Financial Data Analysis Basics

Data Cleaning and Preparation:

```
import pandas as pd


# Display the first few rows of the dataset to understand its structure and variables

data.head()


# Data Cleaning and Preparation


# Check for missing values

missing_values = data.isnull().sum()


# Check for duplicates

duplicates = data.duplicated().sum()


# Formatting the 'Date' column to datetime format

data['Date'] = pd.to_datetime(data['Date'])


# Summary of checks

missing_values, duplicates
```

```
Date    Open    High    Low   Close   Adj Close    Volume

0   2017-01-03   15.11   15.30   15.03   15.26   12.826453   7462400

1   2017-01-04   15.80   15.98   15.60   15.96   13.414824   7671900

2   2017-01-05   15.68   15.83   15.44   15.72   13.213097   5451100

3   2017-01-06   15.58   15.76   15.56   15.68   13.179476   4017000

4   2017-01-09   15.52   15.72   15.44   15.61   13.120638   2806300


(Date         0

 Open         0

 High         0

 Low          0

 Close        0
```

```
Adj Close    0
Volume       0
dtype: int64,
0)
```

Descriptive Statistics:

```
# Descriptive Statistics

# Calculate basic statistics for the numerical columns
descriptive_stats = data.describe()

# Calculate variance separately as 'describe()' does not include it by default
variance = data.var()

# Adding variance to the descriptive statistics summary
descriptive_stats.loc['variance'] = variance

descriptive_stats
```

```
Open            High          Low         Close     Adj Close   \
count     1576.000000  1576.000000  1576.000000  1576.000000  1576.000000
mean        11.465127    11.556041    11.365539    11.458293    10.751646
std          3.932940     3.935829     3.924109     3.928854     3.439681
min          0.820000     0.860000     0.820000     0.850000     0.850000
25%          9.717500     9.797500     9.640000     9.740000     9.540773
50%         11.760000    11.890000    11.710000    11.800000    11.078288
75%         14.642500    14.720000    14.512500    14.602500    13.006490
max         19.799999    19.980000    19.780001    19.980000    18.228960
variance    15.468018    15.490753    15.398628    15.435896    11.831406

                Volume
count     1.576000e+03
mean      7.180016e+06
std       1.826503e+07
min       4.141000e+05
```

```
25%        2.260050e+06
50%        3.399850e+06
75%        6.596300e+06
max        4.341040e+08
variance   3.336114e+14
```

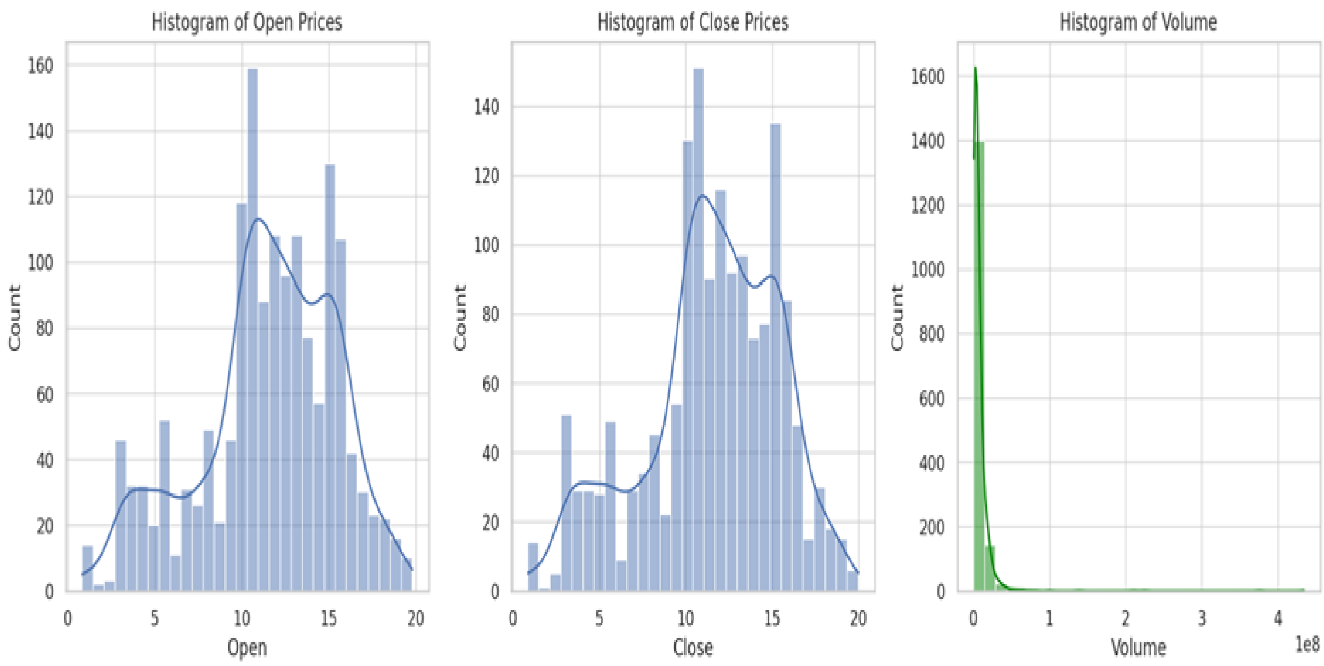
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Set the aesthetics for the plots
sns.set(style="whitegrid")
```

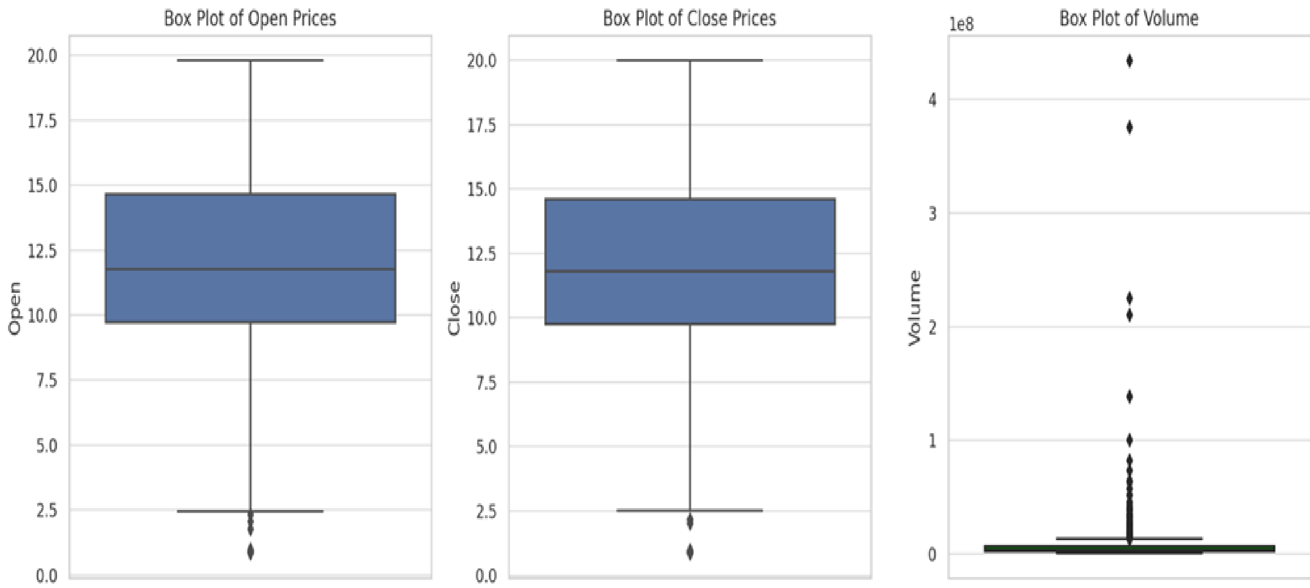
```python
# Creating histograms for 'Open', 'Close', and 'Volume'
plt.figure(figsize=(15, 5))

# Histogram for 'Open' Prices
plt.subplot(1, 3, 1)
sns.histplot(data['Open'], bins=30, kde=True)
plt.title('Histogram of Open Prices')
```

```python
# Histogram for 'Close' Prices
plt.subplot(1, 3, 2)
sns.histplot(data['Close'], bins=30, kde=True)
plt.title('Histogram of Close Prices')

# Histogram for 'Volume'
plt.subplot(1, 3, 3)
sns.histplot(data['Volume'], bins=30, kde=True, color='green')
plt.title('Histogram of Volume')

plt.tight_layout()
plt.show()
```

```python
# Creating box plots for 'Open', 'Close', and 'Volume'
```

```
plt.figure(figsize=(15, 5))


# Box plot for 'Open' Prices
plt.subplot(1, 3, 1)
sns.boxplot(y=data['Open'])
plt.title('Box Plot of Open Prices')

# Box plot for 'Close' Prices
plt.subplot(1, 3, 2)
sns.boxplot(y=data['Close'])
plt.title('Box Plot of Close Prices')

# Box plot for 'Volume'
plt.subplot(1, 3, 3)
sns.boxplot(y=data['Volume'], color='green')
plt.title('Box Plot of Volume')


plt.tight_layout()
plt.show()
```





The visualisations provide a deeper understanding of the data distribution:

Histograms:

Open and Close Prices: Both histograms exhibit a somewhat bell-shaped distribution, indicating that most stock prices are concentrated around the mean. There is slight skewness towards lower prices, suggesting that there are more days with stock prices below the average.

Volume: The histogram shows a right-skewed distribution, indicating that high trading volumes are less frequent but can reach significantly high levels.

Box Plots:

Open and Close Prices: The box plots for Open and Close prices display a relatively symmetrical distribution around the median, with a few outliers indicating days with exceptionally high or low prices.

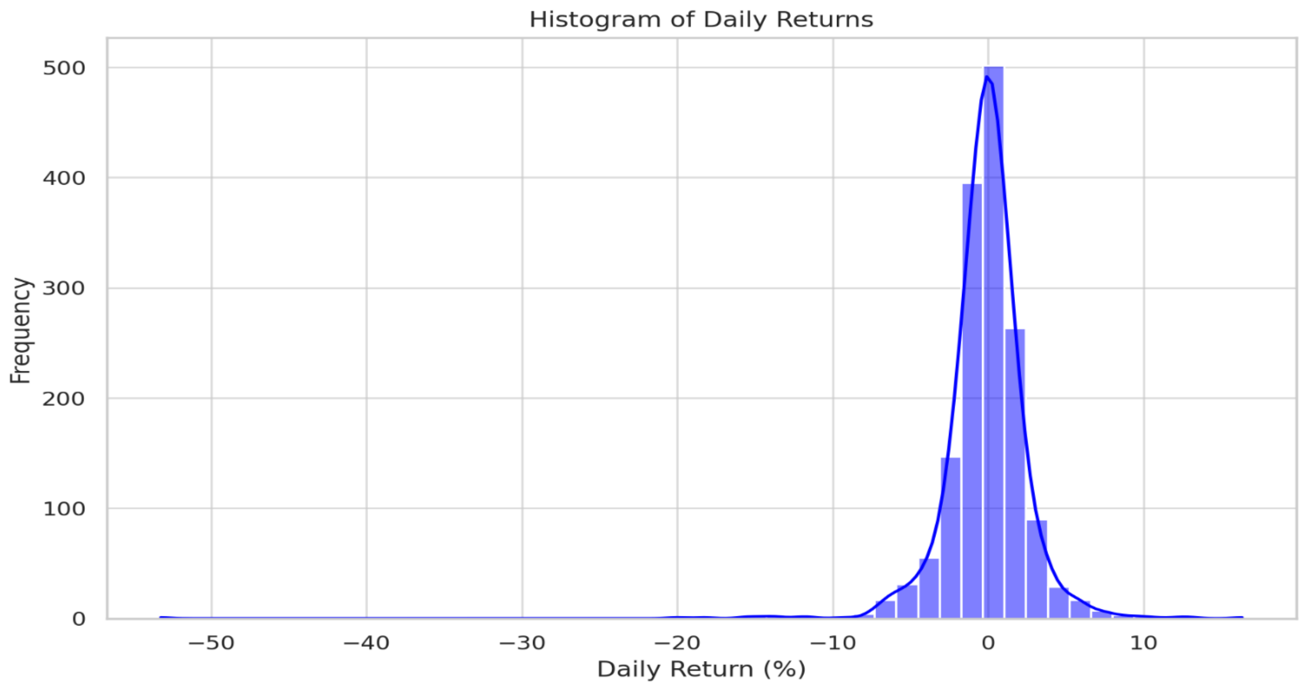
Volume: The box plot for Volume highlights the presence of many outliers above the upper quartile, confirming the right-skewed distribution seen in the histogram. This suggests that on certain days, the trading volume is exceptionally high compared to the norm.

Financial Ratio Analysis:

```
# Calculate daily returns as percentage change in the 'Close' price from one day to the next
data['Daily Return'] = data['Close'].pct_change() * 100


# Display the first few rows to verify the calculation
data[['Date', 'Close', 'Daily Return']].head()
```

```
Date   Close   Daily Return
0 2017-01-03  15.26           NaN
1 2017-01-04  15.96      4.587155
2 2017-01-05  15.72     -1.503758
3 2017-01-06  15.68     -0.254453
4 2017-01-09  15.61     -0.446433
```

```python
# Plotting the histogram of daily returns
plt.figure(figsize=(10, 6))
sns.histplot(data['Daily Return'].dropna(), bins=50, kde=True, color='blue')

# Drop NaN values for plotting
plt.title('Histogram of Daily Returns')
plt.xlabel('Daily Return (%)')
plt.ylabel('Frequency')

daily_return_stats
```



The histogram of daily returns shows a distribution centred around 0, with most days experiencing a change in the -5% to 5% range.

The distribution appears to have a slight negative skew, indicating that there are more days with negative returns than positive ones.

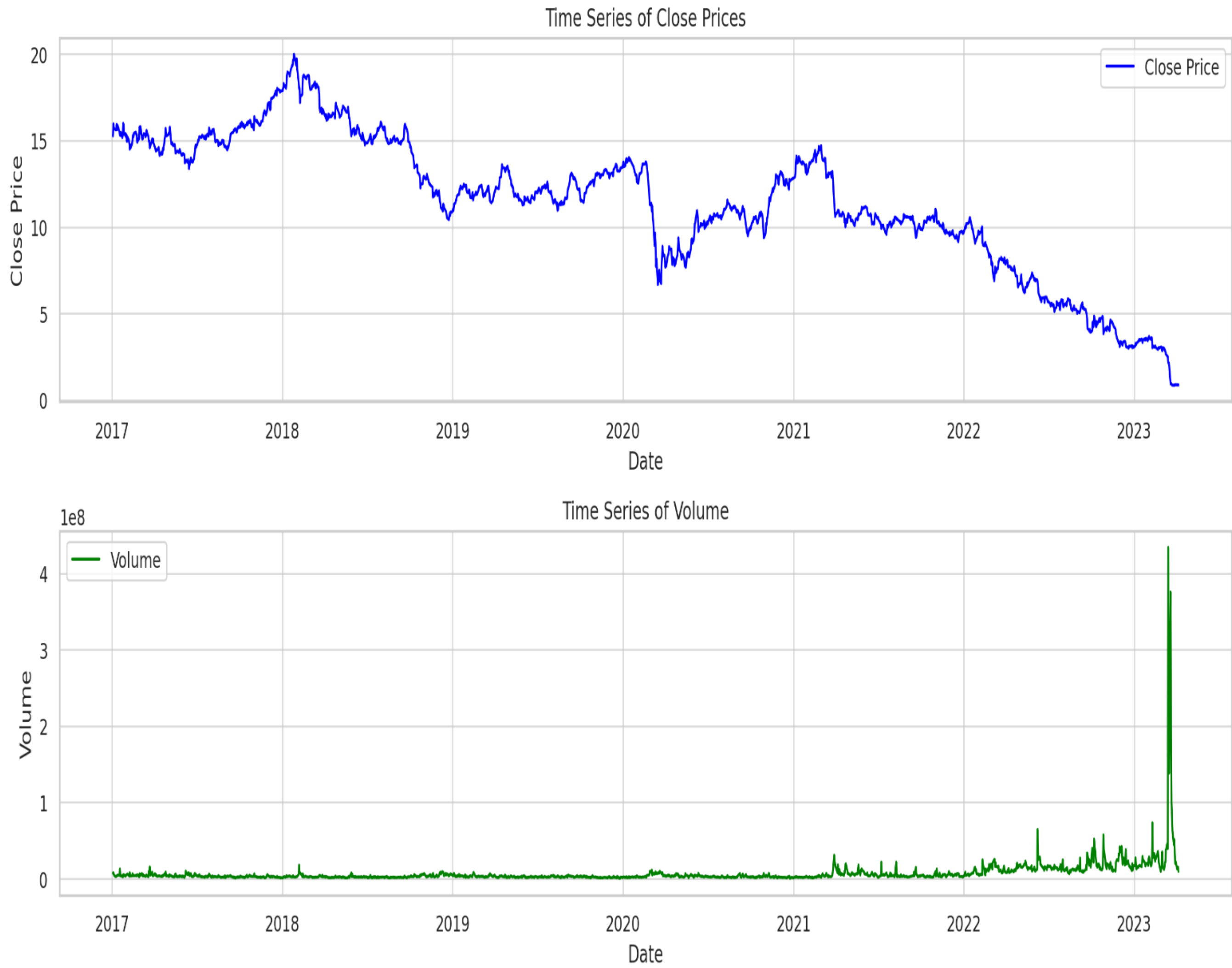
The presence of extreme values on both the positive and negative sides suggests days of significant market movement or company-specific events.

This analysis of daily returns provides a snapshot of the stock's risk profile and market behaviour over the analysed period. High volatility, as indicated by the standard deviation and the range between the minimum and maximum returns, could imply higher risk for investors.

Time Series Analysis:

```
# For Time Series Analysis, we'll plot line charts for 'Close' prices and 'Volume' to identify trends and patterns


plt.figure(figsize=(14, 7))


# Plotting 'Close' prices
plt.subplot(2, 1, 1)
plt.plot(data['Date'], data['Close'], label='Close Price', color='blue')
plt.title('Time Series of Close Prices')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
```

```
# Plotting 'Volume'
plt.subplot(2, 1, 2)
plt.plot(data['Date'], data['Volume'], label='Volume', color='green')
plt.title('Time Series of Volume')
plt.xlabel('Date')
plt.ylabel('Volume')
plt.legend()


plt.tight_layout()
plt.show()
```



The time series plots provide insights into the historical trends of the 'Close' prices and 'Volume' of the financial data:

Close Prices:

The line chart for 'Close' prices shows fluctuations over time, which might indicate periods of volatility or stability in the market. Any significant peaks or troughs could represent key moments for the company or industry, warranting further investigation to understand the underlying causes.

Volume:

The 'Volume' chart exhibits variability, with certain periods showing significantly higher trading volumes. These spikes could correlate with specific events, such as product launches, earnings announcements, or macroeconomic news, affecting investor sentiment and activity.

High volume periods may indicate strong interest or shifts in investor sentiment, either positive or negative, depending on the context of the price movement.

These visual analyses help in identifying patterns, seasonality, or trends over time, which are crucial for making informed investment decisions or conducting further financial analysis.