Stock Volatility Forecasting using Machine Learning Models

Data Pre Processing

Data preprocessing involves cleaning and transforming raw data into a suitable format for analysis. In this assignment, preprocessing helps ensure that the financial data is properly structured (e.g., converting cumulative data into quarterly values), allowing for more accurate model training and predictions.The first step in pre-processing was to convert cumulative financial data to quarterly values.For features like Revenue, Net Income, and Operating Cash Flow, which are recorded cumulatively, the data is transformed into quarterly values using the .diff() method. This ensures that financial data is aligned to a common time scale (quarterly).Converting cumulative values to quarterly metrics standardised the time scale, allowing the model to better understand the variations over time, which is critical for making accurate predictions about volatility.The data set was checked for null/missing values and no missing values were found hence we proceeded to the EDA.

Exploratory Data Analysis

A crucial initial step in data analysis that involves examining the dataset to understand its structure, patterns, and relationships. We start by performing correlation analysis, from our analysis we notice :

- Return has a small positive correlation with Volatility (~0.18).
- Features like Net Income, Total Assets, and EPS show a small to moderate negative correlation with Volatility.
- High intercorrelations are seen between financial features like Revenue, Net Income, and Gross Profit (values close to 1), indicating these are closely related to each other but less directly related to volatility.
- Investing Cash Flow and Financing Cash Flow show moderate negative correlations with many financial features but have a weak relationship with Volatility.

For the next part of our EDA we have generated **histograms** for each numerical feature to observe their distributions, which helps identify skewness or outliers. We also generate **box plots** to detect outliers in these features, providing insights into the range and variability of the data.

We now want to study the relation of our features with the target variable we generate scatter plots to examine the relationship between each numerical feature and the target variable, 'Volatility'. We iterate over the numerical features and create a **scatter plot** for each one to visually identify any trends, correlations, or potential patterns between the feature and 'Volatility'.

- **Open , Close, High/Low vs. Volatility**: The scatter plot shows a concentration of data points near lower volatility values, with no strong linear or clear trend. Most data points cluster around lower 'Volatility', indicating little correlation between 'Open' prices and 'Volatility'.Similar to the 'Open' plot, the data points are densely packed near low volatility values, with a few outliers. There is no apparent linear relationship between 'Close' prices and stock volatility.Both 'High' and 'Low' prices follow a similar pattern to 'Open' and 'Close'. The data points remain clustered around lower 'Volatility', suggesting that the daily price range is not strongly predictive of volatility in this dataset.
- **Volume vs. Volatility**: The scatter plot reveals a spread of points, indicating that as trading volume increases, volatility remains low for most observations. However, some outliers show higher volatility at extreme volume levels, possibly suggesting that unusually high trading volumes could be linked to increased volatility in a few instances.
- **Return vs. Volatility**: This scatter plot exhibits a clearer relationship. As 'Return' values increase or decrease, we observe a more scattered distribution of volatility values, suggesting a potential nonlinear relationship between stock returns and volatility.
- **Total Assets/Total Liabilities vs. Volatility**: Both plots show minimal correlation, with data points clustering at lower volatility levels, indicating that these financial indicators are not strongly related to short-term volatility.

Since the predictions are time series focused we generated time series **line plots** for several features, such as 'Open', 'Close', 'High', 'Low', 'Avg_Price', and 'Volatility', against the 'Date'. The time series plots show fluctuations over time with occasional peaks and troughs, but overall there is a gradual downward trend followed by some recovery later in the period. This suggests that the stock prices experienced periods of volatility but eventually stabilised.

Feature Engineering :

1. Price Range (High - Low):
    - EDA Insight: Earlier analysis on features like High, Low, Open, and Close revealed that individually, these features did not have a strong correlation with volatility. However, the price fluctuations during a trading day (difference between High and Low) could provide additional information about volatility.
    - Reason for Extraction: By creating the Price_Range feature, we attempt to capture intra-day volatility, which could be more predictive of overall stock volatility than the absolute prices themselves.
2. Daily Spread (Close - Open):
    - EDA Insight: Scatter plots showed minimal correlation between Open, Close, and Volatility. However, by focusing on the difference between the closing and opening prices (Daily_Spread), we aim to capture the volatility within a single day.
    - Reason for Extraction: This new feature is essential because daily price changes could indicate market reaction and hence be more correlated with volatility.
3. Cyclical Features (Month_sin, Month_cos):
    - EDA Insight: Time series analysis of features like Volatility, Open, and Close across months revealed possible seasonal trends.
    - Reason for Extraction: To account for seasonality, cyclical transformations (sin and cos values of the month) were created, helping the model capture seasonal effects that could influence stock prices and volatility.
4. Financial Ratios:
    a. Price to Earnings (Price_to_Earnings):
        - EDA Insight: Scatter plots and correlation matrices indicated financial metrics like EPS, Revenue, and Net Income were inversely related to volatility.
        - Reason for Extraction: The Price_to_Earnings ratio could reflect market sentiment about a stock's valuation and its potential volatility. It adds a more granular financial perspective compared to the raw prices or earnings alone.
    b. Profit Margin (Net Income / Revenue):
        - EDA Insight: Net Income, Gross Profit, and Revenue showed significant intercorrelation. This prompted us to derive ratios, as financial ratios are more robust indicators of a company's performance.
        - Reason for Extraction: Profit_Margin helps assess how efficiently a company generates profit relative to its revenue, which might correlate with stock volatility during financial uncertainty.
    c. Return on Assets (ROA):
        - EDA Insight: Similar to the Profit_Margin, ROA measures how well a company is using its assets to generate profits, which can be important in predicting long-term stability and volatility.
        - Reason for Extraction: It reflects how efficiently management is using assets, which might impact volatility through investor sentiment.

5. Risk Indicators:

    a. Debt-to-Equity Ratio: High debt levels can increase a company's risk, which may lead to stock price volatility, especially during economic downturns.
    b. Cash Flow to Liabilities: This ratio evaluates liquidity and financial health, influencing investor confidence, which can cause volatility.
    c. Earnings Volatility: Rolling standard deviations of Net Income capture earnings variability over time, which is likely to have a direct effect on stock volatility.

6. Profitability Consistency:

    - EDA Insight: Stability in profitability could provide a key indicator of volatility, with larger fluctuations in profits likely leading to more stock volatility.

- Reason for Extraction: By tracking the difference between consecutive quarters' Gross Profit, we can assess how consistent a company's earnings are over time.

Data - Transformation

Based on the column skewness we perform log transformation to a few columns and then we want all our columns to be scaled to a common range.

Log transformation helps in normalising skewed distributions, which improves model performance by reducing the dominance of outliers. Scaling is important because many machine learning models (e.g., gradient boosting) are sensitive to feature ranges, and a common range ensures that no feature disproportionately influences the model.

Feature Selection

After performing Exploratory Data Analysis (EDA) and Feature Engineering, we expanded our feature set significantly. To enhance prediction accuracy and eliminate variables that don't contribute effectively to the target prediction, we employed three feature selection methods: Hybrid Feature Selection, Lasso Feature Selection, and Random Forest Feature Selection. We opted for the Hybrid approach in Sequential Feature Selection to combine the advantages of both forward and backward selection. However, with 613 stocks requiring one-hot encoding, this method proved computationally expensive, leading us to focus on the top 20 features. Next, we applied Lasso Feature Selection, which shrinks the coefficients of less important features to zero. Initially, cross-validation provided an optimal alpha value, but since many features exhibited non-linear relationships with the target, we lowered the alpha to retain more significant features. We also experimented with Random Forest Feature Selection, where feature importance is ranked using Gini impurity. As this method handles non-linear relationships effectively, it was a strong candidate. Ultimately, the feature sets from all three methods were quite similar, but Lasso Feature Selection produced the best validation RMSE. Its ability to handle non-linearities while offering greater interpretability made it preferable to Random Forest, which was more computationally expensive. Additionally, correlation analysis showed limited multicollinearity, reducing the risk of Lasso's known drawback.

After selecting the most important features, we created lagged features to address the lack of data for November 2023. Since we have data from January to October 2022, we leveraged previous months' values to generate volatility predictions. For monthly updated features, we created a single lag using the previous month's data. For quarterly updated features, we generated two lagged features—lag-1 and lag-3—to capture both recent and seasonal trends, ensuring a more comprehensive prediction model.

Model Selection:

1. Support Vector Regression (SVR): is a regression algorithm that classifies data using the same ideas as Support Vector Machines (SVM). The SVR model selects the best line within an epsilon margin of tolerance, relying on a limited number of data points, or support vectors, to inform its decision.

Why SVR Works for Volatility Prediction:

- Since volatility and characteristics (such as price changes) sometimes exhibit complicated patterns, SVR's kernel technique aids in capturing this non-linearity in stock market data.
- Support vectors improve SVR's capacity to generalize on previously unseen data by helping it concentrate on important data points and strengthening its resistance to noise.

2. Random Forest Regressor: Random Forest is an ensemble learning method that mixes numerous decision trees, each trained on a random selection of characteristics and data. It returns the average prediction of all trees.

Why Random Forest Works for Volatility Prediction:

- Random Forest succeeds at handling high-dimensional data and detecting non-linear correlations, which are common in stock market data.

- The ability to select the most essential features (such as historical stock price or volume) allows the model to focus on the aspects that have the greatest influence on volatility.
- Its ensemble nature ensures that the model performs effectively even when dealing with volatile and noisy data.

3. Gradient Boosting Regressor (GBR): is an ensemble learning technique that builds decision trees sequentially. Each subsequent tree seeks to rectify the faults (residuals) made by the preceding trees, thereby increasing the model's precision.

Why Gradient Boosting Helps Predict Stock Volatility:

- **Correcting Residuals:** Stock volatility can be influenced by many variables with complex relationships. Gradient Boosting iteratively corrects its predictions by building on past model errors, allowing it to efficiently **manage such complex, nonlinear relationships**.
- **Fine-Tuning through Hyperparameters: RandomizedSearchCV** helps optimize the model by finding the best set of hyperparameters (e.g., number of trees, depth of trees, learning rate). This ensures that the model performs well without overfitting or underfitting the data.
- **Robustness to Overfitting:** The combination of small learning rates and subsampling reduces the risk of overfitting, ensuring that the model generalizes well on unseen data. This is crucial for handling noisy stock market data, where overfitting could lead to poor predictions on real-world volatility.

Cross-Validation Setup: For each model, we applied **5-fold cross-validation** to ensure that the model's performance is reliable and generalises well across different subsets of the data. This process works as follows:

- The dataset is split into 5 equal parts.
- In each fold, 4 parts are used for training, and 1 part is used for validation.
- The model is trained 5 times, each time using a different part for validation and the rest for training.
- After training, the performance scores i.e RMSE from all 5 folds are averaged, giving a final cross-validation score.

Cross-validation helps prevent overfitting by ensuring that the model does not learn too much from one specific subset of data. It also gives a good estimate of how the model will perform on unseen data, which is critical in stock volatility prediction tasks.

Hyper-Parameter Tuning: For all models, we applied RandomizedSearchCV, which is a form of combined hyperparameter tuning. This technique randomly samples from a hyperparameter grid to find the best performing combination for the model. By combining different parameter settings for each model, we ensured that the models were fine-tuned for optimal performance.

Model Comparison:

- **Random Forest Regressor** performs best overall, with the lowest **Test RMSE (0.9461)** and good stability across different ranges of volatility. It excels at handling non-linear relationships and provides solid generalisation, making it the most reliable model for stock volatility prediction.
- **Support Vector Regressor** with **Test RMSE (0.9772)**also performs well for low volatility but shows a noticeable decrease in accuracy as volatility increases. This makes SVR suitable for **less volatile markets**, but not as effective for capturing large swings in volatility.
- **Gradient Boosting Regressor** struggles more than the other models, with the highest **RMSE (1.4528)**. While it can model complex relationships, its higher test error suggests potential overfitting or that the model struggles to capture the full range of volatility dynamics.

Error Analysis:

- For **low to medium volatility** (0–5), all models show reasonably good performance, with predictions close to the true values.
- For **higher volatility** (above 6), all three models show increased prediction errors, but **Random Forest** handles the extreme volatility better than GBR and SVR, as shown by its tighter fit and fewer deviations from the ideal line.

Visual Insights:

- The **plots** for each model show a clear trend: predictions are accurate for **low volatility values**, but the models start to deviate as volatility increases. This is particularly true for **Gradient Boosting**, where the scatter points increasingly deviate from the red line above volatility values of 10.
- The **Random Forest** scatter plot shows the least deviation, indicating that the model generalises better across the volatility spectrum.

Conclusion:

- **Random Forest Regressor** emerges as the top-performing model for predicting stock volatility, given its **lower RMSE**, **better generalisation**, and **stable performance across volatility ranges**.
- **Support Vector Regressor** is a good choice for scenarios with **low to moderate volatility** but struggles with higher volatility values.
- **Gradient Boosting Regressor**, while powerful, seems to require additional tuning to avoid overfitting or better capture the full range of volatility values.