[redacted]

Introduction

This research evaluates the Radial Basis Function (RBF) Neural Network against other machine learning algorithms, aiming to predict mid-price movements in Limit Order Book (LOB) data, a common regression task. The impact of principal components on both regression and classification tasks was also studied. Challenges arose due to LOB data characteristics. To improve performance, nine experimentally derived features were added, along with 11 features from previous Financial Machine Learning II coursework, enhancing the model's feature set.

Data Pre-processing

We scrutinised the mid-price target variable in our regression task, discovering 2,932 rows with zero values. This was due to either absent AskPrice1, BidPrice1, or both, accounting for 0.52% of all rows. This could skew the regression, possibly owing to periods of sparse trading activity. To ensure accurate data, these rows were deleted. The purified data was then divided into training (60%), validation (20%), and testing (20%) sets. We also generated a target variable for the classification task.

Feature Engineering

To boost the model's performance, the number of relevant features was increased, and irrelevant ones reduced. Ntakaris et al. (2019) reported superior results from manual feature crafting over automated feature extraction using LSTM Autoencoders. Thus, an additional 20 features were added to the initial 40-feature set, primarily representing the LOB state, as follows:

Table 1 Feature engineering

No	Features	Description
1	Financial Duration	This feature, indicating the time gap between successive data points, serves as a proxy for market activity level, as suggested by Ntakaris et al. (2019a). Incorporating this feature allows for tracking fluctuations in trading frequency, thus capturing dynamic liquidity changes and potential shifts in the behaviors of market participants.

2	MidPrice_OIB	The weighted mid-price, termed MidPrice_OIB, is an alternate to the standard mid-price, as per Ntakaris et al. (2019a). These metric weighs bid and ask prices by their volumes, highlighting price levels with concentrated trading interest, offering insight into the asset's perceived value by market participants.
3	VolImbalance	Volume imbalance, the ratio of bid volume to total volume at optimal bid and ask prices (Ntakaris et al., 2019a), reflects immediate supply-demand disparity, potentially influencing short-term price shifts.
4	BA_spread	The bid-ask spread measures liquidity and trading risk by calculating the difference between the best bid and ask prices (Ntakaris et al., 2019a).
5	AccumulatedVolume Difference	This feature quantifies the supply and demand imbalance in the Limit Order Book (LOB) by computing the disparity between the accumulated volumes of bids and asks at multiple levels (Cont and Kukanov, 2014). It indicates the market's inclination towards buying or selling the asset in the near term.
6	WeightedAverageBid Price	The weighted average bid price reflects the aggregated buying prices across multiple bid levels (Cartea and Jaimungal, 2015).
7	WeightedAverageAs kPrice	The weighted average asks price reflects the aggregated selling prices (Cartea and Jaimungal, 2015).
8	OrderFlowImbalance	This feature captures short-term buying or selling pressure by comparing bid and ask volumes across multiple levels, providing insights into market momentum (Biais, Hillion, and Spatt, 1995).
9	RelativeBidVolume	This feature captures the relative liquidity at the best bid price by comparing the current best bid volume ratio to the average bid volume across multiple levels (Ranaldo, 2004).

10	RelativeAskVolume	The relative ask volume captures the liquidity at the best ask price by comparing the current best ask volume ratio to the average ask volume across multiple levels (Ranaldo, 2004)..
11	PriceMomentum	The price momentum feature compares the current mid-price with a historical mid-price, aiding in trend identification and potential market pattern recognition (Jegadeesh and Titman, 1993).
12	BA_spread2	Bid ask spread at level two.
13	BA_spread3	Bid ask spread at level three.
14	BA_spread4	Bid ask spread at level four.
15	BA_spread5	Bid ask spread at level five.
16	BA_spread6	Bid ask spread at level six.
17	BA_spread7	Bid ask spread at level seven.
18	BA_spread8	Bid ask spread at level eight.
19	BA_spread9	Bid ask spread at level nine.
20	BA_spread10	Bid ask spread at level ten.

In their earlier work, "Financial Machine Learning II", the author had studied 11 out of the 20 engineered features listed in Table 1. The new set of features, namely BA_spread2 to BA_spread10, were introduced to explore the potential influence of different bid-ask levels on the performance of a machine learning model. These bid-ask spreads provide insights into market liquidity, a key econometric indicator that could enhance the model's predictive capacity (Harris, 2003).

Table 2 RMSE of 10-fold time series cross-validation using XGBoost

Number of features	Mean	Median
40 features	4,913	1,283
51 features	3,567	1,914
60 features (bid bid-ask spread level 2 to 10 included)	3,965	1,694

The effect of these additional features was evaluated using a 10-fold cross-validation. Interestingly, the results in Table 2 exhibited a decline in mean performance while the median performance improved. This divergence might suggest that the model's performance varied quite substantially across different folds. Given that the mean is susceptible to outliers, the decrease might point to poor performance on a few folds, possibly due to overfitting or noise associated with the bid-ask spread features (Hastie et al., 2009).

Conversely, the rise in median performance indicates that the addition of bid-ask spread features improved the model's performance on at least half of the folds. This suggests that the new features provided valuable information, enhancing the model's predictive capacity in a majority of scenarios.

However, caution must be exercised while interpreting these results. Bid-ask spreads, while providing deeper market insights, can also introduce volatility. Such complex nonlinear interactions may not always be effectively captured by the model across all folds, leading to the observed discrepancies. It highlights the need for refining these features and possibly incorporating further pre-processing steps to optimize their utility.

Feature Selection

In addition to feature engineering, the author also conducted feature selection method using 0.1 variance threshold and PCA. This technique is designed to pinpoint and discard features from the dataset that show a variance of 0.1 or less, signifying low variance. The goal of setting such a threshold is to remove features with insufficient variation, as they are unlikely to contribute valuable information to further analysis or modelling efforts. As a result, this process aids in trimming the dataset's dimensionality, potentially enhancing the performance and efficiency of subsequent machine learning algorithms. In this specific analysis, the author discovered that two features, Vol imbalance and Price momentum, fell under this low-variance category.

[redacted]

The author also utilised Principal Component Analysis (PCA), that transforms original variables into a new set of uncorrelated variables called principal components. These components are chosen such that they capture a certain percentage of the dataset's variance. In this case, the author sought components that explained 99% of the variance and found that 25 principal components served this purpose.

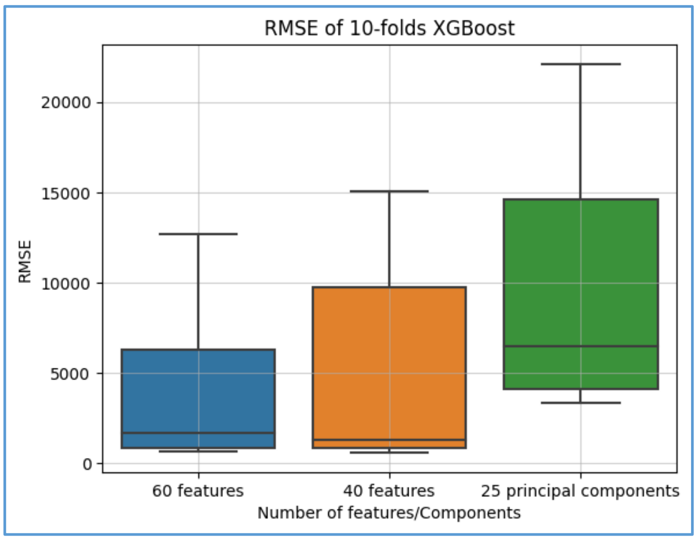

Figure 1 RMSE comparison between 60 features, 40 features and 25 principal components.

However, when these 25 components were used in a regression task, the Root Mean Square Error (RMSE) - a measure of the differences between values predicted by a model and the values observed – dropped as shown in Figure 1. This indicates that the model's predictive accuracy improved, as a lower RMSE value typically means a better fit to the data.

However, in the first run as shown in Table 3 below, the SVM classifier with 60 features achieved an accuracy of 0.74738, while the classifier with 25 principal components had an accuracy of 0.54029. This suggests that the reduced dimensionality might have led to a loss of important information, causing the lower performance with 25 principal components.

Table 3 SVM classifier with original features vs Principal Component

Fold	SVM Classifier with original 60 features	SVM Classifier with 25 principal components
1	0.74738	0.54029
2	0.89678	0.87742
3	0.93082	0.93082
4	0.93449	0.93449
5	0.89035	0.89035
6	0.90955	0.90955
7	0.88929	0.88929
8	0.88689	0.88689
9	0.90525	0.90525
10	0.91517	0.91517

However, in the subsequent runs, the accuracies for both classifiers are virtually identical, indicating that the reduced-dimension model has become as effective as the original one.
As more data becomes available for training the model, the classifier can better understand the underlying patterns in the data, leading to an improvement in accuracy. This improvement can occur for several reasons:

1. Better Representation of the Problem Space: More data can provide a more comprehensive representation of the problem space, including a wider variety of scenarios and edge cases. This greater coverage can improve the model's ability to handle different situations.
2. Improved Learning from Features: With more data, each feature has more examples from which the model can learn. In the case of SVM, which tries to find the optimal hyperplane separating different classes, more data can help identify a better hyperplane that generalizes well to unseen data.

Thus, for the classification problem, the dimensionality reduction did not adversely impact the model's performance, and even provided efficiency gains in terms of computational resources.

RBF NN

In the ongoing Limit Order Book (LOB) data analysis, we combined a Radial Basis Function (RBF) Neural Network with KMeans clustering. This strategy hinges on using the means and standard deviations (stds) extracted from the KMeans clusters as input for the RBF Neural Network. However, the inherent randomness in the initialization of KMeans centroids can sometimes lead to clusters that don't necessarily provide the optimal data grouping.

An additional challenge emerges from the data features that predominantly display zero values, resulting in zero values in their means and variances. This lack of variation across observations makes these features less beneficial for the RBF Neural Network model. It inhibits the effective calculation of the Gaussian basis function, a core part of the RBF model's operations.

In response, the author has adopted a feature selection process that excludes the cluster centres and stds with zero values. This step effectively ensures that only the features with the most variability, and hence informational value, are incorporated into the RBF NN. While this method effectively tackles the issue of zero values, it's based on the assumption that the excluded features have no significant consequence for the RBF Neural Network's performance. Therefore, future research should explore more advanced feature selection or imputation methods that can address such challenges more comprehensively.

Model Selection, Model Rejection and Result

An RBF Neural Network uses Gaussian activation functions for its hidden layer neurons. The Gaussian function for a given neuron i is defined as:

$$\phi_i(x) = \exp(-\|x - c_i\|^2 / (2\sigma_i^2))$$

Where x is the input vector, c_i is the centre (mean) of the Gaussian for neuron i, and σ_i is the Gaussian's standard deviation (spread). The output of the RBF NN is a weighted sum of these Gaussian functions.

When we train the RBF NN with a small amount of data, the Gaussians' means (c_i) and standard deviations (σ_i) are determined based on this limited data. If the data subset does not represent the whole dataset, the Gaussian functions may not accurately capture the full data distribution. This can lead to overfitting, where the model performs well on the training data but poorly on unseen data.

When we train the RBF NN with more data, the Gaussians have more information to learn from. The means (c_i) become better estimates of the true means of the data, and the standard deviations (σ_i) become better estimates of the true spread of the data. This allows the Gaussian functions better to capture the overall structure and variations in the data.

The author made use of Radial Basis Function Neural Network (RBF NN), applying iterative loops to each feature individually and subsequently averaging the resultant output over the 60 features. However, this methodology led to a substantial Root Mean Square Error (RMSE) of 969,395 when employing online training, as evidenced in Table 4. This RMSE significantly exceeds the values observed with other regression models as shown in the table. Two potential justifications can be identified for this elevated RMSE:

1. The unique multidimensional relationships between features could have been lost due to the individual training of each feature.
2. Noise within the Limit Order Book (LOB) data may have led the RBF NN to erroneously learn incorrect patterns.

[redacted]

Table 4 RMSE of different models from Financial Machine Learning II, author's previous work.

Online iteration	LSTM AE 60 features	LSTM AE 20 latent	MLP	Vanilla LSTM	Random Forests	XGBoost
1	6801.212989	6669.309465	6405.397087	6111.198211	7370.568034	7092.782294
2	6796.652324	6725.706775	6464.807039	5466.380136	6922.046901	5733.497829
3	75220.25376	80807.9289	85305.36442	40798.79827	89262.53276	88679.67588
4	31062.86466	6362.48498	36112.70157	27032.98565	31297.25676	4920.329302
5	6116.52472	5307.332408	6359.069232	5271.140211	6399.077296	6182.607953
6	1453.591891	1708.29498	1389.73472	583.5261255	318.946214	275.3053548
7	4490.239894	5149.406566	2445.770109	2000.321467	4629.383784	2656.590514
8	2116.571344	2019.415155	2135.162202	1983.669234	2452.387788	2612.019615
9	1973.031398	1926.461759	2000.930815	1828.362925	1997.751038	2065.600576
10	8469.755017	12184.12209	6396.830073	12153.31278	13174.06132	9475.655667
Mean	14450.0698	12886.04631	15501.57673	10322.9695	16382.40119	12969.4065
Median	6456.588522	5834.908694	6377.949653	5368.760173	6660.562098	5326.913565

Table 5 RMSE of RBF NN trained with different amount of data.

No	Amount of data used	RMSE
1	10% (55,971 rows)	969,395
2	100% (559,716 rows)	343,976

When the model is trained on only 10% of the data (55,971 rows), the RMSE is quite high (969,395) as shown in Table 5. This suggests the model is not capturing the underlying patterns in the data effectively due to limited training examples, leading to a higher prediction error. This result stipulates that as sample size expands, sample mean, and standard deviation converge towards their respective population values. Therefore, by using more data for training, the parameters of the Gaussian functions in the RBF NN are likely to be more accurately represented, leading to the formation of a superior model.

[redacted]

Conclusion

The RBF Neural Network model's performance was found to be sensitive to the amount of training data and the inclusion of certain features. When trained with a limited subset of the data, the model demonstrated a high RMSE, suggesting inadequate predictive capability. However, as the training dataset size increased, the model's predictive performance improved significantly, demonstrating the importance of large datasets in training complex models.

In terms of feature engineering, the introduction of bid-ask spread features exhibited mixed results. While mean performance decreased, the median performance improved. This pattern hints at the model's varying performance across different folds and points to the need for further exploration and refinement of these features.

Feature selection using a variance threshold and PCA helped reduce data dimensionality. While PCA improved the model's performance in the classification task, its application in the regression task saw a decrease in performance. This reflects the inherent trade-offs when employing PCA: while it aids in dimensionality reduction, it may not always preserve the essential information for every task.

[redacted]

References

Biais, B., Hillion, P. and Spatt, C. (1995) ‘An Empirical Analysis of the Limit Order Book and the Order Flow in the Paris Bourse’.

Cartea, Á. and Jaimungal, S. (2015) ‘RISK METRICS AND FINE TUNING OF HIGH-FREQUENCY TRADING STRATEGIES: risk matrics and hf trading strategies’, *Mathematical Finance*, 25(3), pp. 576–611. Available at: https://doi.org/10.1111/mafi.12023.

Caruana, R. and Niculescu-Mizil, A. (2006) ‘An empirical comparison of supervised learning algorithms’, in *Proceedings of the 23rd international conference on Machine learning - ICML ’06. the 23rd international conference*, Pittsburgh, Pennsylvania: ACM Press, pp. 161–168. Available at: https://doi.org/10.1145/1143844.1143865.

Cont, R. and Kukanov, A. (2014) ‘Optimal order placement in limit order markets’. arXiv. Available at: http://arxiv.org/abs/1210.1625 (Accessed: 12 April 2023).

Harris, L. (2002) ‘TRADING AND EXCHANGES: Market Microstructure for’.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*. New York, NY: Springer New York (Springer Series in Statistics). Available at: https://doi.org/10.1007/978-0-387-84858-7.

Jegadeesh, N. and Titman, S. (1993) ‘Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency’.

Ntakaris, A. *et al.* (2019) ‘Feature Engineering for Mid-Price Prediction With Deep Learning’, *IEEE Access*, 7, pp. 82390–82412. Available at: https://doi.org/10.1109/ACCESS.2019.2924353.

Ranaldo, A. (2004) ‘Order aggressiveness in limit order book markets’, *Journal of Financial Markets*, 7(1), pp. 53–74. Available at: https://doi.org/10.1016/S1386-4181(02)00069-1.

[redacted]

Bibliography

Christopher M Bishop (1995) *Neural Networks for Pattern Recognition*. Clarendon Press.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep learning*. Cambridge, Massachusetts: The MIT Press (Adaptive computation and machine learning).

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*. New York, NY: Springer New York (Springer Series in Statistics). Available at: https://doi.org/10.1007/978-0-387-84858-7.

Harris, L. (2002) 'TRADING AND EXCHANGES: Market Microstructure for'.

Keras (2023) 'Keras official documentation'. Available at: https://keras.io/getting_started/.

Makridakis, S. *et al.* (2022) 'Statistical, machine learning and deep learning forecasting methods: Comparisons and ways forward', *Journal of the Operational Research Society*, pp. 1–20. Available at: https://doi.org/10.1080/01605682.2022.2118629.

Michael Nielsen (2015) *Neural Networks and Deep Learning*. Determination Press. Available at: http://neuralnetworksanddeeplearning.com/.

Ntakaris, A. *et al.* (2018) 'Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods', *Journal of Forecasting*, 37(8), pp. 852–866. Available at: https://doi.org/10.1002/for.2543.

scikit-learn (2023) 'Scikit-learn official documentation'. Available at: https://scikit-learn.org/stable/.

Tsantekidis, A. *et al.* (2020) 'Using Deep Learning for price prediction by exploiting stationary limit order book features', *Applied Soft Computing*, 93, p. 106401. Available at: https://doi.org/10.1016/j.asoc.2020.106401.

XGboost (2023) 'XGBoost official documentation'. Available at: https://xgboost.readthedocs.io/en/stable/.