stock-and-airline-passengers-data

June 29, 2024

-ation: Time Series Analysis of Nvidia Stock and Airline Passengers Data

0.1 1. Intro- duction

This project involves time series analysis of two datasets: Nvidia (NVDA) stock prices and airline passengers data. The aim is to perform various statistical analyses and model predictions using techniques like moving averages, exponential moving averages, ARIMA, and SARIMA models.

0.2 2. Datasets

- **Nvidia Stock Data**: Historical stock prices for Nvidia Corporation fetched using the `yfinance` library.
- **Airline Passengers Data**: Monthly totals of international airline passengers from 1949 to 1960.

0.3 3. Libraries Used

- `pandas`: Data manipulation and analysis.
- `numpy`: Numerical operations.
- `matplotlib`: Data visualization.
- `statsmodels`: Statistical modeling.
- `yfinance`: Fetching financial data.

0.4 4. Data Loading and Preprocessing

0.4.1 Nvidia Stock Data

- Data for Nvidia stock prices was downloaded and loaded into a DataFrame.
- The data was inspected and cleaned for any missing values.

0.4.2 Airline Passengers Data

- Data for airline passengers was read from a CSV file.
- Missing values were handled and the 'Month' column was converted to datetime format for proper time series analysis.

0.5 5. Exploratory Data Analysis (EDA)

0.5.1 Nvidia Stock Data

- **Plotting Open Prices**: The opening prices of Nvidia stocks were plotted to observe the trend over time.
- **Simple Moving Averages (SMA)**: Calculated and plotted to smooth out short-term fluctuations and highlight longer-term trends.
- **Exponential Moving Averages (EMA)**: Calculated and plotted to give more weight to recent prices, thereby capturing more recent trends more effectively.

0.5.2 Airline Passengers Data

- **Plotting Passengers Data**: The data was plotted to visualize the monthly totals of international airline passengers.
- **ADF Test for Stationarity**: Conducted to check if the time series is stationary. Stationarity is essential for accurate modeling in time series analysis.
- **Differencing for Stationarity**: Applied to make the time series data stationary by removing trends and seasonality.

0.5.3 Autocorrelation and Partial Autocorrelation

- **Autocorrelation Function (ACF)**: Measures the correlation between the time series and its lagged values. Plotted to identify the extent of correlation with previous time steps.
- **Partial Autocorrelation Function (PACF)**: Measures the correlation between the time series and its lagged values while controlling for the values of the time steps in between. Plotted to identify the direct effect of previous time steps.

0.6 6. Modeling

0.6.1 ARIMA Model for Airline Passengers Data

- **Train-Test Split**: The data was divided into training and testing sets to evaluate the performance of the ARIMA model.
- **ARIMA Model Creation and Fitting**: An ARIMA model was fitted to the training data and used to make predictions.

0.6.2 SARIMA Model for Airline Passengers Data

- **Seasonality Consideration**: The airline passengers data shows clear seasonal patterns. To model this, a Seasonal ARIMA (SARIMA) model was used.
- **SARIMA Model Creation and Fitting**: A SARIMA model was fitted to the training data and used to make predictions. This model incorporates both seasonal and non-seasonal factors in the time series data.

0.7 7. Definitions of Important Models and Concepts

0.7.1 Moving Averages (MA)

- **Simple Moving Average (SMA)**: An average of the data points within a certain window. It smooths out short-term fluctuations and highlights longer-term trends or cycles.

- **Exponential Moving Average (EMA)**: A type of moving average that places a greater weight and significance on the most recent data points. This makes it more responsive t- o recent price changes compared to SMA.

0.7.2 Stationarity

A time series is said to be stationary if its statistical properties such as mean, variance, and autocorrelation are all constant over time. Stationarity is important for time series modeling because many statistical methods a- ssume or require a stationary time series.

0.7.3 Differencing

A technique used to make a non-stationary time series stationary by subtracting the previous observation from the current observation. It helps remove trends and seasonality from the data.

0.7.4 Autocorrelation and Partial Autocorrelation

- **Autocorrelation Function (ACF)**: A measure of the correlation between a time series and its lagged values. The ACF plot helps in identifying the extent of correlation with previous time steps and can guide in choosing the appropriate model parameters.
- **Partial Autocorrelation Function (PACF)**: A measure of the correlation between a time series and its lagged values while controlling for the values of the time steps in between. The PACF plot helps in identifying the direct effect of previous time steps and is used to determine - the order of the autoregressive part of the model.

0.7.5 ARIMA Model

ARIMA (AutoRegressive Integrated Moving Average) is a class of models that explains a given time series based on its own past values (autoregressive part), the differenced values (integrated part), and a moving average model applied - to the lagged forecast errors (moving average part).

0.7.6 SARIMA Model

SARIMA (Seasonal AutoRegressive Integrated Moving Average) extends the ARIMA model by explicitly modeling seasonal effects. It incorporates seasonal autoregressive (SAR), seasonal differencing (SD), and seasonal moving average (SMA) components along with non-seasonal components. This model is particularly useful for time series data wi- th strong seasonal patterns.

0.7.7 ADF Test (Augmented Dickey-Fuller Test)

A statistical test used to determine if a time series is stationary. It tests the null hypothesis that a unit root is present in a time series sample. A lower p-value indicates that the null hypothesis can be r- ejected, suggesting the time series is stationary.

0.8 8. Data Visualization

Visualizations were used extensively to understand data trends and patterns. Examples include: - Line plots to visualize stock prices over time. - Comparison of actual- vs. predicted values in time series forecasting.

0.9 9. Model Evaluation Metrics

To evaluate the performance of the time series models, metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) were used. These - metrics help quantify the accuracy of the predictions.

0.10 10. Hyperparameter Tuning

For both ARIMA and SARIMA models, parameters such as p (autoregressive order), d (differencing order), q (moving average order), P (seasonal autoregressive order), D (seasonal differencing order), Q (seasonal moving average order-), and m (seasonal period) were tuned to achieve the best model fit.

0.11 11. Conclusion

This project demonstrated the application of various time series analysis techniques on Nvidia stock data and airline passengers data. Techniques such as moving averages, exponential moving averages, differencing, ARIMA, and SARIMA models were used to analyze and forecast the time series data. The results provide insights into the trends and patterns within the datment and let me know if there are any additional details or sections you would like to include. me know if there are any additional details or sections you would like to include.

[ ]:
```
# first install yfinance
import yfinance as yf
import pandas as pd
from datetime import datetime
```

[70]:
```
# Fetch data for Nvidia Corporation
Nvda_data = yf.download('NVDA')
# Display the first few rows of the fetched data
Nvda_data.head()
```

```
[**********************100%%**********************]  1 of 1 completed
```

[70]:

Date	Open	High	Low	Close	Adj Close	Volume
1999-01-22	0.043750	0.048828	0.038802	0.041016	0.037621	2714688000
1999-01-25	0.044271	0.045833	0.041016	0.045313	0.041562	510480000
1999-01-26	0.045833	0.046745	0.041146	0.041797	0.038337	343200000
1999-01-27	0.041927	0.042969	0.039583	0.041667	0.038218	244368000
1999-01-28	0.041667	0.041927	0.041276	0.041536	0.038098	227520000

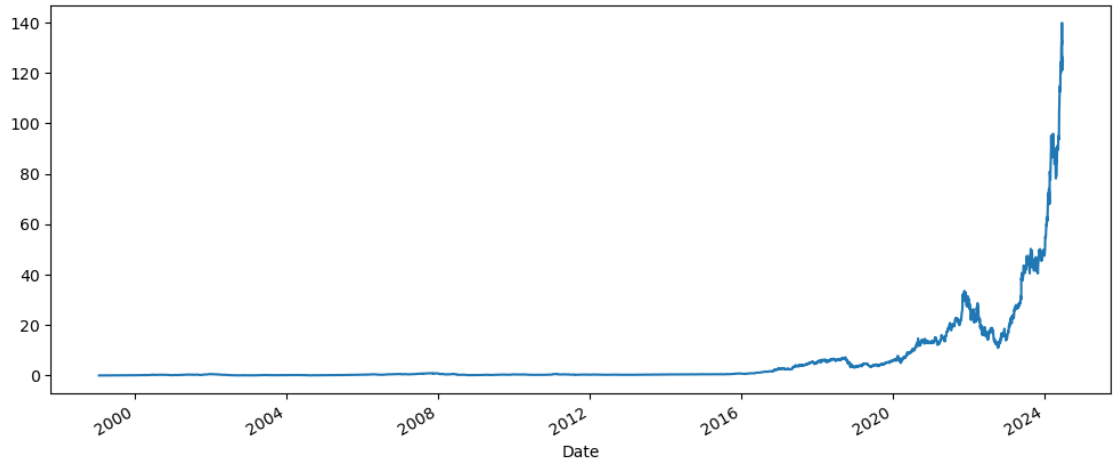
[68]:
```
Nvda_data.index
```

[68]:
```
DatetimeIndex(['1999-01-22', '1999-01-25', '1999-01-26', '1999-01-27',
               '1999-01-28', '1999-01-29', '1999-02-01', '1999-02-02',
               '1999-02-03', '1999-02-04',
               …
               '2024-06-14', '2024-06-17', '2024-06-18', '2024-06-20',
```

```
               '2024-06-21', '2024-06-24', '2024-06-25', '2024-06-26',
               '2024-06-27', '2024-06-28'],
              dtype='datetime64[ns]', name='Date', length=6400, freq=None)
```

[153]:
```
# plotting entire data
# Simple moving average-helps in smoothing of graph
Nvda_data['Open'].plot(figsize=(12,5))
```

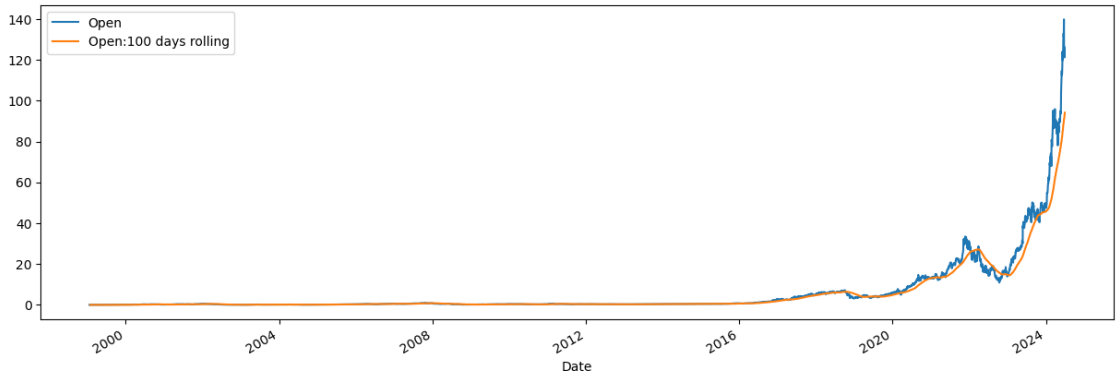
[153]: <Axes: xlabel='Date'>



[86]:
```
Nvda_data['Open:100 days rolling']=Nvda_data['Open'].
 ↪rolling(window=100,min_periods=5).mean()
```

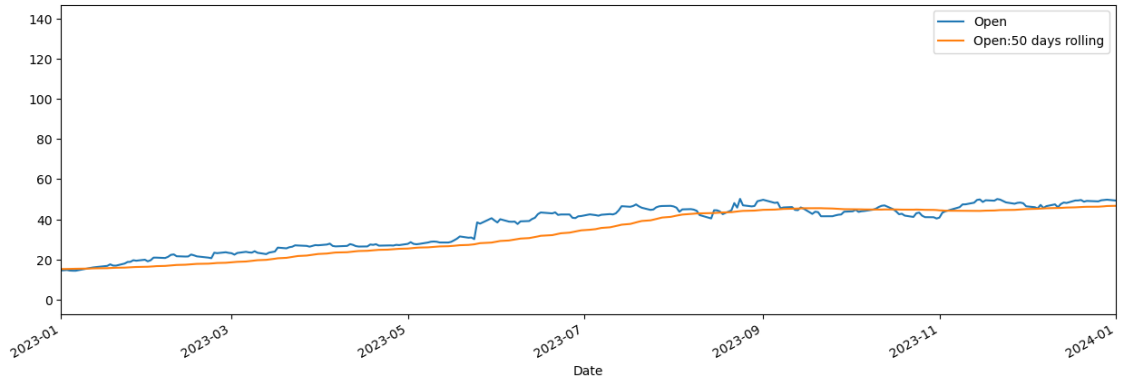
[88]:
```
Nvda_data[['Open','Open:100 days rolling']].plot(figsize=(15,5))
```

[88]: <Axes: xlabel='Date'>



[48]:
```
#plotting by focusing on data from 2023-2024
Nvda_data[['Open','Open:50 days rolling']].
 ↪plot(xlim=['2023-01-01','2024-01-01'],figsize=(15,5))
```

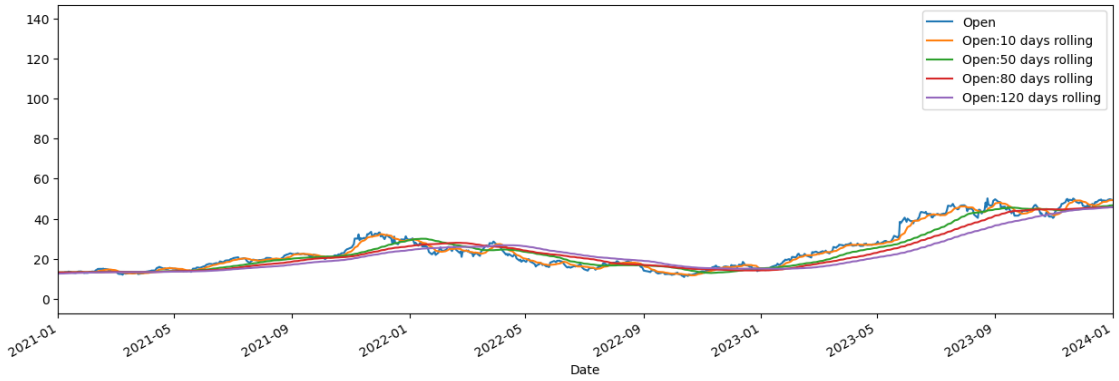
[48]: <Axes: xlabel='Date'>



[60]:
```
# rolling window 10,50,80,120 days
Nvda_data['Open:10 days rolling']=Nvda_data['Open'].
 ↪rolling(window=10,min_periods=2).mean()
Nvda_data['Open:50 days rolling']=Nvda_data['Open'].
 ↪rolling(window=50,min_periods=2).mean()
Nvda_data['Open:80 days rolling']=Nvda_data['Open'].
 ↪rolling(window=80,min_periods=2).mean()
Nvda_data['Open:120 days rolling']=Nvda_data['Open'].
 ↪rolling(window=120,min_periods=2).mean()
```

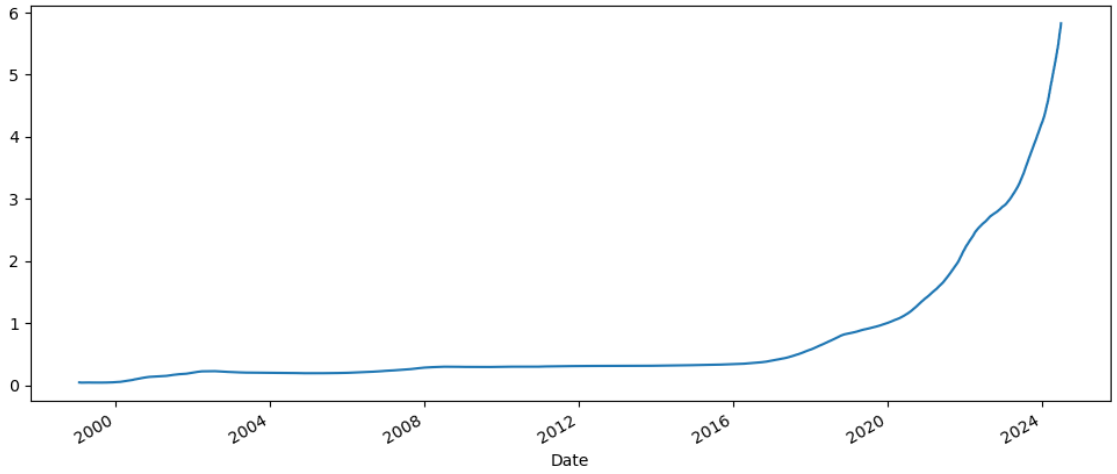
[64]:
```
#Plotting rolling window 10,50,80,120 days
Nvda_data[['Open','Open:10 days rolling','Open:50 days rolling','Open:80 days␣
 ↪rolling','Open:120 days rolling']].
 ↪plot(xlim=['2021-01-01','2024-01-01'],figsize=(15,5))
```

[64]: <Axes: xlabel='Date'>



[66]:
```
# expanding
#Cumulative moving average-CMA
Nvda_data['Open'].expanding().mean().plot(figsize=(12,5))
```

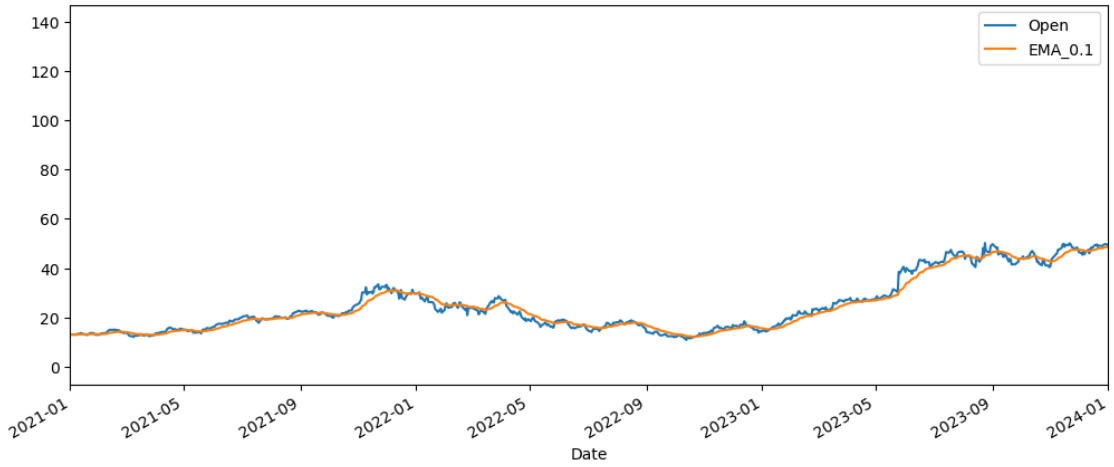
[66]: <Axes: xlabel='Date'>



[140]:
```
# EMA of Nvidia shares- Exponential moving average-Removing lags
# Smoothing factor - 0.1
Nvda_data['EMA_0.1']=Nvda_data['Open'].ewm(alpha=0.1,adjust=False).mean()
```

[108]:
```
#plotting EMA
Nvda_data[['Open','EMA_0.1']].
  ↪plot(xlim=['2021-01-01','2024-01-01'],figsize=(12,5))
```

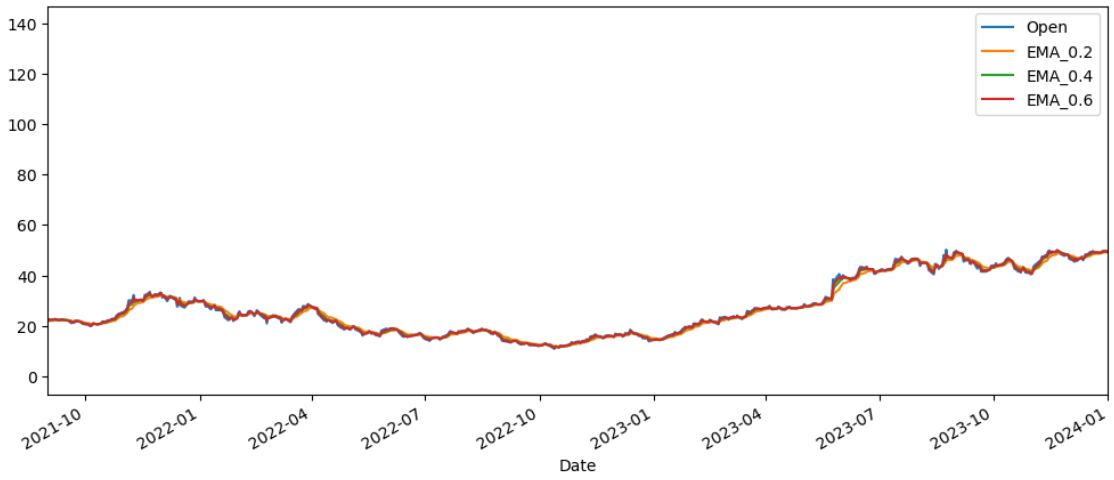
[108]: <Axes: xlabel='Date'>



[110]:
```
# Smoothing factor - 0.2,0.4,0.6
Nvda_data['EMA_0.2']=Nvda_data['Open'].ewm(alpha=0.2,adjust=False).mean()
Nvda_data['EMA_0.4']=Nvda_data['Open'].ewm(alpha=0.4,adjust=False).mean()
Nvda_data['EMA_0.6']=Nvda_data['Open'].ewm(alpha=0.6,adjust=False).mean()
```

[138]:
```
#plotting EMA -0.2,0.4,0.6
Nvda_data[['Open','EMA_0.2','EMA_0.4','EMA_0.6']].
  ↪plot(xlim=['2021-09-01','2024-01-01'],figsize=(12,5))
```

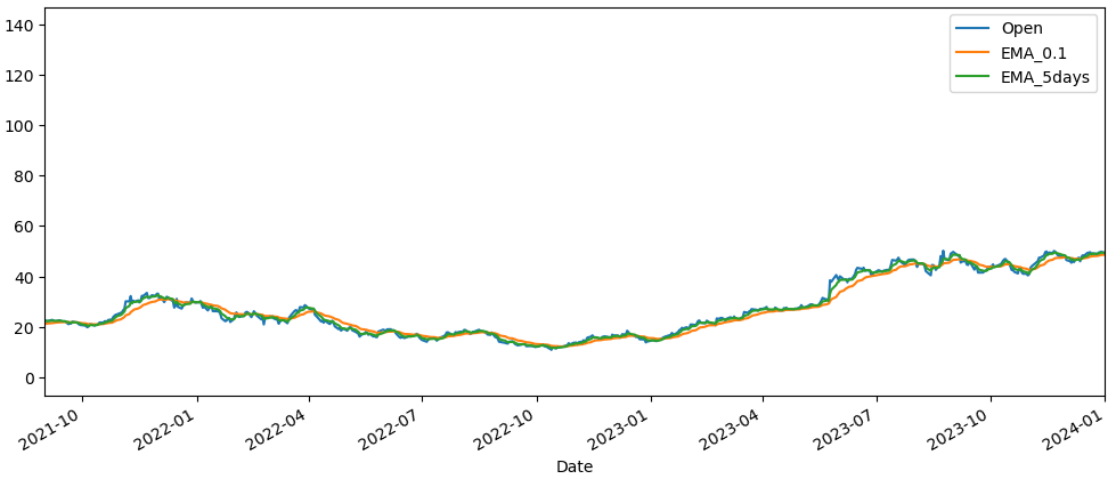
[138]: <Axes: xlabel='Date'>



[148]:
```
# Smoothing factor - 0.1,EMA_5days
Nvda_data['EMA_5days']=Nvda_data['Open'].ewm(span=5).mean()
```

[150]:
```
#plotting EMA -0.1, EMA_5days
Nvda_data[['Open','EMA_0.1','EMA_5days']].
 ↪plot(xlim=['2021-09-01','2024-01-01'],figsize=(12,5))
```

[150]: <Axes: xlabel='Date'>



0.12 ARIMA and SARIMA Models

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sms
import pandas as pd

%matplotlib inline
```

[163]:
```
df_airline=pd.read_csv('Downloads/airline_passengers.csv')
df_airline.head()
```

[163]:

	Month	Thousands of Passengers
0	1949-01	112.0
1	1949-02	118.0
2	1949-03	132.0
3	1949-04	129.0
4	1949-05	121.0

[165]:
```
df_airline.isnull().sum()
```

[165]:
```
Month                       0
Thousands of Passengers     1
dtype: int64
```

[167]:
```
df_airline.tail()
```

[167]:
```
                                                  Month  \
140                                             1960-09
141                                             1960-10
142                                             1960-11
143                                             1960-12
144  International airline passengers: monthly tota…

     Thousands of Passengers
140                    508.0
141                    461.0
142                    390.0
143                    432.0
144                      NaN
```

[169]:
```
df_airline.dropna(axis=0,inplace=True)
```

[171]:
```
df_airline.isnull().sum()
```

[171]:
```
Month                     0
Thousands of Passengers    0
dtype: int64
```

[173]:
```
df_airline.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 144 entries, 0 to 143
Data columns (total 2 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Month                    144 non-null    object
 1   Thousands of Passengers  144 non-null    float64
dtypes: float64(1), object(1)
memory usage: 3.4+ KB
```

[175]:
```
df_airline['Month']=pd.to_datetime(df_airline['Month'])
```

[177]:
```
df_airline.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 144 entries, 0 to 143
Data columns (total 2 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Month                    144 non-null    datetime64[ns]
 1   Thousands of Passengers  144 non-null    float64
```

```
dtypes: datetime64[ns](1), float64(1)
memory usage: 3.4 KB
```

[179]:
```
df_airline.head()
```

[179]:
```
        Month  Thousands of Passengers
0 1949-01-01                    112.0
1 1949-02-01                    118.0
2 1949-03-01                    132.0
3 1949-04-01                    129.0
4 1949-05-01                    121.0
```

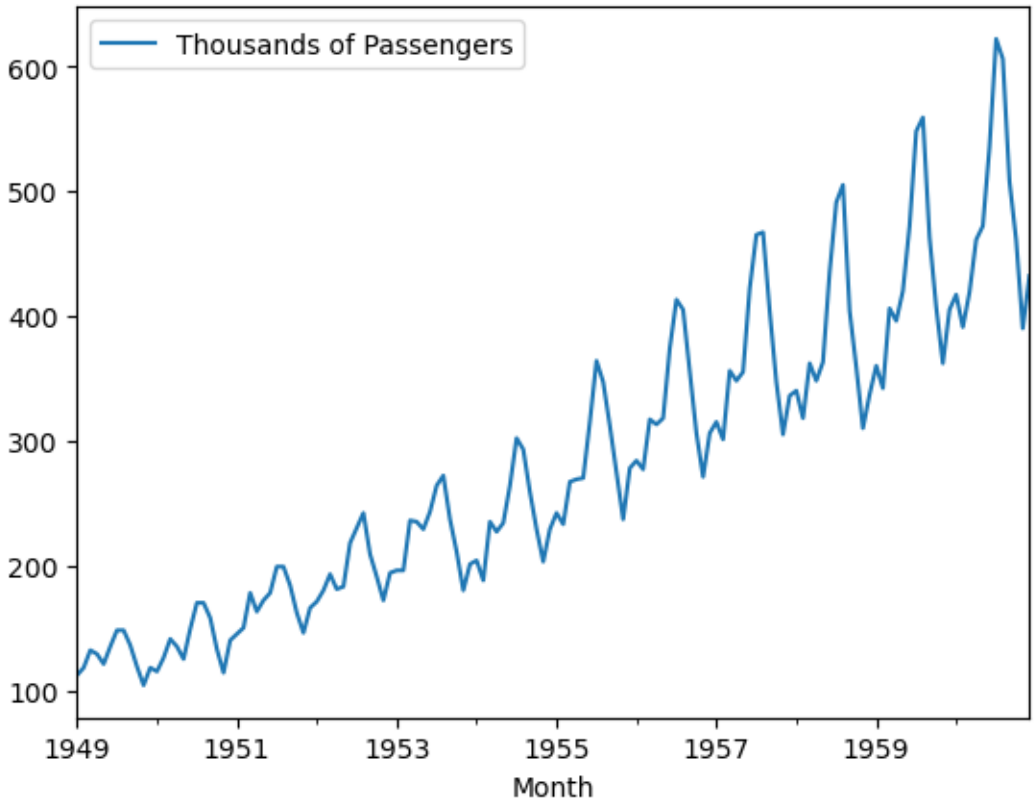
[185]:
```
df_airline.set_index('Month',inplace=True)
```

[187]:
```
df_airline.head()
```

[187]:
```
            Thousands of Passengers
Month
1949-01-01                    112.0
1949-02-01                    118.0
1949-03-01                    132.0
1949-04-01                    129.0
1949-05-01                    121.0
```

[189]:
```
df_airline.plot()
```

[189]:
```
<Axes: xlabel='Month'>
```



[191]:
```
from statsmodels.tsa.stattools import adfuller
```

[193]:
```
def adf_test(series):
    result=adfuller(series)
    print('ADF Statistics: {}'.format(result[0]))
    print('p- value: {}'.format(result[1]))
    if result[1] <= 0.05:
        print("strong evidence against the null hypothesis, reject the null␣
 ↪hypothesis. Data has no unit root and is stationary")
    else:
        print("weak evidence against null hypothesis, time series has a unit␣
 ↪root, indicating it is non-stationary ")
```

[195]:
```
adf_test(df_airline['Thousands of Passengers'])
```

```
ADF Statistics: 0.8153688792060597
p- value: 0.9918802434376411
weak evidence against null hypothesis, time series has a unit root, indicating
it is non-stationary
```

[197]:
```
# Techniques Differencing
df_airline['Passengers First Difference']=df_airline['Thousands of␣
 ↪Passengers']-df_airline['Thousands of Passengers'].shift(1)
```

[199]:
```
df_airline.head()
```

[199]:

Month	Thousands of Passengers	Passengers First Difference
1949-01-01	112.0	NaN
1949-02-01	118.0	6.0
1949-03-01	132.0	14.0
1949-04-01	129.0	-3.0
1949-05-01	121.0	-8.0

[201]:
```
adf_test(df_airline['Passengers First Difference'].dropna())
```

```
ADF Statistics: -2.829266824169992
p- value: 0.0542132902838265
weak evidence against null hypothesis, time series has a unit root, indicating
it is non-stationary
```

[203]:
```
# Techniques Differencing
df_airline['Passengers Second Difference']=df_airline['Passengers First␣
 ↪Difference']-df_airline['Passengers First Difference'].shift(1)
```

[205]:
```
adf_test(df_airline['Passengers Second Difference'].dropna())
```

```
ADF Statistics: -16.384231542468527
p- value: 2.732891850014085e-29
strong evidence against the null hypothesis, reject the null hypothesis. Data
has no unit root and is stationary
```

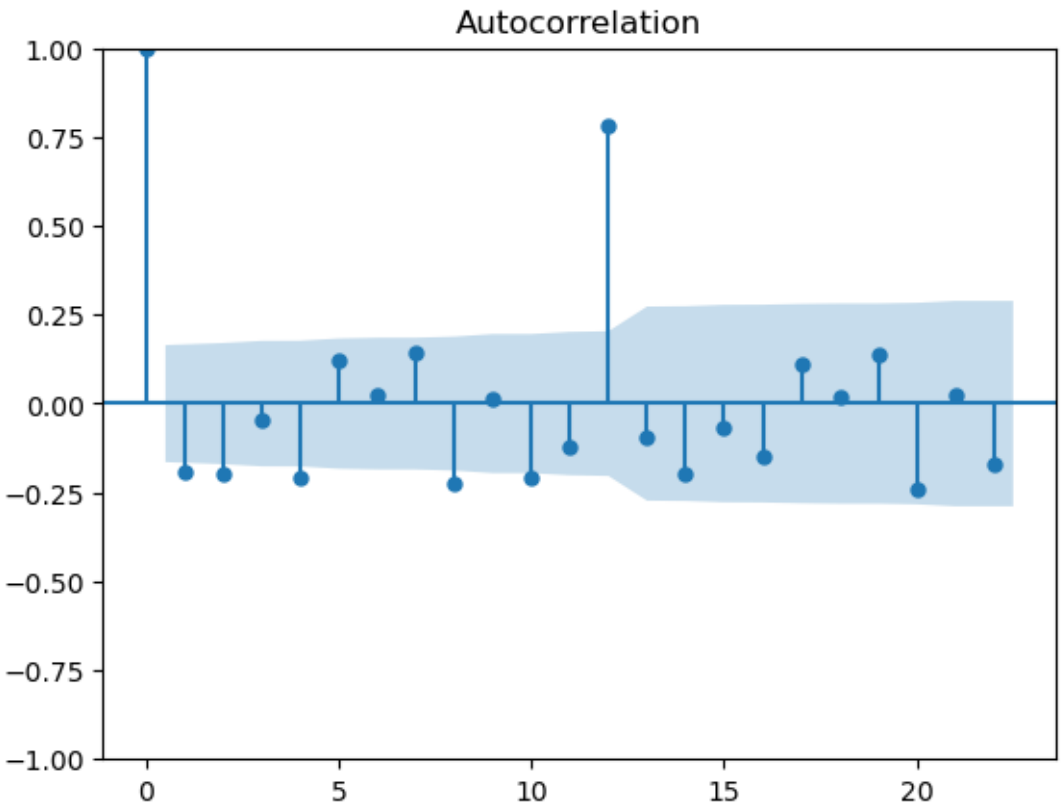
[207]:
```
# 12 months
# Techniques Differencing
df_airline['Passengers 12 Difference']=df_airline['Thousands of␣
 ↪Passengers']-df_airline['Thousands of Passengers'].shift(12)
```

[209]:
```
adf_test(df_airline['Passengers 12 Difference'].dropna())
```

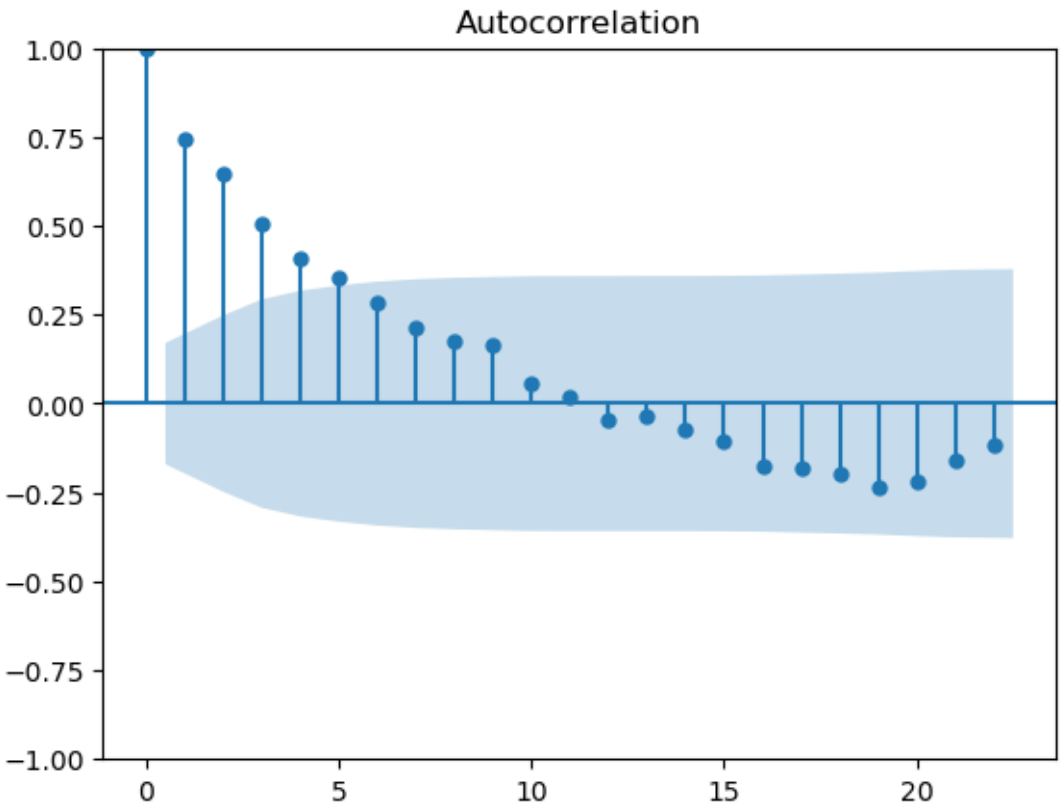
```
ADF Statistics: -3.3830207264924805
p- value: 0.011551493085514982
strong evidence against the null hypothesis, reject the null hypothesis. Data
has no unit root and is stationary
```

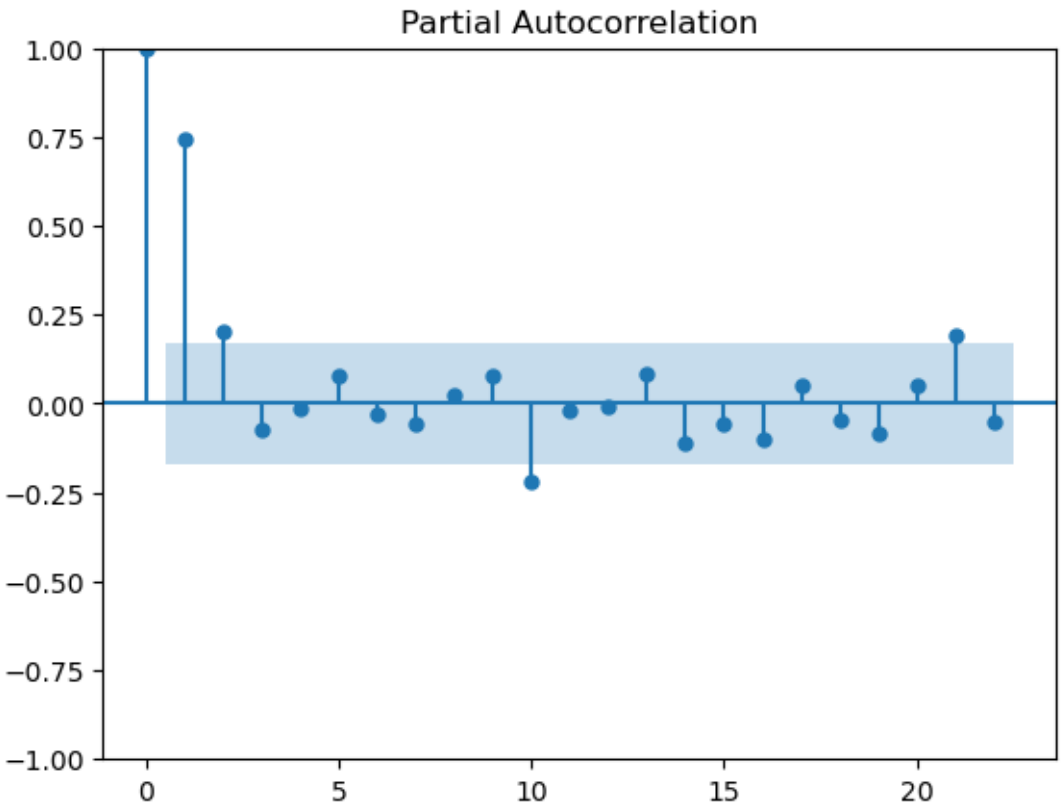
[211]:
```
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
```

[213]:
```
acf = plot_acf(df_airline["Passengers Second Difference"].dropna())
```

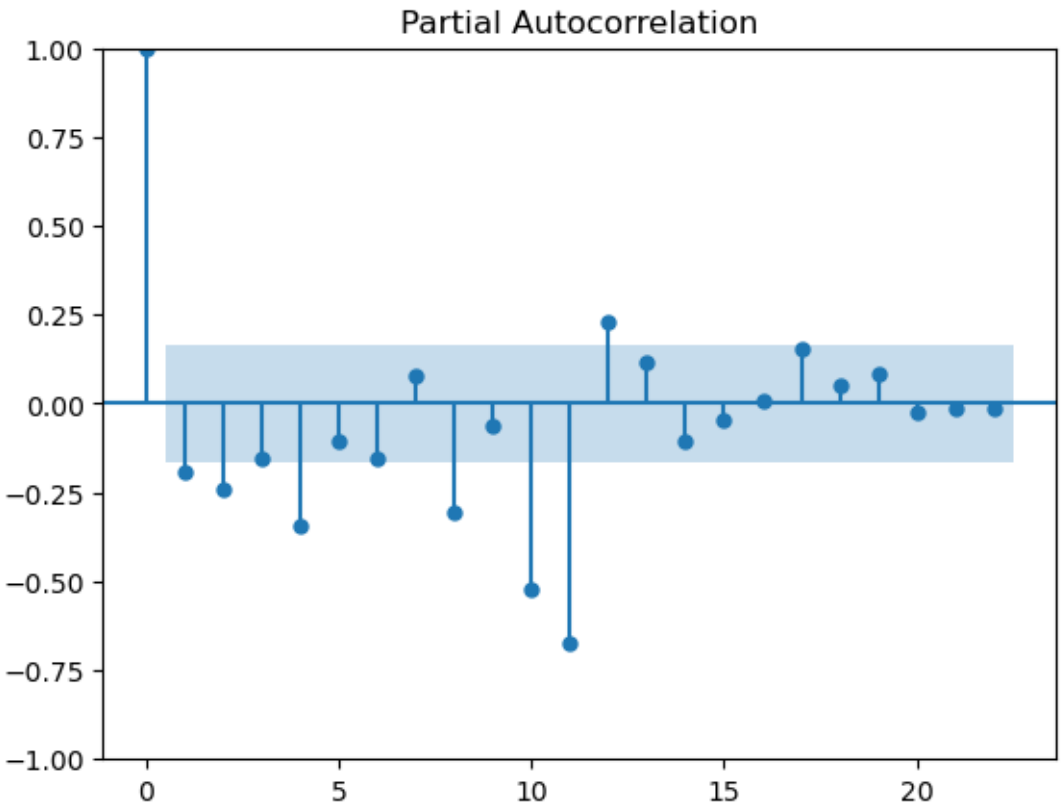


[222]:
```
acf12 = plot_acf(df_airline["Passengers 12 Difference"].dropna())
pacf12 = plot_pacf(df_airline["Passengers 12 Difference"].dropna())
```

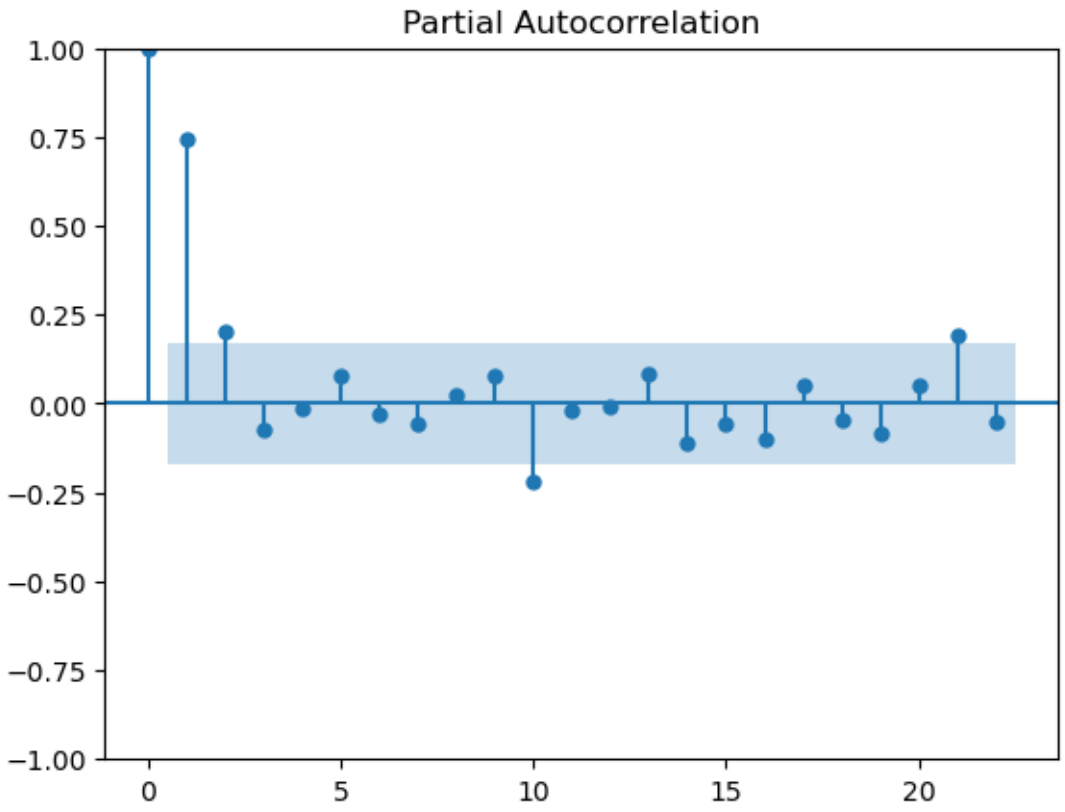

Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20



[224]:
```
result = plot_pacf(df_airline["Passengers Second Difference"].dropna())
```



[226]:
```
pacf12 = plot_pacf(df_airline["Passengers 12 Difference"].dropna())
```



[228]:
```
# split train and test data
df_airline
```

[228]:

Month	Thousands of Passengers	Passengers First Difference	Passengers Second Difference	Passengers 12 Difference
1949-01-01	112.0	NaN	NaN	NaN
1949-02-01	118.0	6.0	NaN	NaN
1949-03-01	132.0	14.0		
1949-04-01	129.0	-3.0		
1949-05-01	121.0	-8.0		
…	…	…		
1960-08-01	606.0	-16.0		
1960-09-01	508.0	-98.0		
1960-10-01	461.0	-47.0		
1960-11-01	390.0	-71.0		
1960-12-01	432.0	42.0		

```
1949-03-01                          8.0                         NaN
1949-04-01                        -17.0                         NaN
1949-05-01                         -5.0                         NaN
…                                   …                           …
1960-08-01                       -103.0                        47.0
1960-09-01                        -82.0                        45.0
1960-10-01                         51.0                        54.0
1960-11-01                        -24.0                        28.0
1960-12-01                        113.0                        27.0

[144 rows x 4 columns]
```

[230]:
```
from datetime import datetime,timedelta
train_dataset_end=datetime(1955,12,1)
test_dataset_end=datetime(1960,12,1)
```

[232]:
```
train_data=df_airline[:train_dataset_end]
test_data=df_airline[train_dataset_end+timedelta(days=1):test_dataset_end]
```

[234]:
```
##prediction
pred_start_date=test_data.index[0]
pred_end_date=test_data.index[-1]
```

[236]:
```
test_data
```

[236]:
```
            Thousands of Passengers  Passengers First Difference  \
Month
1956-01-01                    284.0                         6.0
1956-02-01                    277.0                        -7.0
1956-03-01                    317.0                        40.0
1956-04-01                    313.0                        -4.0
1956-05-01                    318.0                         5.0
1956-06-01                    374.0                        56.0
1956-07-01                    413.0                        39.0
1956-08-01                    405.0                        -8.0
1956-09-01                    355.0                       -50.0
1956-10-01                    306.0                       -49.0
1956-11-01                    271.0                       -35.0
1956-12-01                    306.0                        35.0
1957-01-01                    315.0                         9.0
1957-02-01                    301.0                       -14.0
1957-03-01                    356.0                        55.0
1957-04-01                    348.0                        -8.0
1957-05-01                    355.0                         7.0
1957-06-01                    422.0                        67.0
1957-07-01                    465.0                        43.0
1957-08-01                    467.0                         2.0
```

1957-09-01	404.0	-63.0
1957-10-01	347.0	-57.0
1957-11-01	305.0	-42.0
1957-12-01	336.0	31.0
1958-01-01	340.0	4.0
1958-02-01	318.0	-22.0
1958-03-01	362.0	44.0
1958-04-01	348.0	-14.0
1958-05-01	363.0	15.0
1958-06-01	435.0	72.0
1958-07-01	491.0	56.0
1958-08-01	505.0	14.0
1958-09-01	404.0	-101.0
1958-10-01	359.0	-45.0
1958-11-01	310.0	-49.0
1958-12-01	337.0	27.0
1959-01-01	360.0	23.0
1959-02-01	342.0	-18.0
1959-03-01	406.0	64.0
1959-04-01	396.0	-10.0
1959-05-01	420.0	24.0
1959-06-01	472.0	52.0
1959-07-01	548.0	76.0
1959-08-01	559.0	11.0
1959-09-01	463.0	-96.0
1959-10-01	407.0	-56.0
1959-11-01	362.0	-45.0
1959-12-01	405.0	43.0
1960-01-01	417.0	12.0
1960-02-01	391.0	-26.0
1960-03-01	419.0	28.0
1960-04-01	461.0	42.0
1960-05-01	472.0	11.0
1960-06-01	535.0	63.0
1960-07-01	622.0	87.0
1960-08-01	606.0	-16.0
1960-09-01	508.0	-98.0
1960-10-01	461.0	-47.0
1960-11-01	390.0	-71.0
1960-12-01	432.0	42.0

Month	Passengers Second Difference	Passengers 12 Difference
1956-01-01	-35.0	42.0
1956-02-01	-13.0	44.0
1956-03-01	47.0	50.0
1956-04-01	-44.0	44.0

1956-05-01	9.0	48.0
1956-06-01	51.0	59.0
1956-07-01	-17.0	49.0
1956-08-01	-47.0	58.0
1956-09-01	-42.0	43.0
1956-10-01	1.0	32.0
1956-11-01	14.0	34.0
1956-12-01	70.0	28.0
1957-01-01	-26.0	31.0
1957-02-01	-23.0	24.0
1957-03-01	69.0	39.0
1957-04-01	-63.0	35.0
1957-05-01	15.0	37.0
1957-06-01	60.0	48.0
1957-07-01	-24.0	52.0
1957-08-01	-41.0	62.0
1957-09-01	-65.0	49.0
1957-10-01	6.0	41.0
1957-11-01	15.0	34.0
1957-12-01	73.0	30.0
1958-01-01	-27.0	25.0
1958-02-01	-26.0	17.0
1958-03-01	66.0	6.0
1958-04-01	-58.0	0.0
1958-05-01	29.0	8.0
1958-06-01	57.0	13.0
1958-07-01	-16.0	26.0
1958-08-01	-42.0	38.0
1958-09-01	-115.0	0.0
1958-10-01	56.0	12.0
1958-11-01	-4.0	5.0
1958-12-01	76.0	1.0
1959-01-01	-4.0	20.0
1959-02-01	-41.0	24.0
1959-03-01	82.0	44.0
1959-04-01	-74.0	48.0
1959-05-01	34.0	57.0
1959-06-01	28.0	37.0
1959-07-01	24.0	57.0
1959-08-01	-65.0	54.0
1959-09-01	-107.0	59.0
1959-10-01	40.0	48.0
1959-11-01	11.0	52.0
1959-12-01	88.0	68.0
1960-01-01	-31.0	57.0
1960-02-01	-38.0	49.0
1960-03-01	54.0	13.0

```
1960-04-01                          14.0                          65.0
1960-05-01                         -31.0                          52.0
1960-06-01                          52.0                          63.0
1960-07-01                          24.0                          74.0
1960-08-01                        -103.0                          47.0
1960-09-01                         -82.0                          45.0
1960-10-01                          51.0                          54.0
1960-11-01                         -24.0                          28.0
1960-12-01                         113.0                          27.0
```

[264]:
```
# creating a ARIMA model
from statsmodels.tsa.arima.model import ARIMA
```

[240]:
```
train_data
```

[240]:
```
            Thousands of Passengers  Passengers First Difference  \
Month
1949-01-01                    112.0                          NaN
1949-02-01                    118.0                          6.0
1949-03-01                    132.0                         14.0
1949-04-01                    129.0                         -3.0
1949-05-01                    121.0                         -8.0
…                               …                            …
1955-08-01                    347.0                        -17.0
1955-09-01                    312.0                        -35.0
1955-10-01                    274.0                        -38.0
1955-11-01                    237.0                        -37.0
1955-12-01                    278.0                         41.0

            Passengers Second Difference  Passengers 12 Difference
Month
1949-01-01                           NaN                       NaN
1949-02-01                           NaN                       NaN
1949-03-01                           8.0                       NaN
1949-04-01                         -17.0                       NaN
1949-05-01                          -5.0                       NaN
…                                     …                         …
1955-08-01                         -66.0                      54.0
1955-09-01                         -18.0                      53.0
1955-10-01                          -3.0                      45.0
1955-11-01                           1.0                      34.0
1955-12-01                          78.0                      49.0

[84 rows x 4 columns]
```

[266]:
```
model_ARIMA = ARIMA(train_data['Thousands of Passengers'], order=(0,2,0))
results_ARIMA = model_ARIMA.fit()
```

```
print(results_ARIMA.summary())
```

```
C:\Users\ADMIN\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency
information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
C:\Users\ADMIN\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency
information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
C:\Users\ADMIN\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency
information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
                                SARIMAX Results
==============================================================================
===
Dep. Variable:     Thousands of Passengers   No. Observations:
84
Model:                       ARIMA(0, 2, 0)   Log Likelihood
-385.792
Date:                      Sat, 29 Jun 2024   AIC
773.584
Time:                              20:04:37   BIC
775.991
Sample:                          01-01-1949   HQIC
774.550
                               - 12-01-1955
Covariance Type:                         opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2       714.5858    102.414      6.977      0.000     513.858     915.314
==============================================================================
===
Ljung-Box (L1) (Q):                   4.59   Jarque-Bera (JB):
1.74
Prob(Q):                              0.03   Prob(JB):
0.42
Heteroskedasticity (H):               3.19   Skew:
0.31
Prob(H) (two-sided):                  0.00   Kurtosis:
3.36
==============================================================================
===

Warnings:
```

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

[268]:
```
model_Arima_fit=model_ARIMA.fit()
```

[270]:
```
model_Arima_fit.summary()
```

[270]:

Dep. Variable:	Thousands of Passengers	**No. Observations:**	84
Model:	ARIMA(0, 2, 0)	**Log Likelihood**	-385.792
Date:	Sat, 29 Jun 2024	**AIC**	773.584
Time:	20:05:36	**BIC**	775.991
Sample:	01-01-1949 - 12-01-1955	**HQIC**	774.550
Covariance Type:	opg		

	coef	**std err**	**z**	**P> |z|**	**[0.025**	**0.975]**
sigma2	714.5858	102.414	6.977	0.000	513.858	915.314

Ljung-Box (L1) (Q):	4.59	**Jarque-Bera (JB):**	1.74
Prob(Q):	0.03	**Prob(JB):**	0.42
Heteroskedasticity (H):	3.19	**Skew:**	0.31
Prob(H) (two-sided):	0.00	**Kurtosis:**	3.36

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

[272]:
```
test_data
```

[272]:

Month	Thousands of Passengers	Passengers First Difference
1956-01-01	284.0	6.0
1956-02-01	277.0	-7.0
1956-03-01	317.0	40.0
1956-04-01	313.0	-4.0
1956-05-01	318.0	5.0
1956-06-01	374.0	56.0
1956-07-01	413.0	39.0
1956-08-01	405.0	-8.0
1956-09-01	355.0	-50.0
1956-10-01	306.0	-49.0
1956-11-01	271.0	-35.0
1956-12-01	306.0	35.0
1957-01-01	315.0	9.0
1957-02-01	301.0	-14.0
1957-03-01	356.0	55.0
1957-04-01	348.0	-8.0
1957-05-01	355.0	7.0
1957-06-01	422.0	67.0
1957-07-01	465.0	43.0

1957-08-01	467.0	2.0
1957-09-01	404.0	-63.0
1957-10-01	347.0	-57.0
1957-11-01	305.0	-42.0
1957-12-01	336.0	31.0
1958-01-01	340.0	4.0
1958-02-01	318.0	-22.0
1958-03-01	362.0	44.0
1958-04-01	348.0	-14.0
1958-05-01	363.0	15.0
1958-06-01	435.0	72.0
1958-07-01	491.0	56.0
1958-08-01	505.0	14.0
1958-09-01	404.0	-101.0
1958-10-01	359.0	-45.0
1958-11-01	310.0	-49.0
1958-12-01	337.0	27.0
1959-01-01	360.0	23.0
1959-02-01	342.0	-18.0
1959-03-01	406.0	64.0
1959-04-01	396.0	-10.0
1959-05-01	420.0	24.0
1959-06-01	472.0	52.0
1959-07-01	548.0	76.0
1959-08-01	559.0	11.0
1959-09-01	463.0	-96.0
1959-10-01	407.0	-56.0
1959-11-01	362.0	-45.0
1959-12-01	405.0	43.0
1960-01-01	417.0	12.0
1960-02-01	391.0	-26.0
1960-03-01	419.0	28.0
1960-04-01	461.0	42.0
1960-05-01	472.0	11.0
1960-06-01	535.0	63.0
1960-07-01	622.0	87.0
1960-08-01	606.0	-16.0
1960-09-01	508.0	-98.0
1960-10-01	461.0	-47.0
1960-11-01	390.0	-71.0
1960-12-01	432.0	42.0

Month	Passengers Second Difference	Passengers 12 Difference
1956-01-01	-35.0	42.0
1956-02-01	-13.0	44.0
1956-03-01	47.0	50.0

1956-04-01	-44.0	44.0
1956-05-01	9.0	48.0
1956-06-01	51.0	59.0
1956-07-01	-17.0	49.0
1956-08-01	-47.0	58.0
1956-09-01	-42.0	43.0
1956-10-01	1.0	32.0
1956-11-01	14.0	34.0
1956-12-01	70.0	28.0
1957-01-01	-26.0	31.0
1957-02-01	-23.0	24.0
1957-03-01	69.0	39.0
1957-04-01	-63.0	35.0
1957-05-01	15.0	37.0
1957-06-01	60.0	48.0
1957-07-01	-24.0	52.0
1957-08-01	-41.0	62.0
1957-09-01	-65.0	49.0
1957-10-01	6.0	41.0
1957-11-01	15.0	34.0
1957-12-01	73.0	30.0
1958-01-01	-27.0	25.0
1958-02-01	-26.0	17.0
1958-03-01	66.0	6.0
1958-04-01	-58.0	0.0
1958-05-01	29.0	8.0
1958-06-01	57.0	13.0
1958-07-01	-16.0	26.0
1958-08-01	-42.0	38.0
1958-09-01	-115.0	0.0
1958-10-01	56.0	12.0
1958-11-01	-4.0	5.0
1958-12-01	76.0	1.0
1959-01-01	-4.0	20.0
1959-02-01	-41.0	24.0
1959-03-01	82.0	44.0
1959-04-01	-74.0	48.0
1959-05-01	34.0	57.0
1959-06-01	28.0	37.0
1959-07-01	24.0	57.0
1959-08-01	-65.0	54.0
1959-09-01	-107.0	59.0
1959-10-01	40.0	48.0
1959-11-01	11.0	52.0
1959-12-01	88.0	68.0
1960-01-01	-31.0	57.0
1960-02-01	-38.0	49.0

```
1960-03-01                    54.0                    13.0
1960-04-01                    14.0                    65.0
1960-05-01                   -31.0                    52.0
1960-06-01                    52.0                    63.0
1960-07-01                    24.0                    74.0
1960-08-01                  -103.0                    47.0
1960-09-01                   -82.0                    45.0
1960-10-01                    51.0                    54.0
1960-11-01                   -24.0                    28.0
1960-12-01                   113.0                    27.0
```

[274]:
```
##prediction
pred_start_date=test_data.index[0]
pred_end_date=test_data.index[-1]
print(pred_start_date)
print(pred_end_date)
```

```
1956-01-01 00:00:00
1960-12-01 00:00:00
```

[276]:
```
pred=model_Arima_fit.predict(start=pred_start_date,end=pred_end_date)
residuals=test_data['Thousands of Passengers']-pred
```

[278]:
```
pred
```

[278]:
```
1956-01-01     319.0
1956-02-01     360.0
1956-03-01     401.0
1956-04-01     442.0
1956-05-01     483.0
1956-06-01     524.0
1956-07-01     565.0
1956-08-01     606.0
1956-09-01     647.0
1956-10-01     688.0
1956-11-01     729.0
1956-12-01     770.0
1957-01-01     811.0
1957-02-01     852.0
1957-03-01     893.0
1957-04-01     934.0
1957-05-01     975.0
1957-06-01    1016.0
1957-07-01    1057.0
1957-08-01    1098.0
1957-09-01    1139.0
1957-10-01    1180.0
```

```
1957-11-01    1221.0
1957-12-01    1262.0
1958-01-01    1303.0
1958-02-01    1344.0
1958-03-01    1385.0
1958-04-01    1426.0
1958-05-01    1467.0
1958-06-01    1508.0
1958-07-01    1549.0
1958-08-01    1590.0
1958-09-01    1631.0
1958-10-01    1672.0
1958-11-01    1713.0
1958-12-01    1754.0
1959-01-01    1795.0
1959-02-01    1836.0
1959-03-01    1877.0
1959-04-01    1918.0
1959-05-01    1959.0
1959-06-01    2000.0
1959-07-01    2041.0
1959-08-01    2082.0
1959-09-01    2123.0
1959-10-01    2164.0
1959-11-01    2205.0
1959-12-01    2246.0
1960-01-01    2287.0
1960-02-01    2328.0
1960-03-01    2369.0
1960-04-01    2410.0
1960-05-01    2451.0
1960-06-01    2492.0
1960-07-01    2533.0
1960-08-01    2574.0
1960-09-01    2615.0
1960-10-01    2656.0
1960-11-01    2697.0
1960-12-01    2738.0
Freq: MS, Name: predicted_mean, dtype: float64
```

[280]:
```
residuals
```

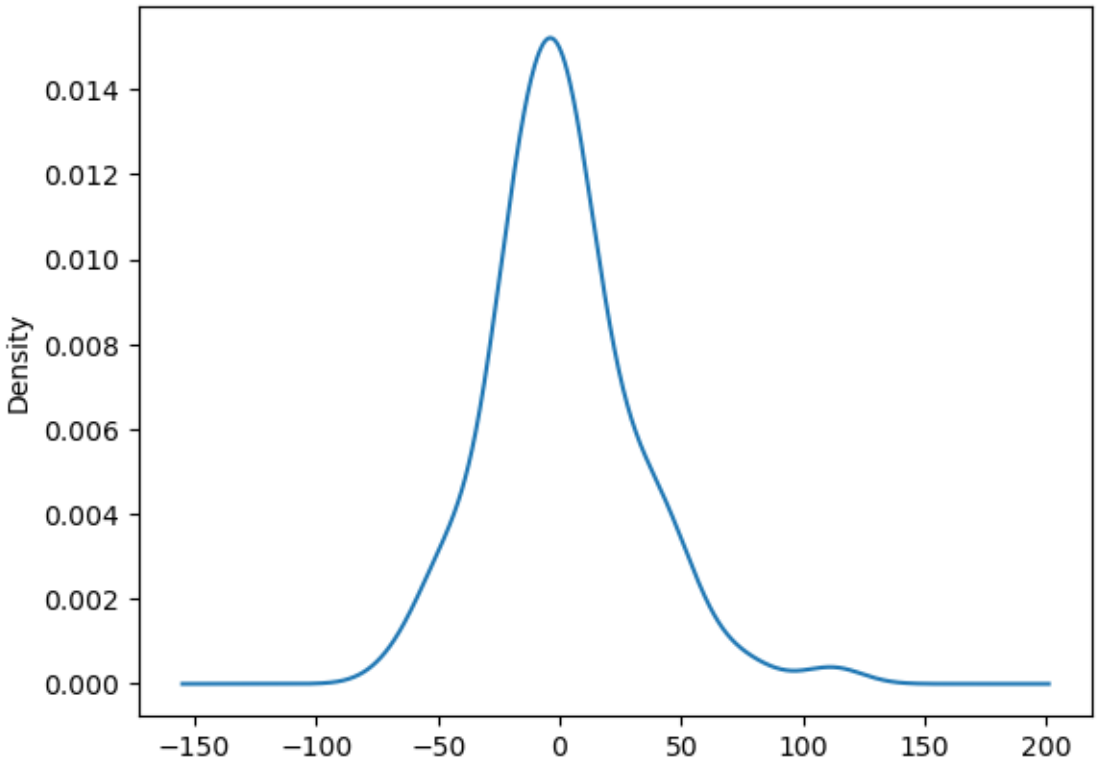
[280]:
```
Month
1956-01-01     -35.0
1956-02-01     -83.0
1956-03-01     -84.0
1956-04-01    -129.0
```

```
1956-05-01    -165.0
1956-06-01    -150.0
1956-07-01    -152.0
1956-08-01    -201.0
1956-09-01    -292.0
1956-10-01    -382.0
1956-11-01    -458.0
1956-12-01    -464.0
1957-01-01    -496.0
1957-02-01    -551.0
1957-03-01    -537.0
1957-04-01    -586.0
1957-05-01    -620.0
1957-06-01    -594.0
1957-07-01    -592.0
1957-08-01    -631.0
1957-09-01    -735.0
1957-10-01    -833.0
1957-11-01    -916.0
1957-12-01    -926.0
1958-01-01    -963.0
1958-02-01   -1026.0
1958-03-01   -1023.0
1958-04-01   -1078.0
1958-05-01   -1104.0
1958-06-01   -1073.0
1958-07-01   -1058.0
1958-08-01   -1085.0
1958-09-01   -1227.0
1958-10-01   -1313.0
1958-11-01   -1403.0
1958-12-01   -1417.0
1959-01-01   -1435.0
1959-02-01   -1494.0
1959-03-01   -1471.0
1959-04-01   -1522.0
1959-05-01   -1539.0
1959-06-01   -1528.0
1959-07-01   -1493.0
1959-08-01   -1523.0
1959-09-01   -1660.0
1959-10-01   -1757.0
1959-11-01   -1843.0
1959-12-01   -1841.0
1960-01-01   -1870.0
1960-02-01   -1937.0
1960-03-01   -1950.0
```

```
1960-04-01   -1949.0
1960-05-01   -1979.0
1960-06-01   -1957.0
1960-07-01   -1911.0
1960-08-01   -1968.0
1960-09-01   -2107.0
1960-10-01   -2195.0
1960-11-01   -2307.0
1960-12-01   -2306.0
dtype: float64
```

[282]:
```
model_Arima_fit.resid.plot(kind='kde')
```

[282]:
```
<Axes: ylabel='Density'>
```



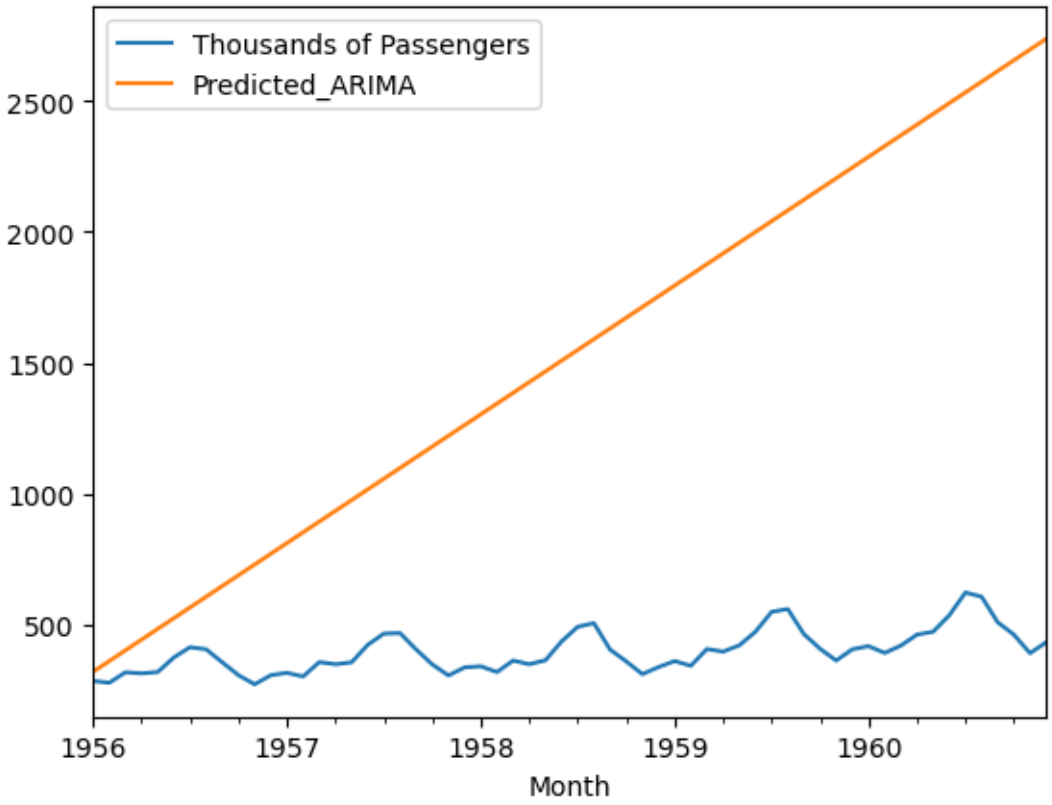
[284]:
```
test_data['Predicted_ARIMA']=pred
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_12624\95659616.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

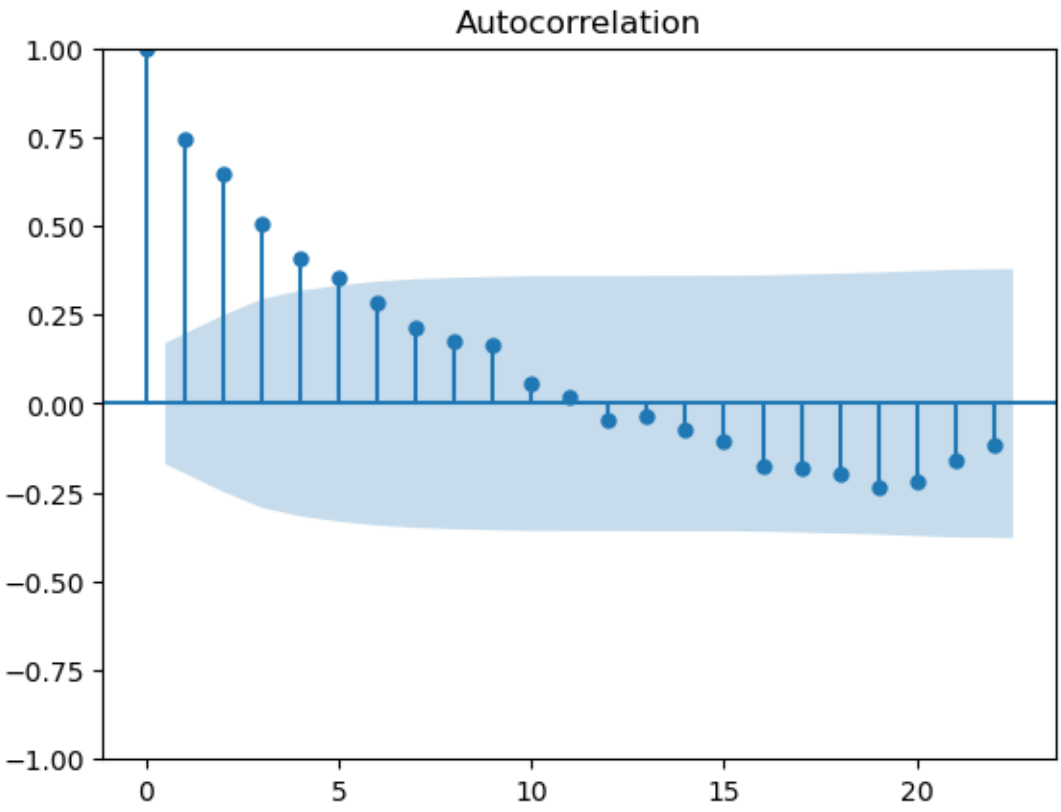
```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  test_data['Predicted_ARIMA']=pred
```

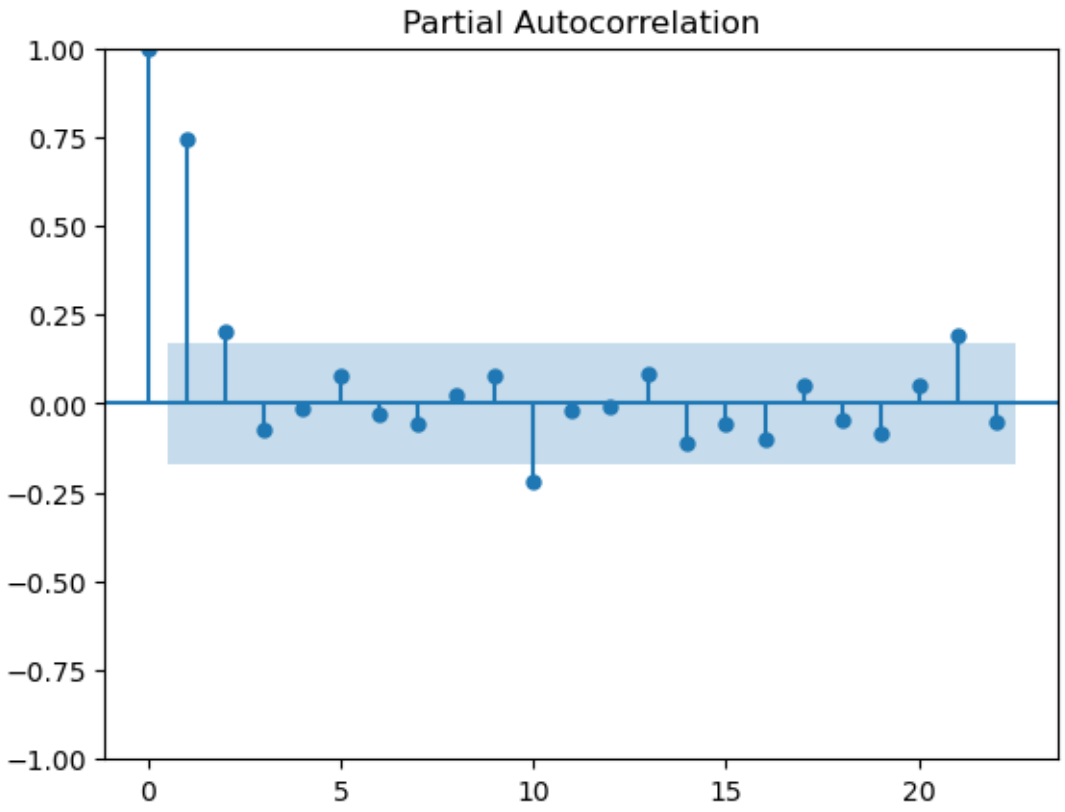
[292]:
```
test_data[['Thousands of Passengers','Predicted_ARIMA']].plot()
```

[292]: <Axes: xlabel='Month'>



[288]:
```
acf12 = plot_acf(df_airline["Passengers 12 Difference"].dropna())
pacf12 = plot_pacf(df_airline["Passengers 12 Difference"].dropna())
```


Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20



[296]:
```
## create a SARIMA model
from statsmodels.tsa.statespace.sarimax import SARIMAX
```

[298]:
```
model_SARIMA=SARIMAX(train_data['Thousands of␣
↪Passengers'],order=(3,0,5),seasonal_order=(0,1,0,12))
```

```
C:\Users\ADMIN\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency
information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
C:\Users\ADMIN\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency
information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
```

[302]:
```
model_SARIMA_fit=model_SARIMA.fit()
```

```
C:\Users\ADMIN\anaconda3\Lib\site-packages\statsmodels\base\model.py:607:
ConvergenceWarning: Maximum Likelihood optimization failed to converge. Check
mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

[304]:
```
model_SARIMA_fit.summary()
```

[304]:

Dep. Variable:	Thousands of Passengers	**No. Observations:**	84
Model:	SARIMAX(3, 0, 5)x(0, 1, [], 12)	**Log Likelihood**	-265.240
Date:	Sat, 29 Jun 2024	**AIC**	548.481
Time:	20:13:43	**BIC**	568.971
Sample:	01-01-1949 - 12-01-1955	**HQIC**	556.638
Covariance Type:	opg		

	coef	std err	z	P> |z|	[0.025	0.975]
ar.L1	0.5982	0.937	0.638	0.523	-1.239	2.435
ar.L2	0.8311	0.232	3.582	0.000	0.376	1.286
ar.L3	-0.4523	0.894	-0.506	0.613	-2.204	1.299
ma.L1	0.1839	1.164	0.158	0.874	-2.097	2.465
ma.L2	-0.5340	1.262	-0.423	0.672	-3.007	1.939
ma.L3	-0.0987	0.384	-0.257	0.797	-0.851	0.654
ma.L4	-0.1273	0.337	-0.377	0.706	-0.788	0.534
ma.L5	0.2471	0.357	0.693	0.488	-0.452	0.946
sigma2	87.7334	81.111	1.082	0.279	-71.240	246.707

Ljung-Box (L1) (Q):	0.02	**Jarque-Bera (JB):**	2.68
Prob(Q):	0.88	**Prob(JB):**	0.26
Heteroskedasticity (H):	2.05	**Skew:**	0.46
Prob(H) (two-sided):	0.09	**Kurtosis:**	2.77

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

[306]:
```
test_data.tail()
```

[306]:
```
            Thousands of Passengers  Passengers First Difference  \
Month
1960-08-01                    606.0                         -16.0
1960-09-01                    508.0                         -98.0
1960-10-01                    461.0                         -47.0
1960-11-01                    390.0                         -71.0
1960-12-01                    432.0                          42.0

            Passengers Second Difference  Passengers 12 Difference  \
Month
1960-08-01                        -103.0                      47.0
1960-09-01                         -82.0                      45.0
1960-10-01                          51.0                      54.0
1960-11-01                         -24.0                      28.0
1960-12-01                         113.0                      27.0

            Predicted_ARIMA
Month
```

```
1960-08-01            2574.0
1960-09-01            2615.0
1960-10-01            2656.0
1960-11-01            2697.0
1960-12-01            2738.0
```

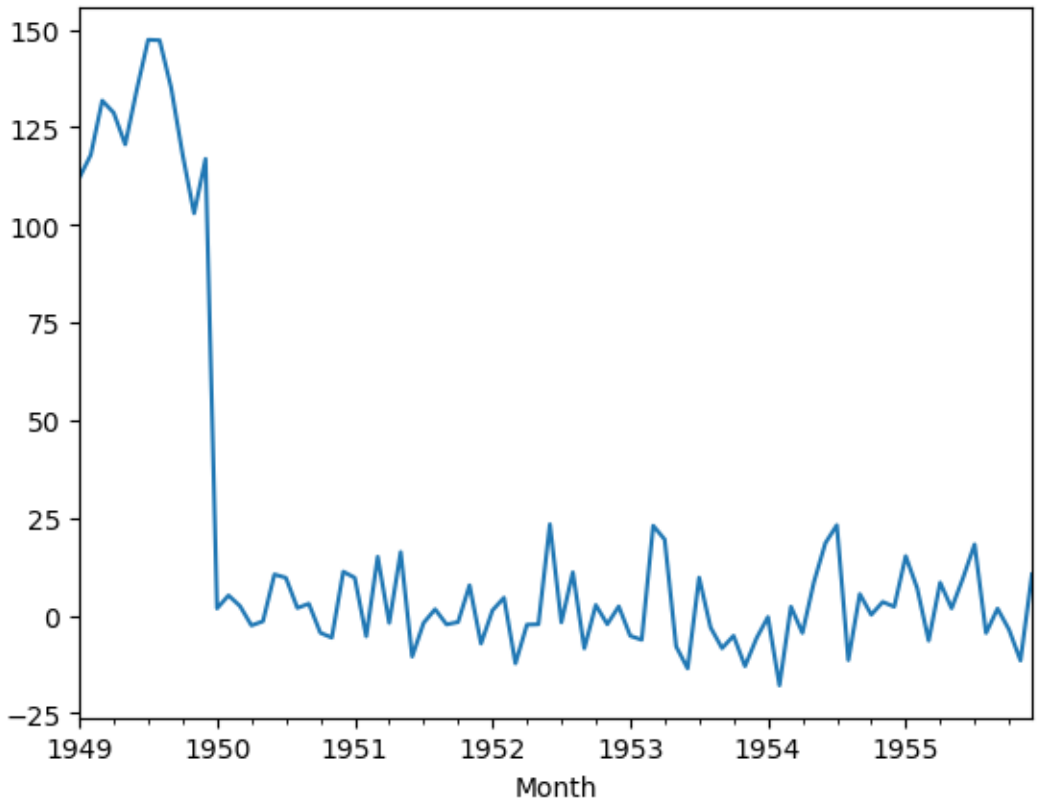
[308]:
```
##prediction
pred_start_date=test_data.index[0]
pred_end_date=test_data.index[-1]
print(pred_start_date)
print(pred_end_date)
```

```
1956-01-01 00:00:00
1960-12-01 00:00:00
```

[310]:
```
pred_Sarima=model_SARIMA_fit.
 ↪predict(start=datetime(1956,6,6),end=datetime(1960,12,1))
residuals=test_data['Thousands of Passengers']-pred_Sarima
```

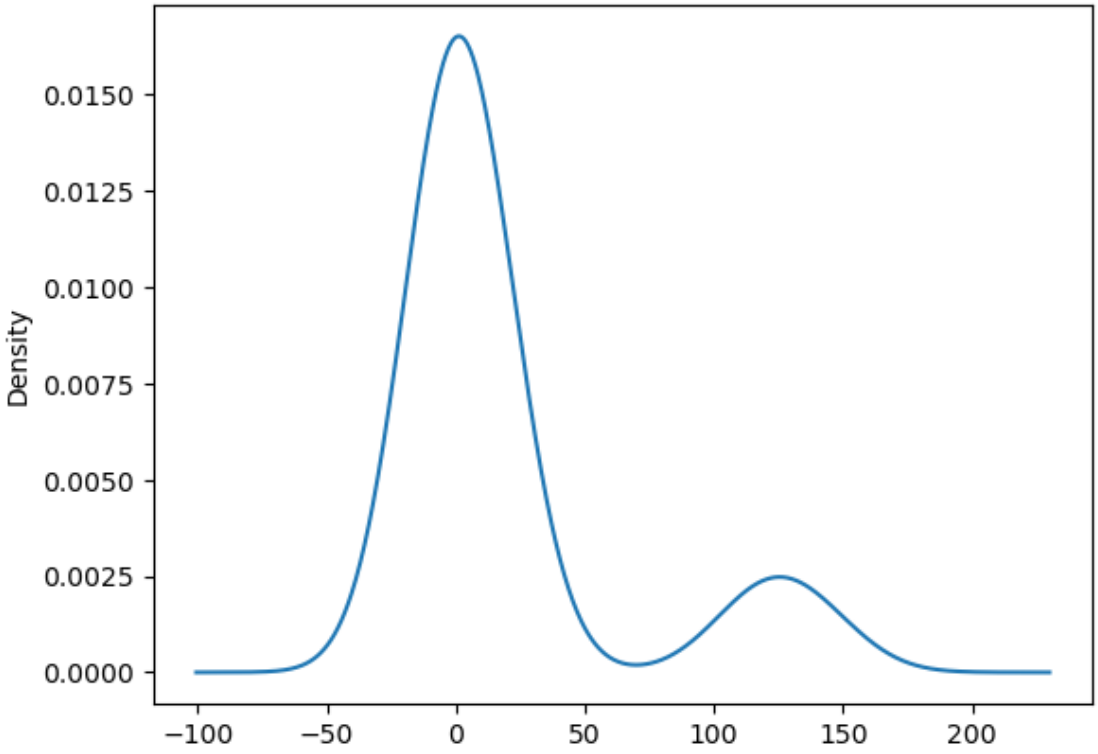
[312]:
```
model_SARIMA_fit.resid.plot()
```

[312]:
```
<Axes: xlabel='Month'>
```



[314]:
```
model_SARIMA_fit.resid.plot(kind='kde')
```

[314]: <Axes: ylabel='Density'>



[316]:
```
test_data['Predicted_SARIMA']=pred_Sarima
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_12624\1367177785.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  test_data['Predicted_SARIMA']=pred_Sarima
```

[318]:
```
test_data
```

[318]:

Month	Thousands of Passengers	Passengers First Difference	\
1956-01-01	284.0	6.0	
1956-02-01	277.0	-7.0	

1956-03-01	317.0	40.0
1956-04-01	313.0	-4.0
1956-05-01	318.0	5.0
1956-06-01	374.0	56.0
1956-07-01	413.0	39.0
1956-08-01	405.0	-8.0
1956-09-01	355.0	-50.0
1956-10-01	306.0	-49.0
1956-11-01	271.0	-35.0
1956-12-01	306.0	35.0
1957-01-01	315.0	9.0
1957-02-01	301.0	-14.0
1957-03-01	356.0	55.0
1957-04-01	348.0	-8.0
1957-05-01	355.0	7.0
1957-06-01	422.0	67.0
1957-07-01	465.0	43.0
1957-08-01	467.0	2.0
1957-09-01	404.0	-63.0
1957-10-01	347.0	-57.0
1957-11-01	305.0	-42.0
1957-12-01	336.0	31.0
1958-01-01	340.0	4.0
1958-02-01	318.0	-22.0
1958-03-01	362.0	44.0
1958-04-01	348.0	-14.0
1958-05-01	363.0	15.0
1958-06-01	435.0	72.0
1958-07-01	491.0	56.0
1958-08-01	505.0	14.0
1958-09-01	404.0	-101.0
1958-10-01	359.0	-45.0
1958-11-01	310.0	-49.0
1958-12-01	337.0	27.0
1959-01-01	360.0	23.0
1959-02-01	342.0	-18.0
1959-03-01	406.0	64.0
1959-04-01	396.0	-10.0
1959-05-01	420.0	24.0
1959-06-01	472.0	52.0
1959-07-01	548.0	76.0
1959-08-01	559.0	11.0
1959-09-01	463.0	-96.0
1959-10-01	407.0	-56.0
1959-11-01	362.0	-45.0
1959-12-01	405.0	43.0
1960-01-01	417.0	12.0

1960-02-01	391.0	-26.0
1960-03-01	419.0	28.0
1960-04-01	461.0	42.0
1960-05-01	472.0	11.0
1960-06-01	535.0	63.0
1960-07-01	622.0	87.0
1960-08-01	606.0	-16.0
1960-09-01	508.0	-98.0
1960-10-01	461.0	-47.0
1960-11-01	390.0	-71.0
1960-12-01	432.0	42.0

Month	Passengers Second Difference	Passengers 12 Difference	\
1956-01-01	-35.0	42.0	
1956-02-01	-13.0	44.0	
1956-03-01	47.0	50.0	
1956-04-01	-44.0	44.0	
1956-05-01	9.0	48.0	
1956-06-01	51.0	59.0	
1956-07-01	-17.0	49.0	
1956-08-01	-47.0	58.0	
1956-09-01	-42.0	43.0	
1956-10-01	1.0	32.0	
1956-11-01	14.0	34.0	
1956-12-01	70.0	28.0	
1957-01-01	-26.0	31.0	
1957-02-01	-23.0	24.0	
1957-03-01	69.0	39.0	
1957-04-01	-63.0	35.0	
1957-05-01	15.0	37.0	
1957-06-01	60.0	48.0	
1957-07-01	-24.0	52.0	
1957-08-01	-41.0	62.0	
1957-09-01	-65.0	49.0	
1957-10-01	6.0	41.0	
1957-11-01	15.0	34.0	
1957-12-01	73.0	30.0	
1958-01-01	-27.0	25.0	
1958-02-01	-26.0	17.0	
1958-03-01	66.0	6.0	
1958-04-01	-58.0	0.0	
1958-05-01	29.0	8.0	
1958-06-01	57.0	13.0	
1958-07-01	-16.0	26.0	
1958-08-01	-42.0	38.0	
1958-09-01	-115.0	0.0	

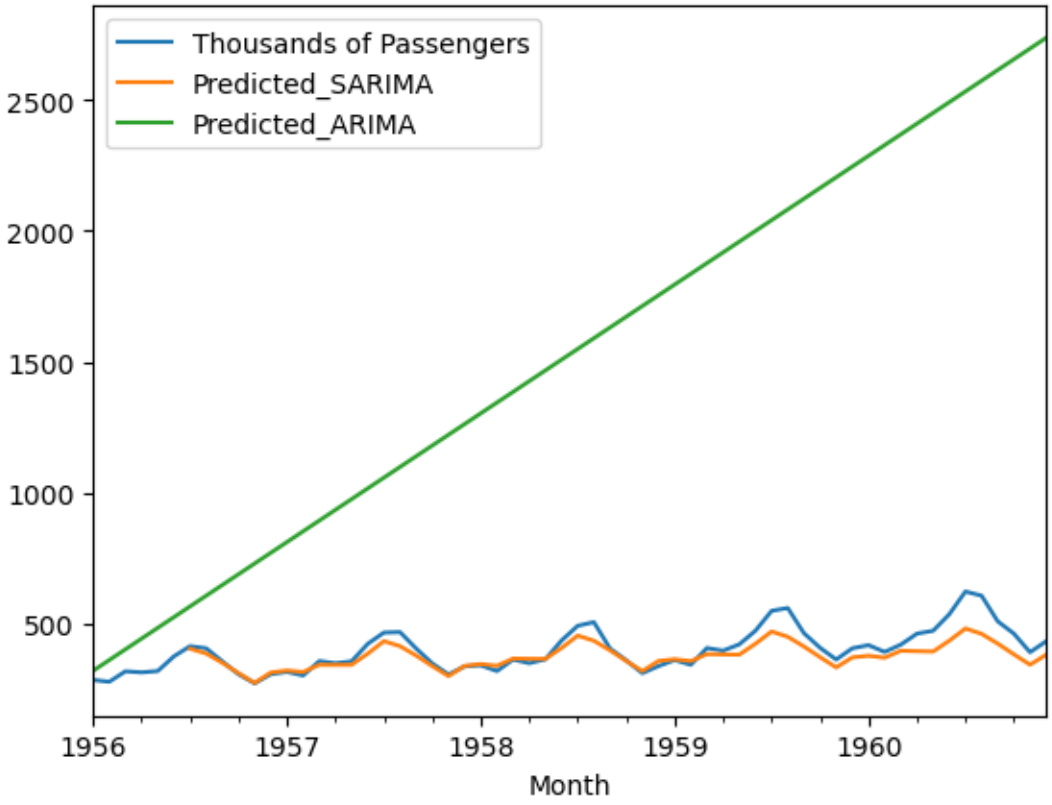
1958-10-01	56.0	12.0
1958-11-01	-4.0	5.0
1958-12-01	76.0	1.0
1959-01-01	-4.0	20.0
1959-02-01	-41.0	24.0
1959-03-01	82.0	44.0
1959-04-01	-74.0	48.0
1959-05-01	34.0	57.0
1959-06-01	28.0	37.0
1959-07-01	24.0	57.0
1959-08-01	-65.0	54.0
1959-09-01	-107.0	59.0
1959-10-01	40.0	48.0
1959-11-01	11.0	52.0
1959-12-01	88.0	68.0
1960-01-01	-31.0	57.0
1960-02-01	-38.0	49.0
1960-03-01	54.0	13.0
1960-04-01	14.0	65.0
1960-05-01	-31.0	52.0
1960-06-01	52.0	63.0
1960-07-01	24.0	74.0
1960-08-01	-103.0	47.0
1960-09-01	-82.0	45.0
1960-10-01	51.0	54.0
1960-11-01	-24.0	28.0
1960-12-01	113.0	27.0

Month	Predicted_ARIMA	Predicted_SARIMA
1956-01-01	319.0	NaN
1956-02-01	360.0	NaN
1956-03-01	401.0	NaN
1956-04-01	442.0	NaN
1956-05-01	483.0	NaN
1956-06-01	524.0	NaN
1956-07-01	565.0	403.397378
1956-08-01	606.0	385.070356
1956-09-01	647.0	349.444809
1956-10-01	688.0	310.217282
1956-11-01	729.0	272.563396
1956-12-01	770.0	312.435030
1957-01-01	811.0	319.956105
1957-02-01	852.0	314.010395
1957-03-01	893.0	341.987096
1957-04-01	934.0	341.787933
1957-05-01	975.0	341.792335

```
1957-06-01          1016.0       384.517759
1957-07-01          1057.0       432.324520
1957-08-01          1098.0       413.170720
1957-09-01          1139.0       376.919580
1957-10-01          1180.0       336.920924
1957-11-01          1221.0       298.659832
1957-12-01          1262.0       337.810356
1958-01-01          1303.0       344.744254
1958-02-01          1344.0       338.122666
1958-03-01          1385.0       365.533269
1958-04-01          1426.0       364.699369
1958-05-01          1467.0       364.159338
1958-06-01          1508.0       406.287644
1958-07-01          1549.0       453.571870
1958-08-01          1590.0       433.855514
1958-09-01          1631.0       397.103696
1958-10-01          1672.0       356.574378
1958-11-01          1713.0       317.834219
1958-12-01          1754.0       356.483626
1959-01-01          1795.0       362.959663
1959-02-01          1836.0       355.864423
1959-03-01          1877.0       382.837852
1959-04-01          1918.0       381.555910
1959-05-01          1959.0       380.598794
1959-06-01          2000.0       422.302998
1959-07-01          2041.0       469.189572
1959-08-01          2082.0       449.071549
1959-09-01          2123.0       411.940820
1959-10-01          2164.0       371.030900
1959-11-01          2205.0       331.929856
1959-12-01          2246.0       370.218475
1960-01-01          2287.0       376.350933
1960-02-01          2328.0       368.913569
1960-03-01          2369.0       395.560003
1960-04-01          2410.0       393.953541
1960-05-01          2451.0       392.685300
1960-06-01          2492.0       434.081606
1960-07-01          2533.0       480.672225
1960-08-01          2574.0       460.262014
1960-09-01          2615.0       422.849815
1960-10-01          2656.0       381.662566
1960-11-01          2697.0       342.293874
1960-12-01          2738.0       380.319227
```

[320]:
```
test_data[['Thousands of Passengers','Predicted_SARIMA','Predicted_ARIMA']].
↪plot()
```

[320]: <Axes: xlabel='Month'>



0.13 Thank you !