POLITECNICO DI TORINO
Repository ISTITUZIONALE

Discovering profitable stocks for intraday trading

Original
Discovering profitable stocks for intraday trading / Baralis, ELENA MARIA; Cagliero, Luca; Cerquitelli, Tania; Garza, Paolo; Pulvirenti, Fabio. - In: INFORMATION SCIENCES. - ISSN 0020-0255. - 405:(2017), pp. 91-106. [10.1016/j.ins.2017.04.013]

Availability:
This version is available at: 11583/2673213 since: 2021-03-30T14:22:17Z

Publisher:
Elsevier Inc.

Published
DOI:10.1016/j.ins.2017.04.013

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
Elsevier postprint/Author's Accepted Manuscript

© 2017. This manuscript version is made available under the CC-BY-NC-ND 4.0 license
http://creativecommons.org/licenses/by-nc-nd/4.0/.The final authenticated version is available online at:
http://dx.doi.org/10.1016/j.ins.2017.04.013

(Article begins on next page)

24 April 2024

Discovering profitable stocks for intraday trading

Elena Baralis, Luca Cagliero*, Tania Cerquitelli, Paolo Garza, Fabio Pulvirenti

Dipartimento di Automatica e Informatica, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129, Torino, Italy

Abstract

Intraday traders buy and sell financial instruments in the short term, typically within the same trading day. Stocks are notable examples of financial instruments. However, since hundreds of stocks are listed on the stock exchange selecting on each trading day the most tradeable stocks is a challenging task, which is commonly addressed through manual inspection of historical stock prices and technical indicators. This paper aims at discovering tradeable stocks on a given trading day by analyzing the historical prices assumed by the same stocks or by other ones on the preceding days by means of regression and weighted sequence mining techniques. The use of regression and weighted sequence mining techniques allows traders to automatically consider a potentially large number of candidate stocks and to effectively analyze their price variations across consecutive days. The experimental results, which were achieved on data acquired from different markets

*Corresponding author. Tel.: +39 011 090 7179. Fax: +39 011 090 7099.

Email addresses: elena.baralis@polito.it (Elena Baralis), luca.cagliero@polito.it (Luca Cagliero), tania.cerquitelli@polito.it (Tania Cerquitelli), paolo.garza@polito.it (Paolo Garza), fabio.pulvirenti@polito.it (Fabio Pulvirenti)

and under different market conditions, show that sequence mining algorithms yield profits higher than both regression techniques and naive strategies.
Keywords: Stock Data Mining, Sequence Mining, Weighted Data

1. Introduction

Trading is the process by which private investors or traders buy and sell financial instruments over an electronic network, typically through online trading platforms. Intraday trading is one of the mostly used trading strategies. It entails buying and selling financial instruments in the short term, typically within the same trading day, to make money from fast stock price variations. Stocks are among the most popular financial instruments. Hundreds of stocks are listed on financial markets spread all over the world. To choose the right stock to trade, financial analysts commonly perform fundamental or technical analyses [32]. Fundamental analyses entail the study of the overall state of the underlying company, while technical analyses focus on the variations of stock prices, which are assumed to reflect all external influences. This work relies on technical analyses.

Since stock prices vary over time, traders constantly monitor them to forecast future market trends. On each trading day, intraday traders usually trade a limited number of stocks. Typically, they buy and sell the same stock multiple times per day. However, choosing day by day the most profitable stock to trade is challenging, because (i) a large number of stocks is listed on the stock exchange, and (ii) the manual inspection of financial data is time consuming and, sometimes, practically unfeasible.

In recent years, many automatic intraday trading systems (e.g. [12, 14])

have been proposed in literature. Most of them focus on automatically generating trading signals (e.g. buy stock X if its price decreases by 1%) for a selection of stocks that are deemed as particularly convenient to trade by domain experts. However, deciding day by day what stocks are more convenient to trade is a parallel yet appealing research problem.

To recommend the right stock to trade on the next day, we aim at considering not only the price of the same stock on the days before, but also those of the other stocks. For example, the today's price variations of stocks X and Y may influence the price variation of stock Z on the next day. To this purpose, we apply two orthogonal data mining techniques [15], i.e., regression and sequence mining, on historical stock data. Note that data mining approaches, such as regression and sequence-based algorithms, analyze historical data to discover patterns that are likely to hold in the near future. Although these techniques are incapable of detecting anomalous behaviors, this limitation is less critical in the context of intraday trading, because the discovered trends are expected to hold just in the short term (e.g. in the following hours).

Regression algorithms predict the value of continuous variables based on the analysis of historical data. In our context, the goal is to predict the daily percentage variation of the price of a stock on the next trading day by analyzing the historical prices assumed by the market stocks on the preceding days. The stocks with maximal predicted variation are recommended as most tradeable stocks on the subsequent trading day.

Sequence mining is an unsupervised data mining technique to discover recurrent sequences of items in large datasets [27]. A sequence is an ordered list

of itemsets, where an itemset is a set of items occurring at a given time stamp. In our context of analysis, items are stocks, while time stamps correspond to the closures of consecutive trading days. Given the k best-performing stocks on the past and the current trading days, a sequence indicates that if an arbitrary set of stocks is in the top-k list on the preceding days then a given stock is likely to occur in the top-k list on the next day. To overcome the limitations of traditional sequence mining approaches, the concept of weighted sequence is introduced and tailored to the problem under analysis. Specifically, weighted sequences are exploited, instead of traditional ones, to weigh differently the occurrences of different stocks on the same trading day according to their daily profits. The most recurrent and profitable sequences of stocks are considered to perform stock recommendation.

Regression and sequence mining algorithms allow traders to automate historical stock data exploration and, thus, to scale the analyses towards large stock markets. Furthermore, they inherently consider the influence between multiple stock price variations on consecutive days.

To demonstrate the effectiveness of the proposed approach, we conducted a campaign of experiments on market data acquired from three different markets, i.e., the U.S. NASDAQ-100 and Dow Jones indices and the Italian FTSE MIB index. Since the experimental results can be affected by the conditions of the underlying market, we considered both favorable and unfavorable yearly time periods for all markets. The experimental results demonstrate that the weighted sequence-based strategy allow investors to yield returns higher than regression-based and naive approaches.

The paper is organized as follows. Section 2 compares our approach

with most relevant related works. Section 3 thoroughly describes the newly proposed BEst STock FInder (BeSt) system. Section 4 describes the experiments and summarizes the achieved results, Section 5 compares the strategies used to perform stock recommendation in light of the achieved results, while Section 6 draws conclusions and presents the future developments of this work.

2. Related work

Financial data mining entails the application of data mining techniques to data acquired from financial markets. Since market trends that has happened in the past can give investors/traders an idea of what will happen in the future, one of the main goals of financial data mining is to propose new investing strategies. Under this umbrella, two main research directions can be identified: (i) Plan long-term investment strategies based on the analysis of historical data. (ii) Forecast short-term market trends. Hereafter, we separately overview the main long- and short-term data mining strategies proposed in literature. For the sake of clarity, in Table 1 the presented approaches are classified according to (a) the time horizon of the considered investments (long-term, short-term, or hybrid), (b) the technique used to analyze data (e.g. regression, time series, sequence mining, neural networks), and (c) the output type, i.e., intraday trading signals (e.g. buy stock X at price Y) or other kinds of recommendations (e.g. consider stocks X and Y in your future trades).

(i) Long-term investing strategies. Many attempts to propose new long-term investment strategies based on the analysis of historical financial data

have been made. For example, in [13] the authors proposed to use a matrix factorization strategy to plan diversified stock portfolios for long-term investments. In [4] a Genetic Network Programming algorithm is exploited to build a stock portfolio. Portfolios are modeled as undirected graphs and they are evaluated using parametric risk functions. Portfolios are planned for long-term investments, but stocks may be sold/bought on a daily basis according to the algorithm predictions. In [22] genetic algorithms have been adapted to diversify investments in stock portfolios across multiple assets.

A parallel issue concerns wealth allocation among stocks in the portfolio. The authors in [19] proposed a solution oriented to long-term investments and based on the analysis of the statistical correlations between market windows of all stocks in the historical stock market. Our approach differs from [4, 13, 19, 22], because it recommends stocks for intraday trading (short-term investments) rather than generating long-term stock portfolios.

(ii) Short-term investing strategies. The approach presented in [36] generated trading signals for intraday trading (e.g., buy stock X at price Y) by applying regression techniques on market data consisting of various technical indicators and statistical measures. In [28] the authors used the popular K-Nearest Neighbour classifier to associate stocks with similar behavior in order to generate trading signals, while in [7] the authors leveraged pattern mining to recognize turning points in stock price variations (i.e., the *bull-flag patterns*). Similarly, the authors in [14] applied neural network and decision tree models to address the same task. Unlike [36], they analyzed not only statistics about historical stock prices but also on the content of recently published textual news. The task of considering financial news or social data to

drive stock investments, commonly denoted as media-aware trading [18, 20], is out of the scope of this paper.

Neural Networks revealed to be very spread in stocks market prediction. For example, the works presented in [9, 21, 30] exploited the learning ability of neural networks to separately analyze the series of closing prices of different stocks. Conversely, in this work we focus on studying the correlation between the stock prices of multiple stocks. A common strategy to perform accurate stock trend prediction is to combine multiple data mining models together. For example, in [17], the authors combined Neural Networks or Support Vector Machines to predict stock price variations, in [24] a learning ensemble composed of several Neural Networks, optimized with a particle-swarm approach, is proposed, while in [8] and [12] the authors exploited genetic algorithms and rule-based reasoning techniques, respectively to generate trading signals suitable for short-term stock/forex investments. An ensemble of decision tree-based regression models is exploited in [11]. The authors proposed to use decision trees because of their inherent simplicity and interpretability, which allow domain experts to explore the generated models. A similar strategy, coupling Support Vector Regression with Neural Networks has been proposed in [23] to generate trading signals. Unlike all the aforesaid short-term investment strategies, the system presented in this paper analyzes historical stock data to choose the stocks that are worth trading on the next trading day. Hence, our system generates recommendations like *monitor the price movements of stocks X and Y because they are likely to vary significantly on the next trading day* rather than producing trading signals. Hence, our approach is complementary to existing strategies and can

Table 1: Selection of investing strategies.

Approach	Time horizon	Technique	Output type
BEST	**Short-term**	**Weighted Sequence mining & Regression**	**Recommended stocks**
[4]	Hybrid	Genetic Network Programming	Diversified stock portfolios
[7]	Short-term	Pattern mining	Trading signals
[8]	Short-term	Genetic Network Programming	Stock portfolios
[9]	Short-term	Particle Swarm Artificial Neural Networks	Trading signals
[11]	Short-term	Ensemble of Dec. Trees	Trading signals
[12]	Short-term	Rule-based reasoning	Trading signals
[13]	Long-term	Matrix Factorization & Clustering	Diversified stock portfolios
[14]	Short-term	Neural Networks + Dec. Tree + Logistic Regr.	Trading signals
[17]	Short-term	Support Vector Machines and Neural Networks	Single T. Series Forecasting (increasing or decreasing)
[19]	Long-term	Correlation analyses	Wealth allocations
[21]	Short-term	Multi Layer Perceptron Neural Networks	Single T. series forecasting
[22]	Long-term	Genetic Algorithm	Diversified stock portfolios
[23]	Short-term	SVR, Artificial Neural networks and R. Forests	Trading signals
[24]	Short-term	Particle Swarm Ensemble Neural Networks	Single T. Series Forecasting
[28]	Hybrid	K-NN Classifier	Trading signals
[29]	Short-term	Bayesian Neural Network	Single T. Series Forecasting
[30]	Short-term	Neural networks	Single T. series forecasting
[36]	Short-term	Support Vector Machines	Trading signals

be applied upstream to select the stocks for which trading signals are worth generating. Furthermore, to the best of our knowledge, it is the first attempt to apply sequence mining algorithms to stock market data.

3. Best Stock Finder

Best Stock Finder (BeSt) is a new data mining engine targeted to stock data analysis. It analyzes historical stock data to detect the most appealing stocks to consider for intraday trading.

The main components of the BeSt framework, summarized in Figure 1, are the following ones:

- **Stock data retrieval and preparation**, which entails stock data gathering, through the Yahoo! Finance APIs [33], and data preparation

for the subsequent analyses (see Section 3.1).

- **Model generation**, which performs regression or sequence mining analyses on the prepared data (see Section 3.2).

- **Stock recommendation**, which recommends to intraday traders the most appealing stocks to trade on the next trading day, according to the models generated from historical stock data (see Section 3.3).

A more thorough description of each block is given below.

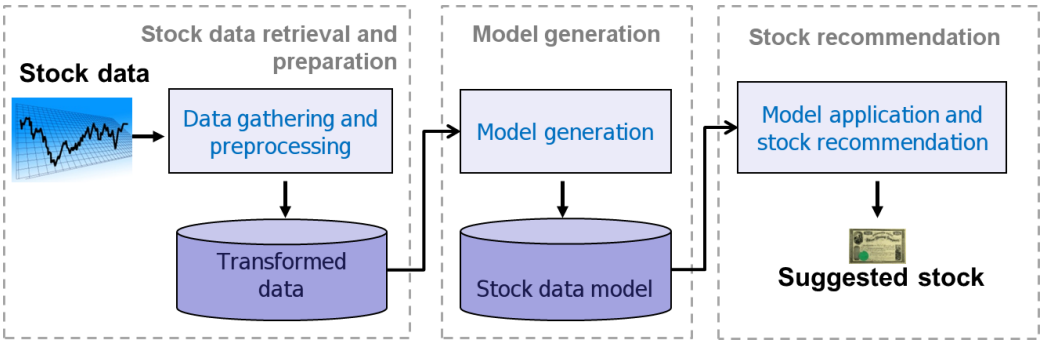


Figure 1: Best Stock Finder.

3.1. Stock data retrieval and preparation

This block acquires historical stock data from financial markets. Given a time period T, a sampling frequency f, and a set S of market stocks, the BeSt system acquires the historical prices of all the stocks in S sampled at frequency f within time period T through the Yahoo! Finance APIs (https://it.finance.yahoo.com/). Specifically, to acquire data we performed several API requests. A Yahoo! Finance API request consists of (i) the list of considered stocks, (ii) the measure of interest (e.g. the opening stock

price, the closing price, the daily exchange volume), (iii) the starting and ending dates of the considered time period, and (iv) the sampling frequency (i.e., daily, weekly, or monthly). Each request produces a different stock dataset, which consists of all the measure values corresponding to the selected stocks sampled at the desired frequency within the considered time period. Although the proposed approach is general and can be applied at different time granularities, hereafter we will consider the daily closing prices on each trading day. The current version of the data crawler relying on the Yahoo! Finance APIs is written in Java and it is freely available for research purposes upon request to the authors.

Stock prices are modeled as items occurring in weighted datasets. Weighted items and datasets were first introduced in [31]. In the context of stock data, items are stocks, while indicator values weigh stock occurrences at each time stamp. We tailored the concept of weighted item to our context of analysis as follows.

Definition 1. Weighted item. *Let T be a time period and $t_k \in T$ a time stamp. Let $s_j \in S$ be an arbitrary stock, and r_k^j the daily return of stock s_j at time stamp t_k (i.e., the difference between the prices of stock s_j at time stamps t_k and t_{k-1}, respectively). The following definitions hold. (i) Every stock $s_j \in S$ is an item. (ii) The pair $\langle s_j, r_k^j \rangle$ is a weighted item, where s_j is an arbitrary stock, while r_k^j is the weight associated with stock s_j at time stamp t_k.*

To perform sequence mining and regression analyses on historical stock data we adopted two complementary data representations: (i) a weighted

sequential data model, to apply sequence mining, and (ii) a structured data model, to apply regression. We will introduce the two data representations below.

Weighted sequential data. Sequences are extracted from sequential datasets. A sequential dataset is a set of input-sequences, each one corresponding to a distinct time window. Time windows are finite sets of time stamps. A reformulation of the concept of input-sequence given in [35] tailored to our context of analysis follows.

Definition 2. Input-sequence. *Let $tw_q \in T$ be an arbitrary time window. Let I be a set of weighted items. An input-sequence $S(tw_q)$ is an ordered sequence $S_i \rightarrow S_j \rightarrow \ldots \rightarrow S_z$, where S_i, S_j, ..., S_z are sets of weighted items in I. Sets S_i, S_j, ..., S_z are, in general, not disjoint. The arrows indicate the temporal order of occurrence of the sets of weighted items within the same time window tw_q, i.e., $t_i \leq t_j \leq \ldots \leq t_z$, $t_i, t_j, \ldots, t_z \in tw_q$.*

Definition 3. Weighted sequential stock dataset. *Let T be a time period and $tw_1, tw_2, \ldots, tw_n$ be a set of time windows in T. A weighted sequential stock dataset D is a set of input-sequences, each one related to a different time window $tw_q \in T$, $1 \leq q \leq n$.*

A sequence is an ordered list of itemsets (i.e., sets of items) occurring within the same time window at consecutive time stamps. A formal definition of sequence can be found in [35]. In our context, it represents an ordered list of stock sets occurring at consecutive time stamps within the same time window. The ordered list indicates the temporal sequence of occurrence of

Table 2: Example of weighted sequential stock dataset

Time period	**Time stamp**	**Input-sequence**
Time window tw_1	t_1	$\langle a, 5\% \rangle$ $\langle b, 5\% \rangle$ $\langle c, 7\% \rangle$
	t_2	$\langle a, 2\% \rangle$ $\langle b, 6\% \rangle$ $\langle c, 7\% \rangle$
Time window tw_2	t_2	$\langle a, 2\% \rangle$ $\langle b, 6\% \rangle$ $\langle c, 7\% \rangle$
	t_3	$\langle a, 1.5\% \rangle$ $\langle b, 7\% \rangle$ $\langle c, 2\% \rangle$
Time window tw_3	t_3	$\langle a, 1.5\% \rangle$ $\langle b, 7\% \rangle$ $\langle c, 2\% \rangle$
	t_4	$\langle a, 0.5\% \rangle$ $\langle b, 7\% \rangle$ $\langle c, 1\% \rangle$

the stock sets. In our context, each time window corresponds to a different pair of consecutive trading days in T and the time stamps associated with the weighted items in the input-sequences correspond to the closing times of the corresponding trading days. To focus the analyst's attention on the best-performing stocks, we consider at each time stamp only the top-k stocks in order of decreasing daily relative return. A sliding window is used to store the returns of the best-performing stocks within consecutive time windows. Such data representation is commonly used for sequence mining extractions [27].

For example, Table 2 reports an example of weighted sequential stock dataset consisting of four trading days (time stamps t_1-t_4). Pairs of consecutive trading days are grouped into three different time windows (tw_1-tw_3). Each input-sequence is associated with a unique pair of consecutive time stamps t_{k-1} and t_k. For the sake of simplicity, let us assume that stocks a, b, and c are the three best-performing stocks, in terms of average daily profit, on all the trading days. For each trading day the daily relative returns of the considered stocks are reported. For instance, stock a made a daily profit equal to 5% on the trading day corresponding to time stamp t_1.

Structured data. A structured datasets consists of a set of records, where

Table 3: Example of structured stock dataset

Time stamp	**Stocks**		
	a	b	c
t_1	5%	5%	7%
t_2	2%	6%	7%
t_3	1.5%	7%	2%
t_4	0.5%	7%	1%

Table 4: Profitable sequences mined from Table 2. Minimum average profit threshold $minprofit$=3%

Sequence	**Average profit (%)**
$b \rightarrow b$	$\frac{18}{3}$
$c \rightarrow b$	$\frac{15}{3}$
$c \rightarrow c$	$\frac{10}{3}$
$\{c, b\} \rightarrow b$	$\frac{13}{3}$

each record is a set of feature-value pairs. The set of considered features is fixed for all records in the dataset. Structured datasets are commonly used in various application domains (e.g., biological data analysis, textual data analysis) [27]). In our context, a record is a set of weighted items, where all items are associated with the same time stamp. It indicates the average daily returns of all the considered stocks at a given time stamp.

Definition 4. Structured dataset. *Let T be a time period and TS be a subset of time stamps $t_k \in T$. Let I be a set of weighted items. A record $R(t_k)$ is the set of all weighted items $\langle s_j, r_k^j \rangle \in I$ associated to time stamp t_k. A structured dataset is a set of records $R(t_k)$, one for each time stamp $t_k \in T$.*

Table 3 reports the structured dataset corresponding to the sample data

in Table 2. The dataset consists of four records, one per trading day (time stamps t_1-t_4). For each trading day the daily returns of all the stock are reported. For example, at time stamp t_1 stock a made a daily return equal to 5%, b gained 5%, and c 7%.

3.2. Model generation

This block applies regression and sequence mining techniques to analyze historical stock data. In the following, we will separately analyze the sequence mining and regression algorithms.

3.2.1. Sequence mining from weighted sequential stock data

Sequence mining focuses on discovering recurrent sequences of itemsets from large datasets. In our context, the goal is to discover recurrent sequences of profitable stocks.

The traditional sequence mining problem entails extracting from a sequential dataset all frequent sequences, i.e., all sequences whose frequency of occurrence (support) in the source data is above a given threshold [15]. The support of a sequence is defined as the fraction of time windows in which the sequence occurs at least once.

For example, let us consider again the sequential stock dataset reported in Table 2. Sequence $c \rightarrow b$ has support equal to 100%, because it occurs in all the time windows. In our context, frequent sequences represent combinations of stocks that sequentially co-occur in the considered time period in a large enough number of time windows. Based on sequence $c \rightarrow b$, the occurrence of stock c on the k-th trading day (i.e., at time stamp t_k) implies the occurrence of stock b on the next day (i.e., at time stamp t_{k+1}). However, the support

measure does not consider the daily profits of the stocks.

To evaluate sequences according to the profit of the corresponding stocks, we extend the traditional sequence mining problem to cope with weighted data. The key idea is to give higher importance to the time windows in which the corresponding stock returns are relatively high. To this aim, we extend the definition of support of a sequence by considering also the daily relative return of the stocks in the sequence within each time window. Hence, we introduce the concept of *average sequence profit* as follows.

Definition 5. Average sequence profit. *Let D be a weighted sequential stock dataset, T be the corresponding time period, $tw_q \in T$ be an arbitrary time window, and $S(tw_q)$=$S_i \rightarrow S_j \rightarrow \ldots \rightarrow S_z$ be an arbitrary input-sequence in D. Let S^*=$I_i \rightarrow I_j \rightarrow \ldots \rightarrow I_z$ a sequence occurring in D, where I_i, I_j, ..., I_z are sets of stocks. Let minret(I_x, S_x) be the least return of any stock in I_x associated with input-sequence S_x, i.e., minret(I_x, S_x) = $min_{j|s_j \in I_x \wedge \langle s_j, r_k^j \rangle \in S_x} r_k^j$. Let minret($S^*$, $S(tw_q)$) be the least return of any stock in S^* for any S_x occurring in tw_q, i.e., minret(S^*, $S(tw_q)$) = $min_{I_x \in S^* \wedge S_x \in S(tw_q)} minret(I_x, S_x)$. The average profit of sequence S^* is the average of the least returns of any stock in the sequence over all time windows, i.e., $\frac{\sum_{tw_q \in T} minret(S^*, S(tw_q))}{|\{tw_q \in T\}|}$.*

For example, Table 4 reports the sequences whose average profit is above 3% (i.e.,$minprofit$=3%). For example, the least daily returns of sequence $c \rightarrow b$ are 6% (stock b) in time window 1, 7% (stocks c and b) in time window 2, and 2% (stock c) in time window 3. Hence, the average profit is equal to 5%.

Algorithm 1 WeightedSequenceMining(D, $minprofit$)

Input: D, a weighted sequential dataset
Input: $minprofit$, a minimum average profit treshold
Output: $\mathcal{F}$, the set of *profitable sequences*
/*Generate the equivalent dataset D_e*/
1: D_e=$\emptyset$
2: **for all** weighted sequence $IS(tw_q)$ in D **do**
3: $\mathcal{EIS}(tw_q)$ = equivalentSequenceSet($IS(tw_q)$)
4: insert $\mathcal{EIS}(tw_q)$ into D_e
5: **end for**
/*Mine all profitable sequences from D_e*/
6: $\mathcal{F}$ = weightedSPADE(D_e, $minprofit$)
7: **return** $\mathcal{F}$

Considering the least daily return for each time window allows us to discard the sequences that contain non-profitable stocks.

Problem statement. Given a weighted sequential dataset D and a minimum (user-provided) average profit threshold $minprofit$, our aim is to extract all the *profitable sequences*, i.e., all the sequences whose average profit is above $minprofit$.

The mining algorithm

To mine profitable sequences from sequential stock data we extended the established sequence mining algorithm SPADE [35] to successfully cope with weighted data. Our approach entails the following steps: (i) data transformation, and (ii) pushing of item weights into the sequence mining process. Algorithm 1 reports the pseudocode of the mining algorithm, which is thoroughly described below.

Data transformation. This step (Algorithm 1, lines 1-5) entails the transformation of the original weighted sequential stock dataset D into an equivalent version D_e. This procedure is needed to tailor the sequence mining

Algorithm 2 equivalentSequenceSet($IS(tw_q)$)

Input: $IS(tw_q)$, a weighted input sequence
Output: $\mathcal{EIS}(tw_q)$, an equivalent set of sequences
1: **repeat**
2: /*Extract the next equivalent sequence $IS(tw_{qe})$*/
3: min_return = least item weight among all the weights of items in $IS(tw_q)$
4: $IS(tw_{qe}) = IS(tw_q)$ where the weights of all weighted items are set to min_return
5: insert $IS(tw_{qe})$ in $\mathcal{EIS}(tw_q)$
/*Remove the weighted items that are completely represented in the equivalent sequences*/
6: remove all the weighted items with a weight equal to min_return from $IS(tw_q)$
7: decrease the weights of all weighted items in $IS(tw_q)$ by min_return
8: **until** $IS(tw_q)$ is not empty
9: **return** $\mathcal{EIS}(tw_q)$

process to weighted data. It has already been exploited in a different application context (i.e., infrequent pattern mining) to incorporate item weights during itemset extraction [5]. Our algorithm integrates a preprocessing step similar to those applied in [5] to tailor weighted sequential data to the subsquent extraction phase.

D_e includes, for each original input-sequence related to a time window tw_q (denoted as $IS(tw_q)$) an equivalent set of input-sequences, denoted as $\mathcal{EIS}(tw_q)$. For each input-sequence in the transformed version $\mathcal{EIS}(tw_q)$ all the items (i.e., stocks) in the input-sequence have the same weight (i.e., the relative return). The equivalent version D_e of the weighted sequential stock dataset contains a set of equivalent input-sequences $\mathcal{EIS}(tw_q)$ for each input-sequence $IS(tw_q)$ in D (see Algorithm 1, lines 1-5).

For example, Table 5 reports the equivalent set of input-sequences $\mathcal{EIS}(tw_1)$ corresponding to the original input-sequence associated to time window tw_1 in Table 2.

The procedure used to transform the input-sequence is described by Al-

Table 5: Equivalent input-sequence set related to time window 1

Time period	**Time stamp**	**Input-sequence**
Time window $tw_{1(a)}$	t_1	$\langle a, 2\% \rangle$ $\langle b, 2\% \rangle$ $\langle c, 2\% \rangle$
	t_2	$\langle a, 2\% \rangle$ $\langle b, 2\% \rangle$ $\langle c, 2\% \rangle$
Time window $tw_{1(b)}$	t_1	$\langle a, 3\% \rangle$ $\langle b, 3\% \rangle$ $\langle c, 3\% \rangle$
	t_2	$\langle b, 3\% \rangle$ $\langle c, 3\% \rangle$
Time window $tw_{1(c)}$	t_1	$\langle c, 1\% \rangle$
	t_2	$\langle b, 1\% \rangle$ $\langle c, 1\% \rangle$
Time window $tw_{1(d)}$	t_1	$\langle c, 1\% \rangle$
	t_2	$\langle c, 1\% \rangle$

gorithm 2. The least weight (i.e., the minimum stock return) in the original input-sequence is considered first (e.g., the relative return 2% of stock a in our example). Then an equivalent sequence is generated $IS(tw_{qe})$ (Algorithm 2, line 4). $IS(tw_{qe})$ has the same items (stocks) of $IS(tw_q)$ but the weights of all items are set to the minimum stock return (min_return). In the example, the first generated equivalent input-sequence is the one identified by time window $tw_{1(a)}$ in Table 5. It contains the same items of the initial sequence but the weights are set to 2% (the least among the weights of all stocks). Then, the weighted items (i.e., the pairs (stocks, return)) that are already completely represented in the generated equivalent input-sequence are removed from the initial sequence $IS(tw_q)$ (Algorithm 2, line 6) and the weights of the others are decreased by min_return (line 7). In our example, after these updates, $IS(tw_q)$ becomes $\langle a, 3\% \rangle \langle b, 3\% \rangle \langle c, 5\% \rangle \rightarrow \langle b, 4\% \rangle \langle c, 5\% \rangle$. The loop is repeated on the update version of $IS(tw_q)$ to generate the next equivalent sequence (the one identified by time window $tw_{1(b)}$ in Table 5) and then $IS(tw_q)$ is update again (it becomes $\langle c, 2\% \rangle \rightarrow \langle b, 1\% \rangle \langle c, 2\% \rangle$). The iterative procedure ends when $IS(tw_q)$ becomes empty (i.e., when all

the weighted items of the initial sequence are "represented" by means of the weighted items of the set of equivalent sequences). For example, recalling the running example, other two equivalent sequences are generated (the last two reported in Table 5).

While each original input-sequence in D is characterized by weighted items with different weights, each equivalent input-sequence in D_e is characterized by the same weight for all its weighted items. As discussed below, the equivalent dataset version is profitably exploited by our modified version of SPADE to mine profitable sequences.

Pushing of item weights into the sequence mining process. In SPADE [35] candidate sequences are generated, stored in an hash-tree, and then compared with the original input-sequences to compute their support values. Frequency counts are performed by scanning the hash-tree and by matching the sub-sequences of the input-sequences of the dataset. The sequence frequency count is increased by one every time the considered sequence occurs in any sub-sequence of an input-sequence. This approach is not applicable to mine profitable sequences of stocks, because the algorithm cannot handle item weights. Hence, to perform profitable sequence mining we need to count average profits instead of simple frequency counts. In the modified version of SPADE, every time the considered sequence occurs in any sub-sequence of an equivalent input-sequence its frequency count is increased by the weight of the weighted items of the matched sub-sequence (all the items of an equivalent sequence have the same weight by construction). At the end of the counting process, the average profit is computed dividing the weighted frequency of each sequence by the number of input sequences.

Thanks to the property stated below, the average profit can be straightforwardly computed by scanning the input-sequences of the equivalent dataset version D_e and by averaging the profit values associated with any stock in the sequence.

Property. Given an arbitrary sequence S, the average profit count of S performed on the equivalent dataset version D_e corresponds to those performed on the original dataset version D.

A sketch of proof of the aforesaid property is given below.

Proof. Let $IS(tw_i)$ be an arbitrary input-sequence of D and $\mathcal{EIS}(tw_q)$ its equivalent set of input-sequences in D_e. Let $\langle s_j, r_k^j \rangle$ be the least weighted item contained in $IS(tw_q)$ by considering only the items s_j occurring in both $IS(tw_q)$ and S. By construction, every input-sequence in D_e containing stock s_j contains all the other stocks in $\mathcal{S}$ as well. Hence, the average profit of sequence S in D_e is the average of the sum of the least stock returns associated with all the equivalent input-sequence in $\mathcal{EIS}(tw_q)$, which, by construction, correspond to the average profit computed over the original dataset D. $\square$.

Analysis of algorithm complexity.. The complexity of the preprocessing step is $O(|D| \cdot |I|)$, where $|D|$ is the dataset size (the number of considered time windows multiplied by the number of time stamps per time window) and $|I|$ is the size of candidate stock set. In the performed experiments, $|D|$ is approximately equal to 30, while $|I|$ ranges between 40 and 100. Note that the real complexity of the preprocessing step is significantly lower than the upper bound estimate because all rows in the sequential dataset contain the same stocks and many stocks had the same daily return on several trading

days.

Let $I_f \subseteq I$ be the subset of stocks that achieved at least one daily return above the given threshold. The complexity of the weighted sequence mining process is $O(|I_f|^2)$ and it corresponds, in the worst case, to those of generating all 2-length sequences from traditional (unweighted) datasets [35]. Finally, the complexity of the ranking and scan operations performed on the output sequences is $O(|D| \cdot log(|D|))$. Since, in the datasets used for model generation $|D| << |I|$, the complexity of the overall data mining process is $O(|I_f|^2)$.

3.2.2. Regression from structured stock data

Our goal is to predict the daily relative return of each candidate stock on the following trading day (i.e., at time stamp t_k) based on the stock prices on the preceding days (i.e., at time stamps t_{k-1}, t_{k-2}, ...). To this aim, we first consider historical stock data tailored to the structured data model (see Section 3.1), we sampled historical data using windowing techniques [26], and then we trained different regression models to capture the most significant underlying patterns useful for prediction purposes.

Windowing. Windowing entails sampling temporal data according to the corresponding time stamps. To predict the return r_k^j of stock s_j at time stamp t_k, the list of the stock returns at the previous time stamps is considered. Specifically, given a fixed window size W, we consider the returns of stock s_j at time stamps t_{k-1}, t_{k-2}, ..., t_{k-W}. A sliding window is used to change the value of k and shifting the window accordingly [27].

For example, by setting W=2 in the first row we consider t_3 as prediction time stamp and [t_1, t_2] as training window. In the second row the window is shifted on the right, thus, t_4 becomes the prediction time stamp and [t_2,

Table 6: Windowing (W=2) on the structured stock dataset in Table 3

a (t_{k-2})	a (t_{k-1})	a (t_k)	b (t_{k-2})	b (t_{k-1})	b (t_k)	c (t_{k-2})	c (t_{k-1})	c (t_k)
5%	2%	1.5%	5%	6%	7%	7%	7%	2%
2%	1.5%	0.5%	6%	7%	7%	7%	2%	1%

t_3] the training window. For each row the returns of each stock are listed. For instance, in the first row column a (t_{k-2}) assumes value 5% because it corresponds to the return of stock a at time stamp t_1, while in the second row it assumes value 2% because it corresponds to the return of stock a at time stamp t_2.

Model learning. To predict the return of each stock we applied three established regression techniques: (i) Support Vector Machines, (ii) Linear Regression, (iii) Decision Tree. More details on the prediction models are given in [26]. Specifically, we considered the LibSVM, Linear Regression, and RepTree algorithms available in RapidMiner (rapid-i.com) [16]. For each algorithm we used the standard configuration suggested by the respective authors.

3.3. Stock recommendation

Based on the data mining models generated from historical stock data, the last step of the BeSt system recommends to intraday traders the most appealing stocks to trade on the next trading day. The strategy used to recommend tradeable stocks depends on the type of prediction model considered.

Recommendation based on sequences. Intraday traders commonly trade stocks that are most likely to outperform the market. Profitable se-

quences represent sequences of stocks whose returns are relatively high according to historical data. To mine profitable sequences we considered historical data referring to a time period preceding but relatively close to the present, so that the stock data distribution is assumed to be the same in the near future.

To recommend tradeable stocks at time stamp t_k we first consider the Q best-performing stocks ($TOPQ(t_{k-1})$) on the training day corresponding to time stamp t_{k-1} (hereafter denoted as $B(t_{k-1})$). Then, for each stock s_j in $TOPQ(t_{k-1})$ we pick the first K sequences in order of decreasing average profit. In case of ties, the first K sequences in alphabetical order are considered. The selected sequences are in the form $s_j \rightarrow STB$, where STB is the subset of appealing stocks to trade, i.e., the stocks that are most likely to yield high profits on the next trading day. If $K > 1$, the union of the STB sets is considered. The final set of suggested stocks is obtained as the union of the STB parts of the first k selected rules. In Section 4 we experimentally analyzed the impact of parameters Q and K and of the size of the analyzed time period on the performance of the BeSt system.

For example, let us consider again the historical data reported in Table 2 and the profitable sequences reported in Table 5. Profitable sequences are extracted by enforcing a minimum average profit equal to 3% to early discard non-profitable patterns. If on a certain trading day (e.g., at time stamp t_{k-1}) stock c ranked first in terms of average profit, analysts may consider the top ranked profitable sequence in the form $c \rightarrow STB$. Specifically, $c \rightarrow b$ has maximal average profit ($\frac{15}{3}$) and, thus, it could be deemed as interesting for supporting investors' decisions. According to the selected sequence, stock c

is an interesting stock to trade, because it is likely to achieve relatively good profits in the short-term.

Recommendation based on regression models. For each stock we predicted the value of the daily relative return on the next day based on the historical prices of all the considered stocks on the preceding days (see Section 3.2.3). The stock with maximal expected return is recommended. In case of tie, the first one in alphabetical order is considered. In Section 4 we experimentally analyzed the impact of the sliding window size W on the performance of the BeSt system.

3.4. Driving long/short investments using the BeSt *system*

Trading with long positions entails betting on an increase in value of a stock, whereas trading with short positions entails betting on a decrease in value of the same. Recommending stocks for long intraday trades implies choosing the stocks whose prices are following an increasing trend. Conversely, recommending stocks for short intraday trades entails considering stocks whose prices are following a decreasing trend.

The BeSt system recommends stocks that are worth trading on the next day for both long and short investments. In the former case, historical data consist of the positive relative returns achieved by the analyzed stocks. In the latter, they consist of the daily relative losses achieved by each stock. Hence, according to the input data, the BeSt system is able to give recommendations for both long- and short-position trades.

4. Experiments

We performed several experiments on real stock data acquired by means of the Yahoo! finance APIs [33]. All the experiments were performed on a quad-core 3.30 GHz Intel Xeon workstation with 16 GB of RAM, running Ubuntu Linux 12.04 LTS.

The experimental section is organized as follows. Sections 4.1 and 4.2 describe the main characteristics of the real datasets considered in this study and the procedure used to validate the performance of the BeSt system, respectively. Section 4.3 summarizes the characteristics of the trading strategies tested in this study. Sections 4.4 and 4.5 analyze the profits made by each strategy and the effect of the algorithm configuration settings, respectively. Finally, Section 4.6 analyzes the scalability of the proposed strategies with respect to the number of stocks.

4.1. Stock datasets

In our experiments we considered the daily closing prices of the stocks belonging to three different indices: NASDAQ-100, Dow Jones, and FTSE MIB. NASDAQ and Dow Jones are two major American indices, while the *FTSE MIB* is the benchmark stock market index for the Borsa Italiana, i.e., the Italian national stock exchange. The NASDAQ-100 index is made up of the 100 largest companies of the NASDAQ index, which mainly belong to the technology and biotechnology sectors. To consider a larger and diversified pool of U.S. stocks, we considered also the Dow Jones index, which comprises stocks belonging to a variety of sectors.

Since the outcomes of the experiments can be affected by the conditions of

the analyzed markets, we investigated the performance of the BeSt system in two opposite situations: (i) a favorable market condition, i.e., the boom of U.S. markets in year 2013, and (ii) an unfavorable condition, i.e., the 2011 stock markets fall due to fears of contagion of the European sovereign debt crisis to Spain and Italy. Situation (i) is an example of bull market condition, where stock prices followed an increasing trend and traders were likely to perform long investments. Conversely, Situation (ii) corresponds to a bear market, where a decreasing trend appeared and traders mainly performed short investments.

To set up the experiments, we generated the following data collections:

- *U.S. 2013.* This collection consists of the daily closing prices of the stocks belonging to the NASDAQ-100 and Dow Jones indices on all trading days of year 2013.

- *U.S. 2011.* This collection consists of the daily closing prices of the stocks belonging to the NASDAQ-100 and Dow Jones indices on the all trading days of year 2011.

- *Italy 2013.* This collection consists of the daily closing prices of the FTSE MIB stocks on the all trading days of year 2013.

- *Italy 2011.* This collection consists of the daily closing prices of the FTSE MIB stocks on the all trading days of year 2011.

4.2. Experimental design

To validate the performance of the BeSt system we simulated intraday stock investments at different points of time. Training datasets were gen-

erated by merging stock data acquired on 15 consecutive trading days. To simulate long positions, we assumed to buy the stock recommended by the system on the day after the end of the training period at the closure price and to sell the stock on the subsequent day, again at the closure price. More specifically, on day d_n we generated a prediction model based on the historical stock prices acquired in the time period $[d_{n-15}, d_{n-1}]$, where d_* are trading days. Then, we buy the stock recommended by considering both the prediction model and the list of best-performing stocks on day d_n. The buy operation is completed immediately before the closure of day d_n. Finally, we sell the same stock immediately before the closure of day d_{n+1}. For the sake of simplicity, the buy and sell prices are assumed to be exactly equal to the closing prices on days d_n and d_{n+1}, respectively. For example, if the training time period is between November, 15 2013 and November, 29 2013, we simulated a buy operation of the recommended stock on November, 30 2013 immediately before the closure time and the sell of the same stock on December, 1 2013 immediately before the closure time. Similarly, to simulate short positions we assumed to sell the stock on the day after the end of the training period immediately before the closure time and to buy the stock on the subsequent day immediately before the closure time.

We generated stock recommendations based on different data mining models and we computed the average daily profits made by each strategy based on the historical stock prices. Note that the estimates of intraday profits are approximated, because intraday traders could trade a stock multiple times per day based on multiple long/short signals. For the sake of simplicity we separately tested long and short investments. Furthermore, we

approximated the transaction costs to 0.15% and we set the stop loss (i.e., the maximum loss for each operation) to 1%[1]. Note that since transaction costs depend on the policy of the broker, their precise simulation is out of scope of this work. To compute the average daily profits, we executed each strategy once per day by using the preceding two-week time period as training period. In case a strategy generated multiple recommendations per day, the corresponding profits were averaged. On the other hand, the days on which no recommendations are available were disregarded.

4.3. Tested strategies

We compared the performance achieved by the following strategies:

- Weighted SEQuence mining (W-SEQ),
- UNWweighted SEQuence mining (UNW-SEQ),
- Support Vector Machines using a linear kernel (LibSVM Linear),
- Support Vector Machines using a polynomial kernel (LibSVM Polyn.),
- Linear Regression (LR),
- Decision Trees (RepTree),
- Neural Networks (NN),
- Arima Time Series Forecasting (Arima),

[1]The stop loss is the automatic closure of a trading position in case the percentage loss exceeds a given value. It is commonly used by intraday traders to prevent significant losses [32].

- Random choice (Random).

To evaluate the performance of Support Vector Machines, we used the LibSVM [6] implementation provided by the respective authors and we exploited two different kernel functions (i.e., linear, polynomial).

To test Linear Regression, Decision trees, and Neural Networks we exploited the implementations available in the data mining and machine learning suite Rapid Miner 5.3 [16]. For Arima, we used the R implementation [25] provided by the authors.

Finally, we tested also a random strategy, which randomly selects a stock per day.

4.4. Evaluation of different recommendation strategies

Table 7 summarizes the results achieved by all the tested strategy with both long and short positions on the U.S. and Italy data collections. We recall that long positions were simulated in year 2013, whereas short positions were simulated in 2011. In the following, the quality of the proposed strategies is evaluated by means of standard mathematical/statistical metrics, such average and standard deviation, computed over the daily profits of the stocks recommended by each approach over the days of considered collections. The daily profit of a generic stock that is bought at day d_n and sold at day d_{n+1} is given by

$$\frac{closure\_price(d_{n+1}) - closure\_price(d_n)}{closure\_price(d_n)} - transaction\_cost$$

where transaction cost is set to 0.15%.

To gain insights into the performed recommendations, Table 7 reports for each strategy the average daily profit, the standard deviation of the

Table 7: U.S. and Italy data collections. Average daily profits (%), standard deviations, number of recommendations, and maximal gains (%).

U.S. 2013									
Long positions									
	W-SEQ Q=3, K=1	UNW-SEQ Q=3, K=1	SVM Linear W=2	SVM Polyn. W=2	LR W=2	RepTree W=2	NN W=2	Arima	Random
Average daily profit (%)	0.39	0.20	0.18	0.11	0.13	0.18	0.14	0.11	0.06
Standard deviation	1.92	1.45	1.51	1.36	1.35	1.40	1.11	1.28	1.39
# of recommendations	144	242	219	219	219	219	219	219	243
Max daily profit (%)	14.19	8.99	6.18	5.52	5.83	5.52	4.57	4.74	10.91
U.S. 2011									
Short positions									
	W-SEQ Q=3, K=1	UNW-SEQ Q=3, K=1	SVM Linear W=2	SVM Polyn. W=2	LR W=2	RepTree W=2	NN W=2	Arima	Random
Average daily profit (%)	0.72	0.22	0.40	0.48	0.53	0.38	0.27	0.38	0.31
Standard deviation	2.41	1.69	2.39	2.40	2.23	2.32	1.57	1.74	1.51
# of recommendations	97	245	191	191	190	190	191	191	244
Max daily profit (%)	12.02	9.15	20.85	20.85	10.82	20.85	7.7	8.27	5.58
Italy 2011									
Short positions									
	W-SEQ Q=3, K=1	UNW-SEQ Q=3, K=1	SVM Linear W=2	SVM Polyn. W=2	LR W=2	RepTree W=2	NN W=2	Arima	Random
Average daily profit (%)	0.99	0.56	0.83	0.86	0.70	0.66	0.34	0.59	0.67
Standard deviation	2.52	2.04	2.31	2.35	2.19	2.50	1.96	2.19	2.16
# of recommendations	81	253	207	206	211	200	211	211	254
Max daily profit (%)	7.72	8.48	11.86	13.56	10.75	13.56	8.16	12.84	11.98
Italy 2013									
Long positions									
	W-SEQ Q=3, K=1	UNW-SEQ Q=3, K=1	SVM Linear W=2	SVM Polyn. W=2	LR W=2	RepTree W=2	NN W=2	Arima	Random
Average daily profit (%)	0.63	0.36	0.43	0.36	0.38	0.75	0.43	0.48	0.30
Standard deviation	2.72	2.06	2.23	2.02	1.86	2.52	2.23	1.95	1.52
# of recommendations	125	261	176	175	175	175	176	178	261
Max daily profit (%)	22.83	22.83	14.54	14.54	14.54	16.29	14.54	12.22	8.57

daily profit, the total number of recommendations, and the daily maximal gain[2] computed over the time period of each considered collection. Due to the lack of space, the detailed daily profits per stock are available at http://dbdmg.polito.it/wordpress/discovering-profitable-stocks-for-intraday-trading/. The average daily profit and the standard deviation of the daily profit provide the information about the potential profitability of each strategy and its variability, respectively. Ideally, high average daily profits and small standard deviation values are desirable. The total number of recommendations is reported in Table 7 as well. This statistics indicate how frequently each strategy recommends a stock to trade in the analyzed time period. Note that strategies that produce a higher number of recommendations does not necessarily yield higher profits, because transaction costs and stop losses may reduce gains.

Independently of the market condition and of the considered time period, W-SEQ performed best in terms of average daily profit (e.g. against SVM Linear, +0.21% on the U.S. 2013, +0.32% on the U.S. 2011, +0.16% on the Italy 2011, +0.20% on the Italy 2013). The random choice appears to be the worst-performing strategy, whereas the results achieved by SVMs, Decision trees, Linear Regression, and Neural Network were roughly comparable. Since on some trading days no sequences are applicable, the number of recommendations of W-SEQ is fairly lower than those of all the other approaches (e.g. on the U.S. 2013 W-SEQ is characterized by 144 recommendations against the 219 recommendations made by SVMs).

We compared also the performance of W-SEQ with that of a modified

[2]The maximal loss has been omitted because it corresponds to the stop loss value (1%).

sequence-based algorithm version, denoted as UNWweighted SEQuence mining (UNW-SEQ), which ignores item weights, i.e., on each trading day it considers all stocks yielding a positive relative return, independently of the actual daily profit. The results show that UNW-SEQ generated a larger number of recommendations than W-SEQ (e.g. 242 against the 144 of W-SEQ on the U.S. 2013 data collection), but the average daily profit was halved (0.20% UNW-SEQ vs. 0.39% W-SEQ). Therefore, based on the achieved results, weighting stock occurrences by the corresponding daily return appeared to be particularly effective in stock recommendation.

Statistical analysis. To validate the statistical significance of the differences between the nine considered strategies, we used the Friedman test by using the same procedure that is usually used to compare multiple classifiers on a subset of datasets [10]. Specifically, the following steps were applied to perform the Friedman test.

A) The daily return of each strategy on each trading day is computed.

B) The strategies are ranked on each trading day according to the daily return computed at Step A. To perform a fair comparison, we considered only the trading days on which all the strategies produced a stock recommendation.

C) For each strategy, its average ranking computed over all the considered trading days is computed.

D) The observed differences between the average rankings of the considered strategies are compared with the critical difference threshold CD that establishes whether the difference is statistically significant. CD is defined as follows: $CD = q_\alpha \frac{\sqrt{k(k+1)}}{6N}$, where N is the number of trading

Table 8: U.S. and Italy data collections. Average rankings and Friedman test.

Strategy	Mean rank			
	U.S. 2013 Long positions (CD = 0.03)	U.S. 2011 Short positions (CD = 0.05)	Italy 2011 Short positions (CD = 0.07)	Italy 2013 Long positions (CD = 0.05)
W-SEQ	4.94	**5.39**	**5.28**	5.68
UNW-SEQ	**4.90**	5.73	6.09	**5.49**
SVM Linear	6.17	6.33	5.49	6.79
SVM Polyn.	6.19	6.32	5.87	6.12
LR	5.63	5.85	5.85	6.32
RepTree	6.15	6.52	5.94	5.64
NN	5.21	5.81	6.08	6.79
Arima	5.96	5.75	6.3	5.74
Random	5.61	5.77	5.45	5.75

days, k is the number of compared strategies, α is the significance level, and q_α is the critical value. We set the significance level a to 95%. Hence, the corresponding critical value q_α is 2.394.

We performed the Friedman test separately for each of the four pairs (collection, long/short position strategy) reported in Table 8. In Table 8, the best average mean rank value for each column is written in boldface. For all the considered collections, the majority of the difference between the mean ranks of the considered strategies are significantly different (i.e., for almost all pairs of strategies the difference between their average rankings is greater than CD). On average, sequence-based approaches performed significantly better than regression-based approaches.

Model analysis. To gain insights into the characteristics of the model generated by W-SEQ, we analyzed the sequences used to perform stock recommendations on the U.S. 2013 collection. In the following, we consider one representative training day (January, 9 2013). The experiments were performed by setting Q=3 and $K = 1$, which is the standard configuration of W-SEQ). By setting Q to 3, three stocks were used to trigger stock recom-

mendation on each trading day. For example, on January, 8 2013 the three best-performing U.S. stocks were CELG, DISCA, and ILMN, respectively. From historical data the following top-ranked sequences were extracted and selected:

(1st) DISCA $\rightarrow$ MU (0.78%)

(2nd) DISCA $\rightarrow$ VTRX (0.70%)

(3rd) DISCA $\rightarrow$ PCLN (0.61%)

(4th) CELG $\rightarrow$ MU (0.54%)

where the percentage average profit of each sequence is reported in brackets. By setting K=1, only the top-ranked sequence is used to perform stock recommendation. Hence, stock MU was recommended as the most profitable stock to buy immediately before the closure on January, 8 2013. On January, 9 2016 the intraday trade on MU yielded a 0.95% gain (excluding the transaction costs). This sequence highlights a correlation between the increase of the price of stock $DISCA$ on one day and the increase of those of stock MU on the following day. Such information can be exploited by traders to have more insights into the W-SEQ stock recommendation process and to evaluate the reliability of stock recommendations. Unlike W-SEQ, the majority of the considered regression algorithms do not provide any humanly-interpretable information and, thus, model exploration is challenging. Among the considered regression algorithms, only the decision tree-based one (RepTree) can provide humanly-interpretable information based on the paths of the tree used to perform the predictions. For instance, for the U.S. 2013 collection and the trading day August, 1 2013, the EXPE stock is recommended by the strategy based on RepTree. The path/rule LCMA(d_n) $\geq$ 1.39% $\rightarrow$

EXPE(d_{n+1})=1.28% was used to predict the expected return of stock EXPE on day d_{n+1} (where d_{n+1} is August, 1 2013 in this example). This path shows that a return equal to 1.28% was predicted for stock EXPE on August, 1 2013 because the profit of stock LCMA is higher than 1.39% on July, 31 2013. Hence, based on this rule, a positive return of stock LCMA implies a positive return of stock EXPE on the day after.

4.5. Effect of different configuration settings

We analyzed the effect of varying the values of the main system parameters on the average daily profits. For the sake of brevity, hereafter we will report the results achieved by considering long positions on the U.S. 2013 collection and short positions on the Italy 2011 collection. Similar results were obtained by considering different datasets and trading positions.

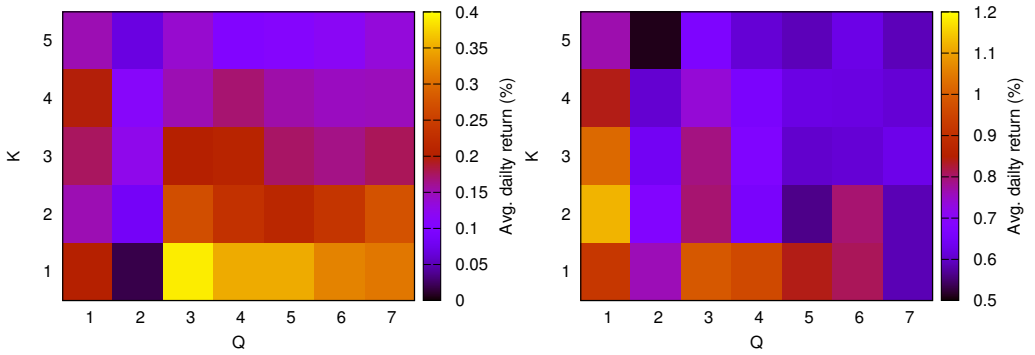
4.5.1. Analysis of the effect of the W-SEQ parameters

The W-SEQ strategy has two main parameters: (i) the number Q of best-performing stocks that are exploited at time stamp t_{k-1} to select the most profitable sequences, and (ii) the number K of recommended stocks, based on the sequences selected for each stock from the top-Q list, at time stamp t_{k-1}.

The heat map chart reported in Figure 2 shows the average daily profits achieved by varying the values of Q and K. Independently of considered market, the best average daily returns were achieved by setting Q to 3 and 4 and K to 1.

To thoroughly analyze the standard configuration of W-SEQ (Q=3, K=1), we also plotted, in Figures 3-4, the average daily profits achieved by varying

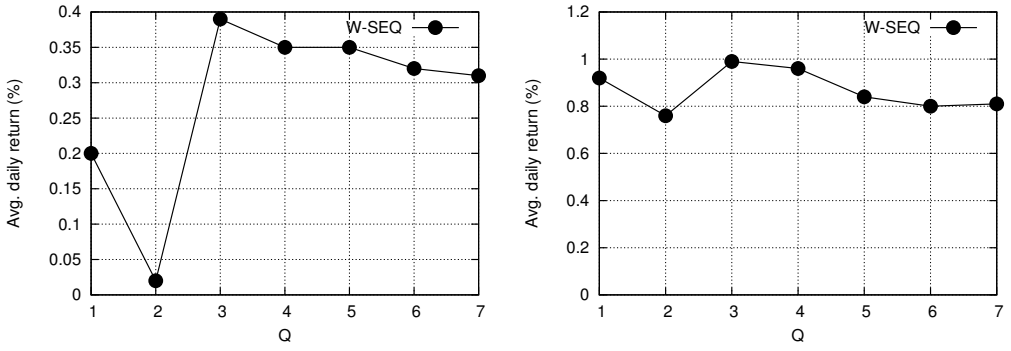
the values of Q (with K=1) and K (Q=3), respectively.



(a) U.S. 2013 (long investments) (b) Italy dataset 2011 (short investments)

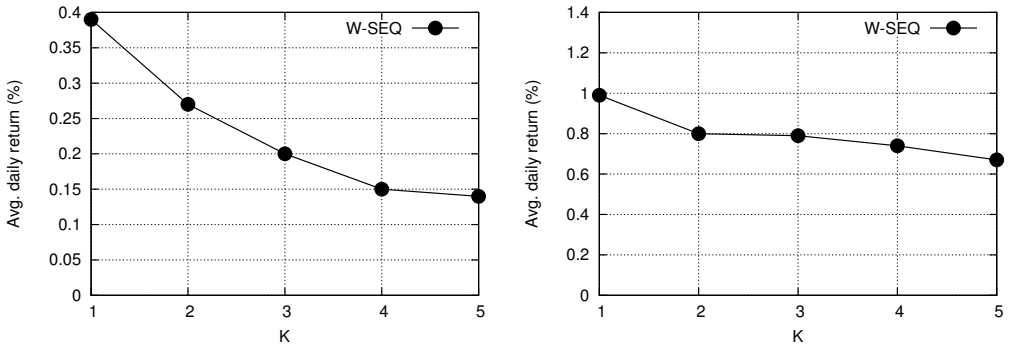
Figure 2: Effect of Q and K on the average daily net return.

Effect of Q. Independently of the considered market and of the underlying conditions, the average daily profit of the W-SEQ strategy is roughly stable while setting Q values between 3 and 4 (see Figure 3). Setting Q values out of this range may bias the recommendation process, because recommendations are driven by a too small/large number of best-performing stocks.



(a) U.S. 2013 (long investments) (b) Italy dataset 2011 (short investments)

Figure 3: Effect of Q on the average daily net return. K=1.



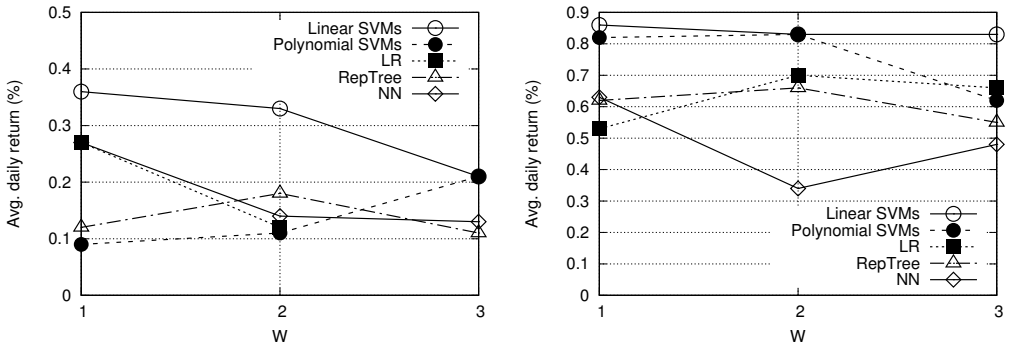
(a) U.S. 2013 (long investments) (b) Italy dataset 2011 (short investments)

Figure 4: Effect of K on the average daily net return. Q=3.

Impact of K. W-SEQ profits on average decrease while increasing the value of K (see Figure 4). Considering more than one sequence, and hence recommending more than one stock, results in biasing the quality of the prediction, because lower-profit sequences are more likely to produce unreliable recommendations.

4.5.2. Analysis of the effect of regression model parameters

The window size W slightly affects the performance of the stock recommendation system (see Figure 5). The highest profits were achieved, on average, by setting relatively small W values (i.e., between 2 and 3). Larger W values may bias the quality of the prediction, because the correlation between the past stock price variations and the future outcomes becomes weaker. Based on the achieved results, we recommend to set W to 2 on real stock data.



(a) U.S. 2013 (long investments) (b) Italy dataset 2011 (short investments)

Figure 5: effect of W on the average daily return.

4.6. Scalability

Figure 6 reports the average execution time of the considered strategies while varying the number of analyzed stocks. The experiments were performed on the U.S. 2013 data collection by varying the number of input stocks from 25 to 125. Similar results were obtained on the other markets.

The achieved results show that all the considered strategies were able to perform a prediction (i.e., select the stocks to trade/buy) in no more than two minutes. For most strategies the execution time is approximately twenty seconds. Hence, the proposed strategies can be efficiently applied to real stock data.

Regression-based algorithms scale almost linearly with the number of stocks while sequence-based approaches[3] are characterized by a non-linear scalability.

[3]The execution times of algorithms W-SEQ and UNW-SEQ are similar. Thus, their curves are overlapped in Figure 6.

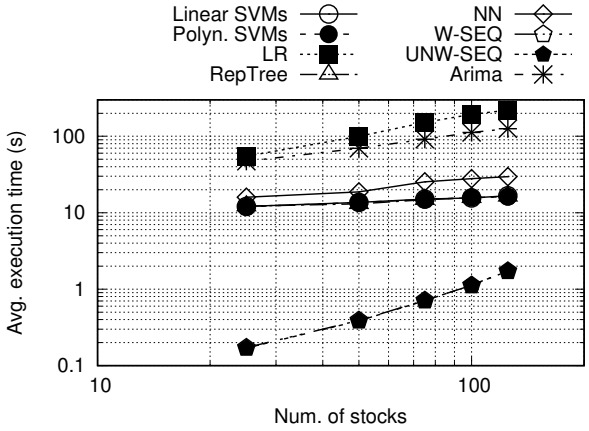


Figure 6: Scalability w.r.t. number of stocks. U.S. 2013 (long investments)

5. Discussion

To summarize the achieved results, in this section we compared the strategies used to predict stock returns in terms of (i) profitability, (ii) interpretability of the generated model, (iii) robustness to noise and presence of missing values in the source data, and (iv) training and prediction time.

Profitability. Based on the performed experiments, the weighted sequence-based W-SEQ algorithm appeared to be the most effective approach, because it achieved the highest average daily profits on a pool of datasets acquired from different markets and under different market conditions. Nevertheless, Decision trees and Support Vector Machines achieved fairly good results on most of the tested datasets and they performed better than Linear Regression in most cases. As expected, Random is the worst-performing strategy, because it relies on a naive strategy.

Interpretability. In the financial domain, traders are often interested in discovering the underlying trends behind financial data. Exploring the predic-

tion models can be an effective way to dig deep into historical stock data and to discover useful knowledge. Sequence-based strategies and decision trees generate readable models that are potentially manageable by domain experts through manual inspection. The model complexity depends on both the distribution of the analyzed data and the configuration setting of the considered algorithm (see Section 4.5). Conversely, Support Vector Machines and Linear Regression models are not easily interpretable.

Robustness to noise and presence of missing values. In our experiments, stock data have no missing values or non-sense values. However, noisy values or missing values may occur when extraordinary events happen (e.g. splits, capital increases, stock hanging). To handle these situations, we recommend to regenerate the prediction models on a new version of the dataset excluding the corresponding stocks to avoid introducing bias in stock recommendation. As discussed below, the cost of model regeneration is quite limited.

Training and prediction time. Support Vector Machines (SVMs) and Linear Regression typically require longer times to build the prediction model than decision trees (i.e., few minutes vs. few seconds). On the other hand, finding the best configuration setting for Decision trees can be a challenging task [27]. The training time of sequence-based approaches is comparable to those of Decision tree. However, as discussed in Section 4.6, setting the input parameters of the W-SEQ algorithm is rather simple, because the model is not sensitive to small parameter value variations.

6. Conclusions and future work

This paper presents BEst STock FInder (BESt), a new data mining approach to analyzing historical stock data and to discovering profitable stocks for intraday trading. The proposed approach, which relies on regression and sequence mining techniques, considers the daily stock price variations as a metric to evaluate the importance of a sequence of best-performing stocks across consecutive days.

The benefits of the proposed technique can be summarized as follows: (i) The ability to capture the underlying trends in stock price movements without the need for manually exploring financial data. (ii) The scalability of the proposed approach, which allows traders to consider a large number of candidate stocks. (iii) The higher profits generated by sequence-based techniques compared to standard approaches (e.g. Support Vector Machines), according to the experimental results achieved on multiple markets and under different market conditions. (iv) The interpretability of sequence-based models, which may help traders taking appropriate decisions.

The current version of the system does not generate trading signals on the recommended stocks. As future work, we plan to integrate effective trading signal generators into the designed framework. Furthermore, we aim at investigating the incrementality of our approach. Specifically, when new training objects are available, none of the proposed strategies can be incrementally updated. Thus, the model should be regenerated as soon as new training data are available. Hence, we aim at developing new incremental solutions for extracting sequences and frequent patterns (e.g., [1, 2]) and studying their application to stock data.

References

[1] E. Baralis, L. Cagliero, T. Cerquitelli, V. D'Elia, P. Garza, Expressive generalized itemsets, Inf. Sci. 278 (2014) 327–343.

[2] E. Baralis, L. Cagliero, P. Garza, Enbay: A novel pattern-based bayesian classifier, IEEE Trans. Knowl. Data Eng. 25 (2013) 2780–2795.

[3] E. Baralis, L. Cagliero, S. Jabeen, A. Fiori, S. Shah, Multi-document summarization based on the yago ontology, Expert Syst. Appl. 40 (2013) 6976–6984.

[4] J.M. Berutich, F. Lpez, F. Luna, D. Quintana, Robust technical trading strategies using {GP} for algorithmic portfolio selection, Expert Systems with Applications 46 (2016) 307 – 315.

[5] L. Cagliero, P. Garza, Infrequent weighted itemset mining using frequent pattern growth, IEEE Trans. Knowl. Data Eng. 26 (2014) 903–915.

[6] C.C. Chang, C.J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27.

[7] T.l. Chen, F.y. Chen, An intelligent pattern recognition model for supporting investment decisions in stock market, Information Sciences 346 (2016) 261–274.

[8] Y. Chen, S. Mabu, K. Hirasawa, J. Hu, Genetic network programming with sarsa learning and its application to creating stock trading rules,

in: Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, pp. 220–227.

[9] W.C. Chiang, D. Enke, T. Wu, R. Wang, An adaptive stock index trading decision support system, Expert Systems with Applications 59 (2016) 195–207.

[10] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.

[11] V. Dhar, Prediction in financial markets: The case for small disjuncts, ACM Trans. Intell. Syst. Technol. 2 (2011) 19:1–19:22.

[12] L. Dymova, P. Sevastjanov, K. Kaczmarek, A forex trading expert system based on a new approach to the rule-base evidential reasoning, Expert Systems with Applications 51 (2016) 1 – 13.

[13] R. de Frein, K. Drakakis, S. Rickard, Portfolio diversification using subspace factorizations, in: Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on, pp. 1075–1080.

[14] T. Geva, J. Zahavi, Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news, Decis. Support Syst. 57 (2014) 212–223.

[15] J. Han, M. Kamber, Data mining: concepts and techniques, Morgan Kaufmann, San Fransisco, 2006.

[16] M. Hofmann, R. Klinkenberg, RapidMiner: Data Mining Use Cases and Business Analytics Applications, Chapman & Hall/CRC, 2013.

[17] Y. Kara, M.A. Boyacioglu, Ö.K. Baykan, Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange, Expert systems with Applications 38 (2011) 5311–5319.

[18] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, J. Allan, Language models for financial news recommendation, in: Proceedings of the Ninth International Conference on Information and Knowledge Management, CIKM '00, pp. 389–396.

[19] B. Li, S.C. Hoi, V. Gopalkrishnan, Corn: Correlation-driven nonparametric learning approach for portfolio selection, ACM Trans. Intell. Syst. Technol. 2 (2011) 21:1–21:29.

[20] Q. Li, T. Wang, Q. Gong, Y. Chen, Z. Lin, S. Song, Media-aware quantitative trading based on public web information, Decision Support Systems 61 (2014) 93–105.

[21] M.M. Mostafa, Forecasting stock exchange movements using neural networks: Empirical evidence from kuwait, Expert Systems with Applications 37 (2010) 6302–6309.

[22] V. Parque, S. Mabu, K. Hirasawa, Global portfolio diversification by genetic relation algorithm, in: ICCAS-SICE, 2009, pp. 2567–2572.

[23] J. Patel, S. Shah, P. Thakkar, K. Kotecha, Predicting stock market index using fusion of machine learning techniques, Expert Systems with Applications 42 (2015) 2162–2172.

[24] M. Pulido, P. Melin, O. Castillo, Particle swarm optimization of ensemble neural networks with fuzzy aggregation for time series prediction of the mexican stock exchange, Information Sciences 280 (2014) 188–204.

[25] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[26] A. Rajaraman, J.D. Ullman, Mining of Massive Datasets, Cambridge University Press, New York, NY, USA, 2011.

[27] P.N. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, Addison-Wesley, 2006.

[28] L.A. Teixeira, A.L.I. De Oliveira, A method for automatic stock trading combining technical analysis and nearest neighbor classification, Expert systems with applications 37 (2010) 6885–6890.

[29] J.L. Ticknor, A bayesian regularized artificial neural network for stock market forecasting, Expert Systems with Applications 40 (2013) 5501 – 5506.

[30] J.Z. Wang, J.J. Wang, Z.G. Zhang, S.P. Guo, Forecasting stock indices with back propagation neural network, Expert Systems with Applications 38 (2011) 14346–14355.

[31] W. Wang, J. Yang, P.S. Yu, Efficient mining of weighted association rules (WAR), in: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'00, pp. 270–274.

[32] T. Williams, V. Turton, Trading Economics: A Guide to Economic Statistics for Practitioners and Students, The Wiley Finance Series, Wiley, 2014.

[33] YahooFinance, Yahoo Finance Website. Last access December 2015, 2014.

[34] M.J. Zaki, Sequences mining in categorical domains: Incorporating constraints, in: 9th ACM International Conference on Information and Knowledge Management, pp. 422–429.

[35] M.J. Zaki, Spade: An efficient algorithm for mining frequent sequences, Mach. Learn. 42 (2001) 31–60.

[36] K. Zbikowski, Using volume weighted support vector machines with walk forward testing and feature selection for the purpose of creating stock trading strategy, Expert Systems with Applications 42 (2015) 1797 – 1805.