Discussion Papers in Economics

No. 13/2017

Exploiting social media with higher-order Factorization Machines: Statistical arbitrage on high-frequency data of the S&P 500

Julian Knoll
Technische Hochschule Nürnberg Georg Simon Ohm

Johannes Stübinger
University of Erlangen-Nürnberg

Michael Grottke
University of Erlangen-Nürnberg

ISSN 1867-6707

Friedrich-Alexander-Universität Erlangen-Nürnberg
Institute for Economics
https://www.iwf.rw.fau.de/research/iwf-discussion-paper-series/

Exploiting social media with higher-order Factorization Machines: Statistical arbitrage on high-frequency data of the S&P 500

Julian Knoll[a], Johannes Stübinger[b], Michael Grottke[b]

[a] *Technische Hochschule Nürnberg Georg Simon Ohm, Keßlerplatz 12, 90403 Nürnberg, Germany*
[b] *University of Erlangen-Nürnberg, Department of Statistics and Econometrics, Lange Gasse 20, 90403 Nürnberg, Germany*

2017-06-09

Abstract

Over the past 15 years, there have been a number of studies using text mining for predicting stock market data. Two recent publications employed support vector machines and second-order Factorization Machines, respectively, to this end. However, these approaches either completely neglect interactions between the features extracted from the text, or they only account for second-order interactions.

In this paper, we apply higher-order Factorization Machines, for which efficient training algorithms have only been available since 2016. As Factorization Machines require hyperparameters to be specified, we also introduce the novel adaptive-order algorithm for automatically determining them.

Our study is the first one to make use of social media data for predicting high-frequency stock returns, namely the ones of the S&P 500 stock constituents. We show that, unlike a trading strategy employing support vector machines, Factorization-Machine-based strategies attain positive returns after transactions costs for the years 2014 and 2015. Especially the approach applying the adaptive-order algorithm outperforms classical approaches with respect to a multitude of criteria, and it features very favorable characteristics.

Keywords: Finance, Factorization Machine, social media, statistical arbitrage, high-frequency data.

Email addresses: julian.knoll@th-nuernberg.de (Julian Knoll), johannes.stuebinger@fau.de (Johannes Stübinger), michael.grottke@fau.de (Michael Grottke)

1. Introduction

With the internet providing textual data to a great extent, academic interest in predicting the stock market based on textual input has surged over the past few years. Nassirtoussi et al. (2014) gave a structured literature review on the theoretical and technical foundations of the existing literature. One stream of text mining for market prediction focuses on high-frequency data, i.e., intra-day market prices fluctuations. Following Nassirtoussi et al. (2014), key representatives of this category are Peramunetilleke and Wong (2002), Antweiler and Frank (2004), Schumaker and Chen (2009b), Groth and Muntermann (2011), Schumaker et al. (2012), and Lugmayr and Gossen (2013). These studies differ in the algorithms applied. The work by Peramunetilleke and Wong (2002), belonging to the class of decision rules, assigned certain probabilities to keywords describing how likely the related event will occur. Both Antweiler and Frank (2004) and Groth and Muntermann (2011) applied a set of algorithms in order to find textual signs which allow for statements about risk exposure in stock markets. Schumaker and Chen (2009b), Schumaker et al. (2012), and Lugmayr and Gossen (2013) used support vector machines (SVMs) for market prediction based on online text mining. This very common machine learning approach is used for classification and regression analysis. It exhibits the drawback of neglecting any interaction between features. Further steps were taken by Chen et al. (2014), who modeled second-order interactions using a so-called Factorization Machine (FM) to predict daily data of the Chinese stock market index.

This paper extends all these previous approaches by also including higher-order interactions to predict high-frequency returns based on the S&P 500. We make the following main contributions to the existing literature. First, we present an application of higher-order FMs outside the recommender domain. Second, we introduce the adaptive-order algorithm, which determines all hyperparameters required by the higher-order FM model. Third, we provide the first study combining information from social media with high-frequency data to forecast the short-term development of stocks. Fourth, we benchmark the strategies employing FM models with a strategy using SVMs as well as a naive buy-and-hold strategy. Fifth, analyzing the S&P 500 stock constituents in the years 2014 and 2015, we demonstrate the our FM-based approaches are able to attain positive returns in a highly liquid market for a recent time period.

The remainder of this paper is structured as follows. Section 2 presents the concept of FMs. In Section 3, we develop the adaptive-order algorithm. Section 4 briefly describes our data and the software used. Section 5 provides the methodology of our back-testing framework. In Section 6, we present the results and discuss our key findings. Finally, Section 7 concludes this work and

provides directions for further research.

2. Previous work on Factorization Machines

As a supervised machine learning approach, FMs extract structure from available data (model training) and use the obtained information to predict or classify new data (model application). During the model training phase, the FM approach optimizes a specific statistical model based on a set of n training examples containing information about p features ($n, p \in \mathbb{N}$). All training examples are represented by a feature matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and a corresponding target vector $\boldsymbol{y} \in \mathbb{R}^n$. Afterwards, the extracted model is applied to new cases for which only the feature information is available, and the corresponding target values are predicted.

In the following, we describe the development of the FM approach, from the originally-proposed second-order FMs and their application (see Section 2.1) to the recent work on higher-order FMs (see Section 2.2). The corresponding models are defined based on one training example reflected by a feature vector $\boldsymbol{x} \in \mathbb{R}^p$ (one row of the feature matrix $\boldsymbol{X}$) and a corresponding target value $y \in \mathbb{R}$ (one value of the target vector $\boldsymbol{y}$).

2.1. Second-order Factorization Machine

2.1.1. Model

The second-order FM model introduced by Rendle (2010) factorizes all pairwise interactions between features based on a matrix $\boldsymbol{V}^{(2)}$ which contains parameters to weight the interactions.[1] The estimated target value $\hat{y}$ given $\boldsymbol{x}$ is defined as

$$\hat{y}(\boldsymbol{x}) := v^{(0)} + \sum_{j_1=1}^{p} x_j v_{j_1}^{(1)} + \sum_{j_1=1}^{p} \sum_{j_2=j_1+1}^{p} x_{j_1} x_{j_2} \sum_{f=1}^{k_2} v_{j_1,f}^{(2)} v_{j_2,f}^{(2)}, \tag{1}$$

with $v^{(0)} \in \mathbb{R}$, $\boldsymbol{v}^{(1)} \in \mathbb{R}^p$, and $\boldsymbol{V}^{(2)} \in \mathbb{R}^{p \times k_2}$. The model parameters fulfill different roles: one parameter captures the global intercept ($v^{(0)}$), p parameters capture a weight for each feature ($v_1^{(1)}, \ldots, v_p^{(1)}$), and $p \cdot k_2$ parameters capture the second-order interactions between the features ($v_{1,1}^{(2)}, \ldots, v_{1,k_2}^{(2)}, \ldots, v_{p,1}^{(2)}, \ldots, v_{p,k_2}^{(2)}$).

The hyperparameter $k_2 \in \mathbb{N}$ determines the number of values factorizing the weights of the second-order interactions and thus the degree of generalization of the FM. Typically, an FM model contains less parameters (pk_2) than a naive polynomial regression (p^2) because $k_2 \ll p$.

[1] We slightly adapt the model formulation to allow its simple extension to higher orders.

According to Rendle (2012b), the key benefit of this model is that the second-order term can be calculated in linear time complexity with p. Model training and application are thus very efficient. Furthermore, the result of the model is only affected by features that are non-zero, which significantly increases efficiency when working with sparse feature matrices.

2.1.2. Applications

Initially, Rendle (2010) described the FM approach based on an example in the context of recommender algorithms. Therefore, many of the applications of second-order FMs are related to this domain: Pan et al. (2015) used seconder-order FMs for collaborative filtering based on both explicit feedback (ratings) and implicit feedback (mouse clicks). They assumed the compressed knowledge of user homophily and item correlation to be similar. Consequently, they developed a two-step algorithm involving the mining of compressed knowledge and the integration of this knowledge into a second-order FM, which they referred to as Compressed Knowledge Transfer via FM. Hong et al. (2013) predicted user interests and individual decisions in tweets. Since they had to cope with a pervasive cold-start problem, they modeled the interests of users exploiting rich information features, including textual content, based on so-called Co-FMs. Loni et al. (2014) studied the problem of improving recommendations by transferring knowledge from an auxiliary domain (e.g., songs) into a target domain (e.g., videos). In a simulation study of this so-called cross-domain collaborative filtering, second-order FMs attained the best results of all considered algorithms. Yan et al. (2015) employed Field-aware FMs for the RecSys 2015 Contest and achieved the third-best result in an e-commerce item recommendation problem. Their method can be divided into three steps: (1) mapping the top-N recommendation task to a binary classification problem; (2) using second-order FMs and gradient boosting decision trees to obtain derived features; (3) building an ensemble of two second-order FM models trained on different feature sets to obtain recommendations.

Since it is also possible to employ the FM approach to solve problems outside the recommender domain, there are some publications using second-order FMs as a general predictor in the area of machine learning. Dai et al. (2015) analyzed algorithms for predicting novel drug indications based on drug-disease associations. To this end, they also proposed a specific matrix factorization model. In comparison to this tailored approach, second-order FMs as a general predictor showed quite similar results, with the added benefit that they do not require any adaption to the specific problem at hand. Oentaryo et al. (2014) developed an approach called Hierarchical Importance-

aware FM to predict the response behavior in mobile advertising. Response data often suffer from the cold-start problem, and the predictions need to include cost-varying instances. Therefore, the authors included importance-aware and hierarchical learning mechanisms into the second-order FM approach. Tsai et al. (2015) analyzed whether the community related to an individual in a social network influences his or her reputation. They utilized second-order FMs to infer the latent social influence between authors from the patterns of their collaborations in scientific papers. The goal of this approach was to rank the reputation of top scientists based on their network of collaborations with other scientists. Chen et al. (2014) used second-order FMs to exploit data from a Chinese social network to forecast the Shanghai Composite Index. Their text mining approach especially benefited from the good performance of FMs on sparse feature data.

2.2. Higher-order Factorization Machine

Rendle (2010) also presented the higher-order FM model (referring to it as the "d-way Factorization Machine"), but he failed to provide any insights into efficient training algorithms. It was not until 2016 that Knoll (2016) as well as Blondel et al. (2016) found a way to train the higher-order FM model in an efficient way. Finally, Grottke and Knoll (2017) described a non-recursive approach to formulating higher-order FM terms with linear complexity. The higher-order FM model factorizes all interactions up to the dth order ($d \in \mathbb{N}$), and it is defined as follows:

$$\hat{y}(\boldsymbol{x}) := v^{(0)} + \sum_{l=1}^{d} \sum_{j_1=1}^{p} \dots \sum_{j_l=j_{l-1}+1}^{p} \Big(\prod_{m=1}^{l} x_{j_m} \Big) \Big(\sum_{f=1}^{k_l} \prod_{m=1}^{l} v_{j_m,f}^{(l)} \Big), \tag{2}$$

with $v^{(0)} \in \mathbb{R}$, $\boldsymbol{V}^{(l)} \in \mathbb{R}^{p \times k_l}$, and $k_l \in \mathbb{N}_0$. The hyperparameter k_l is usually referred to as the number of lth-order factors, the number of factors of order l, or the lth-order dimensionality. The linear weights for each feature – contained in the vector $\boldsymbol{v}^{(1)} \in \mathbb{R}^p$ in Equation (1) – are now included in the expression for the higher-order terms, by defining $\boldsymbol{V}^{(1)} \in \mathbb{R}^{p \times k_1}$ with the restriction of $k_1 \in \{0, 1\}$. In general, the term $\sum_{f=1}^{k_l} \prod_{m=1}^{l} v_{j_m,f}^{(l)}$ gathers the interactions of order l, from $l = 1$ for linear weights to $l = d$ for interactions of the highest order included.

To identify a higher-order FM with a specific structure, we use the notation $FM(k_1, k_2, \dots, k_d)$; while the first hyperparameter k_1 shows whether linear weights are used in the model (1) or not (0), the following hyperparameters determine the number of factors for the second to the dth order. For instance, $FM(1, 5, 0, 2, 1)$ identifies a fifth-order FM with linear weights, 5 second-order factors, no third-order factor, 2 fourth-order factors, and 1 fifth-order factor.

2.3. Learning methods

For optimizing the parameters of the second-order FM model, Rendle (2010) originally introduced the Stochastic Gradient Descent method. Later, Rendle (2012a) enhanced it with an approach which automatically controls for regularization. Furthermore, Rendle et al. (2011) proposed the coordinate descent method, sometimes referred to as alternating least squares. It served as a basis for the Markov Chain Monte Carlo (MCMC) learning method originally described by Freudenthaler et al. (2011). This approach attempts to estimate each model parameter with unknown mean and unknown precision using a Gibbs sampler based on a normal gamma hyperprior. In contrast to the aforementioned methods, it incorporates insensitive hyperparameters, and it thus does not require determining a learning rate nor setting any regularization values. An overview of several existing learning methods for training second-order FM models has been provided by Rendle (2012b).

3. Adaptive-order algorithm

Per se, higher-order FMs are able to factorize interactions of any arbitrary order; however, one has to specify the highest included order d as well as the values of the hyperparameters $k_2, \ldots, k_d$. An expert who wants to investigate a familiar data set could guess these values based on his/her experience using a trial-and-error approach. Another way to find these hyperparameters is a (out of sample) grid search over all possible parameters. Due to the large number of hyperparameters and the huge amount of possible values, this is usually very time consuming. In this section, we therefore present a novel three-step approach, called "adaptive-order algorithm", using cross validation to learn a higher-order FM with a suitable hyperparameter setting (see Algorithm 1).

Step A determines the highest included order d and a score r_l ($l \in \{2, \ldots, d\}$) reflecting the importance of each of the orders included. First, the data set $(\boldsymbol{X}, \boldsymbol{y})$ is split into 10 disjoint, equally-sized subsets, and a function is defined which returns the median of the root mean squared error (RMSE) of a (out of sample) 10-fold cross validation. Second, using the above-mentioned function, a loop evaluates different FMs starting with $FM(1)$, which is equivalent to a support vector machine with a linear kernel and serves as a standard of comparison. The FMs evaluated after that possess 1 factor at the respective highest order in addition to the linear weights ($FM(1,1)$, $FM(1,0,1)$, $FM(1,0,0,1)$, etc.). The loop ends when the median RMSE of the current FM is higher than the median RMSE of the FM evaluated before. At the end of Step A, we calculate a relevance score r_l for each order l from 2 to d, by subtracting the RMSE of $FM(1)$ from the RMSE obtained for the lth-order FM.

Algorithm 1 Adaptive-order algorithm

Input: data available during model training $\Rightarrow (\boldsymbol{X}, \boldsymbol{y})$

Output: higher-order FM with a suitable setting $\Rightarrow FM(k_1, \ldots, k_d)$

Step A – Determine d and the ratio between the values of $k_2, \ldots, k_d$

$eval_A$: function returning the median of the RMSEs of a 10-fold cross validation based on $(\boldsymbol{X}, \boldsymbol{y})$ for a specified FM (train model out of sample)

$d \leftarrow 1$;

$e_1 \leftarrow eval_A(FM(1))$;

loop

$k_l \leftarrow 0$ for $l \in \{1, \ldots, d+1\}$;

$k_1 \leftarrow 1$; $k_{d+1} \leftarrow 1$;

$e_{d+1} \leftarrow eval_A(FM(k_1, \ldots, k_{d+1}))$;

if $d > 0$ **and** $e_d < e_{d+1}$ **then break**;

$d \leftarrow d + 1$;

end loop

$r_l \leftarrow \max(e_l - e_1, 0)$ for $l \in \{2, \ldots, d\}$;

Step B – Determine h and the values of $k_2, \ldots, k_d$

$eval_B$: function returning the RMSE value of an 80/20 validation based on $(\boldsymbol{X}, \boldsymbol{y})$ for a specified FM (train model out of sample)

$h \leftarrow 1$;

$k_l \leftarrow \max(\lfloor \frac{r_l}{\sum_{i \in \{2,\ldots,d\}} r_i} + 0.5 \rfloor, 1)$ for $l \in \{2, \ldots, d\}$;

loop

$h \leftarrow h + 1$;

$k_l^* \leftarrow \max(\lfloor h \frac{r_l}{\sum_{i \in \{2,\ldots,d\}} r_i} + 0.5 \rfloor, 1)$ for $l \in \{2, \ldots, d\}$;

if $eval_B(FM(k_1, \ldots, k_d)) < eval_B(FM(k_1^*, \ldots, k_d^*))$ **then break**;

$k_l \leftarrow k_l^*$ for $l \in \{1, \ldots, d\}$;

end loop

Step C – Determine final model parameters

Optimize the model parameters of $FM(k_1, \ldots, k_d)$ based on the whole data set $(\boldsymbol{X}, \boldsymbol{y})$;

Step B compares various dth-order FMs with different numbers of model parameters and identifies the most favorable setting. Since the larger amounts of model parameters used here cause longer run times, we employ a (out of sample) validation with 80 percent of the data set $(\boldsymbol{X}, \boldsymbol{y})$ in the training subset and 20 percent of the data set $(\boldsymbol{X}, \boldsymbol{y})$ in the validation subset, rather than a 10-fold cross validation. Specifically, a loop evaluates FMs with different numbers of factors (controlled by h) but constant ratios between the numbers of factors at the various orders. Moreover, we set a minimum of 1 factor for each order included. The loop ends when the RMSE of the previously-considered FM with less factors is lower than the RMSE of the current FM.

At Step C, the final FM model specified by the hyperparameters $k_1, \ldots, k_d$ is trained based on the whole data set $(\boldsymbol{X}, \boldsymbol{y})$ that is available during the model training. The result of this final step is an optimized statistical model which reflects the structure found in the data set.

In our implementation of the adaptive-order algorithm, as well as for training all FMs in our simulations, we employ the MCMC learning method to optimize the parameters. Moreover, we use the same trivial insensitive method hyperparameters as Rendle (2012b). Regarding the run time of the method, we identify two key aspects: First, the run time is influenced by the total number of factors $h = \sum_{l=2}^{d} k_l$ – the higher this value, the more model parameters have to be optimized and thus the longer the runtime will be. Second, due to cache calculations, the run time depends on the specified order of these factors; for example, a third-order factor takes about twice as much time as a second-order factor to be optimized, and a fourth-order factor requires even thrice as much time.

One crucial benefit of the adaptive-order algorithm is that the 10-fold cross validation can be calculated in a parallel fashion. To compute the 80/20 validation in parallel, we calculate it in blocks of 10. If the FM which produces the lowest RMSE is not the one with largest number of model parameters in this block, then this FM is chosen. Otherwise, we proceed with the next block. Furthermore, we square r_l to increase the spread of the ratios, we multiply h by 2 to spread the number of model parameters evaluated, and we set $d \geq 3$ because we want to make sure that a higher-order (rather than a second-order) FM is chosen.

To validate our adaptive-order approach, we conduct a simulation study using a data set that has been created synthetically. The elements of the $5,000 \times 50$ matrix $\boldsymbol{X}$ have been set to either 0 or 1 in equal proportions. When generating the target vector $\boldsymbol{y} \in \mathbb{R}^{5,000}$, randomly-drawn interactions from the second to the fifth order (4 of each) were taken into account. To obtain valid results, the simulation makes use of a 100-fold cross-validation approach. Figure 1 shows the resulting boxplot. We compare the RMSE obtained by a grid search over FMs having either 0, 5, 10, 15, or 20 second-,

third-, fourth-, and fifth-order factors (a total of 625 combinations) with the RMSE achieved by running the adaptive-order algorithm.

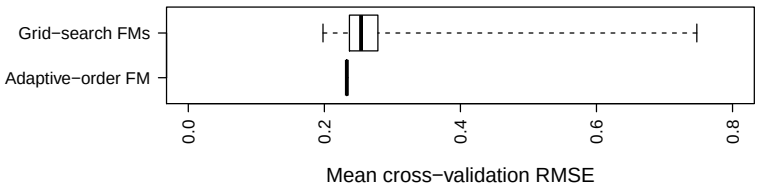


Figure 1: Comparison of the RMSE values of the grid search and the adaptive-order FM approach.

The most important point is that the RMSE of the adaptive-order FM is located in the first quartile of the RMSE values of the grid search or, to be more specific, 81.12 percent of the FMs examined during the grid search resulted in a higher mean cross-validation RMSE than our approach. This is even more impressive if we consider the fact that the adaptive-order FM takes only 0.16 percent of the time consumed by the conducted grid search. Furthermore, this simulation study shows the advantage of incorporating higher orders into the FM model: the best second-order FM is located at rank 614 out of 625. In summary, the adaptive-order algorithm does not exactly hit the optimum FM with respect to the mean cross-validation RMSE, but our approach performs strongly with respect to efficiency and feasibility.

4. Data and Software

The first part of our data set originates from Twitter, which is one of the largest news and social network services with 313 million active users and 1 billion visits to sites with embedded tweets per month (Twitter, 2017). Communication within this network is based on tweets, i.e., messages restricted to 140 characters. A tweet is approximately as long as a newspaper headline, and it thus contains the key information about a given topic. For our application, we procured the full archive of tweets concerning the S&P 500 companies from January 2014 to December 2015 directly from Twitter. Specifically, we obtained approximately 10 million tweets containing an official company name, for example “Apple Inc.”. Focusing on the official company names avoids falsely considering tweets which are not related to the financial market, such as the statement “The apple tastes good”. In addition, we received a time stamp with each tweet, specifying the second when it was created.

The second part of our data set are the stock prices of the S&P 500 index constituents from January 2014 to December 2015. This highly liquid subset of the U.S. stock market covers 80

percent of available market capitalization (S&P 500 Dow Jones Indices, 2015). Following Krauss and Stübinger (2017), we remove the survivor bias from our data by conducting a two-stage procedure. First, we use QuantQuote (2016) to produce a list of the S&P 500 constituents from December 2013 to December 2015. Then, this information is converted into a binary matrix, where "1" indicates that a stock was a constituent of the S&P 500 on the following day, while "0" represents the opposite case. Second, for all these index constituents, we download minute-by-minute data from January 2014 to December 2015 from QuantQuote (2016). Stock splits, dividends, and further corporate actions necessitate an adjustment of the data set. Combining both data sets leads to a complete replication of the S&P 500 index constituency and the appropriate price series.

The entire methodology in this paper and all relevant evaluations have been implemented in `R`, a statistical programming language (R Core Team, 2017). For text mining, we rely on the packages `Matrix` by Bates and Maechler (2016) and `tm` by Feinerer and Hornik (2017). For time series handling, we employ the packages `TTR` by Ulrich (2016) and `xts` by Ryan and Ulrich (2014). Most of the performance evaluation is conducted using the `PerformanceAnalytics` package by Peterson and Carl (2014). The core of our implementation is based on the `FactoRizationMachines` package originally developed to train third-order FM models (Knoll, 2016). We extend this package in line with the approach introduced by Grottke and Knoll (2017) to the higher orders and enhance it by the MCMC learning method.

5. Methodology

For our back-testing framework, we used 10 million tweets concerning the S&P 500 index constituents and their corresponding minute-by-minute stock prices from January 2014 to December 2015 (see Section 4). In the spirit of Jegadeesh and Titman (1993) and Gatev et al. (2006), we split our data into 464 overlapping study periods (see Figure 2). Each study period includes a 40-day formation period (Section 5.1) in which the model is in-sample trained and a 1-day out-of-sample trading period (Section 5.2). Typically, the S&P 500 index consists of 500 stocks, and for each stock there are 391 minute-by-minute data points per day. On average, we observe 14,000 tweets related to the S&P 500 constituents per day. Consequently, during one simulation run spanning the period from January 2014 to December 2015 we process approximately $500 \cdot 41 \cdot 391 \cdot 464 = 3,719,192,000$ stock prices and $41 \cdot 14,000 \cdot 464 = 266,336,000$ tweets. Note that most tweets were processed multiple times due to the overlapping nature of the study periods.

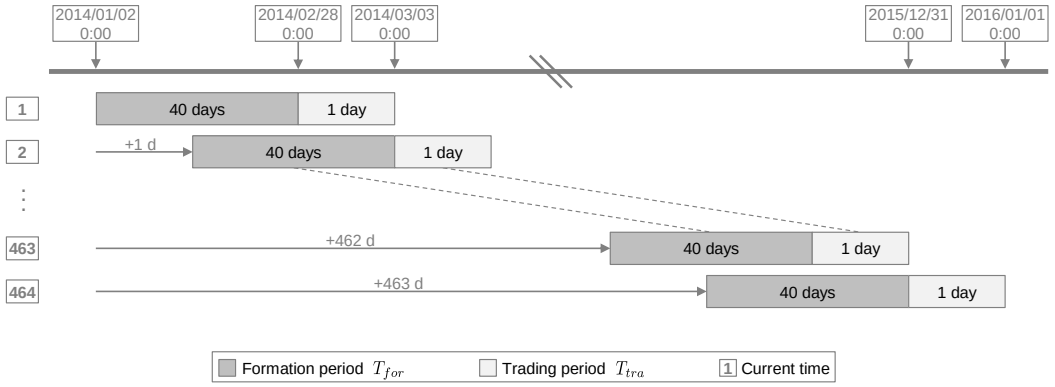


Figure 2: The back-testing framework deals with 464 overlapping study periods from January 2014 to December 2015. Each study period consists of a 40-day formation and a 1-day trading period.

5.1. Formation period

During the formation period T_{for} we aim to extract the pieces of information contained in the tweets that are suited to predict the future development of the stock market. Therefore, we create the document-term matrix, which serves as the feature matrix $\boldsymbol{X}$ in our simulation study. To build the document-term matrix, we follow seven steps: (1) We select the time stamp of the tweet, its text and the corresponding language from the original Twitter data set. (2) The tweets are allocated to the S&P 500 constituents using pattern matching, such that every tweet is assigned to at least one company. (3) We consider only tweets in English and transform them into lower-case form. (4) Some transformation functions are applied, eliminating uniform resource locators, punctuation makers, stop words, and numbers. (5) We restrict the data set to tweets which were created between 9.30 am and 4 pm on a trading day, to establish a relationship between tweets and future returns. (6) We stem the adapted tweets using Porter's algorithm (Porter, 1980), a standard procedure in text processing. (7) We create the document-term matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$, which describes the frequency of terms that arise in the collection of our tweets. The rows of $\boldsymbol{X}$ characterize the tweets and the columns describe the stemmed features. We use binary weights, i.e., element (i, j) of the document-term matrix indicates if tweet i includes the term j ("1") or not ("0"), $i \in \{1, \ldots, n\}, j \in \{1, \ldots, p\}$. The whole document-term matrix consists of $p = 238,637$ columns for all unique terms. For each study period, the matrix is reduced by omitting the columns related to terms not appearing in T_{for}. This ensures that only information available at a certain moment is fed into the trading algorithm.

After creating the document-term matrix, we match the tweets with the corresponding 20-

minute returns, which serve as the target vector $\boldsymbol{y}$ in our simulation study. To be more specific, each element of the target vector is represented by the adapted discrete 20-minute return, i.e., the relative change of the price from the minute the corresponding tweet has been created to 20 minutes in the future (see Figure 3). We choose 20 minutes following Gidofalvi and Elkan (2001) and Schumaker and Chen (2009a), who examined the forecast period based on minute-by-minute data.

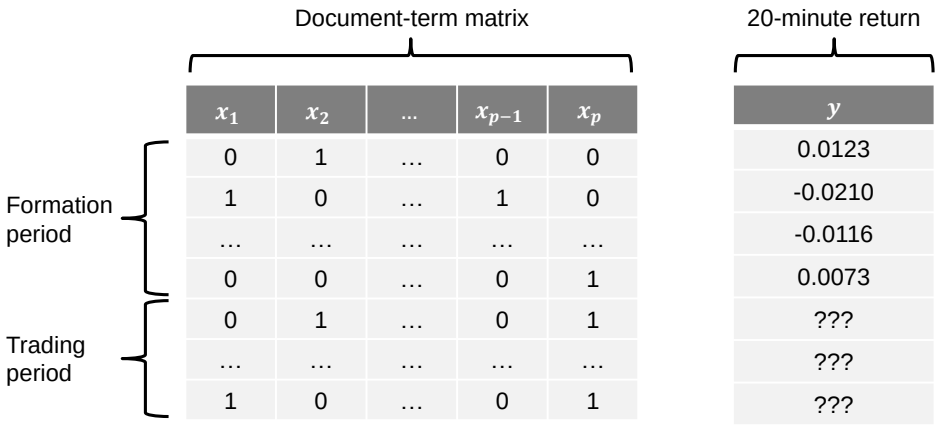


Figure 3: Document-term matrix $\boldsymbol{X}$ (containing information about the terms used in the tweets) and target vector $\boldsymbol{y}$ (reflecting the 20-minute returns).

To summarize, the ith row of the document-term matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ describes the stemmed terms $x_1, \ldots, x_p$ of the ith tweet, and the corresponding 20-minute return is the ith element of the target vector $\boldsymbol{y} \in \mathbb{R}^n$. To connect the document-term matrix with the future returns, different models can be employed. In our simulation study, we use three strategies based on FMs and one strategy using support vector machines:

- Strategy using support vector machines (SVM): This simple approach is based on the assumption that the relationship between stemmed terms and future returns can be explained by one global intercept and weights for each feature. The support vector machine with a linear kernel does not consider any interactions between the features, resulting in the model $\hat{y}(\boldsymbol{x}) := v^{(0)} + \sum_{j_1=1}^{p} x_j v_{j_1}^{(1)}$, where $v^{(0)} \in \mathbb{R}$, $\boldsymbol{v}^{(1)} \in \mathbb{R}^p$.
- Strategy based on second-order FMs (SFM): We expand the baseline approach by adding pairwise interactions between the features. Consequently, we obtain the model described in Equation (1) and set $k_2 = 1$. For more details see Section 2.1.

- Strategy employing third-order FMs (TFM): Enhancing the second-order FM model in Equation (1) by adding a term for third-order interactions results in the third-order FM model $\sum_{j_1=1}^{p}\sum_{j_2=j_1+1}^{p}\sum_{j_3=j_2+1}^{p} x_{j_1}x_{j_2}x_{j_3}\sum_{f=1}^{k_3} v_{j_1,f}^{(3)}v_{j_2,f}^{(3)}v_{j_3,f}^{(3)}$, where $\boldsymbol{V}^{(3)} \in \mathbb{R}^{p\times k_3}$. Like in our SFM strategy, we set $k_2 = k_3 = 1$.

- Strategy using FMs with the adaptive-order algorithm (AFM): Factorizing all interactions up to the dth order results in the higher-order FM model given by Equation (2). In our strategy we determine the highest included order d ($d \in \mathbb{N}$) and the number of factors $k_2, ..., k_d$ ($k_2, ..., k_d \in \mathbb{N}_0$) using Algorithm 1 presented in Section 3.

Following Gatev et al. (2006), we select the s most suitable target stocks ($s \in \mathbb{N}$), the so-called top stocks, for each strategy. To this end, we choose the s stocks which attain the lowest root relative squared error.[2] These top stocks are transferred to the trading period (see Section 5.2). Summarizing, each study period consists of the following three steps: (1) create the document-term matrix, (2) optimize the parameters of the different models, and (3) select the top stocks for the trading period.

5.2. Trading period

In the 1-day trading period T_{tra}, we consider the top s stocks selected in the formation period. For each stock and every newly-arriving price at minute t we determine the discrete minute-by-minute return z_t ($t \in T_{tra}$). We assume that tweets contain pieces of information that have a significant effect on stock returns in the future. If our assumption holds and the relationship between terms and future returns is reflected by the model, we are in the position to identify market inefficiencies. Increasing deviations of the target value from 0 depict higher chances of success. In times of market turmoil, the entry threshold should rise. We aim to capture these facts with a trading strategy based on Bollinger bands (Bollinger, 1992, 2001). Therefore, we calculate the running mean μ_t and standard deviation σ_t over a u-minute moving window:

$$\mu_t = \frac{1}{u}\sum_{i=0}^{u-1} z_{t-i}, \quad t \in T_{tra}, t-i \in T_{for} \uplus T_{tra}, \tag{3}$$

$$\sigma_t = \sqrt{\frac{1}{u}\sum_{i=0}^{u-1}(z_{t-i}-\mu_t)^2}, \quad t \in T_{tra}, t-i \in T_{for} \uplus T_{tra}. \tag{4}$$

[2] We consider only top stocks with a quorum of 80 tweets in the formation period, i.e., on average 2 tweets per day. This procedure avoids selecting stocks which are driven only by a few outliers.

Using these statistics, we determine the upper (lower) Bollinger band, which is defined to be located b standard deviations above (below) the moving mean.

If there are no tweets concerning the considered stock at minute t, we do not execute any trade. If there is a tweet at t, we predict the 20-minute return $\hat{y}$ using SVM, SFM, TFM, and AFM.[3] We define the following trading entry signals:

- $\sum_{i=0}^{u-1} z_{t-i} + \hat{y} > \mu_t + b \cdot \sigma_t$, i.e., the stock is undervalued. In consequence, we invest 1 USD in the stock and reverse the trade after 20 minutes.
- $\sum_{i=0}^{u-1} z_{t-i} + \hat{y} < \mu_t - b \cdot \sigma_t$, i.e., the stock is overvalued. In consequence, we sell short the stock with 1 USD and reverse the trade after 20 minutes.
- $\mu_t - b \cdot \sigma_t \leq \sum_{i=0}^{u-1} z_{t-i} + \hat{y} \leq \mu_t + b \cdot \sigma_t$, i.e., the stock is in its 'normal' region. In consequence, we do not execute any trade.

In accordance with of Avellaneda and Lee (2010), we hedge market exposure trade-by-trade with appropriated investments in the S&P 500 to pursue a classical long-short investment strategy. We generate a dollar-neutral portfolio to be in conformity with Gatev et al. (2006) and the literature.

For our short-term trading strategy, we follow Bollinger (1992, 2001) and calculate the running mean and standard deviation of the past $u = 10$ minutes. We choose $b = 2$, a value used by Avellaneda and Lee (2010) and Stübinger and Endres (2017), who intended to avoid high transaction costs in comparison with the $b = 1.5$ suggested by Bollinger (1992, 2001).

Following Gatev et al. (2006), the overall return is calculated on employed capital, the most common metric in financial literature. Specifically, we divide the sum of daily payoffs across the portfolio by the number of stocks that actually open during the trading period. Following Prager et al. (2012), we pragmatically assume transaction costs of 2 basis points per share per half turn. This assumption seems realistic given our high turnover strategy in a highly liquid stock market based on minute-by-minute data from 2014 to 2015.

6. Results

Following Huck (2009, 2010) and Krauss et al. (2017), we run a full-fledged performance evaluation of the strategies involving FMs (AFM, TFM and SFM) in comparison with the simple

[3] If $t + 20 \notin T_{tra}$, then we do not execute any trade.

SVM-based strategy and a naive S&P 500 buy-and-hold strategy (MKT). In the following sections, we analyze the top $s = 5$ stocks of each strategy for each study period in 2014 and 2015.

First, we evaluate risk-return characteristics and trading statistics for each strategy (Section 6.1). Most of the selected performance metrics have been explained by Bacon (2008). Second, we focus on AFM and examine the exposure to common systematic factors (Section 6.2), perform a bootstrap trading, and check the robustness of our strategy (Section 6.3). Third, we check the influence of specific terms regarding the future returns (Section 6.4) and investigate which orders are chosen when using the AFM approach (Section 6.5).

6.1. Strategy performance

Table 1 describes daily return characteristics for the top 5 stocks per strategy from March 2014 until December 2015, relative to the S&P 500 buy-and-hold benchmark strategy. We observe statistically significant returns for AFM, TFM, and SFM with Newey-West (NW) t-statistics above 4.90 prior to transactions costs. Even after transaction costs, this statement remains true for AFM (3.37) and TFM (2.08). As expected, SVM generates a negative return of -0.04 percent per day after transaction costs – a statistically significant result. We see that AFM returns 0.11 percent per day prior to transaction costs and 0.06 percent after transaction costs. Compared to TFM, SFM, SVM, and the S&P 500, AFM outperforms the rigid approaches and the naive market strategy. Also, the standard deviation of the S&P 500 is approximately two times higher than the ones of the strategies involving FMs. In contrast to the general market, all strategies exhibit both high kurtosis before and right skewness after transaction costs – a pleasant characteristic for potential investors. In line with the methodology due to Mina and Xiao (2001), we report historical Value at Risk (VaR) levels. We see that the tail risk is much less compared to an investment in the S&P 500; for example, the historical VaR 1 % is -0.90 percent for AFM versus -2.12 percent for the buy-and-hold strategy. We observe a similar picture for the maximum drawdown level of 4.81 percent for AFM, compared to 12.63 percent for the benchmark. The hit rate, i.e., the percentage of days with non-negative returns, of AFM clearly outperforms the market with approximately 61.6 percent after transactions cost, compared to 52.2 percent of the market.

	Before transaction costs				After transaction costs				
	AFM	TFM	SFM	SVM	AFM	TFM	SFM	SVM	MKT
Mean return	0.0011	0.0008	0.0008	0.0001	0.0006	0.0003	0.0003	-0.0004	0.0002
Standard error (NW)	0.0002	0.0002	0.0002	0.0001	0.0002	0.0002	0.0002	0.0001	0.0004
t-Statistic (NW)	5.7574	4.9031	4.2413	0.9119	3.3658	2.0754	1.7332	-3.7549	0.5279
Minimum	-0.0131	-0.0162	-0.0134	-0.0090	-0.0153	-0.0170	-0.0138	-0.0094	-0.0402
Quartile 1	-0.0005	-0.0005	-0.0007	-0.0004	-0.0010	-0.0010	-0.0013	-0.0010	-0.0042
Median	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003
Quartile 3	0.0024	0.0021	0.0020	0.0007	0.0019	0.0014	0.0014	0.0001	0.0048
Maximum	0.0398	0.0340	0.0340	0.0135	0.0385	0.0336	0.0336	0.0131	0.0383
Standard deviation	0.0044	0.0041	0.0040	0.0020	0.0044	0.0040	0.0039	0.0020	0.0086
Skewness	2.8209	1.9386	2.0625	0.9520	2.7008	1.8575	2.0940	0.2796	-0.2803
Kurtosis	18.5605	14.1759	13.9471	10.0704	18.3778	15.2340	14.7624	9.0761	2.2883
Historical VaR 1 %	-0.0085	-0.0093	-0.0090	-0.0058	-0.0090	-0.0097	-0.0090	-0.0071	-0.0212
Historical CVaR 1 %	-0.0112	-0.0124	-0.0107	-0.0075	-0.0122	-0.0133	-0.0108	-0.0085	-0.0300
Historical VaR 5 %	-0.0040	-0.0045	-0.0044	-0.0028	-0.0045	-0.0048	-0.0047	-0.0036	-0.0144
Historical CVaR 5 %	-0.0067	-0.0073	-0.0069	-0.0045	-0.0073	-0.0078	-0.0072	-0.0054	-0.0200
Maximum drawdown	0.0274	0.0327	0.0399	0.0312	0.0481	0.0837	0.0748	0.1625	0.1263
Share with return ≥ 0	0.6724	0.6681	0.6724	0.6681	0.6164	0.5819	0.5991	0.5733	0.5216

Table 1: Daily return characteristics for the top 5 stocks of AFM, TFM, SFM, and SVM compared to an S&P 500 long-only benchmark (MKT) from March 2014 until December 2015. NW denotes Newey-West standard errors with one-lag correction, and CVaR denotes the Conditional Value at Risk.

Table 2 depicts statistics on trading frequency, which bear a similar pattern for all strategies. Approximately 1.4 of the 5 considered stocks are traded per day across all systems. This relatively low number is based on the fact that a trade is opened only if there exists a tweet in the previous minute and if the prediction, based on this tweet, is outside the Bollinger bands. On average, the strategies perform 0.5 round-trip trades per 1-day trading period. The similarity across all strategies is most likely caused by the fact that they all employ the Bollinger bands. However, the return characteristics exhibit substantial differences caused by the considered extent of interactions between the features.

	AFM	TFM	SFM	SVM
Average number of target stocks traded per 1-day period	1.3772	1.4763	1.4224	1.2629
Average number of round-trip trades per target stock	0.3914	0.4470	0.4272	0.4427
Standard deviation of number of round-trip trades per target	0.7671	0.9299	0.8671	1.0165

Table 2: Trading statistics for the top 5 stocks of AFM, TFM, SFM, and SVM from March 2014 until December 2015, per 1-day trading period.

In Table 3, the annualized risk-return measures for all strategies are summarized. After transaction costs, AFM produces 16.77 percent p.a., compared to 8.76 percent for TFM, 7.85 for SFM, and -8.72 percent for SVM. Thus, the strategies involving FMs clearly outperform the market with an average return of 4.72 percent p.a.. For all strategies, the mean excess return[4] is identical to the mean return because the risk-free rate equals zero in the years 2014 and 2015. AFM attains approximately half the standard deviation of the market, leading to a Sharpe ratio of 2.42 for AFM after transaction costs. This tendency is reinforced when considering only the negative return deviations from the mean: The downside deviation for AFM (2.94 percent) is significantly lower compared to the S&P 500 (9.77 percent). Consequently, the Sortino ratio, scaling returns by their downside deviation, is at 5.06 percent for AFM.

	Before transaction costs				After transaction costs				
	AFM	TFM	SFM	SVM	AFM	TFM	SFM	SVM	MKT
Mean return	0.3075	0.2287	0.2128	0.0212	0.1677	0.0876	0.0785	-0.0872	0.0472
Mean excess return	0.3075	0.2287	0.2128	0.0212	0.1677	0.0876	0.0785	-0.0872	0.0472
Standard deviation	0.0705	0.0644	0.0630	0.0315	0.0695	0.0630	0.0615	0.0318	0.1365
Downside deviation	0.0294	0.0322	0.0310	0.0202	0.0332	0.0356	0.0340	0.0254	0.0977
Sharpe ratio	4.3620	3.5503	3.3797	0.6711	2.4150	1.3916	1.2763	-2.7418	0.3453
Sortino ratio	10.4547	7.1067	6.8655	1.0468	5.0557	2.4646	2.3096	-3.4303	0.4828

Table 3: Annualized risk-return measures for the top 5 stocks of AFM, TFM, SFM, and SVM compared to a S&P 500 long-only benchmark (MKT) from March 2014 until December 2015.

Academic literature about statistical arbitrage depicts fluctuating performance over time, so we perform a sub-period analysis in accordance with Do and Faff (2010), Bowen and Hutchinson (2016) and Krauss et al. (2017). Figure 4 describes the development of an investment of 1 USD before transaction costs (left) and after transaction costs (right) for all strategies, comparing them with an S&P 500 long-only benchmark (MKT). The S&P 500 shows strong variance and large drawdowns, such as the fear over spread of Ebola in October 2014 and the unsettledness about rate hikes of the Federal Reserve in August 2015. The simple SVM-based strategy produces non-mentionable returns – when considering transaction costs there is even a declining development of the invested 1 USD. In comparison, the stratgies employing FMs fare much better. For AFM, we observe that 1 USD invested in March 2014 grows to above 1.60 USD before transaction costs and

[4]The mean excess return is defined as the average of the difference between the return of the strategy and the risk-free rate.

to above 1.30 USD after transaction costs. Performance does not decline across time and seems to be robust against drawdowns. As expected, TFM and SFM exhibit a similar behavior, which is clearly worse than the one of AFM. So far, no academic study has presented a statistical arbitrage strategy with increasing development in recent time periods – see, for example, Avellaneda and Lee (2010) and Rad et al. (2016). As we have seen, especially the trading strategy based on AFM outperforms classic approaches in a multitude of comparisons. Therefore, detailed results of AFM will be examined in the following subsections.

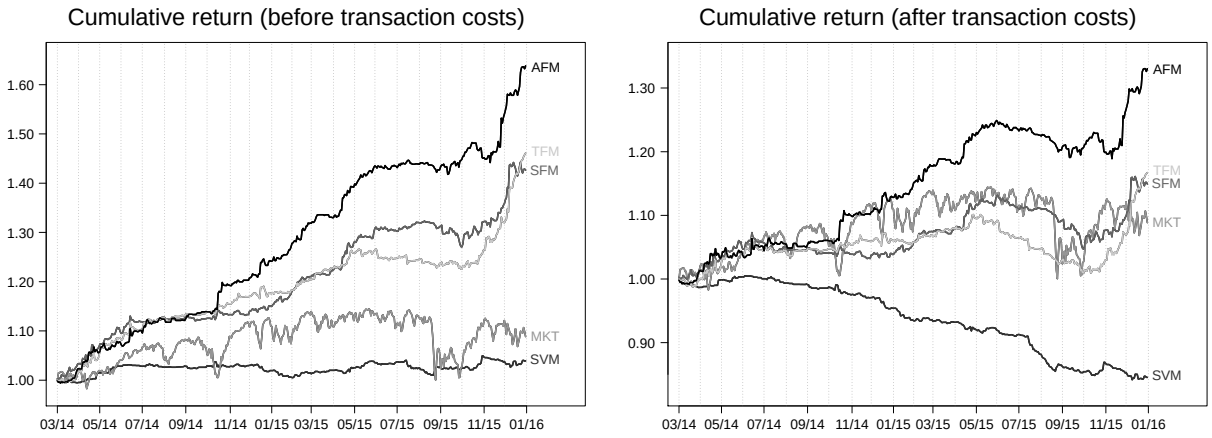


Figure 4: Development of an investment of 1 USD for the top 5 stocks of AFM, TFM, SFM, and SVM before transaction costs (left) and after transaction costs (right) compared to an S&P 500 long-only benchmark (MKT) from March 2014 until December 2015.

To benchmark the different strategies with respect to the run time required for coming up with decisions, we measure the time one study period consumes with our implementation running on a Intel Xenon CPU E5-2665 0 @ 2.40 GHz. The results in Table 4 are hardly surprising: The fastest strategy is SVM, followed by SFM and TFM; AFM is by far the slowest approach. This is pretty reasonable because the adaptive-order algorithm includes several model training steps with a higher number of factors. In our implementation these steps are parallelized; if we run the cross-validation steps of the algorithm without a parallel implementation the run time increases to 1340.52 seconds. However, the most important finding in this analysis is that all run times are clearly within the limits set by trading reality: The run time for the model training, which is carried out each night, needs to be below 63,000 seconds (the time period between 4 pm and 9.30 am), while the one for model application should not exceed a few seconds (due to the minute-by-minute frequency of the incoming data). Even AFM is thus a feasible strategy.

	AFM	TFM	SFM	SVM
Model training (formation period)	223.42	32.61	29.33	20.82
Model application (trading period)	0.41	0.33	0.15	0.08

Table 4: Run time statistics for AFM, TFM, SFM, and SVM in seconds.

6.2. Common risk factors

Table 5 evaluates the exposure of AFM for the top 5 stocks after transaction costs to common sources of systematic risk. Following Krauss and Stübinger (2017), we perform the Fama-French 3-factor model (FF3) covered by Fama and French (1996), the Fama-French 3+2-factor model (FF3+2) discussed by Gatev et al. (2006), and the Fama-French 5-factor model (FF5) outlined by Fama and French (2015). The first model measures exposure to overall market, small minus big market capitalization (SMB), as well as high minus low book-to-market stock (HML). The second model augments the first one by capturing a momentum and a short-term reversal factor. The third model extends the first model by adding the factors robust minus weak (RMW) profitability and conservative minus aggressive (CMA) investment behavior. We downloaded the data required to compute these factor models from Kenneth French's website.[5] We observe small and insignificant loadings for MKT, SMB, momentum, reversal, RMW, and CMA – driven by the long-short portfolio we are constructing. Loading on HML is small but significant and possesses the expected positive sign. The FF3+2 model shows the highest explanatory power with an adjusted R^2 of 0.0075. Overall, AFM generates statistically significant and economically substantial daily intercept alphas of 0.07 percent, after transaction costs. These results suggest that AFM does not load on any common sources of systematic risk, while it outperforms classic approaches like the one using SVMs.

[5]We thank Kenneth R. French for providing all relevant data for these models on his website http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/.

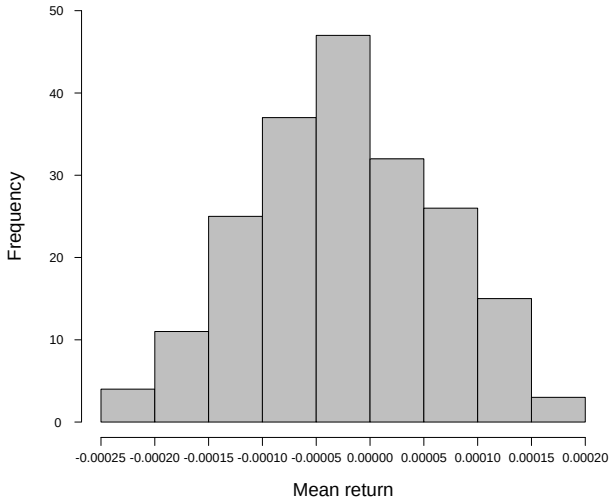
	FF3	FF3+2	FF5
(Intercept)	0.0007**	0.0007**	0.0007**
	(0.0002)	(0.0002)	(0.0002)
Market	−0.0007	−0.0139	0.0051
	(0.0236)	(0.0254)	(0.0250)
SMB	0.0330	0.0178	
	(0.0400)	(0.0409)	
HML	0.1055*	0.0717	
	(0.0463)	(0.0561)	
Momentum		−0.0277	
		(0.0348)	
Reversal		0.0489	
		(0.0374)	
SMB5			0.0290
			(0.0450)
HML5			0.0533
			(0.0645)
RMW5			−0.0450
			(0.0833)
CMA5			0.1327
			(0.1170)
R^2	0.0114	0.0182	0.0147
Adj. R^2	0.0049	0.0075	0.0039
Num. obs.	464	464	464
RMSE	0.0044	0.0044	0.0044

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 5: Exposure to systematic sources of risk after transaction costs for the daily returns of the top 5 stocks of AFM from March 2014 to December 2015. Standard errors are given in parentheses.

6.3. Robustness checks

Following Gatev et al. (2006), we compare the results of AFM with 200 sets of bootstrapped random tradings. Specifically, the entry signals of AFM are connected with prices of random stocks from the S&P 500 at the appropriated time. Figure 5 describes the daily return characteristics of bootstrap random trading before transaction costs. As expected, the average daily return of -0.002 percent is very close to zero, compared to the return of AFM of 0.11 percent (see Table 1). This value corresponds to the monthly bootstrapping results by Gatev et al. (2006) and Stübinger et al. (2016). This finding suggests that the constructed FM reveals temporal deviations and captures customized trading rules.



	Bootstrap trading
Mean return	-0.00001845
Standard deviation	0.00008988
Median	-0.00002101

Figure 5: Daily return characteristics before transaction costs for the top 5 stocks of AFM of bootstrap trading from March 2014 until December 2015.

In Section 5, we set the length of the moving average to 10 minutes ($u = 10$) and the trading threshold to two standard deviation ($b = 2$), motivated by the literature. In order to avoid data snooping, we examine the robustness of our results in the light of these parameters. Table 6 varies both b and u and reports the annualized mean return and Sharpe ratio of AFM for the top 5 stocks, before and after transaction costs.

We see that our results before transaction costs exhibit strictly positive annualized returns up to 40.18 percent and satisfactory Sharpe ratios of up to 5.23. The optimal strategies seem to occur for lower levels of u. After transaction costs, lower values are located at lower levels of b because the higher trading frequency generates higher transaction costs which cannot be compensated by the achieved returns. For a minimum trading threshold of $b = 2$, annualized returns and Sharpe ratios still remain significantly positive. A lower u increases the information density of the considered past returns. This fact appears to be beneficial for the combination of high-frequency data and our trading strategy. In summary, our initial parameter settings clearly did not hit the optimum with respect to annualized return and Sharpe ratio, but they led to acceptable results.

		Before transaction costs					After transaction costs				
	$u \setminus b$	1	1.5	2	2.5	3	1	1.5	2	2.5	3
Return	5	0.1431	0.2857	0.4018	0.3140	0.2478	-0.0900	0.0793	0.2540	0.2244	0.1823
	10	0.1270	0.1948	0.3075	0.2841	0.1937	-0.1027	-0.0017	0.1677	0.2053	0.1535
	20	0.0888	0.0916	0.2015	0.1675	0.1210	-0.1332	-0.0866	0.0679	0.0893	0.0886
Sharpe ratio	5	2.4190	3.9804	5.2302	4.0337	3.1916	-1.5444	1.1224	3.3543	2.9331	2.3818
	10	2.0808	3.0252	4.3620	3.9318	2.7496	-1.7339	-0.0274	2.4150	2.8984	2.2224
	20	1.3950	1.3827	2.6270	2.1784	2.0946	-2.1498	-1.3340	0.8952	1.1777	1.5631

Table 6: Yearly returns and Sharpe ratios for a varying number of minutes to use in the window (u), and the multiple b of the standard deviation for the top 5 stocks of AFM from March 2014 until December 2015.

6.4. Term analysis

To identify the terms driving the returns of the AFM approach, we create the word cloud in Figure 6 containing the 100 terms[6] which lead to the highest returns in our simulation. The size of a term reflects the return achieved by our trading strategy based on tweets containing the respective term. For this analysis, we aggregate the returns over all trades; i.e., one term could have both a negative or a positive connotation in relation to the predicted return. For example, the term "buy" could be related to a positive stock price development at one trading day and to a negative one at another.

At first sight, there are many terms in the word cloud which either could be considered as suggestions for trading behavior (e.g., "buy", "sell", "hold") or classify stocks as positive or negative (e.g., "upgrade", "downgrade", "top"). We use black to emphasize these terms in the word cloud because they include clear trading recommendations. Additionally, several other terms are related to a branch (e.g., "energy", "technology", "communication") or to the common domain of stock trading (e.g., "news", "stock", "report"). In general, the word cloud supports our assumption that the tweets contain information on how the stock prices will develop in the future.

[6]We exclude names or stock symbols of the S&P 500 constituents in our analysis.

Figure 6: Word cloud based on the return weighted terms of AFM without company names and company related stock symbols.

6.5. Dimensionality of Factorization Machines

To examine the influence of the factors per order obtained by running the adaptive-order algorithm, we analyze the different FM models optimized for the 464 trading periods in Figure 7. The left plot shows the distribution of the highest included order d among the FM models chosen, while the right one presents the number of FM models which contained a specific number of factors (k_l) for the corresponding order l.

Regarding the left plot, we see that the maximum value of the highest included order is 8. The most-frequently-chosen value is 4, followed by 3, 5, and 6; 7 and 8 are chosen very rarely. Furthermore, we note that the highest included order is never 2 because the adaptive-order approach has been designed to return higher-order FM models. Starting with order 4, this plot shows a decreasing frequency for higher orders, which is in line with the attempt of the adaptive-order algorithm to use lower rather than higher orders since this is more time-efficient.

The plot on the right-hand side reveals that the number of factors chosen most frequently is 1, no matter which order we look at. This might be because for each order lower than d the adaptive-order algorithm sets the minimum number of factors to 1. In combination with the algorithm producing higher-order FM models, this explains the eye-catching large bar for 1 third-order factor: if the

second-order factor in Step A of the algorithm obtains a good result, 1 third-order factor is added to the FM model to be in line with the higher-order requirement. We also see a tendency to use a lower rather than a higher number of factors, which is reflected in the design of the adaptive-order algorithm and improves time efficiency.

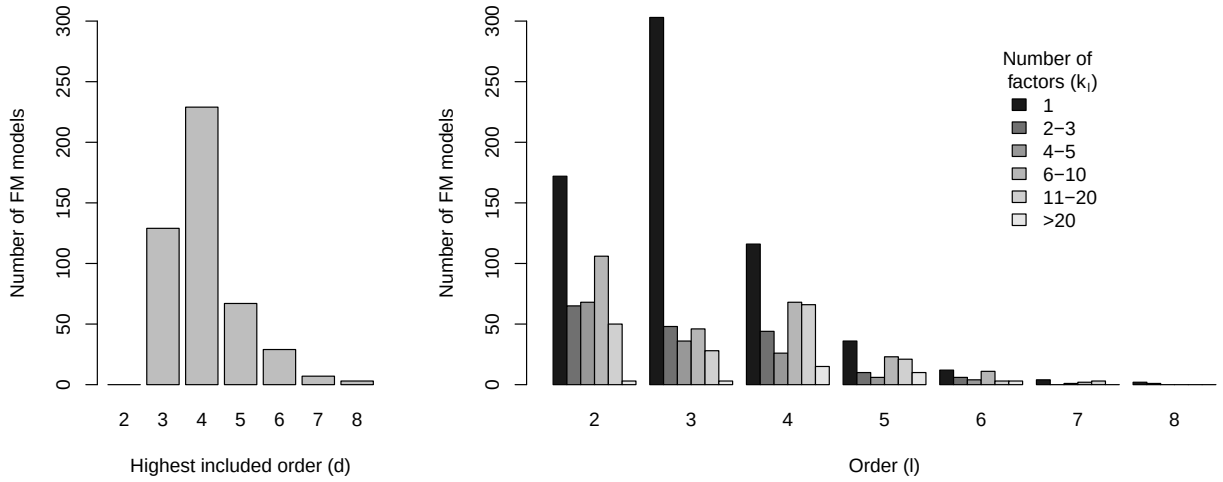


Figure 7: Analysis of the highest included order and the the factors per order obtained by running the adaptive-order algorithm for the 464 trading periods.

7. Conclusions

In this paper, we have proposed the adaptive-order algorithm for selecting the hyperparameters of higher-order FMs in a way that is much more feasible than a grid search. We have shown that strategies employing FMs for the prediction of high-frequency stock returns based on tweets fare rather well, resulting in positive returns even for recent years. At a return of 16.77 percent p.a. after transaction costs, the strategy making use of the adaptive-order algorithm has attained the best outcomes; its Sharpe ratio and Sortino ratio are also considerable higher than for other strategies analyzed. Moreover, it does not seem to load on any systematic sources of risk. While any strategy applying Bollinger bands requires the window size and the multiple of the standard deviations to be selected, we have seen that even the standard parameter settings taking from the literature led to highly acceptable results.

This study may serve as a starting point for future work. For example, it could be used as a blueprint for research applying FMs on different stock indices or employing other machine learning algorithms. Moreover, further financial and economic data (e.g., stock indices, exchange rates, or commodity prices) could be incorporated as features into the FM model, benefitting from the time structure between the added features and the stock returns. Finally, higher-order FMs might be a promising tool for extracting knowledge from text sources in other areas of application, such as classifying websites, or predicting the success of online advertisements.

Acknowledgements

This research was supported by the GfK Verein e. V., which funded the purchase of the Twitter data set. We are especially grateful to Raimund Wildner and Holger Dietrich for their commitment and effort over the course of this project. Furthermore, we would like to thank Ingo Klein, Christopher Krauß, and Benedikt Mangold for many helpful discussions.

Bibliography

Antweiler, W., Frank, M. Z., 2004. Is all that talk just noise? The information content of internet stock message boards. The Journal of Finance 59 (3), 1259–1294.

Avellaneda, M., Lee, J.-H., 2010. Statistical arbitrage in the US equities market. Quantitative Finance 10 (7), 761–782.

Bacon, C. R., 2008. Practical portfolio performance: Measurement and attribution, 2nd Edition. Wiley Finance. Wiley, Chichester, England.

Bates, D., Maechler, M., 2016. Matrix: Sparse and dense matrix classes and methods.

Blondel, M., Fujino, A., Ueda, N., Ishihata, M., 2016. Higher-order Factorization Machines. In: Proceedings of Neural Information Processing Systems.

Bollinger, J., 1992. Using Bollinger bands. Stocks & Commodities 10 (2), 47–51.

Bollinger, J., 2001. Bollinger on Bollinger bands. McGraw Hill Professional.

Bowen, D. A., Hutchinson, M. C., 2016. Pairs trading in the UK equity market: Risk and return. The European Journal of Finance 22 (14), 1363–1387.

Chen, C., Dongxing, W., Chunyan, H., Xiaojie, Y., 2014. Exploiting social media for stock market prediction with Factorization Machine. In: Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies. pp. 142–149.

Dai, W., Liu, X., Gao, Y., Chen, L., Song, J., Di Chen, Gao, K., Jiang, Y., Yang, Y., Chen, J., 2015. Matrix factorization-based prediction of novel drug indications by integrating genomic space. Computational and Mathematical Methods in Medicine 2015.

Do, B., Faff, R., 2010. Does simple pairs trading still work? Financial Analysts Journal 66 (4), 83–95.

Fama, E. F., French, K. R., 1996. Multifactor explanations of asset pricing anomalies. The Journal of Finance 51 (1), 55–84.

Fama, E. F., French, K. R., 2015. A five-factor asset pricing model. Journal of Financial Economics 116 (1), 1–22.

Feinerer, I., Hornik, K., 2017. tm: Text Mining Package.

Freudenthaler, C., Schmidt-Thieme, L., Rendle, S., 2011. Bayesian Factorization Machines. University of Hildesheim.

Gatev, E., Goetzmann, W. N., Rouwenhorst, K. G., 2006. Pairs trading: Performance of a relative-value arbitrage rule. Review of Financial Studies 19 (3), 797–827.

Gidofalvi, G., Elkan, C., 2001. Using news articles to predict stock price movements. Department of Computer Science and Engineering, University of California, San Diego.

Groth, S. S., Muntermann, J., 2011. An intraday market risk management approach based on textual analysis. Decision Support Systems 50 (4), 680–691.

Grottke, M., Knoll, J., 2017. Higher-order Factorization Machines: A non-recursive approach. Submitted.

Hong, L., Doumith, A., Davison, B., 2013. Co-Factorization Machines: Modeling user interests and predicting individual decisions in Twitter. In: Proceedings of the 6th ACM International Conference on Web Search and Data Mining. pp. 557–566.

Huck, N., 2009. Pairs selection and outranking: An application to the S&P 100 index. European Journal of Operational Research 196 (2), 819–825.

Huck, N., 2010. Pairs trading and outranking: The multi-step-ahead forecasting case. European Journal of Operational Research 207 (3), 1702–1716.

Jegadeesh, N., Titman, S., 1993. Returns to buying winners and selling losers: Implications for stock market efficiency. The Journal of Finance 48 (1), 65–91.

Knoll, J., 2016. Recommending with higher-order Factorization Machines. In: Bramer, M., Petridis, M. (Eds.), Research and Development in Intelligent Systems XXXIII. Springer, Cham.

Krauss, C., Do, X. A., Huck, N., 2017. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. European Journal of Operational Research 259 (2), 689–702.

Krauss, C., Stübinger, J., 2017. Non-linear dependence modelling with bivariate copulas: Statistical arbitrage pairs trading on the S&P 100. Applied Economics 23 (1), 1–18.

Loni, B., Shi, Y., Larson, M., Hanjalic, A., 2014. Cross-domain collaborative filtering with Factorization Machines. In: Rijke, M., Kenter, T., Vries, A. P., Zhai, C., Jong, F., Radinsky, K., Hofmann, K. (Eds.), Proceedings of the 36th European Conference on Information Retrieval Research. Springer, Cham, pp. 656–661.

Lugmayr, A., Gossen, G., 2013. Evaluation of methods and techniques for language based sentiment analysis for DAX 30 stock exchange – A first concept of a "LUGO" sentiment indicator. International SERIES on Information Systems and Management in Creative eMedia (1), 69–76.

Mina, J., Xiao, J. Y., 2001. Return to RiskMetrics: The evolution of a standard. RiskMetrics Group.

Nassirtoussi, A. K., Aghabozorgi, S., Wah, T. Y., Ngo, D. C. L., 2014. Text mining for market prediction: A systematic review. Expert Systems with Applications 41 (16), 7653–7670.

Oentaryo, R., Lim, E., Low, J., Lo, D., Finegold, M., 2014. Predicting response in mobile advertising with hierarchical importance-aware Factorization Machine. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining. pp. 123–132.

Pan, W., Liu, Z., Ming, Z., Zhong, H., Wang, X., Xu, C., 2015. Compressed knowledge transfer via Factorization Machine for heterogeneous collaborative recommendation. Knowledge-Based Systems 85, 234–244.

Peramunetilleke, D., Wong, R. K., 2002. Currency exchange rate forecasting from news headlines. Australian Computer Science Communications 24 (2), 131–139.

Peterson, B. G., Carl, P., 2014. PerformanceAnalytics: Econometric tools for performance and risk analysis.

Porter, M. F., 1980. An algorithm for suffix stripping. Program 14 (3), 130–137.

Prager, R., Vedbrat, S., Vogel, C., Watt, E. C., 2012. Got liquidity? BlackRock Investment Institute.

QuantQuote, 2016. QuantQuote market data and software. https://www.quantquote.com.

R Core Team, 2017. R package base: A language and environment for statistical computing.

Rad, H., Low, R. K. Y., Faff, R. W., 2016. The profitability of pairs trading strategies: Distance, cointegration and copula methods. Quantitative Finance 16 (10), 1541–1558.

Rendle, S., 2010. Factorization Machines. In: Proceddings of the 10th International Conference on Data Mining. pp. 995–1000.

Rendle, S., 2012a. Factorization Machines with libFM. ACM Transactions on Intelligent Systems and Technology 3, 1–22.

Rendle, S., 2012b. Learning recommender systems with adaptive regularization. In: Proceedings of the 5th ACM International Conference on Web Search and Data Mining. pp. 133–142.

Rendle, S., Gantner, Z., Freudenthaler, C., Schmidt-Thieme, L., 2011. Fast context-aware recommendations with Factorization Machines. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 635–644.

Ryan, J. A., Ulrich, J. M., 2014. xts: eXtensible time series.

Schumaker, R. P., Chen, H., 2009a. A quantitative stock prediction system based on financial news. Information Processing & Management 45 (5), 571–583.

Schumaker, R. P., Chen, H., 2009b. Textual analysis of stock market prediction using breaking financial news: The AZFin text system. ACM Transactions on Information Systems (TOIS) 27 (2), 1–19.

Schumaker, R. P., Zhang, Y., Huang, C.-N., Chen, H., 2012. Evaluating sentiment in financial news articles. Decision Support Systems 53 (3), 458–464.

S&P 500 Dow Jones Indices, 2015. Equity S&P 500. http://us.spindices.com/indices/equity/sp-500.

Stübinger, J., Endres, S., 2017. Pairs trading with a mean-reverting jump-diffusion model on high-frequency data. FAU Discussion Papers in Economics, University of Erlangen-Nürnberg.

Stübinger, J., Mangold, B., Krauß, C., 2016. Statistical arbitrage with vine copulas. FAU Discussion Papers in Economics, University of Erlangen-Nürnberg.

Tsai, M.-F., Wang, C.-J., Lin, Z.-L., 2015. Social influencer analysis with Factorization Machines. In: Proceedings of the ACM Web Science Conference. pp. 50–51.

Twitter, 2017. https://twitter.com.

Ulrich, J., 2016. TTR: Technical trading rules.

Yan, P., Zhou, X., Duan, Y., 2015. E-commerce item recommendation based on field-aware Factorization Machine. In: Proceedings of the 2015 International ACM Recommender Systems Challenge. pp. 1–4.