The Effect of Topics in Earning Calls on the Immediate Stock Market Reaction

Course: 10,365,1.00 - Computational Statistics (GSERM)

Supervisor: Prof. Dr. Francesco Audrino

Authors: Malte Bleeker, Huynh S. Tha

Date: 31.05.2023

GSERM
Global School in
Empirical Research Methods

Abstract

A positive relationship between Earning Call Sentiment and Stock Market Reaction has already been identified. Still, this utilization for prediction has yet to gain much attention. This paper investigates the predictive performance of earning call sentiment. Further, it proposes the utilization of topic model-based approaches to enable predictive performance that is robust against the presence of positive or negative sentiment on aspects that are of little concern to analysts. All S&P 500 earning calls from 2022 are utilized for this task. All assessed models perform better than a mean-based guess, indicating the value of topics and sentiment analysis for the prediction of stock market reactions. The best-performing models on the unseen test set are Random Forest and XGBoost, followed by the Lasso & Ridge Regression.

Table of Contents

The Analysis of Earning Calls

The SEC requires companies to disclose their financial information quarterly if the firm has more than 500 shareholders and more than $10 Million in assets (SEC, 2023). While not mandatory by legislation, it has become common for firm executives to hold earning calls to explain the reported financial numbers further, elaborate upon certain aspects and provide a future outlook to manage analyst expectations. This is also in the interest of analysts, providing the chance to ask follow-up questions to understand a firm's financial state better. Given the assumed relevance of the content of these calls and with the rise of new methods in Natural Language Processing (NLP), researchers have started to analyze various aspects of such earning calls to automatically extract relevant information, analyze the common sentiment, and more importantly try to predict the stock market reactions in response to these calls. A task can also be described as identifying "music between the noise," given the multiple factors influencing stock prices and also the difficulty of entangling the effect of the actual quarterly reported numbers from the earning call itself. Both events often occur on the same day (the earning call commonly after the market has closed to reduce stock volatility), making it difficult to attribute changes to the information disclosed during the call or reported figures (Huang et al., 2020). Nevertheless, given the complexity of financial markets and the efficient market theory, a forward-looking prediction with any degree of accuracy would be highly relevant and highly relevant. This paper builds upon prior approaches by assessing and benchmarking the predictive performance of topics in earning calls on the stock market reaction across different models. The topic approach is combined with a sentiment analysis that has already been shown in prior research to be somehow predictive of the stock market reaction.

In the following, relevant research in this field is shortly outlined without adequately claiming to capture the entire earning call research landscape. Secondly, the data collection and preparation outline, this paper's central part, and the critical focus are on the methodology, the application of predictive models, and the benchmark. A variety of various methods are applied, and their performance is benchmarked. The paper closes with a summary and discussion of the key finding, including a short, not definite, discussion of limitations.

Natural Language Processing of Earning Calls

Kearney & Liu, 2014 already identified in their review of textual sentiment in finance multiple research articles that investigated various aspects of the tone in earning calls and its implications on financial metrics. Early findings already indicated that a positive tone during earnings calls could mitigate the negative impact of a negative earnings surprise on abnormal returns (Doran et al., 2012). While those findings were originally primarily based on dictionary base methods, similar results were also made with more evolved machine learning-based models that enable contextual understanding (e.g., Bert) (De Amicis et al., 2021). There has been a shift from lexicon-based approaches towards primarily machine learning-based models (Algaba et al., 2020). The variety of use cases in research that combine sentiment with economic key metrics has even led to a term coined "sentometrics" (Algaba et al., 2020). Further also, more granular aspects, like participants' characteristics on the side of the executive team and the analysts, have been investigated and mostly found to have at least a minor impact. Also, other textual features, like vagueness or the degree of forward-looking statements, have gained attention (Qin, 2023). In recent years also, the role of individual topics has been researched (Han et al., 2021). In the paper of Huang et al., 2020 that proposes the FinBert model, a BERT model trained on a large corpus of financial texts, the authors also decide to utilize its domain-optimized embedding to build a classifier for ESG-related sentences in earning call transcripts. The main idea of separating earning call transcripts into topics is to allow the model to assign different weights to different topics and thereby, as an example, remove superficial small talk without disclosing information from a sentiment analysis that tries to predict the stock market reaction. This allows for a more accurate representation of what is discussed in an earning call, how much is spoken of, and how it is discussed. Further, it builds upon the assumption that the combination of what is talked about and how it is talked about, I crucial in accurately capturing the content of something.

The idea of connecting topic models with a perceived performance model for predictive tasks has been introduced previously. Researchers have also utilized Topic Models on research articles (primarily Latent-Dirichlet-Allocation) to predict the citations an article might receive. A task similarly focused on capturing the content of

something and understanding the effect of this topic composition on the performance metric, its "success" is often evaluated (Gupta et al., 2022).

This paper builds upon these ideas, but instead of utilizing explicit domain knowledge or focusing on specific aspects of the earning call, an unsupervised topic model is applied to enable the inclusion of significantly more topics while still representing a feasible approach from the implementation perspective. This also leads to this paper's key research question: How suitable are earning call topics and their sentiment in predicting the stock market reaction to the call? This also includes the question of whether the topic approach does outperform the approach of significantly lower dimensionality that just considers the absolute sentiment.

Methodology

Data Collection & Variables

The data underlying this research are all earning calls of S&P 500 firms in 2022. The central resource for this is the earning call transcripts. In addition, the dates of the earning calls are extracted, and the financial stock market reaction is collected. The sample consists of 1984 earning calls.

Stock Market Reaction: The stock market reaction is captured as defined by Medya et al., 2022, as the percent change in the stock price closing price the day before and the day after the earning call (Shock Based Values = SBV).

Volatility: The stock market's expectation of volatility is also captured and included with the CBOE Volatility Index (an S&P 500 option-based metric). It is captured for the day of the earning call and the change from the day before to the day after.

Industry: The industry is included on a high level based on its GICS Sector category.

Given the high interdependence of an earning call reaction with the market reactions of the firm's key competitors, no stock market changes on an industry level are included. E.g., a strong Market Share increase will drive down competitor shares, while general issues like supply chain constraints can result in a similar directional effect on all firms. Due to our database's limited value and technical constraints, the variable is not included.

Descriptive of the Response Variable: Stock Market Reaction (%)	
Mean	0.00062035
Standard deviation	0.067637792
Variance	0.004574871
Min	-0.339347409
25%	-0.035088324
50%	0.003727862
75%	0,040950166
Max	0.291948932
Sample size	1950

Preparation

Unsupervised Approach: Topic Modelling (BertTopic)

As the unsupervised Topic Model, BertModel, a Bert-based approach is utilized. The model has the advantage of having a better contextual understanding than previous models like LDA while still being able to be applied to large documents and text. Further, it assigns each sentence to a topic in contrast to a word-by-word basis, making it easier to interpret.

Firstly, only the presentation section of each transcript was extracted and separated into its sentences. This resulted in a total of around ~350k sentences. For the next step, the BertTopic package was utilized. Embedding was created for each sentence with Sentence-BERT (S-BERT) to maintain the contextual meaning. This mapped each sentence to a 768-dimensional dense vector space. 1280 GPU cuda cores were utilized for this first step to reduce the computational time. To mitigate the curse of dimensionality, UMAP was applied for the dimensionality reduction, reducing the embedding to 20 – a number that has reportedly been feasible in maintaining most of the structure while enabling reasonable performance for the clustering task. UMAP is also chosen over PCA, given its better ability to maintain local structures. HDBSCAN is selected over the more common KNN approach for clustering to enable outliers and reduce the computational complexity of high-dimensional vector similarity calculations. The number of clusters in topic modeling needs to be set individually and is a highly subjective choice. For this research, it was set to 200. However, only the topics with more than 500 sentences were afterward included as distinct topics in the further analysis. This procedure was chosen to achieve better interpretability with higher topic granularity by initially setting a higher number of topics and only afterward reducing the number used. All other topics were added to the topic null, which by default contains all sentences that could not yet be assigned to a specific topic. Following this, the length in words of all sentences per topic per earning call was utilized to calculate the share of each topic per earning call (cf. Attachment A for an overview of the earning call topics).

Sentiment Analysis (Finbert)

With the share of each topic per presentation available, the previously generated sentences were also used for sentiment analysis. FinBert is a BERT model trained on financial texts that perform better on classification tasks than the pre-trained base model (Huang et al., 2020). Each sentence was tokenized again, and 786- dimensional embedding vectors were created and inputted into the classification model developed by Huang et al., 2020 – providing confidence estimates for the three sentiment classifications "positive," "negative," and "neutral." Again, cuda cores were utilized to reduce the computational time. Each label was then coded as "positive=1", "neutral=0," and "negative=-1" and then multiplied with its confidence. As a result, a score of 0.5 might indicate 50% confidence in neutral & 50% confidence in positive, allowing for a more accurate estimate than just using the classification label directly.

Based on the sentence length, the topic per sentence, and the sentiment estimate per sentence, the share of topic per earning call was calculated, as well as the mean sentiment of each topic-earning call combination. As a result, the new data frame contains the Topic Share ($TopicShare_1$ to $TopicShare_n$) and Topic Sentiment ($TopicSentiment_1$ to $TopicSentiment_n$) for each earning call.

In this section, we present the empirical models to forecast the changes in stock prices following the earning call day. The first step involves employing traditional linear models and their variants to establish baseline results, then machine learning techniques, ranging from random forest to XGBoost and shallow neural networks, to enhance the prediction of the stock market reactions.

The fundamental regression task entails estimating a general function, denoted as $g(xi) = E(yi|xi)$, where yi represents the percentage of the ith firm's price changes, and εi denotes a random error component. The regression function $E(yi|xi)$ corresponds to the conditional expectation of yi given the covariate vectors described in the previous section. We aim to estimate the function $g(\cdot)$ using various methodologies, ranging from standard multiple linear regression to highly nonlinear and discontinuous approaches, such as a random forest.

Model Assessment

To effectively evaluate and assess the performance of those different models, we split the data into training and testing sets (75% and 25%, respectively), followed by applying cross-validation exclusively on the training set. By splitting the data into training and testing sets, we can train the model on a portion of the data while reserving another portion for independent evaluation. This separation enables us to gauge the model's generalization ability to unseen data. However, relying solely on a single train-test split may result in a biased evaluation due to the specific composition of the data. To address this limitation, cross-validation is employed. The training data is divided into five equal folds. Initially, one fold is assigned as the validation set, while the remaining four folds are used for training the model. This allows us to assess the model's performance on the validation fold while leveraging the other folds for training. This process is repeated by shifting to the next fold as the validation set and utilizing the remaining folds as training data. The iteration continues until the model's performance has been evaluated on all five folds, each serving as the validation set once. By completing the cross-validation procedure, we can learn the best-performing hyper-parameters for each model and check the model's ability to generalize to unseen data by running it through the independent test set. In both the initial data split and cross-validation, we employ random shuffling. This random shuffling ensures that the data points are distributed randomly between the training and test sets, preventing any biases that might arise from the original ordering of the data. By randomly shuffling the data, each sample has an equal chance of being included in the training or test set. This promotes a representative distribution of data across the sets and enables reliable model evaluation and accurate estimation of its generalization performance.

Method 1: Regression

Our baseline model follows the standard multiple regression framework, represented by the following form:

$$g(xi; \beta) = xi\beta$$

The regression parameters β, which are estimated using ordinary least squares (OLS), can be defined as follows:

$$\beta = \text{argmin} \parallel y - x\beta \parallel 2$$

In the case of a multilinear regression model, there are no specific hyper-parameters to tune as in other machine learning models. Consequently, we omit the cross-validation process in this scenario and train the model on the entire training set to assess its performance. By training the multilinear regression model on the complete training data, we obtained the following performance metrics:

Training MSE: 0.00406, and training R-squared: 0.12687

These metrics provide insights into the model's accuracy in predicting the training data. The MSE value of 0.00406 indicates an average squared difference between the predicted and actual values. Similarly, the R-squared value of 0.12687 indicates that the model explains approximately 12.69% of the variability in the training data. It is worth noting that the response variable variance is 0.0045725, which provides context for evaluating the model's performance. The model's MSE is lower than the response variable variance, indicating that the model's predictions exhibit less variance than the actual values.

Method 2: Lasso Regression

The LASSO (Least Absolute Shrinkage and Selection Operator) model is a commonly used regularization technique. Tibshirani 1996 introduced it as a method for variable selection and regularization in linear regression models. LASSO combines the least squares estimation with an L1 penalty term to achieve sparse solutions by shrinking the coefficients of irrelevant features toward zero.

In the LASSO model, the goal is to minimize the sum of squared residuals (as in ordinary least squares) subject to a constraint on the sum of the absolute values of the

coefficients. This constraint is proportional to the tuning parameter, lambda, which controls the amount of regularization applied. When lambda is set to zero, the LASSO model reduces to ordinary least squares, and as lambda increases, the model tends to produce sparser solutions. Mathematically, the LASSO model can be represented as:

minimize: RSS + $\lambda * \sum|\beta|$

where:

RSS: the residual sum of squares, measuring the difference between predicted and actual values.

λ: is the tuning parameter controlling the amount of regularization.

$\sum|\beta|$: represents the L1 norm of the coefficient vector β.

The L1 penalty term encourages sparsity in the solution, as it has the effect of shrinking some coefficients to precisely zero. This makes the LASSO model useful for feature selection, as it automatically performs variable selection by setting irrelevant or redundant features' coefficients to zero.

To tune the regularization parameter, lambda, in the Lasso model, we employed a grid search approach during the cross-validation process to select the optimal lambda value from the following options: [0.001, 0.01, 1.0, 10]. After analyzing the results, we identified the best lambda value to be 0.001, and the Lasso model achieved the following performance metrics:

Training MSE: 0.0044386, and training R-squared: 0.04527

The MSE value of 0.0044386 indicates a comparatively low average squared difference between the predicted and actual values. Comparing this to the response variable variance of 0.0045725, we observe that the model's predictions exhibit less variance than the original values. Furthermore, the R-squared value of 0.04527 suggests that the Lasso model explains approximately 4.53% of the variability present in the training data. While this may seem relatively low, it provides a quantitative measure of the

model's ability to capture and account for a portion of the variability within the training dataset.

Method 3: Ridge Regression

The Ridge model, also known as Tikhonov regularization, is another regularization technique frequently used in statistical modeling. It was developed as an extension of ordinary least squares regression to address the issue of multicollinearity, where predictor variables are highly correlated with each other.

The Ridge model uses an L2 penalty term to the least squares objective function, which adds a regularization term proportional to the square of the coefficients. This penalty term discourages large coefficient values, leading to more stable and robust models. Unlike LASSO, the Ridge model does not result in exactly zero coefficients but shrinks them towards zero without eliminating them. Mathematically, the Ridge model can be represented as:

minimize: RSS + λ * $\sum(\beta^2)$

where:

RSS: the residual sum of squares, measuring the difference between predicted and actual values.

λ: is the tuning parameter controlling the amount of regularization.

$\sum(\beta^2)$: represents the L2 norm (Euclidean norm) of the coefficient vector β.

The L2 penalty term encourages small but non-zero coefficients for all features, helping to reduce the impact of multicollinearity and stabilize the model's performance. The choice of the tuning parameter λ determines the balance between fitting the data well (minimizing RSS) and shrinking the coefficients.

We used a grid search approach to identify the optimal lambda value during cross-validation. We considered a range of lambda values, specifically [0.1, 1.0, 10.0, 100,

1000]. We systematically evaluated each lambda value to assess the model's performance. Upon analyzing the results, we determined that the best lambda value for the Ridge model was 1000, and the Ridge model yielded the following performance metrics:

Training MSE: 0.004439, and training R-squared: 0.04517

The mean squared error (MSE) value of 0.004439 indicates a relatively small average squared difference between the predicted and actual values. When comparing this to the variance of the response variable, which is 0.0045725, we observe that the model's predictions demonstrate lower variance than the original values. Additionally, the R-squared value of 0.04517 suggests that the Ridge model can account for approximately 4.52% of the variability present in the training data.

Method 4: Random Forest

Random forest is an ensemble learning method that combines the predictions of multiple decision trees, using bootstrap aggregating (bagging) to improve prediction accuracy (Breiman, 2001). Each decision tree T_i is trained on a randomly selected bootstrap sample S_i from the original training dataset. The construction of each tree involves recursively partitioning the data based on randomly selected subsets and optimal splitting criteria, typically using measures like Gini impurity or mean squared error.

Once the individual decision trees are trained, predictions are made by aggregating their outputs. For regression tasks, the predictions of all trees are averaged to obtain the final prediction. Mathematically, this can be represented as $\hat{y} = (1/m) * \Sigma(T_i(x))$.

Aggregating individual decision trees through bagging helps improve the model's generalization performance by reducing overfitting. Bagging achieves this by introducing randomness through bootstrap sampling, where each tree is trained on a different subset of the data. Combining the diverse predictions from multiple trees leads to a more robust and accurate prediction.

We employed grid search across the folds to determine the optimal hyperparameters for the Random Forest model. This involved systematically testing various combinations of the following parameters and selecting the configuration that yielded the best Mean Squared Error (MSE) performance:

- 'n_estimators': [50, 70, 100] - the number of decision trees to be included in the Random Forest ensemble. Increasing the number of estimators can enhance the model's predictive power but also lead to longer training times and potential overfitting.
- 'max_depth': [7, 9] - the maximum depth or level of each decision tree in the Random Forest. A higher number allows the trees to capture more complex relationships in the data. However, excessively deep trees may lead to overfitting.
- 'max_features': ['sqrt,' 0.5] - the number of features (predictors) randomly selected at each split in a decision tree. It controls the level of randomness and diversity among the trees, hence reducing overfitting.
- 'min_samples_leaf': [3, 5] - the minimum number of samples required to be present in a leaf node. Setting a higher value can prevent overfitting by enforcing a minimum number of samples in each leaf, which promotes generalization.

We trained the Random Forest model for each fold using a specific combination of these parameters. By exhaustively exploring the grid of possibilities, we identified the set of hyper-parameters that minimized the MSE, indicating the optimal configuration for our model.

During the training phase, we determined the best-performing hyperparameters, which are as follows:

{'max_depth': 9, 'max_features': 0.5, 'min_samples_leaf': 5, 'n_estimators': 100}

Additionally, we assessed the model's performance on the training data using these optimized hyperparameters, resulting in the following metrics:

Training MSE: 0.00290, and training R-squared: 0.37606

The mean squared error (MSE) value of 0.00290 indicates a low average squared difference between the predicted and actual values, suggesting that the model's predictions are close to the true values. When comparing this to the variance of the response variable, which is 0.0045725, we observe that the model's predictions exhibit lower variance, indicating good predictive performance. Additionally, the R-squared value of 0.37606 suggests that the model captures approximately 37.61% of the variability present in the training data.

Method 6: XGBoost

XGBoost (Extreme Gradient Boosting) is another machine learning algorithm that can be used for classification and regression tasks. It has become commonly applied in practice due to its low run times and high scalability while providing good performance. The original paper on the algorithm was written by Chen & Guestrin in 2016 (Chen & Guestrin, 2016). It is an ensemble method based on the gradient boosting framework, which combines multiple weak prediction models to create a robust and accurate predictive model. The weak models are decision trees, specifically boosted trees. Boosted trees are constructed iteratively, where each new tree is built to correct the errors made by the previous trees. This iterative process allows XGBoost to learn complex relationships and capture intricate patterns in the data. It optimizes an objective function by iteratively adding trees to minimize the desired loss function. The objective function consists of two components: a loss function that measures the model's performance and a regularization term that controls the complexity of the model. To prevent overfitting, similar to the previously introduced Lasso and Ridge regression, L1 regularization (Sum of absolute coefficients as penalty Term) and L2 regularization (Sum of squared coefficients as penalty term) are utilized (L1 via hyperparameter lambda and L2 via hyperparameter alpha).

We conducted the grid search across the folds to determine the optimal hyperparameters for the XGBoost model. This involved systematically testing various combinations of the following parameters and selecting the configuration that yielded the best Mean Squared Error (MSE) performance. It shall be noted that once optimal

parameter combinations with the predefined values were found, more granular values above/below the hyperparameter values identified at first were included, and the grid search assessment was repeated.

- learning_rate': [0.01, 0.05, 0.1] - The learning rate η (or shrinkage rate) determines the step size at each boosting iteration.
- 'n_estimators': [100, 200, 300] - the number of decision trees to be included in the XGBoost ensemble. Increasing the number of estimators can enhance the model's predictive power, but also lead to longer training times and potential overfitting.
- 'max_depth': [1,3,5,7] - the maximum depth or level of each tree in the XGBoost ensemble. Increasing the max_depth allows the trees to capture more complex relationships in the data but can lead to overfitting.
- 'reg_alpha': [0, 0.1, 0.5] - determines the strength of the L1 regularization term (sum of absolute coefficients as penalty term)
- 'reg_lambda': [1, 5, 10] - determines the strength of the L2 regularization term (sum of squared coefficients as penalty term)

During the training phase, we determined the best-performing hyperparameters, which are as follows:

{learning_rate = 0.05, max_depth = 3, n_estimators = 300, reg_alpha = 0.1, reg_lambda = 5}

The model's performance on the training data using these optimized hyperparameters resulted in the following metrics:

Training MSE: 0.0266938, and training R-squared: 0.2409852

The mean squared error (MSE) value of 0.0266938 indicates a low average squared difference between the predicted and actual values. When comparing this to the variance of the response variable, which is 0.0045725, we observe that the model's predictions exhibit lower variance, indicating better than random or mean-based

predictive performance. Additionally, the R-squared value of 0.2409852 indicates that the model captures approximately 24.1% of the variability present in the training data.

Method 7: Neural Network

Neural networks exist in various versions and structures, also some of which make them useful for continuous prediction tasks. The structure of a neural network typically consists of an input layer that receives the input data, several hidden layers that process and transform the data through a series of interconnected neurons, and an output layer that generates continuous predictions. Each neuron in the hidden layers applies an activation function to introduce non-linearity, enabling it to capture complex relationships in the data. The network's weights and biases are updated during the training process, for which various optimizer algorithms have been developed. All these components, the number of hidden layers, the number of neurons in each layer, the activation functions, and the optimizer algorithm must be defined and selected based on the specific objective and available data.

This section does not represent an exhaustive assessment of the latest structure recommendation but resides with the application of commonly used structures. Similar to a grid search, the following hyperparameters/structures and optimizers are assessed:

'optimizer': ['adam,' 'sgd'] - The Adam optimizer is widely used due to its computational efficiency and adjusts its learning rate dynamically during training. Stochastic Gradient Descent (a simpler predecessor) is more computationally expensive but has been found to have better generalization performance under small sample size conditions.

'kernel_regularizer': Similar to the previously introduced methods, also in neural networks, L1 and L2 regularization can be applied to each layer individually to prevent overfitting.

'epochs': [10, 20, 30] - The number of times the model is trained on the entire training dataset. Too many can lead to overfitting, while too few could lower the ability to capture underlying patterns.

'batch_size': [8, 16, 32] - The number of samples propagated through the network with each training iteration.

'num_layers': [1, 2, 3] - The number of hidden layers in the neural network. Similar to the epochs, too many layers can lead to overfitting, while too few could lower the ability to capture underlying patterns.

'num_neurons': [8, 16, 32] - The number of neurons (units) in each hidden layer of the neural network.

Grid search with the regularization terms, the batch size, and the optimizer was applied to various network structures. Structure-wise, a triangle-like network structure with a larger first hidden layer and a smaller second hidden layer was tested (e.g., Hidden Layer 1: 32 Neuron, Layer 2: 16 Neurons). Also, a diamond shape structure was assessed with three hidden layers, the largest layer being the second hidden layer (e.g., Hidden Layer 1: 8 Neurons, Hidden Layer 2: 16 Neurons, Hidden Layer 3: 8 Neurons). Across the set, the triangle structure outperformed the diamond-structured network. In both structures, a reduction in neurons led to improvements.

As a result, the best-performing structure and hyperparameters of this limited assessment were the following:

{Hidden Layer 1: 8 Neurons, no regularization, Hidden Layer 2: 4 Neurons, L2 Regularization: 0.01, Optimizer: sgd, Batch Size: 32, Epochs: 10}

The model's performance on the training data using these optimized hyperparameters, resulted in the following metrics:

Training MSE: 0.0044346, and training R-squared: 0.0003913

The mean squared error (MSE) value of 0.0044346 indicates the difference between the predicted and actual values. When comparing this to the variance of the response

variable, which is 0.0045725, it can be observed that the model's predictions exhibit a slightly lower variance, indicating better than random or mean-based predictive performance. Additionally, the R-squared value of 0.0003913 indicates that the model captures minimal variability.

Final Benchmark

To benchmark the fine-tuned models, all models were tested on unseen test data (25%, n=488). The following table summarizes and benchmarks all models' test and training performance.

	Linear regression	Lasso	Ridge	Random forest	XGBoost	Neural Network (SGD)
Training MSE	0.00406	0.00444	0.00444	0.00290	0.00425	0.00443
Training R-squared	0.12687	0.04527	0.04517	0.37606	0.02557	0.00039
Testing MSE	**0.00445**	**0.00432**	**0.00432**	**0.00426**	**0.00431**	**0.00511**
Testing R-squared	-0.02354	0.00551	0.00498	0.01847	0.00567	0.00037

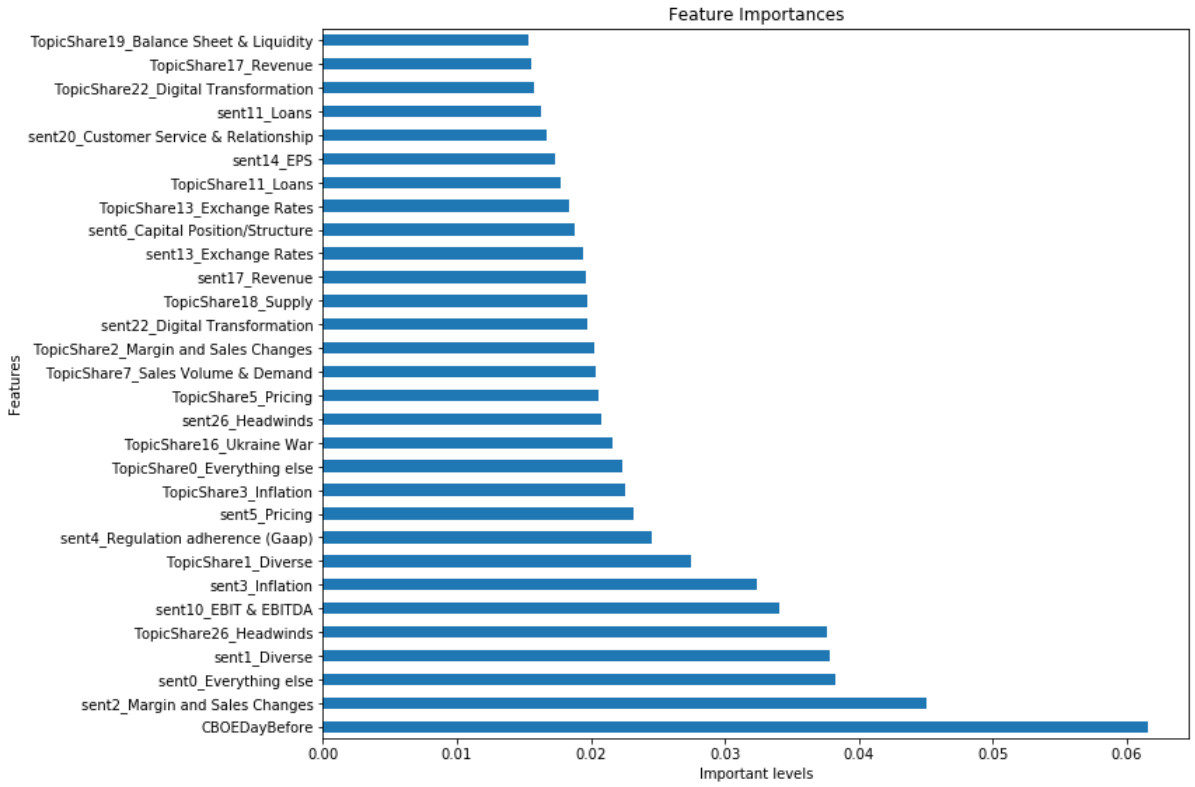
The random forest achieved the best performance (in terms of MSE) on the unseen data. The second-best performance was achieved by the XGBoost model. The ridge and lasso regression did similarly well regarding MSE, followed by the basic linear regression. The neural network achieved the worst performance along all metrics. Except for the Neural Network, all models have a Test MSE below the variance of the response variable (0.004574871)

Random Forest interpretations

Considering that the performance of the random forest model had the best performance on the unseen data, it is essential to delve deeper into the interpretations of the model. In particular, we will focus on examining the different important levels associated with all features, as well as on partial dependence paths of the top 6 features on the target variable and the prediction confidence levels observed in the test set samples.

a. Feature importance

In the context of Random Forest models, conducting feature importance analysis is vital for understanding the predictive power of individual features. This analysis allows us to determine the relative contribution of each feature to the model's predictions. Specifically, feature importances are calculated by measuring the average decrease in impurity across all decision trees within the Random Forest when the values of a particular feature are randomly permuted. We gain valuable insights into each feature's impact on the model's predictions by interpreting these feature importances. Higher feature importances indicate a more substantial influence of a feature on the model, highlighting its significance in predicting the target variable. Conversely, lower importances suggest a relatively lesser impact of the feature. A chart visually represents these importances, displaying the 30 most important features the Random Forest model learned.



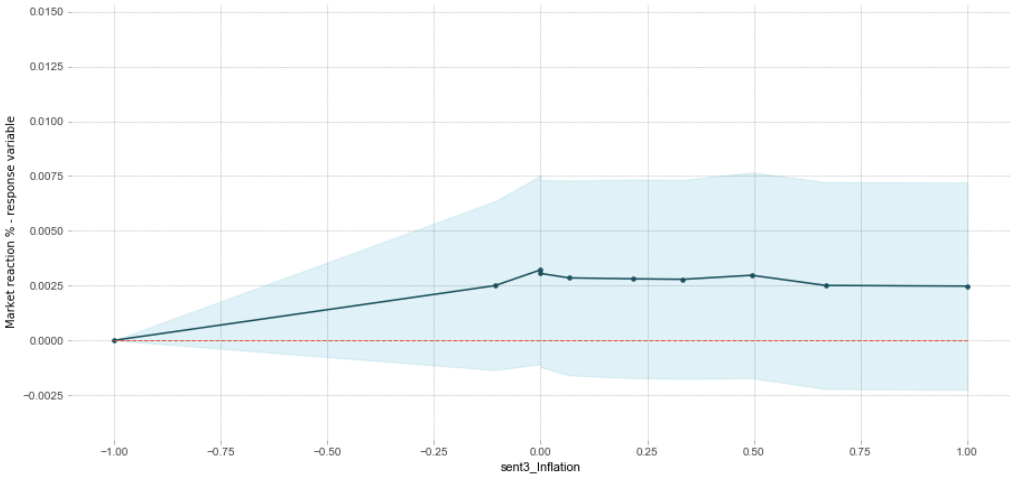
b. Partial dependence

Partial dependence captures the relationship between a specific feature and the target variable while considering the effects of other features. It demonstrates how the predicted target variable changes as the selected feature varies, assuming all other features remain constant.

In a partial dependence plot, the y-axis illustrates the impact of the selected feature on the predicted outcome, taking into account the influence of all other features in the model. It provides insights into the direction of the feature's effect on the response variable. Feature importance offers a relative measure of each feature's contribution, while partial dependence plots facilitate a deeper understanding of the specific direction of influence that each feature has on the response variable. Here are the partial dependence plots for the six most important features (except sent0-outliers):

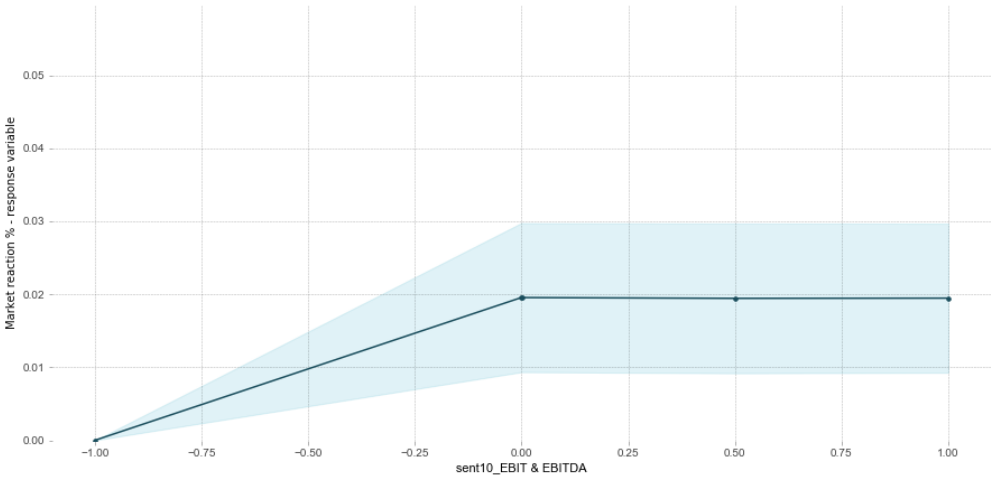
PDP for feature "sent3_Inflation"

Number of unique grid points: 10



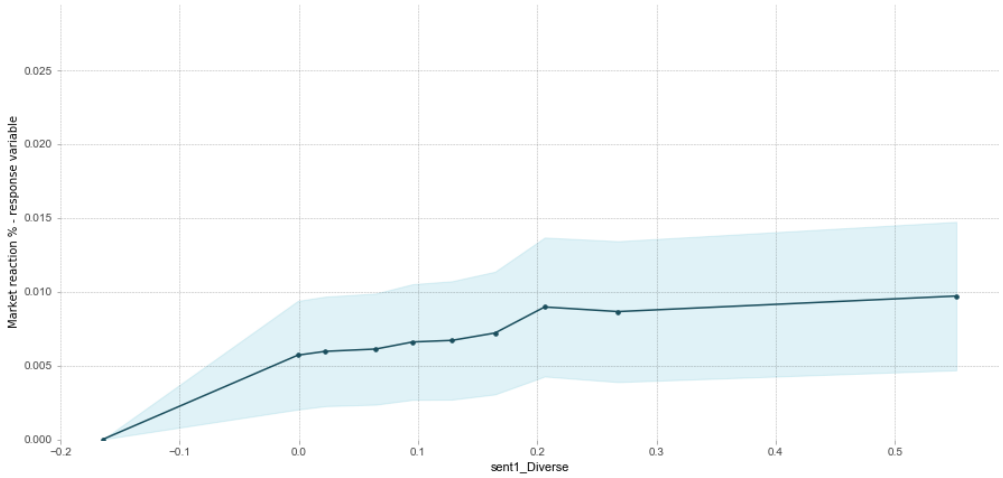
PDP for feature "sent10_EBIT & EBITDA"

Number of unique grid points: 5



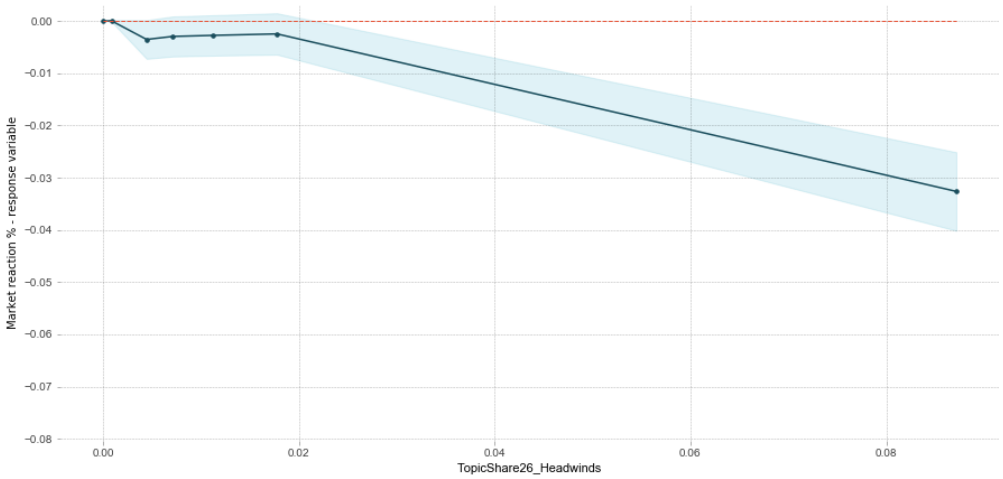
PDP for feature "sent1_Diverse"

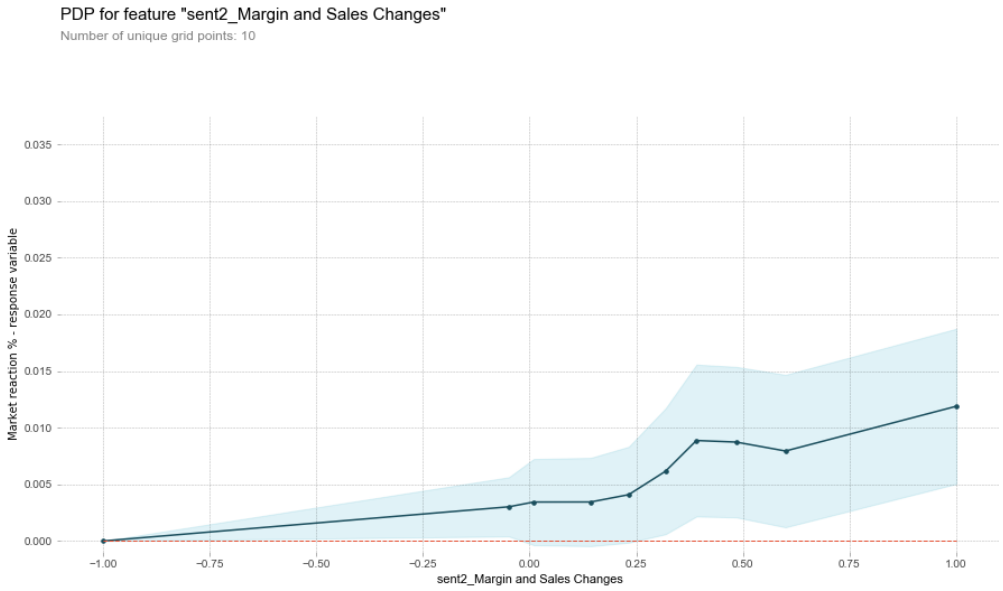
Number of unique grid points: 10

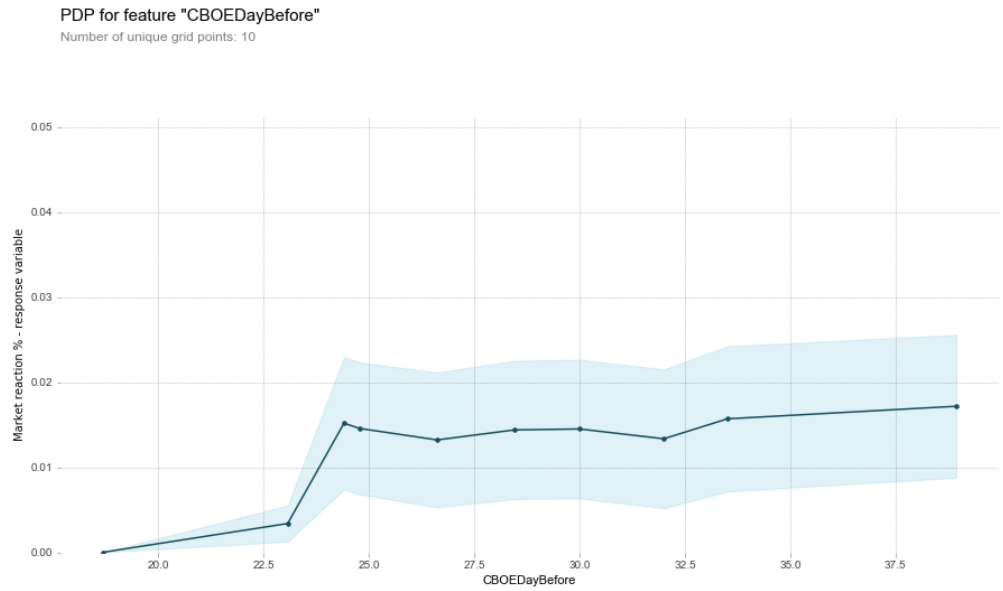


PDP for feature "TopicShare26_Headwinds"

Number of unique grid points: 7





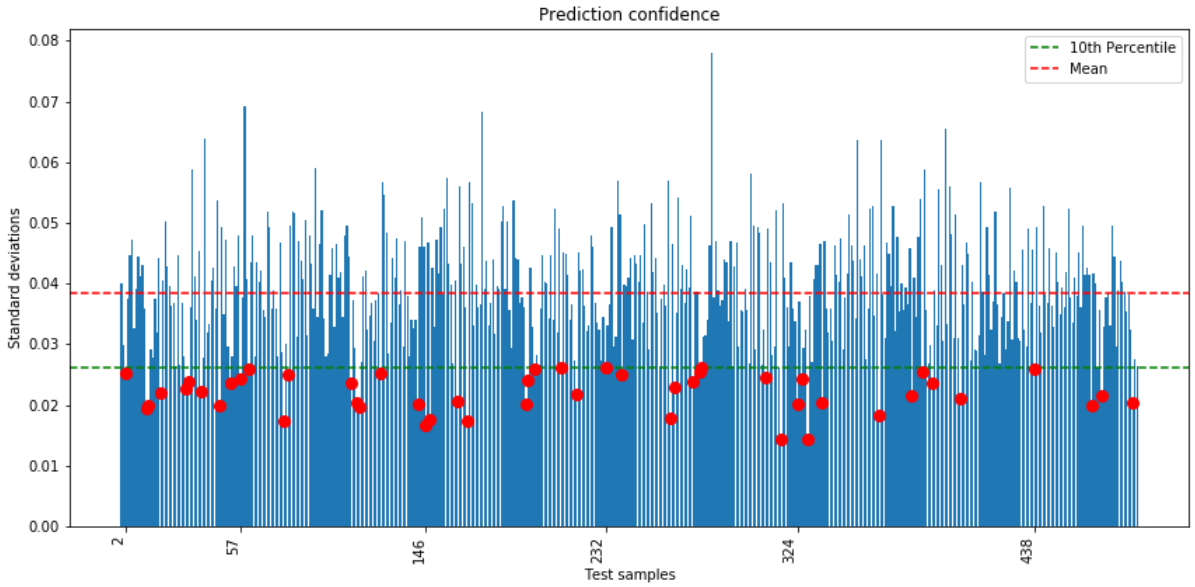


c. Prediction confidence

Analyzing prediction confidence through the standard deviation of test set samples is critical for evaluating the reliability and robustness of a model's predictions. By assessing the standard deviation, we can effectively evaluate the level of uncertainty associated with each prediction. A lower standard deviation reflects more consistent and reliable predictions, instilling higher confidence in the model's accuracy.

Conversely, a higher standard deviation indicates increased variability and uncertainty, leading to diminished confidence in the model's predictions.

The following chart depicts the confidence level, featuring horizontal lines denoting the mean and the highest confidence level at the 10th percentile. The red dots indicate the samples where the confidence level was within the top 10th percentile, indicating a high level of certainty in the model's predictions for these samples.



Discussion

It is remarkable how earnings calls contain valuable information that can aid in evaluating intrinsic value. Our analysis encompassed multiple stages, starting with implementing the BertTopic model to categorize sentences into relevant topic categories. We then performed sentiment analysis on each topic using the Finbert model and applied several machine learners to uncover potential hidden connections between these factors and price changes. Interestingly, throughout our analysis, we did not directly incorporate the two most well-documented approaches in stock evaluation: fundamental analysis and technical analysis. Instead, we explored alternative avenues to extract insights from the earnings calls, leveraging advanced models and methodologies. This highlights the potential of utilizing novel techniques in combination with traditional approaches to gain a more comprehensive understanding of stock evaluation. The linear model worked remarkably well and

underperformed the more complex models like Random Forest or XGBoost only by a small margin. Nevertheless, this indicates that the relationships between the predictors and the response variables are not linear. Further, the linear models also outperform the computationally more complex neural network approach. This approach appears to underperform given the task significantly. However, it shall be noted that only a small set of neural net structures and optimizers were assessed in this task. Hence, this might only serve as an indication but doesn´t allow for any definite conclusions to be drawn.

Reflecting upon the prior literature, understanding the topics further enhances the explanatory and predictive power of the models, compared to a reduction in the total average sentiment. However, it also becomes clear that the predictive power is still meager, which can still be of significant value given the nature of financial markets. Considering the highly statistically significant relationships between Earning Call Sentiment and Stock Market Reaction, one could have also expected a greater predictive performance. It again underlines the differences between evaluating a model or relationship by fitting it to the data or evaluating it based on its performance on an unseen data set.

The following steps could be utilized to further improve upon the model and gain even higher performance. A larger sample could help in making the estimates more robust. Still, it could also enable the use of more topics and, thereby, better granularity, allowing the integration of even more information into the models and reducing the high share of sentences classified as outliers (Topic 0) during this assessment. Another aspect that could further improve the model would be incorporating expert knowledge to evaluate the unsupervised topics and, if suitable, utilize their understanding to create seeded topics (cf. Guided BertTopic Models). Further, this model could be combined with other models that build more financial firm and market-level metrics and changes in these metrics and models that incorporate analyst recommendations or evaluations.

Limitations

One significant drawback of sentiment analysis based on earnings calls is the lack of consideration for stock price movements driven by fundamental analysis at the firm level, typically derived from financial reports. Additionally, the absence of a consensus among analysts regarding their pre-earning call expectations contributes to a lack of agreement on whether the stock price truly reflects the firm's intrinsic value. These factors collectively limit the extent to which insights can be extracted, even with the utilization of complex models.

Furthermore, our analysis focused solely on the year 2022, covering four full quarters, and was limited to the list of the 500 largest companies listed in the US stock market. This restriction inherently limits our ability to fully capture the underlying impact of those topics and their associated sentiments on the overall stock reactions. In addition, the choice of analysis is also unsuitable for identifying differences in magnitude (e.g., A margin increase of 5% or an increase of 20% will be classified as “positive” with the same degree of confidence). It's important to acknowledge that there may also be additional factors and dynamics at play that extend beyond the scope of our analysis, emphasizing the need for further exploration and consideration to gain a more comprehensive understanding.

Attachment A: Topic Contents

Topic Content (subjectively identified common theme)

0	Everything else
1	Diverse
2	Margin and Sales Changes
3	Inflation
4	Regulation adherence (Gaap)
5	Pricing
6	Capital Position/Structure
7	Sales Volume & Demand
8	Organic Growth & Sales
9	Housing/Generations
10	EBIT & EBITDA
11	Loans
12	Healthcare
13	Exchange Rates
14	EPS
15	Quarterly Comparison
16	Ukraine War
17	Revenue
18	Supply
19	Balance Sheet & Liquidity
20	Customer Service & Relationship
21	Sustainability
22	Digital Transformation
23	Fossil Fuels
24	Growth Aspects
25	Demand & Volatility
26	Headwinds

27	Bookings
28	Brand
29	Airline
30	Insurance
31	US Domestic
32	Energy/Fuels
33	Pride
34	Y-o-Y Developments
35	SGA
36	Volume
37	Pipeline
38	Omikron Variant
39	ESG
40	Cloud
41	Electric Technologies
42	Beverages
43	Wireless Infrastructure
44	CapEx
45	M&A
46	Leverage
47	Multiplier x
48	Deposits
49	PreTax Reports
50	Hotels & Gastronomy
51	Highlights
52	NOI
53	Backlog