371.3673 1211.531  1934.105  1612.1024 1353.5139
 1330.0481 1748.1047 1575.8652 1317.6875 1872.7856 1737.0591 1403.7838
 1273.0391 1234.7642 1240.0137 1414.432  1300.219  1297.8947 1299.8444
```

1278.68	1426.6141	1208.2771	1356.5972	1283.8226	1739.9668	1364.4336
1242.8846	1283.6166	1304.3206	1352.0343	1429.5084	1450.4612	1730.4215
1139.5007	1370.532	1611.441	1614.9751	1572.8901	1539.9576	1326.8142
1542.6788	1504.8601	1764.9355	1245.9175	1293.2754	1607.317	1183.8771
1246.0428	1416.9904	1738.8918	1390.2001	1122.9612	1274.1154	1494.618
1365.8163	1458.0974	1321.9692	1720.5029	1281.0718	1283.1833	1385.4495
1342.2816	1405.2263	1349.0986	1154.5426	1661.524	1346.1055	1317.141
1415.5275	1336.6482	1292.6216	1281.9204	1245.883	1775.68	1645.6095
1573.7771	1267.9812	1744.5032	1383.0267	1330.4617	1297.8392	1408.6117
1270.3572	1592.0887	1317.3297	1609.0696	1543.3334	1247.458	1654.3256
1283.0173	1318.4066	1357.1974	1433.836	1343.501	1592.4296	1372.963
1501.176	1702.2039	1278.3516	1663.5883	1721.4946	1465.6284	1311.8967
1242.5391	1135.5974	1662.448	1311.4135	1846.7109	1513.1067	1320.5234
1283.8212	1297.8049	1272.0736	1488.8386	1285.9332	1768.5712	1438.0448
1425.0048	1287.2444	1572.3243	1776.2441	1251.2731	1276.9556	1698.985
1325.8254	1748.8888	1269.4489	1271.9973	1372.6641	1311.3741	1392.1544
1302.9042	1320.5657	1207.6532	1639.0042	1772.4243	1579.0714	1415.5527
1209.155	1435.0444	1467.249	1554.1332	1339.2148	1795.1394	1397.1542
1747.916	1669.2935	1517.7278	1760.9276	1227.056	1393.5388	1317.3552
1426.8419	1334.7557	1452.0264	1332.1422	1344.8751	1647.8969	1328.2809
1413.23	1341.2748	1273.141	1824.6028	1394.3624	1242.9128	1382.8298
1873.4943	1204.1012	1293.1136	1666.6737	1183.718	1323.9946	1326.8599
1515.3857	1662.9457	1364.6366	1501.3649	1207.0576	1759.8098	1231.2573
1234.8068	1275.5709	1299.433	1431.3004	1257.6656	1467.031	1287.0248
1310.6682	1517.8877	1316.6273	1331.2999	1860.5161	1669.3477	1332.1057
1138.7745	1380.8285	1303.7114	1298.4424	1751.7356	1615.1791	1276.3864
1596.1725	1313.7517	1612.1031	1269.102	1314.9797	1377.4923	1643.8844
1579.2393	1319.8379	1293.5724	1860.1747	1274.3198	1608.4713	1587.1663
1244.879	1378.7319	1320.2559	1281.9232	1543.0208	1525.9465	1500.1459
1279.07	1372.5803	1466.9852	1276.0702	1515.3767	1506.4999	1577.6522
1224.3018	1363.6818	1713.1558	1242.7108	1458.133	1293.2662	1526.407
1420.9037	1277.5219	1559.8522	1345.1552	1212.3429	1302.998	1356.5808
1374.2977	1558.4381	1307.9718	1115.7528	1392.3804	1331.8647	1379.9117
1553.872	1336.014	1217.5753	1583.6182	1583.1661	1713.5565	1288.2429
1381.2958	1740.9135	1536.98	1306.6041	1562.9935	1285.7896	1316.4216
1263.8276	1378.0854	1123.1833	1342.8508	1395.6616	1356.9032	1618.904
1398.7333	1365.0759	1642.5986	1323.7134	1369.7854	1331.4812	1423.9309
1308.6188	1325.2765	1277.1478	1496.7229	1379.2478	1438.8896	1319.3661
1451.4161	1287.5803	1135.426	1543.6741	1842.3591	1240.8102	1699.3098
1372.7012	1284.4009	1285.1558	1274.2968	1665.6487	1434.2847	1208.611
1690.3779	1450.391	1679.8824	1632.9539	1302.7577	1294.6521	1271.7325
1394.2062	1841.2067	1212.8605	1290.1649	1655.6014	1328.808	1369.7107
1649.6575	1785.3904	1350.008	1504.1289	1306.9027	1618.1052	1301.9467
1400.7657	1671.5106	1346.824	1334.9899	1371.2576	1364.4167	1326.105
1282.6528	1284.2422	1360.5416	1635.4604	1594.3336	1373.4031	1414.8662
1149.1198	1612.3893	1316.2723	1856.4001	1306.308	1209.5226	1317.1808
1838.4152	1573.8733	1316.6603	1244.446	1277.0359	1658.0624	1673.3661
1742.9031	1751.9768	1277.1068	1289.5853	1312.2646	1331.2972	1253.6708

```
 1258.6484 1724.9952 1641.7196 1133.873  1836.3104 1182.3663 1812.1621
 1243.878  1592.7012 1136.4862]
```

```
r2_score is:0.9940499332197056
MAE: 5.967990948979906
MSE: 189.98874400491664
RMSE: 13.783640448187722
```



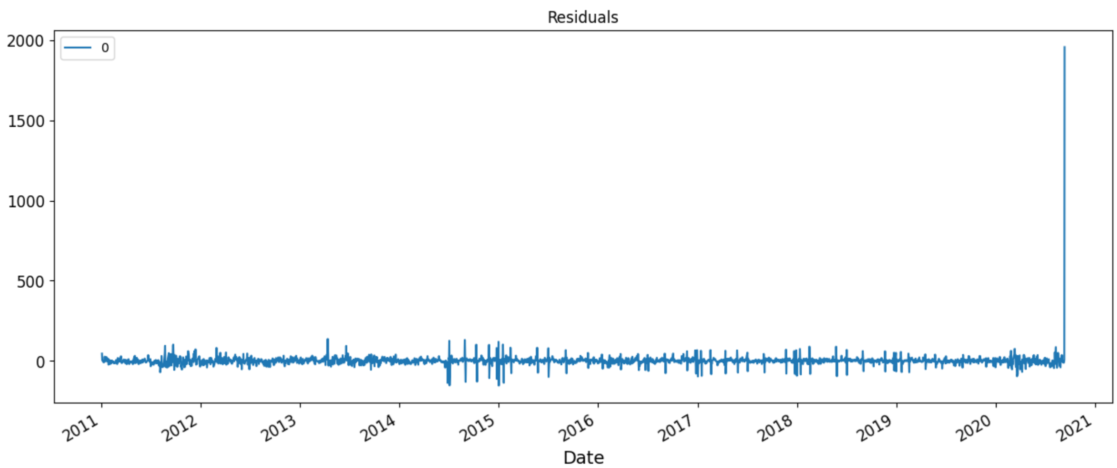
Building the ARIMA model

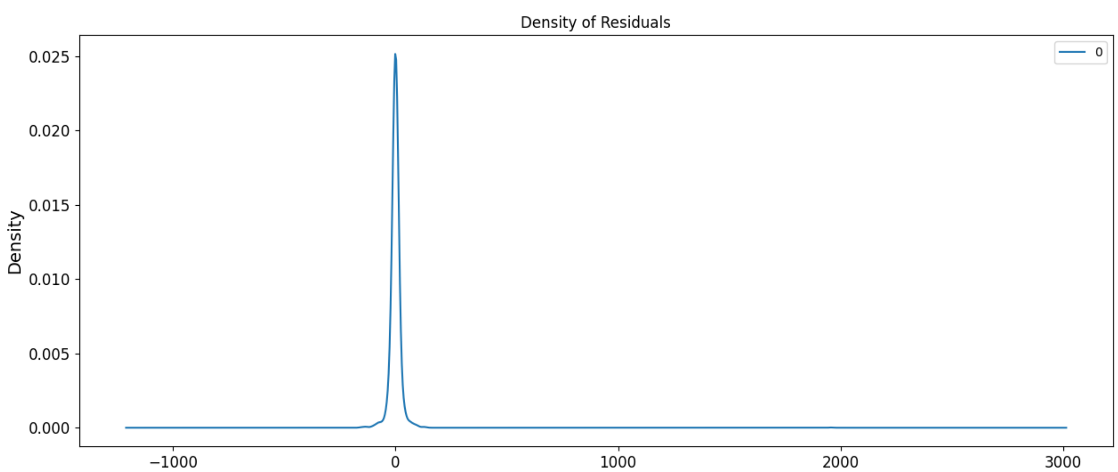
[53]:
```python
import statsmodels.api as sm
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
from pandas import DataFrame

# fit model
model = ARIMA(stationary_check_gold_price, order=(5, 1, 0))
model_fit = model.fit()

# print model summary
print(model_fit.summary())

# plot residual errors
residuals = DataFrame(model_fit.resid)
residuals.plot(title="Residuals")
plt.show()

residuals.plot(kind='kde', title="Density of Residuals")
plt.show()
```

```python
print(residuals.describe())
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)

                               SARIMAX Results
==============================================================================
Dep. Variable:                  Price   No. Observations:                 2531
Model:                 ARIMA(5, 1, 0)   Log Likelihood              -11341.408
Date:                Sun, 01 Sep 2024   AIC                          22694.815
Time:                        07:09:32   BIC                          22729.831
Sample:                             0   HQIC                         22707.520
                               - 2531
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.1807      0.010    -17.657      0.000      -0.201      -0.161
ar.L2         -0.1713      0.012    -13.747      0.000      -0.196      -0.147
ar.L3         -0.0402      0.022     -1.789      0.074      -0.084       0.004
ar.L4         -0.0975      0.018     -5.410      0.000      -0.133      -0.062
ar.L5          0.0290      0.016      1.764      0.078      -0.003       0.061
sigma2       459.1043      5.684     80.771      0.000     447.964     470.245
================================================================================
===
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
```

```
11703.18
Prob(Q):                              0.97   Prob(JB):
0.00
Heteroskedasticity (H):               0.96   Skew:
-0.38
Prob(H) (two-sided):                  0.53   Kurtosis:
13.51
================================================================================
===

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-
step).
```





0

```
count   2531.000000
mean       0.455939
std       44.412299
min     -154.536356
25%       -8.246394
50%       -0.037432
75%        7.817965
max     1957.350000
```

[54]:
```python
stationary_check_gold_price.values
```

[54]:
```
array([1957.35, 1964.3 , 1954.9 , …, 1373.4 , 1378.5 , 1422.6 ])
```

[57]:
```python
from sklearn.metrics import mean_squared_error
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt

X = stationary_check_gold_price.values
size = int(len(X) * 0.7)
train, test = X[0:size], X[size:len(X)]
history = [x for x in train]
predictions = list()

for t in range(len(test)):
    model = ARIMA(history, order=(5, 1, 0))
    model_fit = model.fit()
    output = model_fit.forecast()
    y_pred = output[0]
    predictions.append(y_pred)
    obs = test[t]
    history.append(obs)
    print('predicted=%f, expected=%f' % (y_pred, obs))

error = mean_squared_error(test, predictions)
print('Test MSE: %.3f' % error)

# plot the forecast
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

```
predicted=1212.490496, expected=1201.900000
predicted=1208.876591, expected=1203.100000
predicted=1207.223054, expected=1216.100000
predicted=1215.352889, expected=1214.100000
predicted=1211.395029, expected=1205.100000
predicted=1206.496677, expected=1198.400000
predicted=1200.701240, expected=1205.100000
```

```
predicted=1206.384925, expected=1195.000000
predicted=1197.636004, expected=1236.100000
predicted=1228.556401, expected=1231.200000
predicted=1221.990986, expected=1245.500000
predicted=1241.119519, expected=1235.700000
predicted=1229.127670, expected=1226.000000
predicted=1230.999538, expected=1258.500000
predicted=1251.559874, expected=1262.400000
predicted=1255.801621, expected=1235.300000
predicted=1239.465422, expected=1230.300000
predicted=1233.284087, expected=1233.200000
predicted=1236.111901, expected=1248.200000
predicted=1248.001865, expected=1221.700000
predicted=1224.512974, expected=1222.300000
predicted=1226.976684, expected=1250.600000
predicted=1243.646926, expected=1243.800000
predicted=1242.298600, expected=1237.800000
predicted=1237.989712, expected=1241.400000
predicted=1238.697681, expected=1241.100000
predicted=1242.248677, expected=1244.000000
predicted=1243.741719, expected=1243.500000
predicted=1242.317584, expected=1257.900000
predicted=1254.285040, expected=1273.400000
predicted=1265.566986, expected=1272.200000
predicted=1267.508600, expected=1287.300000
predicted=1280.524604, expected=1286.200000
predicted=1280.950290, expected=1268.300000
predicted=1272.197449, expected=1271.100000
predicted=1272.868244, expected=1281.000000
predicted=1279.793298, expected=1284.500000
predicted=1283.355483, expected=1308.400000
predicted=1300.012697, expected=1317.700000
predicted=1307.813310, expected=1308.000000
predicted=1305.751749, expected=1314.600000
predicted=1311.362195, expected=1313.100000
predicted=1311.809441, expected=1323.600000
predicted=1322.314453, expected=1349.000000
predicted=1339.117371, expected=1345.200000
predicted=1339.259974, expected=1352.000000
predicted=1347.649432, expected=1352.400000
predicted=1347.642565, expected=1350.200000
predicted=1351.086173, expected=1333.900000
predicted=1337.445050, expected=1342.500000
predicted=1344.680266, expected=1315.700000
predicted=1321.892166, expected=1314.400000
predicted=1322.845811, expected=1322.700000
predicted=1321.733081, expected=1282.000000
predicted=1293.973745, expected=1273.000000
```

```
predicted=1284.267713, expected=1276.400000
predicted=1279.971789, expected=1268.000000
predicted=1275.672608, expected=1296.600000
predicted=1291.767024, expected=1306.900000
predicted=1297.338738, expected=1324.200000
predicted=1316.280110, expected=1324.800000
predicted=1315.465010, expected=1309.700000
predicted=1310.976101, expected=1317.400000
predicted=1317.000737, expected=1320.600000
predicted=1319.403684, expected=1286.000000
predicted=1295.177174, expected=1326.500000
predicted=1323.429035, expected=1338.400000
predicted=1327.971316, expected=1323.600000
predicted=1325.846596, expected=1335.900000
predicted=1329.324960, expected=1316.000000
predicted=1318.181575, expected=1326.900000
predicted=1330.407240, expected=1332.500000
predicted=1328.048501, expected=1369.400000
predicted=1360.998462, expected=1307.800000
predicted=1311.703558, expected=1309.500000
predicted=1320.686977, expected=1317.900000
predicted=1315.530993, expected=1308.400000
predicted=1317.690771, expected=1330.400000
predicted=1325.236559, expected=1363.900000
predicted=1349.761703, expected=1364.100000
predicted=1355.388307, expected=1386.800000
predicted=1375.189374, expected=1386.700000
predicted=1376.852270, expected=1373.100000
predicted=1375.148308, expected=1389.900000
predicted=1385.960842, expected=1412.000000
predicted=1403.906084, expected=1393.700000
predicted=1393.301236, expected=1396.100000
predicted=1395.692672, expected=1412.900000
predicted=1406.863015, expected=1419.000000
predicted=1416.044661, expected=1420.600000
predicted=1416.723345, expected=1393.000000
predicted=1396.664202, expected=1395.700000
predicted=1401.152782, expected=1371.200000
predicted=1378.688299, expected=1370.600000
predicted=1380.359337, expected=1373.100000
predicted=1373.764547, expected=1366.200000
predicted=1370.737073, expected=1371.700000
predicted=1371.503275, expected=1361.600000
predicted=1362.939913, expected=1334.000000
predicted=1343.816865, expected=1321.200000
predicted=1331.019935, expected=1334.700000
predicted=1338.188877, expected=1312.900000
predicted=1319.545536, expected=1310.700000
```

```
predicted=1316.755960, expected=1286.100000
predicted=1292.303634, expected=1283.200000
predicted=1293.259810, expected=1302.600000
predicted=1300.364083, expected=1310.600000
predicted=1307.439691, expected=1311.000000
predicted=1307.370602, expected=1312.400000
predicted=1308.649144, expected=1324.000000
predicted=1320.069568, expected=1328.400000
predicted=1324.490923, expected=1321.700000
predicted=1321.180900, expected=1329.000000
predicted=1326.961920, expected=1319.700000
predicted=1320.342402, expected=1335.100000
predicted=1334.012832, expected=1336.400000
predicted=1331.951243, expected=1293.300000
predicted=1303.728392, expected=1284.600000
predicted=1294.858079, expected=1277.900000
predicted=1285.110381, expected=1290.800000
predicted=1295.823929, expected=1283.800000
predicted=1283.230637, expected=1277.800000
predicted=1280.689837, expected=1280.100000
predicted=1279.670433, expected=1247.400000
predicted=1256.470125, expected=1245.900000
predicted=1254.683202, expected=1234.900000
predicted=1240.019574, expected=1212.900000
predicted=1225.419197, expected=1247.050000
predicted=1244.449711, expected=1252.100000
predicted=1245.748906, expected=1243.600000
predicted=1244.590348, expected=1255.900000
predicted=1249.534819, expected=1223.800000
predicted=1229.287283, expected=1211.400000
predicted=1222.492339, expected=1229.600000
predicted=1228.801733, expected=1274.800000
predicted=1265.005586, expected=1276.800000
predicted=1265.001043, expected=1291.600000
predicted=1281.327911, expected=1285.900000
predicted=1277.692011, expected=1373.600000
predicted=1353.444063, expected=1366.600000
predicted=1345.935872, expected=1382.800000
predicted=1375.257701, expected=1387.300000
predicted=1371.489650, expected=1377.600000
predicted=1380.668222, expected=1391.800000
predicted=1388.118325, expected=1377.000000
predicted=1377.719765, expected=1386.200000
predicted=1387.810195, expected=1383.000000
predicted=1380.606669, expected=1415.700000
predicted=1410.224059, expected=1398.400000
predicted=1393.602273, expected=1397.100000
predicted=1399.725992, expected=1411.700000
```

```
predicted=1405.498219, expected=1392.600000
predicted=1396.864706, expected=1411.500000
predicted=1410.126656, expected=1391.300000
predicted=1391.203585, expected=1379.100000
predicted=1388.199005, expected=1395.950000
predicted=1393.291183, expected=1386.800000
predicted=1389.002977, expected=1392.000000
predicted=1392.744858, expected=1367.600000
predicted=1370.417663, expected=1377.800000
predicted=1382.271451, expected=1384.300000
predicted=1381.205172, expected=1364.900000
predicted=1370.574653, expected=1387.100000
predicted=1383.882328, expected=1396.500000
predicted=1389.868159, expected=1424.700000
predicted=1416.827979, expected=1434.500000
predicted=1421.592138, expected=1436.800000
predicted=1431.381833, expected=1468.800000
predicted=1456.670410, expected=1473.900000
predicted=1464.724507, expected=1449.000000
predicted=1451.527487, expected=1468.100000
predicted=1464.983464, expected=1464.300000
predicted=1462.729145, expected=1467.700000
predicted=1469.751533, expected=1446.300000
predicted=1447.781021, expected=1472.200000
predicted=1471.712107, expected=1467.400000
predicted=1463.515811, expected=1453.600000
predicted=1459.003732, expected=1461.800000
predicted=1459.529043, expected=1423.400000
predicted=1432.946268, expected=1408.600000
predicted=1422.011700, expected=1421.000000
predicted=1422.644590, expected=1395.300000
predicted=1404.600621, expected=1392.000000
predicted=1398.637791, expected=1382.200000
predicted=1385.000204, expected=1386.800000
predicted=1391.725752, expected=1360.600000
predicted=1366.083680, expected=1501.000000
predicted=1473.887693, expected=1564.300000
predicted=1518.598550, expected=1558.300000
predicted=1539.817501, expected=1586.200000
predicted=1559.620943, expected=1572.000000
predicted=1565.563561, expected=1575.400000
predicted=1578.716341, expected=1551.800000
predicted=1553.593257, expected=1552.800000
predicted=1559.952823, expected=1575.100000
predicted=1570.447956, expected=1600.000000
predicted=1592.524073, expected=1594.800000
predicted=1588.537544, expected=1606.200000
predicted=1600.709559, expected=1595.800000
```

```
predicted=1593.511944, expected=1604.600000
predicted=1605.534382, expected=1606.200000
predicted=1603.017392, expected=1613.800000
predicted=1612.824090, expected=1607.500000
predicted=1605.831416, expected=1611.300000
predicted=1611.423085, expected=1604.600000
predicted=1604.753555, expected=1592.500000
predicted=1597.381831, expected=1590.600000
predicted=1593.389443, expected=1588.300000
predicted=1590.816411, expected=1591.500000
predicted=1592.676658, expected=1577.800000
predicted=1580.192628, expected=1576.600000
predicted=1579.866229, expected=1574.800000
predicted=1575.787581, expected=1574.600000
predicted=1576.835218, expected=1574.600000
predicted=1574.524929, expected=1572.100000
predicted=1572.851347, expected=1571.900000
predicted=1572.459397, expected=1577.700000
predicted=1576.590825, expected=1595.200000
predicted=1590.376511, expected=1615.200000
predicted=1606.631102, expected=1586.200000
predicted=1586.625604, expected=1572.400000
predicted=1578.628035, expected=1578.200000
predicted=1579.603014, expected=1577.600000
predicted=1581.299947, expected=1603.600000
predicted=1598.527458, expected=1608.800000
predicted=1600.992193, expected=1634.700000
predicted=1626.588647, expected=1644.200000
predicted=1633.083044, expected=1648.700000
predicted=1644.282725, expected=1648.200000
predicted=1643.833160, expected=1666.000000
predicted=1661.501070, expected=1670.400000
predicted=1665.377311, expected=1677.700000
predicted=1674.355147, expected=1672.400000
predicted=1669.623171, expected=1675.300000
predicted=1675.354052, expected=1669.400000
predicted=1669.620654, expected=1660.600000
predicted=1664.465503, expected=1679.900000
predicted=1677.368835, expected=1660.700000
predicted=1662.080058, expected=1652.400000
predicted=1658.151561, expected=1656.400000
predicted=1655.804209, expected=1669.500000
predicted=1669.041400, expected=1686.300000
predicted=1680.047901, expected=1692.800000
predicted=1686.353876, expected=1686.600000
predicted=1684.183640, expected=1690.400000
predicted=1688.879907, expected=1682.700000
predicted=1683.605697, expected=1683.400000
```

```
predicted=1685.598317, expected=1668.900000
predicted=1671.738872, expected=1660.000000
predicted=1666.029765, expected=1677.300000
predicted=1675.909166, expected=1654.800000
predicted=1658.301625, expected=1661.500000
predicted=1664.534563, expected=1645.500000
predicted=1646.523836, expected=1648.100000
predicted=1653.728560, expected=1673.700000
predicted=1666.965079, expected=1687.900000
predicted=1681.287796, expected=1674.800000
predicted=1672.472799, expected=1654.900000
predicted=1658.270950, expected=1662.600000
predicted=1664.903874, expected=1659.800000
predicted=1661.848989, expected=1658.600000
predicted=1661.053046, expected=1659.100000
predicted=1657.930100, expected=1644.900000
predicted=1648.528841, expected=1666.500000
predicted=1664.820724, expected=1669.500000
predicted=1664.928644, expected=1697.000000
predicted=1690.823658, expected=1695.800000
predicted=1687.055575, expected=1695.600000
predicted=1694.603119, expected=1716.600000
predicted=1708.910660, expected=1708.200000
predicted=1706.520389, expected=1713.000000
predicted=1712.562267, expected=1704.000000
predicted=1702.908428, expected=1700.300000
predicted=1704.360775, expected=1692.400000
predicted=1694.603243, expected=1694.400000
predicted=1697.052257, expected=1719.600000
predicted=1714.275409, expected=1710.900000
predicted=1708.184090, expected=1727.200000
predicted=1723.607377, expected=1716.500000
predicted=1712.920156, expected=1742.200000
predicted=1739.662388, expected=1749.500000
predicted=1740.869637, expected=1751.300000
predicted=1749.680043, expected=1727.900000
predicted=1728.894233, expected=1723.200000
predicted=1728.940797, expected=1734.000000
predicted=1733.932389, expected=1714.300000
predicted=1719.433171, expected=1713.300000
predicted=1716.972751, expected=1729.500000
predicted=1725.836166, expected=1724.200000
predicted=1724.633783, expected=1730.300000
predicted=1728.743563, expected=1730.300000
predicted=1727.351192, expected=1725.400000
predicted=1727.224106, expected=1713.200000
predicted=1716.041820, expected=1714.100000
predicted=1716.927431, expected=1682.200000
```

```
predicted=1690.267388, expected=1674.100000
predicted=1683.860960, expected=1714.100000
predicted=1708.368869, expected=1717.500000
predicted=1712.550099, expected=1710.500000
predicted=1709.169328, expected=1707.700000
predicted=1704.616743, expected=1710.900000
predicted=1711.882015, expected=1712.000000
predicted=1712.162477, expected=1700.500000
predicted=1702.748019, expected=1708.300000
predicted=1708.512857, expected=1725.100000
predicted=1720.359965, expected=1722.800000
predicted=1720.697976, expected=1743.300000
predicted=1737.120994, expected=1751.500000
predicted=1743.706314, expected=1744.700000
predicted=1744.053283, expected=1736.000000
predicted=1736.388754, expected=1758.000000
predicted=1754.995761, expected=1768.800000
predicted=1763.284898, expected=1763.200000
predicted=1761.760827, expected=1763.000000
predicted=1760.792535, expected=1773.500000
predicted=1770.889908, expected=1778.600000
predicted=1776.231750, expected=1794.100000
predicted=1788.927986, expected=1777.300000
predicted=1776.159811, expected=1772.700000
predicted=1776.107007, expected=1780.500000
predicted=1779.009959, expected=1771.100000
predicted=1774.170662, expected=1777.600000
predicted=1777.816056, expected=1750.600000
predicted=1754.590698, expected=1763.800000
predicted=1767.602429, expected=1762.100000
predicted=1760.131822, expected=1775.500000
predicted=1775.556577, expected=1767.700000
predicted=1764.354643, expected=1769.000000
predicted=1770.190521, expected=1768.400000
predicted=1767.067257, expected=1767.700000
predicted=1769.200959, expected=1769.800000
predicted=1769.149436, expected=1769.100000
predicted=1768.955223, expected=1730.600000
predicted=1739.134240, expected=1731.800000
predicted=1739.452697, expected=1728.700000
predicted=1731.400995, expected=1737.500000
predicted=1740.667369, expected=1702.600000
predicted=1707.331395, expected=1690.800000
predicted=1700.641751, expected=1692.900000
predicted=1695.739079, expected=1684.600000
predicted=1690.910383, expected=1653.500000
predicted=1662.227024, expected=1659.800000
predicted=1664.814185, expected=1666.500000
```

```
predicted=1666.403185, expected=1672.400000
predicted=1672.681217, expected=1669.800000
predicted=1667.131826, expected=1669.600000
predicted=1669.289893, expected=1637.400000
predicted=1644.113075, expected=1639.900000
predicted=1646.594138, expected=1620.100000
predicted=1625.572534, expected=1616.300000
predicted=1624.800486, expected=1616.100000
predicted=1616.775608, expected=1603.700000
predicted=1608.961619, expected=1599.400000
predicted=1602.772736, expected=1609.600000
predicted=1608.853909, expected=1619.700000
predicted=1617.014952, expected=1617.100000
predicted=1615.137354, expected=1612.900000
predicted=1612.528499, expected=1609.700000
predicted=1610.560746, expected=1612.900000
predicted=1613.690908, expected=1606.000000
predicted=1607.400816, expected=1587.400000
predicted=1592.940144, expected=1603.700000
predicted=1603.955629, expected=1610.500000
predicted=1607.491032, expected=1619.700000
predicted=1617.326213, expected=1617.900000
predicted=1613.605956, expected=1615.000000
predicted=1615.217479, expected=1608.000000
predicted=1609.371113, expected=1576.000000
predicted=1585.018512, expected=1577.100000
predicted=1584.186433, expected=1582.500000
predicted=1583.528883, expected=1580.100000
predicted=1582.881302, expected=1570.400000
predicted=1571.652107, expected=1589.100000
predicted=1586.619092, expected=1591.200000
predicted=1587.791561, expected=1591.600000
predicted=1591.112353, expected=1564.900000
predicted=1568.083200, expected=1575.200000
predicted=1578.811115, expected=1579.300000
predicted=1577.715000, expected=1588.600000
predicted=1588.213584, expected=1578.400000
predicted=1576.520500, expected=1608.900000
predicted=1603.721894, expected=1621.300000
predicted=1611.877698, expected=1597.200000
predicted=1599.607033, expected=1603.500000
predicted=1602.763876, expected=1549.700000
predicted=1560.819054, expected=1577.500000
predicted=1585.394672, expected=1574.000000
predicted=1570.547992, expected=1587.500000
predicted=1590.042202, expected=1566.000000
predicted=1563.469612, expected=1564.500000
predicted=1569.686206, expected=1614.800000
```

```
predicted=1603.676304, expected=1622.200000
predicted=1613.034943, expected=1625.700000
predicted=1620.143103, expected=1627.000000
predicted=1619.866037, expected=1618.400000
predicted=1620.436465, expected=1618.100000
predicted=1619.692925, expected=1612.700000
predicted=1614.368648, expected=1595.500000
predicted=1601.383807, expected=1590.100000
predicted=1594.913547, expected=1586.600000
predicted=1590.023516, expected=1632.800000
predicted=1625.568406, expected=1615.200000
predicted=1609.777382, expected=1612.200000
predicted=1614.150523, expected=1620.500000
predicted=1614.970102, expected=1562.600000
predicted=1577.038913, expected=1563.400000
predicted=1574.614506, expected=1548.600000
predicted=1553.798182, expected=1568.800000
predicted=1574.288643, expected=1557.300000
predicted=1554.742926, expected=1548.100000
predicted=1553.051792, expected=1576.300000
predicted=1569.950225, expected=1588.400000
predicted=1582.356893, expected=1591.600000
predicted=1587.568849, expected=1574.500000
predicted=1573.412764, expected=1536.200000
predicted=1547.287533, expected=1556.800000
predicted=1561.175791, expected=1560.600000
predicted=1559.737067, expected=1583.600000
predicted=1580.553097, expected=1595.100000
predicted=1584.157487, expected=1593.700000
predicted=1590.475071, expected=1604.000000
predicted=1598.859370, expected=1638.600000
predicted=1628.443139, expected=1644.700000
predicted=1636.230329, expected=1634.200000
predicted=1632.079005, expected=1653.400000
predicted=1647.453469, expected=1661.700000
predicted=1656.915396, expected=1663.400000
predicted=1661.665545, expected=1664.000000
predicted=1660.584003, expected=1659.600000
predicted=1659.981838, expected=1641.400000
predicted=1646.242604, expected=1643.000000
predicted=1646.598801, expected=1631.900000
predicted=1635.466076, expected=1642.100000
predicted=1644.010896, expected=1640.600000
predicted=1638.715696, expected=1638.800000
predicted=1640.242037, expected=1650.300000
predicted=1646.783877, expected=1648.700000
predicted=1647.272003, expected=1659.100000
predicted=1656.721192, expected=1679.500000
```

```
predicted=1671.724771, expected=1659.000000
predicted=1659.214466, expected=1659.500000
predicted=1661.245771, expected=1642.500000
predicted=1645.172578, expected=1628.500000
predicted=1637.893346, expected=1612.300000
predicted=1618.872921, expected=1670.000000
predicted=1663.585737, expected=1677.500000
predicted=1666.101607, expected=1669.300000
predicted=1667.772490, expected=1652.200000
predicted=1650.086448, expected=1657.900000
predicted=1661.417220, expected=1684.800000
predicted=1679.962683, expected=1685.500000
predicted=1681.210781, expected=1662.300000
predicted=1664.467010, expected=1642.300000
predicted=1648.442349, expected=1650.000000
predicted=1654.513901, expected=1646.700000
predicted=1649.614634, expected=1666.900000
predicted=1664.380154, expected=1655.500000
predicted=1652.515853, expected=1659.100000
predicted=1660.111628, expected=1642.500000
predicted=1643.598802, expected=1693.700000
predicted=1687.719741, expected=1699.200000
predicted=1687.729959, expected=1710.900000
predicted=1706.255304, expected=1698.100000
predicted=1691.889760, expected=1683.300000
predicted=1689.305004, expected=1671.400000
predicted=1676.569356, expected=1703.000000
predicted=1701.291051, expected=1708.800000
predicted=1703.099697, expected=1721.100000
predicted=1716.307581, expected=1709.900000
predicted=1705.522762, expected=1787.000000
predicted=1772.155733, expected=1773.600000
predicted=1760.306356, expected=1775.100000
predicted=1774.532825, expected=1784.900000
predicted=1774.435010, expected=1770.000000
predicted=1774.839320, expected=1757.100000
predicted=1761.785125, expected=1724.500000
predicted=1733.929989, expected=1726.800000
predicted=1735.538619, expected=1726.300000
predicted=1728.843708, expected=1715.900000
predicted=1721.326075, expected=1723.000000
predicted=1722.422189, expected=1723.300000
predicted=1722.527484, expected=1739.000000
predicted=1736.295926, expected=1729.300000
predicted=1727.138127, expected=1746.400000
predicted=1743.933413, expected=1722.800000
predicted=1723.291861, expected=1737.900000
predicted=1739.879998, expected=1756.800000
```

```
predicted=1748.917430, expected=1747.100000
predicted=1747.601023, expected=1737.800000
predicted=1738.334615, expected=1731.000000
predicted=1733.236226, expected=1731.800000
predicted=1735.128738, expected=1726.300000
predicted=1728.469135, expected=1699.800000
predicted=1707.032594, expected=1664.200000
predicted=1677.134071, expected=1678.000000
predicted=1684.308036, expected=1663.700000
predicted=1668.742659, expected=1654.100000
predicted=1661.300985, expected=1659.500000
predicted=1658.521910, expected=1655.200000
predicted=1657.530991, expected=1630.400000
predicted=1636.861614, expected=1647.300000
predicted=1648.006345, expected=1639.200000
predicted=1639.566744, expected=1631.000000
predicted=1635.977510, expected=1607.500000
predicted=1612.053267, expected=1616.100000
predicted=1620.751107, expected=1619.400000
predicted=1618.935003, expected=1611.900000
predicted=1614.644029, expected=1599.700000
predicted=1602.000015, expected=1565.800000
predicted=1575.754350, expected=1539.900000
predicted=1553.643847, expected=1562.900000
predicted=1566.100351, expected=1594.200000
predicted=1587.744881, expected=1604.700000
predicted=1596.773938, expected=1608.900000
predicted=1600.923616, expected=1611.900000
predicted=1607.124256, expected=1615.600000
predicted=1613.769932, expected=1594.400000
predicted=1597.813194, expected=1595.600000
predicted=1599.133147, expected=1574.600000
predicted=1579.661473, expected=1584.300000
predicted=1588.737180, expected=1659.900000
predicted=1642.506721, expected=1664.200000
predicted=1650.063877, expected=1712.800000
predicted=1696.027865, expected=1709.800000
predicted=1692.742569, expected=1740.900000
predicted=1734.284634, expected=1727.900000
predicted=1719.788984, expected=1730.700000
predicted=1732.688332, expected=1747.000000
predicted=1740.418362, expected=1735.300000
predicted=1736.763335, expected=1745.500000
predicted=1744.084709, expected=1713.400000
predicted=1717.083515, expected=1710.800000
predicted=1718.636121, expected=1685.500000
predicted=1691.481224, expected=1695.700000
predicted=1702.194563, expected=1702.200000
```

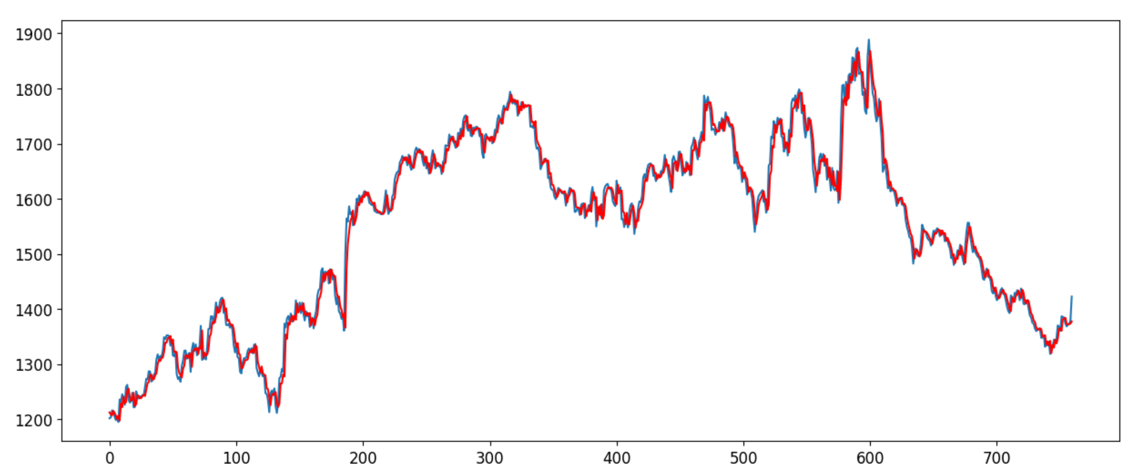
```
predicted=1699.575849, expected=1678.300000
predicted=1684.051420, expected=1724.700000
predicted=1717.412482, expected=1719.800000
predicted=1712.894539, expected=1773.800000
predicted=1763.787447, expected=1781.700000
predicted=1764.620101, expected=1777.800000
predicted=1776.296188, expected=1787.500000
predicted=1780.030222, expected=1758.900000
predicted=1764.049430, expected=1790.900000
predicted=1789.763592, expected=1798.400000
predicted=1791.220753, expected=1790.300000
predicted=1792.269784, expected=1755.300000
predicted=1759.373815, expected=1764.200000
predicted=1769.547190, expected=1728.700000
predicted=1737.288072, expected=1711.000000
predicted=1724.142679, expected=1724.200000
predicted=1724.534621, expected=1746.200000
predicted=1744.118925, expected=1746.700000
predicted=1742.613963, expected=1722.700000
predicted=1724.482278, expected=1699.600000
predicted=1706.885763, expected=1651.500000
predicted=1667.442024, expected=1635.100000
predicted=1650.949074, expected=1611.900000
predicted=1623.570137, expected=1646.000000
predicted=1648.510716, expected=1651.700000
predicted=1645.823232, expected=1675.500000
predicted=1669.886127, expected=1681.800000
predicted=1671.668507, expected=1667.300000
predicted=1668.216063, expected=1681.300000
predicted=1678.423393, expected=1659.700000
predicted=1662.282911, expected=1669.600000
predicted=1672.503535, expected=1634.500000
predicted=1639.043442, expected=1651.900000
predicted=1656.851647, expected=1640.300000
predicted=1639.710821, expected=1614.700000
predicted=1625.057450, expected=1656.000000
predicted=1649.953291, expected=1620.400000
predicted=1623.294200, expected=1615.500000
predicted=1623.195866, expected=1616.300000
predicted=1613.680874, expected=1650.600000
predicted=1648.640433, expected=1592.500000
predicted=1597.806606, expected=1637.500000
predicted=1636.557389, expected=1739.200000
predicted=1709.479200, expected=1805.500000
predicted=1778.897222, expected=1806.600000
predicted=1783.568970, expected=1776.400000
predicted=1769.583282, expected=1812.100000
predicted=1807.109296, expected=1778.500000
```

```
predicted=1781.975752, expected=1823.500000
predicted=1821.821460, expected=1826.800000
predicted=1815.529537, expected=1809.900000
predicted=1814.875314, expected=1856.400000
predicted=1844.538584, expected=1854.400000
predicted=1848.585575, expected=1814.200000
predicted=1822.047599, expected=1869.900000
predicted=1860.884367, expected=1873.700000
predicted=1866.574087, expected=1826.000000
predicted=1836.249101, expected=1828.500000
predicted=1829.583711, expected=1826.700000
predicted=1829.917668, expected=1788.400000
predicted=1801.011204, expected=1794.100000
predicted=1798.297201, expected=1759.800000
predicted=1767.370769, expected=1754.100000
predicted=1764.734535, expected=1858.300000
predicted=1838.638194, expected=1888.700000
predicted=1868.200369, expected=1848.900000
predicted=1846.257153, expected=1818.900000
predicted=1819.584004, expected=1791.200000
predicted=1803.331417, expected=1782.400000
predicted=1795.456196, expected=1755.500000
predicted=1765.654286, expected=1740.200000
predicted=1750.405226, expected=1748.800000
predicted=1751.286305, expected=1781.300000
predicted=1776.869567, expected=1740.000000
predicted=1742.617292, expected=1710.200000
predicted=1720.549901, expected=1648.800000
predicted=1664.358462, expected=1656.200000
predicted=1672.127922, expected=1663.400000
predicted=1665.489339, expected=1641.900000
predicted=1649.778150, expected=1619.000000
predicted=1624.237949, expected=1628.300000
predicted=1631.000083, expected=1613.400000
predicted=1617.883344, expected=1615.000000
predicted=1618.601026, expected=1616.600000
predicted=1615.075911, expected=1612.000000
predicted=1614.264570, expected=1601.300000
predicted=1603.451227, expected=1586.800000
predicted=1591.455767, expected=1596.700000
predicted=1598.354736, expected=1600.900000
predicted=1599.937064, expected=1602.100000
predicted=1601.863045, expected=1589.800000
predicted=1590.299693, expected=1589.000000
predicted=1591.117953, expected=1585.200000
predicted=1586.572947, expected=1561.900000
predicted=1568.199856, expected=1548.800000
predicted=1555.197131, expected=1541.200000
```

```
predicted=1546.231455, expected=1530.200000
predicted=1536.311996, expected=1528.700000
predicted=1531.830894, expected=1512.300000
predicted=1516.413466, expected=1482.300000
predicted=1491.596510, expected=1502.300000
predicted=1504.305891, expected=1509.900000
predicted=1507.812133, expected=1499.700000
predicted=1501.910510, expected=1496.000000
predicted=1495.285741, expected=1500.500000
predicted=1500.579148, expected=1520.100000
predicted=1517.075784, expected=1552.900000
predicted=1543.320634, expected=1546.000000
predicted=1540.210447, expected=1541.500000
predicted=1540.403254, expected=1538.600000
predicted=1537.569473, expected=1529.300000
predicted=1533.371020, expected=1525.600000
predicted=1528.252029, expected=1523.800000
predicted=1525.303502, expected=1515.000000
predicted=1517.909395, expected=1528.600000
predicted=1527.785750, expected=1542.100000
predicted=1537.678165, expected=1538.100000
predicted=1536.747165, expected=1543.300000
predicted=1540.884721, expected=1546.500000
predicted=1544.246569, expected=1541.700000
predicted=1542.613058, expected=1532.000000
predicted=1533.855313, expected=1542.400000
predicted=1542.209089, expected=1535.900000
predicted=1536.249957, expected=1536.300000
predicted=1537.747667, expected=1522.800000
predicted=1524.165073, expected=1526.600000
predicted=1529.180491, expected=1523.200000
predicted=1523.482868, expected=1515.300000
predicted=1518.529680, expected=1508.800000
predicted=1510.729318, expected=1492.200000
predicted=1497.145382, expected=1495.600000
predicted=1498.792810, expected=1479.800000
predicted=1483.180726, expected=1490.400000
predicted=1492.484809, expected=1493.400000
predicted=1490.818826, expected=1506.600000
predicted=1504.876625, expected=1501.100000
predicted=1498.192786, expected=1516.600000
predicted=1514.184732, expected=1502.900000
predicted=1501.759883, expected=1491.200000
predicted=1496.018714, expected=1480.900000
predicted=1483.694715, expected=1514.900000
predicted=1512.697094, expected=1540.100000
predicted=1530.754057, expected=1556.700000
predicted=1548.569930, expected=1556.000000
```

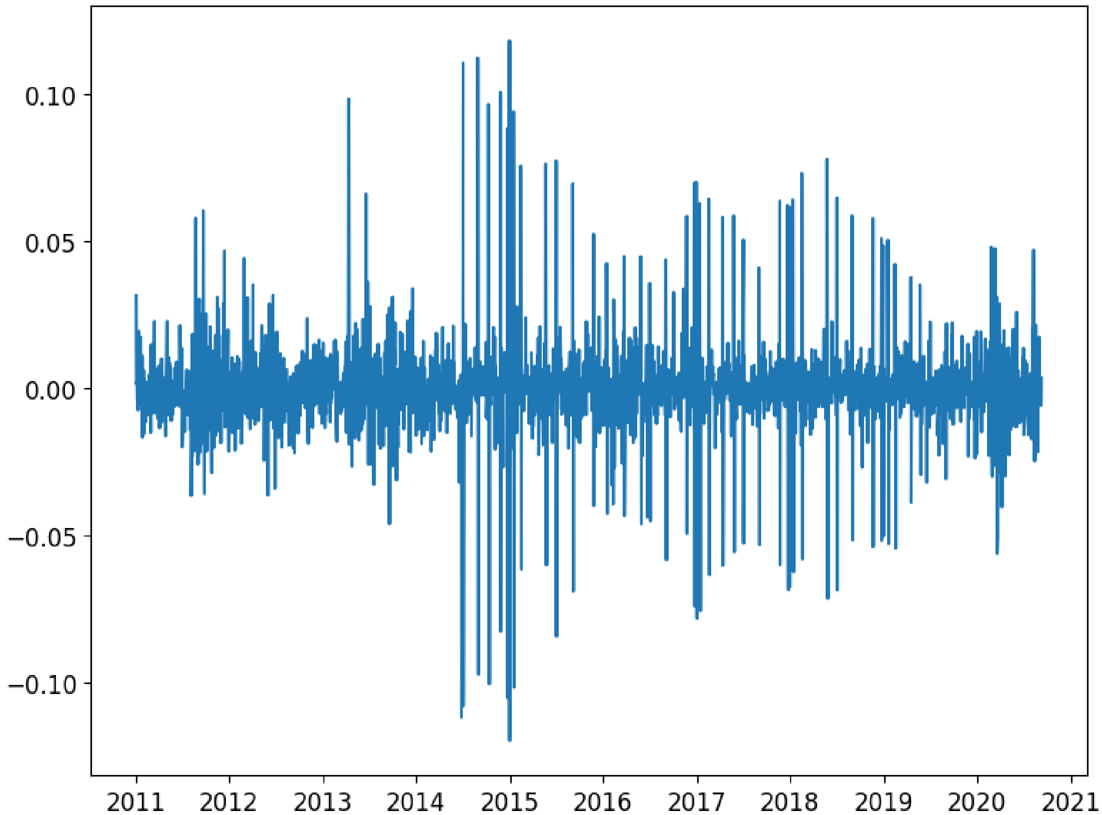
```
predicted=1548.575319, expected=1530.800000
predicted=1533.320740, expected=1516.700000
predicted=1522.725697, expected=1503.000000
predicted=1509.530809, expected=1508.600000
predicted=1513.035382, expected=1503.200000
predicted=1504.492176, expected=1498.300000
predicted=1500.852418, expected=1494.500000
predicted=1495.318161, expected=1492.300000
predicted=1494.256359, expected=1485.300000
predicted=1487.446386, expected=1471.700000
predicted=1475.709378, expected=1454.900000
predicted=1460.691577, expected=1452.900000
predicted=1457.344881, expected=1467.400000
predicted=1466.960180, expected=1473.400000
predicted=1471.179676, expected=1458.500000
predicted=1459.283130, expected=1457.700000
predicted=1458.665605, expected=1451.800000
predicted=1453.454235, expected=1432.200000
predicted=1438.454892, expected=1428.100000
predicted=1432.112881, expected=1438.900000
predicted=1439.025001, expected=1423.800000
predicted=1426.624792, expected=1416.000000
predicted=1419.411300, expected=1419.800000
predicted=1419.891358, expected=1426.100000
predicted=1426.426176, expected=1434.800000
predicted=1432.322010, expected=1437.900000
predicted=1434.982893, expected=1427.500000
predicted=1427.975881, expected=1426.200000
predicted=1427.415305, expected=1415.900000
predicted=1418.361899, expected=1404.000000
predicted=1409.098061, expected=1396.000000
predicted=1399.750076, expected=1392.600000
predicted=1396.059832, expected=1424.600000
predicted=1420.614734, expected=1421.500000
predicted=1417.155974, expected=1412.200000
predicted=1413.196496, expected=1429.300000
predicted=1424.659973, expected=1426.900000
predicted=1425.987504, expected=1434.100000
predicted=1433.338893, expected=1428.200000
predicted=1426.168408, expected=1416.000000
predicted=1419.653674, expected=1437.200000
predicted=1434.918580, expected=1430.700000
predicted=1429.523938, expected=1409.300000
predicted=1414.469789, expected=1408.700000
predicted=1410.311580, expected=1415.300000
predicted=1416.341125, expected=1413.400000
predicted=1414.563896, expected=1400.500000
predicted=1402.335451, expected=1388.200000
```

```
predicted=1392.058490, expected=1384.700000
predicted=1388.360962, expected=1374.700000
predicted=1378.829203, expected=1373.600000
predicted=1376.500096, expected=1364.600000
predicted=1366.804674, expected=1359.900000
predicted=1363.221830, expected=1361.900000
predicted=1362.525710, expected=1364.800000
predicted=1364.976147, expected=1363.400000
predicted=1363.271276, expected=1347.600000
predicted=1350.240099, expected=1348.300000
predicted=1350.721495, expected=1352.300000
predicted=1352.320470, expected=1331.500000
predicted=1336.049400, expected=1339.600000
predicted=1341.019023, expected=1333.800000
predicted=1333.919454, expected=1340.700000
predicted=1342.285518, expected=1318.400000
predicted=1320.074525, expected=1333.000000
predicted=1334.709853, expected=1332.300000
predicted=1329.980705, expected=1344.500000
predicted=1344.196118, expected=1341.000000
predicted=1337.494054, expected=1346.500000
predicted=1346.096714, expected=1370.200000
predicted=1363.899600, expected=1368.100000
predicted=1364.895267, expected=1360.400000
predicted=1360.553289, expected=1386.900000
predicted=1381.362967, expected=1385.700000
predicted=1382.582767, expected=1384.000000
predicted=1384.140142, expected=1373.700000
predicted=1373.102313, expected=1368.500000
predicted=1372.169763, expected=1371.400000
predicted=1372.310958, expected=1373.400000
predicted=1373.702226, expected=1378.500000
predicted=1377.321698, expected=1422.600000
Test MSE: 417.137
```



Using this ARIMA model on differenciated outputs

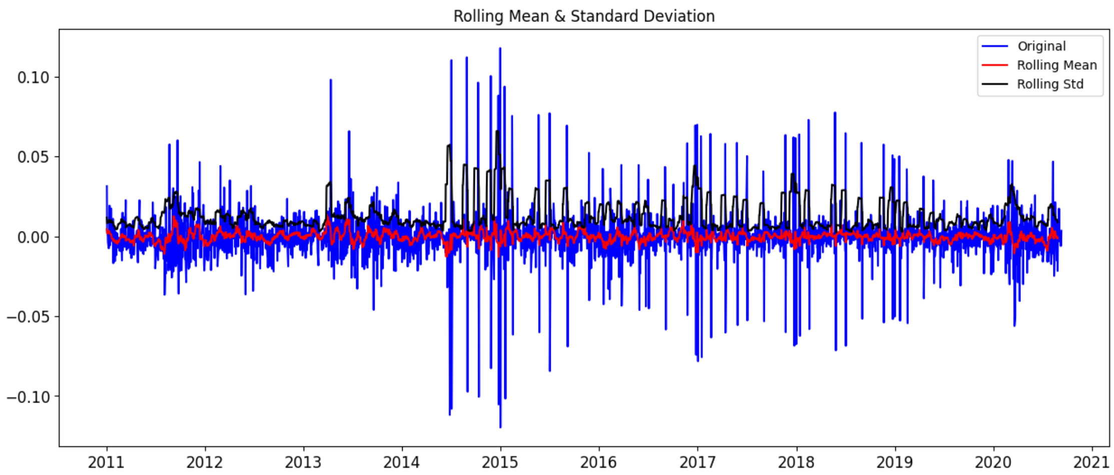
[58]:
```
ts_log_diff = ts_log - ts_log.shift()
plt.figure(figsize=(9,7))
plt.plot(ts_log_diff)
plt.show()
```



[59]:
```
ts_log_diff.head()
```

[59]:
```
Date
2020-09-11         NaN
2020-09-10    0.003544
2020-09-09   -0.004797
2020-09-08   -0.006003
2020-09-07   -0.003144
Name: Price, dtype: float64
```

[60]:
```
ts_log_diff.dropna(inplace=True)
test_stationarity(ts_log_diff)
```




```
Results of Dickey-Fuller Test:
Test Statistic                  -24.003958
p-value                           0.000000
#Lags Used                        5.000000
Number of Observations Used    2524.000000
Critical Value (1%)              -3.432943
Critical Value (5%)              -2.862686
Critical Value (10%)             -2.567380
dtype: float64
```

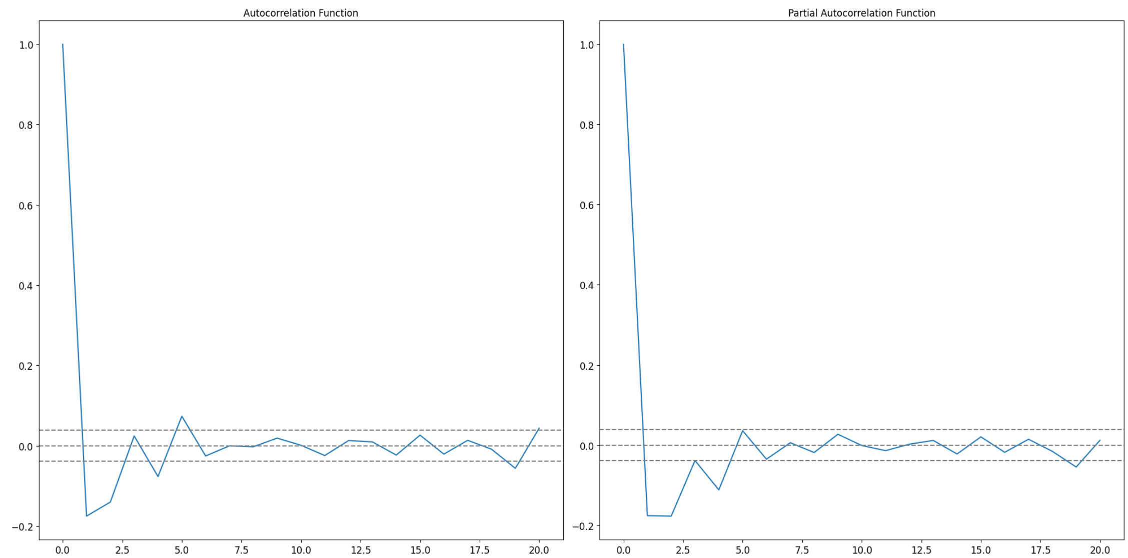
[61]:
```
ts_log_diff.head()
```

[61]:
```
Date
2020-09-10    0.003544
2020-09-09   -0.004797
2020-09-08   -0.006003
2020-09-07   -0.003144
2020-09-06    0.001831
Name: Price, dtype: float64
```

[62]:
```
from statsmodels.tsa.stattools import acf, pacf

lag_acf = acf(ts_log_diff, nlags=20)
lag_pacf = pacf(ts_log_diff, nlags=20, method='ols')
```

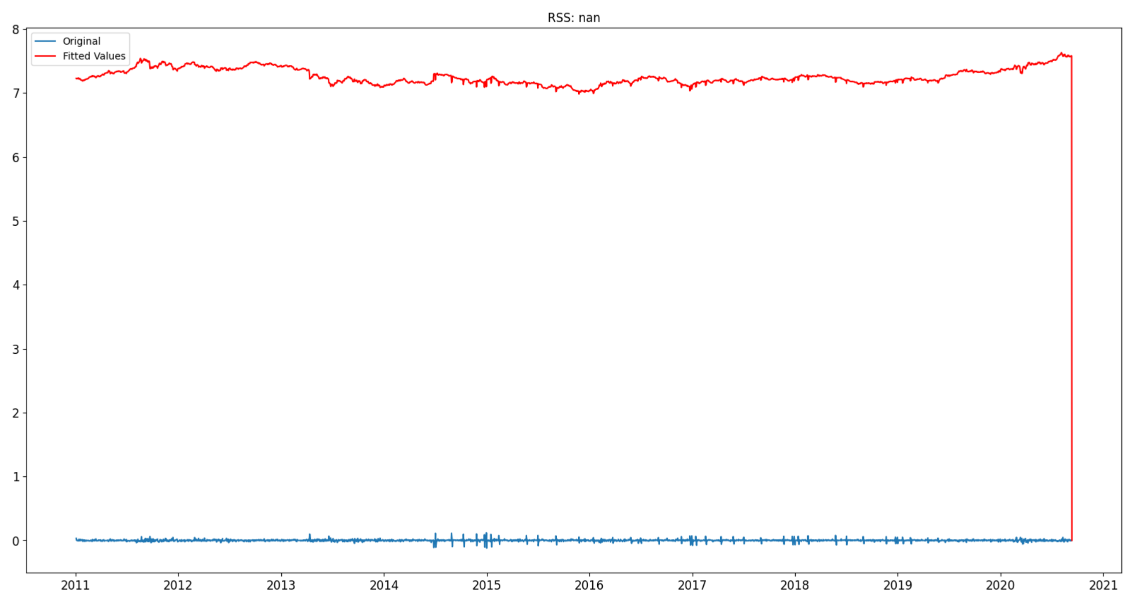
[63]:
```
plt.figure(figsize=(20,10))

#Plot ACF:
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
```

```
plt.axhline(y=-1.96/np.sqrt(len(ts_log_diff)),linestyle='--',color='gray')
plt.axhline(y=1.96/np.sqrt(len(ts_log_diff)),linestyle='--',color='gray')
plt.title('Autocorrelation Function')

#Plot PACF:
plt.subplot(122)
plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(ts_log_diff)),linestyle='--',color='gray')
plt.axhline(y=1.96/np.sqrt(len(ts_log_diff)),linestyle='--',color='gray')
plt.title('Partial Autocorrelation Function')
plt.tight_layout()
plt.show()
```



[65]:
```
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt

# AR model
model = ARIMA(ts_log, order=(1, 1, 0))
results_AR = model.fit()

# Plotting the results
plt.figure(figsize=(20, 10))
plt.plot(ts_log_diff, label='Original')
plt.plot(results_AR.fittedvalues, color='red', label='Fitted Values')
plt.title('RSS: %.4f' % sum((results_AR.fittedvalues - ts_log_diff) ** 2))
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
```



[67]:
```python
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
```

```python
# MA model
model = ARIMA(ts_log, order=(0, 1, 1))
results_MA = model.fit()

# Plotting the results
plt.figure(figsize=(20, 10))
plt.plot(ts_log_diff, label='Original')
plt.plot(results_MA.fittedvalues, color='red', label='Fitted Values')
plt.title('RSS: %.4f' % sum((results_MA.fittedvalues - ts_log_diff) ** 2))
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
```

[76]:
```python
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt

# Combined ARIMA model
model = ARIMA(ts_log, order=(1, 1, 1))
results_ARIMA = model.fit()

# Plotting the results
plt.figure(figsize=(20, 10))
plt.plot(ts_log_diff, label='Original')
plt.plot(results_ARIMA.fittedvalues, color='red', label='Fitted Values')
plt.title('RSS: %.4f' % sum((results_ARIMA.fittedvalues - ts_log_diff) ** 2))
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
```

```
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it has no associated frequency
information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473:
ValueWarning: A date index has been provided, but it is not monotonic and so
will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
```

[77]:
```
predictions_ARIMA_diff = pd.Series(results_ARIMA.fittedvalues, copy=True)
print (predictions_ARIMA_diff.head())
```

```
Date
2020-09-11    0.000000
2020-09-10    7.579347
2020-09-09    7.582180
2020-09-08    7.578740
2020-09-07    7.573900
dtype: float64
```

[78]:
```
predictions_ARIMA_diff_cumsum = predictions_ARIMA_diff.cumsum()
print (predictions_ARIMA_diff_cumsum.head())
```

```
Date
2020-09-11     0.000000
```

```
2020-09-10     7.579347
2020-09-09    15.161527
2020-09-08    22.740267
2020-09-07    30.314166
dtype: float64
```

[79]:
```
predictions_ARIMA_log = pd.Series(ts_log.iloc[0], index=ts_log.index)
predictions_ARIMA_log = predictions_ARIMA_log.
 ↪add(predictions_ARIMA_diff_cumsum,fill_value=0)
predictions_ARIMA_log.head()
```

[79]:
```
Date
2020-09-11     7.579347
2020-09-10    15.158694
2020-09-09    22.740874
2020-09-08    30.319614
2020-09-07    37.893513
dtype: float64
```

[80]:
```
predictions_ARIMA = np.exp(predictions_ARIMA_log)
predictions_ARIMA
```

[80]:
```
Date
2020-09-11    1.957350e+03
2020-09-10    3.831219e+06
2020-09-09    7.520316e+09
2020-09-08    1.471096e+13
2020-09-07    2.863806e+16
                  …
2011-01-07             inf
2011-01-06             inf
2011-01-05             inf
2011-01-04             inf
2011-01-03             inf
Length: 2531, dtype: float64
```

[83]:
```
import numpy as np
from sklearn.metrics import mean_absolute_error

# Ensure that predictions_ARIMA and the actual values have the same length
# Assuming predictions_ARIMA is for the test set, extract the corresponding␣
 ↪test set from goldprice['Price']
test_size = len(predictions_ARIMA)
actual_test_values = goldprice['Price'][-test_size:]

# Now calculate the MAE
error = mean_absolute_error(actual_test_values, predictions_ARIMA)
```

```
print("MAE: %.3f" % error)

# Calculate RMSE
rmse = np.sqrt(np.mean((predictions_ARIMA - actual_test_values) ** 2))
print('RMSE: %.4f' % rmse)
```

```
MAE: 187318155507306782051480191768507153685797169349699122873278839050517062300
428305502008118470599499836446180971429288009569965761633542258471206749652618 46
967116208636791312872961489637470600608359633761912852512543499228887432359556 31
555065891580459969991353886491856817322551799790206291617654320083763 20.000
RMSE: inf
```

Saving the predictions

[85]:
```
from sklearn.ensemble import RandomForestRegressor

rf_model = RandomForestRegressor().fit(X_train,y_train)
preds1 = rf_model.predict(X_test)
preds1
```

[85]:
```
array([1280.193 , 1374.073 , 1344.464 , 1719.095 , 1330.2575, 1290.322 ,
       1142.758 , 1345.439 , 1265.819 , 1304.758 , 1713.282 , 1343.314 ,
       1361.444 , 1283.829 , 1571.714 , 1313.723 , 1334.002 , 1405.699 ,
       1524.442 , 1330.066 , 1754.894 , 1340.84  , 1415.928 , 1264.723 ,
       1328.3265, 1538.194 , 1750.115 , 1201.84  , 1593.015 , 1965.949 ,
       1355.739 , 1341.6765, 1679.31  , 1401.018 , 1321.546 , 1361.133 ,
       1588.2785, 1394.9385, 1649.926 , 1368.629 , 1194.987 , 1361.547 ,
       1427.017 , 1243.056 , 1312.657 , 1285.157 , 1627.973 , 1367.308 ,
       1443.882 , 1335.14  , 1282.66  , 1594.6665, 1354.989 , 1542.866 ,
       1444.81  , 1303.465 , 1272.255 , 1535.8885, 1265.5186, 1522.232 ,
       1373.475 , 1754.11  , 1753.118 , 1585.909 , 1346.269 , 1638.211 ,
       1247.906 , 1677.108 , 1551.7885, 1497.235 , 1627.759 , 1454.716 ,
       1589.018 , 1678.562 , 1570.1   , 1561.8025, 1232.288 , 1233.003 ,
       1324.6225, 1256.995 , 1358.082 , 1492.866 , 1350.229 , 1335.002 ,
       1569.7555, 1596.955 , 1295.896 , 1444.964 , 1600.72  , 1574.5135,
       1283.086 , 1323.3215, 1411.972 , 1432.978 , 1605.7115, 1537.412 ,
       1788.608 , 1340.072 , 1284.518 , 1273.149 , 1789.13  , 1252.229 ,
       1416.335 , 1198.326 , 1366.33  , 1287.827 , 1453.24  , 1495.668 ,
       1574.493 , 1413.966 , 1734.222 , 1430.699 , 1386.362 , 1940.5485,
       1677.73  , 1648.232 , 1306.084 , 1291.8442, 1762.954 , 1426.777 ,
       1325.58  , 1403.498 , 1405.099 , 1423.897 , 1175.544 , 1240.845 ,
       1219.825 , 1525.5235, 1413.583 , 1618.054 , 1554.0655, 1403.933 ,
       1143.7435, 1433.782 , 1518.8375, 1215.046 , 1310.551 , 1215.703 ,
       1373.728 , 1400.205 , 1293.093 , 1326.912 , 1369.55  , 1208.873 ,
       1940.8325, 1611.086 , 1344.3156, 1330.11  , 1744.752 , 1580.455 ,
       1317.522 , 1950.732 , 1740.198 , 1404.459 , 1268.263 , 1236.861 ,
       1231.6825, 1412.599 , 1314.363 , 1300.549 , 1297.903 , 1275.654 ,
       1425.022 , 1203.3025, 1355.217 , 1289.0686, 1740.277 , 1367.66  ,
```

```
1259.0285, 1285.    , 1301.896 , 1350.306 , 1428.713 , 1451.175 ,
1729.39  , 1141.464 , 1358.247 , 1611.016 , 1616.03  , 1572.0145,
1536.539 , 1329.9905, 1544.874 , 1503.58  , 1769.726 , 1240.067 ,
1295.592 , 1601.568 , 1182.847 , 1238.092 , 1415.444 , 1737.2295,
1366.602 , 1123.431 , 1274.501 , 1495.1435, 1367.008 , 1453.759 ,
1322.108 , 1725.975 , 1282.592 , 1281.165 , 1367.0295, 1340.3758,
1410.803 , 1350.104 , 1150.613 , 1662.7   , 1347.011 , 1323.319 ,
1418.025 , 1335.23  , 1295.984 , 1283.687 , 1240.642 , 1775.206 ,
1643.999 , 1575.9695, 1268.2335, 1748.232 , 1385.317 , 1336.73  ,
1299.68  , 1407.658 , 1272.226 , 1591.701 , 1317.84  , 1613.573 ,
1544.263 , 1248.994 , 1657.205 , 1281.058 , 1332.588 , 1357.547 ,
1435.672 , 1343.369 , 1586.715 , 1375.209 , 1503.01  , 1698.12  ,
1278.229 , 1663.05  , 1722.064 , 1467.11  , 1311.209 , 1238.9   ,
1134.0965, 1662.76  , 1295.0545, 1839.83  , 1517.31  , 1333.935 ,
1286.7956, 1295.289 , 1270.263 , 1490.529 , 1283.1175, 1756.906 ,
1426.401 , 1418.752 , 1286.9336, 1570.411 , 1773.814 , 1295.5126,
1284.239 , 1699.675 , 1328.461 , 1753.035 , 1279.2965, 1276.344 ,
1372.914 , 1311.609 , 1392.748 , 1303.756 , 1324.714 , 1205.274 ,
1644.999 , 1774.59  , 1588.3095, 1414.584 , 1229.06  , 1434.567 ,
1454.799 , 1554.669 , 1341.404 , 1787.442 , 1397.2515, 1744.856 ,
1668.783 , 1520.056 , 1764.893 , 1225.895 , 1395.317 , 1324.245 ,
1423.474 , 1335.9398, 1450.142 , 1343.584 , 1344.8506, 1638.301 ,
1336.593 , 1412.505 , 1341.5266, 1270.657 , 1812.206 , 1391.424 ,
1251.719 , 1385.37  , 1887.9475, 1205.279 , 1299.691 , 1665.283 ,
1186.423 , 1316.948 , 1327.514 , 1508.075 , 1666.104 , 1359.594 ,
1503.334 , 1207.5695, 1760.525 , 1230.782 , 1237.9805, 1274.462 ,
1298.755 , 1430.625 , 1256.271 , 1459.595 , 1283.6455, 1329.111 ,
1516.495 , 1319.117 , 1333.16  , 1848.857 , 1670.673 , 1324.5275,
1141.143 , 1381.968 , 1303.652 , 1300.786 , 1757.075 , 1617.309 ,
1274.238 , 1599.025 , 1313.971 , 1614.691 , 1269.371 , 1318.859 ,
1373.839 , 1645.926 , 1580.9085, 1318.556 , 1298.765 , 1849.274 ,
1274.298 , 1604.362 , 1588.364 , 1249.384 , 1382.321 , 1323.98  ,
1286.538 , 1544.768 , 1528.926 , 1497.955 , 1277.283 , 1371.411 ,
1461.112 , 1278.724 , 1520.7655, 1505.582 , 1582.5055, 1227.389 ,
1365.703 , 1717.081 , 1238.154 , 1454.4545, 1296.204 , 1525.002 ,
1418.503 , 1280.988 , 1561.707 , 1342.1806, 1214.031 , 1297.18  ,
1356.085 , 1378.055 , 1560.593 , 1306.781 , 1120.463 , 1388.252 ,
1333.924 , 1387.196 , 1557.3445, 1337.2258, 1218.159 , 1577.197 ,
1587.29  , 1711.773 , 1293.4786, 1382.008 , 1740.114 , 1535.688 ,
1308.226 , 1561.0435, 1283.828 , 1322.918 , 1266.66  , 1378.288 ,
1120.864 , 1343.006 , 1396.524 , 1351.643 , 1618.464 , 1402.206 ,
1369.162 , 1639.94  , 1349.834 , 1375.501 , 1328.459 , 1425.089 ,
1309.695 , 1317.213 , 1276.851 , 1497.777 , 1377.3895, 1447.186 ,
1320.562 , 1450.309 , 1291.6412, 1137.3235, 1544.634 , 1829.263 ,
1243.323 , 1701.817 , 1373.348 , 1288.35  , 1284.165 , 1288.192 ,
1662.43  , 1434.162 , 1217.889 , 1706.765 , 1452.357 , 1677.001 ,
1628.794 , 1291.999 , 1287.1515, 1268.3324, 1394.051 , 1857.558 ,
```

```
        1222.94  , 1298.836 , 1654.83  , 1327.794 , 1369.482 , 1642.453 ,
        1787.931 , 1350.464 , 1499.428 , 1305.881 , 1616.367 , 1305.795 ,
        1399.913 , 1672.065 , 1345.521 , 1335.229 , 1374.012 , 1361.275 ,
        1333.246 , 1286.549 , 1292.0616, 1357.382 , 1637.414 , 1599.718 ,
        1370.807 , 1415.023 , 1150.469 , 1612.642 , 1312.331 , 1949.6715,
        1306.081 , 1205.534 , 1329.597 , 1827.18  , 1577.48  , 1331.735 ,
        1241.4065, 1271.929 , 1657.559 , 1671.72  , 1744.865 , 1743.247 ,
        1277.359 , 1292.7122, 1310.762 , 1334.905 , 1251.8475, 1255.356 ,
        1727.7185, 1643.258 , 1140.88  , 1900.1565, 1182.8255, 1812.567 ,
        1247.133 , 1574.341 , 1134.0895])
```

[86]:
```python
from xgboost import XGBRegressor

xgb_model = XGBRegressor().fit(X_train,y_train)
preds2 = xgb_model.predict(X_test)
preds2
```

[86]:
```
array([1278.1401, 1371.271 , 1355.9695, 1723.461 , 1334.4335, 1297.1885,
       1143.4845, 1345.4817, 1261.3967, 1301.5546, 1712.7925, 1346.2677,
       1361.0388, 1280.0442, 1570.3239, 1307.7097, 1330.5502, 1405.5698,
       1528.4703, 1334.4691, 1751.9741, 1339.1648, 1416.658 , 1269.2885,
       1328.5906, 1542.5172, 1746.5859, 1201.6   , 1589.8477, 1968.6177,
       1368.1272, 1332.0349, 1681.0333, 1411.5604, 1321.3562, 1368.2838,
       1587.745 , 1392.7683, 1653.4155, 1365.9961, 1190.6746, 1366.0255,
       1425.8407, 1248.3562, 1309.78  , 1283.9656, 1634.9989, 1366.5922,
       1443.5984, 1332.3934, 1309.3379, 1592.4606, 1358.125 , 1543.663 ,
       1444.7942, 1301.2952, 1272.8468, 1533.4299, 1265.3164, 1530.9901,
       1368.8342, 1759.8955, 1754.0508, 1556.21  , 1344.388 , 1647.2428,
       1247.8563, 1674.9934, 1552.3225, 1503.549 , 1619.5886, 1457.717 ,
       1614.9015, 1673.7311, 1571.3754, 1561.7765, 1238.839 , 1240.098 ,
       1323.3727, 1261.1567, 1357.4338, 1501.3079, 1351.5875, 1360.682 ,
       1565.3318, 1599.4858, 1296.2756, 1444.8475, 1604.5632, 1580.2817,
       1298.9015, 1335.5979, 1414.1458, 1437.333 , 1595.7356, 1532.4955,
       1787.0574, 1361.242 , 1253.5277, 1262.0485, 1787.2238, 1234.8044,
       1415.709 , 1202.5835, 1361.9067, 1287.6216, 1462.7489, 1496.2802,
       1575.358 , 1413.6388, 1725.308 , 1430.6484, 1384.8676, 1940.5999,
       1668.3163, 1648.9375, 1304.911 , 1284.184 , 1773.7411, 1426.2117,
       1333.1687, 1403.8514, 1402.8187, 1422.5507, 1168.6296, 1239.81  ,
       1214.3531, 1526.9353, 1412.6139, 1612.4258, 1554.7963, 1406.0164,
       1139.9231, 1438.9379, 1515.0935, 1205.7859, 1307.9661, 1218.8116,
       1374.0381, 1400.606 , 1301.995 , 1335.3917, 1371.3673, 1211.531 ,
       1934.105 , 1612.1024, 1353.5139, 1330.0481, 1748.1047, 1575.8652,
       1317.6875, 1872.7856, 1737.0591, 1403.7838, 1273.0391, 1234.7642,
       1240.0137, 1414.432 , 1300.219 , 1297.8947, 1299.8444, 1278.68  ,
       1426.6141, 1208.2771, 1356.5972, 1283.8226, 1739.9668, 1364.4336,
       1242.8846, 1283.6166, 1304.3206, 1352.0343, 1429.5084, 1450.4612,
       1730.4215, 1139.5007, 1370.532 , 1611.441 , 1614.9751, 1572.8901,
```

1539.9576, 1326.8142, 1542.6788, 1504.8601, 1764.9355, 1245.9175,
1293.2754, 1607.317 , 1183.8771, 1246.0428, 1416.9904, 1738.8918,
1390.2001, 1122.9612, 1274.1154, 1494.618 , 1365.8163, 1458.0974,
1321.9692, 1720.5029, 1281.0718, 1283.1833, 1385.4495, 1342.2816,
1405.2263, 1349.0986, 1154.5426, 1661.524 , 1346.1055, 1317.141 ,
1415.5275, 1336.6482, 1292.6216, 1281.9204, 1245.883 , 1775.68 ,
1645.6095, 1573.7771, 1267.9812, 1744.5032, 1383.0267, 1330.4617,
1297.8392, 1408.6117, 1270.3572, 1592.0887, 1317.3297, 1609.0696,
1543.3334, 1247.458 , 1654.3256, 1283.0173, 1318.4066, 1357.1974,
1433.836 , 1343.501 , 1592.4296, 1372.963 , 1501.176 , 1702.2039,
1278.3516, 1663.5883, 1721.4946, 1465.6284, 1311.8967, 1242.5391,
1135.5974, 1662.448 , 1311.4135, 1846.7109, 1513.1067, 1320.5234,
1283.8212, 1297.8049, 1272.0736, 1488.8386, 1285.9332, 1768.5712,
1438.0448, 1425.0048, 1287.2444, 1572.3243, 1776.2441, 1251.2731,
1276.9556, 1698.985 , 1325.8254, 1748.8888, 1269.4489, 1271.9973,
1372.6641, 1311.3741, 1392.1544, 1302.9042, 1320.5657, 1207.6532,
1639.0042, 1772.4243, 1579.0714, 1415.5527, 1209.155 , 1435.0444,
1467.249 , 1554.1332, 1339.2148, 1795.1394, 1397.1542, 1747.916 ,
1669.2935, 1517.7278, 1760.9276, 1227.056 , 1393.5388, 1317.3552,
1426.8419, 1334.7557, 1452.0264, 1332.1422, 1344.8751, 1647.8969,
1328.2809, 1413.23 , 1341.2748, 1273.141 , 1824.6028, 1394.3624,
1242.9128, 1382.8298, 1873.4943, 1204.1012, 1293.1136, 1666.6737,
1183.718 , 1323.9946, 1326.8599, 1515.3857, 1662.9457, 1364.6366,
1501.3649, 1207.0576, 1759.8098, 1231.2573, 1234.8068, 1275.5709,
1299.433 , 1431.3004, 1257.6656, 1467.031 , 1287.0248, 1310.6682,
1517.8877, 1316.6273, 1331.2999, 1860.5161, 1669.3477, 1332.1057,
1138.7745, 1380.8285, 1303.7114, 1298.4424, 1751.7356, 1615.1791,
1276.3864, 1596.1725, 1313.7517, 1612.1031, 1269.102 , 1314.9797,
1377.4923, 1643.8844, 1579.2393, 1319.8379, 1293.5724, 1860.1747,
1274.3198, 1608.4713, 1587.1663, 1244.879 , 1378.7319, 1320.2559,
1281.9232, 1543.0208, 1525.9465, 1500.1459, 1279.07 , 1372.5803,
1466.9852, 1276.0702, 1515.3767, 1506.4999, 1577.6522, 1224.3018,
1363.6818, 1713.1558, 1242.7108, 1458.133 , 1293.2662, 1526.407 ,
1420.9037, 1277.5219, 1559.8522, 1345.1552, 1212.3429, 1302.998 ,
1356.5808, 1374.2977, 1558.4381, 1307.9718, 1115.7528, 1392.3804,
1331.8647, 1379.9117, 1553.872 , 1336.014 , 1217.5753, 1583.6182,
1583.1661, 1713.5565, 1288.2429, 1381.2958, 1740.9135, 1536.98 ,
1306.6041, 1562.9935, 1285.7896, 1316.4216, 1263.8276, 1378.0854,
1123.1833, 1342.8508, 1395.6616, 1356.9032, 1618.904 , 1398.7333,
1365.0759, 1642.5986, 1323.7134, 1369.7854, 1331.4812, 1423.9309,
1308.6188, 1325.2765, 1277.1478, 1496.7229, 1379.2478, 1438.8896,
1319.3661, 1451.4161, 1287.5803, 1135.426 , 1543.6741, 1842.3591,
1240.8102, 1699.3098, 1372.7012, 1284.4009, 1285.1558, 1274.2968,
1665.6487, 1434.2847, 1208.611 , 1690.3779, 1450.391 , 1679.8824,
1632.9539, 1302.7577, 1294.6521, 1271.7325, 1394.2062, 1841.2067,
1212.8605, 1290.1649, 1655.6014, 1328.808 , 1369.7107, 1649.6575,
1785.3904, 1350.008 , 1504.1289, 1306.9027, 1618.1052, 1301.9467,

```
       1400.7657, 1671.5106, 1346.824 , 1334.9899, 1371.2576, 1364.4167,
       1326.105 , 1282.6528, 1284.2422, 1360.5416, 1635.4604, 1594.3336,
       1373.4031, 1414.8662, 1149.1198, 1612.3893, 1316.2723, 1856.4001,
       1306.308 , 1209.5226, 1317.1808, 1838.4152, 1573.8733, 1316.6603,
       1244.446 , 1277.0359, 1658.0624, 1673.3661, 1742.9031, 1751.9768,
       1277.1068, 1289.5853, 1312.2646, 1331.2972, 1253.6708, 1258.6484,
       1724.9952, 1641.7196, 1133.873 , 1836.3104, 1182.3663, 1812.1621,
       1243.878 , 1592.7012, 1136.4862], dtype=float32)
```

[87]:
```
gold_price = pd.read_csv("/content/drive/MyDrive/GoldPrice.csv")
```

[88]:
```
rf_predictions = pd.DataFrame()
rf_predictions['Predicted Price'] = preds1
rf_predictions
```

[88]:
```
     Predicted Price
0          1280.1930
1          1374.0730
2          1344.4640
3          1719.0950
4          1330.2575
..               …
502        1182.8255
503        1812.5670
504        1247.1330
505        1574.3410
506        1134.0895

[507 rows x 1 columns]
```

[89]:
```
xgb_predictions = pd.DataFrame()
xgb_predictions['Predicted Price'] = preds2
xgb_predictions
```

[89]:
```
     Predicted Price
0        1278.140137
1        1371.270996
2        1355.969482
3        1723.461060
4        1334.433472
..               …
502      1182.366333
503      1812.162109
504      1243.878052
505      1592.701172
506      1136.486206
```

[507 rows x 1 columns]

[93]:
```
rf_predictions.to_csv(r'/content/drive/MyDrive/RandomForest.
 ↪csv',header=True,index=False)
```

[94]:
```
xgb_predictions.to_csv(r'/content/drive/MyDrive/XGBoost predictions.
 ↪csv',header=True,index=False)
```