Artificial Intelligence Assignment

Jane Street Real-Time Market Data Forecasting

Paul Fontanges; Stu No. : 461720

Prepared for 24 March , 2025

Paul Fontanges
Student Number : 461720

Abstract

Financial market prediction is a complex and challenging task due to the dynamic and non-stationary nature of market data. In this study, we explore various machine learning techniques to predict market behavior using anonymized trading data from the Jane Street competition. We investigate both distributed computing approaches, leveraging PySpark's Gradient Boosted Trees and Random Forest models, and traditional machine learning libraries, including XGBoost, CatBoost, and Scikit-Learn.

Our methodology includes extensive data preprocessing, feature engineering, and model evaluation using the sample-weighted zero-mean R-squared (R^2) metric. We assess model performance across different financial instruments identified by `symbol_id`, ensuring asset-specific predictions.

The results highlight the superiority of the Random Forest model, particularly in its Scikit-Learn implementation, which achieved the highest R^2 scores. While PySpark models are effective for handling large datasets, their computational cost is significantly higher compared to optimized implementations like XGBoost and CatBoost. Our findings underscore the importance of selecting the appropriate machine learning framework for financial forecasting, balancing accuracy and computational efficiency. Finally, we discuss the ethical implications of AI-driven trading, emphasizing the need for responsible model deployment to ensure market stability and fairness.

Paul Fontanges
Student Number : 461720

Contents

List of Figures

1 Introduction

1.1 Background

Modern financial markets present unique challenges for predictive modeling. Unlike other domains, financial time series often exhibit nonstationary, fat-tailed distributions, and frequent regime shifts. Traditional statistical models, which rely on assumptions of stationary and normality, often struggle to generalize in such environments.

In addition, financial markets are influenced by a combination of economic, geopolitical, and technological factors, making them inherently dynamic. Automated trading strategies must continually adapt to changing market conditions, making robust predictive models a key component of algorithmic trading.

In this context, machine learning has become an essential tool for modern financial firms. Companies like Jane Street leverage advanced quantitative models to inform trading decisions across thousands of financial instruments and more than 200 trading venues arouns the world. By applying data-driven techniques, traders aim to capture patterns in market data and optimize trade execution.

1.2 Problem

The goal of this challenge is to develop a machine learning model capable of predicting market behavior using anonymized real-world trading data. The data set consists of multiple features that represent market conditions and a set of response variables that capture trading outcomes.

To ensure fairness and maintain competitive integrity, some characteristic names and values have been obfuscated. However, the core problem remains representative of real-world trading scenarios. The challenge lies in extracting meaningful signals from anonymized data while handling issues such as missing values, noisy data, and complex temporal dependencies.

1.3 Objectives

The primary objective of this project is to predict the response variable `responder_6` for each of the 39 unique financial instruments identified by `symbol_id`. The final predictions will be compiled into a structured CSV file, mirroring the format of `sample_submission.csv`.

To evaluate the performance of the model , we will use the zero mean R-squared score (R^2), a statistical measure that assesses how well the predicted values match the actual responses. The model should aim to maximize R^2, ensuring that the predictions capture the underlying market dynamics as accurately as possible.

2 Literature Review

The application of machine learning in stock market prediction challenges the Efficient Market Hypothesis (EMH), which suggests that stock prices fully reflect all available information, making past price analysis ineffective. Traditional econometric models such as autoregressive methods, moving average models, and generalized autoregressive conditional heteroskedasticity (GARCH) have long been used to analyze financial time series. However, their limited ability to capture non-linearity and complex market dynamics has led to the adoption of more sophisticated machine learning techniques.

Deep learning approaches, including autoencoders, restricted Boltzmann machines, and recurrent neural networks, have been explored for stock price forecasting. Among these, Long Short-Term Memory (LSTM) networks have demonstrated superior performance in capturing temporal dependencies in financial data compared to traditional classification models such as logistic regression and random forests. However, ensemble methods, particularly random forests and boosting algorithms, have been shown to be highly effective in certain market conditions, especially during periods of high volatility.

Feature engineering techniques have also played a critical role in improving predictive accuracy. Dimensionality reduction methods, such as principal component analysis (PCA), have been integrated with artificial neural networks to enhance feature selection. Additionally, the use of historical stock price patterns and time-series segmentation has led to more robust predictive models. More recently, the adoption of transformer-based architectures has improved the ability to model long-range dependencies in financial time series, offering a promising alternative to recurrent neural networks.

Despite these advancements, there is no definitive consensus on the optimal approach for stock market prediction, as model performance is highly dependent on the dataset characteristics and market conditions. This study builds upon existing methodologies by evaluating different machine learning models applied to financial data, with a focus on predicting market

behavior based on anonymized trading signals.

3 Methodology

3.1 Dataset Analysis

To develop a predictive model for financial market forecasting, we first conducted an extensive analysis of the dataset. The dataset is stored in multiple Parquet files under the `train.parquet` directory, which consists of ten separate files ranging from 450MB to 1.6GB in size.

3.1.1 Dataset Structure

The dataset includes several key types of columns:

- **Temporal columns**: `date_id` and `time_id`, which provide a chronological structure to the data. However, the actual time intervals between different `time_id` values are not necessarily consistent.
- **Identification column**: `symbol_id`, which represents a unique financial instrument.
- **Weight column**: `weight`, used as a weighting factor for calculating the scoring function.
- **Feature columns**: `feature_00` to `feature_78`, which contain anonymized market data.
- **Response columns**: `responder_0` to `responder_8`, which represent anonymized response variables clipped between -5 and 5. The primary target variable for this study is `responder_6`, which we aim to predict.

3.1.2 Symbol ID Distribution

Before training our model, it was crucial to verify the number of distinct `symbol_id` values and ensure that each had sufficient data points. The dataset contains exactly 39 unique `symbol_id`, each associated with at least 600,000 data points, as illustrated in Figure 1.

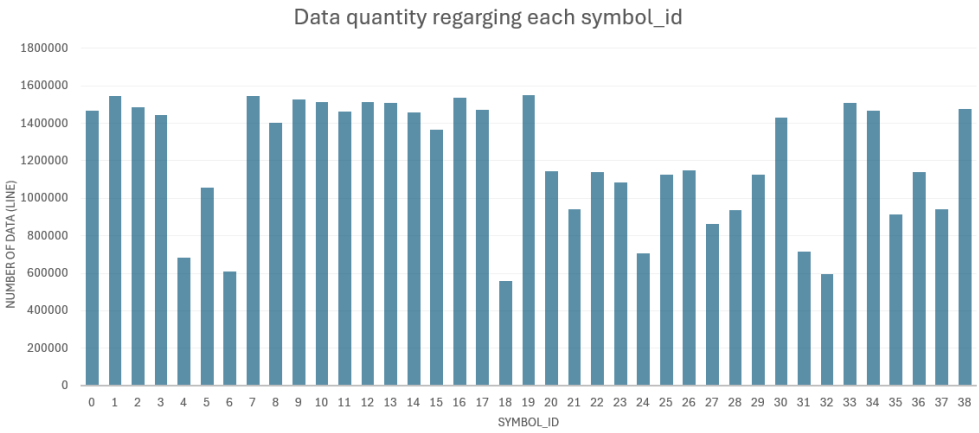

Figure 1: Number of data points per `symbol_id`. Each has at least 600,000 observations, which is sufficient for model training.

3.1.3 Handling Missing Data

A preliminary data inspection revealed that some rows contained missing values (`NaN`) in multiple feature columns. To quantify this, we performed an analysis on the number of fully usable data points, rows without any missing values in the feature set. The results are as follows:

- **Total number of rows**: 47,127,338
- **Number of complete rows (without `NaN`)**: 35,370,822
- **Percentage of complete rows**: 75.05%

This analysis showed that each `symbol_id` retains at least 400,000 complete rows, which is deemed sufficient for training purposes.

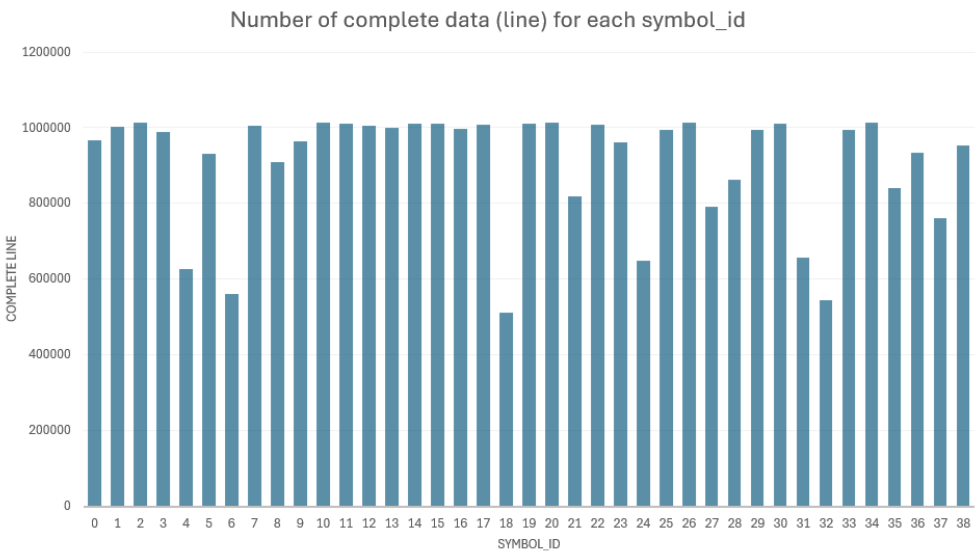

Figure 2: Number of fully complete data points per `symbol_id`.

3.1.4 Creation of a Clean Dataset

Given that a significant portion of the dataset contains missing values, we opted to create a "clean dataset" that includes only fully complete rows. This decision was made to streamline the training process and enhance computational efficiency. The clean dataset was stored in a separate file and used for all subsequent model training steps. The fact is that the dataset is still very large, so we will create a smaller one because training the models takes more than 12 hours, which exceeds Kaggle's limit. After a quick analysis, there are approximately more than 800 data rows per day for each `symbol_id`. Therefore, we will create a smaller dataset that will contain 100 randomly selected rows per day, which is still sufficient for training a model while significantly reducing the dataset size.

3.2 Modeling Approach

To address the problem, we will adopt a multi-faceted approach to model training and evaluation.

First, since the dataset was created using the PySpark library, we will leverage two prediction models available in this framework: `Gradient Boosted Trees` and `Random Forest Regressor`. These models will be trained and evaluated using Spark's distributed computing

capabilities to efficiently process large amounts of data.

Second, in order to compare performance across different libraries, we will also implement models from `xgboost`, `catboost`, and `scikit-learn`. This will allow us to analyze potential improvements in predictive performance and computational efficiency when switching between frameworks.

For training these models, we will use the following set of features:

- Temporal columns: `date_id` and `time_id`.
- All feature columns (`feature_00` to `feature_78`).

Each model will be trained separately for each `symbol_id`, as this variable represents a financial asset, stock, or order. To maintain data integrity and ensure meaningful results, we will not mix data across different `symbol_id`s. Instead, we will report performance metrics independently for each model and each asset.

Regarding the `responder` variables, we have chosen to exclude them from the training process, as they are not supposed to be known at the time of prediction. However, the `weight` column will be incorporated later to compute the final evaluation score.

3.3 Model Evaluation

The performance of our models is assessed using the sample-weighted zero-mean R-squared score (R^2) for `responder_6`. This evaluation metric ensures that predictions are not only accurate but also properly weighted according to the dataset's structure.

The R^2 score is defined as follows:

$$R^2 = 1 - \frac{\sum w_i (y_i - \hat{y}_i)^2}{\sum w_i y_i^2} \tag{1}$$

where:

- y_i represents the ground-truth values of `responder_6`.
- $\hat{y}_i$ represents the predicted values.
- w_i denotes the sample weights.

This metric effectively measures how well the predictions fit the true values while accounting for the relative importance of each sample. A score closer to 1 indicates a better predictive performance, while lower or negative values suggest poor model fit.

4 Results

This section presents the results of our predictive models, comparing their performance in terms of accuracy, feature importance, and computational efficiency. We analyze models implemented using PySpark and those trained with other machine learning libraries such as XGBoost, CatBoost, and Scikit-Learn.

4.1 Results with PySpark Models

To handle the large dataset efficiently, we first trained our models using PySpark, which allows distributed computation. The two models tested were:

- **Gradient Boosted Trees (GBTRegressor)**
- **Random Forest Regressor**

4.1.1 Gradient Boosted Trees (GBTRegressor)

The Gradient Boosted Trees (GBT) model was trained for each `symbol_id` using all available numerical features (`feature_00` to `feature_78`).

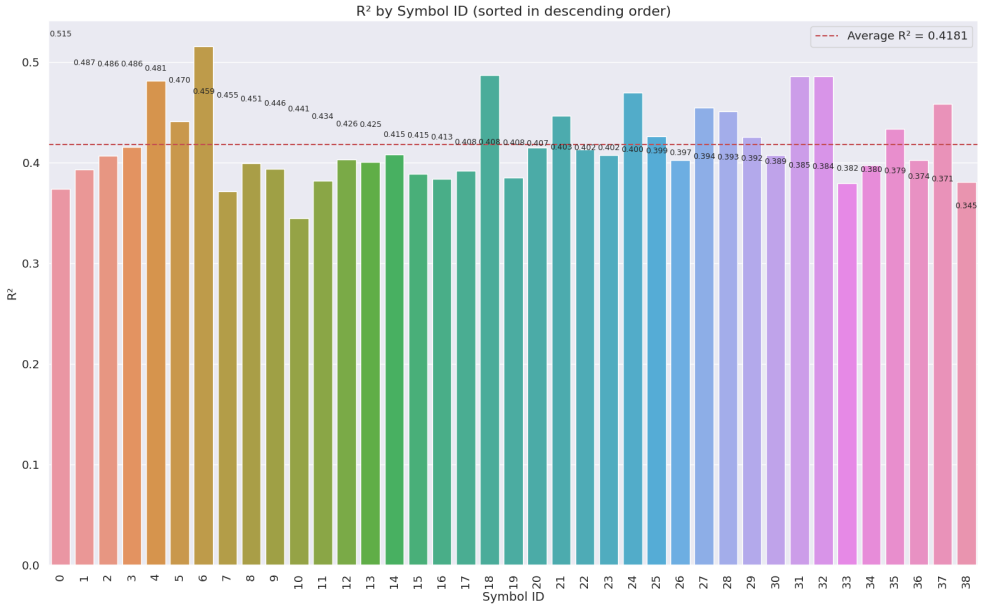

Figure 3: R^2 score results per `symbol_id` for the GBT model.

Figure 3 shows the distribution of R^2 scores across different financial assets.

The average R^2 score obtained with GBT was 0.41, indicating a moderate predictive power. However, we observed significant variability across different assets, with some symbols reaching R^2 values above 0.5, while others remained close to 0.3

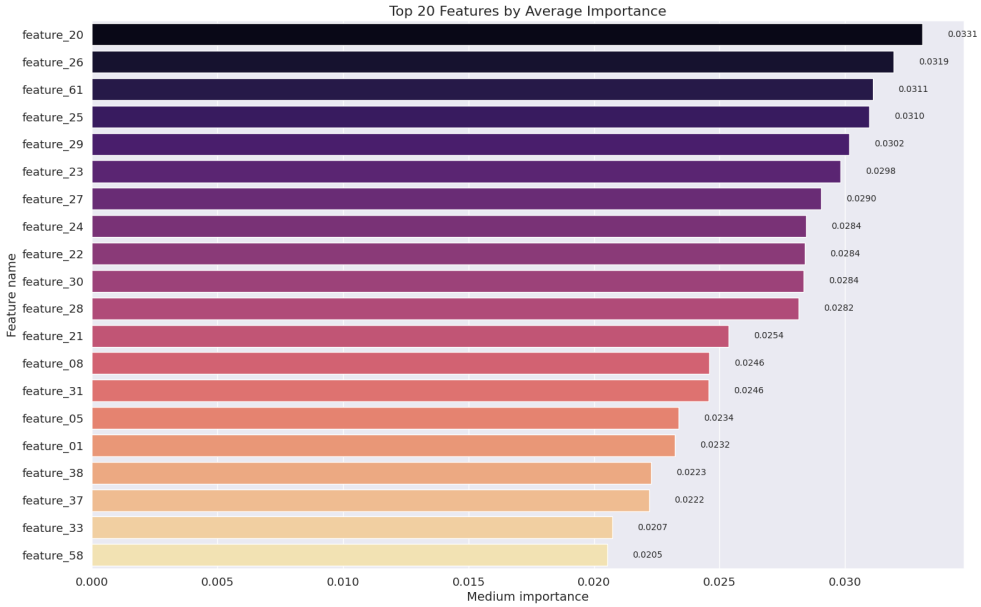

Figure 4: Influences of features on the GBT model

Figure 4 presents the top 20 most important features used by the model.

The total execution time for GBT was 289.34 minutes, making it computationally expensive compared to other models.

4.1.2 Random Forest Regressor

The Random Forest model showed less accurate results than GBT. As shown in Figure 5, the average R^2 score was 0.21, this is well below the results of the GBT model. This regression suggests that the ensemble of decision trees captures the complex relationships of this dataset less well.

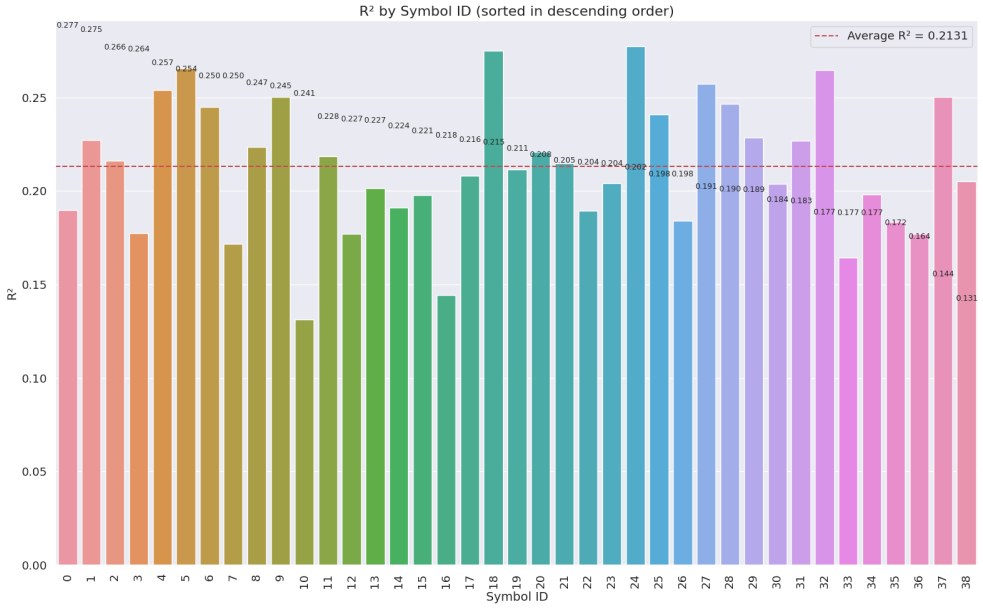

Figure 5: R^2 score results per `symbol_id` for the RandomForest model (by PySpark).

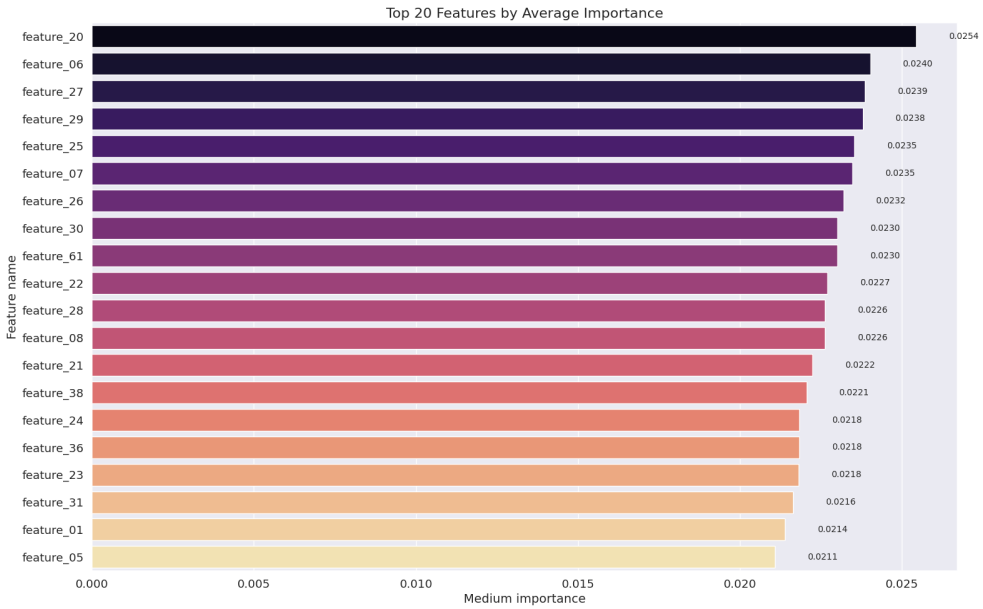

Figure 6: Influences of features on the RandomForest model (by PySpark)

Feature importance analysis (Figure 6) indicated that certain numerical features con-

tributed significantly to the predictions, reinforcing the idea that some financial variables are more predictive than others (especially for `features_20` who was already the most influential `features` for the GBT model.

Despite requiring substantial computation, the training time for Random Forest was lower than GBT, at 196.98 minutes.

4.2 Results with Non-PySpark Libraries

To compare performance, we also trained models using more traditional machine learning frameworks, namely XGBoost, CatBoost, and Scikit-Learn's Random Forest.

4.2.1 CatBoost

The CatBoost model was particularly efficient in training time, completing execution in just a few minutes. However, its predictive performance was a way lower than that of PySpark's GBT, with an average R^2 score of 0.15 as show in Figure 7.

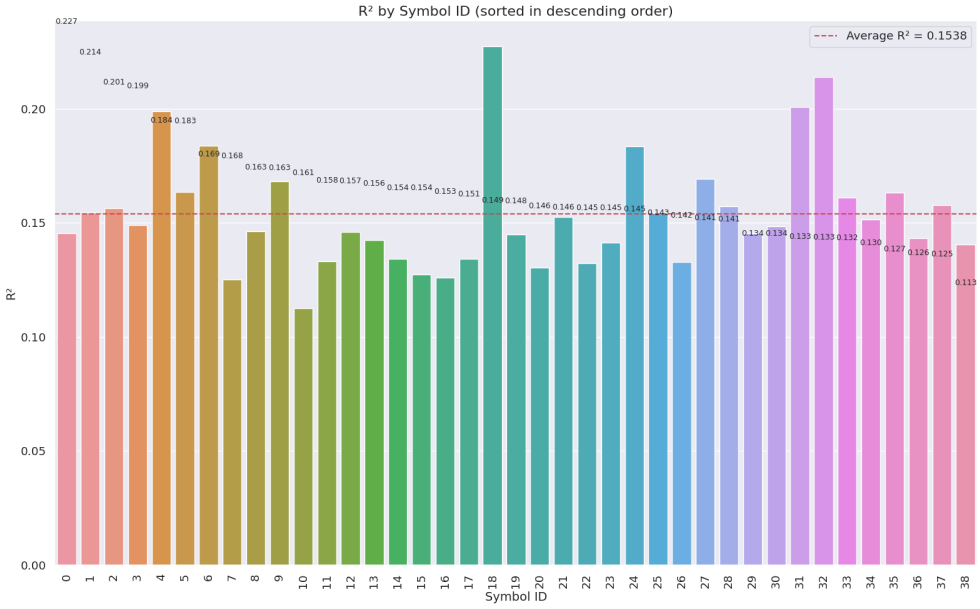

Figure 7: R^2 score results per `symbol_id` for the CatBoost model.

4.2.2 XGBoost

XGBoost, another widely used boosting algorithm, provided a beter results to CatBoost, achieving an average R^2 score of 0.30 (Figure 8) but still lower to GBT. The training time remained reasonable, taking only a few minutes.

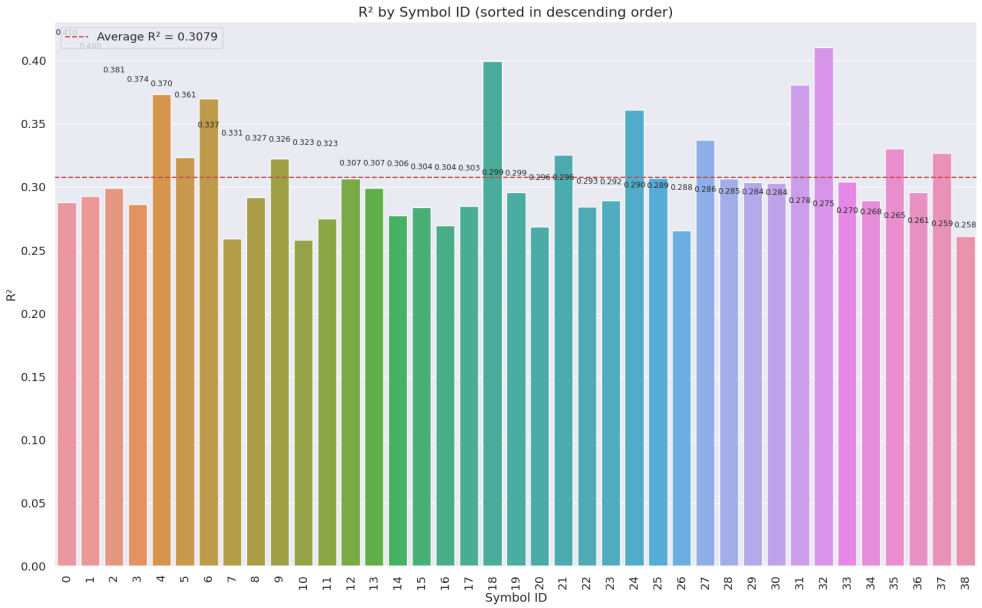

Figure 8: R^2 score results per `symbol_id` for the XGBoost model.

4.2.3 Random Forest (Scikit-Learn)

The Random Forest model implemented with Scikit-Learn exhibited strong performance, obtaining an average R^2 score of 0.61 (Figure 9), slightly higher than the PySpark version. However, it required a much longer training time, approximately 6 hours, making it less practical for large-scale datasets.

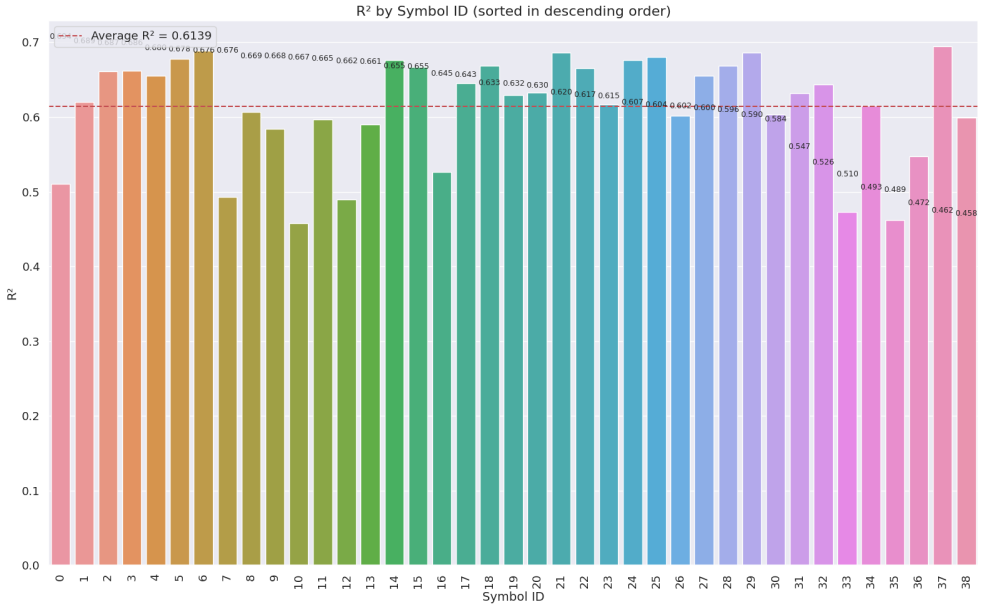

Figure 9: R^2 score results per `symbol_id` for the RandomForest model (By Scikit-Learn).

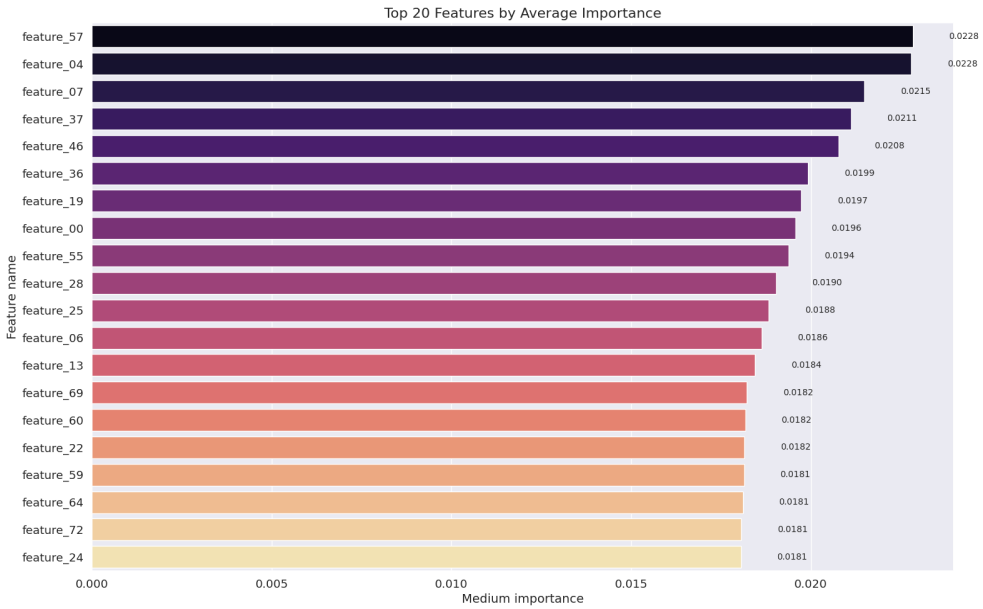

Figure 10: Influences of features on the RandomForest model (by Scikit-Learn)

Feature importance analysis (Figure 10) showed a completely different ranking of feature

influences compared to the other models (espicially for the other RandomForest model from PySpark) which is quite interesting.

4.3 Comparative Analysis and Conclusion

The key findings from our experiments are:

- The **Random Forest model consistently outperformed other models**, achieving the highest R^2 scores in Scikit-Learn implementations but not in the PySpark one.
- Gradient Boosted Trees (GBT) showed promising results.
- XGBoost and CatBoost demonstrated poor predictive power, but fast execution.
- PySpark models are useful for handling large datasets, but non-distributed models such as XGBoost and CatBoost proved to be more efficient for moderate dataset sizes.

Overall, the best trade-off between accuracy and computational efficiency was obtained with the **Random Forest model**, particularly the Scikit-Learn implementation. Despite longer training times, it provided the most reliable and stable predictions across different financial assets.

5 Discussion

The results obtained from our predictive models provide valuable insights into the feasibility of forecasting financial asset movements based on historical data. The comparative analysis of different machine learning models, including Gradient Boosted Trees, Random Forest, XGBoost, and CatBoost, highlights the strengths and limitations of each approach.

One key observation is that the choice of features significantly impacts model performance. Our approach of training separate models for each `symbol_id` ensured that predictions remained asset-specific, avoiding potential biases introduced by mixed financial instruments. However, this also led to computational challenges, as training individual models increased the overall processing time. Optimizing the dataset by selecting a representative subset of time points per day helped mitigate this issue without significantly reducing predictive power.

Another important finding concerns the use of sample weighting in evaluation. The weighted R^2 metric played a crucial role in assessing model quality, as it accounted for variations in data distribution. This reinforces the need for careful consideration of weighting schemes when dealing with imbalanced datasets.

5.1 Ethical Considerations and Challenges

While AI-driven financial predictions can enhance decision-making, they also raise significant ethical concerns. One major issue is the potential for model-driven trading strategies to amplify market volatility. If machine learning models misinterpret patterns or react to noise rather than genuine market trends, they could contribute to increased instability, leading to unintended financial consequences.

Another ethical concern relates to data bias and fairness. If the training data contains inherent biases—such as being overly representative of certain financial conditions or assets—the models may generate predictions that favor particular outcomes, reinforcing existing market inefficiencies. Ensuring diversity and representativeness in training data is crucial to prevent such biases.

Additionally, AI models can create risks in automated decision-making. If financial institutions rely heavily on AI-based predictions without human oversight, unexpected model failures could result in substantial financial losses. Incorporating explainability techniques and robust validation processes is essential to ensure that AI-driven decisions remain transparent and accountable.

Finally, ethical considerations extend to the responsible use of AI in trading. The potential for algorithmic trading strategies to exploit market inefficiencies raises questions about fairness and regulatory oversight. It is important to ensure that AI is used in a manner that promotes market stability and benefits all stakeholders rather than disproportionately favoring those with access to advanced computational resources.

These challenges emphasize the need for ongoing monitoring, ethical AI governance, and the implementation of safeguards to prevent unintended negative consequences in AI-driven financial applications.

6 Conclusions

This study aimed to develop a robust machine learning model for financial market prediction using real-world anonymized trading data. By comparing different modeling approaches, we identified key trade-offs between predictive performance and computational efficiency.

Our findings demonstrate that the Random Forest model consistently outperformed other approaches, particularly in the Scikit-Learn implementation, achieving the highest R^2 scores across different `symbol_id`. Gradient Boosted Trees (GBT) also provided promising results, but at a significantly higher computational cost. While PySpark was useful for processing large-scale data, traditional machine learning frameworks such as XGBoost and CatBoost proved to be more efficient for moderate dataset sizes.

Additionally, we found that feature importance analysis played a crucial role in understanding market dynamics. Temporal features such as `time_id` and selected market indicators had a significant impact on predictions, reinforcing the need for careful feature selection in financial modeling.

Beyond technical considerations, this study also highlighted important ethical concerns associated with AI-driven financial predictions. The potential for models to amplify market volatility, introduce biases, and contribute to automated decision-making errors underscores the importance of regulatory oversight and responsible AI deployment.

Future research directions include:

- Exploring deep learning models such as LSTMs and Transformers to capture long-term dependencies in market data.
- Investigating alternative feature engineering techniques to enhance model interpretability.
- Assessing the impact of different weighting schemes to further refine the evaluation metric.
- Implementing real-time learning strategies to adapt to changing market conditions dynamically.

In conclusion, this study provides valuable insights into financial market prediction, demonstrating the potential and limitations of various machine learning techniques. The

results emphasize the importance of selecting the right model for the right context, balancing performance, efficiency, and ethical considerations in AI-driven trading.

7 Reference

1. Abdulhamit Subasi, Faria Amir, Kholoud Bagedo, Asmaa Shams, Akila Sarirete *Stock Market Prediction Using Machine Learning.* Available `https://www.sciencedirect.com/science/article/pii/S1877050921021128`.

2. Michel Ballings, Dirk Van den Poel, Nathalie Hespeels, Ruben Gryp *Evaluating multiple classifiers for stock price direction prediction.* Available `https://www.sciencedirect.com/science/article/pii/S0957417415003334?via%3Dihub`.