Quantitative Trading with Machine Learning

René M. Glawion
SUID: 06605861
Stanford University
rglawion@stanford.edu

Abstract

We explore the application of Machine Learning for predicting the return of the VW stock by using the information of stock returns in its supply chain. Starting a horse race between Elastic Nets, Decision Trees, XGBoost, and LightGBM, we find that Elastic Nets generates the highest prediction accuracy across forecast horizons. A trading strategy based on this analysis leads to increased trading profits up to three times compared with a simple buy and hold strategy.

1. Introduction

Motivated by the current retreat from 50 years of "Just in Time" manufacturing of carmakers, we analyze the interdependencies of stock prices of Volkswagen AG (VW), which is the biggest car manufacturer in the world, and its suppliers[1]. More specifically, the project aims to utilize machine learning techniques to find correlations and even better causations in the stock price movements of VW and its supply chain, which we exploit to build a trading strategy. We show that an Elastic Net can generate a trading profit of 21.49% compared to a buy and hold strategy, which results in a gain of 7.98% over an eight-year testing period.

Machine learning (ML) techniques have been used for various applications in finance and supply chain management. For example Kou et al. (2019) use it for systematic risk analysis, Akita et al. (2016) employ ML techniques to predict price movements, and Makkar et al. (2019) use it for supply chain optimization. Here we go one step further and apply ML techniques to predict future stock price returns by utilizing the history of the company's own stock and the ones of its suppliers.

Furthermore, in contrast to most of the literature, we not only look at the direction of future price movements of stocks but at the actual values of those returns. We argue that due to the stylized facts of stock market returns, this is the most suitable approach: First, stock market returns feature positive skewness. Hence, it is easy to achieve a high directional forecast accuracy, because naturally more returns are positive. Second, stock returns show excess kurtosis. Thus, it is also crucial to predict the extent of returns. Otherwise, one significant adverse event can wipe out a whole year of trading profits.

[1] See https://www.wsj.com/articles/auto-makers-retreat-from-50-years-of-just-in-time-manufacturing-11620051251.

2. Related work

Brownstone (1996) was one of the first authors to apply machine learning techniques to predict stock prices. He used neural networks to obtain daily Market close prices of the FTSE 100 Index 5 days ahead and 25 days ahead. Similar to our study, he measured the accuracy in terms of the mean square error and terms of the root mean square error. As a benchmark, he applied a multiple linear regression. In his work, the multiple linear regression results in almost the same predictive accuracy as the more complex neural networks.

However, more recent studies have overcome those results when focusing on the direction of stock price movements. For example Ballings et al. (2015) collected stock price data on 5767 European stocks and benchmark single classifiers as well as ensemble methods. They find that random forests perform best in binary predictions. Additionally, they provide an extensive literature review which is summarized in Ballings et al. (2015, Table 1).

In line with that literature, Wu et al. (2006) use decision trees to identify turning points in stock price movements on the Taiwan and NASDAQ stock markets. In their best-case scenario, they can predict 66% of the turning points correctly. A newer study of Patel et al. (2015) compare neural networks, support vector machines, naive-Bayes, and random forest to predict the direction of stock price movements. They also find that random forests perform best on their dataset of the S&P Bombay Stock Exchange (BSE) Sensex between 2003 to 2012. Due to computing time constraints during this study's write-up, we decided to use decision trees in favor of the ensemble method random forests.

Gumelar et al. (2020) use Extreme Gradient Boosting (XGBoost) and compare it to Long Short-Term Memory (LSTM) neural networks. They show that XGBoost is able to outperform LSTM and achieve a prediction accuracy of 99%. Sun et al. (2020) use different ML algorithms to predict the price movements of cryptocurrencies. They find that LightGBM provides the highest accuracy in predicting the ups and downs of those assets with an accuracy of up to 93% over a 2-day period. We, therefore, include both XGBoost and LightGBM in our analysis. Thus, we are also able to compare the decision trees to ensemble methods.

3. Dataset and features

We identified 70 companies in the supply chain of Volkswagen AG from the Volkswagen page (Volkswagen AG, 2021) and several other sources for news about the supply chain of VW (e.g., Patzer (2015)) and the VW Group Award for the best suppliers (e.g., Eisert (2018)). Of those 70 companies, 36 are traded on a public stock exchange, and 30 are traded at the same stock exchange in Frankfurt. For those publicly traded firms, we downloaded the time series for the time period from 01-01-2005 to 01-05-2021 with the help of the yahoo finance API[2]. Those time series include the ticker, open, high, low, and closing prices, as well as the stock exchange, currency, and other stock-specific data. We transform all level data to return data and use those as features. Figure 1 plots the trajectory of the VW stock over our observation period.

Additionally, to capture the shape of the overall economy, we collected a set of macroeconomic variables. Namely, the output gap, CPI, and the yield of the 10-year government bond of Germany, where the headquarter of VW is located. Further, to ascertain the overall structure of the yield curve, we included the 3-month Euribor rate and the 10-year Eurirs rate as well as 1-month and 6-months relative differences of those. Last, we included the iTraxx Crossover together with 1- and 6-month relative differences to model changes in credit risk in the Euro Area. We obtained all time series from Refinitiv Eikon.

4. Methods

4.1. Data processing

The data takes the form $\{X_t^{(i)}, y_{t+h}^{(i)}\}$, where $X_t^{(i)} \in \mathbb{R}^{T \times p}$ is a matrix of features, i.e. relative historic price changes. The rows of $X_t^{(i)}$ represent time, whereas the columns represent lagged stock returns of the specific stocks in the supply chain. With $y_{t+h}^{(i)} \in \mathbb{R}^T$ we denote the future stock return of Volkswagen in period $t+h$. The rows correspond to the same periods as the rows of $X_t^{(i)}$.

[2]See https://pypi.org/project/yfinance/.

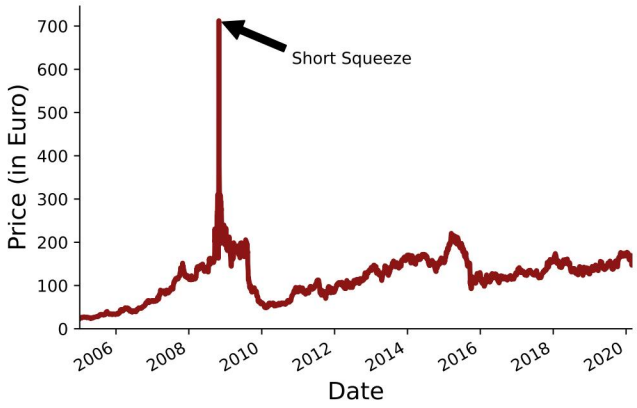

Figure 1. **Trajectory of the VW Stock**
The evolution of the VW stock price during our observation period from 01-01-2005 to 01-05-2020. The peak in 2008 was invoked by a failed takeover attempt of VW by Porsche (AP, 2008).

Missing values are a significant problem in our dataset since many companies were not traded at the beginning of our observation period. To compute our results, we set all missing values to 0. We justify this by arguing that including many zeros will bias all effects towards zero. Thus, if we find a significant effect, the real (unbiased) result will be more substantial. Hence, future work will even provide higher prediction accuracies.

4.2. Models

4.2.1. Elastic Net

The elastic net regression bridges the gap between the lasso and ridge regression. The estimator is given by

$$\min_{\theta \in \mathbb{R}^{p+1}} \frac{1}{2n}||y - X\theta||^2 + \frac{1}{2}(1-\alpha)||\theta||_2^2 + \alpha||\theta||_1,$$

where α is the weight for each penalty. In our case, stocks are excessively correlated; thus, the normal lasso can perform poorly since it has difficulties choosing among correlated variables. Like the ridge regression, the elastic net can shrink those correlated variables into groups with similar coefficients (Efron and Hastie, 2016, p. 316). In our analysis, we tune $\alpha \in [0, .5, 1]$ and choose the best performing estimator on the validation set.

4.2.2. Decision Trees

Decision trees are nonlinear models that break the input space into regions and have separate parameters for each region. In the regression framework, we use the least-squares loss to split the regions

$$L(R) = \frac{\sum_{i \in R}(y_i - \hat{y}_i)^2}{|R|},$$

where R_i corresponds to the region i (CS229 TA Notes, 2021). We tune the hyperparameters for the maximum depth of a tree during our estimation procedure, the minimum sample size for a split, the minimum samples in a leaf of a tree for a split, and the number of maximum features.

4.2.3. XGBOOST

Boosting methods and random forests have a lot in common. They both represent the fitted model by a sum of regression trees. However, there are some stark differences. In contrast to regression, trees boosting methods repeatedly grow shallow trees to the residuals and build up an additive model as a sum of trees (Efron and Hastie, 2016, sec. 17). The basic idea is to fit a model generated by the exponential family of response functions of the form

$$\eta(x) = G_B(x) = \sum_{b=1}^{B} g_b(x; \theta_b),$$

where η is the natural parameter of the conditional distribution $Y|X \sim x$, and the $g_b(x; \theta_b)$ are simple functions of shallow trees. Extreme Gradient Boosting (XGBoost) is a highly optimized version of this algorithm developed by (Chen and Guestrin, 2016) using fewer resources than other variants. It is an open-source project which can be obtained via GitHub (dlmc XGBoost, 2021).

4.2.4. LIGHTGBM

Developed by Microsoft research Light Gradient Boosting Machine (LightGBM) is another gradient boosting framework (Ke et al., 2017). LightGBM was especially designed for higher efficiency and scalability of boosting to large datasets. The main difference to XGBoost is that LightGBM grows the tree leaf-wise instead of level-wise. It will choose the leaf with maximum loss to grow. Holding the number of leafs fixed, leaf-wise algorithms tend to achieve lower loss than level-wise algorithms (Shi, 2007). For mathematical details and proves, we refer to the paper as well as the open-source GitHub repository of LightGBM (LightGBM, 2021).

4.3. Model evaluation criteria

The model we use for our analysis reads

$$\mathbb{E}[y_{t,t+h}|\mathcal{F}_t] \approx g(X_t, \theta), \tag{1}$$

where $y_{t,t+h} \in \mathbb{R}$ denotes the return of the VW stock over the period t to $t+h$, $h \in \{1, 5, 10, 15, 20, 25\}$, $\theta \in \mathbb{R}^{p+1}$ is a vector of weights (hyperparameters) we want to choose optimally in the sense of some given metric, $X_t^{(i)} \in \mathcal{F}_t$ is the matrix of features, which are the daily returns of all companies in the supply chain of VW over the past 28 days, $\mathcal{F}_t$ is the filtration. In our dataset we have $p = 850$ features. The function $g(\cdot)$ depends on the machine learning approach.

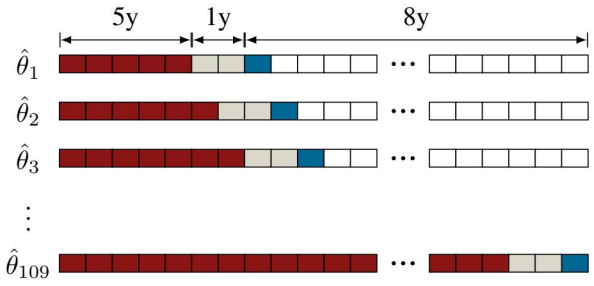

Figure 2. **Training, validation and testing procedure**
The data ranges from January 2005 to February 2020. The training period (red) initially spans 5 years and increases by one month after each validation step. Each of the 109 validation steps delivers a new set of parameter estimates. Each validation window (westar) covers 1 year and is rolled forward. After each validation step, there is one month of out-of-sample testing (blue). Note that the figure is for illustrative purposes only and not drawn to scale.

The model is a regression problem. Thus, we decided to use the root mean squared error (RMSE) metric to fit the models, which is defined by

$$RMSE(y_{t,t+h}, \hat{y}_{t,t+h}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(y_{t,t+h}^{(i)} - \hat{y}_{t,t+h}^{(i)}\right)^2},$$

where y denotes the realized stock return of VW, $\hat{y}$ is the predicted return, and T is the number of samples used for training, validation, and testing.

4.4. Splitting the dataset

To mitigate the risk of overfitting, we divide the dataset into three subsamples. Regularization is controlled by the tuning of the aforementioned hyperparameters. First, using a given combination of hyperparameters, the parameter vector θ^T is estimated on the training sample. Second, the model performance gets evaluated, in terms of forecast RMSE, on the validation sample. A search across provided hyperparameter combinations points to the specification that delivers the lowest error on the validation sample. After that, we use a test sample that the model has never seen before to ultimately determine the performance in terms of the RMSE and several additional metrics.

Further, we do not split the data randomly but rather sequentially to account for our data's time series nature. In the beginning, we start with a five-year training period which, a one-year validation period and a one-month test period. Afterward, we roll over this window, including the test set into our training and validation set, and start with a fresh test month. Figure 2 visualizes this procedure.

5. Results

Table 1 summarizes the prediction performances. First, we notice that for all four models, the error metrics increase with increasing forecast horizon h. For example, using the Elastic Net the RMSE increases from 1.74% for $h = 1$ to 8.25% for $h = 25$ days. Second, for all forecast horizons, the Elastic Net results in the lowest RMSE on the test set, beating LightGBM with a tight margin. The same holds for the MAE. However, looking at the raw forecasting error, we notice that the Elastic Net tends to underestimate the actual performance of the VW stock. In contrast, XGBoost tends to overestimate the performance, and the raw error, $y_{t+h} - \hat{y}_{t+h}$, is on the average negative for all forecast horizons.

Last, we look at the time we needed for each estimation step, which is presented in the last columns of table 1. This is the average time it took our computer to perform the procedure illustrated in figure 2. Again, we observe that overall Elastic Net gave the best performance. For longer time horizons, one can also use decision trees if time is a major concern. However, for a difference in a range of 1 second, we argue that Elastic Nets' much better prediction accuracy is still preferable. If timing is not a major issue, one can also use LightGBM to predict stock prices. However, it takes on average ten times longer to fit and predict the model and the prediction accuracy is lower compared to Elastic Net. Regarding XGBoost, we do not see any practical usage for our specific case. It resulted in one of the worst prediction accuracies and took on average 2 minutes per one month iteration to fit.

Table 1. **Prediction performances on the test set**
The table presents the root mean squared error (RMSE), the mean absolute error (MAE), the raw error ($y_{t+h} - \hat{y}_{t+h}$), and the estimation time for each model.

Model	h	RMSE	MAE	error	time
Elastic Net	1	.0174	.0137	.0001	4.62
	5	.0401	.0326	.0015	4.88
	10	.0553	.0463	.0023	4.81
	15	.0654	.0566	.0021	4.91
	20	.0751	.0666	-.0000	4.92
	25	.0825	.0731	.0007	5.11
Decision Tree	1	.0190	.0145	.0006	11.37
Regressor	5	.0422	.0338	.0005	7.41
	10	.0629	.0514	.0042	4.65
	15	.0728	.0606	.0023	4.11
	20	.0952	.0791	.0014	4.15
	25	.1025	.0890	-.0010	4.20
XGBoost	1	.0176	.0138	-.0002	92.41
	5	.0410	.0336	-.0012	107.03
	10	.0601	.0510	-.0072	105.65
	15	.0714	.0623	-.0100	106.93
	20	.0850	.0747	-.0121	109.67
	25	.0920	.0818	-.0193	113.64
LightGBM	1	.0177	.0139	.0003	44.81
	5	.0408	.0338	-.0002	47.70
	10	.0563	.0478	-.0020	53.42
	15	.0680	.0592	-.0024	54.84
	20	.0815	.0724	-.0085	53.93
	25	.0885	.0783	-.0041	56.28

6. Turning knowledge into profit

Now that we have models which produce reasonable predictions, we can turn to the question of how to make money out of this[3]. Due to our regression analysis, we can think of several trading strategies. One can reproduce the results from the literature and just use the sign of our prediction as a signal to determine whether the stock will go up or down over the next days or months. Additionally, we can also choose thresholds to set up trades.

We set up our strategy as follows: We start with a bankroll of 1,000 EUR. If we do not own stock, we buy one position if our forecast predicts that the stock price will increase by more than the predetermined threshold over the period h. Contrastingly, if we predict a decrease in the stock price below the threshold over the period h, we sell the stock and go short one position. If we predict an increase, we close out the short position and go long again. Here, we abstract from transaction costs. Thus, bid and ask prices are the same, which in our case is the closing price of each day at the Frankfurt Stock Exchange.

Table 2 presents the results of possible strategies. The columns refer to predetermined thresholds and the rows to different forecast horizons h. The entries give the profit on trading, and the numbers in brackets represent the number of transactions used by each strategy. First, we notice the overall pattern that, in general, with longer forecast horizons and larger thresholds, the number of transactions decreases. That is important when replicating one of those strategies in the real world. For example, LightGBM with a one day forecast horizons and a threshold of 0% results in performing over 1,000 transactions. Assuming transaction costs of 1 Euro per trade that is higher than the overall budget of the strategy!

Second, we see that Elastic Net performs well again, especially with low threshold values and higher forecast horizons. A threshold of 0% and a forecast horizon of $h = 20$ results in the best performing strategy of 21.49% overall profit. As a benchmark, we consult a buy and hold strategy. With

[3]Note: Most daily returns are in the range of ±2%. Thus, treating the return series as a martingale and always predicting no change will result in almost the same RMSE. Nevertheless, a trading strategy based on the predictions can still be profitable.

Table 2. **Profits of the trading algorithm**
The table presents the trading profit in percent for each model and forecast horizon. In brackets, we provide the number of trades for each strategy. For comparison, a buy and hold strategy would have resulted in a profit of 7.98%.

		Threshold			
Model	h	0%	.25%	.5%	1%
Elastic Net	1	-9.07	-6.33	-8.69	-7.83
		(283)	(61)	(25)	(9)
	5	9.56	-2.22	-5.24	-7.20
		(127)	(61)	(53)	(13)
	10	7.16	1.09	-5.84	-8.87
		(67)	(53)	(37)	(13)
	15	18.22	14.48	3.03	-2.39
		(79)	(53)	(73)	(21)
	20	21.49	14.38	13.93	-5.28
		(59)	(45)	(57)	(25)
	25	16.61	14.96	10.01	-0.91
		(87)	(91)	(69)	(69)
Decision Tree Regressor	1	-8.02	-5.66	-5.69	-5.43
		(319)	(205)	(165)	(125)
	5	-3.56	-12.39	-7.85	-7.33
		(237)	(233)	(217)	(201)
	10	-6.16	-6.91	-6.36	-3.59
		(435)	(431)	(395)	(367)
	15	-2.22	-4.37	-5.16	-6.69
		(485)	(473)	(461)	(449)
	20	7.97	4.85	7.10	3.11
		(399)	(395)	(395)	(349)
	25	7.82	9.25	6.32	1.64
		(517)	(509)	(517)	(533)
XGBoost	1	0.63	1.41	-2.40	-6.22
		(191)	(165)	(85)	(41)
	5	-8.83	-10.87	-9.54	-11.72
		(439)	(399)	(339)	(303)
	10	-4.75	-7.94	-7.26	-7.09
		(519)	(515)	(491)	(423)
	15	-2.59	4.05	2.28	3.27
		(483)	(503)	(511)	(463)
	20	6.57	7.63	0.75	4.37
		(459)	(459)	(467)	(451)
	25	1.43	1.58	1.03	4.44
		(391)	(391)	(403)	(417)
LightGBM	1	9.85	-5.09	-7.54	-5.98
		(1121)	(409)	(169)	(77)
	5	5.39	-10.95	-3.27	-6.20
		(687)	(619)	(517)	(369)
	10	-12.07	-4.73	0.27	-0.76
		(507)	(545)	(517)	(509)
	15	3.51	7.46	8.06	9.41
		(483)	(451)	(443)	(439)
	20	2.91	6.67	17.04	10.34
		(471)	(447)	(439)	(439)
	25	12.43	10.00	8.18	3.69
		(547)	(531)	(527)	(475)

an initial bankroll of 1,000 Euro, this results in a profit of 7.98% over the testing period. Thus, an Elastic Net can beat that by a factor of roughly 3.

Third, we see that Elastic Net tends to perform better with lower thresholds. Together with the results in table 1 we conclude that the predictions of the Elastic Net tend to lie around zero. Thus, it is interesting to see that our second best model LightGBM tends to perform better with higher thresholds. That gives rise to analyzing larger thresholds with LightGBM regression models. This analysis could result in better performances with even fewer trades so that it is easier to convey those results to actual real-world trading strategies.

7. Conclusion

This study provides evidence that trading strategies based on machine learning models and supply chains can outperform simple buy and hold strategies. The Elastic Net generates a trading profit three times larger than the buy and hold strategy in our best-case scenario. Further, using regression models instead of classification models, we can enhance the trading thresholds and only execute trades with an expected profit above a predetermined threshold.

In future work, it will be valuable to integrate ML methods. For example, Guo et al. (2021) combine both LightGBM and LSTM models to predict the direction of stocks. They only achieve an accuracy of 54.1%, but it would be interesting to enhance their approach to our regression framework with the information of the supply chain. For example, Taiwan Semiconductor Manufacturing Company (TSMC) is a supplier for both Infineon and Nvidia, which are both suppliers of Volkswagen. Motivated by the current capacity limit of TSMC and other chip manufacturers, we will also want to include those in our database. This analysis should help to predict stock price movements of Volkswagen AG itself[4]. Additionally, here we only analyzed the return of a trading strategy. Due to personal risk tolerance and regulations in the banking industry, it will be necessary to measure the ML-based trading strategies' risk in terms of standard deviations, maximum drawdowns, and several other risk measures.

8. Acknowledgements and replication files

We thank Pengda Liu for his help in developing the ideas for this project and helpful discussions throughout this study. We also thank Josie Oetjen and Gabriel Spil for helpful discussions and feedbacks. All replication files are on GitHub https://github.com/rglawion/cs229_project_report.

[4]See for example https://www.wsj.com/articles/car-chip-shortage-ford-vw-gm-11613152294.

References

Akita, R., Yoshihara, A., Matsubara, T., and Uehara, K. (2016). Deep learning for stock prediction using numerical and textual information. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pages 1–6. IEEE.

AP (2008). Squeezing the accelerator. The Economist https://www.economist.com/business/2008/10/29/squeezing-the-accelerator.

Ballings, M., Van den Poel, D., Hespeels, N., and Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert systems with Applications*, 42(20):7046–7056.

Brownstone, D. (1996). Using percentage accuracy to measure neural network predictions in stock market movements. *Neurocomputing*, 10(3):237–250.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

CS229 TA Notes (2021). Friday ta lecture: Decision trees + boosting.

dlmc XGBoost (2021). Xgboost. https://github.com/dmlc/xgboost.

Efron, B. and Hastie, T. (2016). *Computer age statistical inference*, volume 5. Cambridge University Press.

Eisert, R. (2018). Group award 2018 für 19 top-lieferanten:das sind volkswagens beste zulieferer. https://www.automobilwoche.de/article/20180524/NACHRICHTEN/180529953/group-award--fuer--top-lieferanten-das-sind-volkswagens-beste-zulieferer. Accessed: 2021-05-05.

Gumelar, A. B., Setyorini, H., Adi, D. P., Nilowardono, S., Widodo, A., Wibowo, A. T., Sulistyono, M. T., Christine, E., et al. (2020). Boosting the accuracy of stock market prediction using xgboost and long short-term memory. In *2020 International Seminar on Application for Technology of Information and Communication (iSemantic)*, pages 609–613. IEEE.

Guo, Y., Li, Y., and Xu, Y. (2021). Study on the application of lstm-lightgbm model in stock rise and fall prediction. In *MATEC Web of Conferences*, volume 336, page 05011. EDP Sciences.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154.

Kou, G., Chao, X., Peng, Y., Alsaadi, F. E., and Herrera-Viedma, E. (2019). Machine learning methods for systemic risk analysis in financial sectors. *Technological and Economic Development of Economy*, 25(5):716–742.

LightGBM (2021). Lightgbm. https://github.com/microsoft/LightGBM.

Makkar, S., Devi, G. N. R., and Solanki, V. K. (2019). Applications of machine learning techniques in supply chain optimization. In *International Conference on Intelligent Computing and Communication Technologies*, pages 861–869. Springer.

Patel, J., Shah, S., Thakkar, P., and Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert systems with applications*, 42(1):259–268.

Patzer, K.-H. (2015). 22 deutsche lieferanten dürfen mitspielen:fast, die vw-champions league der zulieferer. https://www.automobilwoche.de/article/20150810/NACHRICHTEN/150819994/-deutsche-lieferanten-duerfen-mitspielen-fast-die-vw-champions-league-der-zulieferer. Accessed: 2021-05-05.

Shi, H. (2007). *Best-first decision tree learning*. PhD thesis, The University of Waikato.

Sun, X., Liu, M., and Sima, Z. (2020). A novel cryptocurrency price trend forecasting model based on lightgbm. *Finance Research Letters*, 32:101084.

Volkswagen AG (2021). Das fast programm ist der weg des volkswagen konzerns, die zusammenarbeit mit seinen wichtigsten lieferanten zu intensivieren. https://www.vwgroupsupply.com/one-kbp-pub/de/kbp_public/fast_2/basicpage_for_general_pages__html_2.html. Accessed: 2021-05-05.

Wu, M.-C., Lin, S.-Y., and Lin, C.-H. (2006). An effective application of decision tree to stock trading. *Expert Systems with applications*, 31(2):270–274.