**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.

A general objective of this project is to review and apply probability laws and common distributions relevant to financial mathematics.

A probability law that supports the stock return analysis is the Law of Large Numbers (LLN). The LLN states that as the number of observations increase, the sample mean will converge to the expected value (mean of the distribution). As we already have a dataset spanning several years, we can demonstrate the LLN by analysing how the mean of log returns converges as more data points are included. In financial terms, this helps us see how average returns stabilise over time.

A graph of a number of numbers

Description automatically generated

In this plot, you’ll observe how the cumulative mean fluctuates at first but begins to converge towards the overall mean as more data points are considered.

To understand the moments of a distribution, we can analyse the moments of the log returns to show how stock returns align with probabilistic characteristics.

A black screen with white text

Description automatically generated

The mean of the log returns represents the average return of the stock over the period analysed. The positive mean suggests that, on average, the stock price has increased over time. The value 0.0055 is small for log returns, but over a long period, it can indicate a general upward trend in stock prices.

The variance of 0.0016 suggests a relatively moderate level of volatility in the stock’s returns over the past 5 years. This variance indicates that the stock has a moderate amount of daily fluctuation. Therefore, moderate risk, implying a balanced risk level.

Skewness measures the asymmetry of the distribution of log returns. The negative skewness value of -0.4854 indicates that the distribution has a longer tail on the left side, meaning that there have been large negative returns compared to large positive returns. The negative skewness implies that while the stock has had mostly small positive returns (as suggested by the positive mean), there are occasional large negative returns. In practical terms, this means that while the stocks generally trend upwards, there are larger negative shocks, although they are less frequent.

Kurtosis measures the “tailedness” of the distribution, indicating how likely extreme values are. A kurtosis of 3 corresponds to a normal distribution. Here, kurtosis is slightly below 3, suggesting that the distribution has fewer extreme values than a normal distribution (platykurtic). Distribution is flatter, less peaked. Therefore, most values are around the middle instead of the ends. The lower-than normal kurtosis indicates that the stock’s log returns have fewer extreme price changes than expected under a normal distribution. This means that the stock might not experience as many sharp price jumps or crashes compared to stocks with a higher kurtosis, meaning the stock may be more stable than some others.

To assess a part of our objective and the validity of our log-normal assumption (which is uncertain due to the less than normal expected moments), it would be best to look at another commonly used distribution in finance, the t-distribution.

A graph of a normal distribution

Description automatically generated

Now that we’ve fitted both distributions to the data, we can now perform a KS test to evaluate the goodness of fit.

A screen shot of a computer

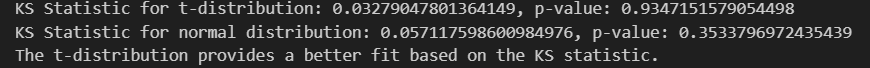
Description automatically generated

We’ve compared the sample distribution of the data (log retuns) against two CDF’s to assess how well the data fits these distributions, importing kstest allows for this. The lambda functions defines a random variable x as the input to a CDF function which assesses the probability that it is less than or equal to a particular value. CDF’s are crucial in the KS test as they’re used to compare the empirical distribution of the log returns against the theoretical distributions.

A screen shot of a computer code

Description automatically generated

These lines perform the KS test using the log returns and the respective CDF’s. Both functions return two values: ks\_stat, which is the ks statistic representing the maximum difference between the empirical distribution and the CDF. P-value, which indicates the probability of observing such a statistic under the null hypothesis that the log returns follow the CDF. After printing these values and using conditional logic to output which CDF is the best fit, we yield…



This discovery contradicts that the log-normal model is the best fit. An explanation for this is due to the negative skewness. This explains that the stock’s log returns have a heavier left tail than a log-normal would expect. The t-distribution in general accounts for heavier tails and so is commonly used in finance. Consequently, the t-distribution captured the negative skew risk much better and so is a better fit.

To conclude this project, let’s summarise our findings:

* The application of the LLN demonstrates that as we extend the time horizon, the cumulative mean of the log returns converges. This illustrates the stabilising behaviour of long-term average returns, reinforcing the idea that, over time, stock prices tend to stabilise around their average value.
* The analysis of skewness and kurtosis in the log returns reveals that while the distribution is approximately normal, it exhibits slight asymmetry and heavier tails. This is characteristic of stock returns, which can show unexpected behaviour and fluctuations. Such moments are critical in understanding the risk associated with stock investments.
* Although, the normal distribution provided a reasonable approximation for the log returns, the t-distribution proved to be a more suitable fit. This is due to its ability to account for heavier tails, indicating that AAPL returns can occasionally experience extreme price swings. This finding underscores the importance of selecting the appropriate statistical distribution when analysing financial data, particularly in the context of risk management and decision-making.