**Probability Distributions Project**

The objective behind this project is to analyse stock prices and understand their underlying statistical distribution. The stock prices are expected to follow a log-normal distribution since stock prices can’t be negative. The main goal is to test this hypothesis using real-world financial data and apply probability and statistical concepts to evaluate the stock price behaviour.

For this project, the stock chosen is Apple Inc. (AAPL) due to it being globally recognised. It has high liquidity i.e. high trading volume therefore, more stable and less prone to irregular price jumps. It also has long historical data. I thought it would be best to aim for stocks with at least 5 years of historical data to ensure there are enough data points for analysis.

A screenshot of a computer code

Description automatically generated

The yfinance library is used to directly fetch financial data. The ‘ticker’ is the identification for the particular stock. The ticker is used to search through yfinance for the market data. The history() method is used to fetch the historical data for a given period, here the start and end dates are specified and I opted for a weekly data interval. Using .to\_csv() saves the historical data to a csv file on your local machine.

A close-up of a computer screen

Description automatically generated

Here, we have loaded the CSV file containing the stock data. The pd.read\_csv() function loads the CSV file into a pandas DataFrame and parse\_dates = True converts the date strings into proper datetime objects.

Using the date column as the index and converting date strings to proper datetime objects is critical for financial data analysis, particularly when dealing with time series data since this allows the data to be aligned well and easy to slice between points for further analysis.

Let’s review what each column in the stock price data tells us:

* Open: The price of the stock at the beginning of the trading day.
* High: The highest price at which the stock traded during the day.
* Low: The lowest price at which the stock traded during the day.
* Close: The price of the stock at the end of the trading day.
* Volume: The no. of shares traded during the trading day. It indicates the liquidity and activity level of the stock.
* Dividends: The cash payments made to shareholders, usually derived from the company’s profits.
* Stock Splits: This reflects any stock splits that occurred, which is when a company divides its existing shares into multiple new shares. This can affect the stock’s price and the no. of shares outstanding.

Now that we have our data, we conduct Exploratory Data Analysis (EDA) to see if any data handling or preprocessing is needed.

A screenshot of a computer

Description automatically generated

We checked the first few rows to see if the data was loaded correctly and as shown it has.

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

After completing EDA, we see that the data is in good shape! There are no missing values, no duplicates and the data types are correct.

Next, I plot the closing price of the stock over time to visualise the price series.

A graph showing a line

Description automatically generated

Now, we calculate the log returns, using the numpy library.

A screenshot of a computer

Description automatically generated

* Np.log(): Calculates the natural logarithm.
* Df[‘Close’].shift(1): Shifts close prices by 1 row to get the previous day’s price.
* Np.log(df[‘Close’]/df[‘close’].shift(1)): calculates the log return i.e. log of today’s price over the log of yesterday’s price.
* Df.dropna(): Drops the first row which will have NaN value because there’s no previous price for the first day.

Calculating these log returns are important because log returns are often used in financial analysis to model the returns of stock prices.

Now, visualising log returns helps understand their distribution.

A graph showing a blue line

Description automatically generated

There is a lot of noise which is expected because weekly log returns generally exhibit a lot of short term fluctuations. This is because stock prices move continuously based on market conditions. Therefore, it’s common to see small-scale changes on a weekly basis. Around the start of 2020, there is a large amplitude of short-term fluctuations, this usually corresponds to market events that cause high volatility. In this case, this corresponds to the onset of the COVID-19 pandemic, which led to extremely high market volatility and sharp price movements. After a period of heightened volatility, it’s common for the amplitude of log returns to settle down, as shown from 2021 onwards. This reflects a return to more stable market conditions, resulting in relatively smaller weekly price changes.

Plotting a histogram to visualise how the log returns are distributed yields…

A green graph with numbers

Description automatically generated

As expected, the stock returns are log-normally distributed due to the non negative nature of stock prices.

Now, we fit the log-returns to a normal distribution using Scipy.stats. The log returns are in log form but we do not fit them to a log normal distribution as the inputs to some values are negative after being computed in log form so we fit to a normal distribution to neglect the errors as a result of this, the data being in log from and being fitted to a gaussian distribution is equivalent to the stock price data in its pre-log form being fitted on to a log normal distribution.

A green and blue line graph

Description automatically generated

Now that the probability density function (PDF) of the normal distribution is overlaid on the histogram to visually assess the fit, this is further proof that the null hypothesis of the data following the reference distribution seems to be acceptable. To quantify this, further statistical analysis was conducted.

The next step in this project is to perform goodness of fit tests such as chi-square and Kolmogorov-Smirnov (KS) tests, to evaluate if the data follows the fitted distribution. In doing this, I had challenges and reached interesting and sensical conclusions:

* When performing chi-squared I initially used bin edges instead of bin centres to get the probability density from the normal distribution. The issue with this is that the bin edges represent the boundary of a bin, and they don’t necessarily represent the values within the bin. The center of a bin is a more accurate reflection of the average value of the data in that bin.
* After using bin center’s, I had a shape mismatch with the observed frequency having 1 more bin than the expected frequency. To alleviate this issue, after calculating the expected frequency, I scaled it by multiplying with the ratio of the sums of the observed frequency and expected frequency.
* The chi-squared statistic was very high indicating that the data did not follow the distribution. From the visual fit of the gaussian on to the data, this was not an expected conclusion. Therefore, I deduced that the bins did not align with the distribution and decided to carry out the KS test to see if this also suggested rejecting the null hypothesis.
* The KS statistic indicated a noticeable difference between the data and the reference distribution, but not extreme. However, the p-value was very small indicating that our null hypothesis should be rejected.
* It should be noted that at this initial point in the project, the data interval was daily and not weekly. This meant that I had a lot of noise than shown in the graphs above, this noise deviated from the normal distribution significantly. Therefore, at this point I decided to aggregate the data to have weekly returns instead. After doing this, the KS test result now suggests that the returns do not significantly deviate from a gaussian shape.
* Carrying out the chi-squared test again after making this modification still resulted in a high statistic, showing that this test is flawed for the data due to it being best suited for binned data and not continuous distributions, which is what the KS test is best suited for.

To conclude, I had a KS statistic of 0.0582. This represents the maximum difference between the empirical cumulative distribution function (CDF) of the log returns data and the theoretical CDF (normal distribution). Since this value is relatively small, it indicates that the 2 distributions are quite similar. The p-value of 0.329 is quite large (above common significance levels like 0.05 or 0.01), meaning that the difference between this data and the normal distribution is not statistically significant i.e. it could easily occur by chance. Therefore, the data could plausibly follow a normal distribution.

Based on the statistical tests, we conclude that the log returns do not significantly deviate from a normal distribution, fully supporting the assumption that the data follows a normal distribution.