Quantitative Finance

ISSN: 1469-7688 (Print) 1469-7696 (Online) Journal homepage: https://www.tandfonline.com/loi/rquf20

Gaussian process-based algorithmic trading strategy identification

Steve Y. Yang, Qifeng Qiao, Peter A. Beling, William T. Scherer & Andrei A. Kirilenko

To cite this article: Steve Y. Yang, Qifeng Qiao, Peter A. Beling, William T. Scherer & Andrei A. Kirilenko (2015) Gaussian process-based algorithmic trading strategy identification, Quantitative Finance, 15:10, 1683-1703, DOI: 10.1080/14697688.2015.1011684

To link to this article: https://doi.org/10.1080/14697688.2015.1011684

Quantitative Finance, 2015
Vol. 15, No. 10, 1683–1703, http://dx.doi.org/10.1080/14697688.2015.1011684

Gaussian process-based algorithmic trading strategy identification

STEVE Y. YANG*†, QIFENG QIAO‡, PETER A. BELING‡,
WILLIAM T. SCHERER‡ and ANDREI A. KIRILENKO*‡§

†Financial Engineering Program, Stevens Institute of Technology, 1 Castle Point on Hudson, Hoboken, NJ 03070, USA
‡Department of Systems and Information Engineering, The University of Virginia, P.O. Box 400747, Charlottesville, VA 22904, USA
§MIT Sloan School of Management, 50 Memorial Drive, Cambridge, MA 02142, USA

(*Received 29 November 2012; accepted 8 January 2015*)

Many market participants now employ algorithmic trading, commonly defined as the use of computer algorithms, to automatically make certain trading decisions, submit orders and manage those orders after submission. Identifying and understanding the impact of algorithmic trading on financial markets has become a critical issue for market operators and regulators. Advanced data feeds and audit trail information from market operators now allow for the full observation of market participants' actions. A key question is the extent to which it is possible to understand and characterize the behaviour of individual participants from observations of trading actions. In this paper, we consider the basic problems of categorizing and recognizing traders (or, equivalently, trading algorithms) on the basis of observed limit orders. These problems are of interest to regulators engaged in strategy identification for the purposes of fraud detection and policy development. Methods have been suggested in the literature for describing trader behaviour using classification rules defined over a feature space consisting of summary trading statistics of volume and inventory, along with derived variables that reflect the consistency of buying or selling behaviour. Our principal contribution is to suggest an entirely different feature space that is constructed by inferring key parameters of a sequential optimization model that we take as a surrogate for the decision-making process of the traders. In particular, we model trader behaviour in terms of a Markov decision process. We infer the reward (or objective) function for this process from observations of trading actions using a process from machine learning known as inverse reinforcement learning (IRL). The reward functions learned through IRL then constitute a feature space that can be the basis for supervised learning (for classification or recognition of traders) or unsupervised learning (for categorization of traders). Making use of a real-world data-set from the E-Mini futures contract, we compare two principal IRL variants, linear IRL and Gaussian Process IRL, against a method based on summary trading statistics. Results suggest that IRL-based feature spaces support accurate classification and meaningful clustering. Further, we argue that, because they attempt to learn traders' underlying value propositions under different market conditions, the IRL methods are more informative and robust than the summary statistic-based approach and are well suited for discovering new behaviour patterns of market participants.

Keywords: Inverse reinforcement learning; Gaussian process; High-frequency trading; Algorithmic trading; Behavioural finance; Markov decision process; Support vector machine

1. Introduction

Financial markets have changed dramatically over the past 10 years or so. These changes reflect the culmination of a decade-long trend from a market structure with primarily manual floor trading to a market structure dominated by automated computer trading. This rapid transformation has been driven by the evolution of technologies for generating, routing and executing orders, which have dramatically improved the speed, capacity and sophistication of the trading functions that are available to market participants.

High-quality trading markets promote capital formation and allocation by establishing prices for securities and by enabling investors to enter and exit their positions in securities wherever and whenever they wish to do so. The one important feature of all types of algorithmic trading strategy is to discover the underlying persistent tradable phenomena and generate

*Corresponding authors. Emails: syang14@stevens.edu, ak67@mit.edu appropriate trading opportunities. These trading opportunities include microsecond price movements that allow a trader to benefit from market-making trades, several minute-long strategies that trade on momentum forecast by market microstructure theories, and several hour-long market movements that surround recurring events and deviations from statistical relationship (Aldridge 2010). Algorithmic traders then design their trading algorithms and systems with the aim of generating signals that result in consistent positive outcomes under different market conditions. Different strategies may target different frequencies, and the profitability of a trading strategy is often measured by a certain return metric. The most commonly used measure is the Sharpe ratio, a risk-adjusted return metric first proposed by Sharpe (1966).

In particular, there is a subgroup within the algorithmic trading strategies called high-frequency trading (HFT) strategies that have attracted significant attention from investors, regulators, policy-makers and academics. According to the US Securities and Exchange Commission, high-frequency traders are "professional traders acting in a proprietary capacity that engage in strategies that generate a large number trades on daily basis." (The SEC Concept Release on Equity Market Structure, 75 Fed. Reg. 3603, 21 January 2010). The SEC characterized HFT as (1) the use of extraordinary high-speed and sophisticated computer programs for generating, routing and executing orders; (2) use of co-location services and individual data feeds offered by exchanges and others to minimize network and other types of latencies; (3) very short time-frames for establishing and liquidating positions; (4) the submission of numerous orders that are cancelled shortly after submission; and (5) ending the trading day in as close to a flat position as possible (that is, not carrying significant, unhedged positions over night). Although many HFT strategies exist today and they are largely unknown to public, researchers have recently shed light on their general characteristics. Several illustrative HFT strategies include: (1) acting as an informal or formal market-maker, (2) high-frequency relative-value trading and (3) directional trading on news releases, order flow or other high-frequency signals (Jones 2012).

In the past few years there have been a number of studies of HFT and algorithmic trading in general. Their primary objective is to understand the economic impact of these algorithmic trading practices on market quality, including aspects such as liquidity, price discovery process, trading costs, etc. On the empirical side, some researchers have been able to identify a specific HFT from the data, and others are able to identify whether a trade is from algorithmic traders. Given the amount of information provided by exchanges and data vendors, it is possible to describe patterns in algorithmic order submission, order cancellation and trading behaviour. It is also possible to see whether algorithmic or HFT activities are correlated with bid–ask spreads, temporary and/or permanent volatility, trading volume, and other market activity and quality measures. Hendershott *et al.* (2004) study the implementation of an automated quote at the New York Exchange. They find that the implementation of auto-quote is associated with an increase in electronic message traffic and an improvement in market quality, including narrowed effective spreads, reduced adverse selection and increased price discovery. These effects are concentrated in large-cap firms, and there is little effect in the small-cap stocks. Menkveld (2012) studies the July 2007 entry of a high-frequency market-maker into the trading of Dutch stocks. He argues that the competition between trading venues facilitated the arrival of high-frequency market-makers and, in general, HFTs, and he shows that high-frequency market-maker entry is associated with 23% less adverse selection. Also the volatility measured using 20 minutes realized volatility is unaffected by the entry of the high-frequency market-maker. Riordan and Storkenmaier (2012) examine the effect of a technological upgrade on the market quality of 98 actively traded German stocks. They conclude that the ability to update quotes faster helps liquidity providers minimize their losses to liquidity demanders, and more price discovery takes place. Boehmer *et al.* (2012) examine international evidence on electronic message traffic and the market quality across 39 stock exchanges over the 2001–2009 period. They find that co-location increases algorithmic trading and HFT, and that the introduction of co-location improves liquidity and the information efficiency of prices. They claim, however, that volatility does not decline as much as it would based on the observed narrower bid–ask spreads. Gai *et al.* (2012) study the effect of two recent 2010 Nasdaq technology upgrades that reduce the minimum time between messages from 950 nanoseconds to 200 nanoseconds. These technological changes lead to a substantial increase in the number of cancelled orders, without significant change in overall trading volume or in bid–ask spreads and depths. Overall, these studies have focused on empirical evidence that an increase in algorithmic trading has positive influence on market quality in general.

On the theoretical side, there are a number of models developed to understand the economic impact of these algorithmic trading practices. Biais *et al.* (2012) conclude that HFT can trade on new information more quickly than non-HFT, generating adverse selection costs, and they also find multiple equilibrium points in their model, and some exhibit social inefficiency over investment in HFT. The model from Jovanovic and Menkveld (2010) shows that HFT can avoid some adverse selection, and can provide some that benefit to uninformed investors who need to trade. Martinez and Rosu (2012) conclude from their model that HFT obtains and trades on information an instant before it is available to others, and it imposes adverse selection on market-makers. Therefore, liquidity is worse and prices are no longer efficient. Martinez and Rosu (2012) focus on HFTs that demand liquidity, and suggest that HFT makes market prices extremely efficient by incorporating information as soon as it becomes available. Markets are not destabilized, as long as there is a population of market-makers standing ready to provide liquidity at competitive prices. Other related theoretical models include Pagnotta and Philippon (2012), who focus on the investment in speed made by exchanges in order to attract trading volume from speed sensitive investors. Moallemi and Saglam (2012) argue that a reduction in latency allows limit-order submitters to update their orders more quickly, thereby reducing the value of the trading option that a limit order grants to a liquidity demander. The common theme in these models is that HFT may increase adverse selection, and it is harmful for liquidity. However, the ability to intermediate for traders who arrive at different times is generally good for liquidity.

Moreover, there have been a number of studies focused on algorithmic traders' behaviours. These studies examine the trading activities of different types of traders and try to distinguish their behavioural differences. Hendershott and Riordan (2013) use exchange classifications to distinguish algorithmic traders from orders managed by humans. They find that algorithmic traders concentrates on smaller trade sizes, while large block trades of 5,000 shares or more are predominantly originated by human traders. Algorithmic traders consume liquidity when bid–ask spreads are relatively narrow, and they supply liquidity when bid–ask spreads are relatively wide. This suggests that algorithmic traders provide a more consistent level of liquidity through time. Brogaard (2012) and Hendershott *et al.* (2004) work with Nasdaq data that flag whether trades involve HFT. Hendershott *et al.* (2004) find that HFT accounts for about 42% of (double-counted) Nasdaq volume in large-cap stocks but only about 17% of volume in small-cap stocks. They estimate a state-space model that decomposes price changes into permanent and temporary components, and measures the contribution of HFT and non-HFT liquidity supply and liquidity demand to each of these price change components. They find that when HFTs initiate trades, they trade in the opposite direction to the transitory component of prices. Thus, HFTs contribute to price discovery and contribute to efficient stock prices. Brogaard (2012) similarly finds that 68% of trades have an HFT on at least one side of the transaction, and he also finds that HFT participation rates are higher for stocks with high share prices, large market caps, narrow bid–ask spreads, or low stock-specific volatility. He estimates a vector autoregressive permanent price impact model and finds that HFT liquidity suppliers face less adverse selection than non-HFT liquidity suppliers, suggesting that they are somewhat judicious in supplying liquidity. Kirilenko *et al.* (2011) use account-level tick-by-tick data on the E-Mini S&P 500 futures contract, and they classify traders into various categories, including HFTs, opportunistic traders, fundamental traders and noise traders. Benos and Sagade (2012) conduct a similar analysis using UK equity data. These different data-sets provide considerable insight into overall HFT trading behaviour.

One of the important goals of learning traders trading strategies is to be able to categorize and identify the market participants, and be able to further understand their influences related to such important economic issues as multiple characterizations of price formation processes, market liquidity, and order flow, etc. (Hasbrouck 1991, Jones *et al.* 1994, Hasbrouchk and Seppi 2001, Gabaix *et al.* 2003, Gatheral 2010). We assert that enhanced understanding of the economic implication of these different algorithmic trading strategies will yield quantitative evidence of value to market policy-makers and regulators seeking to maintain transparency, fairness and overall health in the financial markets.

In particular, traders deploy different trading strategies where each strategy has a unique value proposition under a particular market condition. In other words, we can cast this problem as a sequential decision problem under different conditions. Traders aim to optimize their decisions overtime and consequently maximize their reward under different market conditions. We can theoretically use reward functions to represent the value system that are encapsulated in the various different trading strategies. It is possible to derive new policies based on the reward functions learned and apply them in a new environment to govern a new autonomous process. This process is defined as reward learning under the framework of inverse reinforcement learning (Ng and Russel 2000, Abbeel and Ng 2004, Ramachandran and Amir 2007). For example, a simple keep-or-cancel strategy for buying one unit, the trader has to decide when to place the order and when to cancel the order based on the market condition, which may likely be characterized as a stochastic process. However, the value proposition for the trader is to buy one unit of the security at the lowest price possible. This could be realized in a number of ways. It could be described as a reward function meaning when the system is in a particular state, the trader is always looking for a fixed reward. This notion of value proposition drives the trader to take corresponding actions according to the market conditions. This ultimately constitutes trader's policies or strategies. Therefore, a strategy under a certain value proposition can be consistently programmed in algorithms to achieve its goal of buy-one-unit in an optimal way. Consequently, strategies developed under certain value frameworks can be observed, learned and even reproduced in a different environment (such as a simulated financial market where impact of these strategies can be readily assessed). As documented by Yang *et al.* (2012), Hayes *et al.* (2012) and Paddrik *et al.* (2012), the manipulative or disruptive algorithmic strategies can be studied and monitored by market operators and regulators to prevent unfair trading practices. Furthermore, new emerging algorithmic trading practices can be assessed and new regulations and policies can be evaluated to maintain the overall health of the financial markets.

In this study, we model the trading behaviour of different market participants by the solution to an inverse Markov decision process (MDP). We try to describe how traders are able to take actions in a highly uncertain environment to reach return goals on different horizons. This task can be solved using dynamic programming (DP) and reinforcement learning (RL) based on MDP. The model accounts for traders' preferences and expectations of uncertain state variables. In a general MDP modelling setting, we describe these variables in two spaces: the state space and the action space. From the trading decision perspective, we can parameterize learning agents using reward functions that depend on state and action. We consider the market dynamics in view of the learning agents' subjective beliefs. The agents perform DP/RL through a sense, trial and learn cycle. First, the agents gain state information from sensory input. Based on the current state, knowledge and goals, the agents find and choose the best action. Upon receiving new feedback, the agents learn to update their knowledge with a goal of maximizing their cumulative expected reward. In the discrete-valued state and action problem space, DP and RL methods use similar techniques involving policy iteration and value iteration algorithms (Sutton and Barto 1998, Bertsekas 2007) to solve MDP problems. Formalisms for solving forward problems of RL are often divided into model-based and model-free approaches (Sutton and Barto 1998, Daw *et al.* 2005).

As framed by Abbeel and Ng (2004) under the inverse reinforcement learning (IRL) framework, the entire field of reinforcement learning is founded on the presupposition that the reward function, rather than policy, is the most succinct, robust and transferable definition of the task. However, the reward function is often difficult to know in advance for some real-world tasks, so the following difficulties may arise: (1) We have no experience to tackle the problem; (2) We have experience but cannot interpret the reward function explicitly; (3) The problem we solve may be interacting with the adversarial decision-makers who make all their effort to keep the reward function secret. Rather than accessing the true reward function, it is easier to observe the behaviour of some other agents (teacher/expert) to determine how to solve the problem. Hence, we have motivation to learn from observations. Technical approaches to learning from observations generally fall into two broad categories Ratliff *et al.* (2009). The first category, called imitation learning, attempts to use supervised learning to predict actions directly from observations of features of the environments, which is unstable and vulnerable to highly uncertain environment. The second category is concerned with how to learn the reward function that characterizes the agent's objectives and preferences in MDP (Ng and Russel 2000).

IRL was first introduced in machine learning literature by Ng and Russel (2000) in formulating it as an optimization problem to maximize the sum of differences between the quality of the optimal action and the quality of the next-best action. Other algorithms have been developed or integrated into apprenticeship learning based on this linear approximation of the reward function. The principal idea of apprenticeship learning using IRL is to search mixed solutions in a space of learned policies with the goal that the cumulative feature expectation is near that of the expert (Abbeel and Ng 2004, Syed *et al.* 2008).

Other algorithms have also been developed under the IRL framework. A game-theoretic approach to apprenticeship learning using IRL was developed in the context of a two-player zero-sum game in which the apprentice chooses a policy and the environment chooses a reward function (Syed and Schapire 2008). Another algorithm for IRL is policy matching, in which the loss function penalizing deviations from the expert's policy that is minimized by tuning the parameters of the reward functions (Neu and Szepesvari 2007). The maximum entropy IRL is proposed in the context of modelling real-world navigation and driving behaviours (Ziebart *et al.* 2008). The algorithms for apprenticeship learning using IRL do not actually aim to recover the reward function but instead are only concerned with the optimal policy. Ramachandran and Amir consider IRL from a Bayesian perspective without assuming the linear approximation of the reward function (Ramachandran and Amir 2007). Their model interprets the observations from the expert as the evidence that is used to obtain a posterior distribution over reward using Markov Chain Monte Carlo simulation. Recent theoretical works on IRL such as the framework of the linear-solvable MDP (Dvijotham and Todorov 2004), bootstrap learning (Boularias and Chaib 2010) and feature construction (Levine *et al.* 2010), have also improved the learning performance. IRL has also been successfully applied to many real-world problems, such as the automatic control of helicopter flight (Abbeel *et al.* 2010) and the motion control of an animation system in computer graphics (Lee and Zoran 2010).

We apply a Gaussian process-based IRL (GPIRL) model proposed by Qiao and Beling (2011) to learn the trading behaviours under different market conditions. In this GPIRL, a Gaussian prior is assigned on the reward function and the reward function is treated as a Gaussian process. This approach is similar to that of Ramachandran and Amir (2007), who view the state-action samples from agents as the evidence that will be used to update a prior value in the reward function, under a Bayesian framework. The solution (Ramachandran and Amir 2007) depends on non-convex optimization using Markov Chain Monte Carlo simulation. Moreover, the ill-posed nature of the inverse learning problem also presents difficulties. Multiple reward functions may yield the same optimal policy, and there may be multiple observations at a single state given the true reward function. The GPIRL model aims to address the ill-posed nature of this problem by applying Bayesian inference and preference graphs. Here, we are faced with the challenge of modelling traders' action as non-deterministic policies. In general, agent's policies range from deterministic Markovian to randomized history dependent, depending on how traders incorporate past information and how traders select actions. Due to the uncertainty of the environment and the random error of the measurement in the observations, a deterministic policy could very likely be perceived as a non-deterministic one. Modelling traders' reward function using a Gaussian process is well suited to address these issues. One of the main novel features of this approach is that it not only represents a probabilistic view but is also computationally tractable.

The dynamic nature of financial markets makes it possible to postulate a priori a relationship between the market variables we observe and those we wish to predict. The main contributions of this study can be summarized as follows:

(i) We propose an entirely different feature space that is constructed by inferring key parameters of a sequential optimization model that we take as a surrogate for the decision-making process of the traders. We infer the reward (or objective) function for this process from observation of trading actions using a process from machine learning known as inverse reinforcement learning (IRL).
(ii) We model traders' reward functions using a Gaussian process. We also apply preference graphs to address the non-deterministic nature of the observed trading behaviours, reducing the uncertainty and computational burden caused by the ill-posed nature of the inverse learning problem.
(iii) We suggest a quantitative behavioural approach in categorizing algorithmic trading strategies using weighted scores over time in the reward space, and we conclude that it performs consistently better than the existing summary statistic-based trader classification approach (Kirilenko *et al.* 2011).

The remainder of this paper is organized as follows: First, we discuss the summary statistics approach to trader classification in section 2. We then discuss the framework of which we use to model market dynamics and the traders' decisions in section 3. We extend the MDP and introduce IRL formulation in section 4. We review the original linear IRL formulation and provide a Bayesian probabilistic model to infer the reward function using Gaussian processes. We apply the GPIRL algorithm to the E-Mini S&P 500 Futures market as an experiment in section 5. We show that the GPIRL algorithm can accurately capture algorithmic trading behaviour based on observations of the high-frequency data. We also compare our behaviour-based classification results with the results from Kirilenko *et al.* (2011), and show that our behavioural approach represents a consistent improvement. Finally, we provide concluding remarks about the GPIRL and its applications in section 6.

2. Summary statistics approach to trader classification

Kirilenko *et al.* (2011) suggest an approach to classify individual trading accounts based on the summary statistics of trading volume and inventory and consistency of buying or selling behaviour. Six categories are used to describe individual trading accounts:

(i) High-frequency traders—high volume and low inventory;
(ii) Intermediaries—low inventory;
(iii) Fundamental buyers—consistent intraday net buyers;
(iv) Fundamental sellers—consistent intraday net sellers;
(v) Opportunistic traders—all other traders not classified;
(vi) Small traders—low volume.

In this section, we develop the details for classification rules corresponding to the six categories from Kirilenko *et al.* (2011), and then apply these rules to a real-world futures contract data-set.

2.1. E-Mini market data description

The E-Mini S&P 500 is a stock market index of futures contracts traded on the Chicago Mercantile Exchange's (CME) Globex electronic trading platform. The notional value of one contract is \$50 times the value of the S&P 500 stock index. The tick size for the E-Mini S&P 500 is 0.25 index points or \$12.50. For example, the S&P 500 Index futures contract is trading at \$1,400.00, then the value of one contract is \$70 000. The advantages to trading E-Mini S&P 500 contracts include liquidity, greater affordability for individual investors and around-the-clock trading.

Trading takes place 24 h a day with the exception of a short technical maintenance shutdown period from 4:30 pm to 5:00 pm. The E-Mini S&P 500 expiration months are March, June, September and December. On any given day, the contract with the nearest expiration date is called the front-month contract. The E-Mini S&P 500 is cash-settled against the value of the underlying index and the last trading day is the third Friday of the contract expiration month. The initial margin for speculators and hedgers are \$5,625 and \$4,500, respectively. Maintenance margins for both speculators and hedgers are \$4,500. There is no limit on how many contracts can be outstanding at any given time.

The CME Globex matching algorithm for the E-Mini S&P 500 offers strict price and time priority. Specifically, limit orders that offer more favourable terms of trade (sell at lower prices and buy at higher prices) are executed prior to pre-existing orders. Orders that arrived earlier are matched against the orders from the other side of the book before other orders at the same price. This market operates under complete price transparency. This straight forward matching algorithm allows us to reconstruct the order book using audit trail messages archived by the exchanges and allows us to replay the market dynamics at any given moment.

In this paper, empirical work is based on a month of E-Mini order book audit trail data. The audit trail data includes all the order book events timestamped at a millisecond time resolution, and contains the following data fields: date, time (the time when the client submits the order to the exchange), conf_time (the time when the order is confirmed by the matching engine), customer account, tag 50 (trader identification number), buy or sell flag, price, quantity, order ID, order type (market or limit), and func_code (message type, e.g. order, modification, cancellation, trade, etc.).

2.2. Summary statistic-based classification of E-Mini market data

We apply the set of the statistics-based trader classification rules documented by Kirilenko *et al.* (2011) on our E-Mini data-set. For fundamental traders, we calculate their end of the day net position. If the end-of-the-day net position is more than 15% of their total trading volume on that day, we categorize them either as fundamental buyers or fundamental sellers depending on their trading directions. We also identify Ssall traders as those accounts with a trading volume of nine contracts or less. We apply the criteria (Kirilenko *et al.* 2011) for intermediaries, opportunistic traders and high-frequency traders, and obtain consistent results based on the one-month data. There are two steps involved in this process. First, we ensure that the account's net holdings fluctuate within 1.5% of its end of day level, and second, we ensure the account's end of the day net position is no more than 5% of its daily trading volume. Then if we define HFTs as a subset of intermediaries (top 7% in daily trading volume), we find that there is a significant amount of overlap between HFTs and opportunistic traders. The problem is that the first criterion is not well defined, as the fluctuation of net holdings is vaguely defined. Net holdings could be measured in different ways.

In consultation with the authors of Kirilenko *et al.* (2011), we choose the standard deviation of an account's net position measured on the event clock as a measure of an account's holding fluctuation. With this definition, we find that a 1.5% fluctuation is too stringent for HFTs, because many high trading volume accounts are classified as opportunistic traders, while in reality their end of day positions are still very low compared with other opportunistic traders. Therefore, it is adequate to relax the first criterion requiring that the standard deviation of the account's net holdings throughout the day is less than its end of day holding level. We find that the newly adjusted criteria classify most high volume trading accounts as HFTs, and this classification rule is validated from the registration information we can acquire. Without this adjustment, almost all the top trading accounts are incorrectly classified as opportunistic traders. Table 1 summarizes the results after applying the new classification rule demonstrating that the modified classification criteria identified more HFTs. On average, there are 38 HTF accounts, 181 intermediary accounts, 2658 opportunistic accounts, 906 fundamental buyer accounts, 775 fundamental seller accounts, and 5127 small trader accounts. Over the 4-week period, only 36% of the 120 accounts are consistently

Table 1. The E-Mini S&P 500 futures market data summary.

Date	HFTs	Intermediaries	Opportunistic traders	Fundamental buyers	Fundamental sellers	Total number of accounts[a]	Total trading volume[a]
10/04/2012	39	193	2833	940	818	10 425	3 261 852
10/05/2012	38	162	2598	1191	1055	11 495	3 875 232
10/06/2012	38	167	2401	895	712	9065	2 852 244
10/07/2012	39	196	2726	919	747	9841	3 424 768
10/08/2012	32	162	2511	847	812	9210	3 096 800
10/11/2012	21	118	1428	636	573	6230	1 765 254
10/12/2012	38	186	2687	896	745	9771	3 236 904
10/13/2012	38	187	2582	1020	840	10 297	3 699 108
10/14/2012	30	198	3001	1070	795	10 591	4 057 824
10/15/2012	46	210	3109	890	773	9918	4 437 826
10/18/2012	37	173	2126	869	724	8735	2 458 510
10/19/2012	52	216	3651	1030	974	11 600	5 272 672
10/20/2012	39	176	2949	951	877	10 745	3 956 790
10/21/2012	43	240	3370	952	771	10 980	4 230 194
10/22/2012	32	143	1837	676	629	7370	2 026 234
10/25/2012	38	181	2533	888	684	9228	3 074 558
10/26/2012	37	175	2726	816	709	9568	3 000 628
10/27/2012	45	186	2973	919	820	10 472	3 850 556
10/28/2012	39	185	2873	914	705	9777	3 485 910
10/29/2012	37	160	2247	794	744	8369	3 012 860

[a]The remaining accounts consist of the small trader accounts.

classified as the same type of traders. If we rank these accounts by their daily trading volume, we find that only 40% of the top 10 accounts are consistently classified as the same trader types. The variation occurs among the HFTs, intermediaries and opportunistic traders.

3. Markov decision process model of market dynamics

In this section, we develop a Markov decision process (MDP) model of trader behaviour. This model will then serve as the basis for the inverse reinforcement learning process described in section 4.

3.1. MDP background and notation

The primary aim of our trading behaviour-based learning approach is to uncover decision-makers' policies and reward functions through the observations of an expert whose decision process is modelled as an MDP. In this paper, we restrict our attention to a finite countable MDP for easy exposition, but our approach can be extended to continuous problems if desired. A discounted finite MDP is defined as a tuple $\mathbb{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, where

- $\mathcal{S} = \{s_n\}_{n=1}^{N}$ is a set of N states. Let $\mathcal{N} = \{1, 2, \cdots, N\}$.
- $\mathcal{A} = \{a_m\}_{m=1}^{M}$ is a set of M actions. Let $\mathcal{M} = \{1, 2, \ldots, M\}$.
- $\mathcal{P} = \left\{\mathbf{P}_{a_m}\right\}_{m=1}^{M}$ is a set of state transition probabilities (here $\mathbf{P}_{a_m}$ is a $N \times N$ matrix where each row, denoted as $\mathbf{P}_{a_m}(s_n, :)$, contains the transition probabilities upon taking action a_m in state s_n. The entry $\mathbf{P}_{a_m}(s_n, s_{n'})$ is the probability of moving to state $s_{n'}$, $n' \in \mathcal{N}$ in the next stage.).
- $\gamma \in [0, 1]$ is a discount factor.
- r denotes the reward function, mapping from $\mathcal{S} \times \mathcal{A}$ to $\Re$ with the property that

$$r(s_n, a_m) \triangleq \sum_{n' \in \mathcal{N}} \mathbf{P}_{a_m}(s_n, s_{n'}) r(s_n, a_m, s_{n'})$$

where $r(s_n, a_m, s_{n'})$ denotes the function giving the reward of moving to the next state $s_{n'}$ after taking action a_m in current state s_n. The reward function $r(s_n, a_m)$ may be further reduced to $r(s_n)$, if we neglect the influence of the action. We use $\mathbf{r}$ for reward vector through out this paper. If the reward only depends on state, we have $\mathbf{r} = (\mathbf{r}(s_1), \ldots, \mathbf{r}(s_N))$. If we let $\mathbf{r}$ be the vector of the reward depending on both state and action. We have

$$\begin{aligned}\mathbf{r} &= (\underbrace{\mathbf{r}_1(s_1), \ldots, \mathbf{r}_1(s_N)}, \ldots, \underbrace{\mathbf{r}_M(s_1), \ldots, \mathbf{r}_M(s_N)}) \\ &= (\qquad \mathbf{r}_1, \qquad \ldots, \qquad \mathbf{r}_M).\end{aligned}$$

In an MDP, the agent selects an action at each sequential stage, and we define a *policy* (*behaviour*) as the way that the actions are selected by a decision-maker/agent. Hence, this process can be described as a mapping between state and action, i.e. a random state-action sequence $\left(s^0, a^0, s^1, a^1, \ldots s^t, a^t, \cdots\right)$,† where s^{t+1} is connected to (s^t, a^t) by $\mathbf{P}_{a^t}(s^t, s^{t+1})$.

We also define rational agents as those that behave according to the optimal decision rule where each action selected at any stage maximizes the value function. The *value function* for a policy π evaluated at any state s^0 is given as $V^{\pi}(s^0) = E[\sum_{t=0}^{\infty} \gamma^t r(s^t, a^t)|\pi]$. This expectation is over the distribution of the state sequence $\left\{s^0, s^1, \ldots\right\}$ given the policy $\pi = \left\{\mu^0, \mu^1, \ldots\right\}$, where $a^t = \mu^t(s^t)$, $\mu^t(s^t) \in U(s^t)$ and $U(s^t) \subset \mathcal{A}$. The objective at state s is to choose a policy that maximizes the value of $V^{\pi}(s)$. A policy π^* is an optimal policy, if is

†Superscripts represent time indices. For example s^t and a^t, with the upper-index $t \in \{1, 2, \ldots\}$, denote state and action at the t-th horizon stage, while s_n (or a_m) represents the n-th state (or m-th action) in $\mathcal{S}$ (or $\mathcal{A}$).

then $V^{\pi^*}(s^0) = \sup_\pi E[\sum_{t=0}^{\infty} \gamma^t r(s^t, a^t)|\pi]$. Similarly, there is another function called the *Q-function* (or *Q-factor*) that judges how well an action is performed in a given state. The notation $Q^\pi(s, a)$ represents the expected return from state s when action a is taken and thereafter policy π is followed.

In the infinite-horizon case, the stationary Markovian structure of the problem implies that the only variable that affects the agent's decision rule and the corresponding value function should be time invariant. We then have the essential theory of MDPs (Bellman 1957) as follows:

THEOREM 3.1 (Bellman Equations) *Given a stationary policy π, $\forall n \in \mathcal{N}$, $m \in \mathcal{M}$, $V^\pi(s_n)$ and $Q^\pi(s_n, a_m)$ satisfy*

$$V^\pi(s_n) = r(s_n, \pi(s_n)) + \gamma \sum_{n' \in \mathcal{N}} \mathbf{P}_{\pi(s_n)}(s_n, s_{n'}) V^\pi(s_{n'}),$$

$$Q^\pi(s_n, a_m) = r(s_n, a_m) + \gamma \sum_{n' \in \mathcal{N}} \mathbf{P}_{a_m}(s_n, s_{n'}) V^\pi(s_{n'}).$$

THEOREM 3.2 (Bellman Optimality) *π is optimal if and only if, $\forall n \in \mathcal{N}$, $\pi(s_n) \in \arg\max_{a \in \mathcal{A}} Q^\pi(s_n, a)$.*

Based on the above theorem of MDPs, we have the following equations to represent the Q-function as a the reward function.

$$Q^\pi(s_n, a_m) = \mathbf{r}_m(s_n) + \gamma \mathbf{P}_{a_m}(s_n, :)(\mathbf{I} - \gamma \mathbf{P}_\pi)^{-1} \mathbf{r_m},$$

where $\mathbf{P}_\pi$ represents the state transition probability matrix for following policy π at every state, and $\mathbf{r_m}$ represents the reward vector under action a_m.

3.2. *Constructing an MDP model from order book data*

Figure 1 shows the entire life-cycle of an order initiated by a client of an exchange. The order book audit trail data contains these messages, and the entire order history (i.e. order creation, order modifications, fills, cancellation, etc.) can be retrieved and analyzed. To construct an MDP model of trader behaviour, we first reconstruct the limit order book using the audit trail messages. The order book then contains bid/ask prices, market depth, liquidity, etc. During this process on the E-Mini data described in section 2.1, we processed billions of messages for each trading date, and built price queues using the price and time priority rule.

In this construction process, we choose event tick time rather than natural wall clock time. In other words, every activity including arrival of a new order, cancellation of an existing order, and placement of a market order is counted as one tick. Once we have the order book at any given event tick, we take the market depth at five different levels as our base variables and then discretize these variables to generate an MDP model state space. This study extends the MDP model documented by Yang *et al.* (2012) to obtain five variables, i.e. order volume imbalance between the best bid and the best ask prices, order volume imbalance between the 2nd best bid and the 2nd best ask prices, order volume imbalance between the 3rd best bid and the 3rd best ask prices, the order book imbalance at the 5th best bid and the 5th ask prices, and the inventory level/holding position (see figure 2(b)). Then we discretize the values of the five variables into three levels defined as high (above $\mu + 1.96\sigma$), neutral ($\mu \pm 1.96$) and low (below $\mu - 1.96\sigma$). Based on our observation that the first three best bid and ask prices change the most, we find that it is critical to include the first three-level order book imbalance variables in modelling the limit order book dynamics. As argued by Yang *et al.* (2012), these volume-related variables reflect the market dynamics on which the traders/algorithms depend to place their orders at different prices.

As the volume imbalance at the best bid/ask prices is the most sensitive indicator of the trading behaviour of HFTs, intermediaries and some of the opportunistic traders, we also hypothesize that the volume imbalance at other prices close to the book prices will be useful in inferring trader behaviour. As demonstrated in previous work (Yang *et al.* 2012), the private variable of a trader's inventory level provides critical information about trader's behaviour. Traders in high-frequency environments strive to control their inventory levels as a critical measure of controlling the risk of their position (Easley *et al.* 2010, Brogaard 2010, Kirilenko *et al.* 2011). HFTs and Market-makers tend to turn over their inventory level five or more times a day and to hold very small or even zero inventory positions at the end of the trading session. These observations provide strong support for the introduction of a position variable to characterize trader behaviour in our model. Therefore, together with the volume imbalance variables, we propose a computational model with $3^5 = 243$ states.

Next, we need to define the action space. In general, there are three types of actions: placing a new order, cancelling an existing order, or placing a market order. We divide the limit order book into 10 buckets at any given point of time by the following price markers: the best bid price, the 2nd best bid price, the 3rd best bid price, between the 4th and 5th bid prices, below the 5th best bid price, the best ask price, the 2nd best ask price, the 3rd best ask price, between the 4th and 5th ask prices, and above the 5th best ask price. Then, at any given point of time, a trader can take 22 actions. The price markers used to define the price ranges are illustrated in figure 2. We define the order volume imbalance through buckets. In other words, the boundaries of these buckets define the volume between the two adjacent best price levels. In case of missing prices between the two adjacent best price levels, we only count the total volume between them. Therefore, missing prices between these best price levels will be acceptable for our volume imbalance modelling. We use unit size for all the actions. Orders other than unit size are treated as repeated actions without state transition.

Once we have both the state space and the action space defined, we can then construct an MDP model where traders interact with the order book. All actions from a trader can be correlated with certain states in a probabilistic sense. From trading strategy perspective, traders may choose to either react to or influence order book movements. Although in most of the cases, traders may choose to react to market changes to achieve their investment objectives, anticipative or aggressive strategies are also widely used. To some degree, certain aggressive strategies through which specific actions are designed to create desirable market conditions for themselves can be defined as manipulative, but it is out of the scope of this paper. Over all, a predefined trading strategy would consistently take certain actions under certain market conditions within its permissible or desirable inventory levels until these conditions change. This process can be captured in a transition matrix of an MDP model. Essentially, we have a 243x243 matrix for each action, which

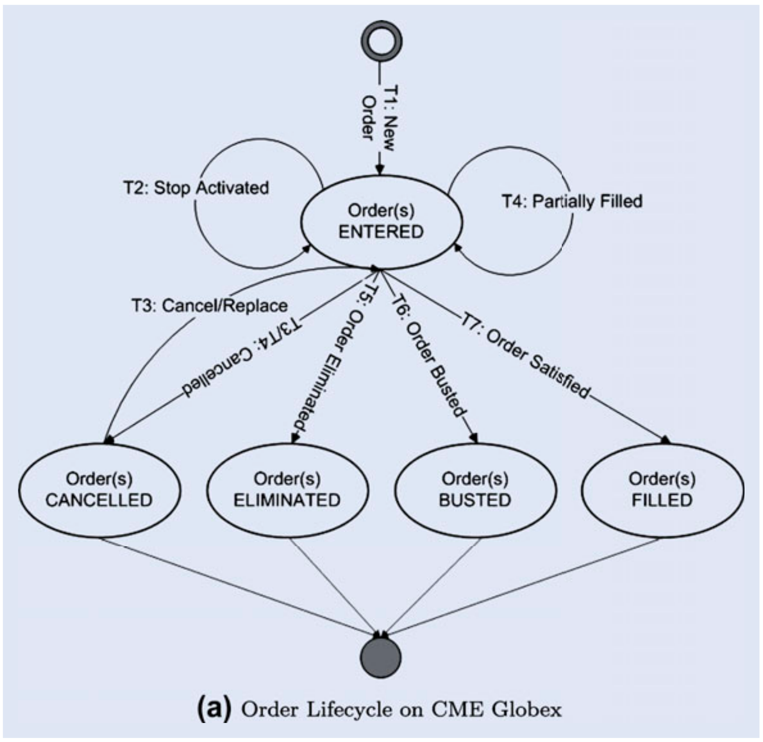

Figure 1. *CME globex order lifecycle*. T1: Trader submits a new order; T2: The state of an order is changed, if a stop is activated; T3: A trader may choose to cancel an order, and the state of an order can be modified multiple times; T4: When an order is partially filled, the quantity remaining decreases; T5: Order elimination is similar to order cancellation except it is initiated by the trading engine; T7: An order may be filled completely; T6: Trades can be eliminated after the fact by the exchanges.

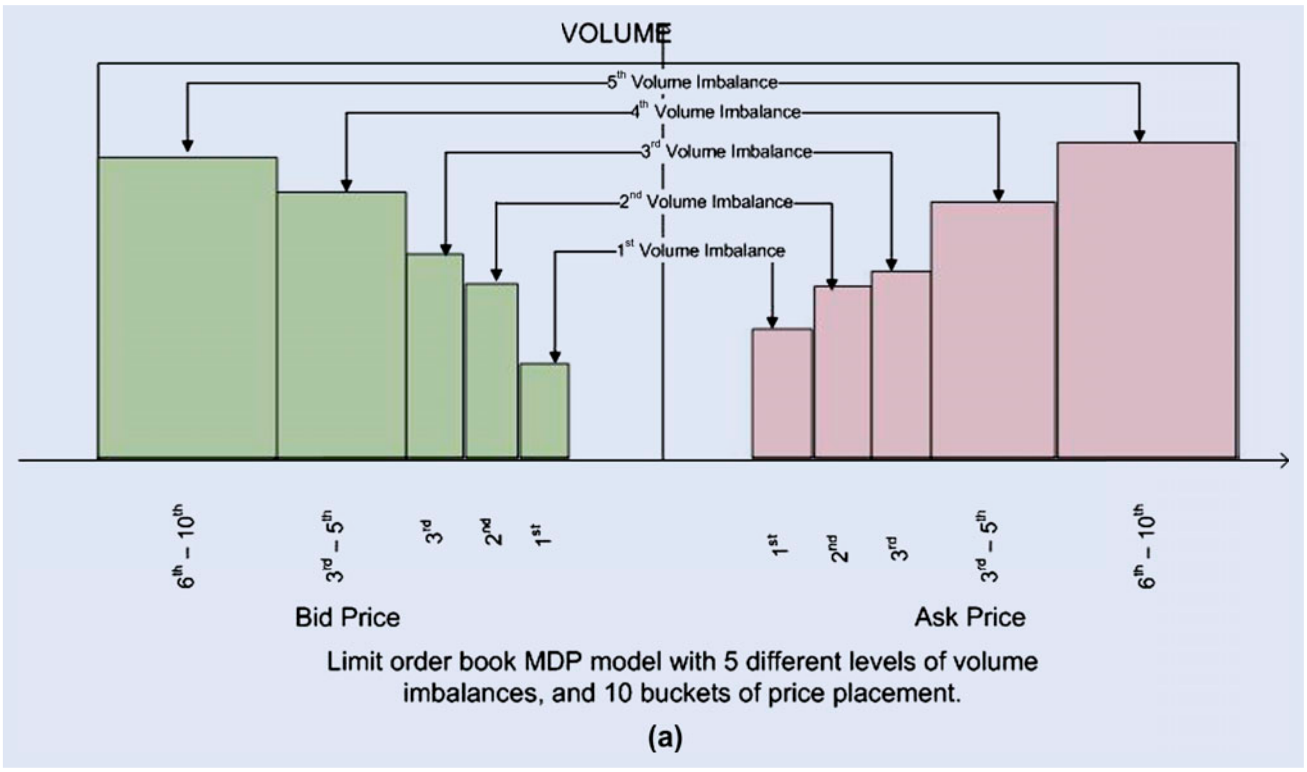

Figure 2. *Order book MDP model:* This graph shows the state variables used in the MDP model.

describes the Markovian decision process of a trading strategy. In reality, these transition matrices are very sparse. For each action, there are only a few desirable states, and most of the elements are zeros. In the end, it is those scarce transitions that reflect the uniqueness of the corresponding strategies. In addition, we define a non-action in which the traders choose to do nothing as the market moves. Eventually, we have a 23x243x243 3-dimensional matrix for each trader based on the frequencies we calculated from the sample data.

4. Inverse reinforcement learning

Given an MDP $\mathbb{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, let us define the inverse Markov decision process (IMDP) $\mathbb{M}_{\mathbb{I}} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{O})$. The process $\mathbb{M}_{\mathbb{I}}$ includes the states, actions and dynamics of $\mathbb{M}$, but lacks a specification of the reward function, r. By way of compensation, $\mathbb{M}_{\mathbb{I}}$ includes a set of observations $\mathcal{O}$ that consists of state-action pairs generated through the observation of a decision-maker. We can define the *inverse reinforcement learning* (IRL) problem associated with $\mathbb{M}_{\mathbb{I}} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{O})$ to be that of finding a reward function such that the observations $\mathcal{O}$ could have come from an optimal policy for $\mathbb{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$. The IRL problem is, in general, highly under-specified, which has led researchers to consider various models for restricting the set of reward functions under consideration. Ng and Russel (2000), in a seminal consideration of IMDPs and associated IRL problems, observed that, by the optimality equations, the only reward vectors consistent with an optimal policy π are those that satisfy the set of inequalities

$$(\mathbf{P}_\pi - \mathbf{P}_a)(\mathbf{I} - \gamma \mathbf{P}_\pi)^{-1}\mathbf{r} \geq \mathbf{0}, \quad \forall a \in \mathcal{A}, \tag{1}$$

where $\mathbf{P}_\pi$ is the transition probability matrix relating to observed policy π and $\mathbf{P}_a$ denotes the transition probability matrix for other actions. Note that the trivial solution $\mathbf{r} = \mathbf{0}$ satisfies the constraints (1), which highlights the under-specified nature of the problem and the need for reward selection mechanisms.

In the machine learning and artificial intelligence literature, a principal motivation for considering IRL problems is the idea of apprenticeship learning, in which observations of state-action pairs are used to learn the policies followed by experts for the purpose of mimicking or cloning behaviour. By its nature, apprenticeship learning problems arise in situations where it is not possible or desirable to observe all state-action pairs for the decision-maker's policy. The basic idea of apprenticeship learning through IRL is to first use IRL techniques to learn the reward function (vector) and then use that function to define an MDP problem, which can then be solved for an optimal policy. Our process is quite different. We learn the reward function with IRL and then directly use the rewards as features for classifying and clustering traders or trading algorithms.

4.1. *Linear IRL*

Ng and Russel (2000) advance the idea choosing the reward function to maximize the difference between the optimal and suboptimal policies, which can be done using a linear programming formulation.

Most of the existing IRL algorithms make some assumption about the form of the reward function. Prominent examples include the model in Ng and Russel (2000), which we term linear IRL (LIRL) because of its linear nature. In LIRL, the reward function is written linearly in terms of basis functions, and effort is made to maximize the quantity

$$\sum_{s \in \mathcal{S}} [Q^\pi(s, a') - \max_{a \in \mathcal{A} \backslash a'} Q^\pi(s, a)], \quad \forall a' \in \mathcal{A}. \tag{2}$$

The optimization problem in Ng and Russel (2000) is equivalent to the following optimization program:

$$\begin{array}{ll} \max_{\mathbf{r}} \sum_{s \in \mathcal{S}} \beta(s) - \lambda \sum_{s \in \mathcal{S}} |\mathbf{r}(s)| & \\ \text{s.t.} & \\ (\mathbf{P}_\pi - \mathbf{P}_a)(I - \gamma \mathbf{P}_\pi)^{-1}\mathbf{r} & \geq \beta(s), \quad \forall a \in \mathcal{A}, \forall s \in \mathcal{S} \\ \beta(s) & \geq \mathbf{0}, \quad \forall s \in \mathcal{S}, \end{array}$$

where λ is a regularization parameter included to encourage sparse solution vectors, and $\beta(s)$ is the lower bound of the function: $Q^\pi(s, a') - \max_{a \in \mathcal{A} \backslash a'} Q^\pi(s, a)$. Yang *et al.* (2012) use this approach to find a feature space that can be used to classify and cluster simulated trading agents. Here, reward is defined as a function of state only, which means that agents do not distinguish actions in seeking rewards under certain conditions. This action-independent approximation for reward is discussed further in the context of the Bayesian IRL framework.

4.2. *Bayesian IRL framework*

Ramachandran and Amir (2007) originally proposed a Bayesian Framework for IRL. The posterior over reward is written as:

$$p(r|\mathcal{O}) = p(\mathcal{O}|r)p(r) \propto \prod_{(s,a) \in \mathcal{O}} p(a|s, r).$$

Then, the IRL problem is written as $\max_r \log p(\mathcal{O}|r) + \log p(r)$. For many problems, however, the computation of $p(r|\mathcal{O})$ may be complicated and some algorithms use Markov chain Monte Carlo (MCMC) to sample the posterior probability. Below, we adopt a different approach that uses the idea of selecting reward on the basis of *maximum a posteriori* (MAP) estimate computed using convex optimization.

4.3. *Gaussian process IRL*

We now turn to an IRL problem that addresses observations from a decision-making process in which the reward function has been contaminated by Gaussian noise. In particular, we assume that the reward vector can be modelled as $r + \mathcal{N}(0, \sigma^2)$, where $\mathcal{N}(0, \sigma^2)$ is Gaussian noise. In the financial trading problem setting, we may observe certain trading behaviour over a period of time, but we may not observe the complete polices behind a particular trading strategy. As discussed earlier, different trading strategies tend to look at different time horizons. Therefore, the observation period becomes critical to the learning process. Furthermore, two types of errors may be introduced into our observations: the first type of error may be introduced during our modelling process. Resolution of these discrete models will introduce errors into our observations. The second potential source of error is the strategy execution process. Execution errors will occur due to the uncertainty of market movements and will eventually appear in our observations, confounding our efforts to determine the true policy. Overall, there are two types of challenges in this learning problem: the uncertainty about reward functions given the observation of decision behaviour and the ambiguity involved in observing multiple actions at a single state.

Qiao and Beling (2011) argue for two different modelling techniques in learning reward functions. To lessen the ambiguity of observing multiple actions at a state, they argue that Bayesian inference should be the basis for understanding the agent's preferences over the action space. This argument is reasonable because the goal of IRL is to learn the reward subjectively perceived by the decision-maker from whom we have collected the observation data. The intuition is that decision-makers will select some actions at a given state because they prefer these actions to others. These preferences are among the countable actions that can be used to represent multiple observations at one state.

Here, we use two examples to demonstrate how the action preference relationships have been constructed based on the MDP model and observed actions. Table 2 shows two example states with multiple observed actions. We then sort the frequency in descending order and construct a two-layer graph: the top layer has the most frequently observed actions and the bottom layer holds all the other actions. Based on this preference observation, we can construct two preference graphs as shown in figure 3. The state transition matrix can be constructed for the entire market for the observation period. In our MDP model, we have a 243×243 matrix for every single action.

In the following, we first introduce the preference theory for the IMDP model, and then we formalize the idea of modelling the reward function as a Gaussian process under the Bayesian inference framework.

4.3.1. Action preference learning.

In this section, we first define the action preference relationship and the action preference graph. At state s_n, $\forall \hat{a}, \check{a} \in \mathcal{A}$, we define the *action preference relation* as:

(i) Action $\hat{a}$ is weakly preferred to $\check{a}$, denoted as $\hat{a} \succeq_{s_n} \check{a}$, if $Q(s_n, \hat{a}) \geq Q(s_n, \check{a})$;
(ii) Action $\hat{a}$ is strictly preferred to $\check{a}$, denoted as $\hat{a} \succ_{s_n} \check{a}$, if $Q(s_n, \hat{a}) > Q(s_n, \check{a})$;
(iii) Action $\hat{a}$ is equivalent to $\check{a}$, denoted as $\hat{a} \sim_{s_n} \check{a}$, if and only if $\hat{a} \succeq_{s_n} \check{a}$ and $\check{a} \succeq_{s_n} \hat{a}$.

An *action preference graph* is a simple-directed graph showing preference relations among the countable actions at a given state. At state s_n, the action preference graph $G_n = (\mathcal{V}_n, \mathcal{E}_n)$ comprises a set $\mathcal{V}_n$ of nodes together with a set $\mathcal{E}_n$ of edges. For the nodes and edges in graph G_n, let us define

(i) Each node represents an action in $\mathcal{A}$. Define a one-to-one mapping $\varphi : \mathcal{V}_n \rightarrow \mathcal{A}$.
(ii) Each edge indicates a preference relation.

Furthermore, we make the following assumption as a rule to build the preference graph, and then we show how to draw a preference graph at state s_n:

At state s_n, if action $\hat{a}$ is observed, we have the following preference relations: $\hat{a} \succeq_{s_n} \check{a}, \forall \check{a} \in \mathcal{A} \setminus \{\hat{a}\}$.

It is, therefore, straightforward to show the following according to Bellman optimality. The variable $\hat{a}$ is observed if and only if $\hat{a} \in \arg\max_{a \in \mathcal{A}} Q(s_n, a)$. Therefore, we have

$$Q(s_n, \hat{a}) > Q(s_n, \check{a}), \ \forall \check{a} \in \mathcal{A} \setminus \{\hat{a}\}$$

According to the definition on preference relations, it follows that if $Q(s_n, \hat{a}) > Q(s_n, \check{a})$, we have $\hat{a} \succ_{s_n} \check{a}$. Hence, we can show that the preference relationship has the following properties:

(1) If $\hat{a}, \check{a} \in \mathcal{A}$, then at state s_n either $\hat{a} \succeq_{s_n} \check{a}$ or $\check{a} \succeq_{s_n} \hat{a}$.
(2) If $\hat{a} \succeq_{s_n} \check{a}$ and $\check{a} \succeq_{s_n} \tilde{a}$, then $\hat{a} \succeq_{s_n} \tilde{a}$.

At this point, we have a simple representation of the action preference graph that is constructed by a two-layer directed graph. We may have either multiple actions at s_n as in figure 4(a) or a unique action at s_n as in figure 4(b). In this two-layer directed graph, the top layer $\mathcal{V}_n^+$ is a set of nodes representing the observed actions and the bottom layer $\mathcal{V}_n^-$ contains the nodes denoting the other actions. The edge in the edge set $\mathcal{E}_n$ can be represented by a formulation of its beginning node u and ending node v. We write the k-th edge as $(u \rightarrow v)_k$ if $u \in \mathcal{V}_n^+$, $v \in \mathcal{V}_n^-$, or the l-th edge $(u \leftrightarrow v)_l$ if $u \in \mathcal{V}_n^-$, $v \in \mathcal{V}_n^-$. Recall the mapping between $\mathcal{V}_n$ and $\mathcal{A}$, the representation $u \rightarrow v$ indicates that action $\varphi(u)$ is preferred over $\varphi(v)$. Similarly, $u \leftrightarrow v$ means that action $\varphi(u)$ is equivalent to $\varphi(v)$.

In the context of financial trading decision process, we may observe multiple actions from one particular trader under certain market conditions. That is to say that the observation data $\mathcal{O}$ may represent multiple decision trajectories generated by non-deterministic policies. To address IRL problems in those cases, Qiao and Beling (2011) propose to process $\mathcal{O}$ into the form of pairs of state and preference graphs similar to the representation shown in figure 5, and then we apply Bayesian inference using the new formulation.

According to Qiao and Beling (2011), we can represent $\mathcal{O}$ as shown in figure 5. At state s_n, its action preference graph is constructed by a two-layer directed graph: a set of nodes $\mathcal{V}_n^+$ in the top layer and a set of nodes $\mathcal{V}_n^-$ in the bottom layer. Under the non-deterministic policy assumption, we adopt a reward structure depending on both state and action.

4.3.2. Gaussian reward process.

Recall that the reward depends on both state and action, and consider r_m, the reward related to action a_m, as a Gaussian process. We denote by $k_m(s_i, s_j)$ the function generating the value of entry (i, j) for covariance matrix $\mathbf{K}_m$, which leads to $\mathbf{r}_m \sim N(0, \mathbf{K}_m)$. Then the joint prior probability of the reward is a product of multivariate Gaussian, namely $p(\mathbf{r}|\mathcal{S}) = \prod_{m=1}^{M} p(\mathbf{r}_m|\mathcal{S})$ and $\mathbf{r} \sim N(0, \mathbf{K})$. Note that $\mathbf{r}$ is completely specified by the positive definite covariance matrix $\mathbf{K}$, which is block diagonal in the covariance matrices $\{\mathbf{K}_1, \mathbf{K}_2 \ldots, \mathbf{K}_M\}$ based on the assumption that the reward latent processes are uncorrelated . In practice, we use a squared exponential kernel function, written as:

$$k_m(s_i, s_j) = e^{\frac{1}{2}(s_i - s_j)^{\mathbf{T}_m}(s_i - s_j)} + \sigma_m^2 \delta(s_i, s_j),$$

where $\mathbf{T}_m = \kappa_m \mathbf{I}$ and $\mathbf{I}$ is an identity matrix. The function $\delta(s_i, s_j) = 1$, when $s_i = s_j$; otherwise $\delta(s_i, s_j) = 0$. Here s_i and s_j are feature vectors. If the feature dimension is 5 or in $\mathbf{R}^5$, the difference between two vectors is a 5-dimension vector. Under this definition the covariance is almost unity between

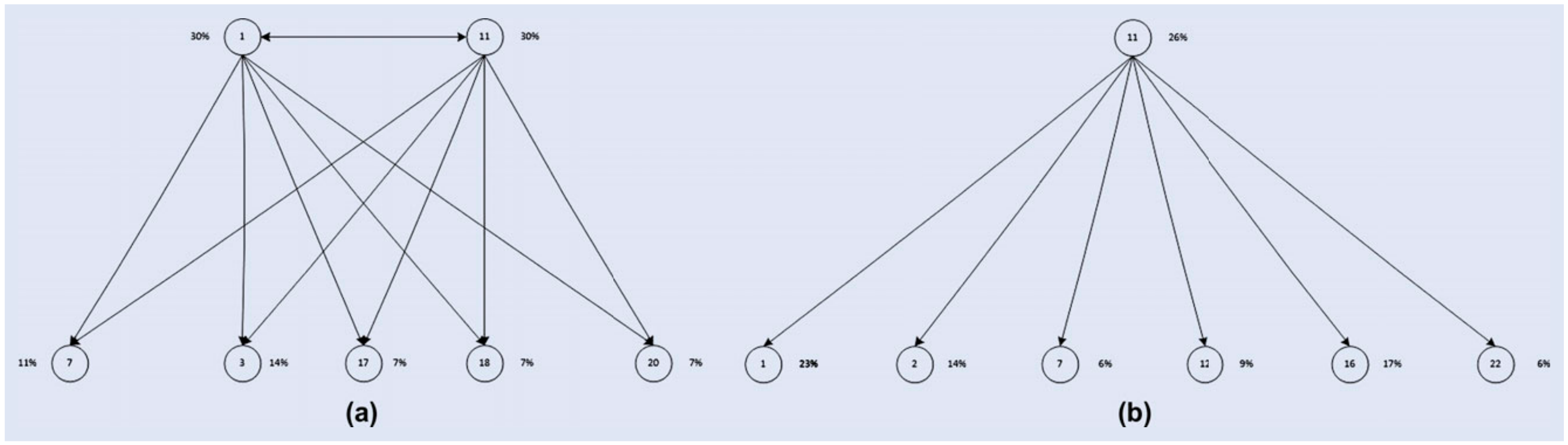

Figure 3. *Action preference graph examples:* (a). This graph shows an example action preference graph at state 158. (b). This graph shows an example action preference graph at state 14.

Table 2. Action preference graph examples.

State	Action	Frequency observed	State	Action	Frequency observed
14	1	0.23	158	1	0.30
14	2	0.14	158	3	0.07
14	7	0.06	158	7	0.11
14	11	0.26	158	11	0.30
14	12	0.09	158	17	0.07
14	16	0.17	158	18	0.07
14	22	0.06	158	20	0.07

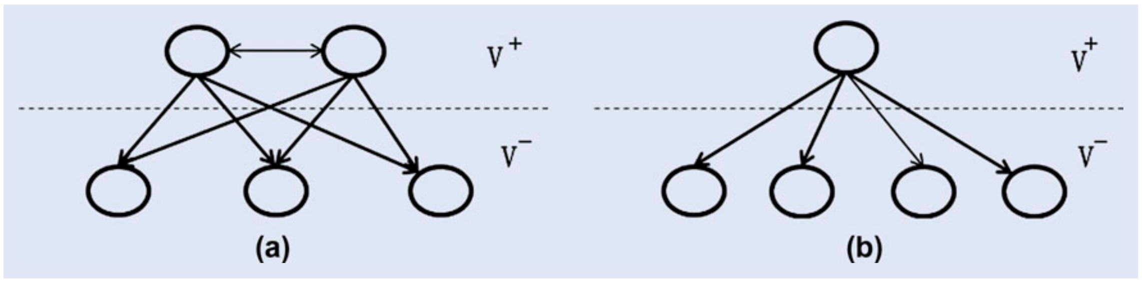

Figure 4. *Examples preference graphs:* (a) An example of observing two actions at a state. (b) An example of a unique observation at a state.

variables whose inputs are very close in the Euclidean space, and decreases as their distance increases.

Then, the GPIRL algorithm estimates the reward function by iteratively conducting the following two main steps:

(1) Get estimation of $\mathbf{r}_{MAP}$ by maximizing the posterior $p(\mathbf{r}|\mathcal{O})$, which is equal to minimize $-\log p(\mathcal{O}|\mathbf{r}) - \log p(\mathbf{r}|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the vector of hyper-parameters including κ_m and σ_m that control the Gaussian process.
(2) Optimize the hyper-parameters by using gradient decent method to maximize $\log p(\mathcal{O}|\boldsymbol{\theta}, \mathbf{r}_{MAP})$, which is the Laplace approximation of $p(\boldsymbol{\theta}|\mathcal{O})$.

4.3.3. Likelihood function and MAP optimization. GPIRL adopts the following likelihood functions to capture the strict preference and equivalent preference, respectively.

$$p((\hat{a} \succ_{s_n} \check{a})_k|\mathbf{r}) = \Phi\left(\frac{Q(s_n, \hat{a}) - Q(s_n, \check{a})}{\sqrt{2}\sigma}\right) \tag{3}$$

$$p((\hat{a} \sim_{s_n} \hat{a}')_l|\mathbf{r}) \propto e^{-\frac{1}{2}(Q(s_n,\hat{a})-Q(s_n,\hat{a}'))^2} \tag{4}$$

In equation (3), the function $\Phi(x) = \int_{-\infty}^{x} N(v|0, 1)dv$, where $N(v|0, 1)$ denotes a standard Gaussian variable.

As we stated earlier, if we model the reward functions as being contaminated with Gaussian noise that has a mean of zero and an unknown variance σ^2, we can then define the likelihood function for both the k-th strict preference relation and the l-th equivalent preference relation. Finally, we can formulate the following proposition:

Proposition 4.1 *The likelihood function, given the evidence of the observed data ($\mathcal{O}$ in the form of pairs of state and action preference graph ($\mathcal{G}$), is calculated by*

$$p(\mathcal{O}|\mathbf{r}) \propto p(\mathcal{G}|\mathcal{S}, \mathbf{r}) = \prod_{n=1}^{N} p(G_n|s_n, \mathbf{r})$$

$$= \prod_{n=1}^{N}\prod_{k=1}^{n_n} p((\hat{a} \succ_{s_n} \check{a})_k|\mathbf{r}) \prod_{l=1}^{m_n} p((\hat{a} \sim_{s_n} \hat{a}')_l|\mathbf{r}), \tag{5}$$

where n_n denotes the number of edges for strict preference and m_n means the number of edges for equivalent preference at state s_n.

In conclusion, the probabilistic IRL model is controlled by the kernel parameters κ_m and σ_m which compute the covariance matrix of reward realizations, and by σ which tunes the noise level in the likelihood function. We put these parameters into the hyper-parameter vector $\boldsymbol{\theta} = (\kappa_m, \sigma_m, \sigma)$. More often than not, we do not have prior knowledge about the hyper-parameters. And then we can apply maximum a posterior estimate to evaluate the hyper-parameters.

Essentially, we now have a hierarchical model. At the lowest level, we have reward function values encoded as a parameter vector $\mathbf{r}$. At the top level, we have hyper-parameters in $\boldsymbol{\theta}$ controlling the distribution of the parameters. Inference takes place one level at a time. At the bottom level, the posterior over function values is given by Bayes' rule:

$$p(\mathbf{r}|\mathcal{S}, \mathcal{G}, \boldsymbol{\theta}) = \frac{p(\mathcal{G}|\mathcal{S}, \boldsymbol{\theta}, \mathbf{r})p(\mathbf{r}|\mathcal{S}, \boldsymbol{\theta})}{p(\mathcal{G}|\mathcal{S}, \boldsymbol{\theta})}. \tag{6}$$

The posterior combines the prior information with the data, reflecting the updated belief about $\mathbf{r}$ after observing the decision behaviour. We can calculate the denominator in equation (6) by integrating $p(\mathcal{G}|\mathcal{S}, \boldsymbol{\theta}, \mathbf{r})$ over the function space with respect to $\mathbf{r}$, which requires a high computational capacity. Fortunately, we are able to maximize the non-normalized posterior density of $\mathbf{r}$ without calculating the normalizing denominator, as the denominator $p(\mathcal{G}|\mathcal{S}, \boldsymbol{\theta})$ is independent of the values of $\mathbf{r}$.

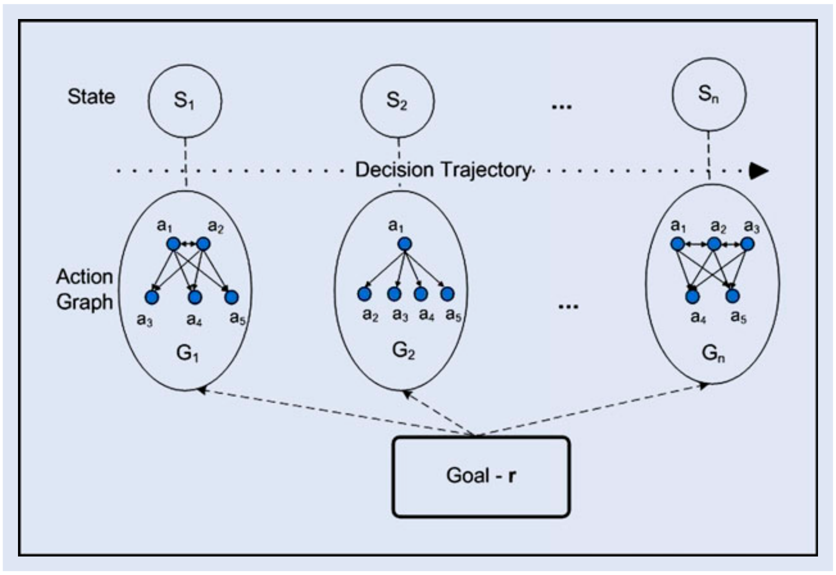

Figure 5. Observation structure for MDP.

In practice, we obtain the maximum posterior by minimizing the negative log posterior, which is written as

$$U(\mathbf{r}) \triangleq \frac{1}{2}\sum_{m=1}^{M} \mathbf{r}_m^T \mathbf{K}_m^{-1}\mathbf{r}_m - \sum_{n=1}^{N}\sum_{k=1}^{n_n} \ln \Phi\left(\frac{Q(s_n,\hat{a}) - Q(s_n,\check{a})}{\sqrt{2}\sigma}\right) + \sum_{n=1}^{N}\sum_{l=1}^{m_n} \frac{1}{2}(Q(s_n,\hat{a}) - Q(s_n,\hat{a}'))^2 \qquad (7)$$

Qiao and Beling (2011) present a proof that Proposition (7) is a convex optimization problem. At the minimum of $U(\mathbf{r})$ we have

$$\frac{\partial U(\mathbf{r})}{\partial \mathbf{r}_m} = 0 \Rightarrow \hat{\mathbf{r}}_m = K_m(\nabla \log P(\mathcal{G}|\mathcal{S},\boldsymbol{\theta},\hat{\mathbf{r}})) \qquad (8)$$

where $\hat{\mathbf{r}} = (\hat{\mathbf{r}}_1, \ldots, \hat{\mathbf{r}}_{a_m}, \ldots, \hat{\mathbf{r}}_m)$. In equation (8), we can use Newton's method to find the maximum of U with the iteration,

$$\hat{\mathbf{r}}_m^{\text{new}} = \hat{\mathbf{r}}_m - \left(\frac{\partial^2 U(\mathbf{r})}{\partial \mathbf{r}_m \partial \mathbf{r}_m}\right)^{-1} \frac{\partial U(\mathbf{r})}{\partial \mathbf{r}_m}$$

5. Experiment with the E-Mini S&P 500 equity index futures market

In this section, we conduct two experiments using the MDP model defined earlier to identify algorithmic trading strategies. We consider the six trader classes defined by Kirilenko *et al*. (2011), namely high-frequency traders, market-makers, opportunistic traders, fundamental buyers, fundamental sellers and small traders. As we argue earlier, the focus of our study will be more on HFTs and market-makers due to the large daily volume and their potential impact to the financial markets. In Kirilenko *et al*. (2011)'s paper, there are only about from 16 to 20 HFTs on the S&P500 Emini market. Although this is a small population, their impact to the market has drawn increased attention from policy-makers, regulators and academia. That is why we focus our attention on this small population. Among the roughly 10 000 trading accounts for the S&P500 Emini market, we narrow down to about 120 accounts based their high daily trading volume. In the first experiment, we select the top 10 trading accounts by their volume and end-of-the-day positions. In this we guarantee our subjects are HTFs. In the second experiment, we randomly select 10 out of the 120 accounts. This selection criterion ensures that our subjects are of either HTF or market-making strategies. With these two experimentations we show the performance of our IRL-based approach to identify the high impact population of the algorithmic trading strategies.

5.1. Trader behaviour identification

Yang *et al*. (2012) examine different trading behaviours using a linear IRL (LNIRL) algorithm with the simulated E-Mini S&P 500 market data. That MDP model contains three variables: the volume imbalance at the bid/ask prices, the volume imbalance at the 3rd best bid/ask prices and the position level. Although this MDP model is relatively simple, it is evident from the experimental results that the IRL reward space is effective in identifying trading strategies with a relatively high accuracy rate.

This paper tries to address two important issues during the modelling process to solve a realistic market strategy learning problem using real market data. The first issue is that in reality, we often do not have a complete set of observations of a trader's policies. As the market presents itself as a random process in terms of both prices and volume, it is unlikely that we will be able to capture all possible states during our observation window. In contrast, the study performed by Yang *et al*. (2012) assumes complete observation of a trader's decision policies for the simulated trading strategies. In other words, the policies simulated by a distribution can be completely captured when the simulation is run long enough. The convergence of these simulated policies and the testing results are consistent with their assumptions. However, when we use real market data to learn about trading strategies, it is necessary to address the incomplete observation problem. The second issue is to the assumption of deterministic policy versus non-deterministic policy. Yang *et al*. (2012) make a deterministic policy assumption. Under the linear feature optimization framework, non-deterministic policies can be represented by a single maximum deterministic policy. In this study, we relax the deterministic policy assumption and allow non-deterministic policies under a Gaussian process framework. As we argue earlier, Gaussian process learning allows us to infer policies even when we have a very limited set of observations. At the same time, we incorporate Gaussian preference learning into our inference process. This approach helps us to incorporate less frequently observed policies into our reward learning process. Together, the proposed GPIRL approach results in a model that relies less on observations and makes fewer assumptions on the polices we are to learn.

5.2. *Multi-class support vector machine trader classifier using GPIRL vs. LNIRL*

This section uses the support vector machine (SVM) classification method to identify traders based on reward functions that we recover from the observations of the trader's behaviours. We select a group of traders whose behaviours are consistently observed during the period we study. The primary reason for choosing the SVM classification method is its flexibility that allows us to explore feature separation in different high dimensional spaces using kernel functions. We aim to compare the performance of the two behaviour learning algorithms LNIRL and GPIRL, and to show that GPIRL perform better in real-world trading strategy identification.

We constructed 80 sample trajectories for each of the top 10 trading accounts. While there are 121 trading accounts consistently traded over the 4-week period, this study focuses on the top 10 trading accounts. We apply both the LNIRL (Ng and Russel 2000, Yang *et al*. 2012), and GPIRL (Qiao and Beling 2011) to these 800 samples. And then we apply the SVM algorithm to the 10 traders using pair-wise classification. For each pair, we first train a SVM classifier (with Gaussian kernel) with 60 randomly selected samples, and test the classification on the remaining 20 samples. We repeat the sampling 100 times and then take the average classification accuracy. We list both LNIRL classification results in table 3, and GPIRL results in table 4. On average, LNIRL gives a classification accuracy of 0.6039, while GPIRL achieves a classification accuracy of 0.9650. This result confirms our earlier assumption that GPIRL performs better when we have incomplete observations, and incorporate non-deterministic policies through Gaussian preference learning.

5.3. *Trading strategy clustering and comparison with the summary statistic-based approach*

Next, we will show that our IRL-based behaviour identification approach is far superior to the statistic-based approach. We will use the top 10 trading accounts as examples to demonstrate improvement of behaviour-based trading strategy identification achieved using the Gaussian Preference IRL model.

In the previous section, we discovered that using reward functions we can reliably identify a particular trading strategy over a period of time with a relatively high accuracy. In this section, we want to study the similarity of reward characterization among the different trading strategies. This problem can be characterized as an unstructured learning problem—clustering. We have the characterization of rewards over the state space and action space, and we aim to group trading strategies based on their similarity over the Cartesian product of the state and action spaces. We also attempt to establish connections between these trading strategy classification definitions established by Kirilenko *et al*. (2011) and our behaviour-based trading strategy clustering.

The first problem we have to address is the dimensionality of the feature space. We essentially have a reward structure over a large set of feature sets. This feature set is a product of the state space and the action space in our computational model. Fortunately, under the LNIRL algorithm, we reduce the feature space to only the state space because in this linear feature expectation optimization problem we only consider reward at a particular state. Under the deterministic policy assumption, we assume that the value function converges at a particular state. In other words, the reward function is not a function of actions. In this case, we have 243 features that must be considered during the clustering. However, under the GPIRL framework we do not assume deterministic policy, and we treat reward as a function of both states and actions. Therefore we have 243x30 features for the latter approach. We also observe that the reward matrix is relatively sparse where there are zero values at many states. To consider computational tractability and efficiency, we first examine the data structure through principal component analysis.

In the LNIRL case, the first two principal components (PCs) explain 79.78% of the data variation, and from the upper left plot in figure 6(a) we see that the first 200 PCs provide nearly 100% explanatory power. In the GPIRL case, the first two PCs only explain 38.98% of the data variation. Looking at the upper left plot in figure 6(b), we see that more PCs are needed to have better represent the data. To balance the accuracy and computational efficiency, we choose the first 200 PCs for the LNIRL and the first 400 PCs for the GPIRL case. This reduction leads to significant gain in computational efficiency and sacrifices less than 2% data variation (lower left figure in both figure 6(a) and (b)). From the upper right plots in both the LNIRL and the GPIRL spaces, we see that the first two PCs give a good representation along the first PC and that in the LNIRL case, the feature vector representation is evenly distributed between the first two PCs. The LNIRL space includes distinctly separated observations. On the other hand, the GPIRL space contains concentrations of observations, but unclear boundaries. In both cases, we would expect the PC dimension reduction technique to achieve relatively good representation of the data variation.

Now, we apply unsupervised learning method to group the trading behaviour observed on a selected group of trading accounts over the observation period. We select 10 trading accounts with the highest average daily trading volume over a period of 4 weeks (20 days) in our first experiment. We define an observation instance as a continuous period covering two hours over which we take all the activities of a particular

Table 3. Pair-wise trader classification using SVM binary classification using LNIRL.

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	0.0000	0.5437	0.5187	0.4812	0.6375	0.4812	0.5312	0.5750	0.7750	0.5937
[2,]	0.5437	0.0000	0.5250	0.5125	0.7437	0.5562	0.4937	0.4250	0.7625	0.6812
[3,]	0.5187	0.5250	0.0000	0.4687	0.6875	0.5250	0.5187	0.5250	0.7312	0.6250
[4,]	0.4812	0.5125	0.4687	0.0000	0.6937	0.5000	0.4937	0.5062	0.6562	0.6625
[5,]	0.6375	0.7437	0.6875	0.6937	0.0000	0.6625	0.7375	0.6875	0.7750	0.5437
[6,]	0.4812	0.5562	0.5250	0.5000	0.6625	0.0000	0.5500	0.5500	0.6500	0.6375
[7,]	0.5312	0.4937	0.5187	0.4937	0.7375	0.5500	0.0000	0.4937	0.8000	0.6125
[8,]	0.5750	0.4250	0.5250	0.5062	0.6875	0.5500	0.4937	0.0000	0.6437	0.6562
[9,]	0.7750	0.7625	0.7312	0.6562	0.7750	0.6500	0.8000	0.6437	0.0000	0.7437
[10,]	0.5937	0.6812	0.6250	0.6625	0.5437	0.6375	0.6125	0.6562	0.7437	0.0000

Note: The columns and rows of this table represent anonymous traders.

Table 4. Pair-wise trader classification using SVM binary classification using GPIRL.

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	0.0000	1.0000	0.9875	0.9750	0.9500	0.9750	0.9625	1.0000	0.9750	1.0000
[2,]	1.0000	0.0000	0.9750	0.9375	0.9875	0.9750	0.9625	0.9625	0.9875	1.0000
[3,]	0.9875	0.9750	0.0000	0.9750	0.9625	0.9875	1.0000	0.9750	0.9750	0.9875
[4,]	0.9750	0.9375	0.9750	0.0000	0.9750	0.9500	0.9375	0.9875	0.9875	0.9750
[5,]	0.9500	0.9875	0.9625	0.9750	0.0000	1.0000	1.0000	0.9625	0.9875	1.0000
[6,]	0.9750	0.9750	0.9875	0.9500	1.0000	0.0000	0.9625	0.8750	0.9125	0.9750
[7,]	0.9625	0.9625	1.0000	0.9375	1.0000	0.9625	0.0000	0.8625	0.9625	0.9875
[8,]	1.0000	0.9625	0.9750	0.9875	0.9625	0.8750	0.8625	0.0000	0.8000	1.0000
[9,]	0.9750	0.9875	0.9750	0.9875	0.9875	0.9125	0.9625	0.8000	0.0000	0.9625
[10,]	1.0000	1.0000	0.9875	0.9750	1.0000	0.9750	0.9875	1.0000	0.9625	0.0000

Note: The columns and rows of this table represent anonymous traders.

trader including placing new orders, modifying and cancelling existing orders and placing market orders. For each trader, we collect four observation instances on each trading date: two observation instances during the morning trading and two observation instances during the afternoon trading. The two observation periods in the morning and in the afternoon have an hour overlap time, but the observations in the morning and the afternoon do not overlap. This observation distribution is selected based on the general theory of intraday U-shaped patterns in volume - namely, that trading is heavy at the beginning and the end of the trading day and relatively light in the middle of the day (Admati and Pfleiderer 1988, Chordia *et al*. 2001, Lee *et al*. 2001, Ekman 2006). We also examined traders' actions throughout the entire trading day. We found that the two-hour observation time is a good cut-off, and with the overlapping instances in both the morning and the afternoon we expect to capture the U-shaped pattern of the market.

We then perform hierarchical clustering and generate a heat map and dendrogram of the observations in both the LNIRL reward space and the GPIRL reward space. The simplest form of matrix clustering clusters the rows and columns of a data-set using Euclidean distance metric and average linkage. For both figure 7(a) and (b), the left dendrogram shows the clustering of the observations (rows), and the top dendrogram shows the clustering of the PCs (columns). It is evident that there is a clear division of the observations (rows) in both cases. Upon closer examination, the left dendrogram contains two clusters: the top cluster and the bottom cluster with a black dividing strip in the middle of the second small cluster. We then zoomed into the small cluster and look for the sources of these observations†. In the LNIRL reward space, we find that the small cluster consists mostly of observations from trader 1 (observations numbered from 1 to 80) and trader 2 (observations numbered from 81 to 160). Observations from trader 9 (observations numbered from 641 to 720) form the black division between these two groups. In the GPIRL reward space, we find that the small cluster consists of three traders: trader 1 (observations numbered from 1 to 80), trader 2 (observations numbered from 81 to 160), and trader 5 (observations numbered from 321 to 400) with the observations from the rest of the traders lying on the other side of the divide. Moreover, we find that the observations from trader 9 (observations numbered from 641 to 720) form the black division between these two groups. These observations show that the majority of the top 10 traders form one group with 2 or 3 traders behaving a little differently. Furthermore, we observe that the clustering has less than perfect purity. In other words, individual observations from the top cluster occasionally lie in the small cluster at the bottom indicating that behaviour changes over time. The interpretation of this observation is that the HFTs may behave like opportunistic traders for a short period of time. We also occasionally

†Note: In both figure 6(a) and (b), we group observations from the same trader together in our data matrix. We have 10 traders and each has 80 observations. From the lower left graph in both (a) and (b), observations are ordered sequentially by trader IDs. For example, observations 1 through 80 come from trader 1, and observations 81 through 160 come from trader 2. This continues along the X-axis up to observations 721 through 800 from trader 10.

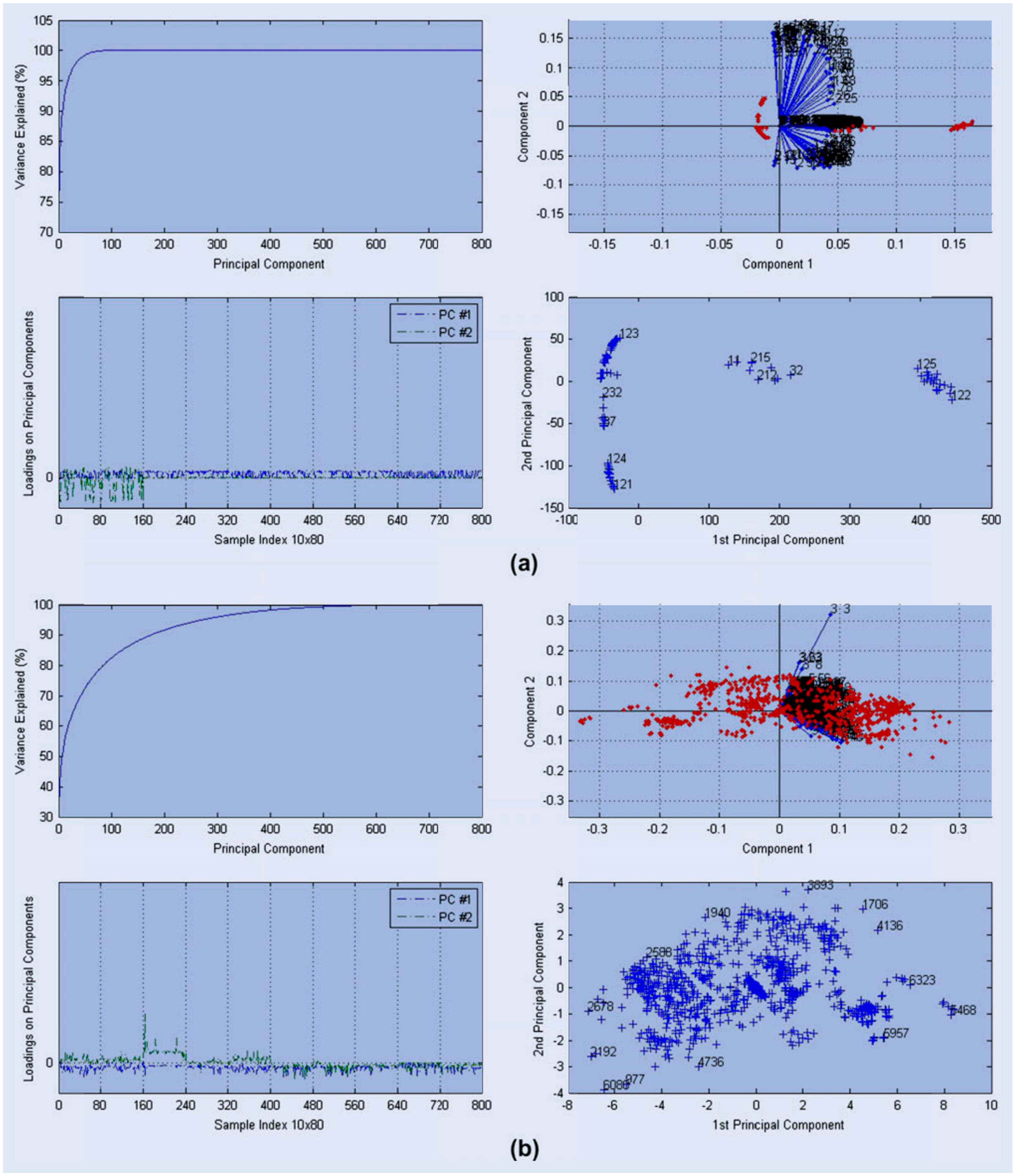

Figure 6. *Principal component representation of the reward data:* (a). Data representation under the first two PCs in the LNIRL reward space; The upper left figure shows cumulative percentage of the data variance explained by the PCs. The lower left figure plots the loadings of all the observations onto the first two PCs; The upper right figure shows the projection of the observation and feature vector onto the first two PCs. The lower right shows the projection of the observation onto the first two PCs with boundary point markers. (b) Data representation under the first two principal components in the GPIRL reward space. The upper left figure shows the cumulative percentage of the data variance explained by the PCs. The lower left figure plots the loadings of all the observations on to the first two PCs. The upper right figure shows the projection of the observation and feature vectors onto the first two PCs. The lower right figure shows the projection of the observation onto the first two PCs with boundary point markers.

observe opportunistic traders behaving like HFTs. In this case, observations cross the divide into the top cluster.

Next we propose a continuous measure of clustering using the hierarchical clustering method. We use the summary statistic-based trader classification method proposed by Kirilenko *et al.* (2011) to create reference labels. For this market data, we do not have true labels on those trading accounts. We aim to improve the labelling methods documented by

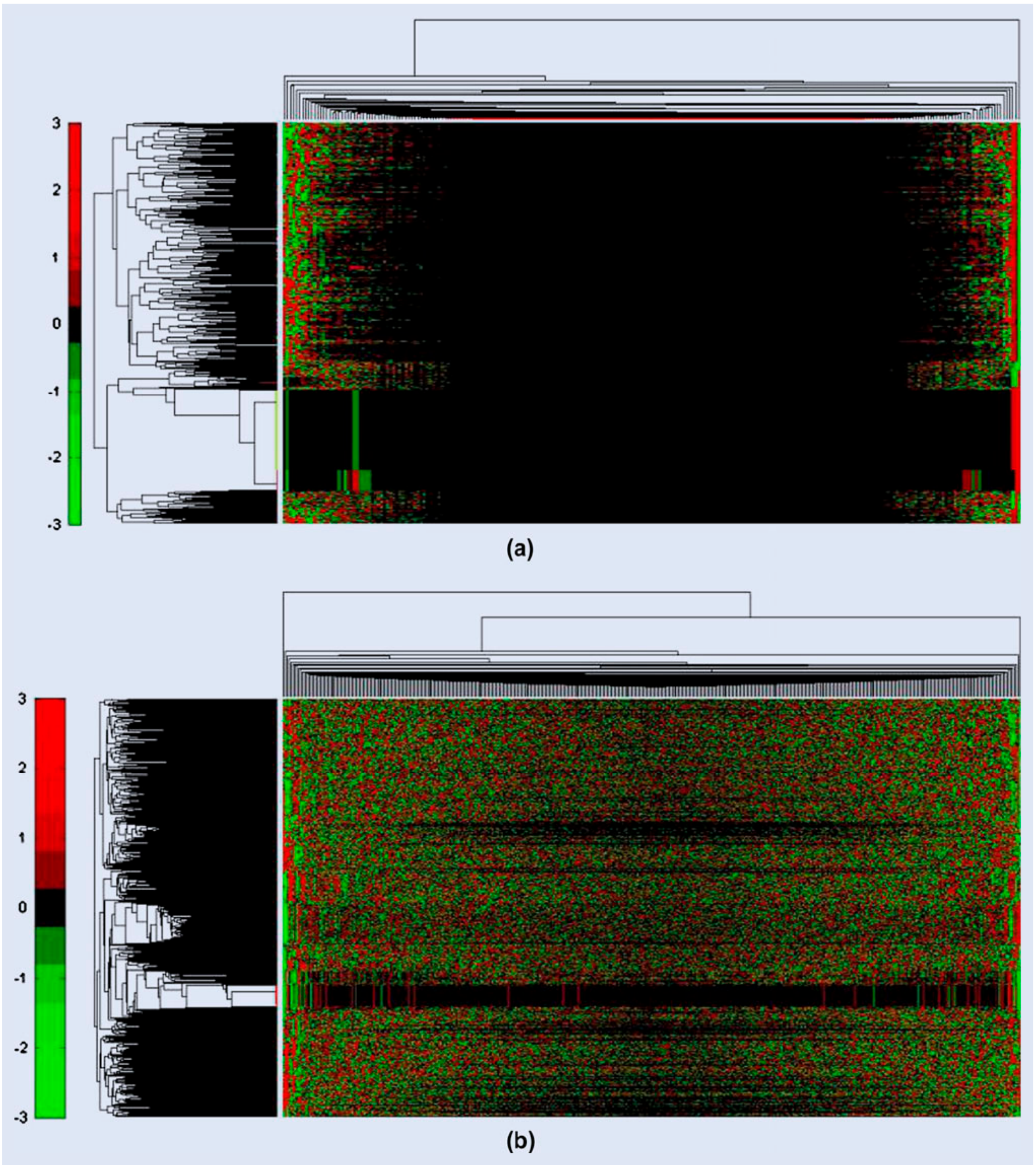

Figure 7. *Hierarchical clustering of data matrix:* (a). Heat map of 800 observations of the LNIRL rewards in the first 200 PCs. (b). Heat map of 800 observations in GPIRL rewards in the first 400 PCs.

Kirilenko et al. The motivation for creating a continuous measure of clustering is to address the potential changes in trading behaviour over time. As we mentioned earlier, we applied the summary statistic-based classification rule on the 200 observations over the 4-week period and found we can only consistently label the traders as a single type 40% of the time. We now define a weighted scoring system to evaluate both the rule-based classifier and the behaviour-based classifier. Among the 6 types of traders defined in the data section, we only concerned with labelling HFTs, intermediaries and opportunistic traders. The other three types of traders, e.g. fundamental buyers, fundamental sellers and small traders, can be reliably identified by their daily volume and their end of day positions. Here, we assign score 2 if a trader is classified as a HFT; we assign score 1 if a trader is classified as an opportunistic trader; and we assign 0 if a trader is classified as an intermediary. Labels for clustering are assigned using the majority voting rule based on the summary statistic classification rule. We then combine the scores using a weight defined as the frequency with which a particular score is assigned to

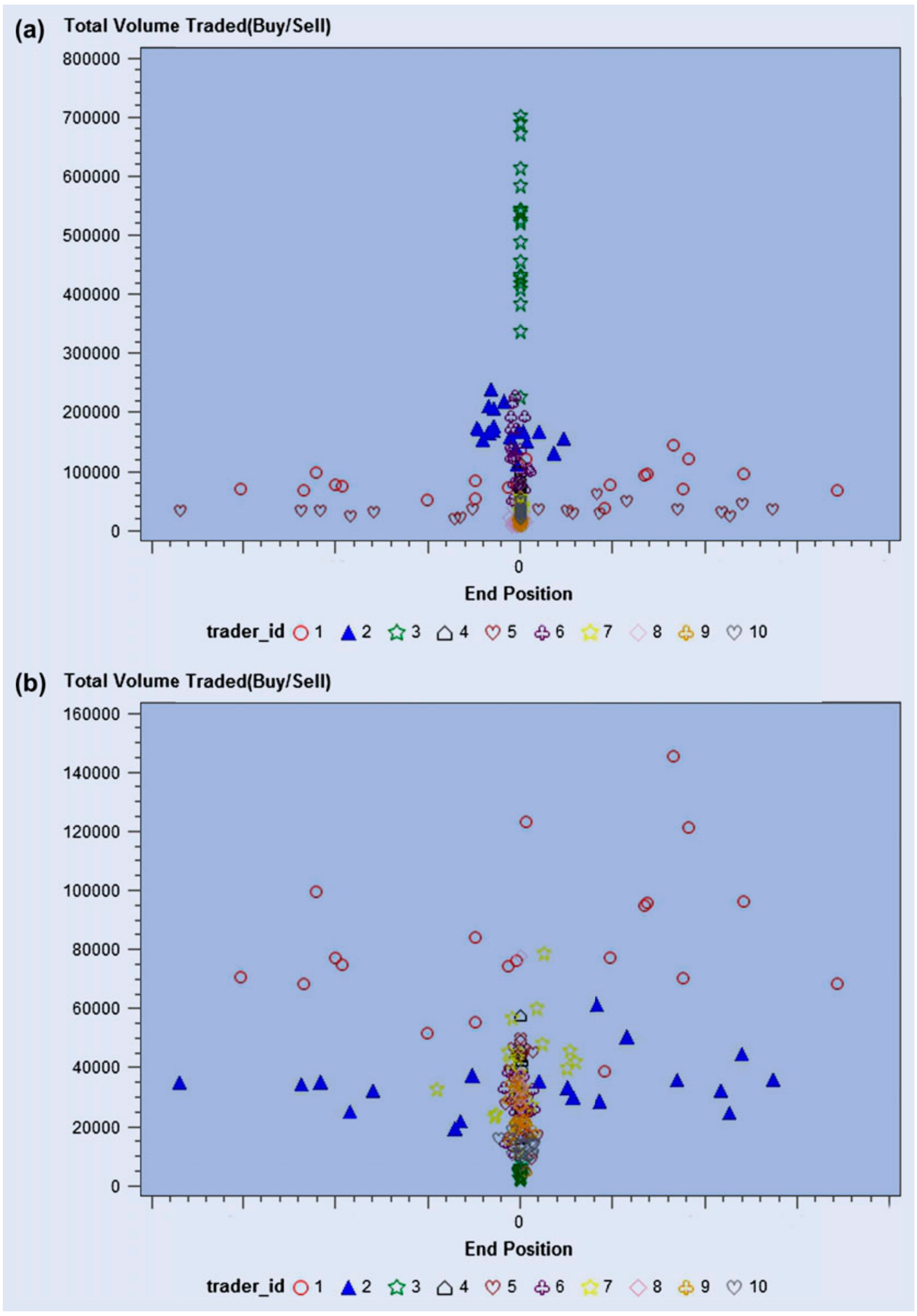

Figure 8. *Top 10 traders: trader's daily trading volume versus daily end position during a 20-day period*. (a) Randomly selected 10 traders: traders 1, 2 and 5 have varying end positions. (b) Traders 1, 2 and 7 have varying end positions.

a particular trader. Here, we want to compare the summary statistic-based trader type classification with the behaviour-based trader type classification in an effort to find connections between these two methods.

The visual representations in figure 8(a) show that trader 1 and trader 5 have a wide range of end of day positions, but their daily trading volumes remains at relatively the same levels. These traders will likely be classified sometimes as HFTs and sometimes as opportunistic traders. While trader 2 exhibits a smaller range of end of day positions than trader 1 and trader 5, the general pattern is very similar to that of traders 1 and 5, and we should classify trader 2 as an opportunistic trader. Based on this manual examination, traders 1, 2 and 5 should be classified as opportunistic traders and the rest should be classified as HFTs. Now, we compare the results of the summary statistic-based classification rule with those of the behaviour-based

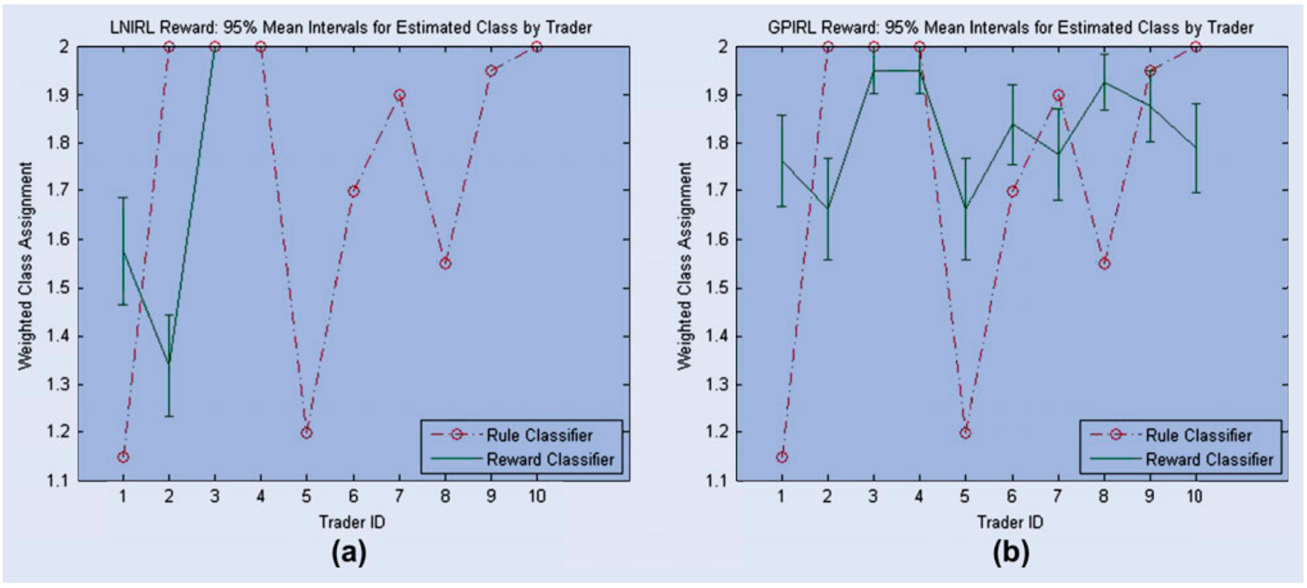

Figure 9. *Trader-type classification compared with the summary statistic-based rule classification for the top 10 traders*. (a) Hierarchical clustering in the LNIRL reward space. (b) Hierarchical clustering in the GPIRL reward space.

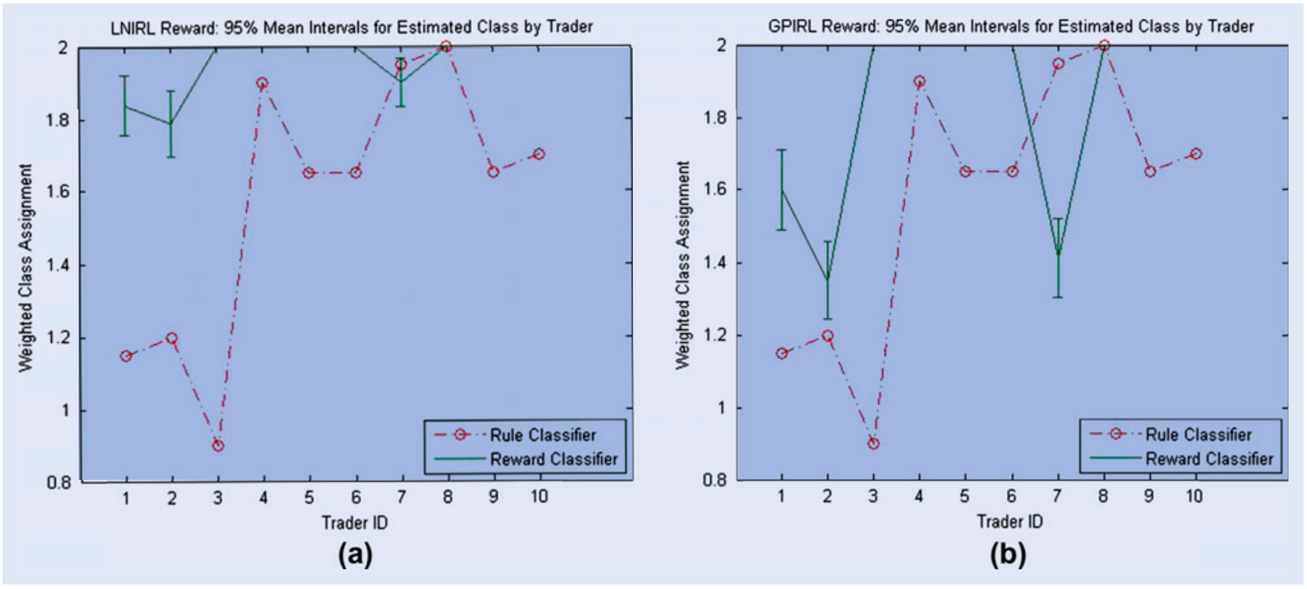

Figure 10. Trader-type classification compared with the summary statistic-based rule classification for the 10 randomly selected traders: (a) Hierarchical clustering in LNIRL reward space. (b) Hierarchical clustering in GPIRL reward space.

classification. Figure 9(a) shows that two groups of traders exist in the LNIRL reward space. Eight out of ten are identified as HFTs and only trader 1 and trader 2 are classified as opportunistic traders. This result is consistent with our observation from the dendrogram in figure 7(a). When we compare this result with the GPIRL reward space, we can locate all three traders (1, 2 and 5) that we identified through the manual process. This result is also consistent with our observation from the heat map in figure 7(b). The statistic-based classification method misclassified trader 2 because the cut-off in the statistic-based approach is based on a simple ratio between the trading volume and the end position. We can see that trader 2 has a relatively small spread of end position. However, the behaviour-based approach can identify this pattern and is able to cluster this trader with other traders with similar patterns.

We run another experiment using 10 randomly selected traders out of the traders with the top 30 trading volumes. We know this selection will only result in three types of traders, i.e. HFTs, intermediaries and opportunistic traders. We feed these 800 observations to both LNIRL and GPIRL algorithms to obtain reward representations of their trading behaviours. Based on visual examination (see figure 8(b)), we see that trader 1, trader 2 and trader 7 are opportunistic traders and the rest are HFTs. We apply the same techniques as before and we use the same cut-off scores (1.85 in the LNIRL reward space (see figure 10(a)), and 1.75 in the GPIRL reward space(see figure 10(b)). As a result, we can accurately identify the two classes of traders using the same cut-off score we used for the top 10 case (see figure 10). The classification in the LNIRL reward space gives the same result as that in the GPIRL reward space, while the statistic-based classification method misclassified trader 3 as an opportunistic trader. Based on the daily end position, daily total trading volume and inventory variance, trader 3 should be classified as a HFT. Again, this misclassification is due to the aggregate cut-off ratio. However, the behaviour-based approach can identify this pattern and is able to cluster trader 3 with other traders with similar behavioural patterns. Overall, we argue that the GPIRL reward space score-based classification rule provides an advantage over the summary statistic-based approach in that it is based on similarity in behaviour and it can be clearly interpreted. Because the GPIRL rule a better reflects traders' choices of actions under different market conditions than the summary statistics, it is well suited for the discovery of new behavioural patterns of the market participants. We also conclude that the GPIRL reward space is more informative and is a superior measure of trading behaviour in terms of the LNIRL reward space.

6. Conclusion

We assume incomplete observation of algorithmic trading strategies and model traders' reward functions as a Gaussian process. We also incorporate traders' action preferences under different market conditions through preference graph learning. The aim of this study is to quantify trader behaviour-based on IRL reward learning under a Bayesian inference framework. We apply both a linear approach (a linear combination of known features) (Abbeel and Ng 2004) and GPIRL (Qiao and Beling 2011) to a real market data-set (The E-Mini S&P 500 Futures), and we conclude that GPIRL is superior to the LNIRL methods, with a 36% greater rate of identification accuracy. Furthermore, we establish a connection between the summary statistic-based classification (Kirilenko *et al.* 2011) and our behaviour-based classification. We propose a score-based method to classify trader types, and because of the transferable property of the reward structure the cut-off score for classifying a group of traders can be applied to different market conditions.

The implication of this research is that reward/utility-based trading behaviour identification can be applied to real market data to accurately identify specific trading strategies. As documented by Abbeel et al. (Abbeel and Ng 2004) and confirmed by many other researchers, the reward function is the most succinct, robust, and transferable definition of a control task. Therefore, the behaviour learned using the reward space has much broader applicability than observed policies. Furthermore, these learned reward functions will allow us to replicate a particular trading behaviour in a different environment to understand their impact on the market price movement and market quality in general.

We also want to note some future research on improving the identification accuracy and discuss applications of this behavioural characterization:

- During our preference learning inference phase, we only considered a simple two-layer preference graph. However, traders' preferences can be further distinguished with multi-layer graphs or other preference learning techniques.
- Our study focused on the sampled algorithmic traders on a market. Future studies can extend these results to a large scale experiment to include market participants (specifically opportunistic traders), and study their behavioural similarity through clustering. We can then associate the group behaviour with market quality measures.
- Under the GPIRL framework, we are able to recover a detailed reward structure. These reward functions can be used to generate new policies under a simulated market condition to understand the full behaviour of certain trading strategies. This framework provides a particularly interesting way for market regulators to see how the various trading strategies will interact during stressed market conditions.

References

Abbeel, P., Coates, A. and Ng, A.Y., Autonomous helicopter aerobatics through apprenticeship learning. *Int. J. Robot. Res.* 2010, **29**, 1608–1639.

Abbeel, P. and Ng, A.Y., Apprenticeship learning via inverse reinforcement learning, Proceedings of the twenty-first international conference on Machine Learning, Banff, AB, Canada, 2004.

Admati, A.R. and Pfleiderer, P., A theory of intraday patterns: Volume and price variability. *Rev. Financ. Stud.*, 1988, **1**, 3–40.

Aldridge, I., *A Practical Guide to Algorithmic Strategies and Trading Systems – High Frequency Trading*, 2010 (John Wiley & Sons: New York).

Bellman, R., *Dynamic Programming*, 1957 (Princeton University Press: Princeton, NJ).

Benos, E. and Sagade, S., High-frequency trading behavior and its impact on market quality: Evidence from the UK equity market Bank of England. Working Paper No. 469, 2012.

Bertsekas, D.P., *Neuro-dynamic Programming*, 2007 (Athena Scientific: Belmont, MA).

Biais, B., Foucault, T. and Moinas, S., Equilibrium high-frequency trading. SSRN Working Paper, 2012.

Boehmer, E., Fong, K. and Wu, J., International evidence on algorithmic trading. EDHEC Working Paper, 2012.

Boularias, A. and Chaib, B., Bootstrapping apprenticeship learning. Advances in Neural Information Processing Systems Conference, Vancouver, BC, Canada, 2010.

Brogaard, J.A., High frequency trading and its impact on market quality. PhD Dissertation, Kellogg School of Management, Northwestern University, 2010.

Brogaard, J., High frequency trading and market quality. SSRN Working Paper, 2012.

Chordia, T., Roll, R. and Subrahmanyam, A., Market liquidity and trading activity. *J. Finance*, 2001, **56**, 501–530.

Daw, N.D., Niv, Y. and Dayan, P., Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nat. Neurosci.*, 2005, **8**, 1704–1711.

Dvijotham, K. and Todorov, E., Inverse optimal control with linearly-solvable MDPs. Proceedings of 27th International Conference on Machine Learning, ACM, Haifa, Israel, 2004.

Easley, D., Lopez, M.M. and O'Hara, M., The microstructure of the "Flash crash". Working Paper, Cornell University, 2010.

Ekman, P., Intraday patterns in the S&P 500 index futures market. *J. Futures Markets*, 2004, **12**, 365–381.

Gabaix, X., Gopikrishnan, P., Plerou, V. and Stanley, H.E., A theory of power-law distributions in financial market fluctuations. *Nature*, 2003, **423**, 267–270.

Gai, J., Yao, C. and Ye, M., The externalities of high-frequency trading. Working Paper, University of Illinois, 2012.

Gatheral, J., No-dynamic-arbitrage and market impact. *Quant. Finance*, 2010, **10**, 749–759.

Hasbrouck, J., Measuring the information content of stock trades. *J. Finance*, 1991, **46**, 179–207.

Hasbrouchk, J. and Seppi, D.J., Common factors in prices, order flows and liquidity. *J. Financ. Econ.*, 2001, **59**, 383–411.

Hayes, R., Paddrik, M.E., Todd, A., Yang, S.Y., Scherer, W. and Beling, P., Agent based model of the E-Mini future market: Applied to policy decisions. Proceedings of 2012 Winter Simulation Conference, Berlin, Germany, 2012.

Hendershott, T., Jones, C.M. and Menkveld, A.J., Does algorithmic trading improve liquidity? *J. Finance*, 2004, **66**, 1–33.

Hendershott, T. and Riordan, R., Algorithmic trading and the market for liquidity. *J. Financ. Quant. Anal.*, 2013, **48**, 1001–1024.

Jones, C.M., What do we know about high-frequency trading? SSRN Working Paper, 2012.

Jones, C.M., Kaul, G. and Lipson, M.L., Transactions, volume, and volatility. *Rev. Financ. Stud.*, 1994, **7**, 631–651.

Jovanovic, B. and Menkveld, A.J., Middlemen in limit-order markets. NYU Working Paper, New York, 2010.

Kirilenko, A., Kyle, A.S., Samadi, M. and Tuzun, T., The flash crash: The impact of high frequency trading on an electronic market. SSR Working Paper Series, 2011.

Lee, Y.T., Fox, C.R. and Liu, Y., Explaining intraday pattern of trading volume from the order flow data. *J. Bus. Finance Account.*, 2001, **28**, 306–686.

Lee, S.J. and Zoran, P., Learning behavior styles with inverse reinforcement learning. ACM SIGGRAPH 2010 papers, pp. 1–7, 2010.

Levine, S., Popovic, Z. and Koltun, V., Feature construction for inverse reinforcement learning. *Adv. Neural Inform. Process. Syst.*, 2010, **24**(6), 7–14.

Martinez, V.H. and Rosu, I., High frequency traders, news and volatility. SSRN Working Paper, 2012.

Menkveld, A., High frequency trading and the new market makers. SSRN Working Paper, 2012.

Moallemi, C.C. and Saglam, M., The cost of latency. SSRN Working Paper, 2012.

Neu, G. and Szepesvari, C., Apprenticeship learning using inverse reinforcement learning and gradient methods. Proceedings of Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, 2007.

Ng, A.Y. and Russel, S., Algorithms for inverse reinforcement learning. Proceedings of ICML, Stanford, CA, USA, 2000.

Paddrik, M.E., Hayes, R., Todd, A., Yang, S.Y., Scherer, W. and Beling, P., An agent based model of the E-Mini S&P 500 and the flash crash. Proceedings of IEEE Computational Intelligence in Financial Engineering and Economics, New York, USA, 2012.

Pagnotta, E. and Philippon, T., Competing on speed. SSRN Working Paper, 2012.

Qiao, Q. and Beling, P., Inverse reinforcement learning with Gaussian process. Proceedings of 2011 American Control Conference, San Francisco, CA, USA, 2011.

Ramachandran, D. and Amir, E., Bayesian inverse reinforcement learning. Proceedings of IJCAI, Hyderabad, India, 2007.

Ratliff, N., Ziebart, B., Peterson, K., Bagnell, J.A., Hebert, M. Dey, A.K. and Srinivasa, S., Inverse optimal heuristic control for imitation learning. Proceedings of AISTATS, St. Petersburg, FL, USA, 2009.

Riordan, R. and Storkenmaier, A., Latency, liquidity and price discovery. *J. Financ. Market*, 2012, **15**, 416–437.

Sharpe, W.F., Mutual fund performance. *J. Bus.*, 1966, **39**, 119–138.

Sutton, R.S. and Barto, A.G., *Reinforcement Learning: An Introduction*, 1998 (The MIT Press: Cambridge, MA).

Syed, U., Bowling, M. and Schapire, R.E., Apprenticeship learning using linear programming. Proceedings of 25th International Conference on Machine Learning, Helsinki, Finland, 2008.

Syed, U. and Schapire, R.E., A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems*, edited by J. Platt, D. Koller, Y. Singer, and S. Roweis, pp. 1449–1456, 2007 (MIT Press: Cambridge, MA).

Yang, S.Y., Paddrik, M.E., Hayes, R.J., Todd, A., Kirilenko, A.A. Beling, P. and Scherer, W., Behavior based learning in identifying high frequency trading strategies. Proceedings of IEEE Computational Intelligence in Financial Engineering and Economics, New York, USA, 2012.

Ziebart, B.D., Mass, A., Bagnell, J.A. and Dey, A.K., Maximum entropy inverse reinforcement learning. Proceedings of the Twenty-third AAAI on Artificial Intelligence, Chicago, IL, USA, 2008.