GROUP WORK PROJECT 03 MScFE 632: Machine Learning in Finance
GROUP NUMBER: 7820

FULL LEGAL NAME	LOCATION (COUNTRY)	EMAIL ADDRESS	MARK X FOR ANY NON-CONTRIBUTING MEMBER
Egor Prokhodtsev	United Kingdom	e.prokhodtsev@gmail.com	X
Regulavalasa Krishna Vamsi	India	krishnavamsi8262@gmail.com	
Enock Joseph Mwesigwa	Uganda	mwesigwaejoseph@gmail.com	

Statement of integrity: By typing the names of all group members in the text boxes below, you confirm that the assignment submitted is original work produced by the group (excluding any non-contributing members identified with an "X" above).	
Team member 1	**Enock Joseph Mwesigwa**
Team member 2	**Regulavalasa Krishna Vamsi**
Team member 3	

Use the box below to explain any attempts to reach out to a non-contributing member. Type (N/A) if all members contributed.
Note: You may be required to provide proof of your outreach to non-contributing members upon request.

Egor Prokhodtsev was reassigned to another group on 7th December 2024, and so did not contribute to the GWP.

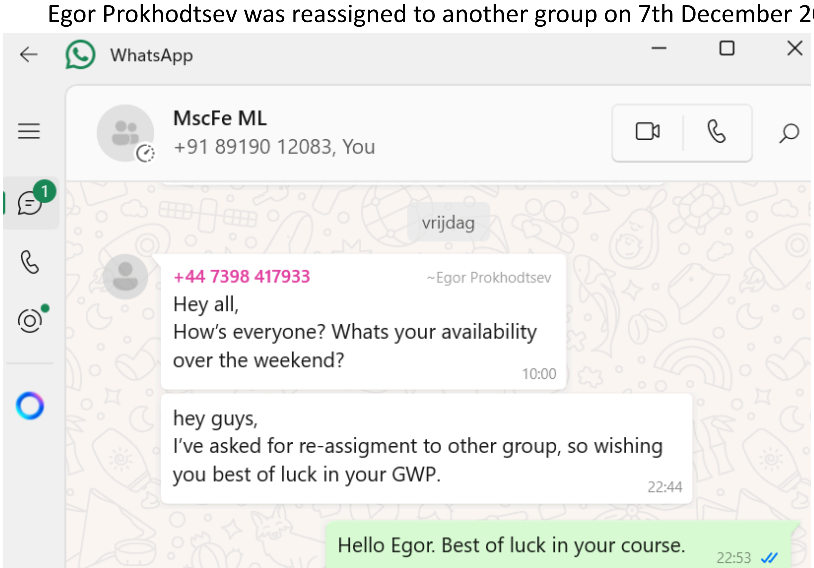


Table of Contents

Step 2: Outline

Issue 1: Optimizing Hyperparameters

Technical Section

- ❖ Model Overview
    - ➢ Hyperparameters
    - ➢ Model Names
- ❖ Techniques
    - ➢ Grid Search
    - ➢ Random Search
    - ➢ Bayesian Optimization
- ❖ Performance Metrics
    - ➢ Evaluation:
    - ➢ Accuracy, Precision, Recall, F1-Score, ROC-AUC
    - ➢ Results: Summarize findings
- ❖ Validation Strategies
    - ➢ Cross-Validation

Non-Technical Section

- ❖ Stakeholder Communication
    - ➢ Hyperparameters: High-level description of what they are and how they affect outcomes.
    - ➢ Optimization Process: General description of how the process works and what methods are used.
    - ➢ Validation & Optimality: Description of how the best version of your model will be chosen
- ❖ Risk Management
    - ➢ Risks: Overfitting, computational intensity, noise sensitivity, and so forth
    - ➢ Behavior: How to tackle these risks properly
    - ➢ Contingency Plans: Something for when things go wrong
- ❖ Ethical Considerations
    - ➢ Data Privacy: Protecting your data ethically
    - ➢ Bias & Fairness: Controlling for bias in your models
    - ➢ Regulations: Laws and industry standards
- ❖ Business Impact Analysis
    - ➢ ROI: Approximate return on investment from optimization.

- ➢ Strategic Alignment: Ways the model supports business objectives.
- ➢ Long-term Vision: The sustaining performance and adaptation to future needs.

Issue 3: Applying Ensemble Learning - Bagging, Boosting, or Stacking

Technical Section

Introduction to Ensemble Learning

- ❖ Bagging (Bootstrap aggregating)
  - ➢ Equations
  - ➢ Implementation
  - ➢ Visualisation and Performance metrics

- ❖ Boosting
  - ➢ Gradient boosting algorithm
  - ➢ Implementation with hyperparameter tuning
  - ➢ Feature importance analysis
  - ➢ Visualisation and performance metrics

- ❖ Stacking
  - ➢ Equations
  - ➢ Implementation in python
  - ➢ Visualisation and performance metrics

- ❖ Comparison of ensemble methods
  - ➢ R-squared scores, Mean squared error, RMSE
  - ➢ Visualisations

Non-Technical Section

- ❖ Overview of ensemble learning methods
- ❖ Bagging
  - ➢ Overview
  - ➢ Key benefits
  - ➢ Performance summary
- ❖ Boosting
  - ➢ Overview

- Key benefits
- Performance summary

- Stacking
    - Overview
    - Key benefits
    - Performance summary
- Practical applications of ensemble methods
- Limitations and considerations
- The way forward and recommendations
- Learn more

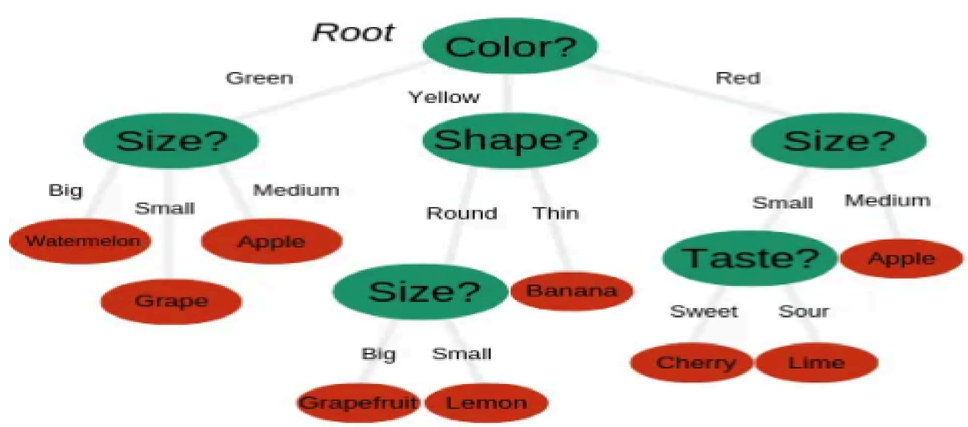
Step 3

Issue 1: Optimizing Hyperparameters

- Wrote by Krishna
- Reviewed by Enock

Model Technicalities
Model Name: Random Forest

It is a supervised learning algorithm based on multiple decision trees. The decision trees have the same nodes, but the subset of data varies for each, which means unique decision outcomes for each. This leads to solving the problem through an average decision outcome of all the trees. Random Forest is used with labeled data to learn and solve classification tasks.
In the following figure, we depict how multiple decision trees classify different kinds of fruits. (Source: Schott)



Hyperparameters and Their Importance

- n_estimators:
  The number of trees in the forest. The more trees, the better the accuracy.
- max_depth:
  How deep each tree can grow. Too deep causes overfitting.
- min_samples_split:
  Minimum data points required to split a tree. Balances overfitting and underfitting.
- Min_samples_leaf:
  Ensure that the leaf has enough data to prevent overfitting.
- Max_features:
  How many features each tree uses. The more features, the more diverse and accurate the model.
- Criterion:
  The rule to split data in the tree. It impacts the performance of the model.

Hyperparameter Optimization Techniques

- Grid Search:
  Grid search searches exhaustively in a given subset of hyperparameters. The best combination is found after training and evaluation.
- Random Search:
  It randomly samples hyperparameters from a distribution. It is much less computationally expensive than grid search and could find good combinations faster.
- Bayesian Optimization:
  This uses a probabilistic model that predicts the performance of different settings of hyperparameters. It is an iterative procedure that focuses on promising areas in the hyperparameter space.

Performance Metrics

- Accuracy: Number of correct predictions out of all the predictions.
  Formula: Accuracy = Correct Predictions / Total Predictions
- Precision: Number of true positives out of all positive predictions.
  Formula: Precision = True Positives / (True Positives + False Positives)
- Recall: Number of true positives out of all actual positives.
  Formula: Recall = True Positives / (True Positives + False Negatives)

- **F1-Score**: The harmonic mean of Precision and Recall.

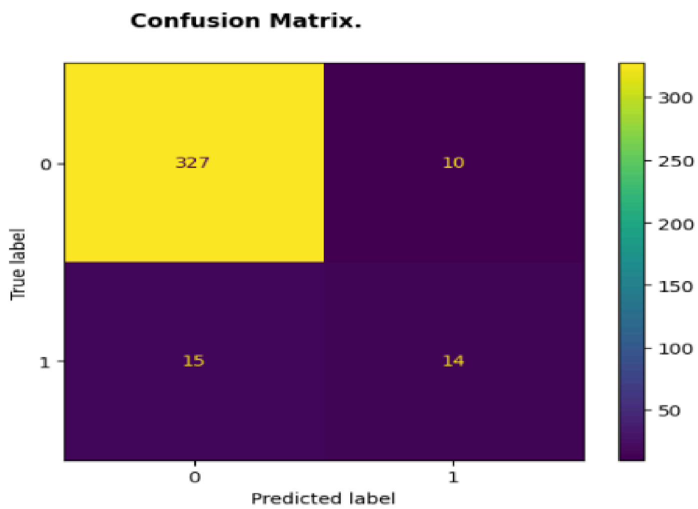
  Formula: F1-Score = 2 × (Precision × Recall) / (Precision + Recall)

Validation Techniques

- Cross-Validation:
  Divides the dataset into k subsets. The model trains and validates k times, each time using a different subset as the validation set.
  Equation:

Optimality Identification

- Comparative Study: Confusion Matrix Analysis
  A confusion matrix is a table that helps evaluate the performance of a classification model in comparison with the actual outcomes of its predictions. It provides an elaborative representation of true positives, false positives, true negatives, and false negatives.



Implementation

Steps of Implementation

- Creating the Data:
  We managed to acquire 5 years of data on the price of Bitcoin, ranging from 2019 to 2024. We further enhanced the data by adding specific rules to enable us to make a guess as to whether the price would increase or decrease.
- Data Correction
  We corrected for missing data so that we could utilize it for our predictions.
- Training the Baseline Model:
  We used a special tool called Random Forest to make a simple guess model. This was our starting point to see how good the guesses are.

- Trying Better Models:
  We then tried many other guesses by changing the settings to see if they could be better using Grid Search, Random Search, and Bayesian Optimization.
- Checking the Performance:

We tested how well each model worked by dividing the data into different parts and checking each one.

- Looking at the Results:
  We used a special table called a confusion matrix to see how many times our guess was correct, and how many times it was wrong.
- Picking the Best Model:
  Finally, we selected the best model based on the good guesses it made at the end; we cared for results such as how many times it got it right (Accuracy), how good it was at finding the correct answers (Precision), and other things
- Results:

	Model	Accuracy	AUC	Log Loss	F1	Precision	Recall
0	Baseline	0.948087	0.974419	0.099631	0.949953	0.952621	0.948087
1	Grid Search	0.945355	0.976261	0.096581	0.947671	0.951097	0.945355
2	Random Search	0.942623	0.974317	0.099236	0.945411	0.949655	0.942623
3	Bayesian Optimization	0.945355	0.975801	0.096457	0.947671	0.951097	0.945355

Interpretations

- **Results Summary**:
  - **Baseline Model**:
    - Performs the best overall with:
      - Accuracy: 94.81%
      - AUC: 97.44%
      - F1 Score: 94.99%
  - **Optimized Models**:
    - **Grid Search**:
      - Accuracy: 94.54%
      - AUC: 97.63%
      - Log Loss is the lowest, suggesting it generalizes well.
    - **Bayesian Optimization**:
      - Accuracy: 94.54%
      - AUC: 97.58%
      - Also performs similarly to Grid Search.
  - **Random Search**:
    - Accuracy: 94.26%

        - AUC: 97.43%
        - Falls slightly behind compared to the other models.
- **Conclusion**:
  The **baseline model** performs similarly to the **optimized models** but is simpler. The **Grid Search** and **Bayesian Optimization** models are a bit better in some areas, but the baseline model remains a strong choice for its simplicity and comparable results.

Non-Tech Section

Stakeholder Communication

- <u>Explanation of Hyperparameters</u>: Hyperparameters are settings we pick before training the model. These are not learned from the data but they control how good or bad the model is.
- In a Random Forest model, some important hyperparameters include the following:
  <u>Number of Trees (n_estimators):</u> It is the number of total decision trees within the forest.
  <u>Maximum Depth (max_depth):</u> It sets a limit to the depth of every tree and it also decides the model complexity.
- Minimum Samples Split (min_samples_split) - Defines the minimum number of samples an internal node must have to be split.
- Minimum Samples Leaf (min_samples_leaf) - Sets the minimum number of samples a leaf node needs to have.
- Maximum Features (max_features) - Specifies how many features to consider when finding the best feature to make the split.

Optimization Process Overview:

- Optimization process refers to finding the best set of hyperparameters that enhance the performance of the model. This is achieved through techniques such as Grid Search, Random Search, and Bayesian Optimization.
- Steps in the process:
  Defining the Search Space: This is the process of setting the range of values for each hyperparameter.
  Selecting an Optimization Technique: This is the process of choosing the method to explore the hyperparameter space (e.g., Grid Search, Random Search, Bayesian Optimization).
  Training and Validation: Running the model with various hyperparameter combinations and evaluating it using a validation set.
  Selecting the Best Configuration: Picking the hyperparameter combination that gives the highest performance (e.g., accuracy, F1 score, precision, recall).

Validation and Optimality Criteria:

- Cross-Validation: The data is divided into several folds, and the model is trained on different subsets to ensure generalization.
- Optimality Criteria: Key metrics include accuracy, precision, recall, F1 score, and ROC-AUC. The goal is to find a balance between complexity and performance, avoiding overfitting while maximizing predictive ability.

Risk Management:

In optimization, risk management is involved with handling the problems of overfitting, underfitting, resource strain, and possible bias in the choice of hyperparameters. Overfitting means that the model works very well on training data but fails on new data. Underfitting happens when the model is too simple to capture the important patterns. The optimization process is also very time-consuming and demanding on resources, and the hyperparameters we choose might introduce bias. To address these issues, we use techniques such as early stopping, which prevents overfitting; data augmentation increases the variety in the training set. Parallel processing speeds up, and continuous monitoring keeps an eye on fairness. Backup plans prepare alternative hyperparameters, and tools help track model performance to be prepared in case things don't go well.

Ethical Consideration and Business Impact Analysis:

For the ethical considerations, we ensure our optimization process complies with the rules of data privacy such as GDPR and CCPA; we anonymize the data and store it in a secure environment. We further identify and minimize bias in the model by doing fairness checks, audits, and being transparent about the limitations. For business impact, we consider the costs associated with data collection and training in contrast to the benefits in terms of better accuracy and efficiency. Our model is closely aligned with business goals, and there's ongoing feedback from stakeholders to ensure that these address critical issues. We focus in the long run on sharpening the model to keep up with things as they change and scaling it to keep driving growth and profitability.

How does Machine Learning Provide Value?

ML tools are quite valuable in solving hyperparameter optimization challenges. Fine-tuning the hyperparameters increases the accuracy and robustness of models such as Random Forests, thus providing better predictions. This results in much more accurate forecasting, reduced errors, and improved risk management. Hence, it would ideally predict market trends with more accuracy, find profitable opportunities, and portray early signs of downturns that would easily support more informed decision making, increased returns, and minimized losses. Systematic hyperparameter optimization through better efficiency and scalability in model development has a positive effect on the process. Essentially, advanced ML tools enhance the investment strategy's ability to adapt to the rapidly changing financial scenario.

Issue 3: Applying Ensemble Learning - Bagging, Boosting, or Stacking

- Wrote by Enock
- Reviewed by Krishna

Technical section

Introduction to Ensemble Learning

The defining feature of ensemble learning is the combination of multiple models to improve the overall predictive performance and robustness. The strengths of multiple models are combined and in the process, overall bias is reduced. Ensemble methods are based on the premise that many weaker models can be combined to perform better than a single strong model.

For N base models $f_1(x)$, $f_2(x)$,..., $f_N(x)$, we can express the prediction of the ensemble method as:

$$\hat{y} = \sum_{i=1}^{N} w_i f_i(x)$$

Where w_i are the weights assigned to each base model.

The major ensemble learning methods are bagging, boosting, and stacking.

Bagging

Overview

Bagging, also known as bootstrapping works by reducing variance through training the same model many times using bootstrapped datasets and then finding the mean of the predictions. Bootstrapping involves creating random and multiple subsets of the original dataset. Bagging can be used with decision trees, and support vector machines which have high variance to prevent overfitting.

Equations

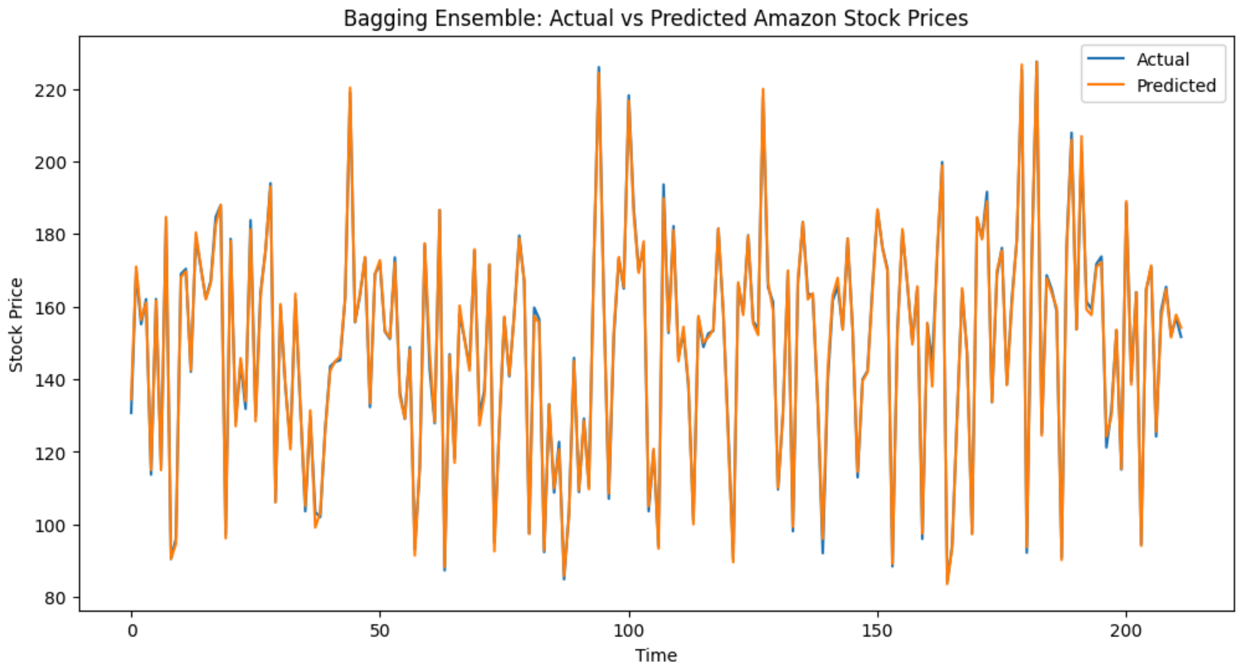
If f_1, f_2,..., $f_M(x)$ are M different models, each trained on a bootstrap sample of the dataset, the bagged predictor is expressed as:

$$f_{bag}(x) = \frac{1}{M}\sum_{i=1}^{M} f_i(x)$$

Implementation

Bagging was used on decision trees to predict Amazon stock using five-year historical financial data from Yahoo Finance (refer to the attached Python Jupyter Notebook for the code).

Visualisation and performance metrics.



The plot shows how closely and accurately the bagging model predictions follow the actual prices. The performance metric table below further highlights how accurate the model's prediction is considering the very high R-squared value very close to 1 and the low RMSE of USD 1.36.

Metrics of the bagging model

Mean squared error	1.85
Root Mean Squared Error	1.36
R-squared	0.998

Boosting

Overview

Boosting works by sequentially training many models to reduce bias. Each model is focused on correcting errors of the previous models. It gives more weight to prior misclassifications in the next iterations. It uses weak learners to give more accurate predictions. It can be used with gradient boosting, adaptive boosting and extreme gradient boosting. It is very effective with complex models having high bias, such as decision trees with shallow depth.

Equations

A weighted boosting ensemble can be expressed as:

$\hat{y} = \sum_{i=1}^{M} \alpha_i f_i(x)$ where α_i is the updated weight from model accuracy.

Gradient boosting algorithm

Our illustration of boosting will use a gradient-boosting algorithm that is expressed as: $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$ where $F_m(x)$ is the model at iteration m , $h_m(x)$ is the weak learner, and γ_m is the step size.

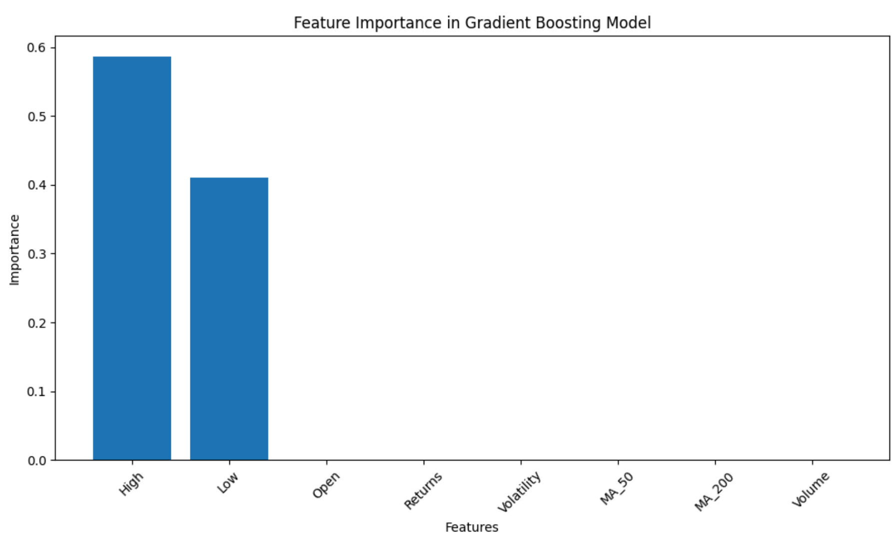
Implementation

We used the same dataset of Amazon's five-year stock data to predict the stock price. Grid search was used to tune the hyperparameters for the gradient boosting model. Refer to the attached Python Jupyter Notebook for the particular code.

The best parameters from the tuning are displayed in the table below.

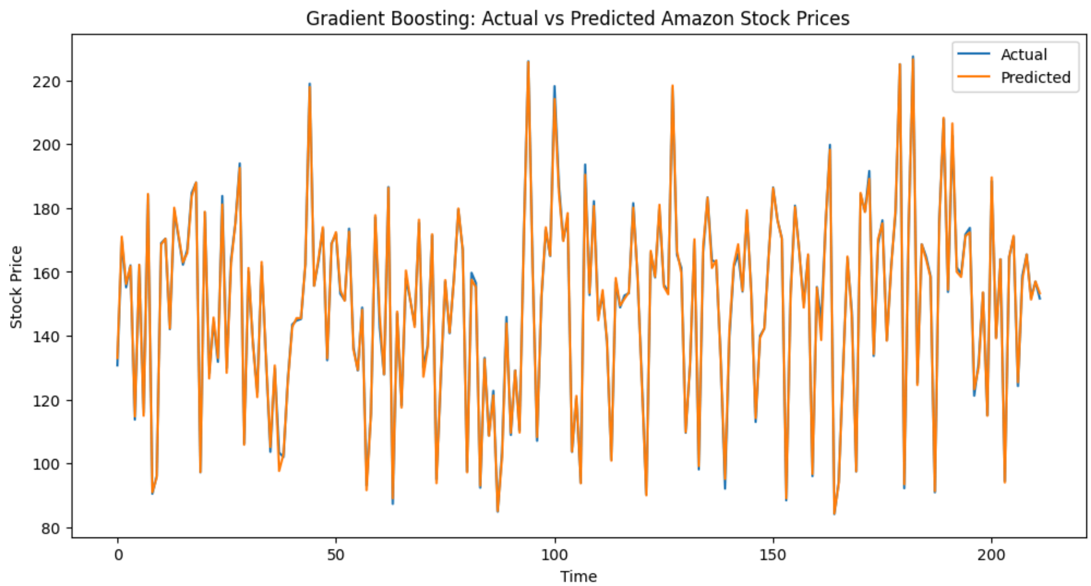
Learning rate	Maximum depth	Minimum samples leaf	Minimum samples split	Number of iterations
0.1	4	1	5	100

Feature importance analysis



The feature importance plot above displays the importance of each feature in the model, with the High and Low being the two most important.

Visualisation and performance metrics



The plot above shows how the boosting model predictions compare with the actual stock prices. The table below shows the model's performance metrics, showing how highly accurate its predictions are.

Metrics for boosting model

Mean squared error	1.60
Root Mean squared error	1.27
R-squared	0.998

Stacking
Overview

Stacking uses a meta-model to combine predictions from heterogeneous models. Its main strength is in its ability to work with different models and datasets, making it very flexible and indispensable with high-dimensional datasets.

Equations

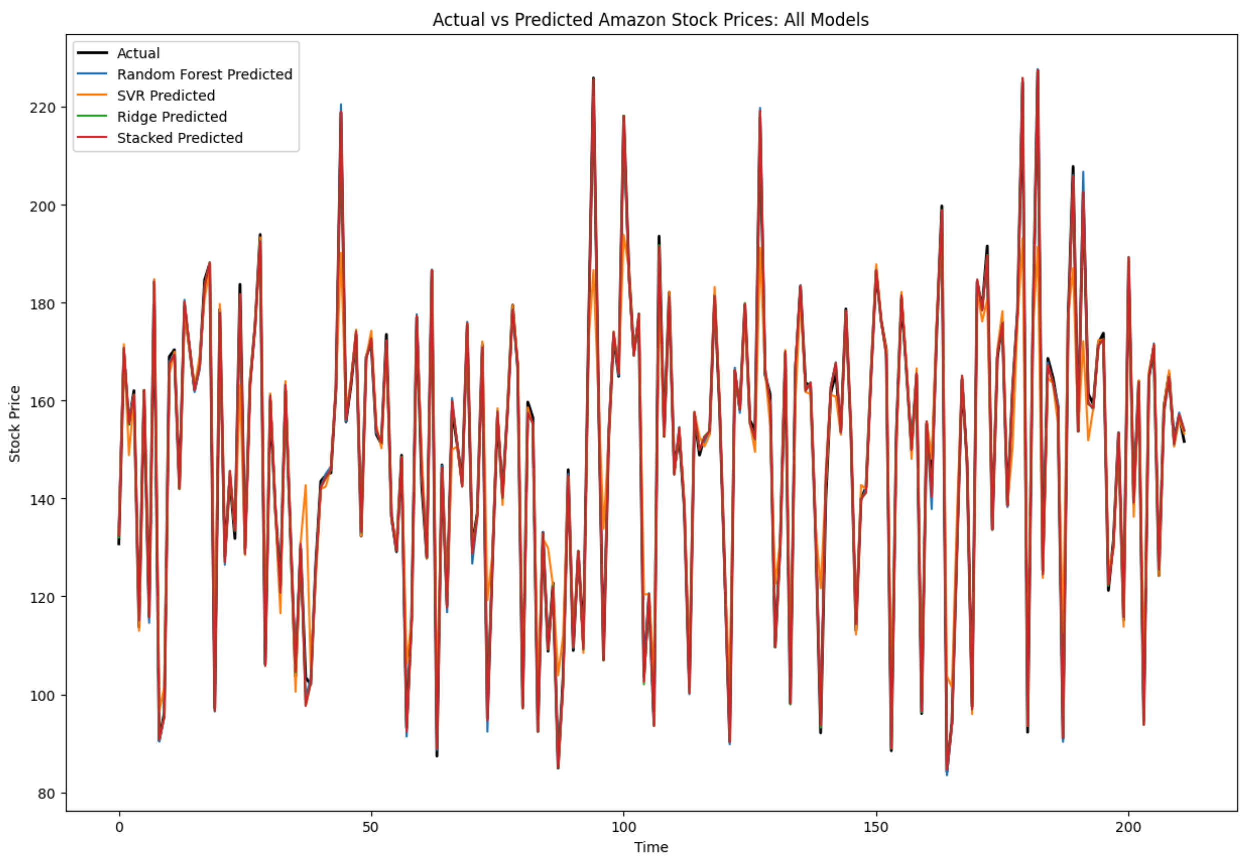
If base models are expressed as $f_1(x)$, $f_2(x)$,..., $f_N(x)$ the meta-model $g(x)$ learns from all of them as expressed below:

$$\hat{y} = g(f_1(x),\ f_2(x),...,\ f_N(x))$$

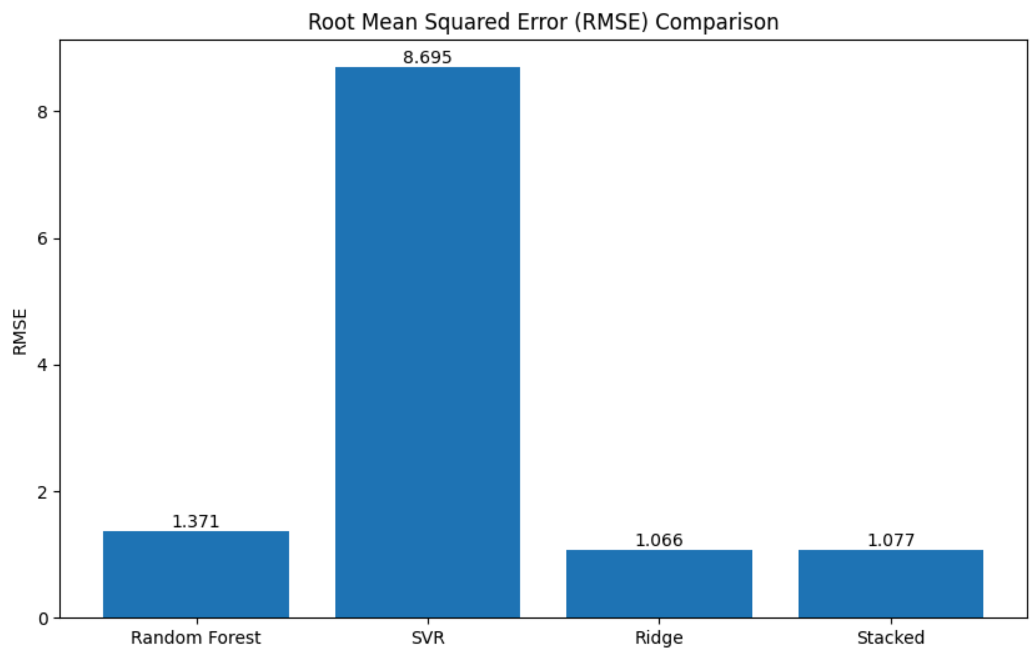
Implementation

Using the same dataset as before, we trained with cross-validation a stacking model with three sub-models: Random forest regression, support vector machine (SVM), and ridge regression model. Refer to the attached Python Jupyter Notebook for the particular code.

Visualisation and performance metrics



The plot above displays the high prediction accuracy of the stacking model and its constituent base models. The plot below displays the comparison of the RMSE values for the stacking model and its constituent base models. The stacking model has a low RMSE value, showing that it has more accurate predictions than for example SVM, one of its base models.



The stacking model also has a highly accurate predictive capability, as shown by the metrics in the table below.

Metrics for the Stacking Model

Mean squared error	1.16
Root Mean Squared error	1.08
R-squared	0.999

Comparison of the ensemble models Performance

Model	Mean squared error	RMSE	R-squared
Bagging model	1.85	1.36	0.998
Boosting model	1.60	1.26	0.998
Stacking model	1.16	1.08	0.999

Based on the implementation of the three ensemble methods to the same task of predicting Amazon's stock price, all methods performed very well with very slight differences in performance metrics as shown in the table above.

Features

Feature	Bagging	Boosting	Stacking
Definition	Uses average or majority vote from multiple models to reduce variance	Sequentially improves weak models by correcting errors to reduce bias	Meta-model generalises predictions from diverse models.
Model type	Homogeneous	Homogeneous	Heterogeneous
Approach	Parallel training of models on bootstrapped datasets	Sequential training, each model improving on the previous one	Predictions from base models are inputs for training the meta-model
Weighting of base models	Equal weights	Weighted according to performance	Learns optimal combination of base model predictions
Strengths	Robust against overfitting	Reduces bias, improving predictive power	Combines strengths of diverse models
	Works well with high-variance models	Can capture complex patterns	Flexible, can be used with any model type
Training complexity	Low since models train in parallel	Moderate because of sequential training of models	High because of training the base models and meta-model

Non-Technical Section

The portfolio strategists asked, “How can the models be used together?” In response, we have explored three ensemble methods and used them in the prediction of the stock price of Amazon Inc., where they have performed with excellent performance metrics. We will give a brief overview of each ensemble method.

Bagging

Bagging can be compared to a team of 100 financial analysts, each analysing a randomly drawn, slightly different subset of Amazon’s historical data. Bagging creates many versions of our prediction model and trains each on a different sample from the historical data. The average of the predictions of all the models is the final prediction of the bagging model. Our bagging model used decision trees for its base models and was highly accurate in predicting Amazon’s stock price, with an average deviation of only USD 1.38 from the actual prices.

Key benefits

- Improves stability and accuracy of the predictions
- It is very effective in handling volatile stock prices.
- It can accurately model rare patterns in the data

Boosting

Overview

Boosting is like having a team of junior financial analysts who learn from their mistakes. Boosting starts with a weak learner, and sequentially adds more models that focus on correcting the errors of the previous ones. Boosting is therefore excellent at capturing complex patterns in the stock price movements that may be overlooked by simpler models. The boosting model’s predictions have an average deviation from the actual prices of only USD 1.26. The boosting model also correctly identified the two important features in the data set as the High and Low features.

Key benefits

- Captures complex relationships in the data
- Adapts to changes in market conditions
- Provides more accurate predictions than most single models.

Stacking

Overview

Stacking can be compared to assembling a panel of experts with different specialities, and then having a super-expert who can combine all their predictions in a way to give the best joint prediction. We used three different model types as base models, then the meta-model combined their predictions to give a final prediction. This was the best model of all three with the lowest average deviation from the actual prices of USD 1.08.

Key benefits

- Comines strengths of different modeling techniques
- Usually performs better than single models
- Stable and robust predictions.

Practical applications

- Informed decision-making: Predictions of ensemble methods can be used to reliably guide decision-making on the investment floor about when to buy, hold, or sell stock. If the model predicts a significant price increase, it indicates an opportunity to buy now and sell when the increase occurs.
- Risk management: Ensemble methods provide accurate and stable predictions that can enable accurate assessment and management of risk in the stock. This can inform the setting of precise stop-loss orders and the size of the investing position.
- Portfolio optimization: The predictions can be used to optimize stock weights in investment portfolios. For example, if the predictions are favourable, it can inform an increase in the portfolio allocation of Amazon stock.
- Predictive modeling: Ensemble methods can be used to simulate possible scenarios by adjusting datasets, features, models, and hyperparameters. This enables the company to effectively prepare for many potential outcomes, creating potential competitive advantages.
- Performance benchmarking: The predictions can reliably be used as a benchmark against which the performance of other trading strategies and human analysts can be measured.

Limitations

- Ensemble methods are usually more computationally intensive than single models and may therefore cost more to operate.
- Ensemble methods are rather “black box” in that they are not easily interpretable, with difficulty in determining how the final prediction was obtained by the model.
- Ensemble methods are slaves to the data. For example, boosting is sensitive to outliers, and bagging requires variability in the data for bootstrap sampling to be effective.
- Ensemble methods also rely on the quality of the input features where poor feature engineering can affect the model performances. Hence they require expert personnel to do accurate feature engineering.

- Ensemble models are usually bulky in size due to many base models, further increasing the cost of storing and deploying them.

Considerations

- Ensure that the base models used in bagging or stacking have some variability in their features and settings. Similar base models reduce the benefits of ensemble learning.
- Hyperparameters in the ensemble models especially bagging and boosting need to be carefully tuned and optimized. Stacking also requires careful selection of the meta-model and its parameters to avoid overfitting.
- Use cross-validation techniques such as k-fold to evaluate the performance of the ensemble to prevent overfitting, and increase robustness.
- A good understanding of the dataset is needed to apply the right ensemble method. For example, bagging works with high variance models, and boosting works with high-bias models.
- Ensemble methods usually require large datasets to be effective, because they train multiple models.

Way forward and recommendations

- Use bagging methods when predicting volatile stock where reducing variance is critical.
- Use boosting methods such as gradient boosting, and XGBoost for short-term trend analysis where you need to capture subtle market patterns.
- Use stacking where you need a well-rounded perspective when making decisions as it combines different models and data.
- During communication, focus on the tangible benefits of ensemble methods to improve interpretability.

Learn more

Here are some journal articles that can be used to learn more about the features, applications, challenges, and improvements of ensemble learning methods in finance.

1. Belhadi, Amine, et al. "An ensemble machine learning approach for forecasting credit risk of agricultural SMEs’ investments in agriculture 4.0 through supply chain finance." *Annals of Operations Research* (2021). <https://doi.org/10.1007/s10479-021-04366-9>.
2. Carta, Salvatore, et al. "A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning." *Applied Intelligence* 51.2021 (2020): 889-905. <https://doi.org/10.1007/s10489-020-01839-5>.
3. Ghosh, Indranil, et al. "Prediction and interpretation of daily NFT and DeFi prices dynamics: Inspection through ensemble machine learning & XAI." *International Review of Financial Analysis* 87 (2023): 102558. <https://doi.org/10.1016/j.irfa.2023.102558>.

4. Li, Yang and Yi Pan. "A novel ensemble deep learning model for stock prediction based on stock prices and news." *International Journal of Data Science and Analytics* 13.2022 (2021): 139-149. <https://doi.org/10.1007/s41060-021-00279-9>.
5. Sonkavde, Gaurang, et al. "Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications." *International Journal of Financial Studies* 11.3 (2023): 94. <https://doi.org/10.3390/ijfs11030094>.

STEP 4

Student Reviews:

Student	Wrote	Reviewed
Team Member A - Krishna	Issue 1	Issue 3
Team Member B - Enock	Issue 3	Issue 1

Team member A(Issue 3 review):

All components of the Bagging model are described accurately; the process of bootstrapping, subset creation, and aggregation of predictions is well explained. Hyperparameters, particularly in Random Forest, are aptly explained

Clarity and Readability:

Clearness on the text-the explanations on both bagging and performance metrics canwell come out. Separating technical and nontechnical sections makes perfect sense in structural terms.

Spelling and Grammar:

There are no egregious spelling mistakes on major sentences. The rest is minor errors in grammar sentence

Suggestions to Improve: Clarify the response for better production quality and writing.

- Consistency: Use consistent terminology like "model" and "classifier."
- Detailed Explanation: Add simple examples or analogies for bootstrapping and Out-of-Bag score to enhance understanding.

Team Member B(Issue 1 Review):

Accuracy and Technical Correctness:

The description of the Random Forest model is correct, describing how multiple decision trees are used and aggregated. The importance of hyperparameters is explained well. The optimization techniques of Grid Search, Random Search, and Bayesian Optimization for hyperparameter tuning are appropriately mentioned. The choice of performance metrics is correct, and their explanations are clear, along with the implementation steps and result interpretations.

Clarity and Readability:

The language is simple and accessible for both technical and non-technical members. The structure is well-organized, making the information easy to understand.

Spelling and Grammar:

No spelling or grammar errors found.

Suggestions for Improvement:

- Clarity: Some sentences could be paraphrased for more precision.
- Examples: Including more specific examples would help explain certain concepts better.

STEP 5:

The GWP1 document has been revisited and updated, and a copy of the revised version is included in the zip folder of the GWP3 results.

References

1. Belhadi, Amine, et al. «An ensemble machine learning approach for forecasting credit risk of agricultural SMEs' investments in agriculture 4.0 through supply chain finance.» *Annals of Operations Research* (2021). <https://doi.org/10.1007/s10479-021-04366-9>.
2. Carta, Salvatore, et al. «A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning.» *Applied Intelligence* 51.2021 (2020): 889-905. <https://doi.org/10.1007/s10489-020-01839-5>.
3. Derbentsev, V., et al. «Comparative Performance of Machine Learning Ensemble Algorithms for Forecasting Cryptocurrency Prices.» *International Journal of Engineering* 34.1 (2021): 140-148.

4. Ghosh, Indranil, et al. «Prediction and interpretation of daily NFT and DeFi prices dynamics: Inspection through ensemble machine learning & XAI.» *International Review of Financial Analysis* 87 (2023): 102558. <https://doi.org/10.1016/j.irfa.2023.102558>.
5. He, Kaijian, et al. «Financial Time Series Forecasting with the Deep Learning Ensemble Model.» *Mathematics* 11.4 (2023): 1054. <https://doi.org/10.3390/math11041054>.
6. Li, Yang et Yi Pan. «A novel ensemble deep learning model for stock prediction based on stock prices and news.» *International Journal of Data Science and Analytics* 13.2022 (2021): 139-149. <https://doi.org/10.1007/s41060-021-00279-9>.
7. Luo, Cuicui. «A comparison analysis for credit scoring using bagging ensembles.» *Expert Systems* 39.2 (2018). <https://doi.org/10.1111/exsy.12297>.
8. Siswoyo, Bambang, et al. «Ensemble machine learning algorithm optimization of bankruptcy prediction of bank.» *IAES International Journal of Artificial Intelligence* 11.2 (2022): 679-686.
9. Sonkavde, Gaurang, et al. «Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications.» *International Journal of Financial Studies* 11.3 (2023): 94. <https://doi.org/10.3390/ijfs11030094>.
10. Arya, Nisha. "Tuning Random Forest Hyperparameters." KDnuggets, 22 August 2022, https://www.kdnuggets.com/2022/08/tuning-random-forest-hyperparameters.html. Accessed 15 October 2024.
11. Schott, Madison. "Random Forest Algorithm for Machine Learning | by Madison Schott | Capital One Tech." Medium, 25 April 2019, https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb. Accessed 15 October 2024.