import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.seasonal import seasonal\_decompose

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_percentage\_error

Импорт необходимых библиотек для работы с данными, графиками, моделями прогнозирования и оценкой метрик.

df = pd.read\_csv('AAP\_data.csv')

df.describe()

Чтение данных AAP\_data.csv и сохранение их в объекте DataFrame. Затем выводится описательная статистика

plt.plot(df.index, df['open'])

plt.plot(df['high'], color='yellow')

plt.plot(df.index, df['low'], color='black')

plt.show()

Построение графиков цен акций: open, high и low с использованием функции plot

df['date'] = pd.to\_datetime(df['date'])

df = df.set\_index('date')

Преобразование столбца date в формат даты с помощью функции to\_datetime. Затем столбец date используется в качестве индекса

df = df.resample('D').last().ffill()

Ресемплирование данных по ежедневному интервалу с помощью метода resample и параметра D (день). В случае, если в дне отсутствуют данные, используется последнее доступное значение с помощью метода last. Пропущенные значения заполняются предыдущими значениями с помощью метода ffill.

print(df.isna().sum())

Проверка наличия пропущенных значений в DataFrame с помощью метода isna для выявления пропущенных значений, а затем метода sum для подсчета количества пропущенных значений в каждом столбце

df['average'] = (df['open'] + df['high'] + df['low'] + df['close']) / 4

Вычисление средней цены акций путем сложения значений столбцов open, high, low и close и деления на 4. Результат сохраняется встолбце average.

print(df.head())

Вывод первых 5 строк DataFrame с помощью метода head для проверки результатов.

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

plt.plot(df.index, df['average'])

plt.xlabel('Дата')

plt.ylabel('Средняя цена акций')

plt.show()

Создание нового графика с помощью функции figure с указанием размера фигуры. Затем строится график изменения средней цены акций с использованием метода plot и указанием осей x (даты) и y (средняя цена акций). График отображается с помощью функции show.

plt.plot(df['high'])

plt.show()

Построение графика изменения максимальной цены акций с использованием метода plot Затем график отображается с помощью функции show.

plt.plot(df['low'])

plt.show()

Построение графика изменения минимальной цены акций с использованием метода plot Затем график отображается с помощью функции show.

plt.plot(df['volume'])

plt.show()

Построение графика изменения объема акций с использованием метода plot. Затем график отображается с помощью функции show.

df\_tra = pd.DataFrame(index=df.index)

df\_tra['volume'] = df.volume

df\_tra

Создание нового DataFrame df\_tra с индексом, идентичным исходному, а затем копирование столбца volume в новый DataFrame.

import numpy as np

df\_tra.volume = np.log(df\_tra['volume'])

Применение натурального логарифма к столбцу volume с помощью функции log

plt.plot(df\_tra)

plt.show()

Построение графика изменения столбца volume в DataFrame df\_tra с помощью метода plot. График отображается с помощью функции show.

from statsmodels.tsa.arima.model import ARIMA

model = ARIMA(df\_tra, order=(2, 1, 2))

Импорт модели ARIMA из библиотеки statsmodels.tsa.arima.model. Создание объекта модели ARIMA с указанным порядком (2, 1, 2).

model\_fit = model.fit()  
Обучение модели ARIMA на данных с помощью метода fit.