Model submission for Jane Street Real-Time Market Data Forecasting Competition

Arya Gadage, Ashna Jain

<u>Introduction</u>

The integration of machine learning in financial markets has opened new avenues for predictive analytics, enabling more informed decision-making. They have revolutionized how markets are studied, risks are assessed, and decisions are made. Their ability to process vast amounts of data and identify patterns has made it a cornerstone of modern finance. However, financial datasets are often complex and high-dimensional posing significant challenges for effective modeling.

In this competition, hosted by Jane Street, we were tasked with building a model to predict financial market responders using real-world data derived from production systems. This competition is uniquely challenging, highlighting the complexities of modeling financial markets, including the difficulties of handling fat-tailed distributions, non-stationary time series, and abrupt shifts in market behavior. The opportunity to work with anonymized market data and attempt to predict future movements felt like a natural extension of what we had studied.

These challenges reflect the daily obstacles faced in trading, making this an ideal project to apply techniques from our coursework, particularly in game theory and probability.

Jane Street's challenge offers a close look at the nuances of financial market modeling. It emphasizes the inherent "unpredictability" and "volatility" in financial markets. By building a robust model, participants can gain insights into financial data analysis and the methodologies employed in risk management and market prediction.

We decided to build a basic machine learning model using neural networks and gradient boosted trees. This report outlines the methodology employed in this project, combining dimensionality reduction through **autoencoders** and predictive modeling using **XGBoost gradient-boosted**

trees. Below, we describe our research process, methodology, findings and challenges we came across during this project.

Dataset

The proprietary data provided in this competition is anonymized and lightly obfuscated with some of the features and responders namely, 79 features and 9 responders but structured to represent real-world trading challenges accurately. The dataset is derived from Jane Street's production systems and is a collection of features and responders related to markets where Jane Street's automated trading strategies are implemented. The goal of the competition is to forecast one of these responders, i.e., responder_6, for up to six months in the future.

The dataset provided for the competition reflects the intricacies of real-world financial data. The training set contained historical data and returns. For convenience, the training set has been partitioned into ten parts. The features {00,.....78} represented anonymized data while the responders {00,....,08} were clipped between -5 and 5. Each row in the train.parquet dataset corresponds to a unique combination of a symbol (identified by symbol_id) and a timestamp (represented by date_id and time_id). The date_id column is an integer which represents the day of the event, while time_id represents a time ordering. An important thing to note is that the real time differences between each time_id are not guaranteed to be consistent. Also, the symbol_id column contains encrypted identifiers. Each symbol_id is not guaranteed to appear in all time_id and date_id combinations. Additionally, new symbol_id values may appear in future test sets.

Dataset

Handling the data properly was essential to build an effective model. First, given the sheer size of the dataset, we sampled subsets of the data, with 5,000 rows per file to create a balanced and computationally feasible subset. This helped us quickly iterate through models without overwhelming our resources, ensuring a manageable dataset size without losing key patterns. We employed a local server to run our code because of the size of data. We split the dataset into training and validation subsets to ensure the model's robustness with 80% of the data being trained and the model was tested on 20% of the data. This reflects real-world scenarios where models must predict on unseen future data. Missing values are common in financial datasets due to system outages, incomplete feeds, or data discrepancies. Financial time-series data requires continuity for technical indicators and predictions. We replaced missing values with zeros to ensure our models have consistent inputs.

Dimensionality and further data analysis

The dataset is highly dimensional. Having such datasets causes issues since it necessitates training a large number of parameters, which may easily lead to overfitting and poor generalization. High dimensionality also translates to lengthy training periods. To solve these challenges, dimensionality reduction methods are often utilized. (GeeksforGeeks, 2022).

To reduce the dimensionality of the set, we initially considered principal component analysis (PCA). PCA is a linear dimensionality reduction technique which forms principal components which are eigenvectors of the data's covariance matrix. PCA is commonly used when the variables are highly correlated with each other, making it logical to reduce their number to an independent set.

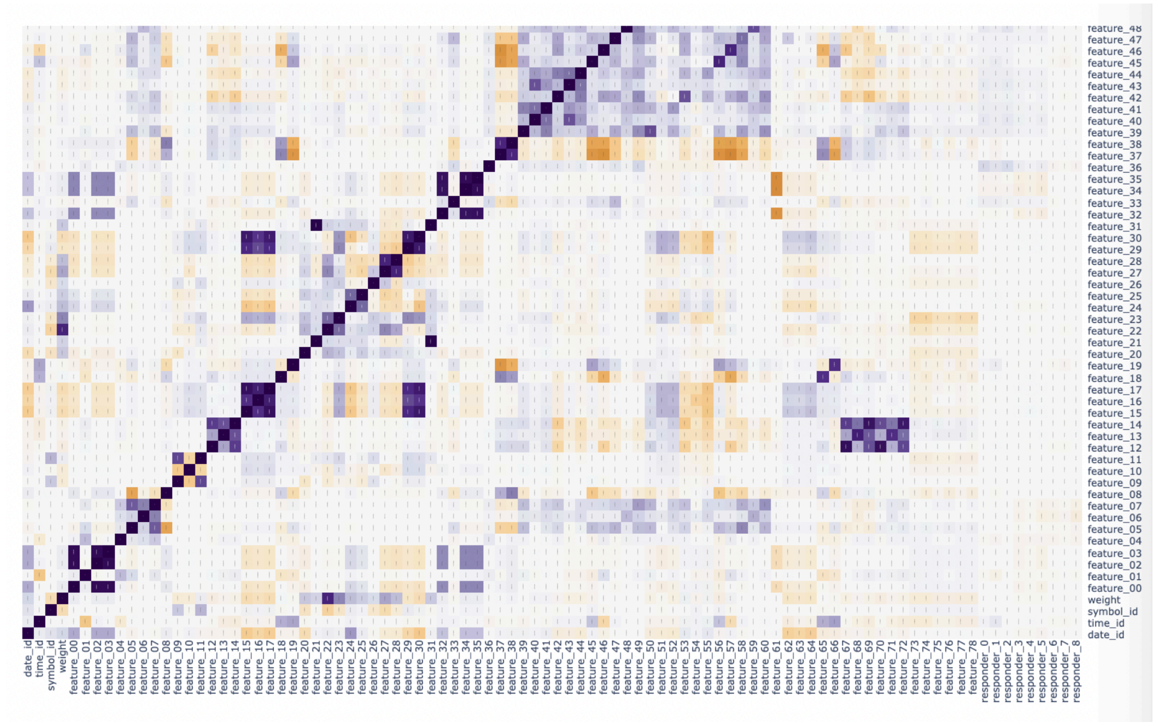

Figure 2 - Heat correlation map of the features(00,....78) and responders(00,....,08)

The major problem we would run into with PCA would be the anonymization of the dataset. Additionally, financial data tends to be non linear. In order to understand the correlation between the features and responders, we generated a heatmap to identify key features. Highly correlated features suggest redundancy, which can inflate model complexity without improving predictive power. The purple indicates very high correlations while the orange and yellow shades indicate low correlation and covariance. The figure below shows clusters of highly correlated features, uncorrelated variables, and even potential redundancies - for instance, features 15, 16, 17 are highly correlated. These features could be price- based features like high yield 2 year, 5 year and 10 year rates or open, close and adjusted prices of a specific stock/ financial instrument. They could also signify volume based relationships like trading volume and volatility measures or indicate time based patterns. Based on these analyses, we decided to input an autoencoder to reduce dimensionality.

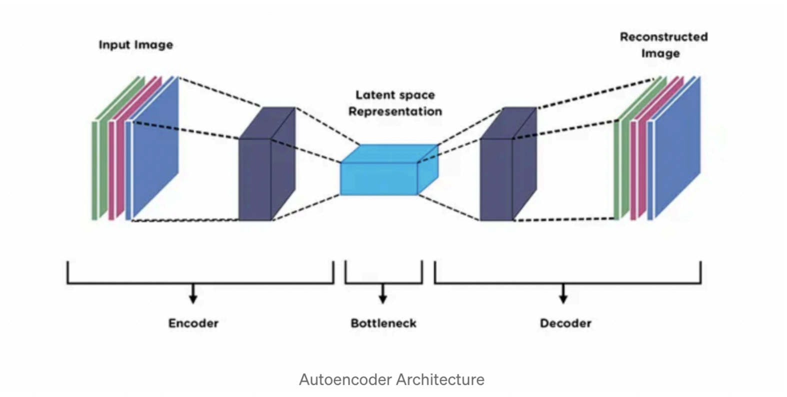

Figure 2 - Figure retrieved from Birla, D. (2019), Basics of Autoencoders, Medium

Autoencoders (AE) are a type of artificial neural network that aims to copy their inputs to their outputs. They work by compressing the input into a latent-space representation also known as a bottleneck, and then reconstructing the output from this representation. Autoencoders consist of an encoder and a decoder. The encoder reduced the 79 features into a 32-dimensional latent representation, effectively identifying the most critical patterns while discarding redundant information.The decoder reconstructed the original features from the latent space, ensuring that the compressed representation retained meaningful data. The model minimized Mean Squared Error (MSE) during training, with the Adam optimizer ensuring efficient convergence.

Gradient boosted Trees

XGBoost is a gradient-boosted decision tree algorithm, widely used in machine learning for its efficiency and accuracy, especially with tabular data. Gradient boosted trees use a method called boosting. Boosting combines weak learners (usually decision trees with only one split, called decision stumps) sequentially, so that each new tree corrects the errors of the previous one. The main idea is to combine many simple models, also known as “weak learners,” to create an ensemble model that is better at prediction.

The model was set up with 2000 estimators, or trees, to iteratively improve the predictions. To optimize training, we implemented a custom learning rate scheduler. The learning rate started at

0.3 and gradually decayed with each epoch, ensuring faster initial training and more precise adjustments in later stages. The model was trained to minimize the Root Mean Square Error, or RMSE, which measures the average error magnitude in the predictions. Additionally, we used early stopping to monitor the model's performance on the validation set. If the validation loss didn't improve for 10 consecutive rounds, training would stop to avoid overfitting.

Results

The project's key findings highlight the potential of combining dimensionality reduction and gradient boosting for financial forecasting. The model achieved a Root Mean Squared Error (RMSE) of ~0.813, indicating moderate prediction accuracy. However, the R^2 score was around -0.008, which suggests that the model struggled to explain the variance in the target variable effectively. Financial modeling is an iterative process, and further refinements are essential to enhance performance. This study highlights the importance of combining technical rigor with domain expertise to advance risk management and decision-making in financial markets.

Discussion

With the preliminary model implemented, there are several ways in which we can improve the predictive power. First, we could experiment with the architecture of the autoencoder. For instance, using a Variational Autoencoder might help capture more complex patterns in the data. Adjusting the size of the latent space—compressing the features into fewer or more dimensions—could also impact performance.

Second, we could explore alternative machine learning models, such as Random Forests or LightGBM, to see how they perform compared to XGBoost. Combining multiple models in an ensemble could also leverage the strengths of different approaches.

Feature engineering is another critical area for improvement. By incorporating domain-specific knowledge, we could derive new features, such as volatility measures or sentiment analysis indicators, to provide the model with richer input data.

Markets evolve constantly, and risk managers must adapt models to account for new patterns, correlations, or shocks. To turn this project into a successful risk management model, we need to work on further extracting the most relevant information from the complex dataset and building predictive tools that generalize well across different market scenarios.

References

1. Desai, M., Zhang, Y., Holbrook, R., O'Neil, K., & Demkin, M. (2024). *Jane Street Real-Time Market Data Forecasting.* Kaggle. Retrieved from https://kaggle.com/competitions/jane-street-real-time-market-data-forecasting
2. GeeksforGeeks. (2022, February 22). *How is autoencoder different from PCA?* Retrieved from https://www.geeksforgeeks.org/how-is-autoencoder-different-from-pca/
3. Birla, D. (2019, March 12). *Basics of Autoencoders.* Medium. Retrieved from https://medium.com/@birla.deepak26/autoencoders-76bb49ae6a8f
4. Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

5. Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science, 313*(5786), 504–507.
6. OpenAI. (2024). *ChatGPT.* Retrieved from https://chat.openai.com/
7. GitHub. (n.d.). *XGBoost Documentation and Examples.* GitHub. Retrieved December 9, 2024, from https://github.com/dmlc/xgboost

Acknowledgements

This research received significant support during the Math 490H course, instructed by Professor Patrick Flaherty, Department of Mathematics and Statistics at the College of Natural Sciences.