Lab 4 Task 1

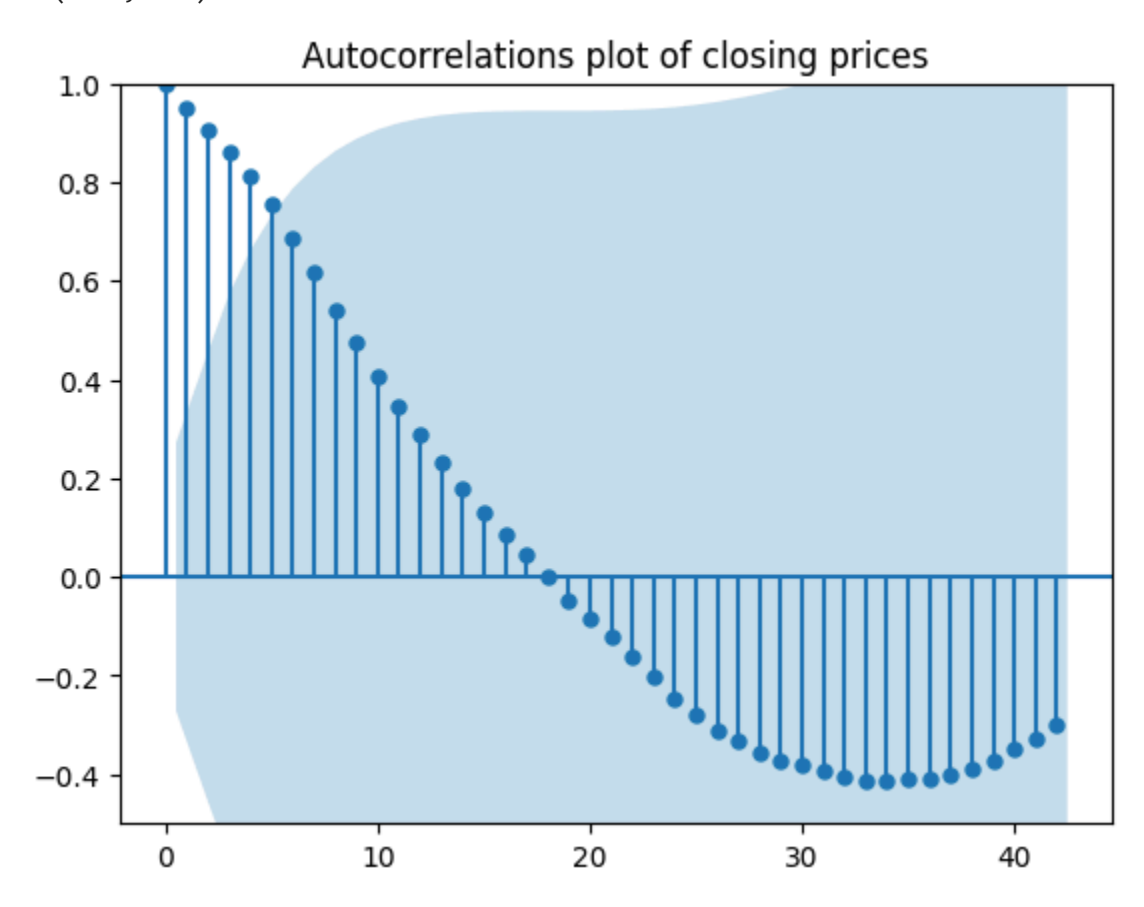
In [315…
```python
import matplotlib.pyplot as plt
import pandas as pd
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.tsa.stattools import adfuller
```

In [316…
```python
data = pd.read_csv('ibm_stock_data.csv', parse_dates=['Date'], index_col=['Date'])
```

Autocorrelation plot for analyzing correlations

In [317…
```python
fig=plot_acf(data['Close'], lags=42, title='Autocorrelations plot of closing prices')
plt.ylim(-0.5, 1.0)
```

Out[317…
```
(-0.5, 1.0)
```



Ljung-Box test

In [318…
```python
acorr_ljungbox(data['Close'], lags=10, return_df=True)
```

Out[318…

	lb_stat	lb_pvalue
1	49.647329	1.840204e-12
2	95.709808	1.647705e-21
3	138.188973	9.288206e-30
4	176.923862	3.412791e-37
5	210.847023	1.355455e-43
6	239.542178	7.030766e-49
7	263.157367	4.372787e-53
8	281.815526	3.037164e-56
9	296.638981	1.347451e-58
10	307.566167	3.905240e-60

Augment Dickey-Fuller test for stationarity

In [319…
```python
adfuller(data['Close'])
```

Out[319…
```
(-0.9398265150729506,
 0.7745742973095588,
 0,
 51,
 {'1%': -3.5656240522121956,
  '5%': -2.920142229157715,
  '10%': -2.598014675124952},
 155.18453180208093)
```

Difference the time series

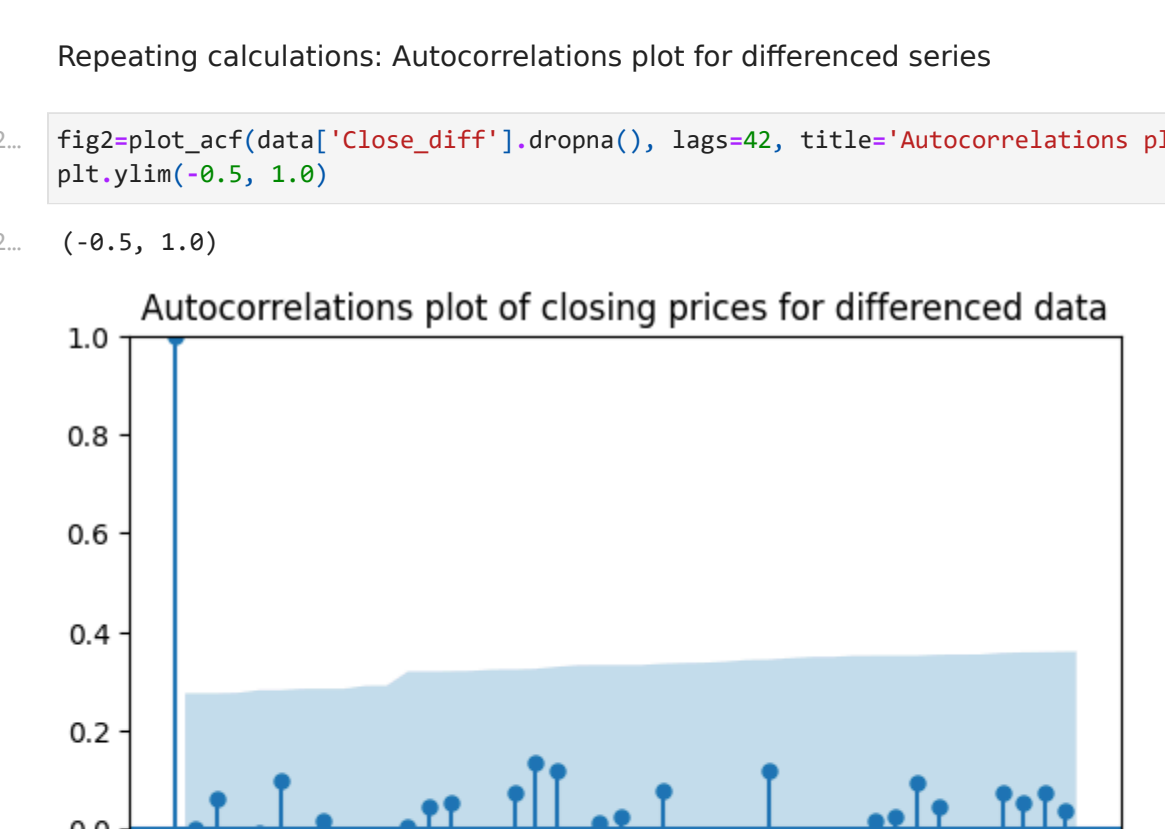
In [320…
```python
data['Close_diff']=data['Close'].diff()
data
```

Out[320…

Date	Open	High	Low	Close	Adj Close	Volume	Close_diff
2022-10-03	120.160004	122.209999	119.599998	121.510002	113.363281	4261700	NaN
2022-10-04	122.800003	125.650002	122.519997	125.500000	117.085762	4566100	3.989998
2022-10-05	124.709999	126.459999	124.230003	125.739998	117.309669	3212900	0.239998
2022-10-06	124.879997	125.300003	121.769997	122.230003	114.035011	5074600	-3.509995
2022-10-07	121.500000	121.800003	118.070000	118.820000	110.853615	4499700	-3.410003
2022-10-10	119.790001	119.959999	117.040001	117.750000	109.855362	5990000	-1.070000
2022-10-11	117.459999	119.230003	116.940002	117.800003	109.902016	4043100	0.050003
2022-10-12	118.000000	118.809998	117.199997	117.570000	109.687431	3338800	-0.230003
2022-10-13	116.099998	122.150002	115.550003	121.790001	113.624504	5837500	4.220001
2022-10-14	121.800003	122.540001	119.839996	120.040001	111.991829	3762400	-1.750000
2022-10-17	121.800003	122.879997	121.430000	121.519997	113.372597	5458600	1.479996
2022-10-18	123.000000	123.940002	121.820000	122.940002	114.697395	5120300	1.420005
2022-10-19	122.360001	123.940002	121.989998	122.510002	114.296234	5906600	-0.430000
2022-10-20	126.250000	128.960007	125.150002	128.300003	119.698044	13623100	5.790001
2022-10-21	128.389999	130.850006	127.589996	129.899994	121.190750	7201300	1.599991
2022-10-24	130.899994	133.110001	129.850006	132.690002	123.793701	5610900	2.790008
2022-10-25	132.000000	133.300003	131.300003	132.929993	124.017601	5957600	0.239991
2022-10-26	133.720001	135.860001	132.809998	135.009995	125.958153	5139000	2.080002
2022-10-27	135.550003	136.399994	134.449997	134.770004	125.734253	3993200	-0.239991
2022-10-28	135.559998	138.860001	135.220001	138.509995	129.223511	5965500	3.739991
2022-10-31	138.059998	138.770004	136.600006	138.289993	129.018250	4915300	-0.220002
2022-11-01	138.250000	138.649994	136.699997	138.199997	128.934265	3590600	-0.089996
2022-11-02	137.750000	140.169998	136.800003	136.830002	127.656143	5360500	-1.369995
2022-11-03	136.419998	136.479996	133.970001	134.470001	125.454361	4442400	-2.360001
2022-11-04	135.649994	137.729996	134.940002	136.960007	127.777412	4178000	2.490006
2022-11-07	136.639999	138.699997	136.509995	138.339996	129.064896	4043100	1.379989
2022-11-08	139.000000	140.929993	138.720001	140.039993	130.650909	5042800	1.699997
2022-11-09	137.949997	138.899994	136.940002	137.389999	129.706833	4720000	-2.649994
2022-11-10	140.259995	141.369995	138.289993	141.229996	133.332092	5389000	3.839997
2022-11-11	141.500000	144.130005	140.960007	143.169998	135.163620	5871300	1.940002
2022-11-14	142.630005	146.080002	142.179993	144.199997	136.136017	5293500	1.029999
2022-11-15	144.080002	146.160004	142.000000	144.339996	136.268173	4727100	0.139999
2022-11-16	144.130005	144.949997	144.009995	144.520004	136.438095	3445200	0.180008
2022-11-17	143.410004	146.179993	143.250000	146.089996	137.920319	3954000	1.569992
2022-11-18	146.559998	148.309998	145.940002	147.639999	139.383636	4661700	1.550003
2022-11-21	147.550003	147.929993	146.449997	146.679993	138.477310	3476200	-0.960006
2022-11-22	147.600006	149.350006	147.020004	149.100006	140.761978	7062100	2.420013
2022-11-23	149.100006	150.460007	148.300003	148.750000	140.431564	3658200	-0.350006
2022-11-25	148.270004	149.490005	148.100006	148.369995	140.072815	2075200	-0.380005
2022-11-28	147.979996	148.240005	145.940002	146.179993	138.005264	3538100	-2.190002
2022-11-29	145.910004	147.169998	145.699997	146.490005	138.297943	2754700	0.310012
2022-11-30	146.190002	149.639999	145.669998	148.899994	140.573166	6377600	2.409989
2022-12-01	149.979996	150.009995	147.339996	149.160004	140.818649	4495900	0.260010
2022-12-02	148.130005	149.160004	147.729996	148.669998	140.356033	2900000	-0.490006
2022-12-05	147.940002	148.929993	146.800003	147.410004	139.166489	2784600	-1.259994
2022-12-06	147.300003	147.800003	146.699997	147.500000	139.251465	2847600	0.089996
2022-12-07	147.330002	148.100006	146.289993	147.270004	139.034332	3971300	-0.229996
2022-12-08	147.899994	149.149994	147.369995	147.779999	139.515808	2665700	0.509995
2022-12-09	147.399994	148.339996	146.970001	147.050003	138.826660	3047600	-0.729996
2022-12-12	147.820007	149.210007	146.940002	149.210007	140.865845	4032800	2.160004
2022-12-13	150.369995	153.210007	149.949997	150.570007	142.149765	8811500	1.360000
2022-12-14	150.470001	151.910004	148.449997	149.860001	141.479507	4205900	-0.710006

In [321…
```python
data['Close_diff'].plot()
```

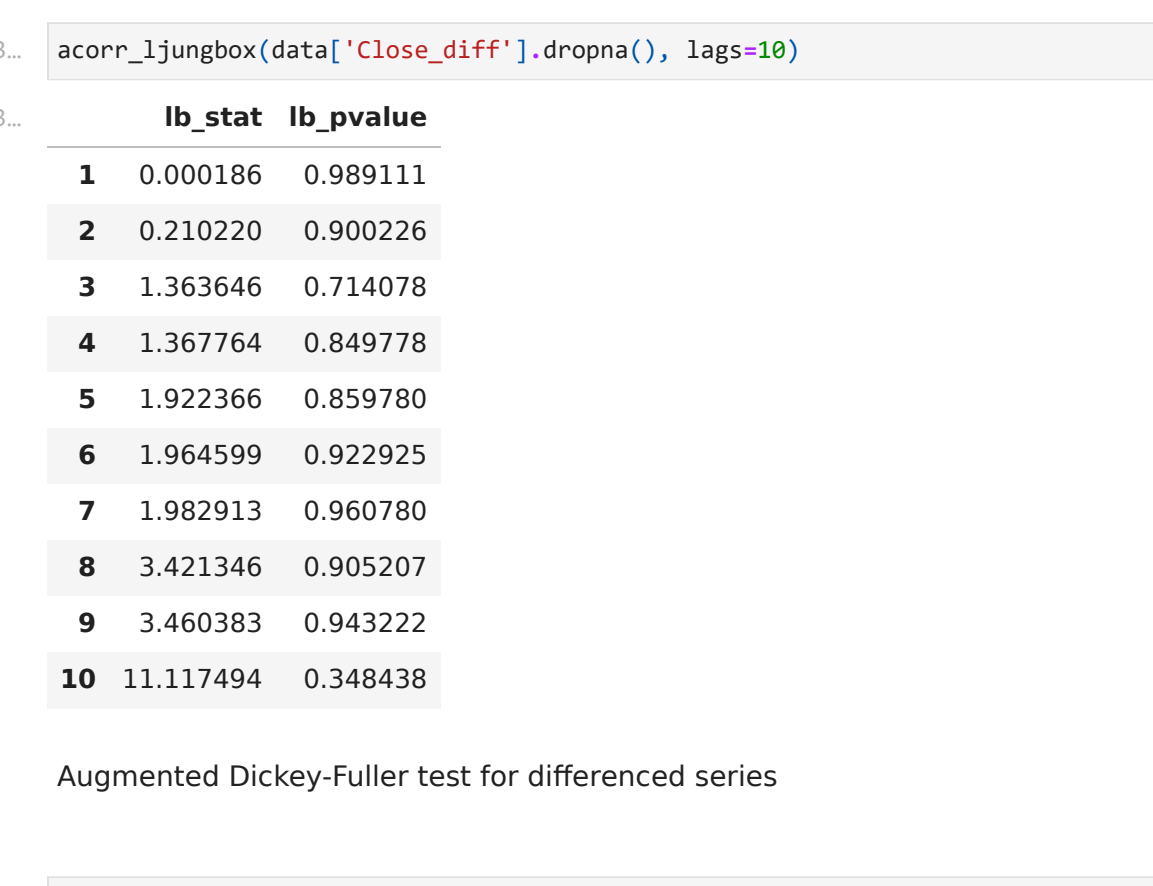
Out[321…
```
<Axes: xlabel='Date'>
```



Repeating calculations: Autocorrelations plot for differenced series

In [322…
```python
fig2=plot_acf(data['Close_diff'].dropna(), lags=42, title='Autocorrelations plot of closing prices for differenced data')
plt.ylim(-0.5, 1.0)
```

Out[322…
```
(-0.5, 1.0)
```



Ljung-Box test for differenced series

In [323…
```python
acorr_ljungbox(data['Close_diff'].dropna(), lags=10)
```

Out[323…

	lb_stat	lb_pvalue
1	0.000186	0.989111
2	0.210220	0.900226
3	1.363646	0.714078
4	1.367764	0.849778
5	1.922366	0.859780
6	1.964599	0.922925
7	1.982913	0.960780
8	3.421346	0.905207
9	3.460383	0.943222
10	11.117494	0.348438

Augmented Dickey-Fuller test for differenced series

In [324…
```python
adfuller(data['Close_diff'].dropna())
```

Out[324…
```
(-3.4975414025237113,
 0.008050461013716191,
 9,
 41,
 {'1%': -3.60098336718852,
  '5%': -2.9351348158036012,
  '10%': -2.6059629803688282},
 153.7775047611643)
```

Task 2

In [325…
```python
df = pd.read_csv('electricity.csv', skiprows=3, header=1, index_col=['Month'])
df.columns = ['Commercial']
df.dropna(inplace=True)
df
```

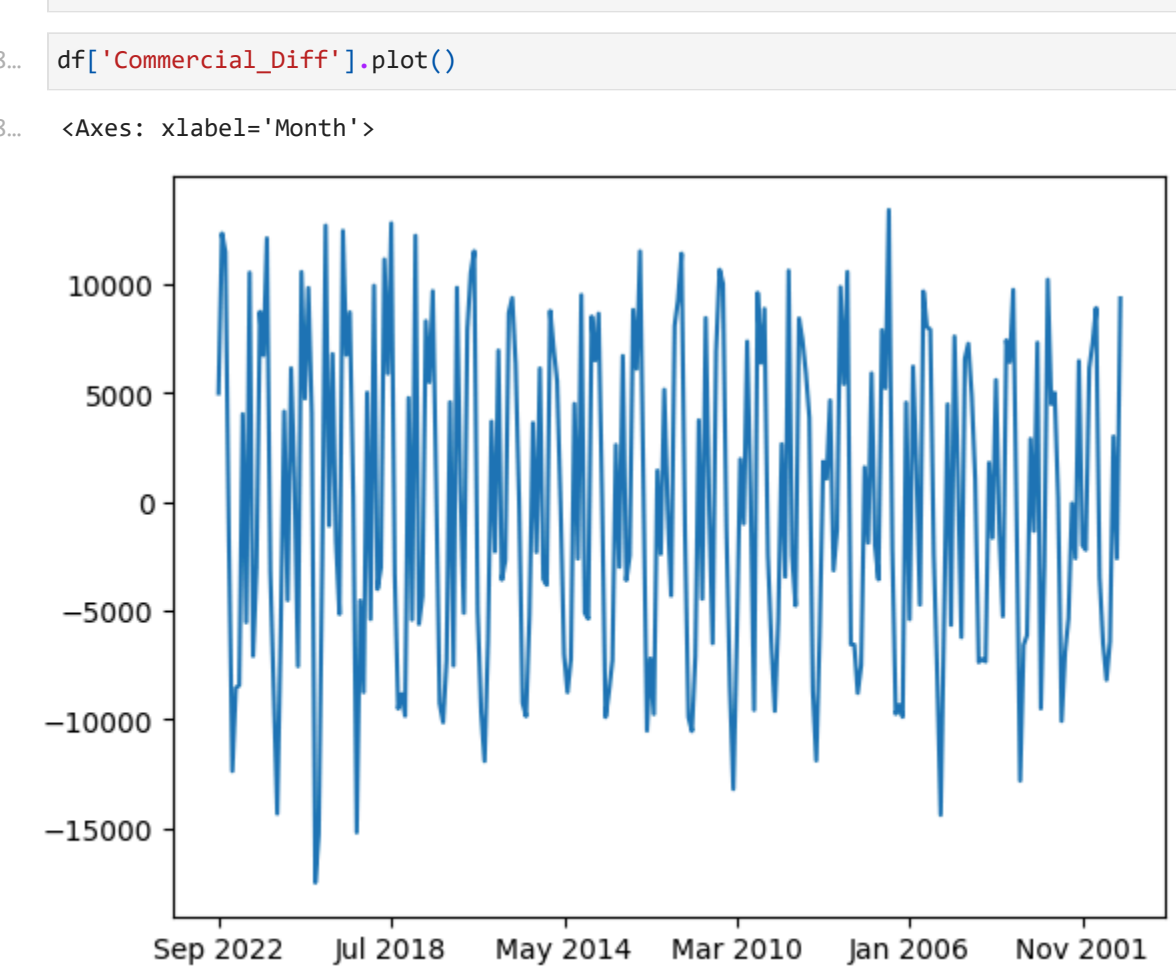
Out[325…

Month	Commercial
Oct 2022	106857.96303
Sep 2022	111851.35754
Aug 2022	124195.27519
Jul 2022	135675.95261
Jun 2022	133951.71136
...	...
Apr 2001	87470.74429
Mar 2001	81060.53414
Feb 2001	84064.24963
Jan 2001	81466.85778
Dec 2000	90825.27235

263 rows × 1 columns

In [326…
```python
df['Commercial'].plot()
```

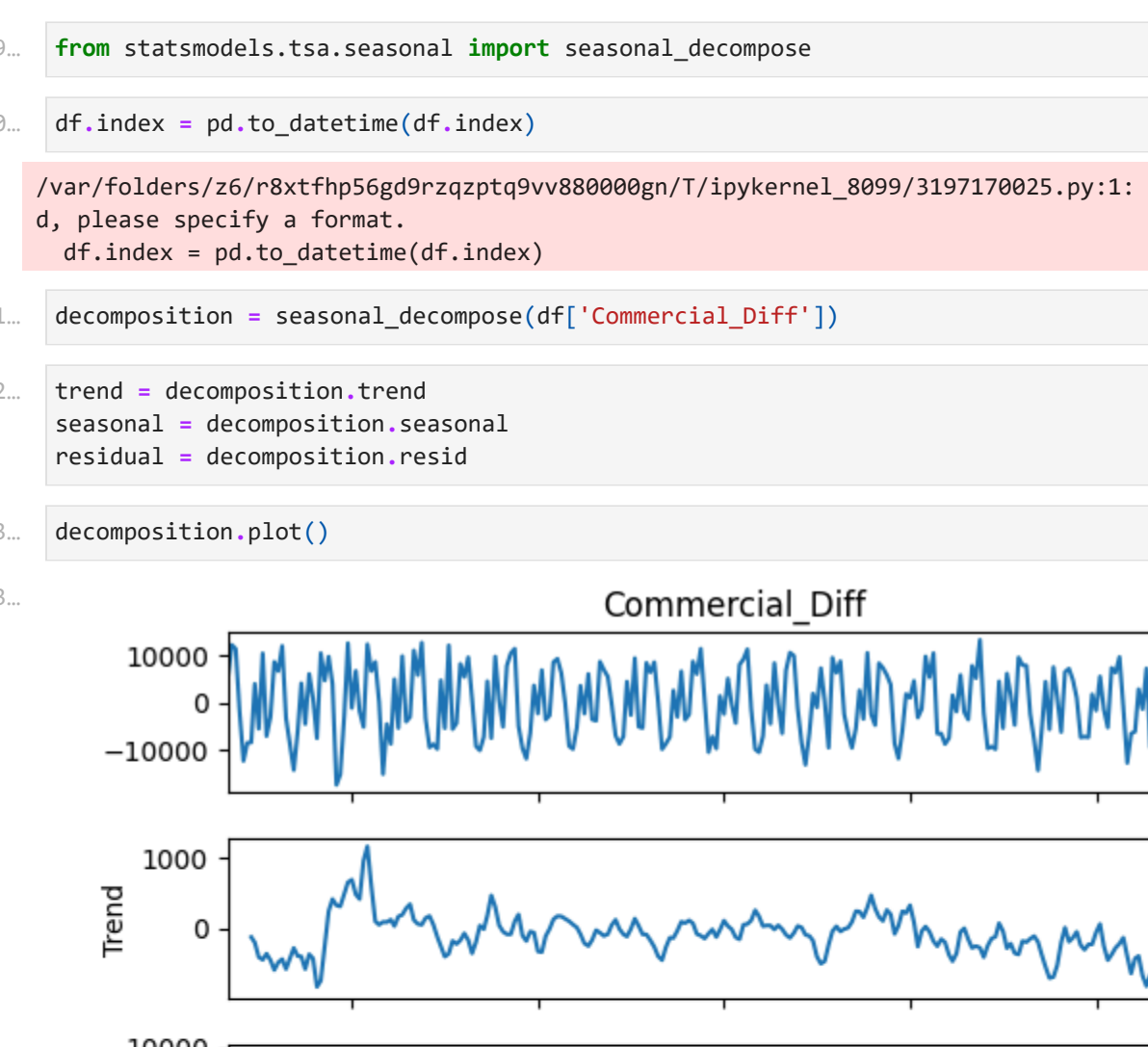
Out[326…
```
<Axes: xlabel='Month'>
```



In [327…
```python
df['Commercial_Diff']=df['Commercial'].diff()
df.dropna(inplace=True)
```

In [328…
```python
df['Commercial_Diff'].plot()
```

Out[328…
```
<Axes: xlabel='Month'>
```



Remove trend and seasonal component

In [329…
```python
from statsmodels.tsa.seasonal import seasonal_decompose
```

In [330…
```python
df.index = pd.to_datetime(df.index)
```

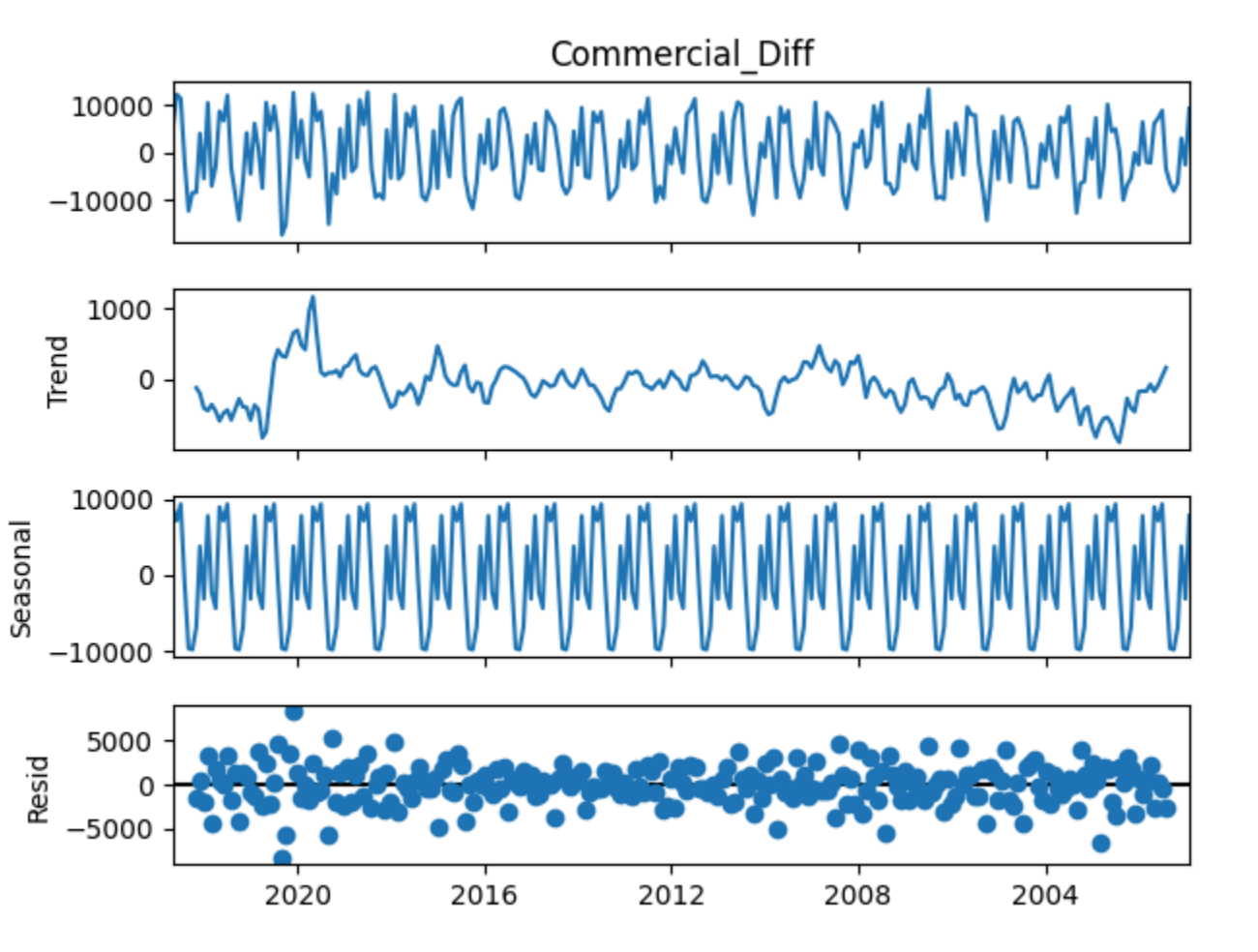
```
/var/folders/z6/r8xtfhp56gd9rzqzptq9vv880000gn/T/ipykernel_8099/3197170025.py:1: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  df.index = pd.to_datetime(df.index)
```

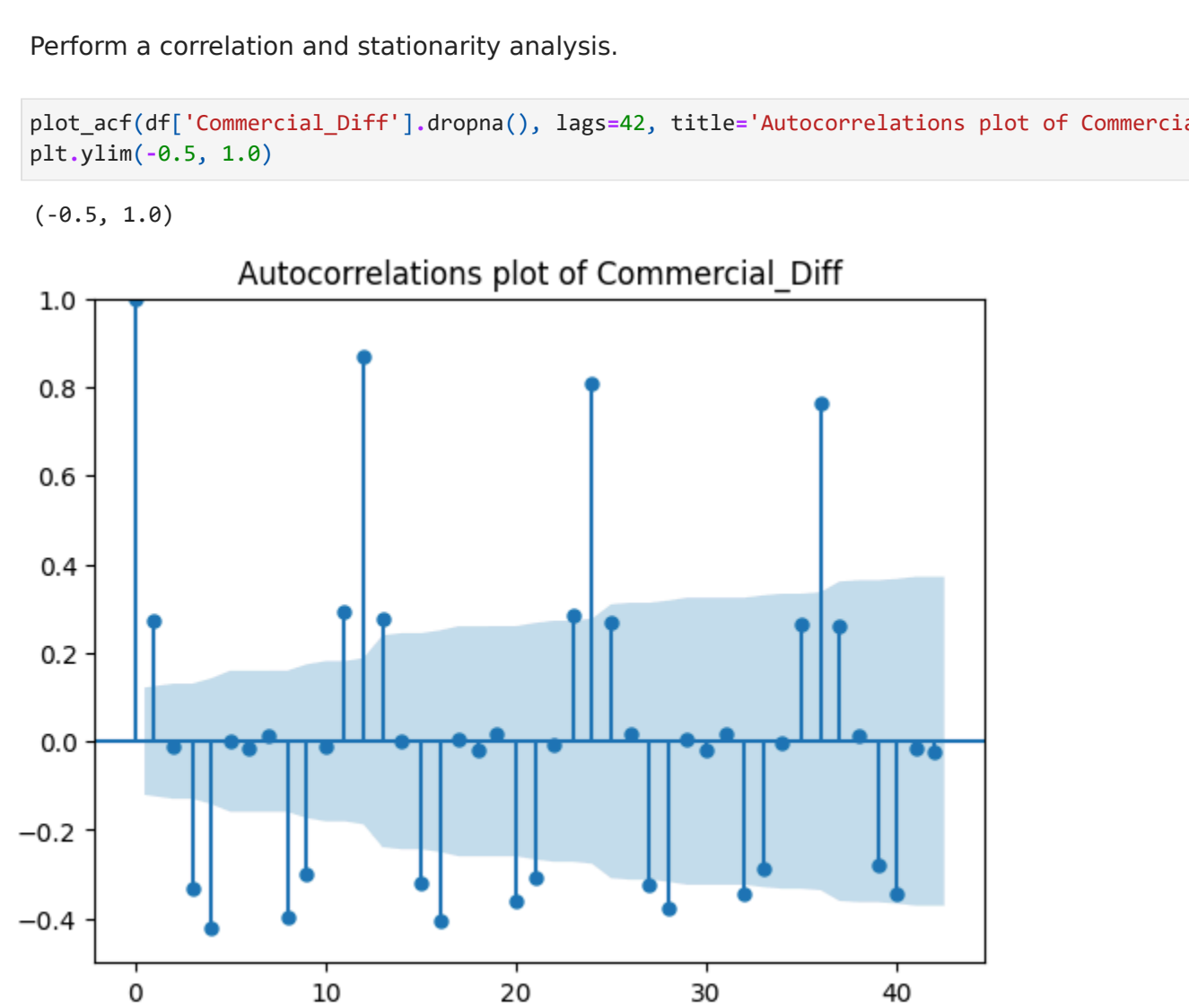
In [331…
```python
decomposition = seasonal_decompose(df['Commercial_Diff'])
```

In [332…
```python
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
```

In [333…
```python
decomposition.plot()
```

Out[333…

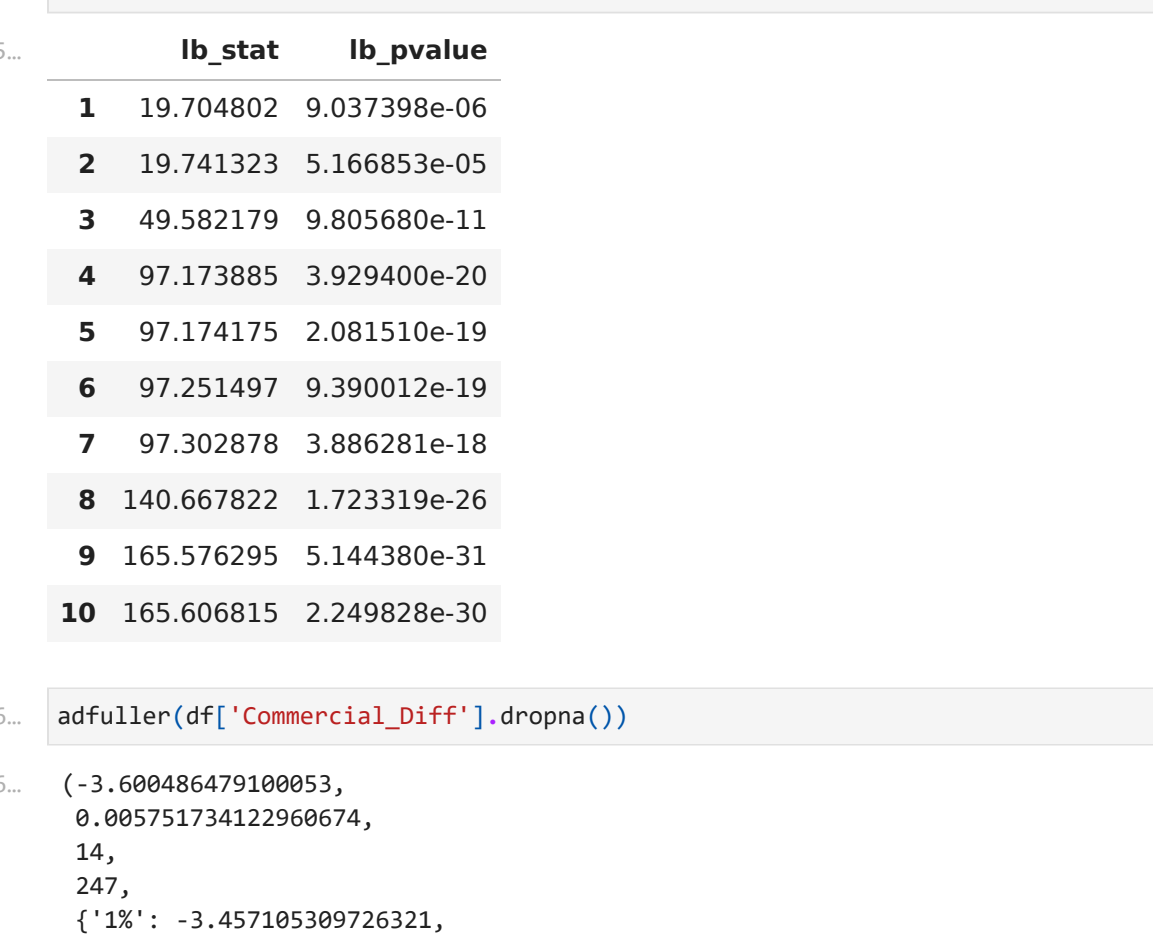




Perform a correlation and stationarity analysis.

In [334…
```python
plot_acf(df['Commercial_Diff'].dropna(), lags=42, title='Autocorrelations plot of Commercial_Diff')
plt.ylim(-0.5, 1.0)
```

Out[334…
```
(-0.5, 1.0)
```



In [335…
```python
acorr_ljungbox(df['Commercial_Diff'].dropna(), lags=10, return_df=True)
```

Out[335…

	lb_stat	lb_pvalue
1	19.704802	9.037398e-06
2	19.741323	5.166853e-05
3	49.582179	9.805680e-11
4	97.173885	3.929400e-20
5	97.174175	2.081510e-19
6	97.251497	9.390012e-19
7	97.302878	3.886281e-18
8	140.667822	1.723319e-26
9	165.576295	5.144380e-31
10	165.606815	2.249828e-30

In [336…
```python
adfuller(df['Commercial_Diff'].dropna())
```

Out[336…
```
(-3.600486479100053,
 0.005751734122960674,
 14,
 247,
 {'1%': -3.457105309726321,
  '5%': -2.873313676101283,
  '10%': -2.5730443824681606},
 4572.673365665536)
```

p-value > 0.05 - This implies that time-series is non-stationary. p-value <=0.05 - This implies that time-series is stationary