Explainable Machine Learning for High Frequency Trading Dynamics Discovery

Henry Han[1*], Jeffrey Yi-Lin Forrest[2], Jiacun Wang[3], Shuining Yuan[4], Fei, Han[5], Diane Li[6]

[1]Department of Computer Science, School of Engineering and Computer Science, Baylor University, Waco, TX 76798 USA.

[2]Department of Accounting, Economics, and Finance, Slippery Rock University, Slippery Rock, PA 16057, USA. Email: jeffrey.forrest@sru.edu

[3]Department of Computer Science and Software Engineering, Monmouth University, West Long Branch NJ 07764 USA. Email: jwang@monmouth.edu

[4]Quantitative Finance, The Gabelli School of Business, Fordham University, New York, NY 10023, USA. Email: syuan26@fordham.edu

[5]College of Computer science, Jiangsu University, Zhenjiang, Jiangsu, China 212013. Email: hanfei@ujs.edu.cn

[6] Department of Business, Management and Accounting, University of Maryland, MD 21853, USA. Email: dli@umes.edu

Corresponding author Email: Henry_Han@baylor.edu

Abstract. High-frequency trading (HFT) plays an essential role in the financial market. However, how to discover and reveal trading dynamics remains a challenge in Fintech. In this study, we propose a novel explainable machine learning approach: Feature-Interpolation-based Dimension Reduction SCAN (FIDR-SCAN) to handle the challenge by creating a trading map. The trading map deciphers an HFT security's trading dynamics by marking the status of each transaction, grouping transactions in clusters, and identifying the trading markers. The proposed method presents new feature interpolation techniques to build a more informative and explainable feature space to unveil hidden trading behaviors. It mines HFT data in their low-dimensional embedding to seek exceptional trading markers and classify the statuses of transactions. We validate the meaningfulness and effectiveness of the trading markers discovered from FIDR-SCAN in trading besides examining its special characteristics. Besides, we apply the proposed algorithm to cryptocurrency data and achieve reliable performance. To the best of our knowledge, this study is the first to use interpretable machine learning to reveal HFT trading dynamics.

Keywords: Explainable AI, high-frequency trading, trading dynamics, feature interpolation

1. Introduction

High-frequency trading (HFT) has become a dominant method of trading since the turn of the millennium, relying on computers to execute buy or sell orders without human intervention. In 2022, it was estimated that HFT generated around 50% of all trading volumes in the financial market [3]. By leveraging algorithmic trading with high frequency, HFT offers the opportunity to profit from even the slightest price change in microseconds or nanoseconds, thanks to large trading volumes. HFT can complete transactions in mere nanoseconds through algorithm-driven trading systems that respond almost instantly after an order is placed. To gain a competitive edge, more and more high-frequency trading algorithms are being implemented in hardware with Field-Programmable Gate Arrays (FPGA), GPU, or related technologies, aiming for ultra-low trading latency. Additionally, rebate policies by exchanges have contributed to the popularity of HFT. For instance, the New York Stock Exchange (NYSE) offers incentives to firms that add liquidity to the market [2].

HFT is challenging the financial market and classic finance theory for its ultra-fast speed and huge volume. It generally brings high turnover rates, high order-to-trade ratios, and high Sharpe ratios. Although HFT is believed to increase market liquidity, this liquidity can vanish in a matter of seconds, offering little benefit to most traders. In fact, HFT can provide unfair advantages to large firms due to their ability to build more favorable liquidating positions. HFT adds higher levels of volatility and unpredictability to the financial market and sees the divergence between bid-ask spreads for large-cap and small-cap stocks. HFT also can generate arbitrages that enable winners to beat their competitors with just a few microseconds of lead time [3,4]. To some degree, HFT makes the market more volatile and unpredictable, greatly increasing the risk of flash crashes for its high-speed trading strategies, traders' similar trading algorithms, and possible illegal trading activities.

High-frequency trading (HFT) data can be difficult to manage due to its sheer volume and speed. HFT data primarily refers to the trading data for a specific stock, rather than auxiliary data such as bid-ask quotes or messages associated with order placement. Unlike traditional finance models such as the Capital Asset Pricing Model (CAPM), which assume daily stock data resolution, HFT data provides resolution at the second, microsecond or even finer level. This special characteristic makes it challenging to apply traditional models to HFT data.

High-frequency trading (HFT) data has unique characteristics compared to general financial data. It is a high-speed nonlinear time-series data that contains microstructure noise from different sources, including bid-ask bounce, latency arbitrage caused by hardware and distance in trading, rebates, or other discounts that can distort real prices [2,4]. This noise can make it difficult to understand the true scenarios in trading and can cause the movements in trading to be more nonlinear or even incomprehensible. However, it is currently unknown how to conduct de-noising so that true price signals can be retrieved.

HFT data is typically low-dimensional, meaning that there are more observations than variables. This can make it difficult to accurately represent and interpret complex trading behaviors. The raw HFT data usually only contains three variables: price, volume, and time, despite containing hundreds or thousands of observations. For example, between October 4th and 15th of 2010, the raw HFT data for Johnson & Johnson (JNJ) included 419,565 transactions (observations). In addition to low dimensionality, HFT data also suffers from a feature scarcity issue. The existing features may not be sufficient for downstream analysis tasks such as clustering, adding an additional layer of complexity to analyzing this type of data. As a result, specific considerations are needed when working with HFT data, including overcoming challenges related to low dimensionality and feature scarcity.

When we aggregate HFT data into larger time intervals, such as one minute, we can add a few more variables to the dataset, such as notation value and open price. However, this aggregated data still suffers from a feature scarcity issue because only a few new features are added. For example, the intraday data of JNJ has only 6 variables across 4191 observations. This lack of features can make it difficult to interpret downstream machine learning results and even for traders to understand the effectiveness of these results.

HFT data is unique in its low dimensionality, with a large number of transactions occurring across only a few features. However, this can lead to a significant amount of data redundancy, as HFT prices may barely change over short trading periods [6-7]. This data can be thought of as "energy concentrated data," with the variance mostly concentrated in the first singular value directions or first two principal components. While there have been efforts to reduce this redundancy through dimension reduction algorithms, it is still unclear which ones are most effective. Moreover, different securities in the HFT market show varying trading characteristics. Large-cap stocks like AAPL, for example, not only have higher trading frequencies than smaller, unknown stocks but are also more liquid. Discovering these differences and understanding their implications for downstream analysis remains a challenge in Fintech.

There have been quite a few studies on HFT from various perspectives, including finance, econometrics, AI, and data science. For example, Aquilina et al. quantified the latency arbitrage in HFT using stock exchange message data [3], while Han and Li estimated stock volatility in HFT using big data analytics [5]. Han *et al.* developed a manifold-learning scanning (M-SCAN) method to identify trading markers in HFT [6], and Aït-Sahalia and Xu applied principal component analysis (PCA) to HFT raw data [7]. Menkveld investigated large institution orders in HFT [8], Brogaard *et al.* examined the role of high-frequency traders (HFTs) in price discovery [9], and Baron *et al.* found that faster HFT firms earned significantly greater profits [10]. Conrad *et al.* studied the relationship between quotations and stock behaviors in the HFT market [11], while Manahov and Zhang used genetic programming trading algorithms to simulate the futures market under HFT [12]. Fischer investigated the role of long short-term memory (LSTM) networks in market prediction [13], and Fang used GARCH and support vector machines (SVM) to design HFT trading algorithms [14]. Additionally, Brogaard *et al.* showed that HFTs provided liquidity during extreme price movements [15], and Xu examined the optimal strategies for high-frequency traders to rationalize their pinging activities [16].

However, previous research has largely neglected the investigation of trading dynamics in HFT data. As the distinctive signature of a stock or portfolio in trading, trading dynamics reveal trading patterns and behaviors, identify potential trading signals, and classify transaction statuses. In essence, trading dynamics provide answers to the fundamental question in HFT: "how do different securities behave in trading?" Given the massive data and lightning-fast trading in HFT, discovering meaningful and insightful trading dynamics is crucial for traders to better understand securities and their movements. By uncovering trading dynamics, we can gain a deeper understanding of the latent mechanisms that drive HFT for different securities and their micro-structures. This knowledge can also inform the development of more competitive and adaptive trading algorithms or schemes for HFT traders.

2. Explainable machine learning for trading dynamics discovery

2.1 Trading dynamics discovery

High-frequency trading (HFT) poses significant challenges to discovering trading dynamics. Firstly, there is a dearth of previous research on the subject, leaving uncertainty about how to identify trading dynamics and which key components to include [6]. Secondly, original HFT data consists of an enormous number of transactions occurring over a brief period due to its exceptional resolution, which may present a computing barrier due to its big data nature. This can be prohibitive in discovering trading dynamics without sufficient computing power. Thirdly, the feature scarcity issue and nonlinearity of HFT data make it unclear how to address them, as well as how to conduct de-noising and distinguish between the trading characteristics of different securities.

To better capture trading dynamics in this study, we used aggregated intraday data with a 1-minute resolution, rather than the raw high-resolution data. This approach involved sampling data within each 1-minute interval to obtain high, low, open, and close prices, as well as volume. By using this method, we

avoided the computing burden associated with processing large amounts of raw data, while still capturing important global trading behaviors that may have been obscured at the raw data level.

High-Frequency Trading (HFT) data is inherently complex, and extracting valuable trading dynamics information from it requires the use of machine learning (ML) techniques. Tasks such as information intelligence seeking, de-noising, redundancy filtering, and distinguishing between different securities all benefit from ML approaches. However, determining which techniques are most effective for analyzing HFT data remains an active research question.

2.2 Low-dimensional embedding clustering

One promising approach is to use dimension reduction techniques to project the high-dimensional HFT data onto a lower-dimensional space. This approach has the potential to reveal intrinsic structures, reduce noise and redundancy, and ultimately expose valuable trading dynamics information. By clustering the resulting low-dimensional embedding, we can extract meaningful patterns and insights from the data.

To achieve this, we can use a locally isometric mapping function f_d to transform an HFT dataset with n transactions and p variables $X = \{x_i\}_{i=1}^n,\ x_i \in \Re^p$, into a low-dimensional embedding: $E = \{e_i\}_{i=1}^n,\ e_i \in \Re^k, k < p,\ s.t.\ f_d: x_i \to e_i$. The mapping function transforms each transaction x_i into its corresponding e_i such that the distance metric, which may not be the Euclidean distance, between pairs of points is approximately preserved, i.e. $\|e_i - e_j\| \approx \|x_i - x_j\|$. Once we have the low-dimensional embeddings, we then apply a clustering algorithm Γ_c to group them into different clusters $C_1, C_2, \cdots C_l$, $C_i \cap C_j = \emptyset$, if $i \neq j$, where each cluster is disjoint from the others and shares the least mutual information possible. This clustering step can extract meaningful patterns and provide insights into the underlying trading structure of the HFT data.

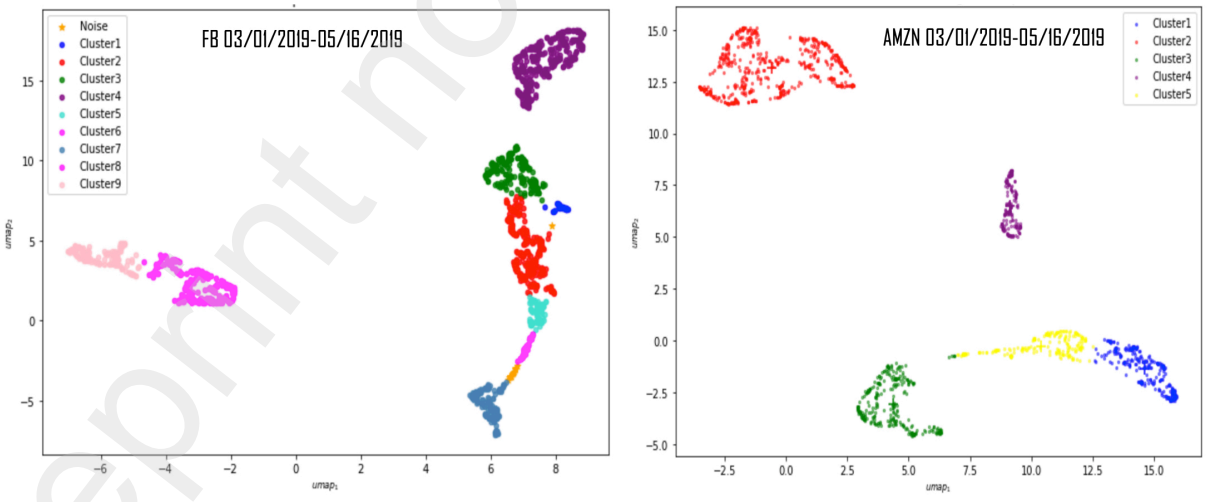

Fig 1. DBSCAN clustering results of the UMAP embeddings of FB and AMZN from 03/01/2019 to 05/16/2019. The clustering results may offer some insights into the trading dynamics, but they lack a clear interpretation and ignore the trading marker identification.

Figure 1 displays the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) clustering results after applying Uniform Manifold Approximation and Projection (UMAP) dimension reduction to two intraday datasets of HFT data from Facebook (FB) and Amazon (AMZN), covering the period from March 1, 2019, to May 16, 2019. UMAP is a state-of-the-art dimension reduction algorithm that excels at capturing local data behavior, while DBSCAN is a density-based clustering algorithm that can identify naturally grouped clusters for data with arbitrary shapes [17-19]. The plots on the left and right side of Figure 1 indicate that there were 10 clusters identified in the FB trading dataset and 5 clusters in the AMZN trading dataset. The spatial relationships between the clusters suggest that AMZN may have more 'orthogonal' transactions in trading than FB, as their clusters are more separated from each other. These insights are valuable for understanding the structure of the data and identifying trading patterns that may not be immediately obvious.

While the DBSCAN clustering algorithm can reveal some information on the trading dynamics of HFT data, it lacks a comprehensive explanation of the clusters. It cannot interpret why different transactions are grouped in one cluster, nor can it provide insight into the trading implications of these clusters that traders need to know to make informed decisions. The lack of interpretability is a critical issue for ML methods used in HFT because of the high stakes involved. It is crucial for traders to understand the trading implications of different clusters to make their trading decisions more adaptive and profitable.

However, most of the ML models used in HFT are non-interpretable "black box" models, including deep learning models, which demonstrate effectiveness but fail to explain why they work [12-13]. As a result, it would be difficult for traders to trust these models, even if their results are sound. To make trading dynamics discovery more explainable and understandable, explainable ML models should be employed, which can provide clear and interpretable insights into the different latent trading mechanisms disclosed by the ML methods. Therefore, achieving interpretability in ML methods for trading dynamics discovery should be a primary focus in Fintech.

2.3 The Explainable machine learning standards

Although trading markers are commonly used in HFT trading systems, they have not been extensively studied in HFT literature. Previous studies have explored extreme price movements (EPM), which are somewhat related to trading markers [15]. However, EPM cannot be considered as a meaningful buying or selling point because it does not take into account the practical trade prices. In our research, we define HFT trading markers based on our previous work [6].

Trading marker. Given a set of transactions $\{x_i\}_{t_0}^{t}$ in a trading interval [t_0, t], a transaction x_t at time t is considered as a trading marker provided its price change ratio is more than a threshold η (e.g., η =0.01%): $|p(t) - p(t_0)|/|p(t_0)| \geq \eta$, where $p(t_0)$and $p(t)$ are transaction prices at time t_0 and t, respectively.

Different threshold values determine different trading markers. General markers have a small price change threshold (e.g., η = 0.01%) and are widely used to make marginal profits through large volumes. To avoid confusion with ad-hoc price changes, they usually have a minimum volume requirement (e.g., volume >

1000). Global markers, on the other hand, have a large threshold (e.g., η = 0.2%) and appear as abrupt price changes caused by large-scale buying or selling in a short trading period. While they appear less frequently than general markers, global markers have a greater impact on trading and provide more liquidity. Our study focuses on global markers in trading dynamics discovery.

Trading dynamics abstraction. Given an HFT dataset of a security with n transactions and p variables $X = \{x_i\}_{i=1}^{n}$, $x_i \in \Re^p$ during a time period $[t_1, t_2]$, trading dynamics can be abstracted as a functional $T_d(X) = \{X_m, \delta(X), \eta(X)\}$. X_m denotes the trading markers, which are significant buying or selling points in trading. The function $\delta(X)$ captures the global transaction pattern or structure, revealing how the trades are organized and interrelated. $\eta(X)$ represents the trading status of each transaction, providing information such as whether it was a sell or a buy signal, the price at which it occurred, and the time of the trade.

To effectively uncover meaningful trading dynamics, an explainable ML model must meet following standards according to the abstraction. Firstly, it should clearly identify and meaningfully explain trading markers, which are key components of trading dynamics and serve as an essential source of HFT liquidity. These markers are local maximum or minimum prices of the HFT price curve within a given time interval. Without the inclusion of trading markers, it would be difficult to accurately explain the underlying dynamics of trading.

Secondly, the ML model should provide comprehensive information on the status of each transaction in trading. By understanding the status of each trading transaction, traders can gain deeper insights into different trading patterns, identify extreme price movements, and decipher the origins of trading markers. For example, annotating the status of each transaction in trading clusters can provide a more meaningful and intuitive visualization of the clustering information, thereby enabling traders to make informed decisions with greater confidence.

Finally, the ML model should unveil trading dynamics by building a more informative feature space that overcomes the issue of feature scarcity in HFT data. The existing available features may be inadequate to provide knowledge-based trading behavior unveiling and trading marker identification. Therefore, it is essential to enrich the feature space by including additional meaningful features in trading dynamics discovery. By doing so, the model can uncover valuable trading dynamics information, reduce noise and redundancy, and ultimately assist traders in making better-informed decisions.

2.4 Trading map

Our study presents an innovative, explainable machine learning approach called Feature-Interpolation-based dimension reduction SCAN (FIDR-SCAN) for discovering high-frequency trading (HFT) dynamics. FIDR-SCAN constructs a "trading map," which is technically an implementation of the trading dynamics abstraction $T_d(X)$. Each trading map allows traders to identify trading markers, annotate transaction statuses, and uncover trading behavior patterns in a feature space enriched with meaningful information.

The trading map can be thought of as a 2D signature of a stock's trading behavior, which enables traders to examine the behaviors of different securities, observe market tendencies, and identify buying and selling points. Additionally, this approach provides more insights into the significance of trading markers and opens up the possibility of reusing them. By merging the trading maps of a stock over a sequence of trading periods, we can address the essential question: "What happens to the stock in HFT?"

Figure 2 shows AAPL's trading map using HFT data from February 1st, 2019 to February 22nd, 2019. The map displays only markers falling in RSI intervals [70, 100] or [0, 20]. RSI is a new feature added to the HFT feature space that indicates whether a stock is oversold or overbought [20]. The trading map groups transactions into 'up' or 'down' clusters, and each marker is annotated with corresponding time, prices, and RSI. By incorporating RSI into the trading map, we are able to categorize markers more effectively, and we find that around half of the markers show strong overbought or oversold signals. The RSI also helps identify essential trading markers by extreme RSI values, which are relatively far from their nearest transaction clusters.

Additionally, each transaction is marked as core, reachable, or outlier points, representing normal, inflection, and trading marker transactions, respectively. We prove that trading markers in the trading map are meaningful peaks/bottoms or intermediate peaks/bottoms in the feature space to support more explainable trading marker discovery.

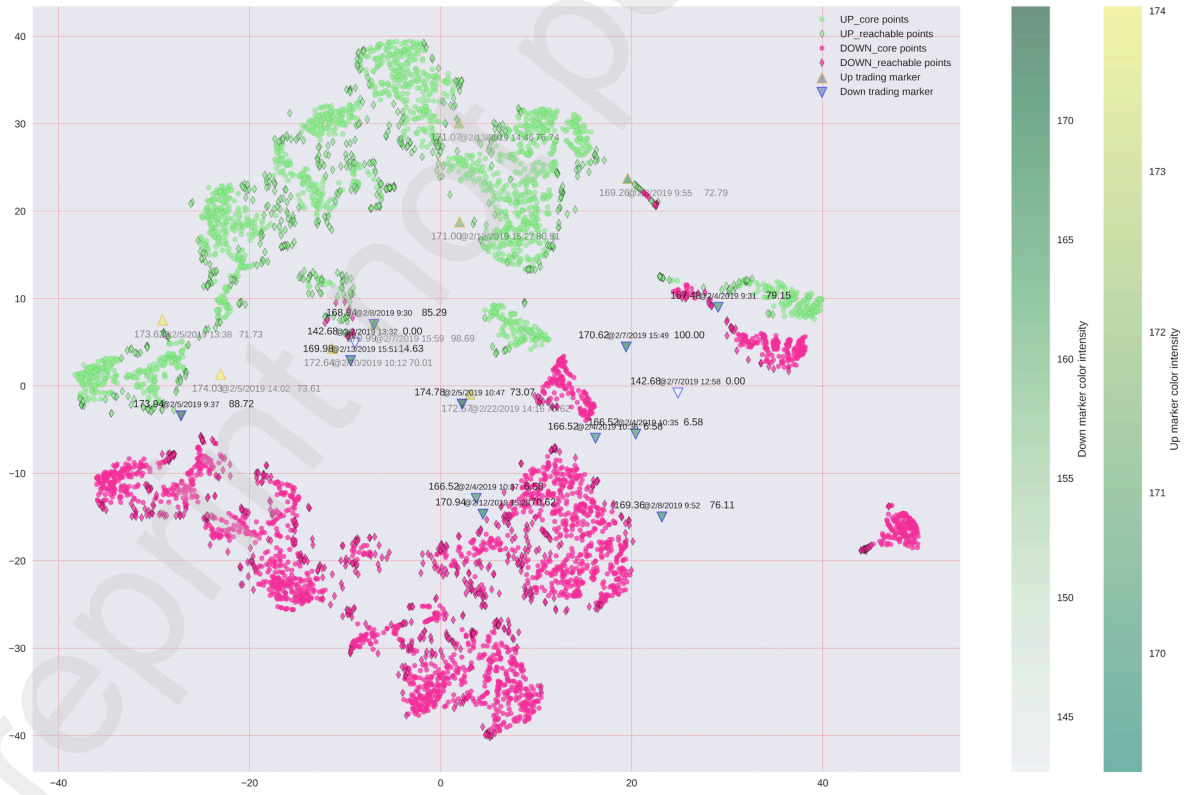

Fig 2. The trading map of AAPL HFT data from 2019/02/01 to 2019/02/22 under FIDR-SCAN. The trading markers that are outliers in DBSCAN clustering are marked with prices, time, where only the trading markers with RSI >=70 or <=20 are illustrated.

The paper is organized into eight sections. In Section 3, we describe the HFT datasets we used in our study. Section 4 introduces our explainable machine learning (ML) algorithm, FIDR-SCAN, which we used to uncover HFT dynamics. We also present novel feature interpolation techniques that we used to enhance the feature space, making it easier to discover trading dynamics that are more interpretable. In Section 5, we validate and analyze trading maps. In Section 6, we evaluate the effectiveness of our trading markers from the perspective of trading profitability via AI trading. In Section 7, we discuss potential enhancements and extensions of our proposed algorithm. Finally, in Section 8, we conclude the study.

3. HFT data and quantification

3.1 **HFT data and Kurtosis analysis**

We obtained the high-frequency trading (HFT) data used in this study from the "intraday prices" API provided by the IEX cloud [21]. We developed our software, *HFTGlean,* to retrieve intraday data for different stocks by communicating with the API. For this study, we selected four representative stocks from different sectors, including three large-cap HFT stocks (AAPL, BAC, and WMT) and one mid-cap stock (AEO) from IT, Banks, Retail, and Fashion sectors.

We collected data for each stock from February 1, 2019, to February 22, 2019, resulting in 5850 observations across 9 features. The 9 features include 4 primitive price variables (high, low, open, and close), 1 volume variable, and 4 auxiliary variables. The high, low, open, and close prices represent the highest, lowest, beginning, and ending prices in each one-minute interval from the raw HFT data. The volume variable is the sum of all volumes of the original HFT transactions in the interval.

The 4 auxiliary variables are *ChangeOverTime, MarketAverage, NotionalValue*, and *NumberOfTrades*. The *ChangeOverTime* variable represents the stock price change ratio in each one-minute interval and is generally a small ratio, with AAPL having a median price change ratio of 0.002 or 0.18% in the trading period. AEO has the lowest price change ratios per minute among the selected stocks, compared to the large caps AAPL, BAC, and WMT. The *MarketAverage* variable represents the average price of the stock per minute, while the *NotionalValue* variable refers to the total value of the position in trading, i.e., the total amount of the stock value at its spot price. The *NumberOfTrades* variable indicates the trading frequency of the stock by representing the number of trades placed per minute.

Figure 2 illustrates the probability density functions of the close price, price change ratio (1-min) (*ChangeOverTime)*, and log (volume) for HFT data on four stocks: APPL, AEO, BAC, and WMT. The non-parametric Gaussian kernel density estimation method was used to generate these distributions, revealing that each stock has its own distinct trading behavior, resulting in different distributions [22]. The figure

also shows that these distributions do not conform closely to normal or log-normal distributions, indicating that each security has its own unique trading dynamics.

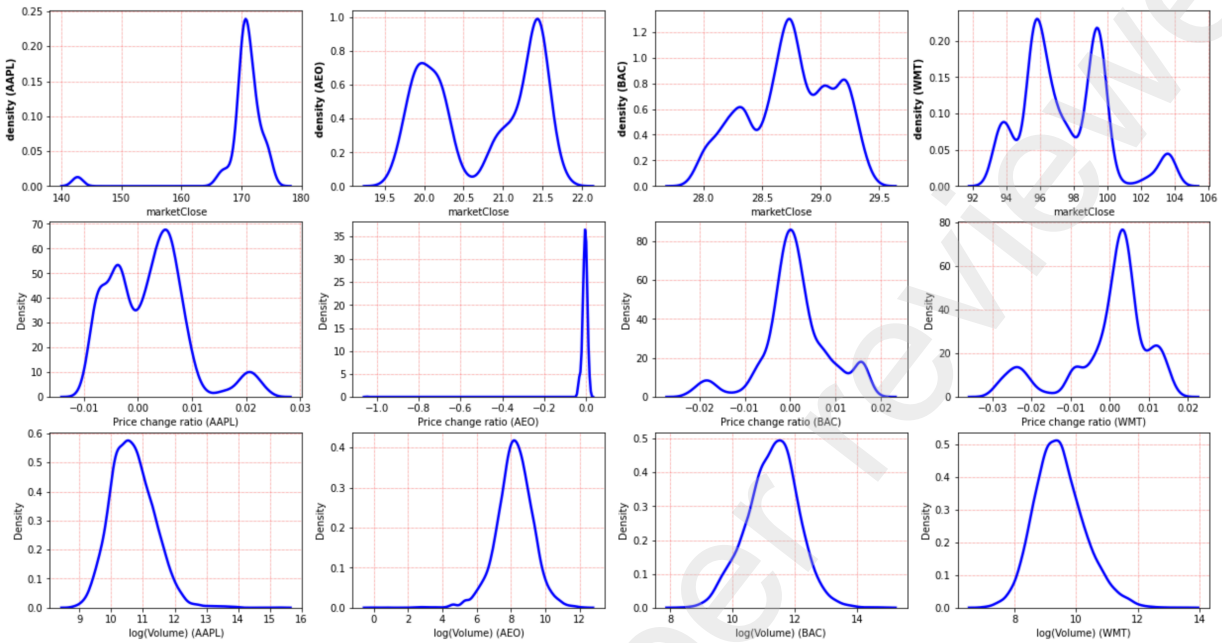

Fig. 3 The close price, price change ratios (1-min), and log (volume) probability density function estimation of the AAPL, AEO, BAC and WMT HFT data February 1, 2019, to February 22, 2019.

Kurtosis analysis. The HFT data's kurtosis analysis further confirms the observation. AAPL's price distribution follows a leptokurtic distribution with a high kurtosis value of 22.2974, indicating a larger number of outliers in trading compared to other stocks. On the other hand, AEO, BAC, and WMT's price distributions are platykurtic with negative kurtosis values indicating fewer outliers in trading. The price change ratio distributions of AAPL, BAC, and WMT are mesokurtic because their kurtosis values fall within the range of 0 to 3, which suggests a lack of extreme price changes. However, the price change ratio distribution of AAPL has an extreme kurtosis value of 1093.53, indicating a higher number of extreme price changes compared to other stocks. Additionally, AEO's log volume distribution is "leptokurtic" with a kurtosis value of 4.0233, while the log volume distributions of the other stocks are "mesokurtic."

3.2. HFT data quantification

High-frequency trading (HFT) data can exhibit complex and dynamic patterns that are challenging to analyze using conventional methods. To address this issue, we employ a new measure called the variance concentration ratio (VCR) proposed by the first author to quantify the distribution of variance in HFT datasets [6,23]. The VCR is defined as the ratio between the largest singular value of the dataset and the total sum of all singular values. Specifically, given an HFT dataset with n transactions and p variables $X \in \Re^{n\times p}$, the VCR is defined as:

$$\beta(X) = s_1 / \sum_{i=1}^{p} s_i \quad (1)$$

where s_i is the i^{th} singular value of X, $i = 1,2 \cdots p$. The VCR aims to measure how data variance distributes along the first singular-value direction. The VCR acts as an index to signal the data variance level of HFT data: $\int x^2 f(x) dx$, where $f(x)$ is the unknown probability density function (*p.d.f.)* of the transaction random variable.

Theorem 1. Given an HFT dataset with n transactions and p variables $X \in \Re^{n \times p}$, then the variance concentration ratio (VCR) β(X) falls in the interval $\left[\frac{1}{\sqrt{p}} \frac{\|X\|_2}{\|X\|_F}, \frac{\|X\|_2}{\|X\|_F}\right)$, namely,

$$\frac{1}{\sqrt{p}} \frac{\|X\|_2}{\|X\|_F} \leq \beta(X) < \frac{\|X\|_2}{\|X\|_F} \quad (2)$$

Proof. According to the Cauchy-Schwarz inequality, we have $\left(\sum_{i=1}^{p} s_i\right)^2 \leq \sum_{i=1}^{p} s_i^2 p$. Then $\beta(X)^2 = \frac{s_1^2}{\left(\sum_{i=1}^{p} s_i\right)^2} \geq \frac{s_1^2}{\sum_{i=1}^{p} s_i^2 p}$. The upper bound of $\beta(X)^2$ can be estimated as $\frac{s_1^2}{\left(\sum_{i=1}^{p} s_i\right)^2} < \frac{s_1^2}{s_1^2 + s_2^2 + \cdots s_p^2} = \frac{s_1^2}{\|X\|_F{}^2} = \frac{\|X\|_2^2}{\|X\|_F{}^2}$. The $\|X\|_2$ and $\|X\|_F$ represent the spectral norm and Frobenius norm of the HFT data respectively. Thus, the VCR will fall in the interval: $\left[\frac{1}{\sqrt{p}} \frac{\|X\|_2}{\|X\|_F}, \frac{\|X\|_2}{\|X\|_F}\right)$, *i. e.*, $\frac{1}{\sqrt{p}} \frac{\|X\|_2}{\|X\|_F} \leq \beta(X) < \frac{\|X\|_2}{\|X\|_F}$.

As we mentioned before, HFT data is energy concentration data with the data variance mostly concentrated in the first singular value direction. The 'energy concentration' is rooted from the high volume and high velocity of HFT data. We officially define the energy concentration data according to VCR as follows.

Energy concentration data. Given an HFT dataset with n transactions and p variables $X \in \Re^{n \times p}$, it is energy-concentration data if and only if its VCR satisfies $\beta(X) > \frac{2}{p}$.

We have found that high-frequency trading (HFT) data exhibits significantly higher values of variance-to-covariance ratios (VCR) compared to other financial data, including option data and financial risk data. Only cryptocurrency data demonstrates similar or equally high VCRs to HFT data (data not presented). This suggests that HFT data is more concentrated in energy than other types of financial data. Furthermore, it is important to note that the VCR values are sensitive to the method of normalization used. The VCR values for raw HFT data can reach up to approximately 99%. However, when standard scaling is applied, the VCR values tend to be lower than those of other normalization methods, with a maximum value of around 30%, still greater than $\frac{2}{p}$ (0.22, p=9) in our data. In this study, the variables in the HFT data are heterogeneous and measured on different scales and units. To enable effective comparison and analysis, we employ standard normalization to normalize the data though other options can be also applied.

Figure 4's left plot presents comparisons of VCRs between raw data and data from the five normalization methods, including *standard*, *minmax*, *robust*, *maxabs*, and *power-transform* normalization, across the four HFT datasets [24]. The results show that the raw data and *maxabs* normalized data have the highest VCR values. In the middle plot, we compare the explained variance ratios of the first two principal components (PCs) under the raw and five normalized data. The findings confirm the previous VCR results, where the

normalized data has the smallest explained variance ratios for the first two PCs, while the raw data and *maxabs* normalized data have the highest explained variance ratios. In the right plot of Figure 4, we present the explained variance ratios of the first two PCs under standard scaling. The results indicate that three datasets (AEO, BAC, WMT) achieve more than 80% of the explained variance ratios for the first two PCs.

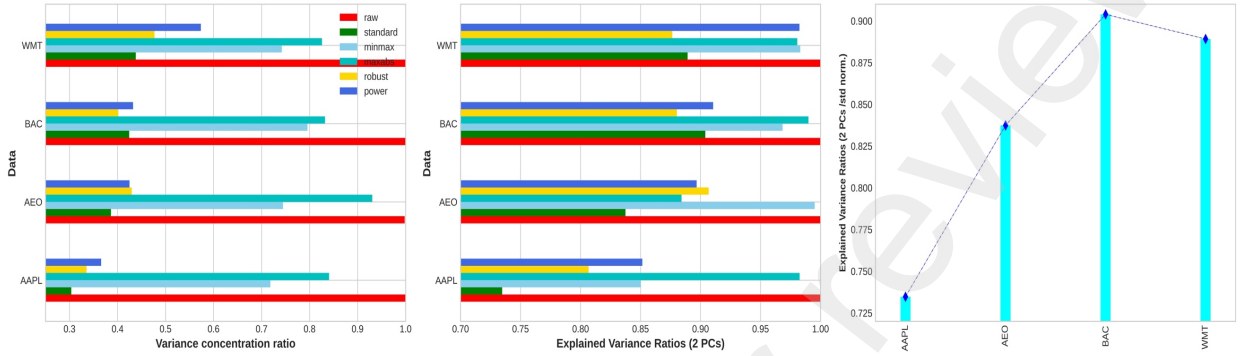

Fig. 4 compares the VCR values and the explained variance ratios of the first two principal components (PCs) between raw data and data from the five normalization methods (standard, minmax, robust, maxabs, and power transform normalization) across the four HFT datasets. The left and middle plots show the VCR values and the first 2 PC explained variance ratios, respectively. The results indicate that the raw data and maxabs normalized data have the highest VCR values and the highest explained variance ratios for the first two PCs. The right plot compares the explained variance ratios of the first two PCs under standard scaling for the four datasets, showing that three datasets (AEO, BAC, WMT) achieve more than 80% of the explained variance ratios for the first two PCs.

The quantification of HFT data using the VCR provides a strong theoretical foundation for uncovering the underlying trading dynamics. To effectively leverage the unique characteristics of HFT data, which is highly concentrated in energy, it is desirable to develop machine learning algorithms that can effectively capture and analyze this property. Technically, energy concentration data is suitable for dimension reduction to examine more subtle data behaviors.

4. Explainable machine learning for trading dynamics discovery

4.1 Explainable ML for trading dynamics discovery

Figure 5 illustrates the flowchart for explainable machine learning (ML) in discovering high-frequency trading (HFT) dynamics by generating a trading map through adherence to the proposed standards. The flowchart commences with the construction of an informative feature space to reveal complex trading behaviors, thereby overcoming the issue of feature scarcity in HFT data. To construct the feature space, this study proposes a novel feature interpolation approach. Subsequently, an 'explainable dimensional reduction algorithm' is employed on the HFT data in the new feature space to generate an explainable trading embedding, which enables the retrieval of more concealed trading dynamics.

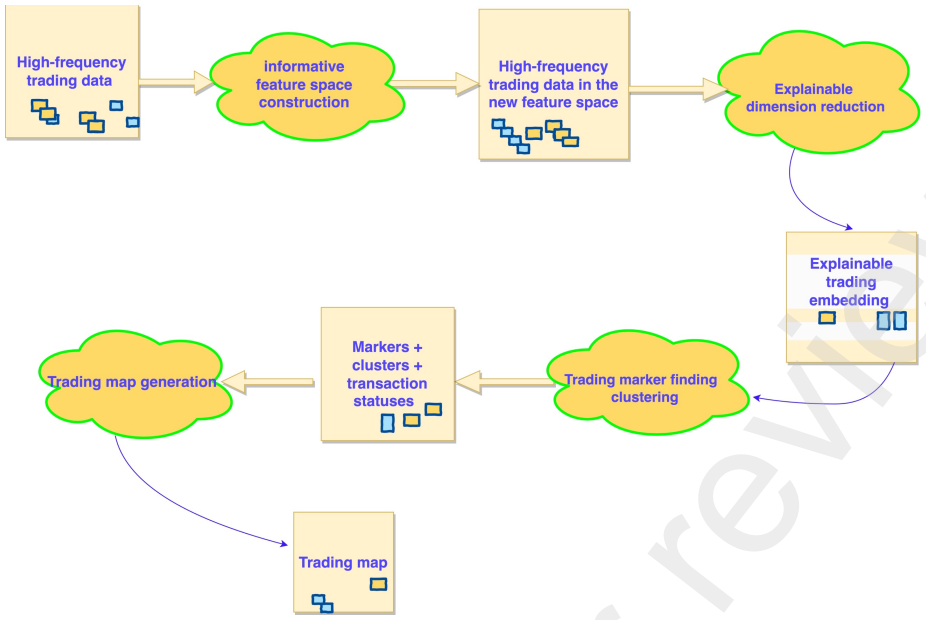

Fig. 5 The flowchart of explainable ML in discovering HFT trading dynamics by generating a trading map. To better capture the complex dynamics of HFT trading, there is an urgent need for an explainable ML algorithm that follows the proposed standards for explainable ML and leverages the distinctive features of HFT data. Such an algorithm should incorporate the trading dynamics abstraction $T_d(X) = \{X_m, \delta(X), \eta(X)\}$ by identifying crucial trading markers, grouping transactions, and marking trading statuses.

The term "explainable dimensional reduction algorithm" refers to a dimension reduction algorithm f_d that is capable of preserving the intrinsic data structure of the HFT data $X \in \Re^{n\times(p+l)}$ after feature interpolation most effectively, i.e., $f_d: X \to E = \{e_i\}_{i=1}^n$, $e_i \in \Re^k, k < (p+l)$, $s.t.$ $f_d: x_i \to e_i$, where E is the explainable trading embedding capturing the essential data characteristics of X while mitigating noise. According to recent research, t-SNE outperforms other popular dimension reduction techniques such as UMAP, principal component analysis (PCA), locally linear embedding (LLE), Hessian locally linear embedding (HLLE), and Local tangent space alignment (LTSA) in keeping intrinsic data structures in terms of maintaining the degree of locality preservation [23]. Therefore, we have chosen to utilize t-SNE to implement f_d in our flowchart to obtain an explainable trading embedding.

After obtaining the explainable trading embedding, we need to apply an interpretable clustering algorithm that can identify trading markers, mark the status of each transaction, and find transaction clusters sharing trading similarity. Unfortunately, general clustering methods like K-means, Affinity Propagation, and spectral clustering methods, and their variants cannot accomplish all of these tasks simultaneously [25-26].

4.1.1 Interpreting DBSCAN: normal transactions, inflection transactions, and trading markers

However, we have devised a different approach by interpreting the density-based clustering algorithm DBSCAN to accomplish it. DBSCAN classifies input data points into core, reachable, and outliers [19]. The core and reachable points are interpreted as normal and inflection transactions respectively. The normal

transactions are those reflecting common trading scenarios or general data behaviors in trading. The inflection transactions are those representing less common or less dominant behaviors in trading, and they are closer to trading markers spatially. The outliers are interpreted as trading markers or at least trading marker candidates that represent rare or unusual behaviors in trading, indicating unexpected or extreme market conditions. We have theoretically proven we can find trading markers through DBSCAN clustering for the explainable trading embedding. In addition, the clustering structures from DBSCAN reveal the transaction clusters in trading. Thus, the DBSCAN after interpreting can accomplish the tasks simultaneously. Finally, we apply a trading map marking step that annotates the DBSCAN clustering results to create the final trading map. This approach provides a more accurate and interpretable clustering result for trading data, enabling better discovery of trading dynamics.

4.2 HFT feature interpolation

We provide more details about the feature interpolation techniques proposed in this study.

Feature interpolation (FI). Given an HFT dataset with n observations and p variables: $X = (x_{ij}) \in \Re^{n\times p}$, feature interpolation is a technique that creates a new feature space, denoted as S', by combining the original features $x_{.1}, x_{.2}, \cdots x_{.p}$ with a set of 'interpolated features' $x_{.p+1}, \cdots x_{.p+l}$. Each interpolated feature is generated by applying a functional map $f_k(.)$ to the original features, such that $x_{.p+k} = f_k\left(x_{.1}, x_{.2}, \cdots x_{.p}\right)$, where $k = 1,2 \ldots l$. Thus, the new feature space S' is spanned by the original features and the interpolated features:

$$S' = span\left(x_{.1}, x_{.2}, \cdots x_{.p}, x_{.p+1}, \cdots x_{.p+l}\right) \tag{3}$$

In general, $f_k(.)$ is a nonlinear functional, although it can be linear in some cases. This is an important consideration when dealing with HFT data after feature interpolation. For example, $f_k(.)$ can be a nonlinear map that calculates the 'local' Relative Strength Index (RSI) based on prices over a 14-minute interval consisting of 14 1-minute prices. This example highlights the complex nonlinear transformations that $f_k(.)$ can undergo to derive features from the original data.

Lemma 1. The singular values of matrix $[A|B] \in \Re^{n\times(p+l)}$ includes all singular values of matrix $A \in \Re^{n\times p}$, where matrix B is added as additional columns to the matrix A.

Proof. Let σ be a singular value of A with a corresponding right singular vector u. Consider the vector $[A|B]u$. Its norm is $\|[A|B]u\|^2 = \sigma^2 + \|Bw\|^2$, where $w \in \Re^1$. Since $\|Bw\|^2$ is non-negative, σ is also a singular value of $[A|B]$ with a corresponding right singular vector of the form $[w'; w]$, where w' is any vector in $\Re^p$ and w is any vector in $\Re^1$. Therefore, all singular values of A are also singular values of $[A|B]$, and thus the singular values of $[A|B]$ include all singular values of A.

Theorem 2. Given an HFT dataset with n transactions and p variables $X \in \Re^{n\times p}$, another HFT data $X_f = [X|X_l] \in \Re^{n\times(p+l)}, where\ X = \left[x_{.1}, x_{.2}, \cdots x_{.p}\right], X_l = \left[x_{.p+1}, \cdots x_{.p+l}\right]$, is obtained by doing feature interpolation for X. Then we have the following results

1) $Tr(X^TX) < Tr\left(X_f^T X_f\right)$.

2) The VCR relationship: $\frac{\sqrt{Tr(X^TX)}}{\sqrt{p+l}\sqrt{Tr\left(X_f^TX_f\right)}}\beta(X) < \beta\left(X_f\right) < \frac{\sigma_1^*}{\sigma_1}\beta(X)$, where σ_1^* and σ_1 are the first singular values of X_f and $X_f, \sigma_1^* \geq \sigma_1$.

Proof. Without loss of generality, we assume each map $f_k: \left(x_{.1}, x_{.2}, \cdots x_{.p}\right) \to x_{.p+k}, k = 1,2 \cdots l$ is nonlinear. According to Lemma 1, the singular values of $X_f = [X|X_l]$ will include other singular values: $\sigma_{p+1}, \sigma_{p+2}, \cdots \sigma_{p+l} > 0$ besides including the original singular values $\sigma_1, \sigma_2, \cdots \sigma_p$. Therefore,

$$Tr(X^TX) = \sum_{i=1}^{p} \sigma_i^2 < \sum_{i=1}^{p+l} \sigma_i^2 = Tr\left(X_f^TX_f\right) \tag{4}$$

It is noted that the singular values of X: $\sigma_1, \sigma_2, \cdots \sigma_p$ have the relationships: $\sigma_1 \geq \sigma_2, \geq \cdots \sigma_p$, but the extra singular values $\sigma_{p+1}, \sigma_{p+2}, \cdots \sigma_{p+l}$ may contain the entries greater than σ_1.

The VCR of X: $\beta(X) = \frac{\sigma_1}{\sigma_1+\sigma_2+\cdots+\sigma_p} > \frac{\sigma_1}{\sigma_1+\sigma_2+\cdots+\sigma_{p+\sigma_{p+1}\ldots+\sigma_{p+l}}}$, let $\sigma_1^* = \max\{\sigma_1, \sigma_2, \cdots, \sigma_{p,}, \sigma_{p+1,} \cdots \sigma_{p+l}\}$, which is the first singular value of X_f. Then we have $\beta(X) > \frac{\sigma_1^*}{\sigma_{1,}+\sigma_2+\cdots+\sigma_{p,}+\sigma_{p+1}+\cdots+\sigma_{p+l}}\frac{\sigma_1}{\sigma_1^*} = \beta\left(X_f\right)\frac{\sigma_1}{\sigma_1^*}$, thus, $\beta\left(X_f\right) < \frac{\sigma_1^*}{\sigma_1}\beta(X)$.

Similarly, $\beta(X) < \frac{\sigma_1^*}{\sigma_1+\sigma_2+\cdots+\sigma_p} = \beta\left(X_f\right)\frac{\sigma_1+\sigma_2+\cdots+\sigma_p+\cdots+\sigma_{p+l}}{\sigma_1+\sigma_2+\cdots+\sigma_p} < \beta\left(X_f\right)\frac{\sqrt{p+l}\sqrt{Tr\left(X_f^TX_f\right)}}{\sqrt{Tr(X^TX)}}$, thus, $\beta\left(X_f\right) > \frac{\sqrt{Tr(X^TX)}}{\sqrt{p+l}\sqrt{Tr\left(X_f^TX_f\right)}}$.

Therefore,

$$\frac{\sqrt{Tr(X^TX)}}{\sqrt{p+l}\sqrt{Tr\left(X_f^TX_f\right)}}\beta(X) < \beta\left(X_f\right) < \frac{\sigma_1^*}{\sigma_1}\beta(X) \tag{5}$$

4.2.1 Data entropy of HFT data

Feature interpolation can enhance the discovery of explainable trading dynamics by creating a more informative feature space through increasing the entropy of the input HFT data. To measure the impact of feature interpolation on HFT data, data entropy is employed as a quantifying metric. Han *et al.* introduced the concept of data entropy to explain the performance of marker discovery in HFT data [6]. We have the following definition for data entropy and its range estimation.

Data entropy. Given a dataset with n observations and p variables: $X \in \Re^{n\times p}$, its data entropy is defined as

$$h(X) = -\sum_{i=1}^{p} u_i \, log_2 \, u_i \tag{6}$$

where $u_i = \frac{s_i}{\sum_{i=1}^{p} s_i}$, and s_i is the ith singular value of X.

Lemma 2. HFT entropy range estimation. Given an HFT dataset with n transactions and p variables $X \in \Re^{n\times p}$, its data entropy falls in the following interval $(1 - \frac{\|X\|_F^2}{Tr(X^TX)^2})\, log_2\, e \; \leq h(X) \leq (p-1)\, log_2\, e$.

Proof. According to the definition of data entropy, for all $u_i > 0$, we see $1 - \frac{1}{u_i} \leq \ln u_i \leq u_i - 1$, then we have $\sum_{i=1}^{p} u_i(1 - \frac{1}{u_i}) \leq \sum_{i=1}^{p} u_i \ln u_i \; \leq \sum_{i=1}^{p} u_i(u_i - 1)$, multiplying -1 on both sides and simplifying it, i.e., $\sum_{i=1}^{p} u_i(1 - u_i) \leq -\sum_{i=1}^{p} u_i \ln u_i \; \leq \sum_{i=1}^{p}(1 - u_i)$, plugging the $\sum_{i=1}^{p} u_i = 1, \sum_{i=1}^{p} u_i^2 = \frac{s_1^2+s_2^2+\cdots s_p^2}{\left(\sum_{i=1}^{p} s_i\right)^2} = \frac{\|X\|_F^2}{Tr(X^TX)^2}$, to

the inequality, we get the final answer: $1-\frac{\|X\|_F^2}{Tr(X^TX)^2} \le -\sum_{i=1}^{p} u_i \ln u_i \le p-1$. Let $-\sum_{i=1}^{p} u_i \ln u_i = -\sum_{i=1}^{p} u_i \frac{log_2 u_i}{log_2 e}$, we have the final result: $0 < (1-\frac{\|X\|_F^2}{Tr(X^TX)^2})\, log_2 e \le h(X) \le (p-1)\, log_2 e$.

The HFT entropy range estimation suggests that as the number of variables, p, increases, the entropy of the HFT dataset also increases. This is supported by the upper bound of the entropy estimation, which confirms that a large number of variables results in a high entropy HFT dataset. Therefore, adding more variables to the dataset will increase its entropy.

However, the addition of more variables also perturbs the original system, revealing more latent data behaviors that can increase the entropy of the entire dataset. As a result, we can conclude that feature interpolations have the following theorem:

Theorem 3 Given an HFT dataset with n transactions and p variables $X \in \Re^{n\times p}$, $X_f = [X|X_l] \in \Re^{n\times(p+l)}$, where $X = [x_{.1}, x_{.2}, \cdots x_{.p}], X_l = [x_{.p+1}, \cdots x_{.p+l}]$, is the HFT data obtained by doing feature interpolation for X, then $h(X_f) > h(X)$, i.e., adding interpolated variables will increase the data entropy.

Proof. Let $\sigma_1, \sigma_2, \cdots \sigma_p$ be the singular values of X and $\sigma_1, \sigma_2, \cdots \sigma_p, \sigma_{p+1}, \sigma_{p+2}, \cdots \sigma_{p+l}$ be the singular values of X_f.

$h(X_f) = -(\sum_{i=1}^{p+l} u_i \log_2 u_i) = -(\sum_{i=1}^{p} u_i \log_2 u_i) + [-(\sum_{i=p+1}^{p+l} u_i \log_2 u_i)]$, where $u_i = \frac{\sigma_i}{\sum_{i=1}^{p+l}\sigma_i}, i = 1,2,\ldots p+l$. $h(X) = -(\sum_{j=1}^{p} v_j \log_2 v_j)$, $v_j = \frac{\sigma_j}{\sigma_1+\sigma_2\cdots+\sigma_p}, j = 1,2,\ldots p$. Since $u_i \log_2 u_i < v_i \log_2 v_i, i = 1,2,\ldots p$, we have $-(\sum_{i=1}^{p} u_i \log_2 u_i) > h(X)$. Thus, $h(X_f) > h(X)$.

The theorem is supported by real data entropy results. We have calculated the data entropies for four HFT datasets (AAPL, BAC, WMT, and AEO) used in this study, before and after feature interpolations. Our findings show that the data entropies increase after the interpolation procedure. Table 1 compares the entropies of the four datasets before and after feature interpolations, demonstrating a noticeable increase from 3.8% to 12.82%. From an information theory perspective, higher entropy data can be more explainable than lower entropy data. This finding suggests that the feature interpolation procedure can help to increase the explainability of the HFT datasets we examined in our study. The details about fetaure interpolation implementation can be found in the following subsection.

Table 1 Entropy comparisons before and after feature interpolations (FI)

	AAPL	BAC	AEO	WMT
Before FI	0.001463	0.02075	0.006146	0.005376
After FI	0.001632	0.02341	0.006382	0.00563

4.2.2 Feature interpolation implementation.

To implement the proposed feature interpolations, we added two groups of interpolated features to the HFT data. The first group localizes four traditional trading indices that capture oversold or overbought signals, the magnitude of price changes, and price trend strength. We achieved this by updating their trading unit from a day to a 1-minute trading interval. These indices include Bollinger Bands (BB), Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), and Average Directional Index (ADX) [20].

The second group comprises two new liquidity measures proposed in this study, namely pseudo-volatility and close-off-high (COH), in addition to the classic log return. Unlike the original primitive features, the interpolated features are obtained from different nonlinear transforms that model different latent data characteristics in trading. Therefore, these features are more interpretable and informative in describing the behaviors of HFT data than the original ones. We briefly introduce these metrics as follows to make this presentation more self-contained.

The Bollinger Bands are a useful tool for identifying overbought or oversold signals in trading [20]. It consists of three measures in m trading intervals: a moving average $MA = (\sum_{i=1}^{m} p_i)/m,$ an upper band $B_u = \frac{1}{m}\sum_{i=1}^{m} tp_i + 2\sigma_m$, and a lower band $B_m = \frac{1}{m}\sum_{i=1}^{m} tp_i - 2\sigma_m$. The typical price tp_i is the average of high, low, and close prices $tp_i = \frac{p_i^{high} + p_i^{low} + p_i^{close}}{3}$ in the i^{th} trading interval and σ_l is the standard deviation of the typical price in the m trading intervals. When the stock price in the i^{th} trading interval goes below the lower band, i.e., $p_i < B_l,$ it indicates that the stock is oversold. Otherwise, it signals an overbought condition when $p_i > B_u$.

Relative Strength Index (RSI) is an index between 0 and 100 that signals whether a stock is oversold or overbought by measuring the strength or weakness of a stock's price. It is calculated using average price gains and losses for a given number of trading periods (e.g., 20 trading intervals in HFT). RSI > 70 means a security is overbought or overvalued RSI <25 means the security is oversold or undervalued. The detailed calculation can be found in [20].

Moving Average Convergence Divergence (MACD) is an index to measure the buy and sell signal by evaluating price strength [20]. The long position is suggested when the i^{th} trading interval price is above the MACD line value: $p_i > MACD_i$; Otherwise a short position is suggested. The MACD in the i^{th} trading interval is calculated as the difference between two exponential moving average (EMAs) [2]: $MACD_i = EMA_{i,k} - EMA_{i,m}$, where $EMA_{i,k}$ and $EMA_{i,m}$ are the EMAs of the k^{th} and m^{th} trading intervals counting from the existing i^{th} interval and m-k=12.

The Average Directional Index (ADX) is an index that measures the strength of a security's price trend and ranges from 0 to 100 [20]. An ADX value in the range of 75-100, 50-75, or 25-50 indicates an extremely strong, very strong, or strong up/down price trend, respectively. In contrast, a low ADX value (e.g., <25) suggests that there is no trend signal or a weak trend in the security price. The ADX value is calculated based on a moving average (MA) of the price range expansion over a certain trading period (e.g., 14 trading intervals in HFT).

Close-off-high (COH) is a measure that evaluates the relationship between the closing price and the extreme price in every trading interval. A higher COH value indicates higher price oscillations, and extreme large COH values signal an extreme change in price. The COH for the i^{th} trading interval is calculated using the following formula:

$$COH_i = 2\frac{p_i^{high} - p_i^{close}}{p_i^{high} - p_i^{low}} - 1 \tag{7}$$

where p_i^{high}, p_i^{close}, p_i^{low} are the high, close, and low prices in the i^{th} trading interval.

Pseudo-volatility represents the ratio of the difference between the highest and lowest price over the opening price in each time interval. The larger price fluctuation, the higher pseudo-volatility in each time interval. The pseudo-volatility for the i^{th} trading interval is calculated using the following formula:

$$\mathrm{PV}_i = \frac{p_i^{high} - p_i^{low}}{p_i^{open}} \tag{8}$$

Where p_i^{high}, p_i^{low}, p_i^{open} are high, low, and open prices in the i^{th} trading interval.

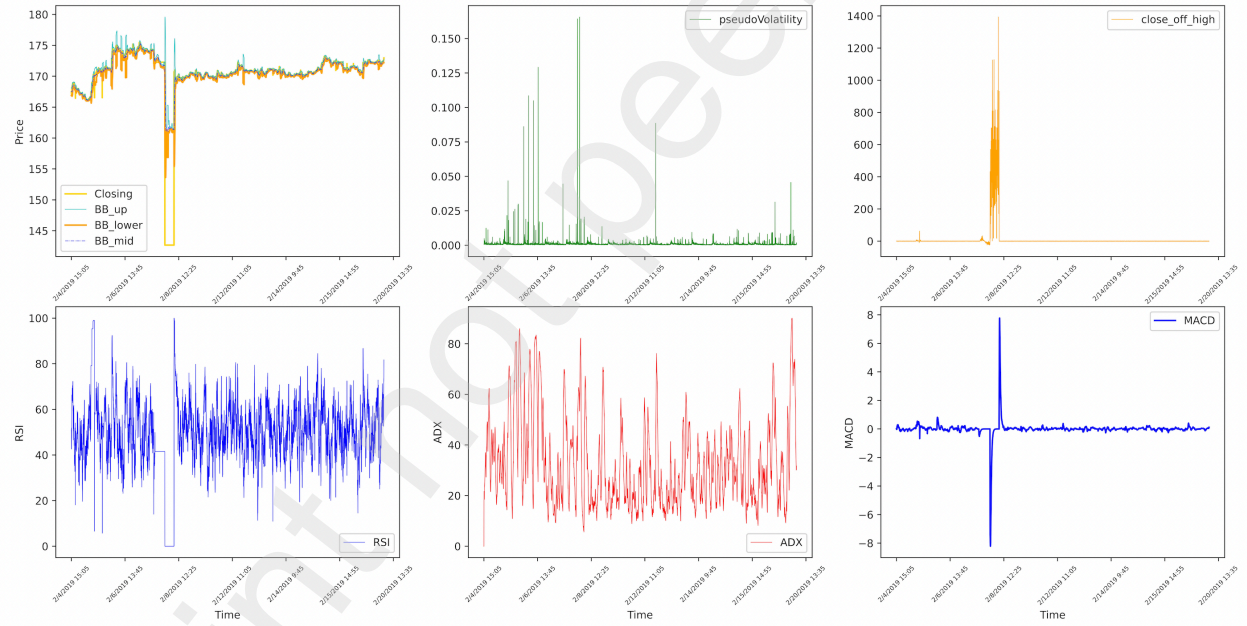

Fig 6. shows six interpolated variables added to the AAPL HFT data, providing valuable insights into market trends, volatility, and momentum. BB, MACD, COH, RSI, and pseudo-volatility are particularly useful for identifying extreme price movements and potential buy/sell opportunities.

Figure 6 illustrates the six interpolated variables: BB, COH, pseudo-volatility, RSI, ADX, and MACD, which we added to the AAPL HFT data from 2019/02/01 to 2019 02/ 22. The interpolated variables describe trading from different perspectives, but almost all demonstrate strong sensitivity to the extreme price change period where the price drops abruptly in trading. For example, RSI drops to <10 and indicates the security is greatly undervalued; COH increases to a huge value and pseudo-volatility demonstrates a large value.

Correspondingly, ADX moves to <20 and MACD shows a strong buy signal during the period. The interpolated variables provide a detailed understanding of HFT data and offer insights into market trends, volatility, and momentum in trading.

Figure 7 presents a comparison of PCA, t-SNE, and UMAP embeddings of the AAPL HFT data before and after adding interpolated features [17,23,25]. The results indicate that the feature interpolations contribute to uncovering more hidden trading dynamics. Notably, the data embeddings of PCA, t-SNE, and UMAP under the new feature space exhibit significant advantages over their counterparts in the original feature space.

For instance, after the feature interpolation, the t-SNE embedding can separate 'up' and 'down' transactions better than the original t-SNE embedding, which had 'up' transactions buried among the 'down' ones. Similarly, the PCA embedding under the feature interpolation provides a much better separation for 'up' and 'down' transactions than the original features, where most 'up' and 'down' transactions are wired together in the embedding except for a few outliers.

Moreover, the t-SNE embedding shows more 'explainable advantages' than the UMAP and PCA embeddings in separating the 'up' and 'down' groups [23]. It is worth noting that all other HFT datasets share similar characteristics (see the supplement). Thus, adding interpolated features to augment the original HFT feature space contributes to discovering more interpretable trading dynamics.

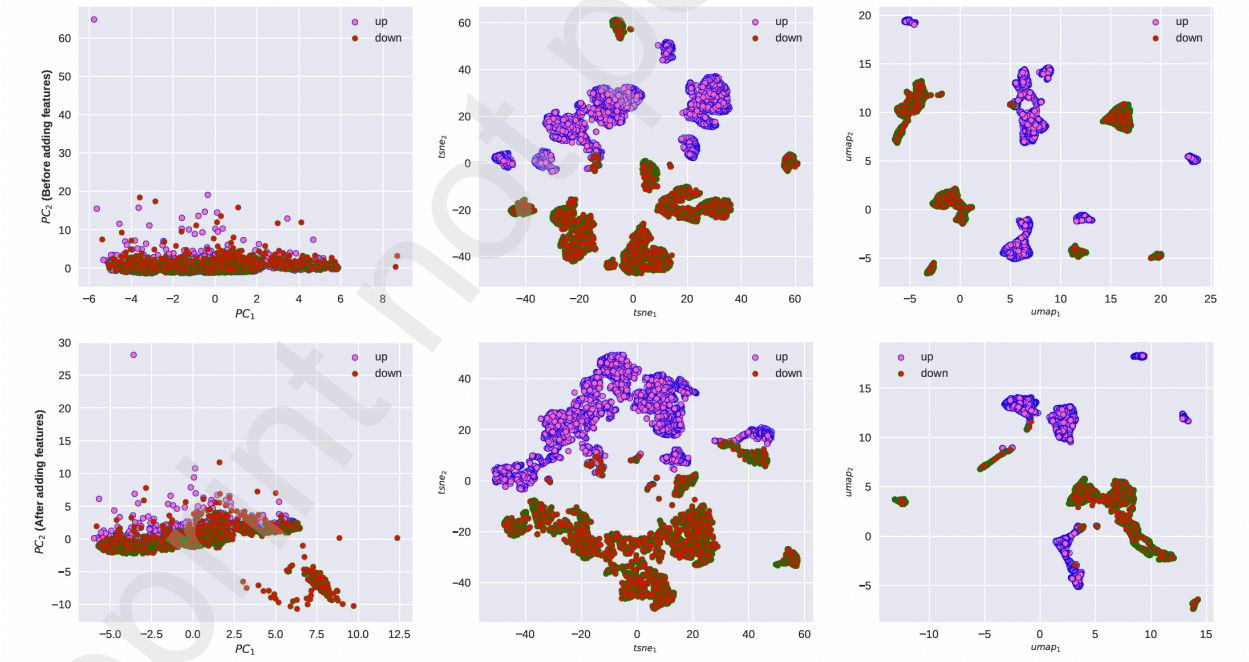

Fig 7. compares the PCA, t-SNE, and UMAP embeddings of AAPL HFT data before and after feature interpolations. The embeddings obtained after feature interpolations demonstrate a more informative representation than the previous ones before feature interpolations. Notably, the t-SNE embedding after feature interpolations exhibits more explainable advantages than the UMAP and PCA embeddings.

4.3 Explainable dimension reduction

t-SNE has the capability to distinguish various stocks' trading behaviors in the embedding space by creating an interpretable trading embedding. While we provide a brief introduction to t-SNE, please refer to [23] for a more detailed discussion of PCA, UMAP, and their explainability [23].

t-SNE calcualtes the low-dimensional embedding of input data $X = [x_1, x_2 \cdots x_n], x_i \in \Re^{p+l}$ by minimizing the Kullback-Leibler (K-L) divergence between a Gaussian distribution $P = (p_{ij})$ in the input space and a normalized Student's t-distribution $Q = (q_{ij})$ in the embedding space:

$$\varphi(P, Q) = KL(P \parallel Q) = \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} p_{ij} \, log \frac{p_{ij}}{q_{ij}} \tag{8}$$

where p_{ij} models the pairwise similarity between points x_i and x_j of the original data and q_{ij} models the pairwise similarity of their corresponding embeddings: y_i and y_j in the embedding space [27-28]. The non-convex objective function is minimized along the negative direction of the gradient: $y_{j+1} = y_j - r_j \frac{\partial \varphi}{\partial y_j}$, where $\frac{\partial \varphi}{\partial y_j} = 4 \sum_{j \neq i}^{n} (p_{ij} - q_{ij}) q_{ij} Z (y_i - y_j)$, $Z = \sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}$, and r_j is the learning rate. The average compelxity of t-SNE $O(nlogn)$ also prepares it well for real-time processing [29-30].

4.4 Trading marker finding clustering

As previously stated, we accomplish 'trading marker finding clustering' by interpreting DBSCAN. This enables us to identify trading markers, label the status of each transaction, and simultaneously discover transaction clusters. We briefly introduce DBSCAN as follows.

4.4.1 DBSCAN

DBSCAN is a density-based clustering algorithm that handles arbitrary-shaped data with noise. DBSCAN classifies points as core, reachable, and outliers (also called noise) in clustering [18-19]. DBSCAN clustering is equivalent to creating a graph with core and reachable points as vertices and edges connecting them but leaving the outliers unconnected. Unlike other clustering algorithms, it doesn't require the number of clusters to be specified in advance, making it particularly useful for datasets with unknown cluster structure or variable cluster sizes.

DBSCAN is a clustering algorithm that groups data points based on their proximity and density. To cluster a given point, which is the corresponding trading embedding of a transaction under t-SNE, DBSCAN first examines its ε-neighborhood, which is the set of all points within a specified distance from the point. If the size of the ε-neighborhood is greater than or equal to the minimum number of points (*minpts*) required to form a dense region, the algorithm initializes a new cluster with the neighbors and marks the original point as a core point. Otherwise, the point is marked as an outlier.

If a point is part of a cluster, its ε-neighborhood will be marked as a part of that cluster, and all points in the neighborhood will be added to the cluster until the density condition is satisfied. These points are called reachable points, as they are reachable from a core point. This process continues until all reachable points

have been assigned to a cluster, and all remaining points that do not belong to any cluster are marked as outliers. The average running time complexity of DBSCAN is $O(nlogn)$ though the worse time complexity is $O(n^2)$.

4.4.2 Implement 'trading marker finding clustering'

The 'trading marker finding clustering' can be implemented in two ways. The first approach involves collecting as many outliers as possible from the DBSCAN clustering and using trading marker thresholds (e.g., a price change threshold of $\eta > 0.2\%$) to identify and collect trading markers. To obtain more outliers, a small radius ε value (e.g., ε=0.5) is required given a *minpts* value. However, this approach is more conservative and computationally complex.

The second approach, which we employ in this study, involves selecting a relatively large radius value (e.g., ε=2) for a given *minpts* value to obtain a smaller set of outliers and treating them as trading markers. This approach is more efficient in capturing important trading markers by filtering out the trivial ones. We find that almost all outliers can be trading markers according to different standards. It is advisable to use a medium value of *minpts* (e.g., 30) in the DBSCAN clustering algorithm. Using a very high *minpts* value can result in grouping together many unrelated points, leading to fewer clusters and a loss of locality preservation. Conversely, a very low *minpts* value may result in the creation of spurious clusters containing outliers. We prove that we can seek trading markers from the outliers from the DBSCAN clustering on the embedding of input data provided the parameters ε and $minpts$ are carefully selected.

Theorem 5 Given the embedding $E = \{e_i\}_{i=1}^n$, $e_i \in \Re^k$ of an HFT dataset with n observations and p variables: $X = \{x_i\}_{i=1}^n, x_i \in \Re^p$, $k < p \ll n$, under an explainable dimesnion reduction algorithm f_d, which is t-SNE in our context, there exists DBSCAN with parameters ε and $minpts$ applied to E such that each outlier is identified as a trading marker.

Proof. Given HFT data $X = \{x_i\}_{i=1}^n$ in the trading time period $[t_0, t]$, we assume that price function $p(t)$ and volume function $v(t)$ of the trading data are known. We then define the trading marker set for each 1-minute trading interval $[t_k, t_{k+1}]$ based on a price change ratio threshold η and a volume cutoff δ, as follows: Let *B(x, η, δ, k)* denote the trading marker set for the *k*-th 1-minute interval, where *x* is a point in the set X. The trading marker set *B(x, η, δ, k)* is defined as:

$$B(x, \eta, \delta, k) = \left\{x: \frac{|p(t_{k+1}) - p(t_k)|}{|p(t_k)|} \geq \eta, v(t_{k+1}) \geq \delta\right\} \quad (9)$$

Here, |B(x, η, δ, k)| is either 1 or 0 depending on whether x belongs to the trading marker set or not. We can define the trading marker set N(x, η) as the union of the trading marker sets for each 1-minute interval, i.e.,$N(x, \eta) = \cup_{k=1}^n B(x, \eta, \delta, k)$.

The image of the trading marker set denoted by $f_d(N(x, n))$ is a subset of E. However, the crowding issue of t-SNE can cause several points in the input space to be mapped to a single point in the embedding space, i.e., $|f_d(N(x, n)| \leq |N(x, \eta)|$ [27]. The crowding issue in t-SNE refers to the phenomenon where the distance between the points in the embedding space *E* is not preserved accurately from the input space X. As a result,

some points in the input space may be clustered together in the embedding space, leading to a reduction in the number of distinct points in $f_d(N(x,n)$.

On the other hand, it is noted that the $f_d(N(x,n))$ will be seperated from $\{E - f_d(N(x,n)\}$ spatially in the embedding space because the good explainaility of t-SNE and the nature of trading markers. To identify the trading markers in the embedding space E, we can use the DBSCAN algorithm. We define minpts as $minpts = n - |N(x,\eta)|$. We also define $\varepsilon = max \cup_{i,j}^{n} \frac{\|e_i - e_j\|}{2}, e_i, e_j \in \{E - f_d(N(x,n)\}$. Using these parameters, we can run DBSCAN clustering on the embedding $E = \{e_i\}_{i=1}^{n}$. In this clustering, all points in the set $E - f_d(N(x,n)$ will be classified as core points and reachable points in one cluster, while all points in $f_d(N(x,n)$ will be classified as outliers. Thus, each outlier will be identified as a trading marker.

It is noted that we label each trading interval as 'up' or down' before clustering according to the difference between the open price and close price in the interval. This approach not only imbues more detailed significance to trading markers but also aids in generating more informative trading maps.

4.4.3 Collision detection. Each trading map is created by DBSCAN clustering the t-SNE embedding of HFT data. However, due to the high density of points in the embedding space, it is possible for multiple common or inflection transactions to be mapped onto the same point, resulting in collisions on the trading map. To address this issue, collision detection must be implemented.

The collision detection process involves calculating pairwise distances of the t-SNE embedding points for a given set of trading markers. A threshold, denoted as β_d, is set as the minimum distance allowed between two points in clustering. If the embedding points of two trading markers, A and B, denoted as a=$f_d(A)$, b=$f_d(B)$ respectively, are such that their distance is less than β_d, i.e., dist(a, b) $< \beta_d$, then one of the markers is automatically 'hidden' from the trading map due to the detected collision. It is worth noting that outliers are less likely to be affected by collision detection compared to other data points.

4.4.4 The explainable trading dynamics discovery algorithm FIDR-SCAN

Algorithm 1: Feature-Interpolation Dimension-Reduction-based SCANing (FIDR-SCAN)

Input:

HFT data: $X \in \mathbb{R}^{n \times p}$ with n observations across p features, $n \gg p$
Explainable dimension-reduction model: f_d (default t-SNE)
$minpts$: the minimum number of points (default 30)
ε: the neighbor radius in clustering (default 2)

Output:

Trading map X_{tmap}

// mark 'up/down' labels for data according to price change in each interval

1. $X \leftarrow Markupdownlabel(X)$

// do feature interpolations to augment the feature space

2. $X_{newFeatures} \leftarrow featureInterpolation(X)$

3. $X \leftarrow [X, X_{newFeatures}]$

//Explainable dimension reduction to produce the explainable trading embedding

4. $X_{embeding} \leftarrow f_d(X)$

5. *//DBSCAN clustering for the trading embedding*
6. $ClusteringIndex,\ transactionStatus, markers\ \leftarrow DBSCAN(X_{embedding}, minpts, \varepsilon)$

// Trading map generation

7. $X_{tmap} \leftarrow GenerateTradingMap(X_{embedding}, ClusteringIndex, transactionStatus, markers)$

8. **Return** X_{tmap}

Algorithm 1 presents the FIDR-SCAN algorithm, which is proposed for discovering explainable trading dynamics. The average complexity of FIDR-SCAN is $O(n + nlogn + l^2)$, which is the result of the $O(n)$ feature interpolation complexity, $O(nlogn)$ average complexities for t-SNE and DBSCAN, and $O(l^2)$ pairwise distance calculation complexity for collision detection. Marking the trading map has an additional complexity of $O(n)$, which is negligible as the number of trading markers is much smaller than the total number of transactions. Therefore, FIDR-SCAN can efficiently create a trading map for a large input dataset in real-time. The worst-case complexity would be $O(n^2)$ due to the $O(n^2)$ time complexity of DBSCAN.

5 Trading map validation and analysis

It is reasonable to validate the meaningfulness of the trading maps generated by FIDR-SCAN by investigating core, reachable, and outliers, which are interpreted as normal transactions, inflection transactions, and trading markers. It will help to answer important queries like 'Are they meaningful and effective classifications of transactions in trading dynamics unveiling?'

Proposition 1. *The probability density functions of normal transactions, inflection transactions, and trading markers identified from FIDR-SCAN exhibit distinguishable characteristics in their respective variables.*

5.1 Probability density function analysis

Our probability density function analysis indicates that the three groups of points are not only meaningful but also demonstrate good interpretations in terms of the interpolated features indicates. Figure 8 compares the probability density functions (*p.d.f.s*) of RSI, ADX, log-return, and pseudo-volatility of the three groups in the AAPL trading map. The three groups demonstrate remarkably different distributions. The trading marker group shows obviously higher means and larger standard deviations of RSI and ADX than normal and inflection transactions. Similarly, it shows exceptional log-return and pseudo-volatility values compared to those of the other two.

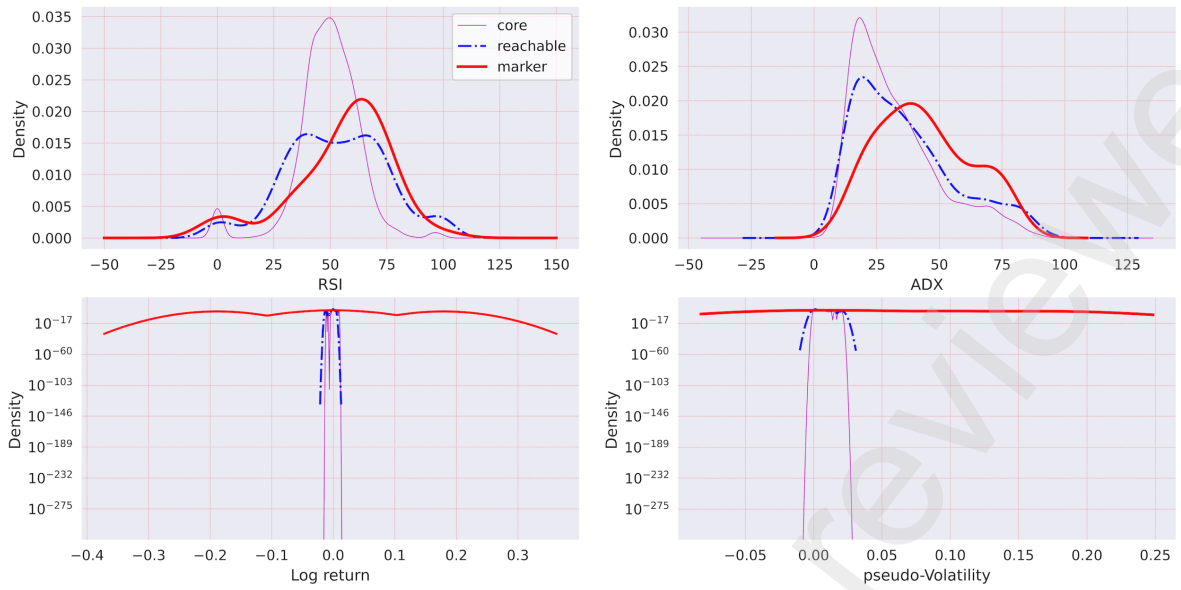

Fig 8. The probability density functions (*p.d.f.s*) of RSI, ADX, log-return, and pseudo-volatility for the core points, reachable points, and trading markers in the AAPL trading map. The distinctive characteristics of the trading markers are evident, with notably higher means and larger standard deviations of RSI and ADX, exceptional log-return and pseudo-volatility values compared to the core and reachable points.

In addition to the variables analyzed in Figure 8, we also examined the probability density functions of several other variables to further investigate the distinctions between the three groups. The results support the notion that trading markers are different from normal and inflection transactions. For example, the *p.d.f.* of the number of trades indicates that trading markers and reachable points have more trades than core points. The *p.d.f.s* of BB-mid, close-off-high (COH), and MACD also suggest that trading markers have different characteristics than the other two groups. Specifically, trading markers have smaller COH ranges, indicating they are potential extreme values in trading. They also have smaller MACD ranges, suggesting they provide stronger buy/sell signals. Please refer to the supplemental materials for more details on the remaining *p.d.f.s*. These findings provide further evidence of the meaningfulness and effectiveness of the trading maps in identifying important patterns in trading dynamics.

5.2. Examining trading markers

Proposition 2. *Assuming a known variable function $f(x)$, if $x^* \in X_m$, the trading marker set identified from FIDR-SCAN during a trading period, then $f(x^*)$ represents a local or global optimum during that period.*

We have observed that the trading markers (outliers) manifest themselves as either the 'maximum/minimum' or 'local maximum/minimum' for various variables other than the price variable. Specifically, for the identified trading markers, the log-return and pseudo-volatility values are either maximum or minimum, while the volume, market notational, and other variable values of these markers are local maximum or minimum. This suggests that the proposed algorithm for trading marker discovery is effective, as it is able to identify outliers in multiple variables beyond just the price variable. By detecting

maximum or minimum values in variables such as log-return and pseudo-volatility, as well as local maximum or minimum values in variables like volume and market notational, the algorithm is able to pinpoint potential trading markers that may not have been identified using traditional methods. We do the following case study for BAC (bank of america) HFT data to demonstrate it.

BAC trading map analysis. Trading maps can distinguish different securities from their individually varied trading maps reflecting their own trading dynamics. It can be hard to distinguish the price plots of two stocks sometimes, but it is easy to detect their trading maps because the latter reflects more trading essentials in a more informative and larger feature space.

Figure 9 displays the trading map of BAC HFT data from 2019/02/01 to 2019/02/22, which has been clustered into 23 groups using FIDR-SCAN with 41 trading markers identified based on the price change ratio η≥0.2%. The trading map suggests that BAC is likely to experience an 'up' market, as the prices of trading markers in the later trading period are higher than those in the early period. Moreover, there are no significantly large changes in trading marker prices, indicating that BAC will not be as volatile as AAPL in trading.

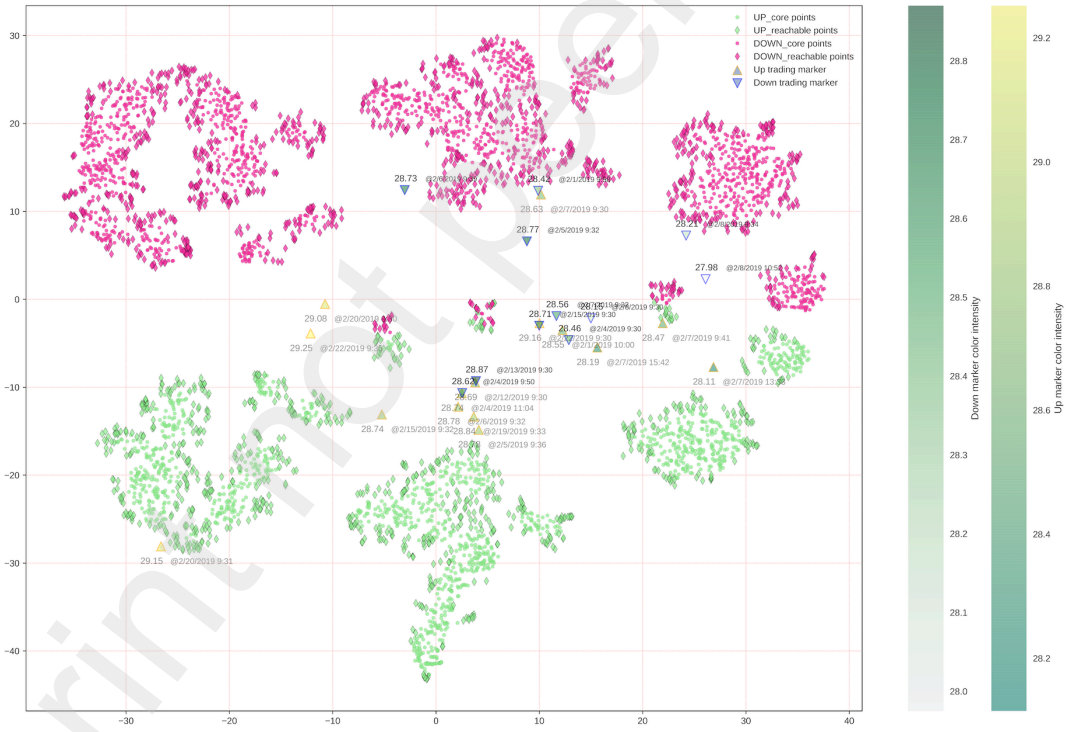

Fig 9. The trading map of BAC HFT data from 2019/02/01 to 2019 02/22 under FIDR-SCAN with 41 trading markers with the price change ratio of $\eta \geq 0.2\%$. The whole market is an 'up' market because the prices of later trading markers are higher than those of the previous ones. Notably, the trading map makes it easy to identify important markers located at \$28.77 and \$29.25, due to their significant distances from their neighboring clusters.

Figure 10 validates the effectiveness of the identified trading markers from the trading map across different variables, such as close price, price change ratio, log-return, and notional values. The markers represent global or local optima for these variables, as illustrated in the subplots. For instance, the price plot shows the markers as meaningful peaks/bottoms or local 'bottoms/peaks', while the change ratios plot shows them as local maximums/minimums. The log-return plot illustrates max/min log returns per trading interval, and the notional value plot indicates local maximum values. These findings support the proposition 2 that a trading marker appears at a spot where the corresponding value achieves a local or global optimal value. Similar results are found in the supplemental for other datasets.

Fig 10. presents visualizations of BAC trading markers from FIDR-SCAN, showcasing values of variables such as close price, change ratio, log return, and notional value. These markers achieve either global or local optima on the variables.

5.3 Parameter tuning.

In addition to interpolated features, FIDR-SCAN utilizes three key parameters, namely perplexity of t-SNE, neighbor radius ε, and *minpts*, to create a trading map for HFT data. Proper configuration or tuning of these parameters is crucial to ensure desirable generation of the trading map.

The perplexity of t-SNE controls neighborhood size and level of global/local data behavior presentation in the explainable dimensional reduction embedding. It's defined by 2 to the power of Shannon entropy: $H(P) = 2^{-\sum_{i \neq j} p_{ij} \log_2 p_{ij}}$, in which p_{ij} is the joint probability to model the similarity between the i^{th} and j^{th} transactions [28]. Small values lead to more local behavior presentation, while large values lead to more global behavior. Empirically, values between 100-200 balance global and local data behavior presentation.

The neighborhood radius ε, which is the minimum distance between two embedding points in explainable dimension reduction, affects the clustering sensitively. A too small ε (e.g., $\varepsilon = 0.5$) will produce many more trading markers in FIDR-SCAN so that the importance of key markers may become less obvious, while a too large one (e.g., $\varepsilon = 5$) will greatly decrease the number of the outliers as well as the number of clusters. Therefore, it is recommended to set ε somewhere between 1.5 and 2 for the sake of producing appropriate clustering.

The *minpts* parameter is the minimum number of entries in a neighborhood for a transaction to be considered as a common transaction in FIDR-SCAN. A small value generates more outliers, while a large one misses important markers. It is recommended to set *minpts* to at least 30 for highly traded large caps (e.g., AAPL). Grid search is not recommended due to the slow real-time implementation.

6 AI trading: reuse trading markers

We validate the effectiveness of identified trading markers by designing profitable trading algorithms. This effort aims to answer traders' queries about the markers' profitability in a chosen trading period. More importantly, it is equivalent to reusing the identified markers within a future similar market via an 'AI trading'. We evaluate the earning percentage of a trading algorithm in a trading period $[t_s, t_e]$ to assess the efficiency of using trading markers in trading. The earning percentage is the ratio of the difference between the trading account balances at the start and end times d_{t_s} and d_{t_e} and to the initial balance:

$$earning_p = \frac{d_{t_e} - d_{t_s}}{d_{t_s}} \tag{10}$$

The baseline (default) earning percentage is obtained by doing nothing and just following the market. We compare the earning percentage of each stock using identified markers with the baseline earning percentage to evaluate their effectiveness.

6.1 Market trend probing.

We validate/reuse our trading markers by creating profitable trading schemes based on the detected markers in real-time. Our approach assumes no prior knowledge of the market other than the sequence of trading markers. To achieve this, we classify market trends as 'flat', 'up', or 'down' and use the identified markers to probe the market tendency, starting from the beginning of the trading period.

Flat, up, and down markets. Our trading approach involves categorizing the market trend as either 'flat', 'up', or 'down'. A 'flat' market has an equal mean price for the 'up' and 'down' intervals, while an 'up' or 'down' market has a higher or lower mean price for the 'up' intervals, respectively, compared to a threshold. Based on the market trend, we use the identified markers to adaptively select buy, sell, or hold actions in trading, and calculate the earning percentage accordingly.

To estimate the market trend for HFT data, probing only the first N_t trading time intervals (e.g., $N_t = 1000$) may result in an inaccurate estimation. Trading markers can provide important sampling points that, when combined with historical prices, can more accurately estimate the market trading tendency.

To estimate the market trend, we compare the means of the 'up' and 'down' marker prices using the formula: $\Delta p_{\Delta t} = (\frac{1}{l_u}\sum_i^{l_u} p_i^{(u)} - \frac{1}{l_w}\sum_j^{l_d} p_j^{(d)})$, where $\frac{1}{l_u}\sum_i^{l_u} p_i^{(u)}$ and $\frac{1}{l_d}\sum_j^{l_d} p_j^{(d)}$. Here, $p_i^{(u)}$, $p_j^{(d)}$ are the price of the ith 'up' marker and jth 'down' marker, and l_u, l_d are the numbers of 'up', 'down' markers respectively. If $\Delta p_{\Delta t}$ is greater than a set threshold $\delta_p > 0$, the market is considered an 'up' market. Conversely, if $\Delta p_{\Delta t}$ is less than the negative of δ_p, the market is considered a 'down' market. Otherwise, it is a 'flat' market. The threshold is set as $\delta_p = \zeta(\sum_i^{l_u} p_i^{(u)} + \sum_j^{l_d} p_j^{(d)})/l$ where ζ is a cutoff ratio (e.g., 0.15%) and $l = l_u + l_d$.

In trading, the initial amount of money is assigned randomly, and each buy/sell uses 50% of the available funds in the account. Algorithm 2 is a proposed trading algorithm that uses trading markers to validate the effectiveness of the FIDR-SCAN trading marker discovery method. The algorithm's complexity is approximately *O(n),* assuming that the number of trading markers is significantly less than *n*.

Algorithm 2: Trading with trading markers

Input:

HFT data: $X \in \mathbb{R}^{n\times m}$ with *n* observations across *m* features, *n>>m*
Trading period $[t_s, t_e]$
Trading marker list $M = [M_1, M_2, \cdots M_l]$
Initial trading balance d_{t_s}
Cutoff ratio ζ (default: 0.15%)
Cutoff threshold ε (default: 0.05%)

Output:

$earning_p$

1. // align trading markers to each trading interval
2. ***for*** each i^{th} trading interval $t_i \subset [t_s, t_e]$
3. ***if*** $M_i.time \cap t_i == \emptyset$ // no marker found in t_i
4. $M_i.price \leftarrow \infty$
5. // determine the potential market trend by probing
6. $marketTrend \leftarrow probingMarketTrend(M, \zeta)$

7. ***if*** *marketTrend* is up
8. ***for*** each i^{th} trading interval $t_i \subset [t_s, t_e]$
   // if there exists an 'up' tendency locally
   if $(i-1)^{th}$ trading interval t_{i-1} is an 'up' interval and If curent price $\boldsymbol{p_i} > M_{i-1}.price$

$d_{t_s}.buy()$
// current price has a marginal change with respect to the marker
else if $(M_i.price - \boldsymbol{p_i})/\, M_i.price < \varepsilon$
$d_{t_s}.hold()$
// current price has an acceptable change
else
$d_{t_s}.sell()$

9. ***if*** *marketTrend* is down
10. ***for*** each i^{th} trading interval $t_i \subset [t_s, t_e]$
if $M_{i-1}.price > M_i.price$
Sell stocks bought before and buy stocks
$d_{t_s}.update()$
else
Sell stocks bought before and short stocks
$d_{t_s}.update()$
11. ***if*** *marketTrend* is flat
if $M_{i-1}.price > M_i.price$
$d_{t_s}.buy()$
else
$d_{t_s}.buy()$
12. $earning_p \leftarrow d_{t_s}.calculateEarningPercentage(\ )$
13. ***Return*** $earning_p$

Table 2 compares the earning percentages of the four stocks using two methods. The results show that using trading markers can significantly increase the earning percentages, particularly in a 'down' market. For instance, in AEO trading, the earning percentage with reusing markers can reach 2.34%, while the default is -4.54%. This demonstrates that using the identified trading markers can improve trading profitability and prevent losses. Furthermore, even in an 'up' market like WMT, using trading markers can enhance the earning percentage to 7.28%. These findings suggest that the detected trading markers are profitable and that the proposed FIDR-SCAN algorithm is effective in discovering trading dynamics from a profit trading perspective. The corresponding plots of the trading account balance dynamics can be found in the supplemental materials.

Table 2. The comparisons of earning percentages with/o trading markers

	AAPL	BAC	AEO	WMT
Baseline $earining_p$	3.29%	1.80%	-4.54%	5.98%
$earining_p$	8.23%	3.48%	2.34%	7.28%

7 Discussion

This study proposes an explainable ML algorithm FIDR-SCAN to discover trading dynamics by creating a trading map for HFT data. It deciphers trading dynamics by identifying meaningful trading markers and categorizing transactions upon a new feature space. It provides a novel way to find significant trading makers, distinguish different securities' trading behaviors in HFT, and design more profitable trading algorithms. Since FIDR-SCAN has relatively low $O(nlogn)$ complexity, it makes it possible for a trader to

generate a sequence of trading maps for his interested security to exploit the deciphered trading dynamics in real-time trading.

FIDR-SCAN's interpretability is attributed to its meaningful feature interpolations, explainable dimensional reduction, interpretable transaction status classification, and significant trading structure discovery. The interpolated variables increase HFT data entropy, leading to a more informative and interpretable feature space for discovering trading dynamics. The explainable dimensional reduction produces an interpretable trading embedding that captures HFT data's intrinsic structure, providing a solid foundation for discovering trading markers, trading status, and trading structure through DBSCAN interpretation from a trading perspective. Theoretically, we have proven that the outliers from DBSCAN clustering are trading markers if the parameters ε and *minpts* are carefully selected. All of these elements contribute to the discovered trading dynamics' interpretability.

Cryptocurrency data extension. The FIDR-SCAN algorithm can be extended to analyze cryptocurrency data and uncover its underlying trading dynamics on the blockchain. While cryptocurrency data shares some similarities with HFT data, it usually has more features and can appear more volatile. To demonstrate the applicability of FIDR-SCAN, we applied it to 5-minute aggregated ETH data from 08/02/2018 to 08/19/2018. Our results show that incorporating interpolated variables into PCA, t-SNE, and UMAP visualizations can provide more informative insights than using the original features alone. By using Algorithm 2 and trading markers, we achieved earning percentages as high as 19.85%, in contrast to the default earning percentage of -29.42% (see supplemental materials). However, a more customized trading dynamics algorithm may be desirable for cryptocurrency data, as the current feature interpolations are based on HFT data of stocks and more representative features may be required to create a more explainable feature space for cryptocurrency trading. [31]

Trading machine construction. Moreover, the trading markers detected by FIDR-SCAN can be utilized in constructing trading machines. These markers, along with other data points, can be labeled as 'buy' if their prices are $\geq p\%$ of the current price in the trading interval, and 'sell' if their prices are $\leq p\%$ of the current price in the trading interval. Transactions that do not meet these criteria are labeled as 'hold,' as they are less likely to be trading markers and are more likely to be common transactions. The value of $p\%$ can be determined based on specific trading needs, but we suggest setting it between 0.5% and 3.5% for more meaningful trading. Then, the first 80% of labeled data can be used as training data to predict the remaining data.

We used AAPL data to construct trading machines employing a five-layer convolution neural network (CNN), a five-layer deep neural network (DNN), and extremely randomized trees (ET) with 100 decision trees [32-34]. The labeled data is imbalanced, with the 'hold' class accounting for 90.94%. Despite this, the CNN, DNN, and ET models achieve 90.76%, 93.50%, and 91.37% accuracy values, respectively, under the original raw data without resampling. The ET model performs slightly better than the two deep learning models because its forecasting is less 'hijacked' by the majority ('hold') class, as shown by the comparisons of their confusion matrices in Figure 11. However, since the minority classes ('buy' and 'sell') mainly consist

of trading markers, the ET model achieves the best performance under the random-over-sampling (ROS) data, with a 99.57% accuracy among the three [35]. This suggests the effectiveness of using trading markers in building trading machines. Conversely, the deep learning models exhibit poor performance, with only 48.24% and 64.58% accuracy values under the ROS data, as the majority class still 'hijacks' the learning process by classifying the minority classes as the majority class [36].

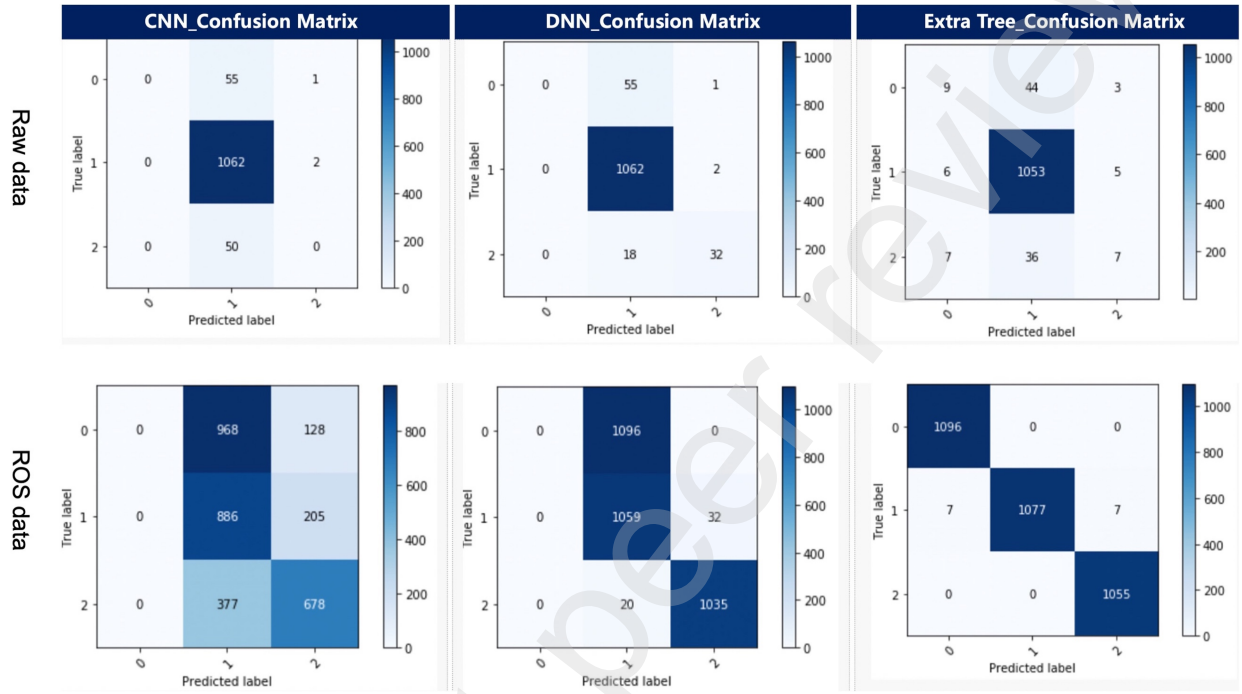

Fig 11. The confusion matrices of the CNN, DNN, and Extra tree models in predicting trading statuses of AAPL data under raw data (w/o resampling) and random-over-sampling (ROS) data. The ROS data under the Extra model achieve the best performance because ROS brings more trading markers in learning.

8 Conclusion

Our proposed FIDR-SCAN algorithm offers an innovative and explainable machine learning approach for creating trading maps from HFT data, enabling the discovery of trading dynamics. By leveraging meaningful feature interpolations and interpretable dimensional reduction, FIDR-SCAN provides a solid foundation for identifying trading markers and conveying the statuses of transactions in an informative 'interpolated' feature space. These trading markers can then be reused in the design of trading algorithms, unlocking new opportunities for profitable AI trading. Moreover, FIDR-SCAN's marker detection capabilities offer a more efficient and explainable way to discover trading markers from HFT data. Furthermore, FIDR-SCAN's ability to construct trading maps opens up exciting possibilities for generating trading signatures from both HFT and cryptocurrency data. In addition, the trading markers identified by

FIDR-SCAN can be leveraged to build trading machines, opening a new avenue for AI trading. Besides, we provide rigorous quantification analysis for HFT data and proposed feature interpolation techniques.

While FIDR-SCAN has proven effective in discovering trading dynamics, there are still limitations to consider. For example, we don't know which features are optimal for this task despite the confirmed effectiveness of our proposed features. The role of entropy in the discovery of trading markers is also unclear, although we do know that it increases trading behavior entropy and leads to more outliers. It can be interesting to investigate the feature selection problem using evolutionary computing techniques [37-38]

While the average complexity of FIDR-SCAN is good for creating real-time trading maps, it may face the risk of running into worst-case complexity for some large datasets. To address these issues, we are exploring evolutionary computation techniques to search for optimal features and investigating big data analytics approaches with more efficient DBSCAN variants (e.g., HDBSCAN) [39-40]. Additionally, we are researching how to use the markers identified by FIDR-SCAN to build more effective and explainable trading machines using state-of-the-art learning methods such as deep reinforcement learning [41-42]. These efforts will enable us to further improve our understanding of trading dynamics and develop better tools for trading in high frequency environments.

In addition, it should be noted that the timestamps are not directly utilized in the clustering process of the proposed algorithm for discovering explainable trading dynamics. It would be intriguing to incorporate time information into the algorithm to explore more insightful clusters in the creation of trading maps. However, it is still uncertain how much the time variable can contribute in conjunction with other interpolated variables. Further research is needed to investigate this potential enhancement.

On the other hand, we are interested in extending the existing algorithms to cryptocurrency data though the existing algorithm seems to have worked well for individual datasets of cryptocurrency. Technically, cryptocurrency data has a slightly larger feature space than HFT data but contains more redundant information and are more volatile in trading [42]. Thus, how to construct an explainable feature space for cryptocurrency data from an interpretable learning perspective will be an interesting topic for future studies [43-44]. Besides exploring the trading map differences between different securities to expose their individualized trading microstructures on the blockchain, it would also be interesting to investigate how to reuse discovered markers more efficiently in future trading by designing more profitable AI trading algorithms for cryptocurrency data [45].

Acknowledgements

This work is partially supported by NASA Grant 80NSSC22K1015, NSF 2229138, and McCollum endowed chair startup fund.

Reference

1. Cespa, G. and Vives, X. 2017. High frequency trading and fragility. Working Papers Series. European Central Bank, 2017
2. Aldridge, I., Krawciw, S., 2017. Real-Time Risk: What Investors Should Know About Fintech, High-Frequency Trading and Flash Crashes. Wiley. ISBN 978-1-119-31896
3. Hendershott, et al.: Does Algorithmic Trading Improve Liquidity? *Journal of Finance*, 2011
4. Matteo, A, Budish, E. O'Neill, P, Quantifying the High-Frequency Trading 'Arms Race': A Simple New Methodology and Estimates (June 25, 2020). SSRN: https://ssrn.com/abstract=3636323
5. Han, H and Li, M: Big data analytics for high frequency trading volatility, *Springer Proceedings of Business and Economics*, Springer, 2018, 352-401
6. Han, H, Teng, H, Xia, L, Wang, Y, Guo, Z, Li, D: Predict High-Frequency Trading Marker via Manifold Learning, *Knowledge-based system*, 213:106662, 2021
7. Aït-Sahalia, Y., & Xu, D. (2019). Principal Component Analysis of High-Frequency Data. Journal of the American Statistical Association, 114(525), 287–303
8. Menkveld,K A: High-Frequency Trading around Large Institutional Orders, Journal of Finance, 2019
9. Brogaard, Jonathan, Terrence Hendershott, and Ryan Riordan, 2014, High-frequency trading and price discovery, *Review of Financial Studies* 27, 2267–2306
10. Baron, M, Brogaard,J, Hagströmer, B, and Kirilenko, A : Risk and Return in High-Frequency Trading." Journal of Financial and Quantitative Analysis,54(3): 993–1024, 2019
11. Conrad, Jennifer, Sunil Wahal, and Jin Xiang, 2015, High-frequency quoting, trading, and the efficiency of prices, *Journal of Financial Economics*116, 271–291
12. Manahov, V, Zhang, H (2019) Forecasting Financial Markets Using High-Frequency Trading Data: Examination with Strongly Typed Genetic Programming, International Journal of Electronic Commerce
13. Fischer, T., Krauss, C, Deep learning with long short-term memory networks for financial market prediction, European journal of Operational Research 2018 654-669
14. Fang, B, Feng, Y (2019) Design of High-Frequency Trading Algorithm Based on Machine Learning, arXiv:1912.10343
15. Brogaaard et al (2018) High frequency trading and extreme price movement, Journal of Financial Economics, 128:2, Pages 253-265, 2018
16. Xu, J (2015), Optimal Strategies of High Frequency Traders. AFA 2015 Boston Meetings Paper, Available at SSRN: https://ssrn.com/abstract=2382378 or http://dx.doi.org/10.2139/ssrn.2382378
17. Becht et al (2019) UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, *Nature Biotech*, **37**: 38–44 (2019)
18. Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. ACM Transactions on Database Systems (TODS), 42(3), 19
19. Ester, et al A density-based algorithm for discovering clusters in large spatial databases with noise, KDD'96 Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 226-231

20. Achelis, S. B. (2000). Technical analysis from A to Z: covers every trading tool. McGraw-Hill.
21. IEX-API: https://iexcloud.io/docs/api/#intraday-prices retrieved 2019.
22. Sheather, S.J.; Jones, M.C. (1991). "A reliable data-based bandwidth selection method for kernel density estimation". *Journal of the Royal Statistical Society, Series B*. **53** (3): 683–690.
23. Han, H, *et al.* Enhance Explainability of Manifold Learning, *Neurocomputing* 500:877-895 (2022)
24. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2825-2830
25. Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *Science*, 315(5814), 972-976
26. Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2, 849-856
27. Der Maaten LV, Hinton GE: Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research* 2008:2579-2605.
28. Li, et al Application of t-SNE to human genetic data, *Journal of Bioinformatics and Computational Biology*, 15:04, 1750017, 2017
29. Laurens VDM: Barnes-Hut-SNE. *NIPS* 2013, 1301.3342.
30. Dmitry,K, Berens, P: The art of using t-SNE for single-cell transcriptomics, *Nature Communications* volume 10: 5416, 2019
31. Han, H, Liu, X, 2021 The challenges of explainable AI in biomedical data science. *BMC Bioinformatics* 22, 443
32. Liu, Q, Li, D, Han, H (2020) Manifold learning analysis for Allele-skewed DNA modification SNPs for psychiatric disorders, *IEEE Access*, 8:1, 33023-33038, 2020
33. Zhang, H, Huang, H, Han, H (2021): A Novel Heterogeneous Parallel Convolution Bi-LSTM for Speech Emotion Recognition, *Appl. Sci.* 11(21), 9897
34. Geurts, P, Ernst, D and Wehenkel, L: Extremely randomized trees. *Machine learning* 63.1 (2006): 3-42.
35. Han, H., Wu, Y., Ren, J., Diane, L. (2022). Forecasting Stock Excess Returns with SEC 8-K Filings. Communications in Computer and Information Science, vol 1725: 3-18
36. Han et al (2022), Interpretable Machine Learning Assessment, https://ssrn.com/abstract=4146556
37. Han, F, Chen W, Ling, Q, Han, H, Multi-objective particle swarm optimization with adaptive strategies for feature selection, *Swarm and Evolutionary Computation*, 62:10847, 2021
38. Oh, I, Lee, J, Moon, B (2014) Hybrid genetic algorithms for feature selection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1424 – 1437
39. Campello, R, Moulavi, D, Zimek, A, Sander, J (2015). "Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection". *ACM Transactions on Knowledge Discovery from Data*. 10 (1): 1–51
40. Moulavi *et al* (2014) Density-Based Clustering Validation Proceedings of the 2014 IEEE International Conference on Data Mining, Shenzhen, China, December 14-17, 2014, pp. 839-844.
41. Briola, et aal (2021) Deep Reinforcement Learning for Active High Frequency Trading, arXiv:2101.07107

42. Deng et al (2016), Deep direct reinforcement learning for financial signal representation and trading, *IEEE transactions on neural networks and learning systems*, 28:3, 653–664, 2016.
43. Kim YB, Kim JG, Kim W, Im JH, Kim TH, Kang SJ, et al. (2016) Predicting Fluctuations in Cryptocurrency Transactions Based on User Comments and Replies. *PLoS ONE* 11(8): e0161197.
44. Yermack, D. (2017). Corporate governance and blockchains. Review of Finance, 21(1), 7-31
45. Sezer et al (2019) Financial Time Series Forecasting with Deep Learning : A Systematic Literature Review: 2005-2019, arXiv:1911.13288