**Data preparation and cleaning**

ts1 ts2

type float64 float64

count 201517.0 936488.0

missing 84.68% 28.81%

unique 3901 10114

min 91.512942 0.60115

25% 95.106221 0.87145

50% 95.462056 0.945

75% 96.042938 1.0278

max 96.852863 1.10805

mean 95.17 0.92

median 95.46 0.94

std 1.282164 0.11992

mode 95.514825 1.03665

1. **Impute missing values: Start by imputing missing values in your original data using appropriate methods, such as interpolation or forward-filling. This step ensures that you have a complete and consistent dataset to work with, which is particularly important for traditional time series models like ARIMA.**

Original data missing values investigation

Missing values

Missing values in ts1: 1113924

Missing values in ts2: 378953

Max missing range in ts1: 3579

Max missing range in ts2: 2139

Max missing range in ts1: 4.97 days

Max missing range in ts2: 2.97 days

Check this time gaps of the intervals in the data (just do once for entire dataframe)

Resample and check for missing days: Saturdays do not appears in the data since the market is closed. so remove Saturday and find the new missing data.

# Assuming 'df' is your DataFrame with a DateTimeIndex

df = df[df.index.dayofweek != 5] # 5 represents Saturday

1. Data Preparation
   1. Identifying the extent of missing data in each time series
      1. What is the distribution of gaps in the missing data without Saturdays (may still include holidays)
      2. Check for holidays
      3. Count missing days not including saturdays
      4. check for evidence of highly variable time gaps in your data, create a histogram of the time differences between adjacent data points
2. **Understand the nature of the missing data**: Determine if the data is missing completely at random (MCAR), missing at random (MAR), or missing not at random (MNAR). This will help you choose an appropriate imputation method.
3. **Choose an appropriate imputation method**: Use a method that aligns with the nature of the missing data and the underlying data generating process. For example, you can use mean or median imputation for MCAR data, regression imputation or k-nearest neighbors for MAR data, and more advanced techniques like expectation-maximization or multiple imputation for MNAR data.

* For MCAR data, you can use simple imputation methods like mean, median, or mode imputation.
* For MAR data, you can use more advanced methods like regression imputation, k-nearest neighbors, or expectation-maximization.
* For MNAR data, you can use techniques like multiple imputation, pattern-mixture models, or selection models.

1. **Perform sensitivity analysis**: Compare the results of your EDA using both the original data with missing values and the imputed data. This will help you understand the impact of the imputation on your analysis and identify potential biases introduced by the imputation method.
2. **Document your assumptions and methods**: Clearly document the assumptions you made about the missing data and the imputation method you used. This will help you and others understand the limitations of your EDA and make informed decisions based on the results.

* **For the number of missing days, they are not likely to all be holidays (>100 per year) so reflect actually missing data. That should be considered as reducing the amount of relevant data points.**
* **However for 5 years of 2-minute interval data, it is unlikely that will have highly variable time gaps between data points. This is because the 2-minute interval is small enough to capture most of the variation in the data, and the regularity of the interval reduces the likelihood of gaps.**
* **if you do see evidence of highly variable time gaps in your data, it could indicate that there are issues with the data collection process. there may be issues with the equipment or software used to collect and store the data.**
* **For interval seasonal models like arima, equally spaced time interval data is required, so will need to impute for the missing time intervals** 
  1. Handle missing data (NaN values) by Deciding on an appropriate imputation method (e.g., forward fill, interpolation, or dropping missing data)
  2. Choices to impute or remove.
     1. Removing data points: If you remove these data points, you may lose valuable information, and the remaining data might not be representative of the original time series. This can lead to biased estimates and affect the accuracy of your analysis. However, if the missing data points are not crucial to your analysis or if they represent a small proportion of the total data, removing them might be a reasonable option.
     2. Imputing data points: Imputing the missing data points can help preserve the structure of the time series and maintain the continuity of the data. However, the accuracy of the imputed values depends on the imputation method used and the nature of the missing data. Some common imputation methods include linear interpolation, moving average, and more advanced techniques like state-space models with Kalman filtering.
     3. Fill forward all Saturdays
* Consider different imputation methods:
* Forward filling since it will handle holidays and market closed days by assuming that the rice remained constant. Does not mean that it is appropriate otherwise
* Time weighted interpolation would be useful if there are highly variable gaps in the data

1. Data Exploration

a. Visualize the time series data (ts1 and ts2) to understand their behavior over time

b. Calculate summary statistics (mean, median, standard deviation, etc.) for each time series c. Investigate the correlation between ts1 and ts2

1. Analyzing the nature of the outliers is crucial to determine the appropriate way to handle them. Here are some steps and example approaches to analyze the nature of outliers in your stock price data:
   * 1. Visualize the data: Start by plotting the stock price data to get a visual understanding of the outliers. You can use line plots, box plots, or scatter plots to identify extreme values.
     2. Investigate the context of the outliers: extreme market event was 2008 crash in September and october
     3. Compare with other stocks or market indices: Check if the outliers in your stock price data are also present in other stocks or market indices during the same period. If similar patterns are observed across multiple stocks or indices, it could indicate that the outliers are genuine market events rather than errors or noise.
     4. Analyze the distribution of returns: Calculate the daily returns of the stock price data and analyze their distribution. If the distribution is heavy-tailed or has a high kurtosis, it might indicate that extreme events are more likely to occur, and the outliers could be genuine.
     5. Perform statistical tests: You can use statistical tests, such as the Grubbs' test or the Dixon's Q test, to determine if the outliers are significantly different from the rest of the data. However, keep in mind that these tests assume normality, which might not hold for stock price data.

Keeping imputed values in outliers:

1. **Consistency**: When you perform your analysis and modeling, you will be using the imputed log return data. Therefore, it's essential to ensure that the data you use for outlier detection is consistent with the data used in your models. This will help you identify and address any potential issues that may arise due to outliers in the imputed data.
2. **Influence of imputation**: Imputation methods can sometimes introduce artificial outliers or mask real outliers, depending on the method used and the characteristics of the data. By testing for outliers using the imputed log return data, you can assess the impact of the imputation method on the distribution of returns and make any necessary adjustments to your imputation approach or outlier detection methods.
3. **Model performance**: Outliers can have a significant impact on the performance of your models, especially if they are sensitive to extreme values. By testing for outliers using the imputed log return data, you can ensure that your models are robust to the presence of outliers and that your results are not unduly influenced by extreme values in the imputed data.

Arguments for:

Continuity of the time series is essential for accurate outlier detection for several reasons:

1. Temporal dependencies: Financial time series data, like stock returns, often exhibit temporal dependencies, meaning that the value at a given time point is influenced by the values at previous time points. To accurately detect outliers, it is essential to maintain the correct order and spacing of the data points to preserve these dependencies. Ignoring missing data or leaving gaps in the time series can disrupt the underlying temporal structure, leading to misleading results.
2. Statistical tests and measures: Many outlier detection methods, such as moving averages or z-scores, rely on the calculation of summary statistics or other measures that require a continuous time series. Discontinuities in the time series can distort these calculations and result in incorrect identification of outliers. For example, missing values can artificially lower the mean or standard deviation, causing data points to appear more extreme than they actually are.
3. Context: Outliers are usually identified by comparing individual data points to their surrounding context. In a time series, this context is often based on the immediate past or future values. Discontinuities in the time series can make it difficult to accurately assess the context of a given data point, leading to incorrect outlier detection.
4. Seasonality and trends: Many time series exhibit seasonality or underlying trends, which can influence the identification of outliers. Discontinuities in the time series can obscure these patterns or introduce artificial ones, complicating outlier detection and potentially leading to false-positive or false-negative results.

Let's consider the pros and cons of both approaches step by step:

1. **Removing the outliers:**

Pros:

* + Your model may better capture the "normal" market behavior, as it will be less influenced by extreme events.
  + The model's performance metrics (e.g., mean squared error) may improve, as large outliers can have a significant impact on these metrics.

Cons:

* + By removing the outliers, you may lose valuable information about how your trading strategy performs during extreme market conditions.
  + The model may underestimate the risk associated with your trading strategy, as it will not account for the possibility of large market movements.

1. **Keeping the outliers:**

Pros:

* + Your model will consider the full range of market conditions, including extreme events like the 2008 market crash.
  + The model's risk estimates may be more accurate, as it will account for the possibility of large market movements.

Cons:

* + The model may be overly influenced by the outliers, leading to less accurate predictions during "normal" market conditions.
  + The model's performance metrics may be worse, as large outliers can have a significant impact on these metrics.

To decide whether to remove or keep the outliers, consider the following questions:

* What is the primary goal of your model? If you want to capture the "normal" market behavior, removing the outliers may be more appropriate. If you want to account for extreme market conditions, keeping the outliers may be better.
* How likely are extreme events like the 2008 market crash to occur in the future? If you believe that such events are rare and unlikely to happen again, removing the outliers may be reasonable. If you think that similar events could occur in the future, keeping the outliers may be more appropriate.
* Can your model handle the outliers effectively? Some models, like robust regression or tree-based models, can handle outliers better than others. If your model can effectively handle the outliers, keeping them may be more appropriate.
* Want to account for extreme market conditions
* Black swan events are unpredictable, but current economic climate is very uncertain and there is significant instability in many banking institutions that could lead to extreme market behaviour. (find a reference of this)
* Test volatility based models and deep learning models that are more robust to outliers

1. **Extreme market events**: It is reasonable to assume that extreme market events are meaningful to include in your analysis. These events can provide valuable information about the stock's behavior during periods of high volatility or market stress. Including such events in your analysis can help you better understand the stock's risk profile and potentially identify trading opportunities or risk management strategies.
2. **Outliers due to incorrect measurements**: While it is possible that some outliers in your data are due to incorrect measurements, it is essential to carefully investigate the source of these outliers before making this assumption. In some cases, outliers may indeed reflect the true behavior of the stock, especially during extreme market events or periods of high volatility. In other cases, outliers may be caused by data errors, such as missing or misreported values.

Noise

case of stock price data, you can assume that the data is generally accurate and without measurement errors, as stock exchanges and data providers have strict quality control measures in place.

**Oberve that the stock price may be affected by extreme market events such that it would be relevant to consider that the model should consider extreme price movements and how the price was affected. With the aim of creating a general forecasting model, this is useful information to include in the model training. analyze the impact of extreme market events on both series or to build a model that captures the behavior of the series during such events, you should keep the common outliers. These outliers represent important information about the joint behavior of the series during extreme market conditions.**

**After analyzing the nature of the outliers using these approaches, you can decide whether to keep them in the dataset or apply outlier handling techniques. If the outliers represent genuine extreme events in the stock market, it's generally better to keep them in the dataset, as they could provide valuable information for your time series and LSTM models. However, if the outliers are caused by errors or noise, you might consider handling them using the techniques discussed earlier.**

**By combining the results of these techniques, you can identify and evaluate periodic patterns or cycles in your time series data. Note that the presence of strong periodic patterns may suggest the use of models that can capture seasonality or cyclical components, such as SARIMA, exponential smoothing state space models, or Fourier-transform-based regression models. If significant non-seasonal cycles are detected, you might also consider using models that can capture cyclical components like Bayesian structural time series models or incorporating cyclical features into machine learning models like LSTM or LightGBM.**

1. Preprocessing and cleaning: Perform any necessary preprocessing and cleaning steps to ensure that your data is consistent and meaningful. This may include removing outliers, adjusting for seasonality, or transforming the data to achieve stationarity.
2. Visualize the log returns for ts1 and ts2
3. Hypothesis Formulation

a. Formulate hypotheses based on the suggested lines of thinking:

i. H1: ts1 can predict ts1

ii. H2: ts2 can predict ts2

iii. H3: ts1 can predict ts2

iv. H4: ts2 can predict ts1

1. Data Transformation a. Calculate log returns (r) for each time series using the formula: r = log(p(t)/p(t-1)) b.
2. Model Selection and Evaluation
   1. a. Identify suitable quantitative models for time series analysis (e.g., ARIMA, GARCH, VAR,

COmbnie arima with garch

* 1. Machine Learning models like LSTM, etc.)

b. Split the data into training and hold out, last month

c. For each hypothesis, apply the selected models and evaluate their performance using relevant metrics (e.g., Mean Squared Error, Mean Absolute Error, etc.)

* 1. Evaluate models for H1: ts1 predicting ts1
  2. Evaluate models for H2: ts2 predicting ts2
  3. Evaluate models for H3: ts1 predicting ts2
  4. Evaluate models for H4: ts2 predicting ts1

d. Perform cross-validation and model selection to choose the best model for each hypothesis

e. Investigate the residuals of the best models for each hypothesis to ensure their validity

1. Model Interpretation and Conclusion
   * 1. Analyze the results of the best models for each hypothesis
     2. Draw conclusions based on the model performance and the significance of their predictive power

* Choose to impute using two different methods to handle short gaps and long gaps in the data because will preserve structure of time series and not affect stability of traditional time series modelling. This assumes that the price continues to trend over missing days which may be holidays.
* Would also test using time-weighted imputation, and Expectation maximisation imputation respectively for short and long time gaps.
* For the machine learning models, the gaps equal to or longer than a day in duration can be removed as the models are more robust to irregularities.
* The result of the forecasting with the models and testing accuracy will give insight on how

Modelling

1. Resample the data: Resample your imputed data to the desired frequency (e.g., daily) using aggregation functions like mean, sum, or other relevant statistics. This step is essential for both traditional time series analysis and machine learning models, as it ensures that your data is consistently spaced and can be easily split into training and testing sets.
2. Feature engineering: For machine learning models like LSTM, you may need to create additional features based on your resampled data. This can include lagged variables, moving averages, or other derived features that can help improve your model's performance.
3. Model training and testing: Split your resampled data into training and testing sets, and use these sets to train and evaluate your traditional time series models (e.g., ARIMA) and machine learning models (e.g., LSTM). Be sure to perform cross-validation or other model selection techniques to ensure that your models are robust and generalize well to new data.

Discussion of using imputed vs non imputed in traditional vs machine learning

Consider the accuracy of each and the implications: not realistic to impute accurately over missing days including market close days

**Question**: Which approach offers greater stability in the forecasting in terms of sensitivity to using imputed or removing missing time gaps of greater than 1 day.

**Since forecasting for daily accuracy, information lost or bias introduced by imputation most likely wont have a significant affect on trend prediction, but will be evident in forecasting**

It is possible to compare the results of forecasting between the two sets of models with different approaches to handling missing values, but you should be cautious when interpreting the results. The comparison may not be entirely fair due to the differences in data preprocessing, but it can still provide some insights into the performance of the models under different conditions.

When comparing the forecasts, consider the following points:

1. Consistency: Ensure that the same evaluation metrics and hold-out data are used for both sets of models. This will help maintain consistency in the comparison.
2. Interpretation: Be cautious when interpreting the results, as the differences in handling missing values may impact the performance of the models. It's essential to understand the limitations of each approach and how they might affect the forecasts.
3. Context: Consider the context of your analysis and the goals you want to achieve. If the primary objective is to compare the performance of different models, it might be more appropriate to use the same approach for handling missing values across all models. However, if you want to explore the impact of different data preprocessing techniques on model performance, comparing the results of models with different approaches to handling missing values can provide valuable insights.
4. Introduction
   * Briefly introduce the problem and the objective of your analysis.
   * Provide an overview of the data you are working with, including the time frame, frequency, and any preprocessing steps.
5. Methodology
   * Explain the methods and techniques you have used for forecasting the stock prices, such as time series analysis, machine learning models, or any other approaches.
   * Justify your choice of methods, explaining why they are suitable for this task and the type of data you are working with.
   * Discuss the imputation methods used to handle missing values in the data, and explain why you chose these methods based on the distribution of missing value lengths.
6. Model Selection and Evaluation
   * Describe the process of selecting the best model(s) for forecasting, including any model comparisons, cross-validation, or hyperparameter tuning.
   * Discuss the performance metrics used to evaluate the models, such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or R-squared, and explain why these metrics are appropriate for your analysis.
7. Results
   * Present the results of your analysis, including the forecasts generated by your chosen model(s).
   * Visualize the forecasts and compare them with the actual stock prices, highlighting any interesting patterns or trends.
   * Discuss the accuracy and reliability of your forecasts, and provide any insights or recommendations based on your findings.
8. Limitations and Future Work
   * Acknowledge any limitations of your analysis, such as assumptions made, data quality issues, or potential biases in the models.
   * Suggest possible improvements or extensions to your analysis, such as incorporating additional features, using alternative models, or exploring different time horizons for forecasting.
9. Conclusion
   * Summarize the key findings of your analysis and reiterate the main insights or recommendations.
   * Conclude by emphasizing the value of your work and its potential impact on decision-making in the context of stock price forecasting.

* Assume that from visual analysis there are extreme market events that are meaningful to include
* Assume that outliers would be due to incorrect measurements and do not reflect the behaviour of the stock but verify this, so have to analyse the extreme values of the returns to clarify the nature of the outliers
* analyse the nature of outliers or noise in your non-normal, large tail price returns data and determine if it's due to measurement error or part of the price behaviour Test: Perform Outlier analysis to detect extreme values On separate un imputed data. Use Tukey, pot exp max, hamper identifier to find.
  + Statistical analysis: Perform a statistical analysis of the price data to identify potential outliers. You can use methods like Tukey's Fences, Peaks Over Threshold, or Hampel Identifier, as mentioned earlier. This will help you detect extreme values in the data.
  + Volatility clustering: Analyze the volatility of the stock's price data. If you observe volatility clustering, where periods of high volatility are followed by more high volatility, it may indicate that the outliers are related to extreme market events.
  + Pot method
  + Visual inspection: Start by plotting the price returns data and visually inspect the outliers. This can give you an initial idea of the presence of extreme events or measurement errors.
  + Descriptive statistics: Calculate descriptive statistics such as mean, median, standard deviation, skewness, and kurtosis. High skewness and kurtosis values can indicate the presence of extreme events or outliers.
  + Autocorrelation analysis: Check for autocorrelation in the returns data. If there's significant autocorrelation, it may suggest that the outliers are part of the price behaviour.
  + Rolling window analysis: Apply a rolling window analysis to calculate the local mean and standard deviation of the returns. If the outliers are consistently present in different windows, it may indicate that they are part of the price behaviour.
* Conclude: Keep extreme changes
* Assume noise is not due to measurement error but inherent in the data since cannot verify otherwise

Self-documenting code implies that the code is written in a clear and readable manner, making it easy for others (including future developers or even the author) to understand its purpose, logic, and structure without the need for extensive external documentation or comments. For production-ready code, self-documenting code is highly desirable, as it can greatly improve the maintainability, extensibility, and overall quality of the software.

Here are some key aspects of self-documenting code in the context of production-ready code:

1. **Meaningful variable, function, and class names**: Using clear, descriptive names for variables, functions, and classes helps convey the purpose and functionality of each element in the code.
2. **Consistent naming conventions and code style**: Adopting consistent naming conventions and following a standardized code style helps improve the readability and understanding of the code.
3. **Modularity**: Breaking code into smaller, reusable, and testable functions or classes can make it easier to understand the overall structure and purpose of the codebase.
4. **Avoiding magic numbers**: Replacing hardcoded values (magic numbers) with well-named constants or variables can make the code more self-explanatory and easier to maintain.
5. **Error handling**: Clear and robust error handling can help prevent unexpected behaviors in production and make the code more reliable.
6. **Comments where necessary**: While self-documenting code should reduce the need for comments, there may still be instances where adding comments can provide valuable context or explanation for complex algorithms, edge cases, or workarounds.
7. **Adherence to SOLID principles**: Following SOLID principles (Single Responsibility, Open/Closed, Liskov Substitution, Interface Segregation, and Dependency Inversion) can lead to better-structured, more maintainable code.

By focusing on writing self-documenting code, developers can create production-ready software that is more maintainable, scalable, and easier for others to work with, ultimately contributing to the success of the project.