**DAV EXPERIMENT 5**

import pandas as pd

import statsmodels.api as sm

import matplotlib.pyplot as plt

from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

# Load the dataset (wine.csv in this case)

df = pd.read\_csv('wine.csv')

# Assuming you want to analyze the 'quality' column

data = df['quality']

# Plot the time series data

plt.figure(figsize=(12, 6))

plt.plot(data)

plt.title('Time Series Data')

plt.xlabel('Time')

plt.ylabel('Values')

plt.show()

# Autocorrelation Function (ACF) plot

plot\_acf(data, lags=40)

plt.title('Autocorrelation Function (ACF)')

plt.show()

# Partial Autocorrelation Function (PACF) plot

plot\_pacf(data, lags=40)

plt.title('Partial Autocorrelation Function (PACF)')

plt.show()

# Fit ARIMA model to determine p and q values

arima\_model = sm.tsa.ARIMA(data, order=(2, 1, 2))  # You may need to adjust the order

arima\_results = arima\_model.fit()

# Extract the values of p and q from the fitted ARIMA model

p\_value, d\_value, q\_value = arima\_results.model.order

print(f'Estimated p value: {p\_value}')

print(f'Estimated d value: {d\_value}')

print(f'Estimated q value: {q\_value}')

# Plot p and q in a graph

plt.bar(['p', 'q'], [p\_value, q\_value])

plt.title('Estimated p and q values')

plt.xlabel('Parameter')

plt.ylabel('Value')

plt.show()

# AR Model (using AutoReg)

ar\_model = sm.tsa.AutoReg(data, lags=2).fit()

ar\_predictions = ar\_model.predict(start=1, end=len(data))

plt.figure(figsize=(12, 6))

plt.plot(data, label='Original')

plt.plot(ar\_predictions, label='AR Model', linestyle='dashed')

plt.title('AR Model')

plt.xlabel('Time')

plt.ylabel('Values')

plt.legend()

plt.show()

# Moving Average (MA) Model

ma\_order = 2  # You can adjust the order

ma\_model = sm.tsa.SARIMAX(data, order=(0, 0, ma\_order), seasonal\_order=(0, 0, 0, 0)).fit()

ma\_predictions = ma\_model.predict(start=1, end=len(data))

plt.figure(figsize=(12, 6))

plt.plot(data, label='Original')

plt.plot(ma\_predictions, label=f'MA({ma\_order}) Model', linestyle='dashed')

plt.title(f'MA({ma\_order}) Model')

plt.xlabel('Time')

plt.ylabel('Values')

plt.legend()

plt.show()

# AutoRegressive Moving Average (ARMA) Model

arma\_order = (2, 1)  # You can adjust the order

arma\_model = sm.tsa.ARIMA(data, order=(arma\_order[0], 0, arma\_order[1])).fit()

arma\_predictions = arma\_model.predict(start=1, end=len(data))

plt.figure(figsize=(12, 6))

plt.plot(data, label='Original')

plt.plot(arma\_predictions, label=f'ARMA({arma\_order[0]}, 0, {arma\_order[1]}) Model', linestyle='dashed')

plt.title(f'ARMA({arma\_order[0]}, 0, {arma\_order[1]}) Model')

plt.xlabel('Time')

plt.ylabel('Values')

plt.legend()

plt.show()

# AutoRegressive Integrated Moving Average (ARIMA) Model

arima\_order = (2, 1, 2)  # You can adjust the order

arima\_model = sm.tsa.ARIMA(data, order=arima\_order).fit()

arima\_predictions = arima\_model.predict(start=1, end=len(data), typ='levels')

plt.figure(figsize=(12, 6))

plt.plot(data, label='Original')

plt.plot(arima\_predictions, label=f'ARIMA({arima\_order[0]}, {arima\_order[1]}, {arima\_order[2]}) Model', linestyle='dashed')

plt.title(f'ARIMA({arima\_order[0]}, {arima\_order[1]}, {arima\_order[2]}) Model')

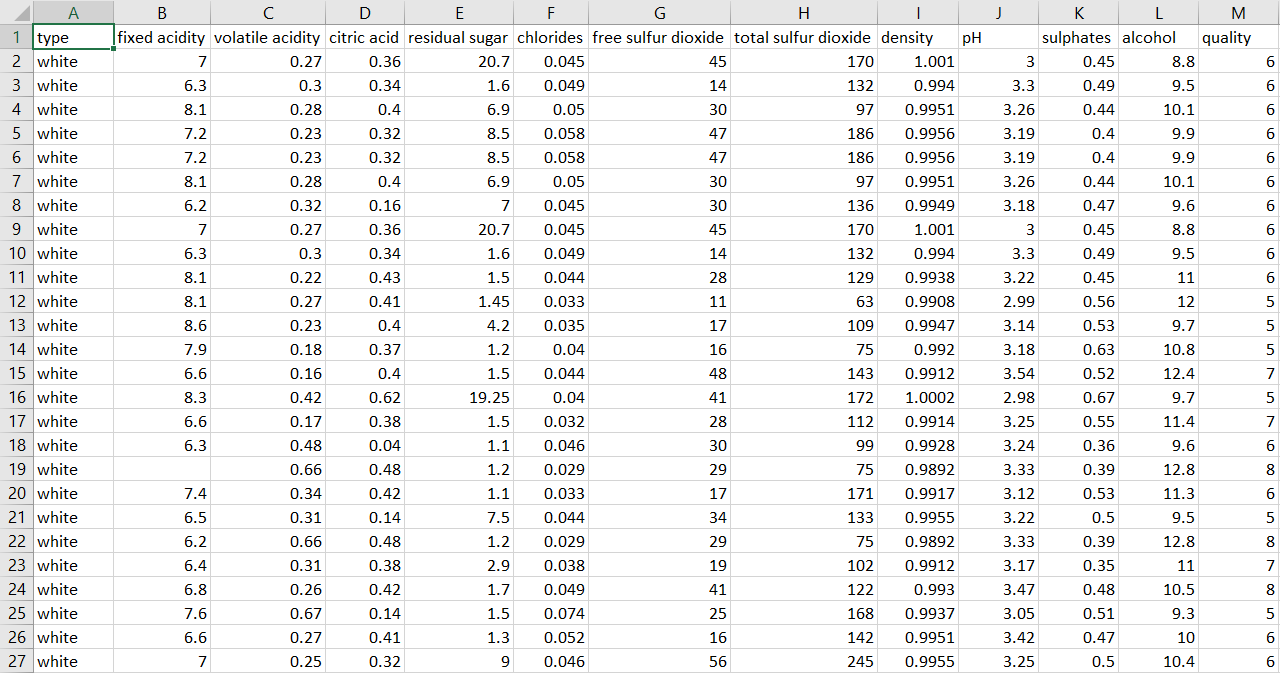
plt.xlabel('Time')

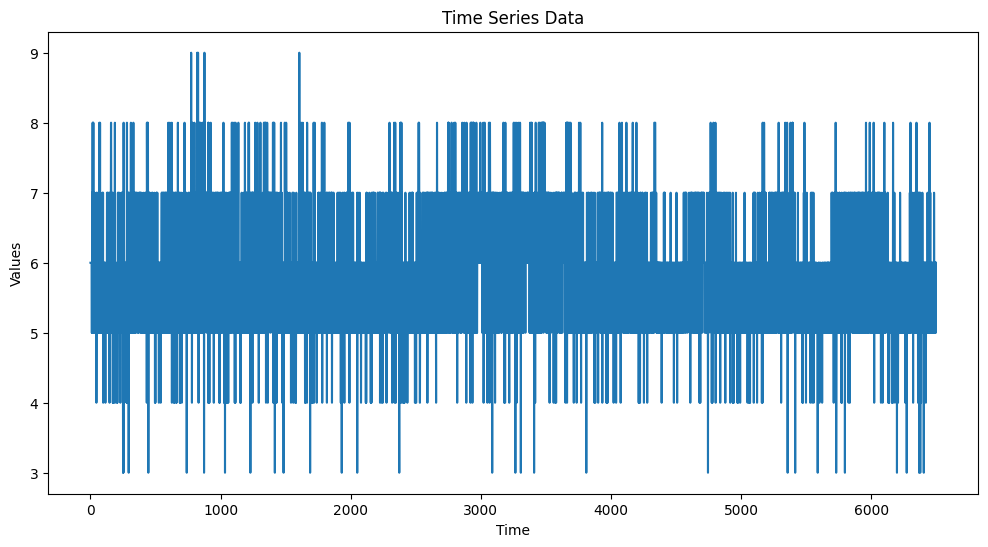
plt.ylabel('Values')

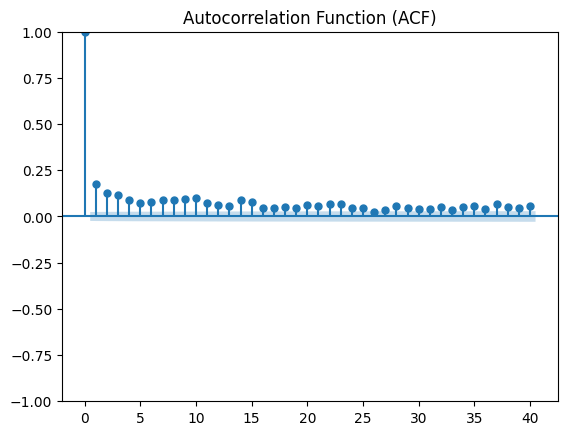
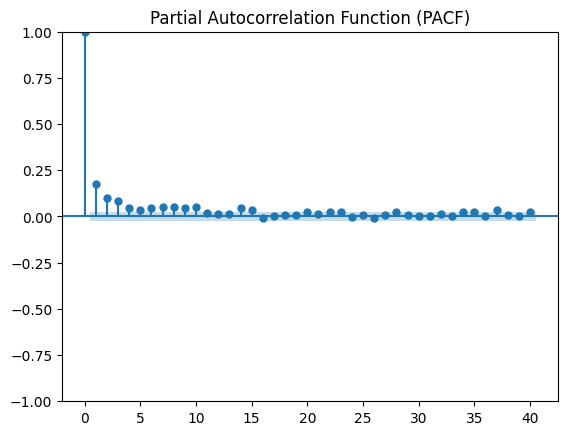
plt.legend()

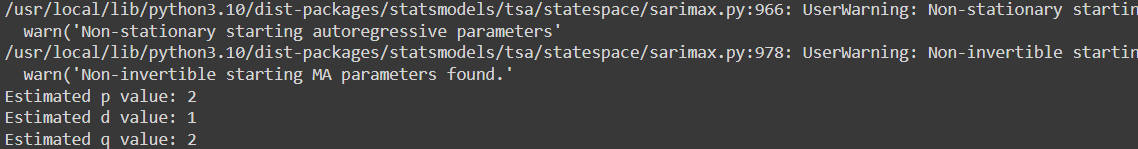
plt.show()

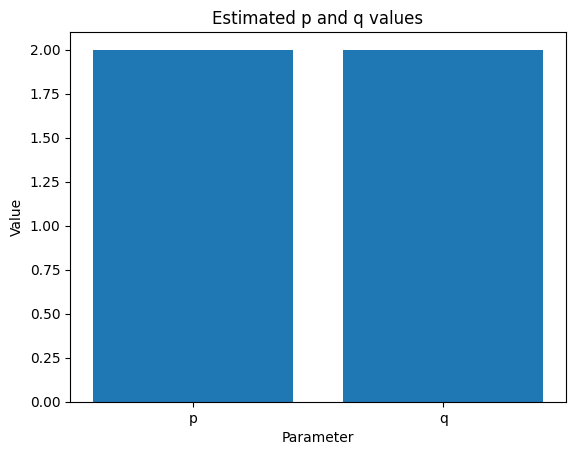
*wine.csv*

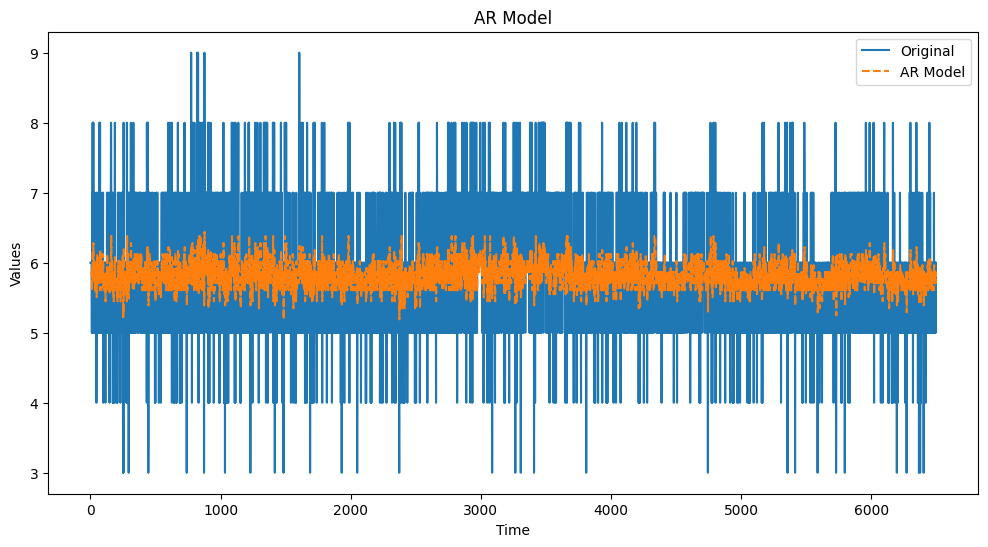


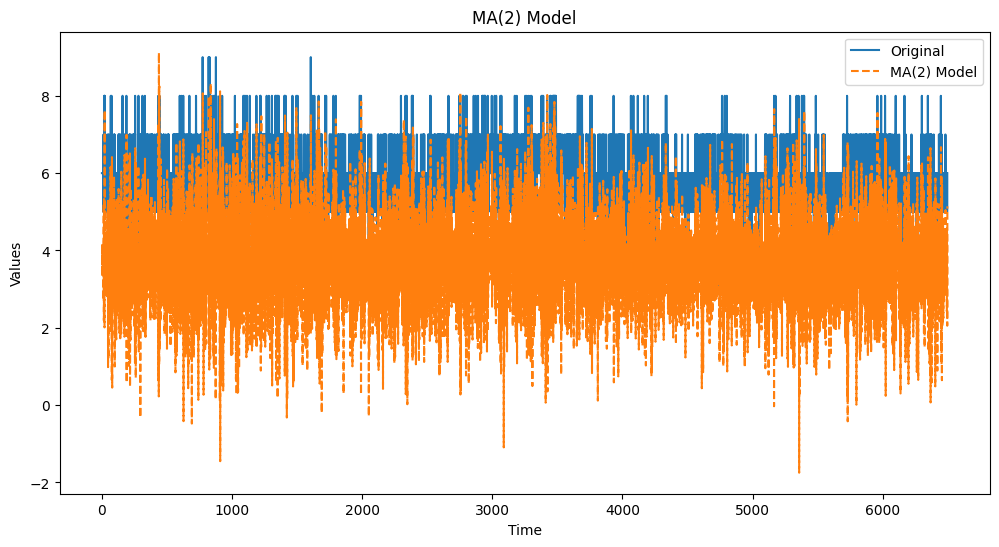


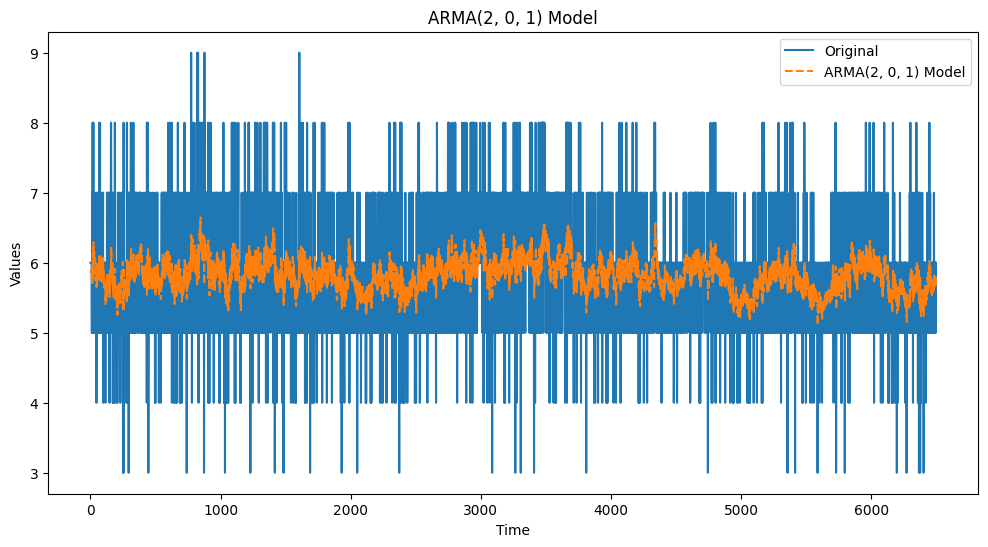
 

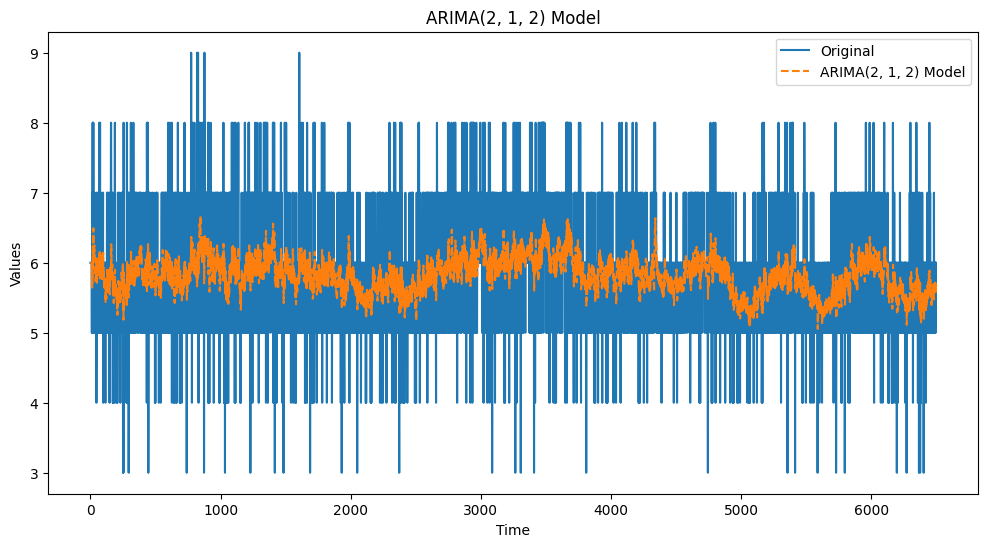












*Conclusion*: The ARMA model combines the autoregressive and moving average components to create a powerful tool for time series forecasting, enabling analysts to capture and predict patterns in data for informed decision-making.