Discussion Papers
in Economics

No. 14/2018

# Separating the signal from the noise – financial machine learning for Twitter

Matthias Schnaubelt
University of Erlangen-Nürnberg

Thomas G. Fischer
University of Erlangen-Nürnberg

Christopher Krauss
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/

# Separating the signal from the noise – financial machine learning for Twitter

Matthias Schnaubelt[a,1,*], Thomas Fischer[a,1], Christopher Krauss[a,1]

[a] *University of Erlangen-Nürnberg, Department of Statistics and Econometrics, Lange Gasse 20, 90403 Nürnberg, Germany*

## Abstract

Most statistical arbitrage strategies in the academic literature soley rely on price time series. By contrast, alternative data sources are of growing importance for professional investors. We contribute to bridging this gap by assessing the price-predictive value of more than nine million tweets on intraday returns of the S&P 500 constituents. For this purpose, we design a machine learning pipeline addressing specific challenges inherent to this task. At first, we engineer domain-specific features along three categories, i.e., directional indicators, relevance indicators and meta features. Next, we leverage a random forest to extract the relationship between these features and subsequent stock returns in a low signal-to-noise setting. For performance evaluation, we run a rigorous event-based backtesting study across all tweets and stocks. We find annualized returns of 6.4 percent and a Sharpe ratio of 2.2 after transaction costs. Finally, we illuminate the machine learning black box and unveil sources of profitability: First, results are both driven and limited by the temporal clustering of tweets, i.e., the majority of profits stem from tweets clustered closely together in time, corresponding to high-event situations. Second, the importance of included features follows an economic rationale, e.g., tweets with positive sentiment tend to yield positive returns and vice versa. Third, we find that stocks of medium market capitalization and from the consumer and technology sectors contribute most to our results, which we interpret as a trade-off between tweet coverage and tweet relevance.

*Keywords:* Finance, statistical arbitrage, machine learning, random forests, trading strategy backtesting, social media

[*] Corresponding author

*Email addresses:* `matthias.schnaubelt@fau.de` (Matthias Schnaubelt), `thomas.g.fischer@fau.de` (Thomas Fischer), `christopher.krauss@fau.de` (Christopher Krauss)

[1] The authors have benefited from many helpful discussions with Ingo Klein.

## 1. Introduction

Financial market predictions are a challenging task due to the high level of noise and the widely accepted, semi-strong form of market efficiency (Fama, 1970). Nonsurprisingly, a growing number of investors seeks to exploit non-price-based data sets to predict future returns. With the ever-growing availability of real-time supplementary data and increasingly sophisticated predictive models, such as deep learning, the use of alternative data sources is of growing importance in both research and industry: In a 2016 interview, Osman Ali, portfolio manager at Goldman Sachs Asset Management, mentions the use of internet web traffic in the context of a data-driven investment process (Goldman Sachs, 2016). Other market participants exploit satellite imagery and advanced image processing techniques to predict retail sales in near real-time using the traffic at a store's parking lots as proxy (Turner, 2015), or to predict corn production from the world's largest supplier through large-scale analysis of a plant's color from space (Wilson, 2015). The automated processing of written natural language, gathered for example from news, patent filings and central bank statements, is of growing interest for business applications: For instance, J.P. Morgan recently announced that it is pursuing research in natural language processing "to augment the way portfolio managers and research analysts digest information" (J.P. Morgan, 2018).

Apart from a growing number of applications in the industry, there is surging academic interest in leveraging alternative data in a finance context, and especially in financial market predictions using news, ad-hoc announcements or messages from blogging platforms such as Twitter. Consequently, the surveys of Nassirtoussi et al. (2014) and Oliveira et al. (2017) cover more than 40 works, which can be divided into two groups.

The first group focuses on aggregated measures of sentiment: For example, Antweiler and Frank (2005) use an aggregated bullishness measure for messages from Yahoo! Finance and Raging Bull to find an effect of sentiment on intraday Dow Jones market volatility and a small effect on returns. Tetlock (2007) and Tetlock et al. (2008) analyze aggregated news articles with respect to their effect on a company's earnings and daily stock market reaction, and similarly Bollen et al. (2011) find that daily mood states from Twitter are predictive for the Dow Jones Industrial Average. Sprenger et al. (2013) aggregate the polarity of single tweets to a daily sentiment score and analyze its relation to S&P 100 stock returns, message volume and trading volume. Das and Chen (2007) apply dictionary-based and machine learning methods on posts from stock message boards to assess the correlation between aggregated sentiment and market return as well as volatility.

The second group of works concentrates on the reaction of a single stock's price immediately after the release of a text message. For example, Schumaker and Chen (2009) and Schumaker et al. (2012) classify financial news articles using support-vector regression with regard to their effect on short-term intraday stock returns. Groth and Muntermann (2011) evaluate text mining and machine learning techniques on corporate disclosures for intraday risk management applications. Hagenau et al. (2013) examine the added value of context-capturing features for the prediction of intraday stock returns from ad-hoc announcements. Knoll et al. (2018) apply higher-order factorization machines, an algorithm oftentimes used for recommender systems, to predict future stock returns and trade accordingly.

With our work, we aim at expanding on the current state of research with a large-scale empirical study, building on more than nine million tweets and a survivor-bias free dataset for all S&P 500 constituents on minute-resolution. Specifically, we answer the following open research questions: (1) How can a machine learning pipeline be designed to extract a functional relationship between tweets and a stock's subsequent market reaction? (2) Can the predictions of the machine learning system be translated to statistically and economically significant excess returns when submitted to a rigorous, event-based backtesting engine? (3) Do the findings follow a clear economic rationale, when analyzing sources of profitability? By answering these questions, we make the following contributions to the literature:

1. *Tweet-specific machine learning pipeline:* We develop a tweet-specific machine learning pipeline, aiming to extract actionable signals from a tweet's content in a low signal-to-noise setting. For this purpose, we start by developing handcrafted, domain-specific features along three categories. The first category contains directional indicators, pertaining to potential market reactions. Examples include general words such as "good" or "decrease" and context-specific n-grams such as "beats expectations", and are motivated by the financial literature on sentiment extraction, see for example Henry (2008), Loughran and McDonald (2011) or Hagenau et al. (2013). The second category contains relevance indicators, modeling the likelihood of a tweet containing noise as opposed to market-relevant information. For example, the presence of the terms "report" or "Reuters" may indicate that the tweet covers financial news. The third category contains meta features, pertaining to the twitter network. A prominent example is whether the tweet is a retweet or not, indicating that information is further disseminated. We use random forests to extract a functional relationship between these features and a stock's subsequent market reaction, as they are known to handle noisy data very well and are less affected by overfitting (see, for example, Hastie et al. (2009)). Finally, we develop a filtering logic to derive actionable market signals from more than nine million noisy tweets.

2. *Event-based intraday backtesting with execution constraints:* Next, we subject the identified trading signals to rigorous backtesting, i.e., we simulate the historic execution of a trad-

ing strategy based on single-tweet trading signals and evaluate its financial performance. Hereby, we pay attention to the specifics of event-based trading strategies such as execution constraints, trading costs and capital constraints. We contrast our results to a number of benchmarks, which are a buy-and-hold strategy, a strategy based on k-nearest neighbors, and a naïve sentiment-based prediction model. We find that our tweet-based trading strategy is able to generate statistically significant annualized returns of 6.4 percent with a Sharpe ratio close to 2.2, after accounting for transaction costs and limiting single-stock risk exposure.

3. *Illumination of the black box:* Finally, we illuminate the machine learning black box, uncover sources of profitability, and rigorously ask whether they are in line with economic rationale. First, we show that returns are driven by the temporal distribution of tweets. The majority of profits stems from tweets clustered closely together in calendar time, which corresponds to high-event situations for a stock that can often be profitably exploited. Second, we find our features to correspond well to subsequent market reactions. Tweets with words or n-grams of positive (negative) connotation are generally followed by an increase (decrease) of the stock addressed in the tweet – which is in line with the financial literature on sentiment extraction, see for example Tetlock (2007). Third, we find that our model preferably selects consumer and technology stocks – presumably driven by the business-to-consumer nature of the underlying companies. Also, medium market capitalization stocks rank higher in per-tweet profitability as high or low market capitalization stocks in the S&P 500 universe. Medium size stocks may provide a compromise between tweet coverage (sufficient volume) and tweet relevance (large business-to-consumer stocks tend to show many noisy tweets).

The remainder of this paper is organized as follows (see also figure 1): In section 2, we describe the data sets and software used for our work. Section 3 details all building blocks of our methodology, i.e., the generation of features and targets, the training of predictive models, the generation of trading signals, and finally the backtesting framework. Our results are presented in section 4, and we conclude in section 5.

## 2. Data and software

Our empirical study employs data from two sources concerning S&P 500 stocks: First, a collection of short messages from Twitter, and second, minute-by-minute price time series aggregated from trades. We focus on S&P 500 stocks as they cover about 80 percent of U.S. equity market capitalization (S&P Dow Jones Indices, 2018).

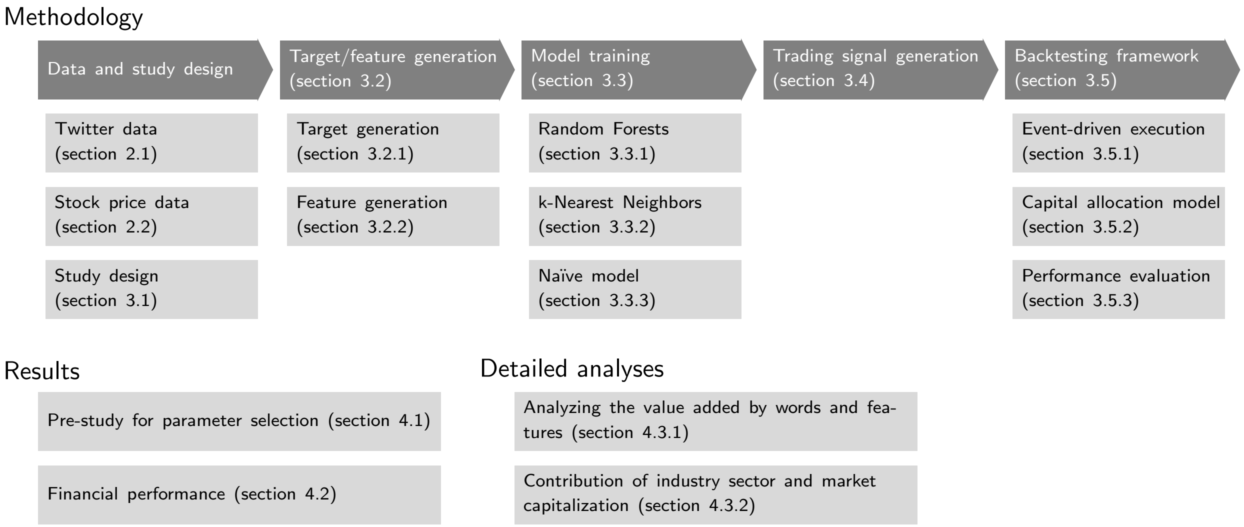

Methodology
Data and study design
Target/feature generation (section 3.2)
Model training (section 3.3)
Trading signal generation (section 3.4)
Backtesting framework (section 3.5)
Twitter data (section 2.1)
Stock price data (section 2.2)
Study design (section 3.1)
Target generation (section 3.2.1)
Feature generation (section 3.2.2)
Random Forests (section 3.3.1)
k-Nearest Neighbors (section 3.3.2)
Naïve model (section 3.3.3)
Event-driven execution (section 3.5.1)
Capital allocation model (section 3.5.2)
Performance evaluation (section 3.5.3)
Results
Pre-study for parameter selection (section 4.1)
Financial performance (section 4.2)
Detailed analyses
Analyzing the value added by words and features (section 4.3.1)
Contribution of industry sector and market capitalization (section 4.3.2)

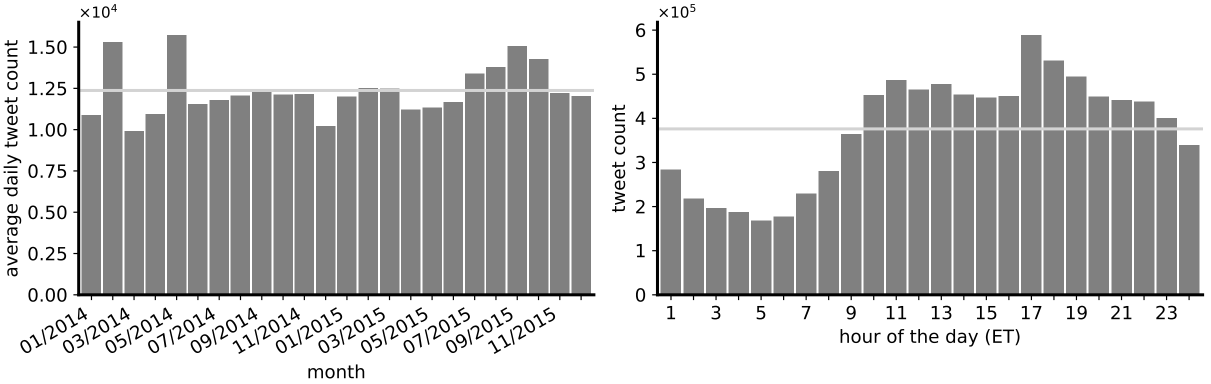
Figure 1: **Overview over this paper's organization.**

### *2.1. Twitter data*

As first data source, our study employs an extensive data set from the microblogging platform Twitter, which has more than 300 million monthly active users world-wide (Statista, 2018b), of which half are based in the United States (Statista, 2018a). Moreover, Twitter is widely accepted in the financial community (Sprenger et al., 2014). On Twitter, information is exchanged between users almost in real-time by sending short text messages. These messages are called *tweets* and consist of 140 characters only – at least in the initial form of the Twitter network.[2] Twitter offers a number of unique features to amplify interactions between users: tweets can *mention* other users via a reference to their user name, as well as forward other user's tweets as a *retweet*. These functions are accessed by using special markup in the text body of the tweet, and are therefore accessible to text processing.

Our data set is retrieved directly from Twitter by utilizing the full company name as search term, which allows us to focus our analysis on tweets with information relevant to specific companies. In line with our high-frequency stock price data, all S&P 500 constituent company names from January $1^{st}$, 2014 to December 31, 2015 are used as search keys. Full company names (i.e., including the form of business entity such as 'Inc.') are used to unambiguously identify companies, since some company names are solely composed of English words. Next to the tweet's text body, the final data set contains a unique identifier assigned by Twitter, a language tag and the time of posting in universal coordinated time. Our sample contains 9.026 million tweets, of which 6.474 million are tagged as containing English language text. The average tweet length is 108 characters. Figure 2(a)

[2] The character limit was increased to 280 characters in November 2017 (Twitter, 2017).



(a) average daily tweet count by month

(b) total tweet count by hour of the day (Eastern Time)

Figure 2: **Distribution of tweets by month and hour of the day.** Light gray lines mark the mean value.

displays the average daily tweet count by month, which is close to the average of $1.2 \times 10^4$ tweets per day. The hourly volume of tweets is lower during night time in Eastern Time, but roughly constant during day-time (compare figure 2(b)).

### *2.2. Stock price data*

As second data source, we obtain minute-by-minute stock prices for S&P 500 constituents from QuantQuote (2018) covering the full period of our Twitter sample. For each minute with traded volume, the data contains the first (open) price, the last (close) price, the maximum (high) price and the minimum (low) price from the set of trades executed during this minute. The data are adjusted for dividend payments, splits and other corporate actions.

### *2.3. Data preparation and consolidation*

To prepare our data sets for further processing, we first remove all tweets that are not in English language. Second, we remove all tweets that have been posted outside the core trading session of the New York Stock Exchange[3] or fall on exchange holidays (New York Stock Exchange, 2018). In a third step, we match tweets with stocks using the following procedure: For every tweet, we check whether it contains the full company name of any S&P 500 constituent. We merge the respectice price time series for all stocks mentioned in the tweet to yield a consolidated data set. If more than one stock is mentioned in the tweet, we include this tweet multiple times in the final data set. We ensure that tweets and price data are aligned to the same time zone by converting all timestamps to US Eastern Time.

[3] On regular (non-holiday) trading days, the core trading session begins at 9:30 and ends at 16:00 (Eastern Time).

### *2.4. Software*

Data is stored in a PostgreSQL database (PostgreSQL Global Development Group, 2018) to facilitate access. Most of the data preprocessing is performed in Python (Python Software Foundation, 2016) with the additional packages numpy (Van Der Walt et al., 2011), pandas (McKinney, 2010) and psycopg2 (Varazzo, 2011). Text processing is done using the Natural Language Toolkit (Bird et al., 2009) and the package sentimentAnalysis (Pröllochs et al., 2015) for the statistical programming language R (R Core Team, 2016). All machine learning models are implemented using scikit-learn (Pedregosa et al., 2011).

## 3. Methodology

Our methodology consists of five steps. First, we split our consolidated data into training and out-of-sample trading sets. Second, we generate the prediction targets and features (inputs) for our prediction models. Third, we train different predictive models. Fourth, we apply these models and select the most promising predictions for trading signal generation. Fifth, we backtest the derived trading signals in a simulated trading environment. In the following, we describe each step in greater detail.

### *3.1. Study design*

Following common practice in the statistical arbitrage literature (see, for example, Gatev et al. (2006)), we use a time-based split of our data into training and out-of-sample trading sets to validate model performance on unseen future observations only. Similar to Fung et al. (2003), and to obtain training samples covering different market states, we use a half-year (26 calendar weeks) training period followed by 4 weeks of out-of-sample trading. Then, we roll forward by 4 weeks to obtain non-overlapping trading periods. We hereby align training and trading periods to start on Mondays to allow for model re-training on weekends. Our first trading period starts on June 30, 2014, which means that we have 20 independent training/trading sets. The first three trading periods are used for parameter calibration, and fully out-of-sample backtesting is performed on the remaining 17 trading periods beginning on September 22, 2014 and lasting until December 31, 2015.

### *3.2. Target and feature generation*

#### *3.2.1. Target generation*

Our aim is to forecast a stock's price development following a tweet, i.e., we want to predict the return from the first price after the tweet to the price $\Delta T$ minutes later. We assign a binary class label to every sample in the training period for use as classification targets. We choose the formulation as a classification problem as previous studies indicate its superior performance over regression problems in the context of financial markets predictions (Enke and Thawornwong, 2005; Leung et al., 2000) and have frequently been used in news-based stock price prediction applications (Nassirtoussi et al., 2014).

We denote the price of stock $s \in \{1, \dots, S\}$ at time $t$ as $P^{c}_{s,\ t}$, where $c \in \{O,\ C\}$ denominates either the opening or closing price of the respective minute. Our target generation proceeds as follows: Given a tweet with timestamp $t_{tw.}$ mentioning some stock $s_{tw.}$, we calculate the multi-period return for a time window $\Delta T$ following the tweet. To this end, we consider the opening price of the minute bar following the tweet's minute of release $P^{O}_{s_{tw.},\ t_{tw.}+1}$ and the price $P^{C}_{s_{tw.},\ t_{tw.}+\Delta T}$ following $\Delta T$ minutes later to align target generation with our tweet-based trading strategy introduced in section 3.5:

$$r_{s_{tw.},\ t_{tw.}} = \frac{P^{C}_{s_{tw.},\ t_{tw.}+\Delta T}}{P^{O}_{s_{tw.},\ t_{tw.}+1}} - 1 \tag{1}$$

We remove tweets from our sample for which $t_{tweet} + \Delta T$ is outside of trading hours. Following Sprenger et al. (2013), we calculate abnormal returns as $ar_{s_{tw.},\ t_{tw.}} = r_{s_{tw.},\ t_{tw.}} - \beta_{s_{tw.},\ d(t_{tw.})} r^{\text{market}}_{t_{tw.}}$. We estimate the stock's beta $\beta_{s,t}$ from the 252 daily returns prior to the tweet's day of posting $d(t_{tw.})$ and use the S&P 500 index as market benchmark. As Dudoit and Fridlyand (2003) suggest that random forests perform poorly on imbalanced targets, we decide to compare the abnormal return $ar_{tw.}$ following a tweet with the median return of all tweets of the same day $\tilde{ar}_{d(t_{tw.})}$ to obtain balanced binary classes relative to the daily cross section of tweets, i.e.,

$$y_{s_{tw.},\ t_{tw.}} = \begin{cases} 0 & \text{if } ar_{s_{tw.},\ t_{tw.}} \leq \tilde{ar}_{d(t_{tw.})} \\ 1 & \text{else.} \end{cases} \tag{2}$$

We choose a forecast horizon of $\Delta T = 120\,\text{min}$ to compute tweet returns, in line with Peramunetilleke and Wong (2002) and Deinert et al. (2018). This value is motivated by the following aspects: First, we aim for minimizing microstructural effects on our intraday application, hence we avoid short-term forecasts. Second, we need sufficient stock volatility to have the potential to cover full-turn transaction costs. Third, we target to capitalize on the data with a high turnover intraday strategy, for which the maximum return time window is limited by the available trading time of 6.5 hours per day. Hence, a return time window of 120 minutes presents a good compromise between these aspects.

### *3.2.2. Feature generation*

In the next step, we construct features for every tweet. We identify three major feature categories building on the existing literature (Henry, 2008; Loughran and McDonald, 2011; Sprenger et al., 2013; Schumaker et al., 2012; Hagenau et al., 2013) to obtain a holistic view on possible feature candidates and to guide further feature engineering:

1. *Directional indicators:* The first category comprises English-language words and n-grams that are assumed to contain predictive information on the future direction of a stock's price. Examples include general words such as "good" or "increase" and context-specific n-grams such as "beats expectations". To reduce the dimensionality of our feature space and to improve generalization, we arrange words or n-grams with similar semantic meaning into distinct groups to find a set of domain-specific dictionaries that capture clearly defined, delimited concepts. Following a number of studies (for example Tetlock et al. (2008); Das and Chen (2007); Henry (2008); Loughran and McDonald (2011); Peramunetilleke and Wong (2002) or Pröllochs et al. (2015)), we count the number of occurences for words from these dictionaries to compute feature values.[4] Oliveira et al. (2016) conjecture that general-purpose dictionaries might not be well-suited for modelling stock market reactions. We therefore include the finance-specific positive and negative dictionaries of Henry (2008) and Loughran and McDonald (2011) for feature construction.
2. *Relevance indicators*: In a second category, we subsume all English-language words that indicate the importance of a given tweet for future stock market returns. These features serve to identify less relevant tweets and thus to suppress noise. Examples are words such as "profit", "forecast" and "sales". As for directional indicators, feature values are calculated as word counts using the respective feature-specific dictionary.
3. *Meta features:* The last feature category contains features going beyond the textual content of the tweet. This allows to account for some of the unique functions of the twitter network, such as retweeting or the presence of a URL. Tweet meta features are computed using regular expressions to match twitter-specific markup or company names.

We list all features that create our feature vector along with relevant references and examples in table 1.

[4] Features are represented in the following way: We first split every lower-case tweet into a list of tokens using the Twitter-aware tokenizer implementation of the Natural Language Toolkit (Bird et al., 2009). We then count the number of occurrences of a tweet's tokens in the feature-specific word list.

| Category | Identifier | Description |
|---|---|---|
| **Directional indicators** | *posWordsHE*<br>*negWordsHE* | We include word counts according to the positivity and negativity word lists from Henry (2008) as directional features. |
| | *posWordsLM*<br>*negWordsLM* | We include words counts according to the positive and negative sentiment word lists from Loughran and McDonald (2011) as directional features. |
| | *upgrade*<br>*downgrade* | We add two binary features based on the presence of the words "upgrade" and "downgrade" (and inflected words). |
| | *posNGrams*<br>*negNGrams* | Building on the research by Hagenau et al. (2013), we include word counts from lists of positive and negative bi- and trigrams which express a relation to some expectation, for example "above expectation", "beats estimates" or "worse than expected". |
| **Relevance indicators** | *forecast* | Based on the observation of Hagenau et al. (2013) that the word "forecast" is of relevance, we include this and synonymous words in a dictionary and add the respective word count as feature. Further examples are the words "predicted" and "prognosis". |
| | *report* | Tweets may contain keywords indicating information related to a company's balance sheet or earnings report. We therefore include a feature indicating the presence of a financial report, based on results by Schumaker et al. (2012), Hagenau et al. (2013) and Sprenger et al. (2013). Examples are the words "profit", "earnings", "revenue", "report", "quarter" and "statement". |
| | *demand* | We add a dictionary-based feature that suggests whether the tweet contains information on the market demand situation a firm is facing, which is based on observations of Sprenger et al. (2013). This dictionary includes the words "demand" and "sales". |
| | *news* | Inspiried by Schumaker et al. (2012), a dictionary-based feature with news agency names (such as "Bloomberg" or "Reuters") is added. |
| | *exchange* | Inspiried by Schumaker et al. (2012), we add a feature that encodes the presence of the name of a stock exchange, for example, "NYSE", in the tweet. |
| **Meta features** | *stockCount* | We add a feature that contains the count of full company names mentioned in the tweet. |
| | *mention*<br>*retweet* | We add binary features indicating whether the tweet contains a mention or is marked as retweet, which both are features appearing as some of the most frequently selected features in Sprenger et al. (2013). |
| | *url* | Similar to Sprenger et al. (2013), we add a feature indicating the presence of a URL in the tweet. |

Table 1: **Model features.** This table lists all elements of the feature vector used for tweet classification. We consider features from three categories, i.e., directional indicators, relevance indicators and meta features. The given feature identifiers are used throughout this work.

### *3.3. Model training*

In the following, we briefly outline the machine learning models used for our empirical application. Throughout this section, we concentrate on classification models and retain the following notation: The $K$-class classifier is trained on $N$ training samples $(\mathbf{x}_i, y_i)$, where $y_i$ represents the target class, i.e., $y_i \in \{0, \ldots, K-1\}$, and $\mathbf{x}_i \in \mathbb{R}^p$ denotes the input feature vector.

#### *3.3.1. Random Forests*

The following description loosely follows Hastie et al. (2009) and is adapted to reflect the implementation of Pedregosa et al. (2011). Random forests were proposed by Breiman (2001) and work by aggregating the predictions of an ensemble of bootstrapped, decorrelated binary decision trees. Due to their predictive performance and straightforward (parallelized) training, random forests have become a popular machine learning method. For our particular application, it also seems favorable that random forests are known to be fairly robust to noise, especially when compared to tree-based boosting techniques (Khoshgoftaar et al., 2011).

The training of an ensemble of $B$ trees introduces randomization from two sources: First, each tree $T_b$ is grown on a bootstrap sample $Z_b$ of the original training data, which is randomly drawn from the training data by selecting $N$ samples with replacement. Second, $m$ features are randomly chosen from the $p$ available features at each newly-grown node of the tree to reduce the correlation between individual trees. Then, the best combination $(j, s)$ out of the $m$ features and their respective split points is chosen by maximizing the information gain:

$$IG = Q(\mathcal{D}_{left} \cup \mathcal{D}_{right}) - \left[ \frac{|\mathcal{D}_{left}|}{|\mathcal{D}_{left} \cup \mathcal{D}_{right}|} Q(\mathcal{D}_{left}) + \frac{|\mathcal{D}_{right}|}{|\mathcal{D}_{left} \cup \mathcal{D}_{right}|} Q(\mathcal{D}_{right}) \right] \tag{3}$$

Here, $\mathcal{D}_{left}$ and $\mathcal{D}_{right}$ denote the data sets of the left and right child node after the split $(j, s)$, $|\cdot|$ denotes the cardinality of a set and the node impurity $Q(\mathcal{D})$ can be given by different measures. The node is then split at the chosen split-point into a left and a right daughter node. This procedure is repeated until the leaf nodes of the tree are pure or some other stopping criterion, e.g. the maximum tree depth $J$, is reached.

The ensemble of trees $\{T_b\}_{b=1}^{B}$ can then be used to make predictions for a new observation $\mathbf{x}$: Every single tree $T_b$ in the ensemble predicts a probability $\hat{p}_b^k(\mathbf{x}) \in [0, 1]$ for $\mathbf{x}$ to be in class $k$, which is given by the fraction of samples of class k in the leaf. It follows that $\sum_k \hat{p}_b^k(\mathbf{x}) = 1$ holds. The predicted class probabilities of the random forest ensemble are then given by the equal-weight mean over all $B$ trees, i.e.,

$$\hat{p}_{RF}^k(\mathbf{x}) = B^{-1} \sum_{b=1}^{B} \hat{p}_b^k(\mathbf{x}). \tag{4}$$

The final predicted class $\hat{k}_{RF}(\mathbf{x})$ of the random forest classifier is the class with highest probability, i.e., $\hat{k}_{RF}(\mathbf{x}) = \text{argmax}_k \, \hat{p}_{RF}^k(\mathbf{x})$.

The main tuning parameters of random forests are the number of trees $B$, the number of randomly selected features $m$, the maximum tree depth $J$ and the selected node impurity measure $Q(\mathcal{D})$. Since choosing a high number of trees does not lead to overfitting and typically improves predictive accuracy (Hastie et al., 2009; Biau and Scornet, 2016), we choose $B = 10000$. We set $m = \lfloor \sqrt{p} \rfloor$, which is the default value for classification (Hastie et al., 2009). It has been found that limiting the maximum tree size, for example, by restricting the maximum tree depth, further reduces the risk of overfitting and improves performance (Segal, 2004). We therefore limit the amount of feature interactions by setting $J = 4$.[5] Regarding the selection of a node impurity measure, we choose the Gini index, which is the default setting of the implementation in scikit-learn (Pedregosa et al., 2011).

### *3.3.2. k-Nearest Neighbors*

We include a k-nearest neighbors classifier as a benchmark model, a simple and model-free classification method, which has been successfully used for text classification (Harish et al., 2010) and in the domain of financial news processing (Groth and Muntermann, 2011). For a given new observation $\mathbf{x}$, the algorithm determines the set $\mathcal{N}_k^d(\mathbf{x})$ of those $k$ samples from the $N$ training observations $(\mathbf{x}_i, y_i)$ that are closest to $\mathbf{x}$ with respect to some metric $d$. The predicted class is then given by the majority class in $\mathcal{N}_k^d(\mathbf{x})$:

$$\hat{k}_{KNN}(\mathbf{x}) = \underset{c}{\operatorname{argmax}} \left|\{\mathbf{x}_i \in \mathcal{N}_k^d(\mathbf{x}) \mid y_i = c\}\right| \tag{5}$$

We use the default metric in scikit-learn (Pedregosa et al., 2011), which is the Euclidean distance. To reduce overfitting, the number of neighbors is set to $k = 10$.[6]

### *3.3.3. Naïve sentiment-based model*

As a second benchmark, we follow Das and Chen (2007) and compute results from a naïve classifier based on sentiment word lists. Specifically, we classify a tweet as long (short) with probability of one if the difference between positive and negative words is equal to or exceeds a certain threshold $T_N$. We use the positivity and negativity word lists of Henry (2008) to determine the positive and negative word counts $W^+$ and $W^-$. We set $T_N = 3$, which is determined from the 12-week calibration period.

## *3.4. Trading signal generation*

Given the high number of tweets in our sample (which is of the order of thousands of tweets per day, compare section 2.1), it seems unlikely that all of them represent a relevant, high-quality trading signal. This necessitates the introduction of a signal filter that reduces the total number of potential trading signals while favouring those with higher expected predictive accuracy. Our model consequently filters tweets along the following two dimensions:

[5] We find that this parameter setting yields good performance on our 12-week calibration period which we use for parameter search prior to out-of-sample backtesting.

[6] This parameter has been tuned to achieve optimal performance on our 12-week calibration period.

- *In-sample training accuracy*: Following common practice in statistical arbitrage trading strategies (see, for example, Gatev et al. (2006)), we identify the $M_s$ most suitable stocks for each study period during model training. To this end, we rank all stocks in our training data by their in-sample predictive accuracy, and select the top $M_s$ stocks for the subsequent trading phase.
- *Out-of-sample prediction probability*: Along with the predicted class $\hat{k}(\mathbf{x})$, our machine learning models estimate a probability $\hat{p}^{\hat{k}}(\mathbf{x})$ for correctly predicting the most likely class $\hat{k}$. As in Fischer and Krauss (2018), considering only those predictions in the extremes of the ranking improves trading signals. Among other models, random forests are known to generally predict uncalibrated probabilities, i.e., the predicted probabilities are not consistently estimated. We circumvent this problem by ranking the estimated probability for tweet $i$ with respect to the $N_r = 5000$ preceding tweets[7]. The rolling window is chosen such that tweet $i$ is evaluated relative to the subset of tweets of the last one to two days. If the resulting percentage rank $r_i$ of tweet $i$ exceeds a cut-off $p_r$, i.e., $r_i \geq 1 - p_r$, the tweet is selected for trading.

The resulting subset of tweets is then transferred to the trading backtest. Tweets with predicted class $\hat{y}_i = 1$ are selected for the long portfolio; tweets with $\hat{y}_i = 0$ are selected for the short portfolio. Note that trading signals soley rely on features from the tweet, and do not incorporate any information from the time series of stock prices. Therefore, the strategy is by design unsusceptible to microstructural effects leading to spurious results. To select suitable parameter settings for our empirical application, we optimize both parameters $M_s$ and $p_r$ on a small hold-out sample[8] of our data, and present results in section 4.1.

### *3.5. Event-based backtesting framework*

Once suitable trading signals have been identified, we run a trading simulation (backtest) to assess the financial performance of the strategy. Hereby, we address a number of challenges arising from our tweet-based intraday trading strategy: First, the backtesting methodology must be able to process event-driven trading signals occurring at arbitrary points in time with varying frequency. Second, it immediately follows that the amount of invested cash varies over time (depending on the number of trading signals), which necessitates stock position tracking and cash management. Third, cash utilization and leverage have to be limited to appropriately reflect real-world constraints. To meet these requirements, we decide to employ an event-driven backtesting framework as described

[7] To initialize the rolling window for the first tweets of a trading period, we hold back the last 5000 tweets from the respective training period and perform out-of-sample predictions.

[8] Hold-out sample in this sense refers to an part of the data saved for calibration purposes and which is not included in the final backtest.

for example in Halls-Moore (2014). The following description of our backtesting framework proceeds along its main components: Once every minute we create an order for each trading signal. A simulated broker tries to fill these orders, thereby respecting execution constraints and transaction costs. A position tracking component continuously monitors current stock positions and available cash.

### *3.5.1. Event-driven order execution model*

The trading simulation keeps track of a (wall-clock) simulation time and processes events in time-ordered fashion. The trading strategy is evaluated every minute $t$ of the exchange opening hours during the backtest period and may generate a set of orders $\{o_i\}_t$ to be subsequently executed by the simulated broker. An order $o_i = (s_o, x_o, t_o)_i$ with submission time $t_o$ constitutes the request to buy (for $x_o > 0$) or sell (for $x_o < 0$) a number of $|x_o|$ units of share $s_o$. Simulated order execution starts in the minute following order submission, i.e., in $t_o + 1$. This corresponds to the one-period-waiting rule employed by Gatev et al. (2006) and ensures that there is sufficient time for the trading signal to be processed.

Once the order can be filled, the transaction is priced at the open price $P^O_{s_o,\ t_o^{ex.}}$ of minute bar $t_o^{ex.}$. We calculate the costs incurred with this order as $P^O_{s_o,\ t_o^{ex.}} \cdot |x_o| \cdot c_s$ from relative transaction costs $c_s$, which may depend on the stock traded. We assume $c_s = 2\,\text{bp}$ for all S&P 500 stocks and for trading years 2014 and 2015 (Jha, 2016; Prager et al., 2012) and $c_s = 0.5\,\text{bp}$ for the highly liquid SPY index ETF[9] which we use as market benchmark.

During the backtest simulation, we keep track of the current stock portfolio $\mathcal{P}^s_t$, i.e., the position invested in a stock $s$ at time $t$, and the currently available cash $\mathcal{C}_t \in \mathbb{R}$, and change these quantities upon execution of an order.[10] Based on the current state of the portfolio, we impose the following constraints for order execution to simulate typical constraints for risk management: First, orders are executed only when execution does not violate a leverage limit $L_{lev.}$. To check the leverage limit, we calculate the cumulated projected absolute position after execution of an order $o = (s_o, x_o, t_o)$ (as well as its market hedge order) and check that it is smaller than the maximum leveraged initial capital $\mathcal{C}_0$:

$$\sum_s |\mathcal{P}^s_t| + P^O_{s_o,\ t_o} \cdot |x_o| \ \leq\ L_{lev.} \cdot \mathcal{C}_0 \tag{6}$$

If the leverage limit is violated, the order is filled up to the volume not violating the condition in

[9] We estimate trading costs for the SPY ETF as follows: The bid-ask spread typically is one price increment, i.e. 0.01 USD, for prices around 200 USD for the examined time period. The half-spread transaction costs would therefore amount to 0.25 bp, which we double to obtain a conservative estimate of total transaction costs.

[10] Negative values of $\mathcal{C}_t$ would indicate the borrowing of cash. The strategy starts with positive cash, i.e., $\mathcal{C}_0 > 0$.

equation 6. The constraint on maximum leverage simulates typical margin requirements imposed by brokers, which is why we set $L_{lev.} = 2.0$ for the remainder of our study. The second execution constraint imposed by our backtesting framework limits the overall position in a single stock $s$ to mitigate short-term intraday default risk and to reduce single-stock risk exposure on portfolio level: The position in a single stock $s$ must not exceed the single-stock position limit $L_{stock}$, which is checked before an order is finally executed:

$$|\mathcal{P}_t^s + P_{s_o,\, t_o}^O \cdot x_o| \;\leq\; L_{stock} \cdot \mathcal{C}_0 \;\; \forall\; s \in \mathcal{S} \setminus \{SPY\} \tag{7}$$

If an order partially violates the single-stock position limit, it is only filled up to the amount not violating the limit. Given that we close positions after a relatively small time horizon of two hours, we diversify our portfolio to a minimum of three stocks at any point in time, and set $L_{stock} = 1/3$.

### *3.5.2. Capital allocation model*

When designing event-based statistical arbitrage strategies, the problem of capital allocation arises: Since events and derived trading signals may occur at arbitrary points in time, the strategy needs to decide which amount to invest following an event without knowledge of possible subsequent events. This situation differs strongly from periodic trading strategies often found in literature, where the portfolio is rebalanced at regular points in time, e.g., on a daily basis. We choose to follow a naïve approach: We invest a fixed amount of 10 percent in every tweet up to the leverage and single-stock position limits. Following common practice for statistical arbitrage strategies in the literature (Avellaneda and Lee, 2010), we hedge any stock order with an opposing order of the S&P 500 index (SPY) to obtain a dollar-neutral portfolio. Once orders have been successfully executed, we schedule orders for position reversal after a holding time of $\Delta T = 120\,\text{min}$.

### *3.5.3. Performance evaluation*

We perform a comprehensive analysis of our backtest results from two different perspectives: First, we evaluate trading results on a per-tweet basis, i.e., the series of returns following every tweet selected by trading signal filtering. Second, we calculate a daily return series from the daily ending cash $\mathcal{C}_t$ after market close. Following Fischer and Krauss (2018), we compute the mean return per day, standard deviation and distributional properties, daily risk metrics, such as the value at risk, as well as annualized return and risk metrics.

## 4. Results

The presentation of our results proceeds in three parts: First, we present the results of the pre-study for parameter selection for the signal filter on the three month hold-out set (section 4.1). Second, using the obtained parameters, we discuss the trading results of our strategy in section 4.2 and compare them to results of the k-nearest neighbors model, the naïve model, and a buy-and-hold benchmark. Third, we analyze our results in-depth in section 4.3 and examine the value added by the different components of our approach.

### *4.1. Pre-study for parameter selection on a three month hold-out set*

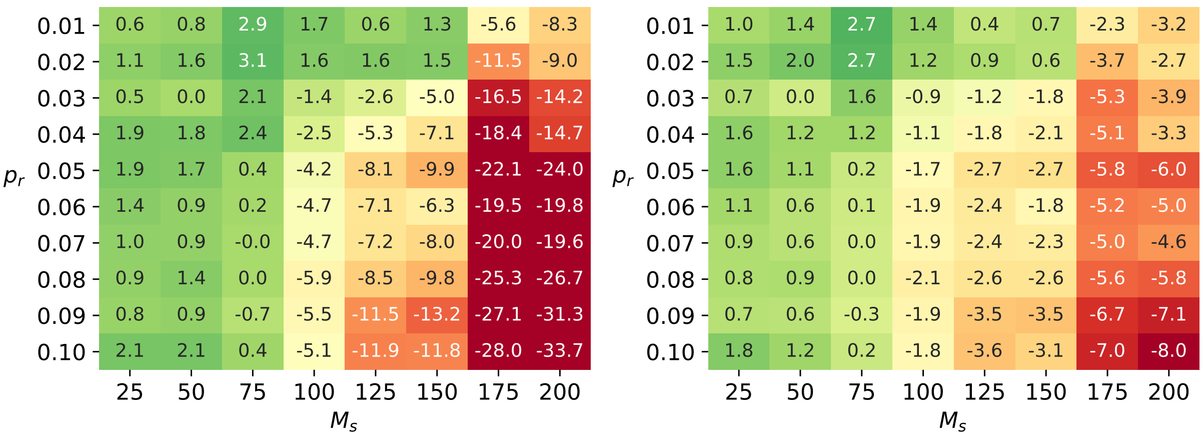
As discussed in section 3.4, the trading strategy is configured using two tweet filtering parameters, i.e., the top-$M_s$ stocks according to in-sample accuracy, and the minimum predicted tweet probability $p_r$. To find adequate values for these parameters, we perform a grid search over a large range of different parameter combinations using the three months following the first training data set (precisely, 12 trading weeks from June 30, 2014 to September 19, 2014). We evaluate the annualized return and Sharpe ratio after transaction costs to find a suitable parameter configuration.

Results are shown in figure 3. We intepret the observed behavior in the following way: Increasing the selective strictness of our signal filter by lowering $M_s$ or $p_r$ generally improves the quality of the identified trading signals. However, this also decreases the number of tweets selected for trading and consequently the number of times a positive return after transaction costs can be achieved. Consequently, optimal performance should be achieved at intermediate values of $M_s$ and $p_r$.

We observe that good performance is achieved for less than roughly one fifth of the available stocks. This is in line with a plethora of statistical arbitrage strategies, which typically restrict the number of traded stocks to only a fraction of the stock universe accessible to the strategy (see, for example, Fischer and Krauss (2018) or Asness (1997)). In the light of the high number of tweets posted every day, it is not surprising that better performance is achieved only when heavily reducing the number of traded tweets, i.e., by selecting small values of $p_r$. We finally select $M_s = 75$ and $p_r = 2\%$ for the model based on a random forest and calibrate our benchmark models in a similar fashion.

### *4.2. Financial performance*

After selecting suitable parameters for the signal filter, we perform a detailed analysis of the financial performance of our tweet-based trading strategy on the remaining data, thereby leaving out the calibration period previously used for parameter selection. Also, we present the results of the performed robustness checks.



(a) Annualized return

(b) Annualized Sharpe ratio

Figure 3: **Backtest results for parameter selection.** The heat map shows backtest results after transaction costs for the 12-week hold-out calibration period from June 30, 2014 to September 19, 2014.

### *4.2.1. Per-tweet return characteristics*

First, we evaluate the performance of our predictive models on a per-tweet basis. Table 2 depicts return statistics for the tweets selected by the trading signal filter in the 15 months out-of-sample backtest period from September 22, 2014 to December 31, 2015. We report simple and excess returns after transaction costs for the random forest (RF), the k-nearest neighbors (KNN) and the naïve sentiment-based model (N), thereby taking into account the predicted direction of return. Excess returns are calculated as the difference between simple return and corresponding market return to align with our market-hedged dollar-neutral backtest portfolios.

In terms of mean return, we find that the random forest outperforms our benchmark models by far: With average excess returns of 48 bp after transaction costs, the RF-based strategy achieves excess returns that are a factor of ten higher than excess returns from our benchmark models. With the exception of the k-nearest neighbors model, the models achieve higher mean returns from tweets with a sell signal. This is in line with the findings of Tetlock (2007) that negative words have a larger effect on stock returns. All mean returns are found to be significantly different from zero at the 5 percent level, and the machine learning models RF and KNN yield mean returns significant at the 1 percent level. Compared to other statistical arbitrage strategies, the tweet-based random forest strategy achieves a rather high share of positive returns. Overall, the random forest seems to be able to capture interactions of directional indicators with relevance and meta features during training, which improves predictions when compared to the naïve model based on sentiment features alone. Compared to the model-free KNN, the RF model seems to generalize better.

We find that all models select a similar number of tweets from the entirety of tweets in the back-

| | simple return | | | excess return | | |
|---|---|---|---|---|---|---|
| | RF | KNN | N | RF | KNN | N |
| Count | 1245 | 1409 | 1013 | 1245 | 1409 | 1013 |
| Mean return (short) | 0.0055 | -0.0002 | 0.0024 | 0.0053 | 0.0001 | 0.0020 |
| Mean return (long) | 0.0010 | 0.0010 | 0.0004 | 0.0026 | 0.0006 | 0.0001 |
| Mean return | 0.0048 | 0.0005 | 0.0009 | 0.0048 | 0.0004 | 0.0006 |
| Standard error | 0.0003 | 0.0002 | 0.0003 | 0.0003 | 0.0001 | 0.0003 |
| t-statistic | 17.3115 | 3.1035 | 2.8277 | 18.1364 | 2.6816 | 2.1500 |
| Minimum | -0.0624 | -0.0236 | -0.0330 | -0.0608 | -0.0250 | -0.0235 |
| First quartile | -0.0007 | -0.0019 | -0.0054 | -0.0007 | -0.0018 | -0.0044 |
| Median | 0.0033 | -0.0002 | -0.0004 | 0.0030 | 0.0001 | -0.0004 |
| Third quartile | 0.0101 | 0.0035 | 0.0038 | 0.0094 | 0.0019 | 0.0024 |
| Maximum | 0.0522 | 0.0374 | 0.0695 | 0.0532 | 0.0350 | 0.0410 |
| Standard deviation | 0.0097 | 0.0057 | 0.0100 | 0.0094 | 0.0052 | 0.0086 |
| Share ≥0 | 0.7157 | 0.4918 | 0.4699 | 0.7012 | 0.5174 | 0.4492 |

Table 2: **Per-tweet return statistics.** The table depicts per-tweet returns for the random forest (RF) and contrasts them to the performance of the k-nearest neighbors (KNN) and naïve sentiment-based model (N). We evaluate simple returns as well as excess returns relative to the S&P 500 performance, both after transaction costs.

test period. The naïve model selects the fewest tweets (1013), followed by the random forest with 1245 selected tweets and the k-nearest neighbors with 1409 tweets. We might carefully hypothesize that all models are similarly affected by signal degeneration.

### *4.2.2. Daily and annualized risk-return metrics*

We proceed with the evaluation of results from our backtest trading simulation. Table 3 summarizes daily and annualized risk-return metrics for the random forest and compares them to the performance of the k-nearest neighbors and naïve models as well as a simple buy-and-hold benchmark of the overall market.

*Return characteristics*: Panel A of table 3 shows that the random forest achieves the highest average daily return, with 3.6 bp before and 2.5 bp after transaction costs. We find that both daily mean return figures are statistically significant, with Newey-West t-statistics of 3.12 and 2.47, respectively, compared to a critical value of 1.96 at a 5 percent significance level. The distributional properties of the series of daily returns can be explained by the fact that the strategy does not trade every single trading day, which we would expect given the non-uniform timing of our tweet-based trading signals. In consequence, the median daily return is zero and returns exhibit a high share of zero or positive returns. When compared to our benchmark models, we observe that both the KNN model and the naïve model exhibit weakly positive daily mean returns. However, the t-statistic reveals that these returns are statistically not significantly different from zero after transaction costs. Despite its weaker per-tweet mean excess return, the KNN model outperforms the naïve benchmark in terms of daily return characteristics both before (1.0 bp) and after transaction costs (0.5 bp), which might be due to the fact that the KNN model selects more tweets for trading.

| | | before transaction costs | | | after transaction costs | | | |
|---|---|---|---|---|---|---|---|---|
| | | RF | KNN | N | RF | KNN | N | MKT |
| A | Mean return | 0.00036 | 0.00010 | 0.00007 | 0.00025 | 0.00005 | 0.00001 | 0.00017 |
| | Standard error | 0.00012 | 0.00005 | 0.00008 | 0.00010 | 0.00005 | 0.00008 | 0.00053 |
| | t-statistic | 3.12274 | 2.06040 | 0.88199 | 2.47098 | 0.99520 | 0.07080 | 0.32902 |
| | Minimum | -0.00634 | -0.00281 | -0.00580 | -0.00649 | -0.00297 | -0.00801 | -0.04088 |
| | First quartile | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | -0.00450 |
| | Median | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | -0.00014 |
| | Third quartile | 0.00015 | 0.00000 | 0.00000 | 0.00008 | 0.00000 | 0.00000 | 0.00526 |
| | Maximum | 0.01952 | 0.00942 | 0.01049 | 0.01620 | 0.00886 | 0.01028 | 0.03839 |
| | Share ≥0 | 0.78638 | 0.85759 | 0.79567 | 0.75542 | 0.83591 | 0.78019 | 0.49845 |
| | Standard deviation | 0.00210 | 0.00088 | 0.00135 | 0.00181 | 0.00086 | 0.00138 | 0.00952 |
| | Skewness | 5.96995 | 5.12692 | 2.80364 | 5.57576 | 4.48488 | 1.88569 | -0.16933 |
| | Kurtosis | 48.54164 | 49.33279 | 26.57825 | 46.77958 | 45.15215 | 25.37833 | 4.81163 |
| B | 1 percent VaR | -0.00212 | -0.00154 | -0.00426 | -0.00231 | -0.00195 | -0.00463 | -0.02419 |
| | 1 percent CVaR | -0.00374 | -0.00216 | -0.00478 | -0.00389 | -0.00255 | -0.00557 | -0.03173 |
| | 5 percent VaR | -0.00094 | -0.00079 | -0.00135 | -0.00107 | -0.00102 | -0.00150 | -0.01597 |
| | 5 percent CVaR | -0.00193 | -0.00137 | -0.00283 | -0.00210 | -0.00162 | -0.00319 | -0.02106 |
| | Max. drawdown | 0.00804 | 0.00555 | 0.01412 | 0.01080 | 0.00608 | 0.02542 | 0.11910 |
| C | Return p.a. | 0.09561 | 0.02562 | 0.01662 | 0.06408 | 0.01192 | 0.00113 | 0.03299 |
| | Volatility p.a. | 0.03336 | 0.01397 | 0.02148 | 0.02869 | 0.01361 | 0.02196 | 0.15115 |
| | Sharpe ratio p.a. | 2.75399 | 1.81709 | 0.77784 | 2.17919 | 0.87768 | 0.06244 | 0.29017 |
| | Sortino ratio p.a. | 10.38778 | 4.48616 | 1.40040 | 6.59148 | 1.78890 | 0.09980 | 0.41098 |

Table 3: **Daily and annualized risk-return metrics.** We depict result for the models random forest (RF), k-nearest neighbors (KNN) and a naïve sentiment-based approach (N), both before and after transaction costs. Results are compared to the general market (MKT) as given by the S&P 500 index.

The general market achieves roughly half the mean daily returns of the random forest strategy after transaction costs. These returns are however not significantly different from zero and have a five times larger standard deviation. Return from the RF model are skewed to the right, whereas returns from the general market are skewed to the left.

*Risk characteristics*: Panel B of table 3 reveals that the random forest has favourable risk characteristics: It has a 1 percent daily value at risk (VaR) of -0.23 percent, which is close to the KNN model with a 1 percent VaR of -0.20 percent and about half the figure of the naïve model (-0.46 percent). The risk characteristics of the general market show than an investment in the general market is about one order of magnitude riskier than an investment using the tweet-based RF strategy. Comparing our results to those discussed by Gatev et al. (2006) as an example of a quantitative statistical arbitrage strategy, we find that our 1 percent daily VaR is relatively low.

*Annualized risk-return characteristics*: In panel C of table 3, we present the annualized risk-return metrics of our tweet-based trading strategies and the buy-and-hold market benchmark. With an annualized return of 9.56 percent before and 6.41 percent after transaction costs, the random forest outperforms all benchmark models and the market benchmark.[11] The buy-and-hold strategy achieves annualized returns of 3.30 percent, and the k-nearest neighbors model yields 1.19 percent per annum. The random forest model has critical full-turn transaction costs of 23 bp, which leaves a clear margin to the transaction costs utilized by our study (5 bp). The panel also reports the Sharpe ratio, which relates annualized returns to the risk taken, as expressed by the annualized volatility. The random forest achieves a Sharpe ratio of 2.18 after transaction costs, which is considerably higher than the Sharpe ratio of the market benchmark (0.29).

Figure 4 depicts the development of cumulative profits over the out-of-sample trading period before and after transaction costs. We observe that the positive returns of the tweet-based strategy are accumulated quite uniformly over the whole period, with a number of outstandingly positive trading days. Compared to the buy-and-hold benchmark, the volatility of the tweet-based strategies is much smaller, as expected from the risk metrics.

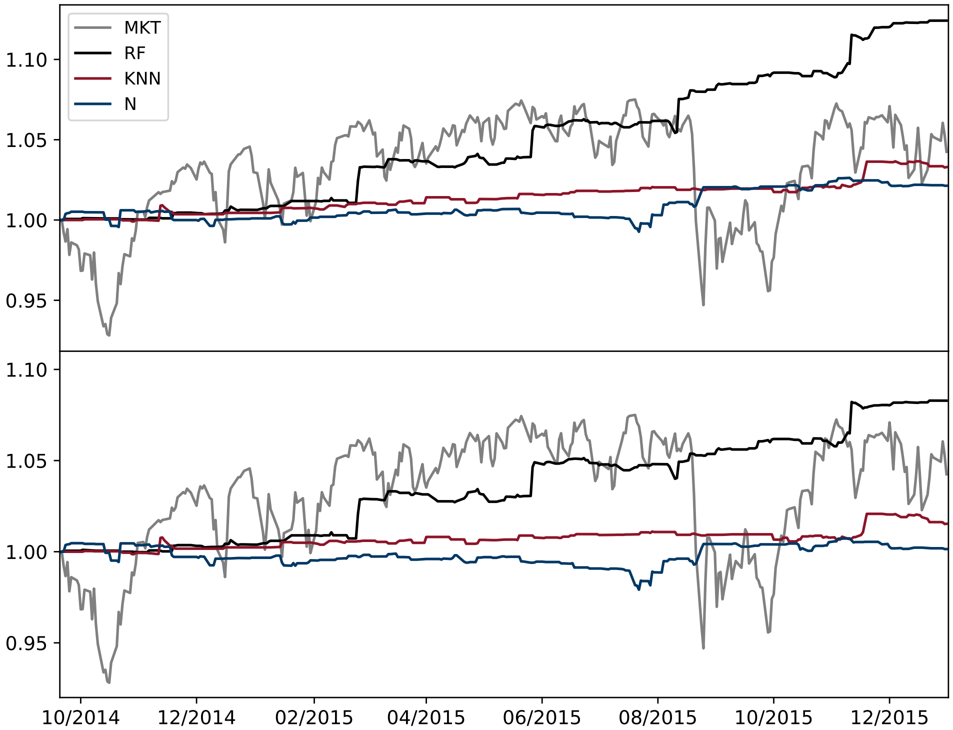


Figure 4: **Development of cumulative profits.** The figure plots cumulative profits for the trading strategies based on random forests (RF), k-nearest neighbors (KNN) and the naïve sentiment-based model (N) as well as for a simple buy-and-hold strategy of the overall market (MKT), after transaction costs.

[11] Compared to per-tweet returns, annualized returns of the trading strategy seem small. We investigate this seemingly severe discrepancy in section 4.2.3.

|  | fraction of clustered tweets | mean excess return |  | t-statistic |  |
|---|---|---|---|---|---|
|  |  | clustered | unclustered | clustered | unclustered |
| no clustering | 0% | – | 0.0048 | – | 18.1364 |
| same second ($n = 1\,\text{sec}$) | 22.2% | 0.0086 | 0.0037 | 19.8118 | 12.0438 |
| within $n = 10\,\text{sec}$ | 54.1% | 0.0079 | 0.0012 | 22.0089 | 3.6092 |
| within $n = 60\,\text{sec}$ | 60.6% | 0.0074 | 0.0009 | 21.9409 | 2.4045 |
| within $n = 600\,\text{sec}$ | 71.4% | 0.0067 | 0.0002 | 21.3598 | 0.5234 |

Table 4: **Analysis of temporal tweet clustering**. Tweets are divided into clustered and unclustered tweets, whereby a tweet is said to be clustered if another tweet mentioning the same stock occurs withing the given clustering window $n$. We calculate the fraction of clustered tweets as well as the mean excess per-tweet return after transaction costs and respective t-statistics for clustered and unclustered tweets seperately.

### *4.2.3. Exploring the difference between per-tweet and per-day returns*

When comparing mean per-tweet excess returns and daily return figures, we observe a stark contrast: While the average selected tweet yields an excess return of 48 bp after transaction costs, we find much lower daily returns of 2.5 bp. To further explore this difference, we perform two analyses: First, we examine the temporal distribution of tweets and reveal the occurrence of tweet clusters. Second, we investigate how the single-stock risk constraint subsequently limits annualized performance.

*Temporal tweet clustering:* In the first analysis, we investigate whether related tweets occur in clusters and whether these clusters are followed by higher trading returns. We therefore split all tweets selected for trading into two non-overlapping groups: The first group contains only tweets whose stock has been mentioned by at least one other tweet in the past $n$ seconds (clustered group). The second group contains all other tweets (unclustered group). In the next step, we compare mean per-tweet returns for the two groups, using different values for $n$. The results are shown in table 4. We make the following observations: First, tweets selected for trading oftentimes occur in clusters, i.e., the respective stock is mentioned by several tweets in a short amount of time. Second, the mean return of the clustered group significantly exceeds the mean return of the unclustered group. Both are an indication that news supported by several tweets carry more weight and result in higher returns. Third, this effect diminishes with increasing window size $n$.

*Single-stock position limit:* Our second analysis shows how the single-stock risk constraint affects the trading performance on portfolio level. We proceed as follows: Keeping all other parameters unchanged, we run our strategy backtest for different values of the single-stock position limit $L_{stock}$. Figure 5 plots annualized returns and volatilities as well as Sharpe ratios for different values of $L_{stock}$. We make the following observations: First, annualized returns scale approximately linearly with single-stock risk. We find that an increase of single-stock exposure by 10 percentage points yields additional annualized returns of 1.2 percentage points. Allowing arbitrarily large positions for a

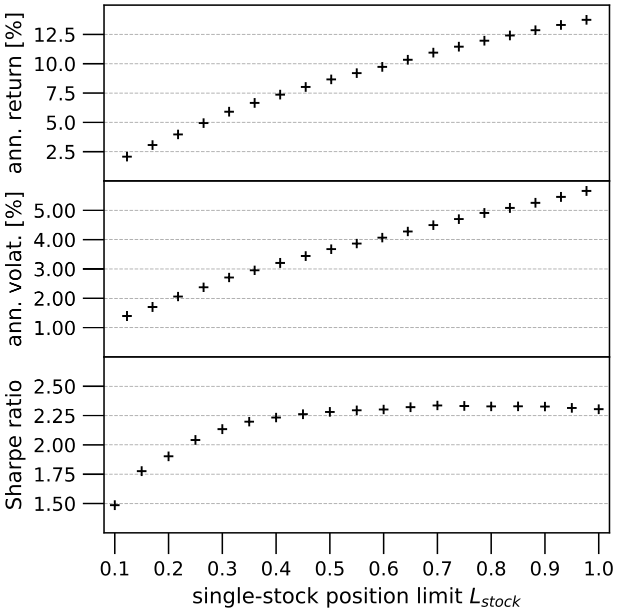


Figure 5: **Influence of single-stock risk on strategy performance.** The plots display annualized returns and volatilities as well as Sharpe ratios for different values of the single-stock position limit $L_{stock}$.

single stock (up to a maximum leverage of $L_{lev.} = 2.0$) and hence higher cash utilization, yields annualized returns of 13.7 percent, which is considerably higher than the value of 6.4 percent under a reasonable limit on single-stock risk constraint. This result is well in line with results from related studies in the literature (Hagenau et al., 2013; Knoll et al., 2018). Second, volatility in portfolio value is lower for low values of $L_{stock}$. Permitting larger single-stock risk increases volatility of the portfolio value with roughly linear scaling. Third, the Sharpe ratio of the tweet-based strategy, i.e., the ratio of returns to volatility, remains constant at values just above 2 for $L_{stock} \geq 0.4$, and declines thereafter down to a minimum of 1.5 for $L_{stock} = 0.1$.

From the above analyses, we can draw the following conclusions: First, a large share of tweets occurs in temporal proximity. Second, these tweet clusters exhibit higher mean returns. Third, we capitalize on this effect by increasing the portfolio weight of positions that are suggested by several tweets by opening larger positions in the respective stock (see section 3.5.2). Finally, we cap this effect to limit the single-stock exposure on portfolio level with a value of $L_{stock} = 1/3$, hereby finding a good balance between risk and return.

### *4.2.4. Robustness checks*

To ensure the robustness of our results, we perform a series of checks.

*Monkey trading:* First, we perform a simulation study to check whether we could have found the tweet-based trading signals by chance alone. With our RF based trading strategy, we select a total of $N = 1245$ tweets with a mean excess return of 48 bp after transaction costs. To estimate the probability of having found these tweets and the corresponding predicted trading signals by

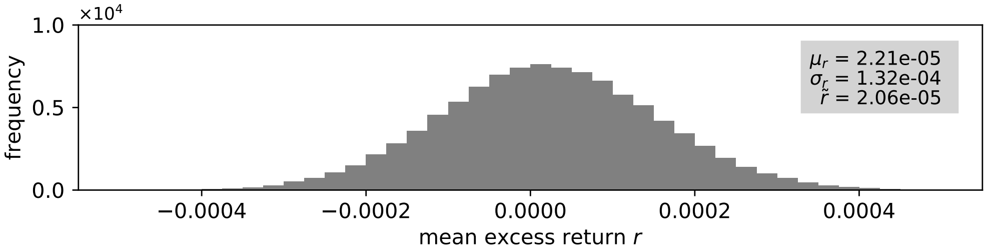


Figure 6: **Distribution of mean returns from randomly sampled trading signals.** We display the empirical distribution of mean excess returns of 100,000 replications of random samples of $N = 1245$ tweet-based trading signals, excluding transaction costs.

| | selected tweets | mean per-tweet excess return | annualized return | Sharpe ratio |
|---|---|---|---|---|
| Minimum | 1202 | 0.0048 | 0.0609 | 2.07 |
| Maximum | 1245 | 0.0049 | 0.0683 | 2.32 |
| Mean | 1224 | 0.0048 | 0.0644 | 2.20 |
| Standard deviation | 11 | 0.0000 | 0.0023 | 0.07 |

Table 5: **Random forest seed robustness.** The table presents distributional characteristics of main performance metrics from the application of the random forest model with 10 different seed values. All return metrics include transaction costs.

chance alone, we randomly sample $N = 1245$ tweets with randomly generated trading signals and calculate their mean excess return. We perform this sampling for 100,000 replications. The resulting empirical distribution is shown in figure 6. We find that the mean excess return of 0.2 bp is very close to zero and the best random sample with a mean excess return of 5 bp per tweet is still one order of magnitude smaller than the mean excess return of the tweets selected by the random forest model.

*Variation of seed:* Second, we investigate whether our results are driven by selecting one outstanding result from a number of runs of the random forest model with different seed values. To this end, we apply the random forest model ten times over the full trading period with different consecutive seed values and calculate main performance metrics. Table 5 depicts the distributional characteristics of these seed runs. There are slight variations in the results in the order of one-tenth standard deviation, but the findings remain robust. The results presented in other parts of this paper are median results with respect to the random forest seed to facilitate reproducibility.

*Different model configurations:* Third, we explore whether results are robust with respect to variations of model configuration. Table 6 presents results from running the model with a different number of trees $B$ in the random forest (compare section 3.3.1) and different values for the size of the rolling window of tweets $N_r$ used for ranking (compare section 3.4). We observe only small changes in results in the order of random fluctuations induced by different seed values.

| Alternative configuration | selected tweets | mean per-tweet excess return | annualized return | Sharpe ratio |
|---|---|---|---|---|
| Number of trees $B = 5000$ | 1326 | 0.0043 | 0.0593 | 1.94 |
| Number of trees $B = 20000$ | 1243 | 0.0048 | 0.0652 | 2.22 |
| Ranking set $N_r = 2500$ | 1295 | 0.0039 | 0.0612 | 1.90 |
| Ranking set $N_r = 10000$ | 1233 | 0.0050 | 0.0659 | 2.18 |

Table 6: **Results from alternative random forest configurations.** The table presents backtest results from alternative configurations of the random forest model. Each alternative configuration changes the stated parameter and keeps all remaining parameters unchanged. All return metrics include transaction costs.

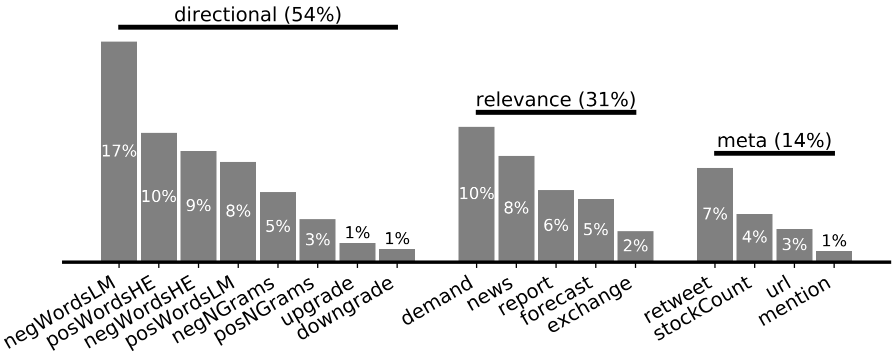


Figure 7: **In-sample feature importance.** Average feature importances are calculated using the mean decrease in the impurity function while growing the random forest.

*4.3. Detailed analyses – illuminating the black box*

To better understand the results of our tweet-based trading strategy, we perform a number of detailed analyses in the following. We first analyze the textual content of selected tweets (section 4.3.1). Then, we analyze whether results are driven by tweets from a specific industry or stocks of specific size (section 4.3.2).

*4.3.1. Analyzing the value added by words and features*

Our trading strategy is solely based on Twitter data for signal generation. In the following three analyses, we shed light onto the decision-making process of our machine learning model. First, we examine in-sample feature importance. Second, we analyze whether tweets with certain feature values are preferably selected. Third, we investigate if there are single words driving our results.

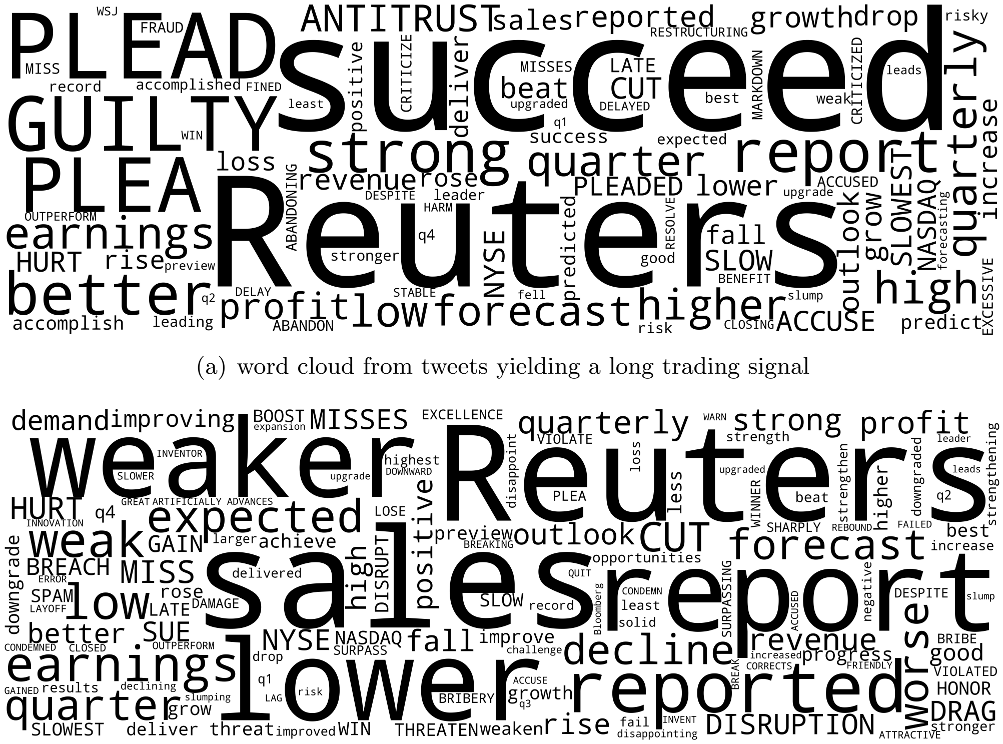
*In-sample feature importance:* In the first analysis, we follow Krauss et al. (2017) and calculate in-sample feature importances, i.e., we consider the mean decrease in the impurity (MDI) function for each feature during training. Figure 7 depicts feature importances averaged over all of our study periods and arranges them into the feature groups introduced in section 3.2.2. We find that the directional feature group is most important in terms of feature importance, followed by the relevance indicators and the meta features. Within the group of directional features, those based on financial dictionaries have a higher importance score, which we attribute to their larger corpus of words. With the exception of the features *upgrade* and *downgrade*, we find that features

|  | mean feature value |  |  |  | mean relative feature value |  |  |
|---|---|---|---|---|---|---|---|
|  | all tweets | selected tweets | long only | short only | selected tweets | long only | short only |
| *posWordsHE* | 0.1358 | 0.2635 | 0.7951 | 0.1587 | 1.9402 | 5.8556 | 1.1684 |
| *negWordsHE* | 0.0518 | 0.1847 | 0.1707 | 0.1875 | 3.5681 | 3.2976 | 3.6214 |
| *posWordsLM* | 0.1435 | 0.2779 | 0.6195 | 0.2106 | 1.9362 | 4.3162 | 1.4671 |
| *negWordsLM* | 0.2723 | 0.6305 | 0.8195 | 0.5933 | 2.3160 | 3.0101 | 2.1791 |
| *upgrade* | 0.0107 | 0.0016 | 0.0049 | 0.0010 | 0.1503 | 0.4565 | 0.0900 |
| *downgrade* | 0.0077 | 0.0032 | 0.0000 | 0.0038 | 0.4178 | 0.0000 | 0.5002 |
| *posNGrams* | 0.0045 | 0.0249 | 0.0683 | 0.0163 | 5.5562 | 15.2392 | 3.6476 |
| *negNGrams* | 0.0012 | 0.0980 | 0.0000 | 0.1173 | 84.5196 | 0.0000 | 101.1797 |
| *forecast* | 0.0119 | 0.1205 | 0.1073 | 0.1231 | 10.1087 | 9.0041 | 10.3264 |
| *report* | 0.0884 | 0.4305 | 0.2829 | 0.4596 | 4.8712 | 3.2012 | 5.2004 |
| *demand* | 0.0197 | 0.5149 | 0.0244 | 0.6115 | 26.1920 | 1.2408 | 31.1103 |
| *news* | 0.0831 | 0.4819 | 0.5073 | 0.4769 | 5.8006 | 6.1062 | 5.7404 |
| *exchange* | 0.0778 | 0.0402 | 0.0488 | 0.0385 | 0.5163 | 0.6271 | 0.4944 |
| *stockCount* | 1.0671 | 1.1936 | 1.3561 | 1.1615 | 1.1186 | 1.2709 | 1.0885 |
| *mention* | 0.0114 | 0.0193 | 0.0000 | 0.0231 | 1.6898 | 0.0000 | 2.0229 |
| *retweet* | 0.1357 | 0.1301 | 0.1463 | 0.1269 | 0.9590 | 1.0785 | 0.9354 |
| *url* | 0.7334 | 0.5759 | 0.7805 | 0.5356 | 0.7853 | 1.0642 | 0.7303 |

Table 7: **Mean feature values.** The four left columns display averaged feature values for all tweets in the backtest period and for those tweets selected for (long/short) trading. The last three columns relate mean feature values for selected (long/short) tweets to the entirety of all tweets.

with negative sentiment yield a higher decrease in impurity than positive ones. In the group of relevance features, indications that the tweet contains information on the sales of the company (feature *demand*) are most important, whereas the presence of an exchange name (for example, “NYSE” or “Nasdaq”) is least important. The tweet meta features *url* and *mention* have the lowest importance for node splitting during training.

*Distribution of feature values:* Second, we analyze how feature values are distributed in the tweets selected for trading and relate these to the entirety of tweets. To this end, we calculate mean feature values for all tweets and for those selected for trading, and repeat the calculation separately for tweets resulting in a long and a short trading signal. Note that mean feature values correspond to the frequency of the flag value ’1’ for binary features. We then divide mean feature values of selected tweets by mean feature values of all tweets to identify more frequent features in selected tweets. Results are shown in table 7. If we compare relative mean feature values with MDI feature importances, we find that results are largely consistent: First, we observe that directional features are more frequent in the tweets selected for trading, with the exception of the features *downgrade* and *upgrade*. Second, the analysis once more confirms that negative directional features play a more prominent role than positive ones. Third, we find that relative mean values for the group of directional features exhibit a weak diagonal form, i.e., features with positive sentiment are



(a) word cloud from tweets yielding a long trading signal

(b) word cloud from tweets yielding a short trading signal

Figure 8: **Visualization of word frequencies.** For the visualization, we calculate word frequencies for words found in feature-specific dictionaries and yielding long or short trading signal, and scale words according to their frequency.

more frequent in tweets yielding a long trading signal and vice versa. This is to be expected if we assume that our strategy is trained on a sustained market reaction following a new message. One notable difference to previous results concerns the influcence of retweeting: While retweeting is the most important meta feature in the MDI in-sample analysis, it is slightly underrepresented in the quorum of tweets selected for trading.

*Analysis of word frequency:* In the third analysis, we investigate which single words contribute most to our results. We calculate the frequencies of words that are both in tweets yielding long or short trading signals and are additionally part of our feature-specific dictionaries. We illustrate results with two word clouds, which are depicted in figure 8, and make the following observations: First, words from directional and relevance features contribute jointly to the results. Second, for both long and short tweets, similar relevance words are found, with the name of a news agency ("Reuters") being most frequent. Third, and as established before, short tweets exhibit a high number of words with negative polarity, while long tweets feature words with positive polarity.

We summarize the findings from the above analyses as follows: All three analyses yield consistent results in line with expectation, i.e., tweets with positive (negative) connotation are more likely to produce long (short) trading signals. This is in line with the financial literature on sen-

| | traded stocks | | | traded vs. full | | full sample | | full vs. S&P 500 | S&P 500 |
|---|---|---|---|---|---|---|---|---|---|
| | mean return | tweet share | stock share | tweet share | stock share | tweet share | stock share | stock share | stock share |
| Technology | 0.0063 | 10.4% | 16.0% | 0.52 | 1.52 | 20.1% | 10.6% | 1.06 | 10.0% |
| Consumer Services | 0.0062 | 51.1% | 19.8% | 3.14 | 1.30 | 16.3% | 15.2% | 1.00 | 15.2% |
| Financials | -0.0010 | 9.0% | 18.3% | 0.39 | 1.06 | 23.2% | 17.3% | 1.00 | 17.3% |
| Basic Materials | 0.0053 | 1.3% | 4.6% | 0.63 | 1.02 | 2.1% | 4.5% | 0.91 | 4.9% |
| Consumer Goods | 0.0057 | 10.4% | 12.2% | 1.20 | 1.00 | 8.7% | 12.2% | 1.00 | 12.2% |
| Health Care | 0.0070 | 7.5% | 9.2% | 0.71 | 1.00 | 10.6% | 9.1% | 0.92 | 10.0% |
| Oil & Gas | -0.0037 | 2.6% | 8.4% | 0.25 | 0.96 | 10.2% | 8.7% | 1.03 | 8.5% |
| Industrials | -0.0000 | 7.4% | 8.4% | 1.16 | 0.56 | 6.4% | 15.0% | 1.01 | 14.8% |
| Utilities | 0.0008 | 0.3% | 3.1% | 0.19 | 0.56 | 1.7% | 5.5% | 0.94 | 5.8% |
| Telecommunications | – | 0.0% | 0.0% | 0.00 | 0.00 | 0.5% | 1.0% | 0.90 | 1.1% |

Table 8: **Contribution analysis by industry sector.** We compare the share of the given industries among the tweets selected for trading and the stocks selected for trading. We also include the corresponding shares for the full sample of tweets as well as the share of stocks by sector in the S&P 500 as reference. Mean per-tweet excess returns are shown after transaction costs. Gray columns list the quotient of respective industry shares.

timent extraction, see for example Tetlock (2007). Furthermore, relevance indicators contribute information alongside directional and meta features.

### *4.3.2. Contribution of industry sector and market capitalization*

Next, we analyze whether stocks from certain industries or of certain sizes are preferably selected for trading. First, we focus on the contribution of specific industry sectors to average returns and selection frequency. Second, we investigate whether tweets with specific market capitalization are selected more frequently.

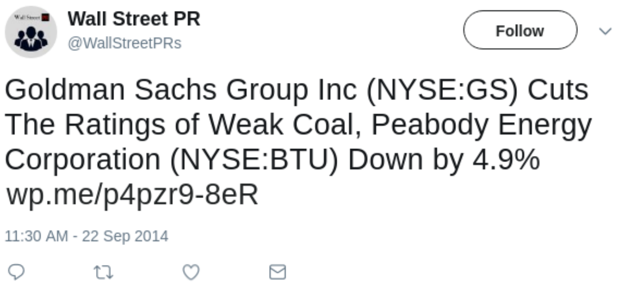
*Selection preference with regard to industry sector:* In the first analysis, we analyze whether specific industries are more frequently selected and if there are differences in average per-tweet returns among industries. We proceed in four steps and present the results in table 8: First, we calculate the relative frequency of stocks as found in the S&P 500 during our backtest period, and compare it with the frequency of industries mentioned in the full sample of tweets (column "full vs. S&P 500"). Second, we compute the industry share within the set of tweets selected for trading and compare it with the industry share in the entirety of tweets (column "traded vs. full / tweet share"). Third, we again compare selected tweets with the entirety of tweets, but this time consider the industry share of stocks as mentioned in the tweets (column "traded vs. full / stock share"). Fourth, we analyze the contribution of industry sectors to the mean per-tweet excess return.

We make the following observations: First, the coverage of industries in tweets is very similar to the share of industries in the S&P 500. Second, tweets mentioning stocks from the consumer services, consumer goods, and industrial sectors are preferably selected for trading. Half of the selected tweets cover the consumer services sector. Third, tweets covering technology, consumer

|  |  | excess return |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
|  | market cap. quartile | mean | t-statistic | stock count | selected vs. full | tweet share | full sample tweet share |
| tweets selected for trading | Q1 | 0.0047 | 0.5537 | 5 | 0.16 | 0.72% | 4.39% |
|  | Q2 | 0.0031 | 1.6296 | 30 | 0.98 | 6.27% | 6.40% |
|  | Q3 | 0.0079 | 17.0061 | 36 | 2.57 | 36.22% | 14.10% |
|  | Q4 | 0.0031 | 12.3652 | 65 | 0.76 | 56.63% | 74.40% |
| selection only by in-sample training accuracy | Q1 | 0.0021 | 3.0127 | 17 | 0.45 | 1.99% | 4.39% |
|  | Q2 | -0.0005 | -3.0602 | 101 | 1.79 | 11.45% | 6.40% |
|  | Q3 | 0.0000 | 0.4463 | 170 | 1.54 | 21.78% | 14.10% |
|  | Q4 | -0.0007 | -15.5969 | 160 | 0.86 | 63.79% | 74.40% |

Table 9: **Tweet share along market capitalization quantile.** We calculate mean per-tweet excess returns (after transaction costs) and the share of tweets in selected and all tweets grouped by the mentioned stock's quartile of market capitalization. The gray column divides the fraction of selected tweets by the respective frequency in the full sample. Column "stock count" lists the number of distinct stocks in the respective group of selected tweets. Q1 refers to the quartile of lowest market capitalization, Q4 to the highest.

services, health care, consumer goods and basic materials stocks achieve higher-than-average mean per-tweet excess returns. These are also the sectors that contribute a high number of tweets to the set of selected tweets. By contrast, tweets from the financial sector do not contribute to the overall positive mean return, albeit providing 9 percent of the selected tweets. The exemplary tweet in figure 9 may provide an interpretation of this finding: The mentioned bank appears as initiator of a rating change for another company, and common sense would suggest that this message has virtually no effect on the bank's stock price. Many of the tweets mentioning financial sector stocks might therefore in fact be reports covering another company. Hence, our model would not be able to explain returns for stocks of the financial sector, which is consistent with our empirical analysis and subject to future research.



Goldman Sachs Group Inc (NYSE:GS) Cuts The Ratings of Weak Coal, Peabody Energy Corporation (NYSE:BTU) Down by 4.9% wp.me/p4pzr9-8eR

11:30 AM - 22 Sep 2014

Figure 9: **Example of a tweet with two mentioned stocks.**

*Selection preference with regard to market capitalization:* In a second analysis, we explore whether our strategy preferably selects stocks with a specific market capitalization. To this end, we calculate mean per-tweet excess returns and the share of tweets mentioning stocks from a specific market capitalization quartile. Results are depicted in table 9. When comparing the distribution of market capitalization for traded tweets versus our full sample, we observe that stocks from the third market capitalization quartile are preferably selected and contribute most to the mean excess return. This conclusion also holds when selecting tweets by their in-sample training accuracy only: Here, stocks from the second and third quartile are preferably selected, i.e., they exhibit higher accuracy values. One possible interpretation of these results could be the following: Tweets concerning stocks of small market capitalization are overall less frequent and therefore less likely to be selected by in-sample training accuracy. On the other hand, stocks from the largest market capitalization quartile are less frequently selected as the high number of tweets on these stocks constitutes a high degree of noise, for example from product reviews or consumer comments less related to stock price. Medium market cap stocks may therefore be a reasonable trade-off.

We summarize the findings of this section as follows: Trading signals are preferably generated for tweets mentioning consumer-related and technology stocks, and for stocks with medium market capitalization. These preferences in stock selection may partly be based on the filtering of traded stocks by in-sample training accuracy (section 3.4), i.e., the model has a better in-sample performance for stocks from these sectors and market capitalization quartiles and preferably selects these for trading.

## 5. Conclusion

In this paper, we use a systematic machine learning approach to extract and assess the predictive value of tweets for subsequent stock price reactions. In total, we work with more than nine million tweets and all S&P 500 constituents in a large-scale empirical study. We contribute to the existing literature in the following ways: First, we provide a self-contained description of a financial machine learning pipeline for automated news-based stock trading, from a stream of tweets as input to the generation of actionable market signals. Using domain knowledge and concepts from the existing literature, we handcraft features along three dimensions, i.e., directional indicators, relevance indicators and meta features. We then formulate the prediction task as a classification problem and apply a state-of-the-art machine learning method in the form of a random forest and a number of benchmark models. In the final step, we derive trading signals using a two-dimensional event filter to reduce the number of events and to increase the signal-to-noise ratio.

Second, we subject the tweet-based trading signals to a backtest in order to assess out-of-sample historical financial performance. We pay special attention to exposing trading signals to a rigorous, event-based trading simulation, thereby taking into account transaction costs and limiting single-stock risk exposure. We perform the backtest on minutely stock price data for all S&P 500 stocks. We find that the strategy based on a random forest clearly outperforms all benchmark models and yields statistically and economically significant annual returns of 6.4 percent and a Sharpe ratio close to 2.2 after taking into account transaction costs, leverage constraints and single-stock position limits.

Third, we unveil sources of probability and challenge the underlying economic rationale: The first set of analyses shows that the majority of profits is generated by temporally clustered tweets, corresponding to high-event situations. A second set of analyses reveals the importance of features and words for our predictions: We find that directional, relevance and meta features contribute with descending importance to our results. Features with positive textual content typically lead to a prediction of rising stock prices, and negative ones to a prediction of falling stock prices. This behavior is in line with the financial literature on sentiment extraction, see for example Tetlock (2007). Relevance indicators are important for both long and short tweets. In a third set of analyses, we show that tweets mentioning consumer-related or technology stocks are preferably selected. Furthermore, tweets mentioning stocks of medium market capitalization are preferred, which we interpret as a trade-off between a sufficient probability of coverage on Twitter on the one hand and a relatively low level of noisy tweets unrelated to a stock's performance on the other hand.

Overall, we have provided a self-contained description of a tweet-based statistical arbitrage trading strategy leveraging state-of-the-art machine learning methods. Addressing the specifics of event-based trading strategies, we have successfully demonstrated that our design is able to extract predictive information from tweets that can be translated to statistically and economically significant excess returns.

### Acknowledgments

The purchase of the Twitter dataset was supported by the GfK Verein e.V.. We are especially thankful to Raimund Wildner and Holger Dietrich for their support.

## Bibliography

Antweiler, W., Frank, M.Z., 2005. Is all that talk just noise? The information content of internet stock message boards. The Journal of Finance 59, 1259–1294.

Asness, C.S., 1997. The interaction of value and momentum strategies. Financial Analysts Journal 53, 29–36.

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

Biau, G., Scornet, E., 2016. A random forest guided tour. TEST 25, 197–227.

Bird, S., Klein, E., Loper, E., 2009. Natural language processing with Python. 1st ed., O'Reilly, Beijing.

Bollen, J., Mao, H., Zeng, X.J., 2011. Twitter mood predicts the stock market. Journal of Computational Science 2, 1–8.

Breiman, L., 2001. Random forests. Machine Learning 45, 5–32.

Das, S.R., Chen, M.Y., 2007. Yahoo! for Amazon: Sentiment extraction from small talk on the web. Management Science 53, 1375–1388.

Deinert, A., Fischer, T., Krauss, C., 2018. Statistical arbitrage in cryptocurrency markets. FAU Discussion Papers in Economics (forthcoming).

Dudoit, S., Fridlyand, J., 2003. Classification in microarray experiments, in: Speed, T. (Ed.), Statistical Analysis of Gene Expression Microarray Data. Chapman and Hall/CRC, New York, pp. 93–158.

Enke, D., Thawornwong, S., 2005. The use of data mining and neural networks for forecasting stock market returns. Expert Systems with Applications 29, 927–940.

Fama, E.F., 1970. Efficient capital markets: A review of theory and empirical work. The Journal of Finance 25, 383–417.

Fischer, T., Krauss, C., 2018. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research 270, 654 – 669.

Fung, G.P.C., Yu, J.X., Lam, W., 2003. Stock prediction: Integrating text mining approach using real-time news, in: 2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings., pp. 395–402.

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

Goldman Sachs, 2016. The role of big data in investing. URL: https://www.gsam.com/content/gsam/global/en/market-insights/gsam-insights/gsam-perspectives/2016/big-data/gsam-roundtable.html.

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

Hagenau, M., Liebmann, M., Neumann, D., 2013. Automated news reading: Stock price prediction based on financial news using context-capturing features. Decision Support Systems 55, 685–697.

Halls-Moore, M., 2014. Successful algorithmic trading. URL: https://www.quantstart.com/successful-algorithmic-trading-ebook.

Harish, B.S., Guru, D.S., Manjunath, S., 2010. Representation and classification of text documents: A brief review. International Journal of Computer Applications Special Issue on Recent Trends in Image Processing and Pattern Recognition 2, 110–119.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The elements of statistical learning: Data mining, inference, and prediction. Springer Series in Statistics. 2nd ed., Springer-Verlag, New York.

Henry, E., 2008. Are investors influenced by how earnings press releases are written? The Journal of Business Communication (1973) 45, 363–407.

Jha, V., 2016. Timing equity quant positions with short-horizon alphas. The Journal of Trading 11, 53–59.

J.P. Morgan, 2018. Natural language processing tested in the investment process through new partnership. URL: https://www.jpmorgan.com/global/detail/1320565833239?source=cib_di_jp_mal0418.

Khoshgoftaar, T.M., Hulse, J.V., Napolitano, A., 2011. Comparing boosting and bagging techniques with noisy and imbalanced data. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 41, 552–568.

Knoll, J., Stübinger, J., Grottke, M., 2018. Exploiting social media with higher-order factorization machines: Statistical arbitrage on high-frequency data of the S&P 500. Quantitative Finance (forthcoming).

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, 689–702.

Leung, M.T., Daouk, H., Chen, A.S., 2000. Forecasting stock indices: a comparison of classification and level estimation models. International Journal of Forecasting 16, 173–190.

Loughran, T., McDonald, B., 2011. When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. The Journal of Finance 66, 35–65.

McKinney, W., 2010. Data structures for statistical computing in Python, in: Proceedings of the 9th Python in Science Conference, pp. 51–56.

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

New York Stock Exchange, 2018. NYSE: Holidays and trading hours. URL: https://www.nyse.com/markets/hours-calendars.

Oliveira, N., Cortez, P., Areal, N., 2016. Stock market sentiment lexicon acquisition using microblogging data and statistical measures. Decision Support Systems 85, 62–73.

Oliveira, N., Cortez, P., Areal, N., 2017. The impact of microblogging data for stock market prediction: Using Twitter to predict returns, volatility, trading volume and survey sentiment indices. Expert Systems with Applications 73, 125–144.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

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

PostgreSQL Global Development Group, 2018. PostgreSQL database management system. URL: www.postgresql.org.

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

Pröllochs, N., Feuerriegel, S., Neumann, D., 2015. Generating domain-specific dictionaries using Bayesian learning, in: ECIS 2015 Completed Research Papers.

Python Software Foundation, 2016. Python 3.5.2 documentation. URL: https://docs.python.org/3.5/.

QuantQuote, 2018. QuantQuote - historical intraday stock data. URL: https://quantquote.com.

R Core Team, 2016. R: A language and environment for statistical computing. Vienna, Austria.

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

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

Segal, M.R., 2004. Machine learning benchmarks and random forest regression, in: UCSF: Center for Bioinformatics and Molecular Biostatistics.

S&P Dow Jones Indices, 2018. S&P 500. URL: https://us.spindices.com/indices/equity/sp-500.

Sprenger, T.O., Sandner, P.G., Tumasjan, A., Welpe, I.M., 2014. News or noise? Using Twitter to identify and understand company-specific news flow. Journal of Business Finance & Accounting 41, 791–830.

Sprenger, T.O., Tumasjan, A., Sandner, P.G., Welpe, I.M., 2013. Tweets and trades: The information content of stock microblogs. European Financial Management 20, 926–957.

Statista, 2018a. Countries with most Twitter users 2018. URL: https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/.

Statista, 2018b. Twitter: Number of active users 2010-2018. URL: https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/.

Tetlock, P.C., 2007. Giving content to investor sentiment: The role of media in the stock market. The Journal of Finance 62, 1139–1168.

Tetlock, P.C., Saar-Tsechansky, M., Macskassy, S., 2008. More than words: Quantifying language to measure firms' fundamentals. The Journal of Finance 63, 1437–1467.

Turner, M., 2015. This is the future of investing, and you probably can't afford it. Business Insider Deutschland 2015.

Twitter, 2017. Giving you more characters to express yourself. URL: https://blog.twitter.com/official/en_us/topics/product/2017/Giving-you-more-characters-to-express-yourself.html.

Van Der Walt, S., Colbert, S.C., Varoquaux, G., 2011. The NumPy array: A structure for efficient numerical computation. Computing in Science & Engineering 13, 22–30.

Varazzo, D., 2011. psycopg2. URL: http://initd.org/psycopg/.

Wilson, J., 2015. Viewed from space, corn output trails government estimates. Bloomberg News 2015.