CalixBoost: A Stock Market Index Predictor using Gradient Boosting Machines Ensemble

Jarrett Yeo Shan Wei and Yeo Chai Kiat

School of Computer Science and Engineering,
Nanyang Technological University, Singapore, Singapore

Abstract

The potential of machine learning has sustained the interest of both academia and industry in stock market prediction over the past decade. This paper aims to integrate modern techniques such as Gradient Boosting Machines (GBMs) into a novel ensemble called CalixBoost which is a resource-efficient and accurate stock index predictor. Data comprising macro-economic metrics and technical financial indicators, as well as sentiment analysis of social media using a simple and fast but effective rule-based model are used in this paper. Other techniques include model tuning with Bayesian Optimization, temporal consistency analysis for invariant feature selection over random trial-and-error, feature importance and inter-feature relationships analysis using a unified game theory approach using Shapley values. Lastly, the model will be evaluated using a novel holdout method, viz. on two separate test datasets whose datapoints are collected under (i) normal economic activity and (ii) during a black swan (financial downturn). The experimental results show that our model outperforms previous methods and can achieve a good prediction performance with 84.88% accuracy, 0.0956 RMSE, 0.0573 MAE and 4.19% MAPE.

Keywords

Gradient Boosting Machines; Time Series Prediction; Game Theory; Ensemble; Bayesian Optimization.

1. Introduction

Using Artificial Intelligence to predict the stock market is hardly a new science – the first use of neural networks in this field dates to ca. 1990 by Kimoto, Asakawa, Yoda, and Takeoka [1]. It has come a long way from 1970 when the Efficient Market Hypothesis (EMH) was first popularized which asserts that market prices always reflect all the information available, and thus, markets are generally efficient [2].

Academia has since, to considerable success, explored otherwise to "beat the market" using a variety of models such as ARIMA [3], ANN [4], SVM [5], LSTM [6] and GBM [7], and also challenged EMH in behavioural psychology studies such as social media sentiment analysis [8].

However, despite the proliferation of successful studies in stock market prediction, research conducted on integrating the application of these novel GBMs, which are also known as Gradient Boosted Regression Trees (GBRTs), as well as other best practices applied outside of research such as temporal consistency analysis in this field are few and far between at the time of writing.

Possible reasons include the perception that such machines are still in their infancy, and the hesitation in using deep learning models since they are sometimes viewed as black boxes whose outputs are difficult to decipher and whose hyperparameters are resource-intensive to tune.

To address these issues, we gather data from Yahoo! Finance for the New York Stock Exchange (NYSE) stock market index price data, U.S. Federal Reserve for macro-economic metrics and Twitter for social media posts from the most influential accounts. Technical financial indicators based on price data were created and sentiment analysis of the Twitter posts was conducted using a rule-based model specifically tuned to analysing social media posts using the VADER library; temporal consistency analysis is finally conducted for invariant feature selection. Next, an ensemble named CalixBoost is assembled using three untuned GBMs – CatBoost, LightGBM and XGBoost – to predict the stock price index. Bayesian Optimization is used to tune the ensemble. The model is evaluated using a holdout method of two separate test datasets whose datapoints are collected under different conditions: first, normal economic activity; and second, during a black swan (financial downturn). The SHAP library based on game theory is used to understand feature importance and inter-feature relationships.

The remainder of this paper is organized into the following sections: Section II reviews related literature, Section III illustrates data collection and pre-processing, Section IV describes the model design and tuning of CalixBoost, Section V shows the experimental results and comparisons with other models, and Section VI concludes this paper.

2. RELATED LITERATURE

Research articles which serve as inspiration for this paper but are not already referenced in later sections or warrant a lengthier discussion are listed below:

2.1. Leveraging social media news to predict stock index movement using RNN-boost

Chen, Yeo, Lau, and Lee [9] proposed a novel hybrid model, RNN-boost, built on top of sentiment analysis of social media posts which can produce good predictions of up to 66.54% relative to other traditional methods. Boosting algorithm Adaboost was used as well. This article is the main inspiration for the project to challenge the status quo by exploring hybrid models and tapping on microblogging data to predict price movements in the stock market.

2.2. On Stock Market Movement Prediction Via Stacking Ensemble Learning Method

Stacked ensemble along with engineered features such as difference can produce good classification results for predicting stocks at 78.10% accuracy [7]. This project will also utilize some of said features as technical indicators.

2.3. Ensemble methods foundations and algorithms

While ensemble is a technique used to assemble weak learning algorithms into an arbitrarily strong learner [10], this project will assemble a meta-classifier and regressor using already-accurate GBMs to achieve a final performance boost.

2.4. Forecasting to Classification: Predicting the direction of stock market price using Xtreme Gradient Boosting

The GBM, XGBoost, proved to be an efficient algorithm with over 87% of accuracy for predicting stock price changes lookback of 60-day and 90-day periods [11]. The inspiration for using the area under curve (AUC) of receiver operating characteristic (ROC) came from this project which was used to evaluate the performance of XGBoost after model training is done. This project will instead use AUC ROC in feature selection first to get rid of features with high false positive rates to create a more resource-efficient model. It would be remiss not to point out that the authors have achieved a classification accuracy of over 99.9% for some stocks which they have raised their concerns of bias during model training. To mitigate such a problem, this project will instead use two datasets for a more robust evaluation of models – one under normal economic conditions (simply labelled as “test” dataset), and the other during a bear stock market (labelled as “black swan”).

2.5. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text

Hutto & Gilbert [12] proposed a rule-based model called VADER for sentiment analysis of social media posts. The authors contend that it is simple yet generalizes more effectively than other state-of-practice methods such as Support Vector Machine (SVM) algorithms, boasting an F1 Classification Accuracy score of 96%. This project will use VADER for sentiment analysis of Tweets for its speed and its attuning to social media posts.

2.6. A Unified Approach to Interpreting Model Predictions

Lundberg and Lee [13] proposed a novel unified game theory framework using Shapley Additive exPlanations (SHAP) values to explain predictions of machine learning models which is widely cited. This project will use SHAP to analyse the feature importance and variable relationships in all models.

3. DATA COLLECTION AND PRE-PROCESSING

3.1. Data Collection

Data used in this paper are summarized as follows:

3.1.1. New York Stock Exchange Index (^NYA)

To offer a fairer basis of comparison, this project will use the same stock index and data as previous projects. ^NYA comprises the major indexes in the United States and is also home to stocks from various industries both local and foreign. It is therefore a suitable weathervane for detecting international market sentiment. The features – Daily Open, High, Low, Close, Adjusted Close prices and Volume – are available in the dataset.

3.1.2. Twitter Posts (Tweets)

Research has shown that microblogging data, especially those from Twitter posts (Tweets), are useful in predicting stock market behaviour [14, 15, 16]. Additional advantages are that Tweets, with their 280-character limit, are relatively easy to analyse than say blog posts or Facebook posts. The Tweets of 25 of the most influential financial, political and news accounts were scrapped. The accounts are: AP, Benzinga, bespokeinvest, BreakoutStocks, Business, CiovaccoCapital, CNBC, FXCM, IBDinvestors, LiveSquawk, LizAnnSonders, MarketCurrents, Markets, MarketWatch, Nyse, nytimesbusiness, Nytimes, realDonaldTrump, ReutersBiz, ReutersUS, Schuldensuehner, Stephanie_Link, Stocktwits, WSJmarkets and YahooFinance.

3.1.3. Macro-economic Indicators

Macro-economic indicators have proven to be useful barometers of stock returns [17], as well as good predictors during economic recession [18, 19]. Therefore, these data are also included in this project to make the models more robust even in a bear stock market. Data were downloaded from the U.S. Federal Reserve and include indicators such as interest rate and bank prime loan rates. The complete list of features is given in Table 1 in which Maturity represents the following: Mi: i-monthly, WKi: i-weekly, Yx: i-yearly, YiP: >i years.

Table 1. U.S. Macro-economic indicators.

S/N	Instrument	Maturity	Frequency
1	FF Federal Funds	Overnight	Daily
2	NFCP Nonfinancial commercial paper	M1, M2, M3	Business Day
3	FCP Financial commercial paper	M1, M2, M3	Business Day
4	PRIME Bank prime loan	N/A	Daily
5	DWPC Discount window primary credit	N/A	Daily
6	TB US government securities / Treasury bills (secondary market)	WK4, M3, M6, Y1	Business day
7	TCMNOM US government securities / Treasury constant maturities / Nominal	M1, M3, M6, Y1, Y2, Y3, Y5, Y7, Y10, Y20, Y30	Business day
8	TCMII US government securities / Treasury constant maturities / Inflation indexed	Y5, Y7, Y10, Y20, Y30	Business day
9	LTAVG US government securities / Inflation-indexed / Long-term average	Y10P	Business day

3.2. Data Pre-Processing

3.2.1. Sentiment Analysis using Valence Aware Dictionary and sEntiment Reasoner (VADER)

Sentiment analysis is conducted by using the five-rule VADER model [12] and implemented as the vader Sentiment library. VADER includes off-the-shelf support for emoticons like :D, emojis like 💘 and 😁, slang words like "sux", initialisms like "lol", negations like "not good", punctuation like "good!!!" and degree modifiers like "kind of good".

VADER is constructed by an optimal list of lexical features specially attuned to understanding sentiment in social media content by generalizing into five rules according to the grammar and syntax of a given text:

i. Lexical features are heavily adapted from dictionaries such as the popular Linguistic Inquiry and Word Count (LIWC) created to understand social media linguistic contexts [20, 21].

ii. Social media-specific lexicon is added to the dictionary such as "LOL" and "WTF".

iii. Using a crowd-sourced approach, human raters indicate both the sentiment polarity (positive or negative) as well as the sentiment valence (how intense of a sentiment a word is) of 9,000 lexical feature candidates.

iv. Lexical features with ratings aggregated across all human raters of ≥ 2.5 standard deviation are removed. Some examples of lexical features with their sentiment polarity and intensity taken from the VADER lexical dictionary [12] are given in Table 2.

Table 2. Examples of Lexical Features with their Sentiment Polarity and Intensity (Hutto & Gilbert, 2014).

Lexical Feature	**Maturity**
“okay”	+0.9
“good”	+1.9
“great”	+3.1
“horrible”	–2.5
“☹”	–1.5
“sucks”	–1.5
“sux”	–1.5

v. General rules are drafted as five sets of heuristics which capture the meaning of texts better by analysing word-order sensitive relationships (cf. bag-of-words) and are given in Table 3 [12].

Table 3. Five Heuristics Rules Used In Vader [12].

S/N	**Rule**	**Example**	**Δ Polarity**	**Δ Valence**
1	!	“Good!”	Same	Stronger
2	CAPS	“GOOD”	Same	Stronger
3	Degree Modifiers	“Very good”	Same	Stronger
		“Marginally good”	Same	Weaker
4	Contrasting Conjunctions	“X is good but Y is bad”	Negative	Stronger
		“X is bad but Y is good”	Positive	Stronger
5	Negation	Positive	Flipped	-

vi. The body of every scrapped Tweet is run through VADER to derive a “composite score”. This is a normalized, weighted composite sentiment score and is interpreted in Table 4.

Table 4. Interpretation of Sentiment from VADER Score.

Sentiment	**Compound Score**
Positive	score ≥ 0.05
Neutral	–0.05 ≤ score < 0.05
Negative	score ≤ –0.05

3.2.2. Technical Price Indicator Engineering

Using the original daily prices and volumes from Yahoo! Finance, many other technical price indicators are derived. The full list of indicators is given in Table 5.

Table 5. Technical Price Features Engineered.

Feature (Symbol)	**Formula**
High (H)	Raw data from Yahoo! Finance
Close (C)	Raw data from Yahoo! Finance
Open (O)	Raw data from Yahoo! Finance
Low (L)	Raw data from Yahoo! Finance
Adj Close (AC)	Raw data from Yahoo! Finance
Volume (V)	Raw data from Yahoo! Finance
Amplitude	$(H_t - L_t) / AC_{t-1}$
Difference	$(C_t - O_t) / AC_{t-1}$
Intraday (I)	$O_{t-1} - AC_{t-1}$
ΔAdj Close (ΔAC)	$AC_t - AC_{t-1}$
ΔVolume (ΔV)	$V_t - V_{t-1}$
Intraday MA_n	*n*-day Moving Average of I
ΔAdj Close MA_n	*n*-day Moving Average of ΔAC
ΔVolume MA_n	*n*-day Moving Average of ΔV
±ΔOpen	1 if $O_t - O_{t-1} > 0$, else -1
Daily Return (DR)	%ΔC
Cumulative Daily Return	Cumulative product of DR
H-L	$H_t - L_t$
C-O	$C_t - O_t$
RSI	14-day period of Relative Strength Index on C
Williams %R	$(H_{max} - C_t) / (H_{max} - L_{min}) * -100$
MA_n	*n*-day Moving Average of C
EMA_n	*n*-day Exponential Moving Average of C
MACD	$EMA_{12} - EMA_{26}$
BB High	21-day C_{avg} + 2 * 21-day C_{std}
BB Low	21-day C_{avg} – 2 * 21-day C_{std}
EMA	Exponential moving average of C with 0.5 decay
Momentum	$C_t - 1$
Feature (Symbol)	Formula
High (H)	Raw data from Yahoo! Finance
Close (C)	Raw data from Yahoo! Finance
Open (O)	Raw data from Yahoo! Finance
Low (L)	Raw data from Yahoo! Finance
Adj Close (AC)	Raw data from Yahoo! Finance
Volume (V)	Raw data from Yahoo! Finance

3.2.3. Lookback Features

This project will utilize time series forecasting for stock market index prediction, where the models will predict output *y* at time *t+1* given in (1).

$$\hat{y}_{t+1} = f(y_{t-k:t}, \boldsymbol{x}_{t-k:t}) \qquad (1)$$

$\hat{y}_{t+1}$ is the model forecast while $y_{t-k:t}$ and $x_{t-k:t}$ are the observations of the output and inputs respectively over lookback window k [22, p. 2]. In this paper, $k = 60$ is used.

3.2.3.1. Feature Scaling: Quantile-based Scaling

Scaling is applied through standardization for all features. While this is not needed in practice for GBMs since they are decision trees which use gradient bosting and not gradient descent [23], scaling is still applied since RNNs like GRU and LSTM require it to achieve faster convergence rates [24].

To deal with outliers in the data, a quantile-based scaler is used to scale all features with the 25th quantile as 0 and the 75th quantile as 1 so that the scaling is not disproportionately influenced by very large outliers.

3.2.3.2. Feature Selection: Temporal Consistency Analysis

This paper conducts a study of time consistency across all features to weed out columns which exhibit poor temporal inconsistency using the Area Under Curve (AUC) of its Receiver Operating Characteristic (ROC) curve.

An ROC curve measures the performance of a binary classification system by plotting the True Positive Rate (TPR) on the y-axis against the False Positive Rate (FPR) on the x-axis. Every data point of the ROC curve indicates an observation in the confusion matrix [11].

Its AUC value is of interest and ranges from 0 to 1, and its results are interpreted as follows [25]:

- AUC = 0: A perfectly inaccurate test
- AUC = 0.5: A test with the results of TPR = FPR, represented by the diagonal line of y = x in the ROC graph, and usually interpreted as the threshold which classification tests should minimally cross
- AUC = [0.7,0.9]: A considerably acceptable / excellent classification test
- AUC > 0.9: An outstanding classification test
- AUC = 1: A perfectly accurate test

The target threshold in this paper is set at AUC=0.5.

As there are only ~20 business days and thus ~20 data points per month per feature, a monthly analysis could not be conducted since comparing data between months will not be statistically significant enough to make a robust assessment.

Therefore, a thorough trimonthly analysis is conducted instead. All data is split into consecutive 3-month periods, then every non-overlapping combination is permutated with all others to determine the AUC score using scikit-learn.

3.2.3.3 Feature Importance Analysis

The Shapley Additive exPlanations (SHAP) framework is used to analyse the feature importance of all models using a unified game theory approach by introducing novel additive feature importance measures which helps to solve the struggle between accuracy and interpretability which has been a problem for deep learning or ensemble models [13, p. 4765]

SHAP introduces a meta-model called an explanation model when predicting a given model [13, p. 4765]. It simplifies the original model by using local methods termed as Additive Feature Attribution Methods (AFAMs) which are conceived to explain the prediction of the original model based on one input feature [13, p. 4766]. AFAMs have one explanation model that is a linear function of binary variables [13, p. 4766].

The individual relationship between inputs of a feature in the original model and the output function is simplified using combined cooperative game theory [13, p. 4768] to derive special Shapley values called SHAP values for every feature when conditioning on a given feature [13, p. 4769]. SHAP values are basically a unique set of simplified inputs which still obey the Shapley value properties of local accuracy, missingness and consistency [13, p. 4768], and thus can be used as a suitable proxy for every given feature [13, p. 4769].

Figure 1 shows a sample SHAP Explanation Model for a single feature. SHAP values are produced which illustrates the change in predicted output when conditioning on said feature. The model explains the change from predicted base value *E[f(z)]* to current output *f(x)* if all other features are unknown. To account for non-linear models or interdependent input features, SHAP averages $\emptyset_i$ values from all permutations.

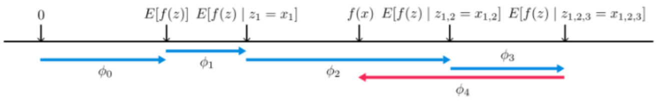

Figure 1. SHAP Explanation Model for 1 Feature [13].

The above can be illustrated using the local accuracy property of AFAM below:

$$f(x) = g(x') = \emptyset_0 + \sum_{i=1}^{M} \emptyset_i x'_i \qquad (2)$$

For a given input *x*, explanation model *g(x')* = original model *f(x)* when *x* = $h_x(x')$ and where $\emptyset_0$ = *E[f(z)]* which is the model output when we have no non-zero input entries [13, p. 4768].

Thus, with every non-zero input entry z_i, a unique Shapley value $\emptyset_i$ (SHAP value) is produced as an approximation in a conditional expectation function of the original model [13, p. 4769].

Since Figure 1 is a single explanation model of a single feature, when one iteration of that of all features are stacked together, we get Figure 2.

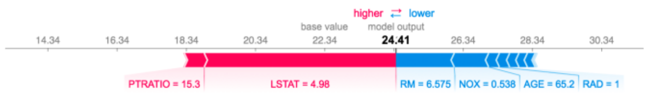

Figure 2. SHAP Explanation Model for 1 Iteration of All Features [13].

The final Explanation Model for N Iteration of all input features is simply N iterations of Figure 2's stacked on top of one another. The final explanation model is shown in Figure 3. The terms on the left of the graph are the input variables, and the *x*-axis represents the feature value. The importance of each feature is thus finally derived here.

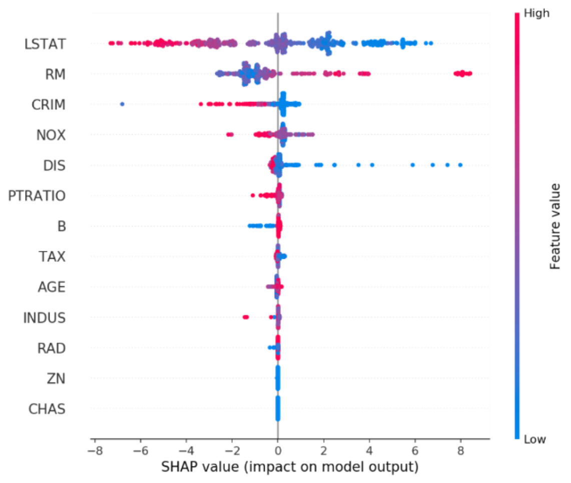

Figure 3. SHAP Explanation Model for All Features [13].

This paper uses the Python implementation "shap" written by the same authors of SHAP.

4. MODEL DESIGN AND TUNING

4.1. Model Design

This section introduces decision trees, ensembles, the three GBMs – CatBoost, LightGBM and XGBoost – which are decision tree ensembles, and our novel CalixBoost which is a stacked ensemble on the GBMs. Default hyperparameters were used in the GBMs.

4.1.1. Decision Tree

Decision Trees, also known as Classification Trees, are trees in which every node is labelled with an input variable, and all arcs stemming from a node represent the possible values of said input. When a set of inputs is given, the tree is traversed until it terminates at a leaf which gives the final class of this observation [26, p. 298]. The objective is to create a decision tree which gives the highest proportion of correct predictions, viz. a tree which minimizes the error of actual results vis-à-vis predicted results. An example of a decision tree is given in Figure 4.

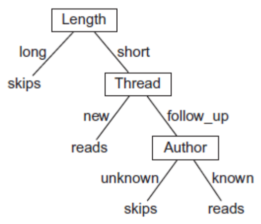

Figure 4. Example of a decision tree [26, p. 298].

4.1.2. Ensemble

Ensembles are a type of supervised learning composite model. Ensemble learning combines weak learning models into an arbitrarily stronger meta-model [26, p. 319]. The two common methods of creating ensembles are bagging and boosting. In bagging, m instances of training data will be bagged into m random sets with replacement [26, p. 319]. On the other hand, in boosting, classifiers which produce wrong results subsequently have a higher probability of being chosen as the next set of training examples [26, p. 320].

4.1.3. XGBoost

XGBoost is an open-source tree boosting machine learning system released in 2014 [27] inspired by the GBRT which was used in the solution that clinched the Netflix Prize in 2009. Since its inception, XGBoost has experienced explosive adoption, having been used by most winning solutions on Kaggle's data science competitions since 2015.

While the concept of GBRT is not new, XGBoost popularized it through its main advantage of scalability and memory efficiency through algorithmic optimizations such as a novel tree learning algorithm for handling sparse data, a weighted quantile sketch procedure which handles instance weights in approximate tree learning, and parallel computing to accelerate model exploration [27, p. 1].

4.1.4. LightGBM

LightGBM is a GBRT released in 2017 by Ke et al. [28] which is attuned for efficiency and scalability when in high feature dimensions and huge datasets. It is reported to quicken training by up to 20 times with similar accuracy vis-à-vis other conventional GBMs. Its authors assert that its innovations over XGBoost and other GBMs are that a small dataset can be used since data samples with larger gradients are used to compute information gain approximately, and that mutually exclusive features are combined into a single feature to reduce the dimensionality of data [28, p. 1].

4.1.5. CatBoost

Prokhorenkova et al. introduced CatBoost in 2019 to resolve the prediction shift problem caused by target leakage present in older GBMs [29]. It proposes permutation-based boosting over classical boosting and guesses categorical features efficiently by using an ordered version of "group(ing) categories by target statistics that estimate expected target value in each category" [29, p. 2].

4.1.6. CalixBoost

CalixBoost Ensemble is a novel Grid Search-based Weighted Average Ensemble proposed in this paper. It is a stacked ensemble of XGBoost, LightGBM and CatBoost. The program first trains the XGBoost, LightGBM and CatBoost models, stores their predictions in memory, then iterates through a pre-set list of weight combinations and calculates the weighted sum of the predictions to get the final ensemble prediction.

The best ensemble used the weights of 0.2:0.1:0.7 for XGBoost:LightGBM:CatBoost.

While Grid Search is a brute force method, the calculation of the ensemble prediction is computationally cheap, thus it is used.

4.1. Model Tuning: Bayesian Optimization

Bayesian Optimization is an automatic approach for maximizing the performance of machine learning algorithms. By generalizing an algorithm's performance using a Gaussian process, users can avoid trial-and-error tuning which is vastly inefficient. Bayesian Optimization can be on par with or exceed optimizations tuned by human experts and other algorithms like Latent Dirichlet Allocation (LDA) and Convolutional Neural Networks (CNNs) [30]. The library used in this paper for tuning hyperparameters (cf. brute-force methods such as Grid Search) is bayesian-optimization.

Bayesian Optimization uses a "Bayesian technique of setting a prior over the objective function and combining it with evidence to get a posterior function" [31]. A 1D Gaussian process estimation of the target function over nine observations is illustrated in Figure 5, along with its ~95% confidence interval shaded area (posterior uncertainty) representing the mean μ ± the standard deviation σ. The solid line represents the target function, while the dotted line shows the prediction.

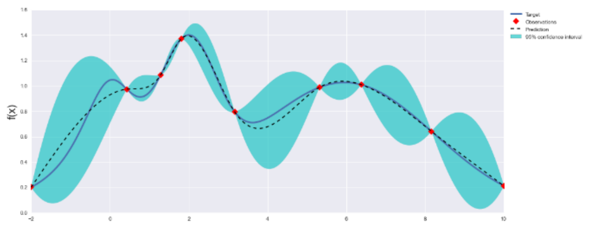

Figure 5. Gaussian Process after 9 Observations [32].

With more iterations, the *posterior uncertainty* decreases. Figure 6 shows the next best space to explore the point maximizing the utility function after said nine observations in Figure 5.

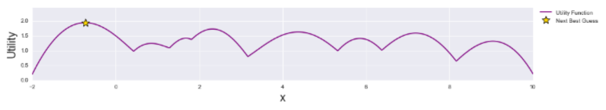

Figure 6. Utility Function after 9 Observations [32].

The algorithm strives to strike a balance between exploration and exploitation using the *Upper Confidence Bound* (UCB)-maximizing strategy used by the *Sequential Design for Optimization* (SDO) algorithm which suggests the best points for evaluation proposed by Cox and John [33]. The formula for UCB is given below:

$$UCB(x) = \mu(x) + \kappa \cdot \sigma(x) \tag{3}$$

κ is the kappa which is used to tune the parameterized acquisition model. In this project, κ = 2.576 is used which is the z-score when confidence level = 99%.

Figure 7 shows the end of Bayesian Optimization after ~80 observations and we observe that the algorithm has indeed balanced the exploration of the entire search space with the exploitation of higher target value areas (marked in dark red) in which more observations are sampled per unit search space.

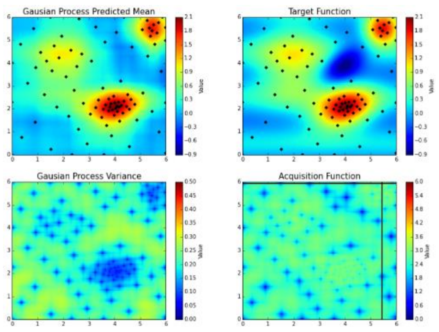

Figure 7. Gaussian Process Predicted Mean, Target Function, Gaussian Process Variance, and Acquisition Function after ~80 observations [32].

5. Results and Comparisons

5.1. Cross Validation

This paper proposes a novel holdout method which splits data into the following partitions:

- Training dataset ("train"): ~85%
- Validation dataset ("validation"): ~5%
- Test dataset #1 ("test"): ~5%
- Test dataset #2 ("blackswan"): ~5%

The term "black swan" stems from the black swan theory by Taleb in 2007. A black swan event is an outlier causing disastrous consequences.

Two test datasets allow for more rigorous evaluation among models: first, the "test" dataset comprises data points collected under normal economic circumstances in the U.S. economy ca. Dec 2019 to Mar 2020, and second, the "blackswan" dataset consists of data points collected under bear market conditions from Mar 2020 to May 2020 [34] during which the U.S. stock market crashed late Feb 2020 due to economic recession from both the COVID19 outbreak and a huge slump in oil prices due to glut.

With this special holdout method, we can evaluate whether each model can not only predict results accurately under ordinary financial situations, but also assess if it is robust enough to generalize well to extraordinary economic circumstances such as COVID-19.

5.2. Model Evaluation

This project evaluates all models based on Root Mean Squared Error (RMSE), Accuracy (Acc), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE). Their formulae are given below where O_t and $\hat{O}_{t+i}$ represent the actual and predicted price of Opening at time t.

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(O_{t+i} - \hat{O}_{t+i})^2} \tag{4}$$

$$Accuracy = \begin{cases} 1, \ if \ |\hat{O}_{t+i} - O_t| = |O_{t+i} - O_t| \\ 0, \qquad\qquad otherwise \end{cases} \tag{5}$$

$$MAE = \frac{1}{N}\sum_{t=1}^{N}\left|O_{t+i} - \hat{O}_{t+i}\right| \tag{6}$$

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\frac{|o_{t+i}-\hat{o}_{t+i}|}{|o_{t+i}|} \qquad (7)$$

For experimental repeatability, the libraries “numpy” and “random” are seeded with integers from 0 to 10, i.e., all results are aggregated over ten trials.

5.3. Experimental Results

The results of the “test” dataset (predicting under normal economic circumstances) are given in Table 6. The best performant *CalixBoost* model is compared against the best results of *RNN-GRU* [8] and *RNNBoost* [9] which were proposed in similar studies on stock market index prediction using social media analytics. The mean performance of *XGBoost*, *LightGBM*, *CatBoost* are additionally provided for comparison purposes. CalixBoost has the highest accuracy and lowest errors of all models.

Table 6. Test Dataset Results.

Model	**Accuracy**	**RMSE**	**MAE**	**MAPE**
RNN-GRU	-	0.8031	0.6254	9.38%
RNNBoost	66.54%	2.0500	1.3200	22.31%
XGBoost	79.19%	0.1052	0.0658	4.87%
LightGBM	80.23%	0.1382	0.0899	6.58%
CatBoost	82.67%	0.1992	0.1630	12.64%
CalixBoost	84.88%	0.0956	0.0573	4.19%

The results for the “blackswan” dataset where the stock market index is predicted in bear market conditions are provided in Table 7 for XGBoost, LightGBM, CatBoost and CalixBoost Ensemble. Models from other studies did not carry out such a test, thus there are no results to show for those. CalixBoost is the most performant; it generalizes well during black swan events such as economic recessions. In fact, CalixBoost’s performance for this dataset is still more performant than RNN-GRU and RNNBoost evaluated under normal economic conditions as seen in Table 6.

Table 7. Blackswan Dataset Results.

Model	**Accuracy**	**RMSE**	**MAE**
XGBoost	75.00%	0.2107	0.1709
LightGBM	73.26%	0.2523	0.2047
CatBoost	73.72%	0.4545	0.3861
CalixBoost	77.91%	0.2002	0.1617

While CalixBoost performs better than other GBMs and models in all areas, we contend that the improvement in performance of 2.2% maximum accuracy is significant but might not be substantial enough to justify training and assembling three GBMs into the CalixBoost ensemble. The authors thus recommend that unless achieving the best performance is of utmost necessity, any of GBMs can be used since they exhibit similar performance and it will be computationally considerably cheaper to train just one model.

5.4. Feature Importance

By using SHAP (see 3.2.3.3 Feature Importance Analysis), the most important features of the GBM models can be analyzed. A sample SHAP graph for XGBoost is given in Figure 8. The results indicate that the features "C-O" and "Difference" are 60%, 80% and 95% more important than the next most important features when maximizing AUC of XGBoost, LightGBM and CatBoost respectively. Both are derived from $Close_t$ - $Open_t$, which suggests that the magnitude and direction of the difference in today's Close and Open prices exercise a huge influence over predicting tomorrow's $Adj\ Close_{t+1}$ price, and intuition suggests the most likely reason is that tomorrow's price is unlikely to deviate vastly from that of today.

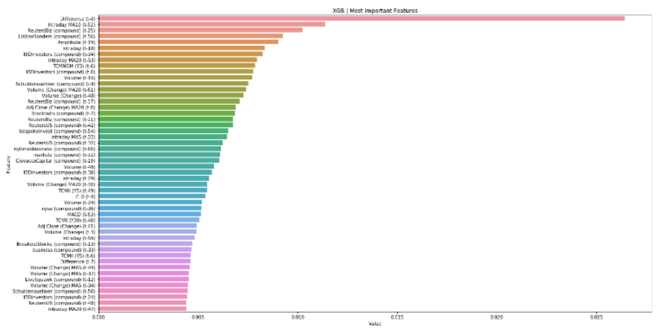

Figure 8. SHAP Feature Importance Graph for XGBoost.

6. CONCLUSION

GBMs can now beat RNNs by a considerable margin in stock market index prediction. The proposed novel GBM ensemble CalixBoost gives an even higher accuracy. Results were evaluated on a novel holdout method of both "test" and "blackswan" to test if models could perform in normal and bear economic situations like COVID-19. CalixBoost is shown to be able to generalize well, predicting the index with 85% and 78% accuracy for both.

In addition, this paper also explored modern approaches so that the workflow can be more accurate and resource efficient. The pipeline processes include using techniques like technical feature engineering, Temporal Consistency Analysis and Bayesian Optimization for more scientific and effective model tuning, using data from both social media sentiment and macro-economic indicators to get the best results, and using game theory to explain feature importance of GBMs – which were in the past avoided for being black boxes – for a better understanding of the relationship between variables and target.

Even though the project follows a methodology of being efficient yet effective, there remain opportunities for further optimization, such as using simpler models like logistic regression or linear regression which are computationally a lot cheaper to train and exploring simpler datasets such as using just technical features derived from the stock market price since they are the most important features by a large margin vis-à-vis social media sentiment and macro-economic indicators. Preliminary studies conducted by the authors using linear regression have already shown to be promising and we wish to continue the research.

ACKNOWLEDGEMENTS

The authors would like to thank Nanyang Technological University for the opportunity to conduct this research.

REFERENCES

[1] Kimoto, T., Asakawa, K., Yoda, M., & Takeoka, M. (1990, June). Stock market prediction system with modular neural networks. In *1990 IJCNN international joint conference on neural networks* (pp. 1-6). IEEE.

[2] Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, *25*(2), 383-417.

[3] Ariyo, A. A., Adewumi, A. O., & Ayo, C. K. (2014, March). Stock price prediction using the ARIMA model. In *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation* (pp. 106-112). IEEE.

[4] Vui, C. S., Soon, G. K., On, C. K., Alfred, R., & Anthony, P. (2013, November). A review of stock market prediction with Artificial neural network (ANN). In *2013 IEEE international conference on control system, computing and engineering* (pp. 477-482). IEEE.

[5] Yang, H., Chan, L., & King, I. (2002, August). Support vector machine regression for volatile stock market prediction. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 391-396). Springer, Berlin, Heidelberg.

[6] Chen, K., Zhou, Y., & Dai, F. (2015, October). A LSTM-based method for stock returns prediction: A case study of China stock market. In *2015 IEEE international conference on big data (big data)* (pp. 2823-2824). IEEE.

[7] Gyamerah, S. A., Ngare, P., & Ikpe, D. (2019, May). On Stock Market Movement Prediction Via Stacking Ensemble Learning Method. In *2019 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)* (pp. 1-8). IEEE.

[8] Chen, W., Zhang, Y., Yeo, C. K., Lau, C. T., & Lee, B. S. (2017). *Stock market prediction using neural network through news on online social networks. 2017 International Smart Cities Conference (ISC2).* doi:10.1109/isc2.2017.8090834

[9] Chen, W., Yeo, C. K., Lau, C. T., & Lee, B. S. (2018). *Leveraging social media news to predict stock index movement using RNN-boost. Data & Knowledge Engineering.* doi:10.1016/j.datak.2018.08.003

[10] Zhou, Z. H. (2012). *Ensemble methods: foundations and algorithms*. CRC press.

[11] Dey, S., Kumar, Y., Saha, S., & Basak, S. (2016). Forecasting to Classification: Predicting the direction of stock market price using Xtreme Gradient Boosting. *PESIT, Bengaluru, India, Working Paper*.

[12] Hutto, C. J., & Gilbert, E. (2014, May). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*.

[13] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems* (pp. 4765-4774).

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

[15] Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of computational science*, *2*(1), 1-8.

[16] Mittal, A., & Goel, A. (2012). Stock prediction using twitter sentiment analysis. *Standford University, CS229 (2011 http://cs229. stanford. edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis. pdf)*, *15*.

[17] Narayan, P. K., Narayan, S., & Thuraisamy, K. S. (2014). Can institutions and macroeconomic factors predict stock returns in emerging markets?. *Emerging Markets Review*, *19*, 77-95.

[18] Estrella, A., & Mishkin, F. S. (1998). Predicting US recessions: Financial variables as leading indicators. *Review of Economics and Statistics*, *80*(1), 45-61.

[19] Chen, S. S. (2009). Predicting the bear stock market: Macroeconomic variables as leading indicators. *Journal of Banking & Finance*, *33*(2), 211-223.

[20] Pennebaker, J. W., Francis, M. E., & Booth, R. J. (2001). Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates*, *71*(2001), 2001.

[21] Pennebaker, J. W., Boyd, R. L., Jordan, K., & Blackburn, K. (2015). *The development and psychometric properties of LIWC2015*.

[22] Lim, B., & Zohren, S. (2020). Time series forecasting with deep learning: A survey. *arXiv preprint arXiv:2004.13408*.

[23] Chen, T. (2015). *Is Normalization necessary? · Issue #357 · dmlc/xgboost*. Retrieved from https://github.com/dmlc/xgboost/issues/357.

[24] Laurent, C., Pereyra, G., Brakel, P., Zhang, Y., & Bengio, Y. (2016, March). Batch normalized recurrent neural networks. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 2657-2661). IEEE.

[25] Mandrekar, J. N. (2010). Receiver operating characteristic curve in diagnostic test assessment. *Journal of Thoracic Oncology*, *5*(9), 1315-1316.

[26] Poole, D. L., & Mackworth, A. K. (2010). *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press. (ISBN-13: 978-0-511-72946-1; ISBN-13 978-0-521-51900-7)

[27] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).

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

[29] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. In *Advances in neural information processing systems* (pp. 6638-6648).

[30] Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems* (pp. 2951-2959).

[31] Brochu, E., Cora, V. M., & De Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599.*

[32] Nogueira, F. (2014). *Bayesian Optimization: Open source constrained global optimization tool for Python.* Retrieved from https://github.com/fmfn/BayesianOptimization.

[33] Cox, D. D., & John, S. (1992, October). A statistical method for global optimization. In *[Proceedings] 1992 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 1241-1246). IEEE.

[34] Wearden, G., Jolly, J., Makortoff, K., Brignall, M., Ambrose, J., Kollewe, J., & Sweney, M. (2020, March 12). Wall Street and FTSE 100 plunge on worst day since 1987 – as it happened. Retrieved June 28, 2020, from https://www.theguardian.com/business/live/2020/mar/12/stock-markets-tumble-trump-europe-travel-ban-ecb-christine-lagarde-business-live

AUTHORS

Jarrett Yeo Shan Wei is a final year college student at Nanyang Technological University, Singapore.

Yeo Chai Kiat is an Assoc. Prof. at Nanyang Technological University, Singapore.