732A92 Text Mining

CLASSIFYING STOCK PRICE MOVEMENTS BASED ON 8-K SEC FILINGS: A COMPARISON OF RECURRENT NEURAL NETWORKS AND GRADIENT BOOSTING DECISION TREES

December 9, 2019

Name: Julius Kittler
Student ID: julki092
Linköping University

Abstract

This text mining project makes use of filings from the U.S. Securities and Exchange Commission (SEC), in particular 8-K filings, to forecast short-term percentage changes in stock prices. 8-K filings have to be published by public companies in the U.S. for certain defined, business-relevant events. The forecasting problem is approached as a classification problem with five classes: large decrease, small decrease, no change, small increase and large increase (of the stock price before and after filing date of the 8-K filing). Instead of using existing data sets, a new, up-to-date data set is created, with data for the year 2018 and the first quarter of 2019. Furthermore, this project makes a contribution by comparing the previously successful methods for this problem: recurrent neural networks (RNNs) with word embeddings and gradient boosting decision trees (GBDT) with bag-of-words.

Contents

1 Introduction **5**
1.1 SEC Filings . . . 5
1.2 8-K Filings . . . 5
1.3 Stock Price Forecasting with 8-K filings . . . 5
1.4 Research Questions . . . 7

2 Theory **7**
2.1 Gradient Boosting Decision Tree (GBDT) . . . 7
2.1.1 GBDT in General . . . 7
2.1.2 CatBoost . . . 8
2.2 Long Short-Term Memory (LSTM) . . . 8

3 Data **9**
3.1 Sources . . . 9
3.2 Retrieval . . . 9
3.3 Processing . . . 9
3.4 Descriptive Statistics . . . 9
3.4.1 General . . . 9
3.4.2 Target Variable . . . 9
3.4.3 Feature Variables . . . 10

4 Method **10**
4.1 Loss Function . . . 10
4.1.1 GBDT . . . 11
4.1.2 LSTM . . . 11
4.2 Training, Testing, Validation . . . 12
4.3 Hyperparameter Tuning . . . 12
4.4 Evaluation . . . 12
4.4.1 Research Question 1: Metrics . . . 12
4.4.2 Research Question 2: Model Interpretation . . . 12
4.4.3 Research Question 3: Backtesting . . . 12

5 Results **12**

6 Discussion **12**

7 Conclusion **12**

8 References **12**

List of Figures

1 Distribution of the target variable before discretization 10
2 Distribution of the target variable after discretization 10

List of Tables

1 Overview of all 8-K sections and events 6
2 Descriptive statistics for the target variable after discretization 11

1 Introduction

Forecasting stock prices has been a relevant problem since the existence of publicly traded companies. Today, it is an even more relevant problem than ever before because we have the technological infrastructure to put our forecasts to practice in form of automatic trading systems. Not only can we obtain massive amounts of financial data via APIs, but we can also execute trades via APIs. With commission free trading becoming the industry standard in the U.S., executing trades via APIs is even offered for free by some companies nowadays [1].

1.1 SEC Filings

The SEC is a government agency in the United States with the mission to "protect investors, maintain fair, orderly, and efficient markets, and facilitate capital formation" [2]. An important task of the SEC is to ensure that publicly traded companies inform their shareholders and the general public about their business.

For instance, the SEC requires companies to publish their quarterly and annual results, and to inform shareholders about certain relevant events. For each of these purposes, companies have to file specific documents. For instance, the annual report corresponds to the 10-K, the quarterly report corresponds to the 10-Q and another report for specific relevant events corresponds to the 8-K filing.

Importantly, SEC filings are actively used by traders when making investment decisions. Many trading platforms such as Webull and thinkorswim also provide traders with the recent SEC filings of any tradable company (along with other information such as fundamental data, news data and historical prices). SEC filings are interesting for stock price forecasting because they are standardized, publicly accessible for free and because they contain relevant, objective and generally accurate information about companies.

1.2 8-K Filings

Companies need to publish an 8-K filing for major events relevant for their business. For instance, such events might be a change in the board of directors, a potential delisting from a stock exchange or a merger. To be precise, there are 31 different 8-K filing events from 9 different sections. A complete list of all events and sections, taken from the official SEC website [3], is shown in the table 1. One 8-K filing document may contain information for several such events. For instance, one particular 8-K filing document may contain information about Item 5.02, Item 6.02 and Item 9.01 at the same time. Every 8-K filing clearly states for which events it contains information. To see some examples of 8-K filings, one may go to the official SEC website. In particular, three examples can be found here: example 1, example 2, example 3.

In general, 8-K filings are due within four business days after the event [4], a relatively short time period. Because 8-K filing correspond to major events for the company and because they need to be published shortly after an event occurred, 8-K filings seem interesting for predicting short-term volatility in the stock market. Moreover, the important information in 8-K filings is generally represented in form of text data, whereas other filings such as the annual report often focus on numerical data represented in tabular form. Text data is relatively simple to extract from HTML documents in order to generate features for training machine learning models (compared to tabular and graphical data).

1.3 Stock Price Forecasting with 8-K filings

This section gives a short overview of previous work about forecasting stock prices with 8-K filings. Since the focus of this project is classification, this literature review also focuses on previous work with classification. There are few public research papers that made use of 8-K filings for stock price forecasts. However, the papers that are publicly available show promising results.

For instance, Lee et al. could achieve an increase in accuracy by 10 percent when including text data from 8-K filings into a baseline model that only used financial metrics [5]. This study was solving a classification task with three classes: price increase ($> 1\%$), price decrease ($< -1\%$) or no relevant change ($< |1\%|$). Compared to a random classification with 33 percent accuracy and a majority-class classification of 35 percent accuracy, the best model of the study could achieve a 55 percent accuracy on the test data. The main model used for this study was a random forest classifier, which outperformed other models such as multi layer perceptron and logistic regression. Unigram features of the text data were used along with non-negative matrix factorization for dimensionality reduction.

Table 1: Overview of all 8-K sections and events

Section	Item	Event
Registrant's Business and Operations	1.01	Entry into a Material Definitive Agreement
	1.02	Termination of a Material Definitive Agreement
	1.03	Bankruptcy or Receivership
	1.04	Mine Safety - Reporting of Shutdowns and Patterns of Violations
Financial Information	2.01	Completion of Acquisition or Disposition of Assets
	2.02	Results of Operations and Financial Condition
	2.03	Creation of a Direct Financial Obligation or an Obligation under an Off-Balance Sheet Arrangement of a Registrant
	2.04	Triggering Events That Accelerate or Increase a Direct Financial Obligation or an Obligation under an Off-Balance Sheet Arrangement
	2.05	Costs Associated with Exit or Disposal Activities
	2.06	Material Impairments
Securities and Trading Markets	3.01	Notice of Delisting or Failure to Satisfy a Continued Listing Rule or Standard; Transfer of Listing
	3.02	Unregistered Sales of Equity Securities
	3.03	Material Modification to Rights of Security Holders
Matters Related to Accountants and Financial Statements	4.01	Changes in Registrant's Certifying Accountant
	4.02	Non-Reliance on Previously Issued Financial Statements or a Related Audit Report or Completed Interim Review
Corporate Governance and Management	5.01	Changes in Control of Registrant
	5.02	Departure of Directors or Certain Officers; Election of Directors; Appointment of Certain Officers; Compensatory Arrangements of Certain Officers
	5.03	Amendments to Articles of Incorporation or Bylaws; Change in Fiscal Year
	5.04	Temporary Suspension of Trading Under Registrant's Employee Benefit Plans
	5.05	Amendment to Registrant's Code of Ethics, or Waiver of a Provision of the Code of Ethics
	5.06	Change in Shell Company Status
	5.07	Submission of Matters to a Vote of Security Holders
	5.08	Shareholder Director Nominations
Asset-Backed Securities	6.01	ABS Informational and Computational Material
	6.02	Change of Servicer or Trustee
	6.03	Change in Credit Enhancement or Other External Support
	6.04	Failure to Make a Required Distribution
	6.05	Securities Act Updating Disclosure
Regulation FD	7.01	Regulation FD Disclosure
Other Events	8.01	Other Events
Financial Statements and Exhibits	9.01	Financial Statements and Exhibits

Another study by Saleh et al. extended the research by Lee et al. [6]. In addition to the data from the 8-K filings, the researchers used text data from Twitter. Furthermore, they used convolutional neural networks

(CNNs) and recurrent neural networks (RNNs) instead of random forests. Again, a random classification was equivalent to 33 percent accuracy. Compared to a majority-class classification of 42 percent without twitter data and 49 percent with twitter data, best test accuracies were 51 percent and 53 percent respectively. Instead of unigram features, word embeddings from GloVe were used.

Holowczak et al. compared the performance of various common algorithms (random forest, naive bayes, support vector machine, k-nearest neighbor and ridge classifier) on a binay classification task (price increase vs. decrease) [7]. However, only 8-K filings for a particular event, item 4.01, were used. The best results were achieved with a linear support vector classifier with a classification accuracy of 54.4 percent on the test data. The text data had been transformed with frequency-inverse document frequency (tf-id) vectorization.

1.4 Research Questions

There are mainly three research questions that this report aims to address. The focus is not only on the stock price classification itself but importantly also on evaluating whether the best model could be used in practice.

1. Research question: Does a RNN with LSTM or the GBDT CatBoost perform better at classifying stock price movements based on text data from 8-K filings?

2. Research question: How can the best model be interpreted: In which scenario does it perform better or worse (event of the 8-K filing, industry and location the company)? What are the most important features?

3. Research question: Could the best model be profitable when used in an automatic trading strategy?

2 Theory

2.1 Gradient Boosting Decision Tree (GBDT)

Gradient boosting with decision trees became popular with the development of XGBoost in 2016 [8] and lightGBM in 2017 [9]. In 2018, another GBDT library called Catboost was published, which claims to handle categorical features particularly well. Furthermore, Catboost was shown to outperform XGBoost and lightGBM on various datasets with default parameters. For this reason, Catboost is used in this project. The implementational details of Catboost are rather involved and are therefore not fully covered in this report. Instead, what follows is a short introduction to GBDT and a summary about differences between Catboost and other GBDT implementations.

2.1.1 GBDT in General

In GBDT, several concepts commonly used in machine learning are combined: gradients, boosting and decision trees. The main idea of **boosting** is to iteratively fit simplistic, additive models (so-called weak models) such that each model reduces the errors made by the ensemble model in the corresponding previous iteration. This can be achieved by fitting each model directly on the residuals of the ensemble model in the previous iteration. Gradient boosting generalizes this approach: each weak model is fitted on the **gradient** of the residuals from the previous iteration [8] [11] [12], where the term residuals refers to a differentiable loss function.

Recall that the gradient takes a scalar function (returning 1 scalar value given P parameters) and returns a vector function (returning P values, each of them the partial derivative of the scalar function w.r.t. the pth parameter). In our case, the scalar function is the differentiable loss function $L^{(t-1)}$:

$$\mathrm{L}^{(t-1)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)}), \tag{1}$$

where $i = 1, ..., N$ is the index for the training observations, $\hat{y_i}^{(t-1)}$ is the prediction for the ith training observation based on the ensemble model from the previous iteration and $l(y_i, \hat{y}_i^{(t-1)})$ is the loss for the ith training observation. If we take the gradient with respect to $\hat{y}_i^{(t-1)}$ as parameters, we get the following:

$$\nabla L^{(t-1)} = \left[\frac{\partial l(y_1, \hat{y}_1^{(t-1)})}{\partial \hat{y}_1^{(t-1)}}, \frac{\partial l(y_2, \hat{y}_2^{(t-1)})}{\partial \hat{y}_2^{(t-1)}}, ..., \frac{\partial l(y_N, \hat{y}_N^{(t-1)})}{\partial \hat{y}_N^{(t-1)}} \right] \tag{2}$$

This is a row vector with N elements, where each element corresponds to the derivative of the loss of the ith training observation with respect to its prediction $\hat{y_i}^{(t-1)}$. For instance, if we used the squared error loss function, $l(y_i, \hat{y}_i^{(t-1)}) = \frac{1}{2}\left(y_i - \hat{y}_i^{(t-1)}\right)^2$, the elements in the gradient vector would be $\frac{\partial l(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}} = -\left(y_i - \hat{y}_i^{(t-1)}\right) = \hat{y}_i^{(t-1)} - y_i$. Since the gradients here are absolute residuals, this corresponds to the simplest case mentioned above: fitting a weak model directly on the residuals of the ensemble model in the previous iteration. Importantly, we are not restricted to the squared error loss function. Instead, we can use any other differentiable loss function.

In every iteration t, we would like to add a weak model that helps to decrease the loss from the previous iteration $L^{(t-1)}$. Since the gradient points into the direction of the maximum increase of the loss function $L^{(t-1)}$ but we would like to decrease the loss, we consider the negative gradient $-L^{(t-1)}$. We start by initializing $\hat{y}_i^{(0)}$ as a constant for all training observations i, for instance as the mean of y from the training data. In every subsequent iteration t, we fit a weak model that predicts the negative gradient. We also learn a parameter that determines the step size towards the negative gradient. When we then add this model to the ensemble classifier, we essentially perform a step of gradient descent (not as usual by changing some parameters w, but instead by adding another weak model to an ensemble classifier). The prediction of the ensemble model for any observation i can be represented as:

$$\hat{y}_i = \sum_{t=0}^{T} \hat{y}_i^{(t)}, \tag{3}$$

where T is the number of iterations, i.e. weak models. Note that $\hat{y}_i$ may not be the final output of the model. For classification, we might pass $\hat{y}_i$ as input to a softmax function to get the predicted class for observation i.

Finally, note that different types of weak models can be used with gradient boosting. However, in GBDT specifically, every weak model is a **decision tree**. Decision trees are tree-shaped models that consist of nodes connected with branches. To make a prediction, for an observation i, its feature vector $\mathbf{x}_i$ is passed from the root node through the tree until it lands in a leaf node. For all vectors $\mathbf{x}_i$ in the same leaf node, the prediction is made. At each node within the tree, a decision is made about the branch along which the $\mathbf{x}_i$ is to be passed further. The decision is made based on a threshold τ_n for the value $\mathbf{x}_i^{(j)}$, where j is the index of the variable used at the threshold and n is the index of the node. Among others, the parameters that need to be learned are the thresholds τ_n, the variable indices j used at the thresholds and depending on the implementation also the structure of the tree (e.g. number of nodes, depth of the tree).

2.1.2 CatBoost

CatBoost resembles other GBDT systems such as XGBoost and lightGBM. However, it distinguishes itself in two main aspects: unbiased gradient estimates and categorical feature handling [10].

The first aspect addresses the following issue. In other GBDT libraries, gradients to be used in iteration t are computed for the same training observations based on which the last weak model in iteration $t-1$ was trained. This may lead to overfitting. CatBoost addresses this issue by training separate ensemble models M_i, where $i = 1, ..., N$ is the index for the training observations. In every iteration of CatBoost, a weak model is added to each of the ensemble models M_i. Importantly, each weak model is trained based on the gradients from the $1, ..., i-1$ previous training observations. This way, the weak model added to M_i is trained without the gradient value of the observation i. CatBoost uses further tricks to reduce computational complexity.

The second aspect relates to categorical features. Two common ways of dealing with categorical features in GBDT are a) one hot encoding, i.e. creating a binary feature for every level of a categorical feature, and b) computing summary statistics. An example of summary statistics described by CatBoost is the following. If the target variable is numeric, we can replace each level of the categorical feature with the mean of the target variable (of all the observations who have the same level of the categorical feature). The problem here is that this may lead to overfitting, in particular if there are only few observations with the same level of the categorical feature. CatBoost's innovation here is the following. First, the data set is shuffled randomly. Then, the categorical variable value is replaced for each observation as described above. However, when computing the summary statistic to replace the categorical variable value of observation i, only the previous observations $1, ..., i-1$ are used.

2.2 Long Short-Term Memory (LSTM)

tbs

3 Data

3.1 Sources

The data used for this text mining project comes from a variety of sources. The stock price data (with daily resolution) was retrieved with the financial API Tiingo. The SEC filings were downloaded from EDGAR, the official archive for SEC filings. The overview of companies by CIK, necessary to merge the SEC filings with the stock prices, was taken from the service Ranked and Filed. The industry categorization (SIC domain) for the companies was taken from SICCODE. Lastly, the overview of the 8-K events, used to extract the 8-K events from each filing, was taken from the SEC documentation.

3.2 Retrieval

The dataset used for training the models was created with the following steps:

1. A list of all 8-K filings for the first quarter of 2019 was retrieved from EDGAR. This list contained a total of 16449 8-K filings, including the CIK number (to identify the company) and the filing date.
2. The list was merged with the data from Ranked and Filed to get the ticker symbol, exchange market and SIC number (representing the industry) of the companies corresponding to each 8-K filing.
3. All 8-K filings from companies that were not listed on the NASDAQ, the NYSE or the AMEX (according to Ranked and Filed) were removed. In particular, the removed 8-K filings corresponded to companies listed on NYSE ARCA, OTC or OTCBB. The resulting list contained a total of 9318 8-K filings.
4. All 8-K filings, for which no stock price data was available from Tiingo were removed from the list. The resulting shortlist contained a total of 8030 8-K filings.
5. For the 8030 8-K filings, the stock price data was extracted from the Tiingo data, making use of the filing date from EDGAR. For each filing, the percentage change from the closing price of the day before the filing date and the open price of the day after the filing date was computed as target variable.
6. The 8030 8-K filings from the shortlist were downloaded from EDGAR. When processing the data (see below) 8-K filings with more than 1 Mio. characters were removed due to file size limitations of the libraries that were used for processing. This left a total of 7975 8-K filings for training the model.

3.3 Processing

Each raw 8-K filing, a text file containing HTML code, was processed as follows. First, graphics and embedded PDFs were removed and HTML tags were removed as well. Second, the resulting text data was tokenized with the natural language processing library spaCy, using the English language model en_core_web_sm. Third, stop words, non-alphabetical tokens and tokens with only one character were removed. Fourth, the remaining tokens were lemmatized with spaCy. Later on, very rare and frequent tokens were removed as well. This will be covered in the section about hyperparameter tuning.

3.4 Descriptive Statistics

3.4.1 General

3.4.2 Target Variable

The target variable is computed for each 8-K filing as follows. Let the company corresponding to the 8-K filing be company C.

The percentage change from the closing stock price of the day before the filing date and the open price of the day after the filing date is computed.

An overview of the discretized target variable can be found in table 2. Also, figures are in Figure 2 shows a boat.

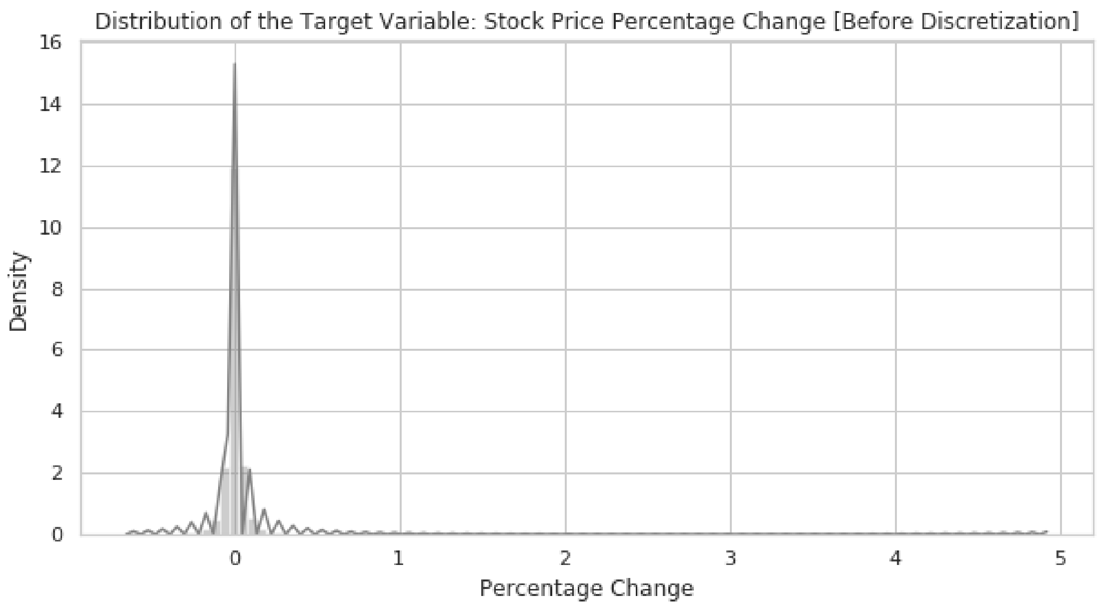

Figure 1: Distribution of the target variable before discretization

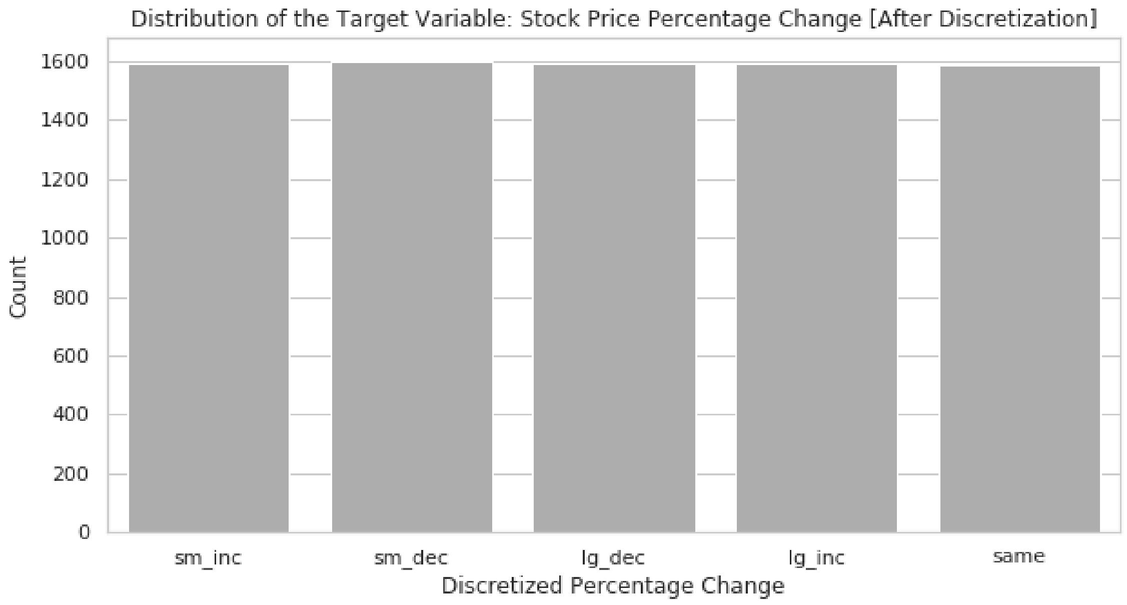

Figure 2: Distribution of the target variable after discretization

3.4.3 Feature Variables

4 Method

4.1 Loss Function

For multi-class classification problems, the softmax function is generally used as loss function (in neural networks and in gradient boosting trees). Hence, it is also used for this project. However, the loss function differs slightly

Table 2: Descriptive statistics for the target variable after discretization

Target	min	max	size	mean	median	std
lg_dec	-0.6387	-0.0226	1597	-0.0692	-0.0465	0.0696
sm_dec	-0.0226	-0.0053	1600	-0.0124	-0.0118	0.0049
same	-0.0052	0.0049	1591	-0.0001	-0.0001	0.0029
sm_inc	0.0050	0.0226	1592	0.0123	0.0116	0.0049
lg_inc	0.0226	4.9105	1595	0.0761	0.0491	0.1509

between the two models.

4.1.1 GBDT

The loss function used for CatBoost is called MultiClass loss function and is essentially a weighted softmax function [13]. It is computed as a scalar value across all observations and classes. The definition is as follows:

$$L = \frac{\sum_{i=1}^{N} w_i \log\left(\frac{\exp a_{i,t_i}}{\sum_{j=0}^{M-1} \exp a_{i,j}}\right)}{\sum_{i=1}^{N} w_i}, \tag{4}$$

where $i = 1, ..., N$ is the index for the observation, $j = 0, ..., M-1$ is the index of the class, $a_{i,j}$ is the model output for observation i and class j, w_i is the weight assigned for observation i during training, and t_i is the index of the true class of observation i so that a_{i,t_i} is the model output for the true class of observation i.

Essentially, the term in the parentheses is the softmax function, which returns a value between 0 and 1 for each observation i to belong to its true class t_i. If the model performs well for observation i, then the softmax value will be large. For instance, if the value is 1, this means that according to the model, we have a probability of 1 that observation i is from class t_i, i.e. from the true class of the observation. In other words, the model would perfectly identify the true class of observation i. Since the log is taken, and since $log(1) = 0$, the loss incurred for this perfectly classified observation will be 0. If the softmax value is smaller than 1, the loss for an observation will be negative and will hence affect the training process.

4.1.2 LSTM

The loss function used for the LSTM is similar to the loss function from CatBoost. However, the observations i are not weighted by any particular weights w_i. Essentially, a softmax function is used in the output layer so that the model directly outputs probabilities for each observation i to belong to class j. Then, the loss function CategoricalCrossentropy is used in TensorFlow [14]. For a multi-class problem, the categorical cross-entropy is defined as follows:

$$L = \frac{\sum_{i=1}^{N} \log\left(\frac{\exp a_{i,t_i}}{\sum_{j=0}^{M-1} \exp a_{i,j}}\right)}{N}, \tag{5}$$

where the notation is the same as in the loss function for GBDT. Note that in other notations this loss function usually has another sum over the classes along with an indicator function that is multiplied with the log (implying whether observation i is from class j). However, here a_{i,t_i} is defined as the model output for the true class of observation i and hence the indicator function is not needed.

4.2 Training, Testing, Validation

4.3 Hyperparameter Tuning

4.4 Evaluation

4.4.1 Research Question 1: Metrics

For evaluating the results, confusion matrix, accuracy and class-wise precision are reported. The precision is particularly important for considering potential potential trading strategies based on the model.

Class-wise precision is computed for each class individually. For a given class c, the class-wise precision is defined as the number of observations of class c that were classified as class c (TP for true positive), divided by the number of any observations that were classified as class c, regardless of whether they are of class c or not (TP + FP for the sum of all true positives and false positives).

$$\text{precision}_c = \frac{TP_c}{FP_c + TP_c} \tag{6}$$

Precision is chosen as prioritized evaluation metric in order to reflect the assumptions that we want to avoid loosing money in a trade and that missing out on a good trade is acceptable. To explain this, it is best to consider an example. Assume that we take a long position whenever the model classifies a large stock price increase (`lg_inc`), i.e. we buy stocks expecting that their stock price will increase. If the stock price decreases instead, we would loose money. To avoid this loss, we ideally want that among all cases where the model classifies a large increase (FP + TP), the stock price actually increases (TP). This means that we care about the precision.

In addition to reporting the evaluation metrics numerically, they are also visualized by certain categorical variables (e.g. industry of the company, event of the 8-K filings, location of the company) in order to identify issues as well as future potentials of the models.

4.4.2 Research Question 2: Model Interpretation

tbd

4.4.3 Research Question 3: Backtesting

tbd

5 Results

6 Discussion

7 Conclusion

8 References

References

[1] *Alpaca - Commission-Free API First Stock Brokerage*, en. [Online]. Available: `https://alpaca.markets` (visited on 11/23/2019).

[2] *SEC.gov | What We Do*. [Online]. Available: `https://www.sec.gov/Article/whatwedo.html` (visited on 11/23/2019).

[3] *SEC.gov | Form 8-K*. [Online]. Available: `https://www.sec.gov/fast-answers/answersform8khtm.html` (visited on 11/23/2019).

[4] W. Kenton, *8-K (Form 8k)*, en. [Online]. Available: `https://www.investopedia.com/terms/1/8-k.asp` (visited on 11/23/2019).

[5] H. Lee, M. Surdeanu, B. MacCartney, and D. Jurafsky, “On the Importance of Text Analysis for Stock Price Prediction.,” in *LREC*, 2014, pp. 1170–1175.

[6] M. Saleh and S. Nair, “Neural based event-driven stock rally prediction using SEC filings and Twitter data,” en, p. 8,

[7] R. Holowczak, D. Louton, and H. Saraoglu, “Testing market response to auditor change filings: A comparison of machine learning classifiers,” en, *The Journal of Finance and Data Science*, vol. 5, no. 1, pp. 48–59, Mar. 2019, ISSN: 24059188. DOI: `10.1016/j.jfds.2018.08.001`. [Online]. Available: `https://linkinghub.elsevier.com/retrieve/pii/S2405918818300096` (visited on 11/28/2019).

[8] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” en, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16*, pp. 785–794, 2016, arXiv: 1603.02754. DOI: `10.1145/2939672.2939785`. [Online]. Available: `http://arxiv.org/abs/1603.02754` (visited on 12/02/2019).

[9] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, “Lightgbm: A highly efficient gradient boosting decision tree,” in *Advances in Neural Information Processing Systems*, 2017, pp. 3146–3154.

[10] A. V. Dorogush, V. Ershov, and A. Gulin, “CatBoost: Gradient boosting with categorical features support,” *arXiv preprint arXiv:1810.11363*, 2018.

[11] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Annals of statistics*, pp. 1189–1232, 2001.

[12] *A Kaggle Master Explains Gradient Boosting | No Free Hunch*. [Online]. Available: `http://blog.kaggle.com/2017/01/23/a-kaggle-master-explains-gradient-boosting/` (visited on 12/06/2019).

[13] *Multiclassification: Objectives and metrics - CatBoost. Documentation*, en, concept. [Online]. Available: `https://catboost.ai/docs/concepts/loss-functions-multiclassification.html` (visited on 12/02/2019).

[14] *Module: Tf.losses | TensorFlow Core r2.0*. [Online]. Available: `https://www.tensorflow.org/api_docs/python/tf/losses` (visited on 12/02/2019).