Python Programming (ACFI827)

Assignment 2: Yahoo Finance Cryptocurrency Data

Deadline: 8 Jan 2026, 12:00 (Noon)

1. Instructions for Conducting Assignment

1.1 How to approach this assignment?

First, try to understand the problem statement and data provided to solve this problem. Break the program into small tasks, then tackle them individually. This assignment is designed to enhance the level of understanding of the given problem. It has four tasks (`Task 1` to `Task 4`). These tasks are ordered logically to an increasing level of difficulty. You probably want to write down essential information and solve some calculations on paper to understand the requirements and check some calculations later by your code. You can check calculations, such as the size of the merged dataset, through simple examples. Manually checking the expected outcome of the code is not a good practice; however, the programmer must understand the desired input and output of the code. A manual calculation to test functions in a code is not a feasible approach for real-life problems because the solution could be very complex, manual checks could be too time-consuming, and, in some cases, impractical.

Make the code more readable and easier to comprehend. Add comments to the code to explain its role and function for future reference. The variable and function names must be meaningful and short. Use GitHub to save and track changes in the code.

In assignment 2, only a report based on your result from Python code will be marked. You can show the Python code in the Appendix of the report.

1.2 What is allowed and not allowed in this assignment?

The Python programming (ACFI827) taught through lectures and labs work is sufficient to complete this assignment. You are free to consult web sources, for example, https://stackoverflow.com, to get ideas. Some of the tasks might have straightforward solutions from the lecturer and lab solutions. You are **not allowed to copy-paste several lines of code written by other programmers and any text (improper paraphrasing) from online sources and course material without clearly referencing** their origin. If your piece of code or idea is inspired by online sources, then add reference as a comment (`#comment is followed by hash in Python`) in the code in the following way to avoid the situation of plagiarism:

```
# <author(s) names> (<year>) <title of program/source code> (<code version>) [<type>]. Web address or publisher.
```

Example:

```
# Sachan, S (2021) String (Python 3.0) [Source code]. https://stackoverflow.com/search?q=string+.....
```

Do not copy-paste the code from other developers on the web. Use it only to study and understand the concept. Turnitin software can detect the plagiarism from online sources.

The following activities contribute towards plagiarism:
1. Sharing Python code to/from classmates and others.
2. Copy-pasting code from a web source.
3. Improper paraphrasing.
This assignment is the equivalent of a standard exam and it counts 50% towards your final mark. Consequently, the University of Liverpool's standard Code of Practice on Assessment applies to this assignment.

What is programming plagiarism?

- "It is using another person's source code and claiming it as your own." (https://www.turnitin.com/blog/what-is-programming-plagiarism-why-is-it-on-the-rise)

Submit Python assignments via the Canvas system. After submission, it will undergo a plagiarism test:

- The Turnitin integrated with Canvas checks for similarities in the codes among all students.
- Plagiarism can be spotted by the examiner while running and evaluating the code.

1.3 How is this assignment assessed?

Total Marks: 100 (Weight = 60%)

Report (`.docx` file): 100 marks

Marks			
Task 1	Task 2	Task 3	Task 4
10	20	30	40

Negative marks will be granted:

- Exceed word limit: maximum 1500 words, excluding appendix
- Poor report structure: A good report should have a title page, list of contents, sections, sub-sections, table and figure numbers, findings/results, reference, sound English
- Insufficient explanation of each task and adopted approach
- Failed to follow the tasks and format specified in the assignment
- Incorrect computational output by the code
- Lack/poor of visualization: Display data analysis and interpretation by charts, graphs, or plots
- No explanation of results displayed through plots
- Absence of conclusion, recommendations and appropriate references

1.4 Submission format and deadline

- The **complete coursework should be submitted as a single report in `.docx` format. You can paste Python code in the appendix.**
- The submission will be via Canvas, and the deadline is 8 Jan 2026 at 12:00 Noon.

1.5. Learning Outcome

Task	Learning Outcome
Task 1 and 2	Task 3 and 4
LO2, L03, L04, S1, S2	L01,LO2, L03, L05, S1, S2, S3, S4, S5, S6, S7, S8

🧩 2. Problem Statement

Cryptocurrencies are **digital or virtual assets** designed to function as a medium of exchange, secured by cryptography and operating on decentralized blockchain networks. They enable **peer-to-peer transactions** without intermediaries like banks, and their values are influenced by **market supply, demand, adoption**, and **external factors** such as regulations and macroeconomic events.

With thousands of cryptocurrencies in circulation, analyzing historical price data is crucial for understanding **market behavior, volatility patterns, and potential investment opportunities**.
This assignment focuses on two prominent cryptocurrencies — **Bitcoin (BTC)** and **Solana (SOL)** — using historical data from **Yahoo Finance** to perform data analysis, visualization, and comparative performance evaluation.

Overview of Bitcoin (BTC)

Bitcoin (BTC), introduced in 2009 by the pseudonymous creator **Satoshi Nakamoto**, is the pioneering cryptocurrency and remains the largest by market capitalization.
It operates on a proof-of-work (PoW) consensus mechanism, where miners validate transactions and secure the network.

Bitcoin is often described as **“digital gold”** due to its limited supply of 21 million coins and its role as a store of value. It is widely used for payments, remittances, and as a hedge against inflation, serving millions of users worldwide.

Overview of Solana (SOL)

Solana (SOL), launched in 2020 by Anatoly Yakovenko and his team at Solana Labs, is a high-performance blockchain platform designed for speed, scalability, and low transaction costs.
It utilizes a hybrid consensus combining proof-of-history (PoH) and proof-of-stake (PoS) to process thousands of transactions per second, making it ideal for decentralized finance (DeFi), non-fungible tokens (NFTs), and Web3 applications. The SOL token is used for staking, governance, and transaction fees within the Solana ecosystem.

📊 Data Source: Yahoo Finance Crypto

Yahoo Finance Crypto provides real-time and historical cryptocurrency data, including daily prices, trading volumes, and market trends for tickers such as **BTC-USD** and **SOL-USD**.
It aggregates information from multiple exchanges and reports the following key metrics:

- **Open** – Opening price of the day
- **High** – Highest price recorded that day
- **Low** – Lowest price recorded that day
- **Close** – Closing price of the day
- **Adj Close** – Adjusted closing price (after accounting for splits or dividends)
- **Volume** – Total number of units traded

The platform is widely used for technical analysis and academic research, and data can be easily accessed through the `yfinance` Python library.

This assignment aims to test your Python coding skills to conduct data exploration, analysis, and visualization skills.
1500 Word count excludes Tables, Figures, caption for Tables and Figures.

- Descriptive statistics and analysis
- Data visualization
- Recommendations and observations based on data analysis & visualization

Retrieve Daily Historical Price Data for Solana (SOL) and Bitcoin (BTC)

Source: Yahoo Finance using the yfinance Python package

In [ ]:
```
# 1️⃣ Install the required library (run this only once or run in console if you are using Spyder or Pycharm)
!pip install yfinance


# 2️⃣ Import the yfinance package
import yfinance as yf


# 3️⃣ Download the historical price data for Solana (SOL) and Bitcoin (BTC)
# collect daily data between 1 January 2022 and the current date (the day when you are doing the assignment)

# Download Solana data
sol = yf.download("SOL-USD", start="2022-01-01", end="2025-10-15")

# Download Bitcoin data
btc = yf.download("BTC-USD", start="2022-01-01", end="2025-10-15")


# 4️⃣ Inspect the first few rows of each dataset
print("Solana Data:")
print(sol.head())

print("\nBitcoin Data:")
print(btc.head())
```

`Task 1` Data Loading and Basic Inspection

(a) Import the datasets for Solana (`SOL-USD`) and Bitcoin (`BTC-USD`) in Python using the `yfinance` library and Pandas DataFrames.
Use the date range from **January 1, 2022**, to **current date**. The current date could be Dec 2025 or Jan 2026, depending on when you are doing the assignment.

(b) Display the **first 8 rows** of each DataFrame (one for SOL and one for BTC).

(c) Print the **shape** (total rows and columns) of both datasets.
Explain what the shape tells you about the data (e.g., how many days of data are available).

(d) Print the **names of all columns** in each DataFrame and describe what each column likely represents (e.g., `'Open'` is the opening price for the day).

`Task 2` Descriptive Statistics

(a) Calculate **basic statistics** (`mean` , `median` , `min` , `max` , `std`) for the `'Open'` , `'High'` , `'Low'` , and `'Close'` prices in both datasets.
Compare the results between the two cryptocurrencies.

(b) Compute the **correlation matrix** for the numerical columns (e.g., `'Open'` , `'High'` , `'Low'` , `'Close'` , `'Volume'`) using Pandas.
Interpret at least two key correlations (e.g., between `'Close'` and `'Volume'`).

(c) Identify **potential outliers** in the `'High'` and `'Low'` prices using the **IQR (Interquartile Range)** rule.
Print the number of outliers found in each dataset and suggest why they might occur (e.g., extreme volatility).

References:

Outlier Detection Methods

`Task 3` Visualization and Time-Based Aggregation

(a) Using **Matplotlib** or **Pandas plotting**, create **line plots** for `'Open'` , `'High'` , `'Low'` , and `'Close'` prices over time for both SOL and BTC.
Annotate at least one significant spike or drop in each plot and discuss possible reasons.

(b) **Resample** the data by **weekly intervals** and calculate the average `'Open'` , `'Close'` , and `'Volume'` .
Display the first 5 rows and discuss observed trends (e.g., smoothing of volatility).

(c) **Volume Analysis:** - (c.1) Plot `'Volume'` over time for both SOL and BTC, identifying spikes above the mean. - (c.2) Create an **overlay plot** comparing `'Volume'` and `'Close'` prices using dual y-axes.

```
    Analyze how high-volume periods relate to price changes.
```

References:

Matplotlib Documentation

`Task 4` Comparative Analysis and Advanced Metrics

(a) Calculate **daily returns** for both SOL and BTC using:

$$\text{daily return} = \frac{\text{Close} - \text{Open}}{\text{Open}}$$

Add them as new columns and compute the **mean** and **standard deviation** of daily returns.

(b) Plot **cumulative returns** to measure the overall performance of each cryptocurrency by calculating the cumulative return over time.

The **cumulative return** measures total compounded growth from daily returns:

$$\text{Cumulative Return}_t = \left(\prod_{i=1}^{t} (1 + R_i) \right) - 1$$

Where:

- R_i = daily return for day i
- $\prod_{i=1}^{t}$ = product of all terms from day 1 to day t
- $(1 + R_i)$ = daily growth factor

Hint: Implementation Formula

In pandas, use vectorized operations:

```
cumulative_returns = (1 + daily_returns).cumprod() - 1
```

(c) Compute the **rolling 30-day volatility**, defined as the standard deviation of daily returns calculated over a 30-day moving window.

Plot the rolling volatility of both cryptocurrencies on a single graph and identify periods where one asset was markedly more volatile than the other.

Explain what these periods might represent in the context of real-world crypto market events (e.g., regulatory announcements, exchange failures, or macroeconomic shocks).

📚 **References:**

Investopedia: Volatility

Pandas Rolling Window