Manuscript Details

Manuscript number	NAJEF_2017_307_R1
Title	From Forecasting to Classification: Predicting the Direction of Stock Market Price Using Tree-Based Classifiers
Article type	Full Length Article

Abstract

Predicting trends in stock market prices has been an area of interest for researchers for many years due to its complex and dynamic nature. Intrinsic volatility in the stock market across the globe makes the task of prediction challenging. Forecasting and diffusion modeling, though effective, cannot be the panacea for the diverse range of problems encountered in predicting trends in the stock market, short-term or otherwise. Market risk, strongly correlated with forecasting errors, needs to be minimized to ensure minimal risk in investment. This paper is the outcome of experiments with a completely different approach: instead of defining this problem as a traditional forecasting-style problem, we just try to predict whether prices will increase or decrease. The problem is posed as a classification problem, where the class labels may be +-1, indicating an increase or a decrease in the price of a stock with respect to n days back. For this purpose, the potential of Random Forests and XGBoosted trees is explored. Random Forests use an ensemble of Decision Trees to improve the accuracy of classification. XGBoost is an engineering solution which aims to speed up the process of growing Gradient Boosted Decision Trees (GBDT). Technical indicators such as Relative Strength Index (RSI), Stochastic Oscillator, etc. are used as features to train the model. The algorithms are shown to outperform the algorithms used in the existing literature.

Keywords	stock price prediction; xgboost; random forests; classification; binary options; machine learning
Taxonomy	Finance, Economics

Corresponding Author	Snehanshu Saha
Corresponding Author's Institution	PESIT Bangalore South Campus
Order of Authors	Suryoday Basak, Snehanshu Saha, Saibal Kar, Luckyson Khaidem, Sudeepa Dey

Submission Files Included in this PDF

File Name [File Type]

cover-letter_NAJEF.pdf [Cover Letter]

reply-reviewer1.pdf [Response to Reviewers (without Author Details)]

reply-reviewer2.pdf [Response to Reviewers (without Author Details)]

author_predicting-direction-stock.1.pdf [Title Page (with Author Details)]

predicting-direction-stock.pdf [Manuscript (without Author Details)]

SupplementaryFile_predicting-direction-stock.pdf [Supporting File]

Highlights.docx [Supporting File]

Submission Files Not Included in this PDF

File Name [File Type]

predicting-direction-stock.zip [LaTeX Source File]

To view all the submission files, including those not included in the PDF, click on the manuscript title on your EVISE Homepage, then click 'Download zip file'.

To
The Editorial Board
The North American Journal of Economics and Finance

SUB: Regarding Submission of *Predicting the Direction of Stock Market Prices Using Tree-Based Classifiers* (manuscript title amended as suggested)

Dear Editorial Board members,

We submit the revised paper for consideration for publication in The North American Journal of Economics and Finance. It gives us immense pleasure to submit our original work to your esteemed journal. We believe the approach and applications brought froth in the paper have significant outcome in a field that requires serious scientific intervention. The manuscript, if published, will create new theory and novel applications in a direction hitherto unexplored. We hope that our contribution will be given full consideration toward publication.

A few highlights of the paper are:

1. Application of Machine learning in diverse portfolio of stocks
2. In-depth contextual analysis of prediction and classification algorithms with mathematical details
3. Reformulation of a traditional forecasting problem as a classification problem
4. Machine classification models developed
5. Recursive error minimization by parameter tuning
6. Significant material added strengthening the claims in the original manuscript.

We must thank the reviewers for insightful comments. We enjoyed the interaction as evident from the uploaded response files. We made every effort to address all concerns and we must admit the suggestions helped structure the manuscript in a way suitable for the journal. It is important for us to point out that this has been an ongoing and an incremental research effort. In the process, we've had multiple phases and we have tried various methods. There are precursors to the current manuscript available in arXiv and ResearchGate, but the authors would like to ensure that this version is an improved version over all its precursors and much more expansive in its breadth as well as its depth. That said, the manuscript that we have submitted to you is not under consideration for publication anywhere else.

The links to previous unpublished works are:

1. https://arxiv.org/abs/1605.00003
2. http://www.researchgate.net/publication/309492895_Forecasting_to_Classification_Predicting_the_direction_of_stock_market_price_using_Xtreme_Gradient_Boosting
3. http://www.researchgate.net/publication/315542322_Exposition_on_Random_Forest_A_stock_example

Sincerely,
Authors of *Predicting the Direction of Stock Market Prices Using Tree-Based Classifiers*

Reply to the Reviewer's (Reviewer 1) Comments on *Predicting the Direction of Stock Market Price Using Tree-Based Classifiers*

June 16, 2018

Before we begin addressing the specific points and nuances raised by the reviewer, we thank the reviewer for very helpful comments and explain the main purpose of the paper.

The aim of the current work is to predict the direction that the price of a stock of a certain company will move towards. This is effectively a gain/loss scenario where we do not try to understand 'how much we have gained' or 'how much we have lost', rather only 'if we have gained anything' or 'if we have lost anything'. This is a deviation from the popular idea of predicting the monetary returns of a stock after a certain time period. This is the problem statement and this is what we have tried to address by using methods in machine learning. Admittedly, we had previously used the word 'returns' in an ambiguous sense in the manuscript and have clarified it in the revised submission. However, with this point clarified, let us address the general comment.

Commment: "The proportion of positive and negative data are in range of 45:55". Here you mean trading days with positive returns? Averaged across all your stocks (in all markets)? This is potentially misleading. You should include the median returns over various n-day periods in your (testing) dataset. If stock prices are stochastic but have positive drift, wouldn't sign predictability get easier with increasing forecast horizon?"

Response: We have rephrased the sentence to avoid any confusion. However, the purpose here is to generalize and develop a method to predict with reasonable efficiency whether the price of a stock will increase or decrease on day $n+t$ as compared to day n. It seems that the mean accuracy is a more representative measure of the efficacy of the system as compared to the median returns. Certainly, in case of a positive drift in the prices of a stock it may be possible to have a better sign predictability, but that does not seem to have a conflict with our method of prediction.

In addition, it appears from the data that most stock prices do not exhibit a strictly negative or a strictly positive drift. Therefore, the purpose of selecting stocks of companies like Nike, Toyota, Facebook, Amazon, Apple etc. is to demonstrate that regardless of the background or domain of a company, regardless of specific fluctuations in the prices over time, the efficacy of our methods hold and these do not succumb to diminishing accuracies.

We now proceed to respond to other specific comments by the reviewers and how we have addressed them in the revised submission.

Summary:

This paper investigates the ability of two machine learning (ML) techniques (Random Forests and Gradient Boosted Trees) to forecast future stock prices. Specifically, forecasting is reformulated as a classification problem; essentially is the stock price expected to go up or down over a given time period. Model inputs are based on signals drawn from technical analysis, to which smoothing is applied by the authors. The ML techniques are presented in some detail with examples. The data set comprises 14 (mostly US) stocks with forecasts made over 3-90 days. Results are presented in terms of statistics such as accuracy, recall, etc. The ML models are held to outperform algorithms used in existing literature. Exploring how the wide range of machine learning techniques can be applied in a financial/economic domain, and how they can be used to complement traditional models, is an interesting area.

Main Comments:

Comment: The abstract should be rewritten to highlight novelty and the contributions of the paper.

Response: We have now re-written the abstract highlighting the novel points in a manner relevant to NAJEF.

Comment: While the discussion of the EM hypothesis is well placed, anomalies could also be discussed. See suggested references below.

Response: We have included the anomalies as obtained from the suggested references as part of the introduction (Pages 1 and 2). Thank you again.

Comment: For me, too much of the paper is devoted to outlining the ML techniques. These are well established and fully documented elsewhere. Readers unfamiliar with the techniques should be directed to primary sources, or authoritative texts. Provide a brief description, giving an intuitive sense of how the ML techniques work. Focus on the benefits such models promise in relation to traditional approaches and to the problem at hand.

Response: We have followed this suggestion carefully. The main sources where the techniques are explored and emphasized are cited in the main text along with a short introduction to the analytical and intuitive aspects of the models developed. While aspects of the algorithms and methods can be found through other sources, our intention amidst all the work was to break the usual treatment of ML as *black-boxes*. The example that we elaborately explained is based on real data samples from the data that we used. We broke down the methods by means of examples to maintain complete transparency of the work that we have done. Upon the reviewer's recommendation, we have moved some of the writing on the theoretical aspects of the algorithms, and the example, to a supplementary document (Sections 2, 3, 4 and 5 in the supplementary material). We have retained the material as a supplementary document as we think that it is highly relevant in the context of the paper.

Comment: While perhaps common in 'Information System' journals, my recommendation is to remove the outline algorithms and examples (e.g. Tables 2, 3, 4, etc). While one figure may be

useful to aid understanding of a tree, the extended example and corresponding figures should be removed (or at least relegated to a separate appendix).

To appear in the finance/economics journal, the paper should be targeted to such an audience. Figure 2 is too basic to merit inclusion. Similarly figure 1 adds little value and could be easily summarised in a few lines.

Derivation of established formula/results (e.g. Chebyshev's Inequality) and basic definitions (e.g. an indicator variable) need not be included and certainly not proved. We have moved the material in these sections to a supplementary document (Sections 3-5 in the supplementary document).

Similarly, it is unnecessary to provide formula or examples for well-established technical indicators [incidentally, why include day 1 in the RSI formula; 15 prices are required for 14 daily returns]. Rather, focus on why these indicators are considered to have predictive ability. See also comments below.

Response: We have shortened the overall presentation of the theorems to aid better readability for economics/finance journals. The appropriate references are cited and we have deleted the discussion and proof of theorems from the main paper, as suggested. However, we have moved them to the supplementary reading material (Sections 2 and 3 in the supplementary file).

Comment: The paper does not adequately discuss feature selection or reference supporting literature to justify the choice of input attributes believed to have predictive ability. What do the models under investigation tell us about variable importance? Do certain technical indicators prove more useful for prediction? If these models really are outperforming existing benchmarks, from a finance perspective we want to understand why.

Response: The following is added to the main text to address this comment.

In recent years, the stock market analysis and prediction have been studied with the aid of methods such as machine learning and text mining. Data mining studies use daily stock data. For example, prediction studies based on support vector machines (SVMs) (Cao and Tay, 2001; Ince and Trafalis, 2007, Atsalakis and Valavanis, 2009, etc.) have been conducted to determine pattern categories. In addition, artificial neural networks (ANNs) (Kimoto et al., 1990; Kohara et al., 1997) have been employed to achieve good predictions even in the case of complex relationships of variables. Typically, autoregressive integrated moving average (ARIMA) model (Pai and Lin, 2005; Wang and Leu, 1996, Moscowitz et al. 2012) are used for identifying and predicting time series variation. Notwithstanding, since behavior and individualized responses play a significant role in dictating the stock turnovers and prices,, a few studies have engaged with word analysis of news articles (Mittermayer, 2004; Nikfarjam et al., 2010; Nyberg, 2011 for the US; Kim et al., 2014, etc.) and its predictive ability. However, most of these studies have some limitations for short-term prediction. First, without filtering for outliers, the predictions based on all historical data leads to potential errors. Second, although the total completion price is determined by a variety of factors such as the foreign purchase closing price and domestic selling completion amount, this set needs to be expanded in order to reduce omitted variables bias. Variables of importance may include, categories of financial ratios, macro, labour market and housing variables and measures of sentiment and leverage (Black et al. 2014; Cochrane, 2008, etc). With respect

to the current paper, it is important to note that the main purpose is to implement two distinct methods on stock data and highlight their advantage over other non-ensemble techniques within the machine learning approaches for analyzing and predicting stock prices. This does not warrant conducting a regression analysis. Therefore, our engagement with feature extraction and assigning of importance to respective variables will follow available wisdom, except that the outcomes will be more efficient due to the choice of models.

References:

Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques–Part II: Soft computing methods. Expert Systems with Applications, 36(3), 5932-5941.

Black, AJ, Klinkowska, O, McMillan, DG and McMillan, FJ. (2014), 'Predicting stock returns: Do commodities prices help?', Journal of Forecasting, 33, 627-639.

Cao, L. and Tay, F. E. (2001). Financial forecasting using support vector machines. Neural Computing & Applications, 10(2):184–192.

Cochrane, J. (2008), 'The dog that did not bark: A defense of return predictability. Review of Financial Studies, 21, 1533-1575.

Ince, H. and Trafalis, T. B. (2007). Kernel principal component analysis and support vector machines for stock price prediction. IIE Transactions, 39(6):629–637.

Kim, Y., Jeong, S. R., and Ghani, I. (2014). Text opinion mining to analyze news for stock market prediction. Int. J. Advance. Soft Comput. Appl, 6(1).

Kimoto, T., Asakawa, K., Yoda, M., and Takeoka, M. (1990). Stock market prediction system with modular neural networks. In Neural Networks, 1990., 1990 IJCNN International Joint Conference on, pages 1–6. IEEE.

Kohara, K., Ishikawa, T., Fukuhara, Y., and Nakamura, Y. (1997). Stock price prediction using prior knowledge and neural networks. Intelligent systems in accounting, finance and management, 6(1):11–22.

Moskowitz, T. J., Ooi, Y. H., & Pedersen, L. H. (2012). Time series momentum. Journal of financial economics, 104(2), 228-250.

Pai, P. F. and Lin, C. S. (2005). A hybrid arima and support vector machines model in stock price forecasting. Omega, 33(6):497–505.

Wang, J.-H. and Leu, J.-Y. (1996). Stock market trend prediction using arima-based neural networks. In Neural Networks, 1996., IEEE International Conference on, volume 4, pages 2160–2165. IEEE.

Mittermayer, M. A. (2004). Forecasting intraday stock price trends with text mining techniques. In System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on,

pages 10–pp. IEEE.

Nikfarjam, A., Emadzadeh, E., and Muthaiyah, S. (2010). Text mining approaches for stock market prediction. In Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on, volume 4, pages 256–260. IEEE.

Nyberg, H. (2011). Forecasting the direction of the US stock market with dynamic binary probit models. International Journal of Forecasting, 27(2), 561-578.

Regarding Feature Importance: Typically, when the feature space has too many variables for confounding, dimensionality reduction becomes a necessity. The feature space needs to be broken into non-overlapping subspaces so that the learning algorithms are trained efficiently to discriminate between classes (feature dependent). However, in this particular class of problems, where the number of features aren't too many, such an exercise doesn't facilitate the computational efficiency of the machine learning algorithms we applied. Notwithstanding, in order to understand the contribution of features toward effective discrimination (may be an academic exercise), we have included this in Section 3.5. As expected, all the features/technical indicators are significant enough. However percentage contribution of OBV steadily increases with the increase in the size of the trading window (please see Table 1 of the revised manuscript), in comparison to the other technical indicators. This is observed to be consistent across all stocks considered in our experiments.

Comment: A complete description of the data should be presented before the results. You indicate that the start date is when the company went public. Does this mean different companies are examined for different periods? The reasons for selecting the 14 companies, or why they span different counties are not clear. Was this in part to aid comparison with the results of existing papers?

Response: We have provided a description of the data in Section 3.5 in the main paper. Further description of the data is not of additional help as the paper does not specifically analyze the data as part of the forecasting problem. Our methods facilitate a *greedy* approach in order to make the best case prediction of gains or losses. We applied our methods on various datasets, such that, we selected stocks of companies from various backgrounds including dot-coms, social media platforms, electronics, sports goods, and automobile. Despite the diversity of the stocks, the results have been good and encouraging. It suggests efficiency of our methods.

Similarly, you need to clarify:

(a) The time period(s) over which training and test sets were chosen (and why).

Response: From the day that the stocks went public till 3rd March 2017 (the date we did the final test runs). The reason for selecting the entire range of data was to be able to work with a lot of data and to see how accurate and resilient a direction-prediction method based on ML can be.

(b) If a range of different training/testing configurations were trialled.

Response: As we have already explained in Section 3, we have worked with different trading windows. These are intervals of 3, 5, 10, 30, 60, and 90 days. The results for these are pre-

sented through Tables 2, 3, A.4-A.7 in the manuscript, and include various 'configurations' as suggested. The longer trading windows offer more accurate prediction across models and configurations.

(c) The number of attributes used (preferably with summary statistics). For example, was PROC calculated over different periods? PCA is referenced earlier, was it used? Were all features smoothed in the same manner? Using what alpha value?

Response: PCA was not used. The features used are described in Section 3.2. If t is the trading window, then all the attributes which are calculated over different time intervals (like PROC) were calculated over a time period of t days. For each experiment run, thus, six features were used.

(d) The ML parameters (e.g. number of trees, number of variables used for each split).

Response: We thank the reviewer for this comment. We used 100 trees with each classifier. The more the number of trees, the better is the prediction, as proven by Chebyshev's inequality. In trees in random forests, one variable is used in each node for a split whereas as function is regressed on the variables in a node in a tree in gradient boosted trees. The Gini impurity criteria then decides which is the best split at a node. We have explained points (c) and (d) in Section 3.4 in the revised manuscript.

(e) The sources on randomness in a random forest (along with their importance) should be clarified. In random forests, a random subset of features is selected for each branching decision.

Response: This is definitely an interesting an relevant question. It is for the issue pointed at by this question that we created an elaborate illustrative example. As explained in Section 3.3 in the manuscript, the number of features used to build a tree are lesser than the total number of features in the data. If there are m features in the data, then typically, $\sqrt{m}$ features are used to grow each tree. Additionally, a subset of the number of samples can also be used to create a decision tree. These are the primary *sources* of randomness in random forests as opposed to plain decision tree classifiers where all features are used in a single tree.

Comment: Results should be presented against a suitable baseline drawn from existing literature. A model that always predicts an upward trend may prove instructive. Can you provide a statistical justification that these models are superior? Are the results consistent if repeated over different time periods?

Response:

- The statistical justification is provided using the Chebyshev's inequality. It is a proof of convergence of the algorithm.
- The paper discusses the merit of using random forests and XGBoost systems as compared to other techniques used under non-ensemble procedures. Indeed, it has been shown in several studies that trained SVM applied on Korean stock market (Kim, 2003), or DT plus ANN applied on Taiwan market (Tsai and Wong, 2009) reach 56% and 67% accuracy overall.

Compared to this the present techniques attain up to about 90% accuracy. In addition, the reason for choosing random forests over decision trees is because Random Forests uses a significant amount of voting-based conclusions as compared to that of decision trees. It runs a *bagging* based routine by using a large number of de-correlated Decision Trees (consider growing forests in random fashion) to classify a predicted class. This course of operation is highly suitable for the stock data and associated classification, as it meticulously examines the feature space to make better judgments over which class to finalize as the expected outcome. We have duly included this in the literature review section.

- Moreover, as we have clarified in Section 4.3 in the revised paper (Page 11), Figure 4 (in the revised manuscript) was a visualization of a comparative analysis done by us for understanding which classifier is superior. Hence, we have included the results of other classifiers: logistic regression, SVM and ANN, based on the same preprocessing steps as those used before implementing random forests and xgboost models.

Comment: Could other performance statistics be included? For example AUC or Brier scores (you mention ROC curves in the appendix but do not use them).

Response: Thank you for this suggestion, we have done the needful. The updated results are presented through Tables 2, 3, A.4-A.7. An exhaustive set of ROC plots are included in the supplementary file in Sections 5 and 6.

Comment: Section 6.3 felt out of place and left me confused. Why consider pharmaceutical stocks? How is your study of sign predictability related to "why certain stocks did not succumb to the aggravated economic crisis"?

Response: From a machine learning and methodology point of view, it is an interesting problem to address. The reasons we want to address stocks of pharmaceutical companies are two. First, it is for the purpose of diversification – we wanted to ensure that we include stocks of different types of companies and we have set no priors to that. However, the use of pharma stocks may be considered as a case study. And second, the low fluctuations in the stocks provides an interesting premise for automation methods for stock trading suggestions and strategies. The results confirm our choice of methods and the higher-than-usual accuracy in turn verifies the nature of the stocks of pharmaceutical companies.

The *sign predictability* is related to the stocks' variance over time. Naturally, the gains or losses of stocks which are stable would be easier to predict than stocks that are relatively noisy. A lower variance in the data implies a better predictive capability of ML classifiers, and from an economic point of view, it accounts for the stability. This is included in section 4.

As the writeup may feel a bit out of place, we have moved the entire section on the discussion of the ML methods applied to pharmaceutical stocks to a section in the supplementary file in Section 5. We have not removed it completely as we believe that it does provide some interesting insights between the lines of finance and machine learning.

Comment: Comparison of the accuracy achieved in this work to the accuracies achieved in available literature." On what basis has figure 15 been compiled? The caption is confusing. Were the

3 additional models run using identical data and features? If these were to be used as comparator models this should be made clear in the earlier methodology.

Response: We have clarified the confusion in the Results section (Section 4.3). The additional classifiers that we have mentioned are those of logistic regression, SVM and ANN. We have implemented these classifiers and have reported the best-case average accuracy across different stocks for a performance baseline. In comparison to this, we observe that the best-case performance by random forests and decision trees beat that of the remaining classifiers. Reiterating what we have already clarified in Section 4.3, the feature extraction and preprocessing were the same for logistic regression, SVM and ANN as they were for RF and XGBoost. Since SVM has been a popular classifier, we have presented a representative sample of ROC curves for this classifier in Section 7 of the supplementary file.

Comment: Over what dates is figure 16 plotted? Does the large stock price drop represent a stock split (and if so why haven't you accounted for this)? What constitutes a 'buy' signal or a 'sell' signal? Are there particular RF voting thresholds that must be breached to trigger a new signal? If so, why are there consecutive buys (and consecutive sells)?

Perhaps if this was clarified you could attempt to assess the economic impact of a trading strategy based on the ML models.

Response: This figure (now figure 7 in the revised submission) was plotted from the first day that the AAPL stock went public. As a representative sample, we have plotted the subsequent gains/losses with time intervals of 90 days. The idea of 'buy' and 'sell' signals is a simplistic notion for gains and losses, respectively. We have changed the terms regarding this as this is not an end-to-end trading strategy tool: we're calling them as buy or sell indicators, and the whole perspective as 'trading indications', as the information of predicted gains and losses might be useful for making trade decisions. Reiterating what we have mentioned earlier, we are not specifically trying to handle fluctuations in data over time as we are not posing this as a time-series analysis. The random forest classifier is able to handle such dips and rises in data and that speaks for the efficacy of our approach.

Minor Comments:

Comment: Rephrase "Market risk, strongly correlated with forecasting errors, needs to be minimized to ensure minimal risk in investment". I assume the point you want to make here is that minimising forecast error would minimise risk.

Response: Thank you, we have made the necessary changes in Section 1, Introduction.

Comment: "measuring the change in price of a stock compared to its price t days back" seems like a clumsy way of saying the "t-day return".

Response: Thank you, we have made the necessary changes in the last paragraph of Section 1, Introduction.

Comment: Wouldn't investors be more interested in establishing which assets might provide positive excess returns?

Response: Indeed that most work on the subject is on forecasting prices such that investors can have excess gains. But reiterating what we have stated in the Introduction (Section 1) of the revised manuscript, that is not the problem that we're addressing here. This is a deviation from the traditional perspective of predicting prices and is only about predicting gains or losses.

Comment: "Wisdom of Crown" should be "Wisdom of Crowds"?

Response: Thank you for this comment. We have rephrased this in paragraph 1 of Introduction (Section 1).

Comment: "some stocks tend to develop linear trends in the long-run". Do you mean that they exhibit momentum or directional trends?

Response: What we meant were directional trends but we have chosen to omit this sentence in the revised submission as long-term linear trends are not something that we're specifically addressing by our methods.

Comment: "Tree based learning methods are non-metric". Perhaps clarify in terms of discrete/continuous versus nominal/ordinal.

Response: Random forests are *non-metric* classifiers, which means that unlike gradient-based methods, there are no learning-parameters which need to be set, and unlike Bayesian methods, it does not require an assumption of a prior distribution. Additionally, this is one of the reasons that has made random forests a popular classifier for various tasks. Now, this does not relate to discrete/continuous valued attributes or nominal/ordinal, as in it's basic formulation, decision trees and random forests can handle both discrete and continuous data.

We thank the reviewer for this suggestion. We have made the necessary clarification in the last paragraph of **Random Forests** in Section 3.3, Page 6 in the revised submission.

Comment: Boosting a classifier means combining the results of many weak predictors to make a strong prediction". Isn't it more that this? A RF does that.

Response: Thank you for this comment. There's a lot of difference between bootstrap aggregation (*bagging*, which is what happens in random forests) and boosting. What happens in an RF is that a certain number of decision trees are randomly constructed on a subset of the feature-space. However, in boosting, every subsequent learner tries to better classify the misclassified samples of the previous learner. While RF takes into consideration the votes of randomly created tree, boosted tree algorithms move towards a better classification in every step. Additionally, the number of levels in a tree in an RF is usually not greatly constrained (unless we want to incorporate regularization by tree-pruning), whereas the number of levels in a tree in a boosted algorithm is usually very less, a ballpark figure of say between 1 and 5 – this is what we mean by a *weak* learner, that the individual learner's capability itself is only slightly better than an arbitrary guess, but together, whilst constructed carefully, an aggregate of similar weak learners can provide very

accurate classifications.

Comment: Why not present all results as percentages?

Response: Generally, accuracy is represented as a percentage whereas the other measures of goodness are represented as fractions or decimals. Trying to keep at par with the general practice, we have not presented all the results as percentages. For instance, the Brier score is usually represented as a number between 0 and 1. And the same applies for sensitivity, specificity, AUC, etc. With all due respect, we do not see how a percentage representation or a fractional representation might affect the interpretability of the results.

Comment: Figure 14 has many series (mostly blue) but only 4 legend items.

Response: Dear Reviewer, we have mentioned in the caption that the blue items correspond to the change in accuracies of the stocks that we have presented in the main analysis (whose results are presented in Tables 2, 3, A.4-A.7). Now, the reason we did not present specific legends corresponding to each of these as the highlight of that figure to show that convergence was accomplished faster for the pharmaceutical stocks.

Comment: Fix Lane (1984) bibliography entry.

Response: Thank you for this comment. We have rectified the entry.

Comment: Review paper to ensure abbreviations are consistently introduced and used.

Response: Thank you. We have ensured this.

Comment: Take care to clarify/standardise use of terms such as 'accuracy rate', 'hit rate' and 'success rate'.

Response: Thank you. We have ensured this.

Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques–Part II: Soft computing methods. Expert Systems with Applications, 36(3), 5932-5941.

Christoffersen, P. F., & Diebold, F. X. (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. Management Science, 52(8), 1273-1287.

Moskowitz, T. J., Ooi, Y. H., & Pedersen, L. H. (2012). Time series momentum. Journal of financial economics, 104(2), 228-250.

Nyberg, H. (2011). Forecasting the direction of the US stock market with dynamic binary probit models. International Journal of Forecasting, 27(2), 561-578.

Authors' reply:

We thank the reviewer for specifically pointing out what needs to be changed. We have made corrections in every statement pointed out and have also revised other sections where we had used similar ambiguous phrases. We have added the suggested references.

Reply to the Reviewer's (Reviewer 2) Comments on *Predicting the Direction of Stock Market Price Using Tree-Based Classifiers*

June 16, 2018

Summary: In this paper the authors studied the prediction of stock market direction using two ensemble learning algorithms: random forest and XGboost. After preprocessing the data six technical indicators are calculated and the two algorithms are implemented using scikit learn and other software packages taking the technical indicators as input features. Discussion of results are made in the case of data taken from 10 specific companies and it is concluded that these two methods outperform logistic regression, neural network ,and SVM.

The following are some of my observations and suggestions:

Comment: The paper is too long. It can be shortened by omitting sections 4.1-4.4,,5.1-5.3 (which contain detail description of decision trees, random forest, XGboost, along with a worked out example, some derivations,etc.) as these are standard material available in data mining /machine learning text books. Section 4.5 and 4.6 about chebychev inequality and its proof can also be omitted. However , a very brief description of random forest and XGboost with the algorithms may be given in the appendix. Again the appendix A with key definitions also are to be omitted. Also the presentation in sections 1,2,and 3 may be shortened.

Response: First of all, the authors thank the reviewer for reading our work intensively and for the very useful comments. We have addressed each issue raised by the reviewer and the detailed responses follow.

We have shortened the paper and have moved the proofs. However, it is worth mentioning that the documentation on the ML techniques applied to the problem at hand is original and therefore, should be made accessible to the readers in some form. Consequently, the parts which contain mathematical notations and proofs have relegated to a supplementary file. In the supplementary part, we request retaining the proof of the Chebyshev's inequality applied to random forests since it provides the theoretical basis of why such classifiers work.

We have also merged the sections containing detail description of decision trees, random forest, XGBoost, along with a worked out example, some derivations, etc. with the supplementary file. Once again, for supporting future research in this area, print or online publication of these materials should be quite useful for the readers.

Comment: The authors claim that random forest and XGboost outperform SVM, ANN, logistic regression based on the values of performance metrics for RF and XGboost. But the corresponding values for SVM and other methods have not been computed for the data set used by the authors. To justify this claim a table containing the values of performance metrics(such as accuracy ,etc) for the three methods SVM, RF, XGboost for the same data set be prepared and the values be compared. This may further be illustrated using ROC curves for the three methods for the same data set.

Response: A sincere thanks to the reviewer for this suggestion.

As per suggestion, we have presented the results for a variety of companies from different backgrounds. Table 2 offers an example and data for other companies are available in Appendix A. The results are encouraging and indicate that these methods work well in a general sense. We have cited sources in the literature for the other methods and have presented more extensive results, in support of the comparative analysis. Fig.4 in the main text presents a comparative study to this effect and for our claim.

A clarification that we have made in Section 4.3 of the revised manuscript is that the comparisons that we have made with the other methods were done by us. Perhaps what was unclear from the previous version of the paper is that the comparisons were not done only with results available in literature. Such an ambiguity should not be prevalent here on.

Further, following the Reviewer's suggestions, we have presented a representative sample of ROC plots for SVM in Section 7 of the supplementary file, after repeating the experiments carefully with SVM. These plots are consistent with what we have presented in Figure 4 in the main test. Note that it is a small section and is not as exhaustive as our analysis using RF and XGBoost as the application of SVM is not the cornerstone of the paper. Notwithstanding, the representative sample of ROC curves should suffice to facilitate the readers' understanding of the efficacy of SVM applied to this problem.

Comment: Authors mention that return on the stock can be predicted using the model. But the model predicts only upward and downward movement. How can one determine the return on stock ? Similarly with out a quantitative predicted value of stock how can one make efficient portfolio management?

Response: We sincerely thank the reviewer for this comment. We understand that this sentence is ambiguous and taken literally, does not make sense in the purview of our work. We have changed this statement appropriately to remove the confusion.

Comment: Stock price prediction is a more general problem then stock direction prediction as the former gives more information than later. So in the title from forecasting to classification does not seem reasonable. This part may be omitted.

Response: We have changed the title.

Comment: The size of the data set should be mentioned in sec.7. Deep learning method may be explored only if available data size is big.

Response: We have mentioned in the revised manuscript that we have used the data (closing prices) from the date that the stocks went public, along with the range of sizes of the datasets, in Section 3.5.

Closing Remarks: We thank the reviewer again for specifically pointing out what needs to be changed. We have made corrections in every statement pointed out and have also revised other sections in line with these comments.

Predicting the Direction of Stock Market Prices Using Tree-Based Classifiers

Suryoday Basak[a], Saibal Kar[b], Snehanshu Saha[a], Luckyson Khaidem[a], Sudeepa Roy Dey[a]

[a] *Authors are affiliated with the Department of Computer Science and Engineering, and Center for Applied Mathematical Modeling and Simulation (CAMMS), PESIT South Campus, Bangalore, India*
[b] *Author is affiliated with the Centre for Studies in Social Sciences, Calcutta, India and IZA, Bonn*

Abstract

Predicting returns in the stock market is usually posed as a forecasting problem where prices are predicted. Intrinsic volatility in the stock market across the globe makes the task of prediction challenging. Consequently, forecasting and diffusion modeling undermines a diverse range of problems encountered in predicting trends in the stock market. Minimizing forecasting error would minimize investment risk. In the current work, we pose the problem as a direction-predicting exercise signifying gains and losses. We develop an experimental framework for the classification problem which predicts whether stock prices will increase or decrease with respect to the price prevailing n days earlier. Two algorithms, random forests, and gradient boosted decision trees (using XGBoost) facilitate this connection by using ensembles of decision trees. We test our approach and report the accuracies for a variety of companies as improvement over existing predictions. A novelty of the current work is about the selection of technical indicators and their use as features, with high accuracy for medium to long-run prediction of stock price direction.

Keywords: stock price movement, xgboost, random forests, machine classification

1. Introduction and Motivation

For a long time, it was believed that changes in the prices of stocks is not forecastable. The well known Random Walk hypothesis (Malkiel & Fama, 1970; Malkiel, 2003), and the Efficient Market Hypothesis (Jensen, 1978), which state that a market is efficient with respect to a current information set $I(t)$ if it is impossible to make economic gains in this market, led to this belief. In other words, if it is impossible to outperform the market owing to the randomness in stock prices, then unless a different (often excessive) type of risk is considered, economic profits cannot rise. It is unclear, however, how such risk would be measured (see the work by Timmermann & Granger (2004), where stock prices are treated as a martingale). Therefore, it should be of little doubt that predicting the trends in stock market prices is a challenging task and that there is high returns from improving value at risk forecasts (Halblieb & Pohlmeier, 2012). On the contrary, the Wisdom of Crowd hypothesis, which emerges from the theory of collaborative filtering, states that many individuals, each with limited information, can provide very accurate assessments if their information is elicited in an appropriate fashion. It is, however, not known to be useful for predicting stock market returns; nonetheless, some individual, as well as institutional investors are able to beat the market to make profits (Avery, Chevalier & Zeckhauser, 2016). The inefficiency of prediction gets accentuated due to various uncertainties involved and owing to the presence of multiple variables all of which can potentially influence the market value on a particular day. Over time, a number of explanatory variables have been added to this enormous literature (see a history of EMH in (Sewell, 2011; Beechey, Gruen & Vickery, 2000) etc.): these include country-specific economic conditions, investors' sentiments towards a particular company, political events, etc. Consequently, stock markets are susceptible to quick changes, which often turn into random fluctuations in the stock prices (for calibration of agent-based dynamics, refer to the work by Recchioni et al. (2015)). Notwithstanding, EMH has certain fault lines that should be mentioned before we develop two algorithms to observe the quality of predictability for stock prices.

Indeed, (Malkiel, 2003) offers a masterly discussion of the critique of the EMH, and suggests that, way back in 1973, he in general advised investors to purchase broad-based index funds that bought and held all the stocks in the market and that charged very low expenses. He admits that by the start of the twenty-first century, the intellectual dominance of the efficient market hypothesis had become far less universal. Many

Predicting the Direction of Stock Market Prices Using Tree-Based Classifiers

Abstract

Predicting returns in the stock market is usually posed as a forecasting problem where prices are predicted. Intrinsic volatility in the stock market across the globe makes the task of prediction challenging. Consequently, forecasting and diffusion modeling undermines a diverse range of problems encountered in predicting trends in the stock market. Minimizing forecasting error would minimize investment risk. In the current work, we pose the problem as a direction-predicting exercise signifying gains and losses. We develop an experimental framework for the classification problem which predicts whether stock prices will increase or decrease with respect to the price prevailing n days earlier. Two algorithms, random forests, and gradient boosted decision trees (using XGBoost) facilitate this connection by using ensembles of decision trees. We test our approach and report the accuracies for a variety of companies as improvement over existing predictions. A novelty of the current work is about the selection of technical indicators and their use as features, with high accuracy for medium to long-run prediction of stock price direction.

Keywords: stock price movement, xgboost, random forests, machine classification

1. Introduction and Motivation

For a long time, it was believed that changes in the prices of stocks is not forecastable. The well known Random Walk hypothesis (Malkiel & Fama, 1970; Malkiel, 2003), and the Efficient Market Hypothesis (Jensen, 1978), which state that a market is efficient with respect to a current information set $I(t)$ if it is impossible to make economic gains in this market, led to this belief. In other words, if it is impossible to outperform the market owing to the randomness in stock prices, then unless a different (often excessive) type of risk is considered, economic profits cannot rise. It is unclear, however, how such risk would be measured (see the work by Timmermann & Granger (2004), where stock prices are treated as a martingale). Therefore, it should be of little doubt that predicting the trends in stock market prices is a challenging task and that there is high returns from improving value at risk forecasts (Halblieb & Pohlmeier, 2012). On the contrary, the Wisdom of Crowd hypothesis, which emerges from the theory of collaborative filtering, states that many individuals, each with limited information, can provide very accurate assessments if their information is elicited in an appropriate fashion. It is, however, not known to be useful for predicting stock market returns; nonetheless, some individual, as well as institutional investors are able to beat the market to make profits (Avery, Chevalier & Zeckhauser, 2016). The inefficiency of prediction gets accentuated due to various uncertainties involved and owing to the presence of multiple variables all of which can potentially influence the market value on a particular day. Over time, a number of explanatory variables have been added to this enormous literature (see a history of EMH in (Sewell, 2011; Beechey, Gruen & Vickery, 2000) etc.): these include country-specific economic conditions, investors' sentiments towards a particular company, political events, etc. Consequently, stock markets are susceptible to quick changes, which often turn into random fluctuations in the stock prices (for calibration of agent-based dynamics, refer to the work by Recchioni et al. (2015)). Notwithstanding, EMH has certain fault lines that should be mentioned before we develop two algorithms to observe the quality of predictability for stock prices.

Indeed, (Malkiel, 2003) offers a masterly discussion of the critique of the EMH, and suggests that, way back in 1973, he in general advised investors to purchase broad-based index funds that bought and held all the stocks in the market and that charged very low expenses. He admits that by the start of the twenty-first century, the intellectual dominance of the efficient market hypothesis had become far less universal. Many financial economists and statisticians began to believe that stock prices are at least partially predictable. Furthermore, a new breed of economists emphasized psychological and behavioral elements of stock-price determination and volatility (Christoffersen & Diebold, 2006), and came to believe that future stock prices

are somewhat predictable on the basis of past stock price patterns as well as certain 'fundamental' valuation metrics. We have discussed this issue further while explaining how variables are chosen in atypical models of stock market prediction. Later, many economists made controversial claims that these predictable patterns enable investors to earn excess risk-adjusted rates of return. From Shiller's behavioral 'bandwagon effect' to 'head and shoulders' and 'double bottoms' formations in stock prices (Shiller, 2000), modest predictive power seems to exist (Lo, Mamaysky & Wang, 2000), nonetheless. (Malkiel & Fama, 1970) states that while the stock market may not be a mathematically perfect random walk, it is important to distinguish statistical significance from economic significance. The statistical dependencies giving rise to momentum are extremely small and are not likely to permit investors to realize excess returns. The anomalies with stock returns were often modeled in a context-specific manner and do not qualify as generalization.

It is well-known that stock market price series are generally dynamic, non-parametric, chaotic and noisy in nature making investments intrinsically risky. In addition, in view of the model specification to follow shortly, we are mindful of the fact that stock market price movement is considered to be a random process with fluctuations that are more prominent in the short-run. It is needless to mention that advanced knowledge of near future stock price movements should help in minimizing this risk. Traders are more likely to buy a stock in the current period whose value is expected to increase in the future and conversely for falling prices. It is straightforward therefore, that accurate prediction of the trends in stock market prices maximizes capital gains and minimizes losses. Thus, it had best be admitted that adding value to this complex and deeply researched topic is not easy, especially in view of the millions of data points that are being generated around the world for every time period under consideration. To this end, this paper presents the use of techniques of Machine Learning (ML) to predict stock prices at the level of a firm to get better insights into the accuracy of price movements. Notably, ML techniques used towards forecasting has a long history. In fact, the use of a classifier system (Beltrametti et al., 1997) is not new to this literature. However, more intensive use of this technique with large set of data points generated every time period is only natural (viz. Xu, *et. al.*, 2018 on risk management for portfolio investments). The two ML algorithms that we use are, namely, *random forests* (RF) and forests of *gradient boosted decision trees* (GBDT). Decision trees in RF, work by finding the best threshold. Based on this, the feature space is recursively split. In comparison, GBDT models approximate regressors to the training samples and find the best split of the aggregate of the regressor functions. In this connection, it is useful to mention that XGBoost is a fairly new invention, offering a tool that reduces the time taken to construct GBDTs by reducing the time for training of a complete model.

It is to be noted that, application of ML models in stock market behavior is a rather recent phenomenon (Khaidem, Saha & Dey, 2016). The approach is a departure from traditional forecasting and diffusion type methods. Standard models used in stock price forecasting involves statistical methods such as time series modeling and multivariate analysis (Gencay, 1999; Timmermann & Granger, 2004; Bao & Yang, 2008), where, the stock price movements are usually treated as a function of time and solved as a regression problem. Conversely, in this paper as we pose it as a classification problem, the class label of each sample is determined by considering the t-day return. In our analysis, we have conducted experiments on $t = 3, 5, 10, 15, 30, 60,$ and 90 days. The goal is to design an intelligent model that learns from the market data using machine learning techniques and predicts the *direction* in which a stock price will change at the closing time everyday. The ability to forecast direction of stock prices for individuals and companies capable of holding on to investments over medium to long-run should be a very useful support to this literature.

2. Related Work

As a relevant perspective, note that, stock market analysis and prediction have been studied with the aid of methods such as machine learning and text mining. The literature below shall highlight some of the papers that use various algorithms under ML techniques and applies it to a limited set of stocks, country-wise and company-wise to evaluate the strength of the propositions. The use of prediction algorithms to determine future trends in stock market prices (Widom, 1995; Hellstrom & Holmstromm, 1998; Gencay, 1999; Li, Yang & Li, 2014; Dai & Zhang, 2013; Timmermann & Granger, 2004; Bao & Yang, 2008) is a way to improve upon the predictive ability and to re-evaluate the efficient market hypothesis and diffusion models (Saha, Routh & Goswami, 2014). The debate is by no means over, since algorithms that can model more complex dynamics of the financial system (Malkiel, 2003) have added to the prevailing controversy about whether stock prices are at all predictable. In addition, researchers considered esoteric ideas in this regard,

such as the correlation between volatility in the stock market and songs in the billboard top 100 (Maymin, 2012), and standard approaches such as event analysis (Khanal & Mishra, 2017) for analysis of stock price reactions to stock dividend announcements. Furthermore, since behavior and individualized responses play a significant role in dictating the stock turnovers and prices, a few studies have engaged with word analysis of news articles (Mittermayer, 2004; Nikfarjam, Emadzadeh, & Muthaiyah, 2010; Nyberg, 2011; Kim, Jeong, & Ghani, 2014) and its predictive ability. ML, however, is a set of techniques that are relatively new. Consequently, several algorithms have been used in stock prediction such as support vector machine (SVM), artificial neural networks (ANN), linear discriminant analysis (LDA), linear regression, K-NN, and naïve Bayesian Classifier (Khan et al., 2014) to approach the subject of predictability with greater accuracy.

Of these, the studies applying data mining commonly use daily stock data. For example, prediction studies based on support vector machines (SVMs) (Cao & Tey, 2001; Ince & Trafalis, 2007; Atsalakis & Valavanis, 2009) have been conducted to determine pattern categories. The relevant literature survey reveals that SVM has been used most of the time in stock prediction research. Typically, autoregressive integrated moving average (ARIMA) model (Pai & Lin, 2005; Wang & Leu, 1996; Moskowitz, Ooi, & Pedersen, 2012) are used for identifying and predicting time series variations. The sensitivity of stock prices to external conditions have duly been considered Li, Yang & Li (2014). These include, the external conditions that are taken into consideration covering daily quotes of commodity prices such as gold, crude oil, natural gas, corn, and cotton in two major foreign currencies (EUR, JPY). These studies also collected daily trading data of 2666 U.S stocks trading (or once traded) at NYSE or NASDAQ from 1^{st} January 2000 to 10^{th} November 2014. This dataset includes the opening price, closing price, highest price, lowest price, and trading volume of every stock for each day over which the data was collected. Features were derived using the information from the historical stock data as well as external variables which were mentioned earlier in this section. For these papers, logistic regression turned out to be the best model with a success rate of 55.65%. In the paper by Dai & Zhang (2013), the data used for the analysis were stock (closing) prices of the company 3M. The data contained daily stock information ranging from 1^{st} September 2008 to 11^{th} August 2013 (1471 data points). Multiple algorithms were chosen to train the prediction system. The algorithms used were logistic regression, quadratic discriminant analysis, and SVM. These algorithms were used for predicting the direction of the stock on the successive day corresponding to a given data sample; it also predicted the price after the next n days. The accuracy of the successive-day prediction model ranged from 44.52% to 58.2%. The results in (Dai & Zhang, 2013) were justified on the grounds that the US stock market is semi-strong efficient, meaning that, neither fundamental nor technical analysis can be used to achieve superior gains. However, the long-run prediction model produced better results which peaked when the time window was 44 days. SVM reported the highest accuracy of 79.3%. In Di (2014), the authors have used 3 stocks (AAPL, MSFT, AMZN) between 4^{th} January 2010 and 10^{th} December 2014. Various technical indicators such as RSI, On Balance Volume, Williams %R, etc. were used as features. Out of 84 features, an extremely randomized tree algorithm, as described by Geurts & Louppe (2014), was implemented for the selection of the most relevant features. These features were then fed to an SVM with RBF kernel for training. Devi (2015) has proposed a model which uses a hybrid cuckoo search with support vector machine (with Gaussian kernel): the cuckoo search method optimizes the parameters of support vector machine. The proposed model used technical indicators: RSI, Money Flow Index, EMA, Stochastic Oscillator, and MACD as features. The data used in the proposed system consists of daily closing prices of BSE-Sensex and CNX - Nifty from Yahoo finance between January 2013 and July 2014. Giacomel, Galante & Pareira (2015) proposed a trading agent based on a neural network ensemble that predicts if a certain stock is going to rise or fall. They evaluated their model using two datasets: the North American and the Brazilian stock markets and achieved hit rates of greater than 56%. They even performed a simulation based on the predictions of their classifier, whose results were promising. In other studies, Artificial Neural Networks (ANNs) (Kimoto et al., 1990; Kohara et al., 1997) have been employed to achieve good predictions even in the case of complex relationships of variables. Boonpeng & Jeatrakul (2015) implemented a one-against-all (OAA-NN) and one-against-one neural network (OAO-NN) to classify buy, hold, or sell data and compared their performance with a traditional neural network. Historical data of Stock Exchange of Thailand (SET) for seven years (3^{rd} January 2007 to 29^{th} August 2014) was selected for testing. It was found that OAA-NN performed better than OAO-NN and traditional NN models, producing an average accuracy of 72.50%. In (Qiu & Song, 2016), an optimized ANN using genetic algorithms (GA) has been tried to predict the direction of the stock market in a similar fashion of the current work; the hit ratio achieved here for two types of data, based on different sets of technical

indicators, for the Nikkei 225 index (Tokyo Stock Exchange) are 61.87% and 81.27%. Clearly, the intense applications of ML have found wide acceptance all around and have arguably been successful in attaining reasonably good accuracy of predictions.

However, most of these studies have some limitations for medium to long-run predictions. First, without filtering for outliers, the predictions based on all historical data leads to potential errors. Second, although the total completion price is determined by a variety of factors such as the foreign purchase closing price and domestic selling completion amount, this set needs to be expanded in order to reduce omitted variables bias. Variables of importance may include, categories of financial ratios, macro, labour market and housing variables and measures of sentiment and leverage (Black et al., 2014; Cochrane, 2008). With respect to the current paper, it is important to note that the main purpose is to implement two distinct methods on stock data and highlight their advantage over other non-ensemble techniques within the machine learning approaches for analyzing and predicting stock prices. This does not warrant conducting a regression analysis. Therefore, our engagement with feature extraction and assigning of importance to respective variables will follow available wisdom, except that the outcomes are expected to be more efficient due to the choice of models.

The focus of the current paper is therefore to implement random forests, and gradient boosted trees on stock data, and to discuss its advantages over non-ensemble techniques. These models have been trained on time intervals of 3, 5, 10, 15, 30, 60, and 90 days days and the results are impressive. Moreover, majority of the related work focused on the time window of 10 to 44 days on an average as most of the previous studies preferred to use metric classifiers on time series data which was not smoothened. Therefore, these models are unable to learn from the data set when it comes to predicting for a long run window. In the method we propose, the data is first preprocessed using exponential smoothing, following which the increase or decrease in prices are calculated. After that, the classification algorithms are applied which do the job of predicting gains or losses. The extension of the time window to 90 days over which predictions are made is almost double the time adopted in previous studies (44 days). Significant improvement in accuracy obtained by our approach clearly establishes the superiority of the model developed.

It is important to note that forecasting techniques are rather insensitive to the crests and troughs in the time-series data. In the event of data point insufficiency or improper data conditioning, incorrect forecasting is common. The forecasting may generate some random value(s) which turns out to be outliers. For example, forecasting the price of a commodity as zero is one such instance. From an economic point of view, unless a commodity is a free good, usually publicly provided, the commodity price is unlikely to be zero in the market transactions. On the contrary, the classification model provides a probabilistic view of the predictive analysis and hence it plays a safer role as it predicts the *direction* of the trend. It uses the likelihood of the situation and hence the results are more trustworthy. A possible way to avoid the inherent problems in forecasting methods is therefore by recasting the problem as one that implements tree-based classifiers in ensemble learning as we adopt here. Our work takes a fresh and *gritty* perspective on the problem and our results are indicative of only potential gains or losses.

The remainder of the paper is organized as follows. Section 3 discusses the methods of preprocessing the classification, the features used (such as Relative Strength Index, Stochastic Oscillator, etc), and touches on how the experiments were performed. In Section 3.5, we discuss the dataset in detail. Section 4 discusses the results obtained, followed by a justification of the working of the method and a brief comparative analysis, establishing the superiority of the proposed system. We conclude by summarizing the results in Section 5. The results are further expanded in Appendix A.

3. Methodology and Analysis

In our experiments, the time series data acquired is first exponentially smoothed. Then the technical indicators are extracted. Technical indicators provide insights to the expected stock price behavior in future. These technical indicators are used as features to train the classifiers. The preprocessing, feature extraction, and classification methods are described in this section.

3.1. Data Preprocessing

Exponential smoothing grants larger weights to the recent observations and exponentially decreases weights of the past observations. The exponentially smoothed statistic of a series Y can be recursively

calculated as:

$$\begin{aligned} S_0 &= Y_0 \\ \text{for } t > 0,\ S_t &= \alpha * Y_t + (1 - \alpha) * S_{t-1} \end{aligned} \tag{1}$$

where α is the smoothing factor and $0 < \alpha < 1$. Larger values of α reduce the level of smoothing. When $\alpha = 1$, the smoothed statistic becomes equal to the actual observation. The smoothed statistic S_t can be calculated as soon as consecutive observations are available. This smoothing removes random variation or noise from the historical data, allowing the model to easily identify the long-term price trend in the stock price behavior. Technical indicators are then calculated from the exponentially smoothed time series data which are later organized into a feature matrix. The target to be predicted in the i^{th} day is calculated as follows:

$$target_i = sign(close_{i+d} - close_i) \tag{2}$$

where d is the number of days after which the prediction is to be made. When the value of $target_i$ is +1, it indicates that there is a positive shift in the price after d days; -1 indicates that there is a negative shift after d days, giving us an idea of the direction of the prices for the respective stock. The $target_i$ values are assigned as labels to the i^{th} row in the feature matrix.

3.2. Feature Extraction from Data

In our solution, we consider only the closing price of a stock and we collect these values for many years. Hence, our input data can be considered to be of the form $(date, price_{closing})$. From the data, the following indicators are calculated:

1. **Relative Strength Index (RSI)**: RSI (Williams, 1978) is a popular momentum indicator which determines whether the stock is over-purchased or over-sold. A stock is said to be overbought when the demand unjustifiably pushes the price upwards. This condition is generally interpreted as a sign that the stock is overvalued and the price is likely to go down. A stock is said to be oversold when the price goes down sharply to a level below its true value. This is a result caused due to panic selling. RSI ranges from 0 to 100 and generally, when RSI is above 70, it may indicate that the stock is overbought and when RSI is below 30, it may indicate the stock is oversold.

2. **Stochastic Oscillator (SO)**: Stochastic Oscillator (Lane, 1984) follows the momentum of the price. As a rule, momentum changes before the price changes. It measures the level of the closing price relative to low-high range over a period of time.

3. **Williams Percentage Range (W%R)**: Williams Percentage Range (Williams, 1972) or Williams %R is another momentum indicator, similar in idea to stochastic oscillator. The Williams %R indicates the level of a market's closing price in relation to the highest price for the look-back period, which is 14 days. It's value ranges from -100 to 0. When its value is above -20, it indicates a *sell signal* and when its value is below -80, it indicates a *buy signal*.

4. **Moving Average Convergence Divergence (MACD)**: The moving average convergence-divergence (MACD) (Appel, 2005) is a momentum indicator which compares two moving averages of prices. The first moving average is a 26-day exponential moving average (EMA) and the second moving average is a 12-day EMA. The 26-day EMA is subtracted from the 12-day EMA. A 9-day EMA of the MACD is considered as the *signal line*, which serves as the threshold for the *buy* or *sell* signals.

5. **Price Rate of Change (PROC)**: The Price Rate of Change (PROC), (Larson, 2015) is a technical indicator which reflects the percentage change in price between the current price and the price over the window that we consider to be the time period of observation.

6. **On Balance Volume (OBV)**: On balance volume (OBV) (Granville, 1976) utilizes changes in volume to estimate changes in stock prices. This technical indicator is used to find buying and selling trends of a stock, by considering the cumulative volume: it cumulatively adds the volumes on days when the prices go up, and subtracts the volume on the days when prices go down, compared to the prices of the previous day.

3.3. Machine Learning Algorithms

Here we describe the algorithms that we have used for classification.

1. **Random Forest**: Decision trees (Quinlan, 1992; Geurts & Louppe, 2014) and random forests (Breiman, 2001) are popular machine learning approaches which can be used to solve a wide range of problems in classification. The basic training principle of decision trees is the recursive partitioning of the feature space using a tree structure, where each child node is split until pure nodes, i.e nodes which contain samples of a single class, are achieved. The splitting is done by the means of a criteria which tries to maximize the purity of the child nodes relative to their respective parent nodes. As maximum purity is ensured in child nodes, subsequently, pure nodes are arrived at. These pure nodes are not split further and constitute the leaf nodes. When a decision tree is used for the classification of a test sample, it is traced all the way down to a leaf node of the tree; as the leaf nodes of a decision tree are pure, the respective test sample is assigned the class label of the training samples of leaf node it arrives at. Random forests use an ensemble of many decision trees to reduce the effects of over-fitting. In a random forest, each tree is grown on a random subset of the feature space. Usually, if each sample in the data set has M features, $m = \sqrt{M}$ features are randomly selected to grow each tree. The reason random forests are preferred over decision trees is that random forests use a significant amount of voting based conclusions as compared to that of decision trees. It runs a bootstrap aggregation (or *bagging*) (Breiman, 2001) based routine by using a large number of de-correlated decision trees to classify a test sample. This course of operations is highly suitable for the stock data and its associated classification, as it meticulously examines the feature space to make better judgments over which class to finalize as the expected outcome.
   Gini impurity is used as the function to measure the quality of split in each node. Gini impurity at node N is given by:

$$G(N) = 1 - (P_1)^2 - (P_{-1})^2 \tag{3}$$

   where P_i is the proportion of the population with class label i. The obvious heuristic approach to choose the best splitting decision at a node is the one that reduces the impurity as much as possible. In order words, the best split is characterized by the highest gain in information or the highest reduction in impurity.
   Random forests are *non-metric* classifiers, which means that unlike gradient-based methods, there are no learning-parameters which need to be set, and unlike Bayesian methods, it does not require an assumption of a prior distribution (Quinlan, 1992). Additionally, this is one of the reasons that has made random forests a popular classifier for various tasks.

2. **Gradient Boosted Decision Trees**: *Boosting* a classifier means combining the results of many weak predictors to make a strong prediction. Boosting has many variants but all of them work by optimally selecting or building successive weak classifiers such that the prediction resulting from many weak classifiers is strong. Gradient boosting is an improvement on decision trees, where each tree is approximated as an aggregate of many regressor functions $f_i(x)$. This is in contrast to the traditional idea of decision trees, wherein a Gini impurity based splitting is directly used to find the best split in the feature space. Each successive function f_i is built such that the misclassification rate successively decreases. This is done by trying to better classify the *residuals*, or the misclassified samples of the i^{th} iteration in the $i + 1^{th}$ iteration. Hence, the error in classification successively decreases. This step-wise aggregation of functions approximates a node in a tree, and eventually, the entire tree is approximated. Once each tree has been optimally approximated, the *structure scores* and *gain* are calculated, based on which the best split is determined. There are obvious advantages of doing this:

since each tree is built carefully, a lot of random trees need not be used in classifying a random sample, substantially decreasing the number of trees required in the forest of classifiers. XGBoost is a framework and library which parallelizes the growth of gradient boosted trees in a forest. The idea of gradient boosting (Friedman, 2000) comes from the principle of gradient descent: a greater number of the misclassified samples of the i^{th} learner should be classified correctly by the $i + 1^{th}$ learner, and so on. This implies that the error in classification reduces as more number of regressors are constructed in each node. More specifically, in the case of gradient boosted decision trees (GBDT), the $i + 1^{th}$ regression function is expected to rectify the mistakes of the i^{th} function. Since the error goes on decreasing as a tree is approximated by a larger number of functions, gradient boosting is considered to be a convex optimization problem. XGBoost (Chen & Guestrin, 2016) aims to minimize the time required to grow trees. This makes GBDTs more practical to use. Here, too, a subset of the features is used to build each tree. However, in the algorithmic description of XGBoost, the set of features being used to grow the tree is deliberately excluded: the assumption is that a subset of features is drawn out and fed into the algorithm.

3.4. Framework and Experimental Setup

The various aspects of the entire system consists of parts which need to be appropriately set. In all the experiments we performed, we used the following as settings for the preprocessing and classification:

1. The value of α for the exponential smoothing function was taken to be 0.095.
2. The trading window was varied as 3, 5, 10, 15, 30, 60, and 90 days.
3. The number of trees in random forests and XGBoost are taken to be 100. The more the number of trees, the better is the prediction, as proven by Chebyshev's inequality (see Section 2 of the Supplementary File).
4. In trees in random forests, one variable is used in each node for a split whereas as function is regressed on the variables in a node in a tree in gradient boosted trees. The Gini impurity criteria then decides which is the best split at a node.

The same settings are used for all the experiment runs for different time windows. The value of α is deliberately kept small in order to cause a small change the prices of the stocks. After the data is preprocessed and prepared for classification, we end up with a dataset containing the technical indicators as features and directions ± 1 as class labels. Following this, the dataset is randomly divided into training and test sets and for each trading window for each stock, the experiments are repeated many times over. The average of the results are presented in Section 4 and Appendix A.

3.5. Data Set & Features

The results are based on data for ten companies (see Appendix A) and all of the data available have been used, i.e., starting from the day they went public till 3^{rd} February 2017. These companies were sampled randomly, without any rigorous consideration of their background or the kind of economic impact they have on society. Some of these companies are software companies (FB, TWTR), electronics companies (AMS), automobile (TATA), etc. We like to emphasize that the diversity of the background of companies thus chosen for analysis of stock prices is crucial for ensuring the efficacy of the algorithms.

The raw values considered from the data include that acquired at the date of entry, the closing price, and the volume, etc. In their raw form, the size of the data corresponding to the stocks of different companies varied anywhere between 10kB and 700kB, with the number of rows (corresponding to closing prices) varying between 1180 and 10,700. Based on the closing prices, the remaining technical indicators (used as features for the learning algorithms) are determined. The data sets do not contain categorical and ordinal variables: all the feature values are continuous. The trends generally observed among most features is non-linear. This makes tree-based classifiers an attractive suite of algorithms for exploration. As a first insight, we present the feature importances as determined by the algorithms in Table 1. From this, we can see that it is seldom the case, especially for shorter trade windows, to be excessively influenced by a single feature or a small feature set. Combining the fact that we used a small number of features with this observation, we see that any need for dimensionality reduction is effectively superseded.

Table 1: Feature Importances: Sample data set

Company	Indicator	Trading Window (in number of days) and Corresponding Importance of Features (in %)						
		3	5	10	15	30	60	90
AAPL	RSI	17.98	17.34	17.49	16.88	16.19	13.81	12.66
	SO	14.23	13.51	12.82	12.43	11.61	10.3	9.41
	W%R	14.12	13.63	12.72	12.56	11.51	10.34	9.28
	MACD	17.8	17.89	18.2	18.16	17.7	16.48	16.34
	PROC	16.74	16.28	15.19	15.53	16	18.96	21.6
	OBV	19.12	21.36	23.57	24.43	26.99	30.11	30.72
FB	RSI	17.43	18.31	18.01	17.3	15.6	14.35	12.06
	SO	14.86	14.5	12.58	11.78	10.25	9.22	9.45
	W%R	14.8	13.13	12.34	11.64	11.42	9.28	10.77
	MACD	17.7	18.26	17.15	17.9	16.18	12.81	14.01
	PROC	16.17	15.77	15.55	16.11	17.19	22.44	22.13
	OBV	19.04	20.03	24.37	25.28	29.37	31.91	31.58

Typically, when the feature space has too many variables for confounding, dimensionality reduction becomes a necessity. The feature space needs to be broken into non-overlapping subspaces so that the learning algorithms are trained efficiently to discriminate between classes (feature dependent). However, in this particular class of problems, where the number of features aren't too many, such an exercise doesn't facilitate the computational efficiency of the machine learning algorithms we applied. However, in order to understand the contribution of features toward effective discrimination (as an academic exercise), we have included this. As expected, all the features/technical indicators are significant enough. However percentage contribution of OBV increases drastically with the increase in the size of the trading window (Please see table 1), in comparison to the other technical indicators. This is observed to be consistent across all stocks considered in our experiments.

4. Results and Discussion

Diversity, reputation and fiscal health in company profiles have been considered before choosing the stock data. Since we have considered stocks from corporations involved in information technology and enabled services, social media, electronics and instrumentation, manufacturing, and pharmaceuticals, the performance of the model accommodates variability. Besides, the model needs to be evaluated for its robustness. The *sign predictability* is related to the stocks' variance over time. Naturally, the gains or losses of stocks which are stable would be easier to predict than stocks that are relatively noisy. A lower variance in the data implies a better predictive capability of ML classifiers, and from an economic point of view, it accounts for the stability. The measures of performance that are used to evaluate the robustness of a binary classifier are accuracy, precision, recall (also known as sensitivity), specificity and the area under the curve (AUC) of the ROC curve. We have also included the Brier score for the classifiers (which is like the mean squared error)[1].

For demonstrating the efficacy of our approach, we present the results of two stocks: those of Apple and Facebook as a representative sample. The entire experiment was implemented on 10 different stocks, the results of which are elaborated in Appendix A.

4.1. Results of Random Forests

The results of classification of the samples from the dataset of stocks of Facebook and Apple are given in Table 2. In general, the accuracy increases as the width of the window is increased. Another important observation is that the F-score also increases with the increase in window-width. These results are presented in Table 2.

[1]For a complete description of all these measures of performance, refer to Section 1 of the Supplementary File

Table 2: Results of classification using random forests.

Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
AAPL	3	65.26	0.71	0.66	0.58	0.68	0.35	0.70
	5	72.55	0.78	0.73	0.67	0.75	0.28	0.79
	10	78.80	0.81	0.80	0.76	0.81	0.21	0.87
	15	82.01	0.85	0.82	0.78	0.84	0.17	0.90
	30	85.34	0.88	0.86	0.81	0.87	0.15	0.93
	60	90.44	0.93	0.90	0.86	0.92	0.10	0.96
	90	93.02	0.95	0.93	0.90	0.94	0.07	0.98
FB	3	67.59	0.72	0.69	0.62	0.71	0.30	0.71
	5	74.15	0.84	0.73	0.62	0.78	0.25	0.80
	10	81.39	0.90	0.81	0.69	0.85	0.19	0.89
	15	86.06	0.89	0.90	0.80	0.89	0.14	0.92
	30	89.89	0.95	0.90	0.80	0.92	0.11	0.94
	60	89.63	0.98	0.89	0.61	0.94	0.10	0.93
	90	94.76	0.98	0.96	0.72	0.97	0.06	0.94

Table 3: Results of classification using XGBoost.

Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
AAPL	3	55.99	0.80	0.56	0.29	0.66	0.44	0.57
	5	59.64	0.77	0.60	0.40	0.67	0.40	0.62
	10	61.50	0.79	0.61	0.41	0.69	0.39	0.66
	15	64.48	0.78	0.64	0.48	0.70	0.36	0.70
	30	68.82	0.85	0.68	0.47	0.76	0.31	0.76
	60	72.61	0.90	0.70	0.49	0.79	0.27	0.83
	90	77.13	0.88	0.77	0.61	0.82	0.23	0.86
FB	3	59.31	0.78	0.60	0.37	0.68	0.41	0.63
	5	65.05	0.79	0.65	0.48	0.71	0.35	0.72
	10	68.75	0.80	0.71	0.53	0.75	0.31	0.75
	15	80.84	0.86	0.85	0.71	0.86	0.19	0.86
	30	86.88	0.93	0.87	0.75	0.90	0.13	0.93
	60	88.76	0.98	0.89	0.57	0.93	0.11	0.89
	90	94.44	0.98	0.96	0.72	0.97	0.06	0.95

4.2. Results of XGBoost

The results of classification as well as the trends observed regarding the change in the classification accuracy and other metrics with the increase in the window-width in the case of XGBoost is similar to the trends observed in the case of random forests. Here, too, the classification accuracy and F-score increase with the increase in the window width. Moreover, the goodness of classification observed for a certain window-width in the case of XGBoost is comparable to the goodness of classification for the same window-width in the case of random forests. These results are presented in Table 3.

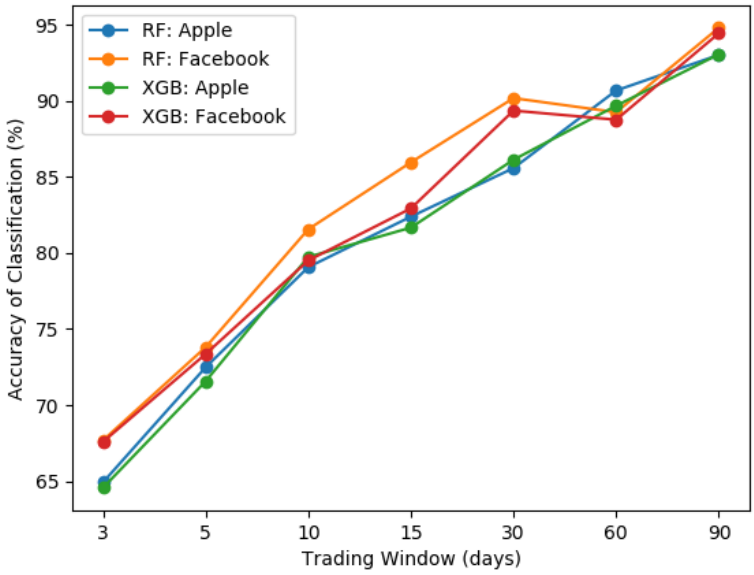


Figure 1: The trend of accuracy against the trading width considered. The accuracy of classification generally increases as the trading window increases for both random forests and XGBoost, used over the two datasets. The values are given in Tables 2 and 3.

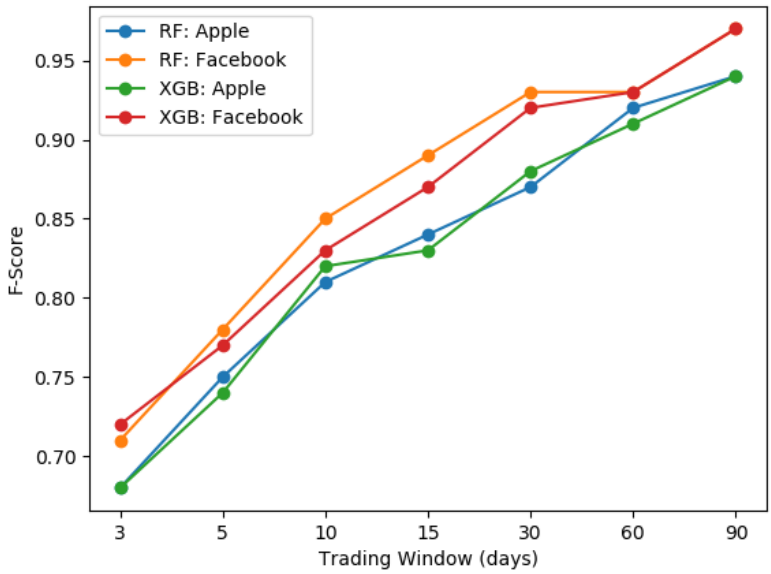


Figure 2: The trend of F-score against the trading width considered. The F-score increases as the trading window increases for both random forests and XGBoost, used over the two datasets. The values are given in Tables 2 and 3.

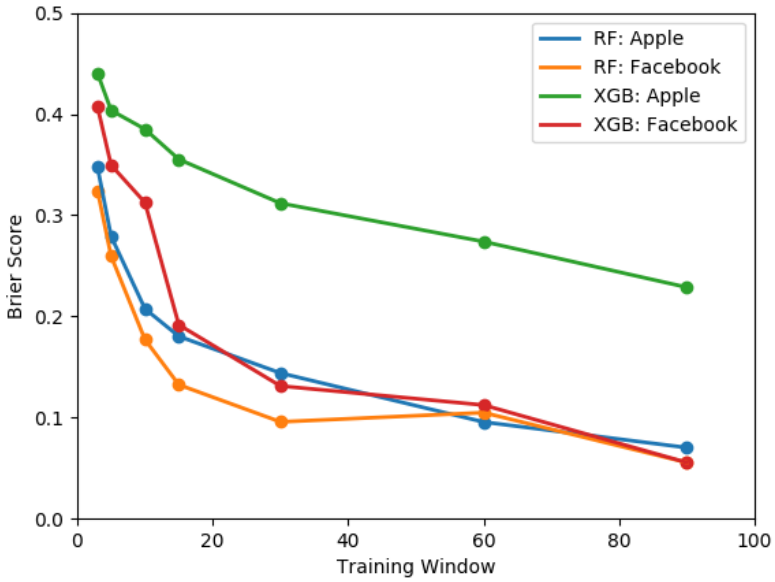


Figure 3: The trend of Brier score against the trading width considered. The Brier decreases as the trading window increases for both random forests and XGBoost, used over the two datasets. The values are given in Tables 2 and 3.

4.3. Comparison of Performance

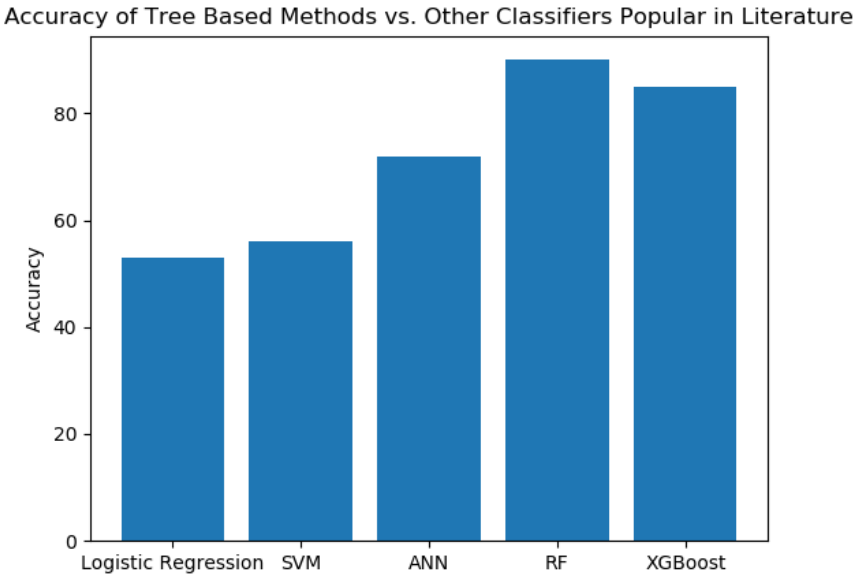


Figure 4: Comparison of the accuracy achieved with different classifiers. From the literature survey, it may be concluded that Logistic Regression performs poorly as compared to SVM (with a linear kernel), RF, and XGBoost. Also, from the bar graph, it is evident that SVM performs with an accuracy of less than 60% and RF and XGBoost outperforms SVM in terms of accuracy (close to 92% on an average, across different stocks). For a representative sample of ROC plots for SVM, refer to Section 7 of the supplementary file.

In this section, we compare the results of the classifiers previously used in literature (keeping the preprocessing steps the same) to compare the performance of random forests and GBDTs using XGBoost. We have implemented these classifiers and have reported the best-case average accuracy across different stocks for a performance baseline (please refer to Figure 4).

In the current work, we have used exponential smoothing to remove random local variation in the data. A detailed survey of the existing literature does not offer this as a preprocessing step to earlier predictions. The current methodology performs better than linear classifiers and the reason for this is the inherent non-linearity in the data. In (Di, 2014), the authors have used a linear classifier as the supervised learning algorithm which yielded a highest accuracy of 55.65%. Our learning model to surpass all these metric classifiers in terms of long term prediction. An important practical question that is not entirely solved, is the criteria for the selection of the kernel function parameters – for Gaussian kernels the width parameter σ – and the value of ϵ in the ϵ loss insensitive function. Overall, the outcome of random forests and XGBoost are better than the classifiers previously tried and this is illustrated in Figure 4. Consistent with our goal, this is a general result implying a trend of performance across various stocks and is not pertaining to the stock of only one corporation.

4.4. Out-of-Bag (OOB) Error Visualization

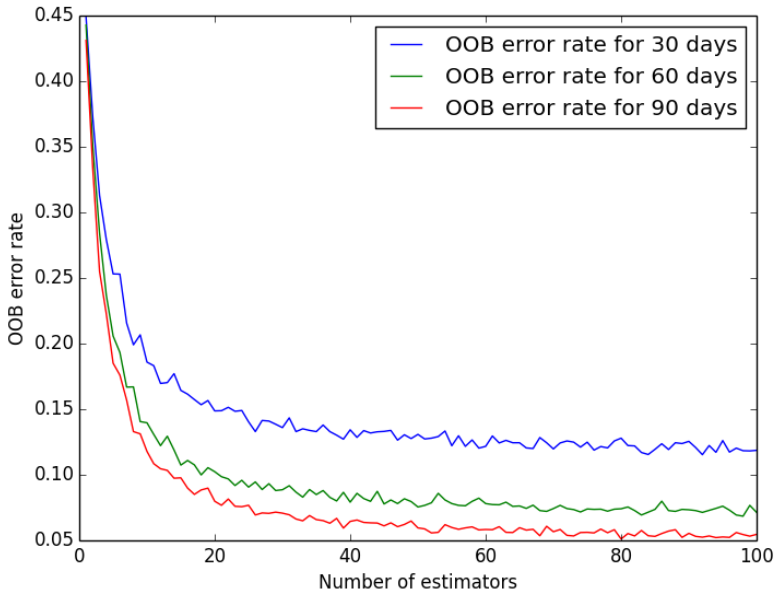


Figure 5: OOB error rate vs Number of estimators

After creating all the decision trees in the forest, for each training sample $Z_i = (X_i, Y_i)$ in the original training set T, we select all bagged sets T_k which does not contain Z_i. This set contains bootstrap datasets which do not contain a particular training sample from the original training dataset. These sets are called out of bags examples. There are n such sets for each n data samples in the original training dataset. OOB error is the average error for each Z_i calculated using predictions from the trees that do not contain it in their respective bootstrap sample. OOB error is an estimate of generalization error which measures how accurately the random forest predicts previously unseen data. We plotted the OOB error rate for our random forest classifier using the AAPL dataset.

From Figure 5, we can see that the OOB error rate decreases rapidly as more number of trees are added in the forest. However, a limiting value of the OOB error rate is reached eventually. The plot shows that the random forest converges as more number of trees are added in the forest. This result also explains why

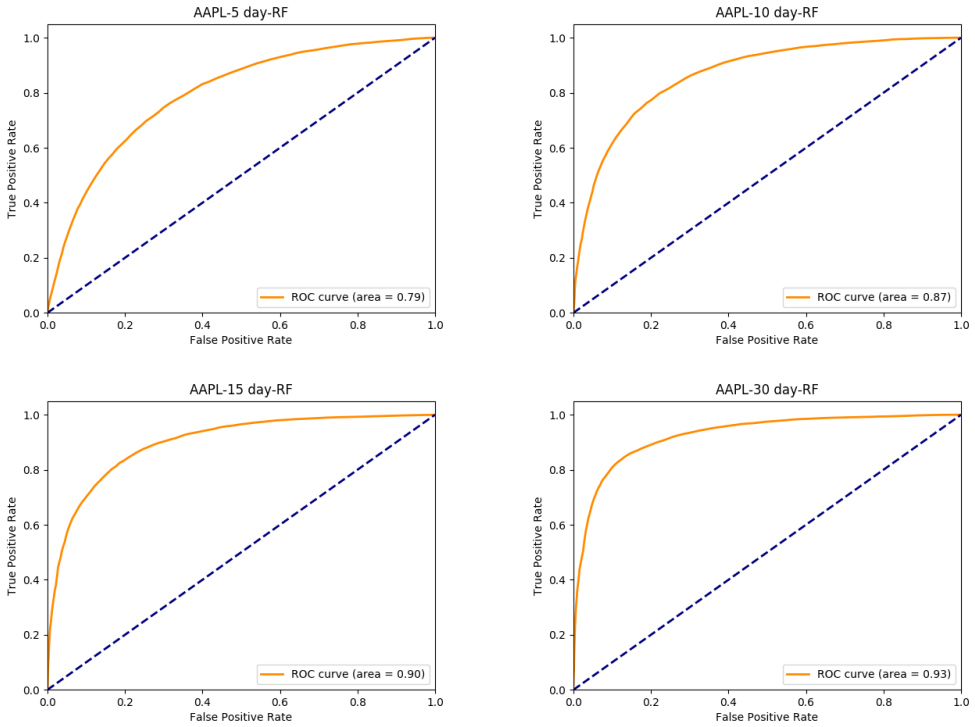


Figure 6: ROC curves plotted for random forests for 5, 10, 15, and 30 day trading windows. There is an improvement in performance with the increase in the trading window. For a more extensive list of ROC plots, refer to Section 5 of the Supplementary File.

random forests do not over-fit as more number of trees are added into the ensemble. A similar justification is for the efficacy of GBDTs.

4.5. Receiver Operating Characteristic Curves (ROC)

To analyze the effect of the duration of the trading window, we have plotted a small set of ROC curves in Figure 6. We observe that with an increase in the trading window, there is generally an improvement in classification accuracy. One reason why the accuracy improves with an increase in the value of t is that the economic indicators are able to capture more information regarding the movement of the prices over a larger time frame. This does not mean that the prices of a stock need to necessarily be increasing or decreasing, but just that with an adequate amount of information, an increase or decrease can be predicted accurately.

We have provided an exhaustive set of ROC plots in Section 5 of the Supplementary File.

4.6. Trading Indications: Inference From the Outcome of the Classifiers

Knowledge discovery from the analysis should create new frontiers or applications such as a trading strategy based on the strengths of the classification accuracy, investigating the behavior of certain classes of stocks. We achieved this as a derivative of the elaborate machine learning exercise.

In Figure 7, a subset of the trading suggestions by the random forest model is shown. The colored dots represent the trading decisions suggested by the model at data points with a 90-day time interval: the red dots suggest a drop in the prices and the blue dots suggest a rise in the prices in a time interval of every 90 days, and consequently, a suggestion of a purchase or sell. It can be seen from the graph that the model suggests to buy if it is predicted that price is going to rise after 90 days and the model suggests to sell if the price is predicted to fall after 90 days. It is an interesting question if the payoff function has a trend suggesting a trading model in time series. Notwithstanding, this is not an end-to-end trading strategy tool, but rather a component in a possibly more elaborate toolkit which can aid in investment decisions.

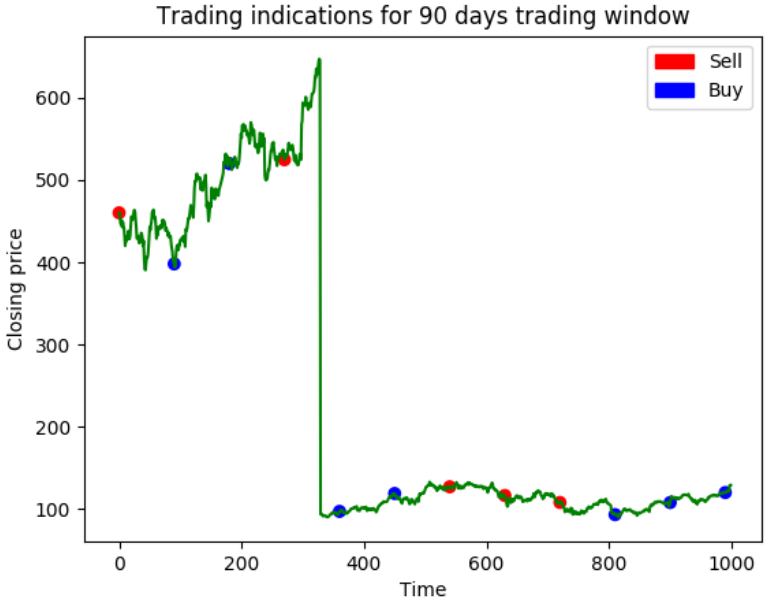


Figure 7: Trading suggestions made by the model on AAPL data. The X-axis is the n^{th} day in the dataset. The blue and red dots represent a small sample of trading suggestions made by the algorithm.

5. Conclusion

Application of machine learning techniques in stock price forecasting demands detailed execution, not only because of several technical complexities, but also because there are many deviations from best practice regulations in different countries. Consequently, analysis and forecasting of stock market activities and prices have often been country-specific or even region-specific allowing for variations in group behavior, culture, financial depth of the country and many other crucial determinants. The available approaches have made significant allowances for these variations and offered important results all along. However, there is room for improvement, where, the forecasting can be recast to bypass these drawbacks. In that sense, the proposed approach is a paradigm shift for available class of problems since it reformulates a traditional forecasting model as a classification problem. The present paper considers stock markets across places and industry-types and still offers a high accuracy for the predictive ability of the models developed - essentially in terms of whether stock prices are going to rise or fall over a period of time. The development of appropriate algorithms to represent big data through machine learning and subsequently, training these algorithms with real-life data involved considerable complexity in view of the well-known analytical and empirical contributions in this field. The paper has shown that the use of machine learning to understand the scope of big data is quite an advantage if models as developed here are applied to forecasting of the direction of stock prices and perhaps, for forecasting the degree to which stock prices change over time.

With reference to the directional prediction, the paper acknowledges that the high accuracy observed, could be a matter of concern. Natural suspicion about inherent bias in training data is common. However, the data set has been tested adequately in favor of non-existence of heavy bias. The absence of the effect of bias in the data has also been ascertained by examining the values of the F-scores in comparison to the accuracy (see Appendix A). Since the purpose of this paper has been to generalize and develop a method to predict with reasonable efficiency whether the price of a stock will increase or decrease on day $n+t$ as compared to day n, we argued that the mean accuracy is a more representative measure of the efficacy of the system as compared to the median returns. It has been discussed that in case of a positive drift in the prices of a stock it may be possible to have a better sign predictability, but that does not seem to have a conflict with our method of prediction. In addition, the data showed that most stock prices do not exhibit a strictly negative or a strictly positive drift. Therefore, the purpose of selecting stocks of companies like Nike,

Toyota, Facebook, Amazon, Apple, etc. was to demonstrate that regardless of the background or domain of a company, and regardless of specific fluctuations in the prices over time, the efficacy of our methods hold and these do not suffer from diminishing accuracies.

In this paper, we have used random forests and XGBoost classifiers, as two useful algorithms to build our predictive model, which produced impressive results. The model is found to be robust in predicting the direction of stock movements. The robustness of our model has been evaluated by calculating various parameters such as accuracy, precision, recall, specificity, and F-score. For all the datasets we have used, we were able to achieve high accuracies for long-term predictions. The comparative analysis testifies the efficacy of our model as it outperforms the models discussed in the literature survey. In addition to that, a novelty of the current work is about the selection of technical indicators and their applications as features. As the background of the problem that has been solved here is primarily that of financial analysis, the flexibility with the use of various features, each with it's own interpretation has been useful.

The paper discusses the merit of using random forests and XGBoost systems as compared to other techniques used under non-ensemble procedures. Indeed, it has been shown in several studies that trained SVM applied on Korean stock market (Kim, 2003), or DT plus ANN applied on Taiwan market (Tsai and Wong, 2009) reach 56% and 67% accuracy overall. Compared to this the present techniques attain 78% accuracy. In addition, the reason for choosing random forests over decision trees is because random forests uses a significant amount of voting-based conclusions as compared to that of decision trees. It runs a bagging based routine by using a large number of de-correlated decision trees (consider growing forests in random fashion) to classify a predicted class. This course of operation is highly suitable for the stock data and associated classification, as it meticulously examines the feature space to make better judgments over which class to finalize as the expected outcome.

Our model can be used for devising new strategies for trading or to perform stock portfolio management, changing stocks according to trend prediction. The proposed model is indeed a novel way to minimize the risk of investment in stock market by predicting the returns of a stock more accurately than existing algorithms applied so far. In future, we could build boosted tree models to predict trends for short time windows. Ensembles of different machine learning algorithms can also be checked for its robustness in stock prediction. We also recommend exploration of the application of deep learning practices in Stock Forecasting involving learning weight coefficients on large, directed, and layered graphs.

6. References

Appel, G. (2005). Technical Analysis Power Tools for Active Investors. ISBN:0-13-147902-4.

Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques – Part II: Soft computing methods.Expert Systems with Applications,36(3), 5932–5941.

Avery, C. N., Chevalier, J. A. & Zeckhauser, R. J. (2016). The CAPS Prediction System and Stock Market Returns. Review of Finance. 20 (4), 1363-1381.

Bao D. & Yang Z. (2008). Intelligent stock trading system by turning point confirming and probabilistic reasoning, Expert Systems with Applications, 34 (1), 620-627.

Beechey M., Gruen D. & Vickery J. (2000). The Efficient Market Hypothesis: A Survey, Research Discussion Paper. Economic Research Department. Reserve Bank of Australia.

Behner, P., Schwarting, D., Vallerien, S., Ehrhardt, M., Beever, C. & Rollmann, D. (2009). Pharmaceutical Companies in the Economic Storm Navigating from a Position of Strength. Technical Report: BooZ& Co Analysis.

Beltrametti, L, Fiorentini, R, Marengo, L and Tamborini, R (1997), A learning-to-forecast experiment on the foreign exchange market with a classifier system, Journal of Economic Dynamics and Control, Volume 21, Issues 8–9, 1543–1575.

Black, A. J., Klinkowska, O., McMillan, D. G. and McMillan, F. J. (2014), Predicting stock returns: Do commodities prices help?, Journal of Forecasting, 33, 627–639.

Boonpeng, S., & Jeatrakul, P. (2016). Decision Support System for Investing in Stock Market by using OAA-Neural Network. 8th International Conference on Advanced Computational Intelligence Chiang Mai, Thailand.

Breiman, L. (2001), Statistics Department, University of California Berkeley, CA 94720. Random Forests.

Bylander, T. & Hanzlik, D. (1999) Estimating Generalization Error Using Out-of-Bag Estimates. AAAI-99 Proceedings.

Cao, L. and Tay, F. E. (2001). Financial forecasting using support vector machines. Neural Computing & Applications, 10(2):184–192.

Chauhan, B., Umesh, B., Ajit, G., Sachin, K. (2014). Stock Market Prediction Using Artificial Neural Networks. International Journal of Computer Science and Information Technology. 5 (1), 904-907.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System, Proceedings of the 22^{nd} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD'16. doi:10.1145/2939672.2939785

Christoffersen, P. F., & Diebold, F. X. (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. Management Science,52(8), 1273-1287.

Cochrane, J. (2008), The dog that did not bark: A defense of return predictability. Review of Financial Studies, 21, 1533-1575.

Dai, Y. & Zhang Y. (2013). Machine Learning in Stock Price Trend Forecasting. Stanford University, http://cs229.stanford.edu/proj2013/DaiZhang-MachineLearningInStockPriceTrendForecasting.pdf

Das, S. P. & Sudersan, P., Support Vector Machines for Prediction of Future Prices in Indian Stock Market. International Journal of Computer Applications. 41 (3), 22-26, 2012.

Devi, K. N., Bhaskaran, V. M. & Kumar, G. P. (2015). Cuckoo Optimized SVM for Stock Market Prediction, IEEE Sponsored 2nd International Conference on Innovations in Information, Embedded and Communication systems (ICJJECS).

Di, X. (2014), Stock Trend Prediction With Technical Indicators using SVM. Stanford University.

Friedman, J. H. (2000). Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics. Vol. 29, p.1189–1232.

Gencay R. (1999). Linear, non-linear and essential foreign exchange rate prediction with simple technical trading rules. Journal of International Economics. Vol. 47, no.1, pp. 91-107.

Geurts, P., & Louppe, G. (2011). Learning to rank with extremely randomized tree. JMLR: Workshop and Conference Proceedings, 14, 49–61.

Giacomel ,F. , Galante, R. & Pareira, A. (2015). An Algorithmic Trading Agent based on a Neural Network Ensemble: a Case of Study in North American and Brazilian Stock Markets. IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology.

Granville, J. E. (1976). Granville's New Strategy of Daily Stock Market Timing for Maximum Profit. ISBN: 0-13-363432-9.

Halblieb, R. & Pohlmeier, W. (2012), Improving the value at risk forecasts: Theory and evidence from the financial crisis, Journal of Economic Dynamics and Control, Volume 36, Issue 8, 1212-1228.

Hellstrom, T. & Holmstromm, K. (1998). Predictable Patterns in Stock Returns. Technical Report Series IMA-TOM-1997-09.

Hunter, J. D. (2007) Matplotlib: A 2D graphics environment. Computing In Science & Engineering. 9(3), 90–95, DOI:10.1109/MCSE.2007.55.

Imandoust, S. B. & Bolandraftar, M., (2014). Forecasting the direction of stock market index movement using three data mining techniques: the case of Tehran Stock Exchange. International Journal of Engineering Research and Applications. 4(6), 106-117.

Ince, H. & Trafalis, T. B. (2007). Kernel principal component analysis and support vector machines for stock price prediction. IIE Transactions, 39(6):629–637.

Jensen M. C. (1978). Some Anomalous Evidence Regarding Market Efficiency. Journal of Financial Economics. 6 (2), 95-101.

Khaidem L., Saha S. & Roy Dey S. (2016). Predicting the direction of stock market prices using random forest. arXiv preprint arXiv:1605.00003.

Khan, W., Ghazanfar, M. A., Asam, M., Iqbal, A., Ahmed, S. & Khan, J. A. (2016). Predicting Trend In Stock Market Exchange Using Machine Learning Classifiers. Sci.Int, 28 (2), 1363-1367.

Khanal, A., R., Mishra, A., K. (2017). Stock price reactions to stock dividend announcements: A case from a sluggish economic period. The North American Journal of Economics and Finance. Volume 42, 338-345. doi.org/10.1016/j.najef.2017.08.002

Kim, Y., Jeong, S. R., & Ghani, I. (2014). Text opinion mining to analyze news for stock market prediction. Int. J. Advance. Soft Comput. Appl, 6(1).

Kimoto, T., Asakawa, K., Yoda, M., and Takeoka, M. (1990). Stock market prediction system with modular neural networks. In Neural Networks, 1990., 1990 IJCNN International Joint Conference on, pages 1–6. IEEE.

Kohara, K., Ishikawa, T., Fukuhara, Y., and Nakamura, Y. (1997). Stock price prediction using prior knowledge and neural networks. Intelligent systems in accounting, finance and management, 6(1):11–22.

Lane, G. (1984) "Lanes Stochastics. Second issue of Technical Analysis of Stocks and Commodities magazine. pp 87-90.

Ed. by Mark Larson (2015). Price Rate of Change: 12 Simple Technical Indicators: That Really Work, Wiley; https://doi.org/10.1002/9781119204428.ch5

Li, H., Yang, Z. & Li, T. (2014). Algorithmic Trading Strategy Based On Massive Data Mining. Stanford University. http://cs229.stanford.edu/proj2014/Haoming.

Lo, Andrew W., Mamaysky, H. & Wang, J. (2000). Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation, Journal of Finance, 55, 1705-1765.

Malkiel, B. G. (2003). The efficient market hypothesis and its critics. The Journal of Economic Perspectives. 17 (1), 59-82.

Malkiel, B. G. & Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. The Journal of Finance. 25 (2), 383-417, 1970.

Mayankkumar, S. R. Y. & Patel, B. (2014). Stock prediction using artificial neural network. International Journal of Innovative Research in Science, Engineering and Technology. 3 (6), 76-84.

Maymin, P. (2012). Music and the market: Song and stock volatility. The North American Journal of Economics and Finance 23 (1), 70-85. doi.org/10.1016/j.najef.2011.11.004

Mittermayer, M. A. (2004). Forecasting intraday stock price trends with text mining techniques. In System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on, pages 30064.2–. IEEE.

Moskowitz, T. J., Ooi, Y. H., & Pedersen, L. H. (2012). Time series momentum.Journal of financial economics, 104(2), 228-250.

Nelson, D. M. Q., Pereira, A. C. M. & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. International Joint Conference on Neural Networks (IJCNN), 1419-1426.

Nikfarjam, A., Emadzadeh, E., and Muthaiyah, S. (2010). Text mining approaches for stock market prediction. In Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on, volume 4, pages 256–260. IEEE.

Nyberg, H. (2011). Forecasting the direction of the US stock market with dynamic binary probit models.International Journal of Forecasting,27(2), 561-578.

Pai, P. F. and Lin, C. S. (2005). A hybrid arima and support vector machines model in stock price forecasting. Omega, 33(6):497–505.

Pedregosa F. et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830, 2011.

Recchioni, M, Tedeschi, G and Gallegati, M (2015), A calibration procedure for analyzing stock price dynamics in an agent-based framework, Journal of Economic Dynamics and Control, Volume 60, 1-25.

Qiu, M. & Song, U. (2016). Predicting the Direction of Stock Market Index Movement Using an Optimized Artificial Neural Network Model, PLOS ONE 11 (5).

Quinlan, J. R. (1986). Induction of Decision Trees, MACH. LEARN 1, pp 81–106

Saha, S., Routh, S. & Goswami, B. (2014). Modeling Vanilla Option prices: A simulation study by an implicit method. Journal of Advances in Mathematics. 6 (1), 834-848.

Sewell M. (2011). History of the Efficient Market Hypothesis. Research Note RN/11/04- UCL Dept. of Computer Science.

Shiller, Robert (2000), Irrational Exuberance, Princeton: Princeton University Press.

Timmermann, A. & Granger, C. W. (2004). Efficient market hypothesis and forecasting. International Journal of Forecasting. 20 (1), 15-27

Wang, J. H. & Leu, J. Y. (1996). Stock market trend prediction using arima-based neural networks. In Neural Networks, 1996., IEEE International Conference on, volume 4, pages 2160–2165. IEEE.

Widom, J. (1995). Research problems in data warehousing. In Proceedings of the fourth international conference on information and knowledge management. CIKM '95 (pp. 25- 30). New York, NY, USA: ACM. 10.1145/221270.221319.

Williams L. R. (1972). New Concepts in Technical Trading Systems. ISBN 978-0-89459-027-6.

Williams, L. R. (1978). The Secret of Selecting Stocks for Immediate and Substantial Gains. First Edition. ISBN 978-0930233051.

Xu, W, Chen, Y, Coleman, C and Coleman, T (2018), Moment matching machine learning methods for risk management of large variable annuity portfolios, Journal of Economic Dynamics and Control, Volume 87, 1-20.

Appendix A. Results

In this appendix, we elaborate the experimental results achieved by performing the experiments.

Table A.4: RF Results Table 1: results of random forests implemented on stocks of AAPL, AMS, AMZN, FB, and MSFT.

Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
AAPL	3	65.26	0.71	0.66	0.58	0.68	0.35	0.70
	5	72.55	0.78	0.73	0.67	0.75	0.28	0.79
	10	78.80	0.81	0.80	0.76	0.81	0.21	0.87
	15	82.01	0.85	0.82	0.78	0.84	0.17	0.90
	30	85.34	0.88	0.86	0.81	0.87	0.15	0.93
	60	90.44	0.93	0.90	0.86	0.92	0.10	0.96
	90	93.02	0.95	0.93	0.90	0.94	0.07	0.98
AMS	3	65.88	0.73	0.67	0.57	0.70	0.34	0.71
	5	68.80	0.74	0.68	0.63	0.71	0.31	0.76
	10	75.44	0.77	0.74	0.74	0.75	0.24	0.83
	15	79.60	0.79	0.78	0.80	0.78	0.21	0.87
	30	83.52	0.82	0.85	0.85	0.83	0.16	0.91
	60	89.24	0.90	0.89	0.89	0.89	0.11	0.95
	90	90.47	0.90	0.91	0.91	0.90	0.09	0.96
AMZN	3	67.09	0.75	0.66	0.58	0.70	0.33	0.72
	5	72.49	0.75	0.73	0.70	0.74	0.27	0.78
	10	77.43	0.81	0.77	0.74	0.79	0.23	0.86
	15	80.35	0.86	0.79	0.74	0.82	0.19	0.89
	30	86.48	0.92	0.86	0.79	0.89	0.13	0.93
	60	89.44	0.94	0.89	0.82	0.91	0.10	0.96
	90	94.79	0.96	0.96	0.93	0.96	0.05	0.98
FB	3	67.59	0.72	0.69	0.62	0.71	0.30	0.71
	5	74.15	0.84	0.73	0.62	0.78	0.25	0.80
	10	81.39	0.90	0.81	0.69	0.85	0.19	0.89
	15	86.06	0.89	0.90	0.80	0.89	0.14	0.92
	30	89.89	0.95	0.90	0.80	0.92	0.11	0.94
	60	89.63	0.98	0.89	0.61	0.94	0.10	0.93
	90	94.76	0.98	0.96	0.72	0.97	0.06	0.94
MSFT	3	67.30	0.72	0.67	0.62	0.70	0.33	0.73
	5	72.68	0.76	0.74	0.69	0.75	0.28	0.80
	10	77.39	0.85	0.76	0.68	0.80	0.23	0.85
	15	81.66	0.87	0.82	0.74	0.85	0.18	0.89
	30	86.15	0.88	0.88	0.84	0.88	0.14	0.93
	60	90.14	0.92	0.91	0.88	0.92	0.10	0.96
	90	92.24	0.94	0.93	0.90	0.93	0.08	0.97

Table A.5: RF Results Table 2: results of random forests implemented on stocks of NKE, SNE, TATA, TWTR, and TYO.

Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
NKE	3	66.22	0.72	0.68	0.59	0.70	0.33	0.72
	5	71.94	0.79	0.72	0.64	0.75	0.28	0.79
	10	76.01	0.83	0.76	0.67	0.79	0.25	0.84
	15	79.93	0.87	0.80	0.70	0.83	0.21	0.88
	30	85.76	0.93	0.85	0.74	0.89	0.15	0.93
	60	89.88	0.93	0.90	0.85	0.92	0.11	0.96
	90	93.31	0.96	0.93	0.88	0.95	0.06	0.98
SNE	3	63.53	0.67	0.63	0.60	0.65	0.37	0.69
	5	70.04	0.71	0.71	0.69	0.71	0.30	0.76
	10	76.90	0.79	0.77	0.75	0.78	0.23	0.85
	15	80.19	0.83	0.79	0.77	0.81	0.19	0.88
	30	83.45	0.84	0.84	0.83	0.84	0.17	0.91
	60	89.16	0.91	0.89	0.87	0.90	0.11	0.95
	90	89.75	0.92	0.89	0.88	0.90	0.11	0.96
TATA	3	67.29	0.62	0.66	0.72	0.64	0.32	0.74
	5	74.46	0.69	0.74	0.79	0.71	0.25	0.81
	10	79.78	0.73	0.81	0.86	0.76	0.21	0.87
	15	83.57	0.75	0.86	0.90	0.80	0.16	0.91
	30	87.64	0.83	0.89	0.92	0.86	0.13	0.94
	60	91.64	0.87	0.92	0.95	0.89	0.08	0.97
	90	95.44	0.94	0.94	0.96	0.94	0.04	0.99
TWTR	3	76.04	0.74	0.75	0.78	0.75	0.24	0.80
	5	75.53	0.78	0.69	0.74	0.73	0.24	0.83
	10	83.62	0.83	0.81	0.84	0.82	0.17	0.92
	15	85.82	0.89	0.79	0.83	0.84	0.14	0.93
	30	87.25	0.81	0.84	0.91	0.83	0.12	0.94
	60	87.82	0.78	0.83	0.93	0.80	0.12	0.94
	90	86.10	0.83	0.83	0.88	0.83	0.15	0.95
TYO	3	66.96	0.54	0.63	0.77	0.58	0.32	0.71
	5	73.70	0.62	0.66	0.81	0.64	0.26	0.79
	10	78.45	0.60	0.81	0.91	0.69	0.22	0.85
	15	84.91	0.74	0.84	0.92	0.79	0.15	0.91
	30	90.23	0.79	0.89	0.95	0.84	0.09	0.94
	60	92.02	0.70	0.95	0.99	0.81	0.08	0.98
	90	93.86	0.75	0.95	0.99	0.84	0.06	0.99

Table A.6: XGBoost Results Table 1: results of XGBoost implemented on stocks of AAPL, AMS, AMZN, FB, and MSFT.

Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
AAPL	3	55.99	0.80	0.56	0.29	0.66	0.44	0.57
	5	59.64	0.77	0.60	0.40	0.67	0.40	0.62
	10	61.50	0.79	0.61	0.41	0.69	0.39	0.66
	15	64.48	0.78	0.64	0.48	0.70	0.36	0.70
	30	68.82	0.85	0.68	0.47	0.76	0.31	0.76
	60	72.61	0.90	0.70	0.49	0.79	0.27	0.83
	90	77.13	0.88	0.77	0.61	0.82	0.23	0.86
AMS	3	58.99	0.78	0.59	0.36	0.67	0.41	0.61
	5	59.87	0.78	0.58	0.41	0.67	0.40	0.64
	10	62.44	0.73	0.60	0.52	0.66	0.38	0.69
	15	64.83	0.69	0.61	0.61	0.65	0.35	0.70
	30	66.81	0.65	0.68	0.69	0.66	0.33	0.74
	60	73.82	0.81	0.71	0.67	0.75	0.26	0.82
	90	77.03	0.79	0.76	0.75	0.77	0.23	0.85
AMZN	3	55.92	0.80	0.56	0.30	0.65	0.44	0.59
	5	58.10	0.82	0.57	0.32	0.67	0.42	0.63
	10	58.52	0.83	0.57	0.32	0.68	0.41	0.66
	15	62.63	0.88	0.60	0.33	0.72	0.37	0.72
	30	67.21	0.90	0.66	0.36	0.76	0.33	0.77
	60	76.22	0.86	0.77	0.62	0.81	0.24	0.84
	90	82.69	0.92	0.82	0.68	0.87	0.17	0.88
FB	3	59.31	0.78	0.60	0.37	0.68	0.41	0.63
	5	65.05	0.79	0.65	0.48	0.71	0.35	0.72
	10	68.75	0.80	0.71	0.53	0.75	0.31	0.75
	15	80.84	0.86	0.85	0.71	0.86	0.19	0.86
	30	86.88	0.93	0.87	0.75	0.90	0.13	0.93
	60	88.76	0.98	0.89	0.57	0.93	0.11	0.89
	90	94.44	0.98	0.96	0.72	0.97	0.06	0.95
MSFT	3	55.18	0.82	0.55	0.26	0.66	0.45	0.56
	5	56.55	0.86	0.56	0.22	0.68	0.43	0.60
	10	60.50	0.85	0.60	0.31	0.70	0.40	0.67
	15	64.60	0.86	0.64	0.35	0.74	0.35	0.70
	30	69.17	0.80	0.71	0.55	0.75	0.31	0.77
	60	74.04	0.83	0.75	0.61	0.79	0.26	0.83
	90	78.03	0.83	0.79	0.72	0.81	0.22	0.87

Table A.7: XGBoost Results Table 2: results of XGBoost implemented on stocks of NKE, SNE, TATA, TWTR, and TYO.

Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
NKE	3	56.53	0.84	0.57	0.23	0.68	0.43	0.58
	5	56.64	0.85	0.57	0.22	0.68	0.43	0.60
	10	60.01	0.92	0.59	0.21	0.72	0.40	0.67
	15	61.93	0.96	0.61	0.15	0.74	0.38	0.67
	30	68.50	0.98	0.67	0.21	0.79	0.31	0.73
	60	72.25	0.90	0.72	0.45	0.80	0.28	0.79
	90	78.90	0.94	0.77	0.55	0.84	0.21	0.87
SNE	3	53.72	0.69	0.53	0.38	0.60	0.46	0.57
	5	56.86	0.71	0.57	0.42	0.63	0.43	0.60
	10	60.16	0.74	0.59	0.46	0.66	0.40	0.66
	15	62.50	0.74	0.60	0.51	0.67	0.38	0.68
	30	66.10	0.64	0.69	0.68	0.66	0.34	0.72
	60	70.21	0.87	0.67	0.51	0.76	0.30	0.77
	90	70.50	0.84	0.67	0.56	0.74	0.30	0.80
TATA	3	55.80	0.40	0.53	0.69	0.46	0.44	0.59
	5	60.51	0.40	0.61	0.78	0.48	0.39	0.66
	10	68.56	0.44	0.76	0.88	0.56	0.31	0.74
	15	70.91	0.50	0.75	0.87	0.60	0.29	0.80
	30	74.11	0.58	0.79	0.88	0.67	0.26	0.85
	60	84.01	0.75	0.83	0.90	0.79	0.16	0.92
	90	87.22	0.81	0.86	0.91	0.83	0.13	0.94
TWTR	3	71.07	0.70	0.69	0.72	0.70	0.29	0.74
	5	73.60	0.75	0.67	0.72	0.71	0.26	0.81
	10	81.12	0.80	0.78	0.82	0.79	0.19	0.87
	15	84.02	0.86	0.78	0.82	0.82	0.16	0.91
	30	84.66	0.77	0.81	0.89	0.79	0.15	0.93
	60	85.06	0.73	0.79	0.91	0.76	0.15	0.93
	90	83.02	0.79	0.80	0.86	0.79	0.17	0.93
TYO	3	59.09	0.32	0.54	0.79	0.40	0.41	0.61
	5	66.12	0.39	0.57	0.82	0.46	0.34	0.67
	10	71.22	0.41	0.75	0.91	0.53	0.29	0.77
	15	78.38	0.54	0.83	0.93	0.65	0.22	0.86
	30	84.03	0.64	0.82	0.93	0.72	0.16	0.90
	60	88.72	0.56	0.94	0.99	0.70	0.11	0.91
	90	92.15	0.65	0.97	0.99	0.78	0.08	0.97

Predicting the Direction of Stock Market Price Using Tree-Based Classifiers: A Machine Learning Approach: *Supplementary File for On-line Publication Only*

1. Key Definitions

In this appendix, we introduce some key terms which have been used frequently through the paper. A basic understanding of the key concepts will bolster the reader's understanding of the methods used for predictive analysis.

1. **Data Structure**: In computer science, a *data structure* refers to the means of organizing and storing data for analysis and manipulation (Horowitz, Sahni & Anderson-Freed, 1992). Knowledge of data structures is fundamental for a computer scientist to judge how data should be stored for a particular application or task. Some popular data structures include stacks, queues, linked-lists, and trees.

2. **Node**: A node in a data structure is essentially an instance of that structure that contains data (Horowitz, Sahni & Anderson-Freed, 1992). A data structure is made of multiple nodes. It is the way in which nodes are connected in a data structure that defines it's characteristics.

3. **Tree Data Structure**: In a tree data structure, every node is the child of a single node (with the exception of the root node) and can have one or more children nodes (Horowitz, Sahni & Anderson-Freed, 1992). The name is inspired from how this data structure looks as subsequent levels of nodes *branch* or *spread out* like in a tree (Figure 1; notice how the subsequent levels of nodes branch out).

4. **Root Node**: In a tree data structure (henceforth referred to as just *tree*), the first node from which other nodes branch out is called the *root node* (Horowitz, Sahni & Anderson-Freed, 1992).

5. **Child Node**: A node in a tree can have one or more subordinate nodes or hierarchically lower nodes. These are called as child nodes of the respective node. In any tree, only the root node is not a child node.

6. **Level**: The level of a set of nodes in a tree is an index of how far away a node is from the root node. The level of the root node is 0. The level of the immediate children of the root node is 1. The level of the children nodes of the nodes at level 1 is 2, and so on.

7. **Parent Node**: If a node exists in a tree, it must be the subordinate of another node (with the exception of the root node) called it's parent node. A parent node of a child is the node that is immediately higher in hierarchy to the respective child node. The level of a node's parent is always one less than the level of the respective child.

8. **Leaf Node**: A leaf node is a node that does not have any children. All finite trees end with leaf nodes. In a classification tree, all leaf nodes need to be *pure*, i.e., they should have entities belonging to any one class only (exceptions to this may be incorporated by *pruning* a tree to prevent overfitting; pruning, in a general sense, is the removal of some leaf nodes with very few entities in them, before or after the tree has been built, so as to prevent overfitting) (Horowitz, Sahni & Anderson-Freed, 1992; Breiman, 2001).

9. **Probability Distribution**: $X = (X_1, ..., X_d)$ is an array of random variables defined on a probability space called as random vectors. The joint distribution of $X_1, ..., X_d$ is a measure on μ on R^d, $\mu(A) = P(X \in A)$, $A \in R^d$ where $d = 1, ..., m$. For example, Let $x = (x_i, ..., x_d)$ be an array of data points. Each feature x_i is defined as a random variable with some distribution. Then the random vector X has joint distribution identical to the data points, x.

10. **Classification Tree**: We define a classification tree (Quinlan, 1992) where each node is endowed with a binary decision: *whether* $x_i \leq k$ *or not*; where x_i is a feature in the data set, and k is some threshold. The topmost node in the classification tree contains all the data points and the set of data is subdivided among the children of each node as defined by the classification. The process of subdivision continues until pure leaf nodes are achieved. Each node is characterized by the feature x_i and threshold k chosen in such a way that minimizes diversity among the children nodes. This is often referred to as Gini impurity.

11. **Random Forest**: Let us represent $h_k(x) = h(x|\theta_k)$ implying decision tree k leading to a classifier $h_k(x)$. Thus, a random forest is a classifier based on a family of classifiers $h(x|\theta_1), ..., h(x|\theta_k)$, which is an ensemble of classification trees with model parameters θ_k randomly chosen from model random vector θ. Each classifier, $h_k(x) = h(x|\theta_k)$ is a predictor of the number of training samples. $y = \pm 1$ is the outcome associated with input data, x for the final classification function, $f(x)$, which can result by a majority voting mechanism of the individual classifiers (Breiman, 2001).

12. **Linear Separability**: Before feeding the training data to the random forest classifier, the two classes of data are tested for linear separability by finding their convex hulls. Linear Separability is a property of two sets of data points where the two sets are said to be linearly separable if there exists a hyperplane such that all the points in one set lies on one side of the hyperplane and all the points in other set lies on the other side of the hyperplane.

    Mathematically, two sets of points X_0 and X_1 in n dimensional Euclidean space are said to be linearly separable if there exists an n dimensional normal vector W of a hyperplane and a scalar k, such that every point $x \in X_0$ gives $W^T x > k$ and every point $x \in X_1$ gives $W^T x < k$. Two sets can be checked for linearly separability by constructing their convex hulls.

13. **ROC**: In statistics, a *receiver operating characteristic* (ROC) curve is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection[1] in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm[1] and can be calculated as (1− specificity). The ROC curve is thus the sensitivity as a function of fall-out. In general, if the probability distributions for both detection and false alarm are known, the ROC curve can be generated by plotting the cumulative distribution function (area under the probability distribution from -∞ to the discrimination threshold) of the detection probability in the y-axis versus the cumulative distribution function of the false-alarm probability in x-axis.

    There are four possible outcomes from a binary classifier. If the outcome from a prediction is positive (p) and the actual value is also p, then it is called a *true positive* (TP); however if the actual value is negative (n) then it is said to be a false positive (FP). Conversely, a true negative (TN) has occurred when both the prediction outcome and the actual value are n, and false negative (FN) is when the prediction outcome is n while the actual value is p. To draw an ROC curve, only the true positive rate (TPR) and false positive rate (FPR) are needed (as functions of some classifier parameter). The TPR defines how many correct positive results occur among all positive samples available during the test. FPR, on the other hand, defines how many incorrect positive results occur among all negative samples available during the test.

    An ROC space is defined by FPR and TPR as x and y axes respectively, which depicts relative trade-offs

between true positive (benefits) and false positive (costs). Since TPR is equivalent to sensitivity and FPR is equal to 1− specificity, the ROC graph is sometimes called the sensitivity vs (1− specificity) plot. Each prediction result or instance of a confusion matrix represents one point in the ROC space. The diagonal divides the ROC space. Points above the diagonal represent good classification results (better than random), points below the line represent poor results (worse than random).

14. **AUC**: The percentage area under the curve (AUC) value is the area covered by the ROC curve in the ROC space. It gives us a quantitative measure of the performance of a classifier, and a quantitative measure of the ROC at a glance.

15. **Brier Score**: The Brier score is the square of the difference between the predicted value or label of an input object and the real value or label. It is similar to the mean squared error and quantifies how aberrant the predictions of the classifier or regression function is.. For our work on classification, the Brier score is given as:

$$BS = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{1}$$

where y_i is the true class label of the i^{th} sample and $\hat{y}_i$ is the predicted label of the same sample, and N is the number of test samples.

2. Random Forest

2.1. The Shannon Entropy

Another function in addition to the Gini impurity which can be used to judge the quality of a split is the Shannon Entropy. It measures the disorder in the information content (in this context, it measures how mixed the population in a node is). The entropy in a node N can be calculated as follows:

$$S(N) = -P_1 log(P_1) - P_{-1} log(P_{-1}) \tag{2}$$

where d is number of classes considered and $P(\omega_i)$ is the proportion of the population labeled as i. Entropy is the highest when all the classes are contained in equal proportion in the node. It is the lowest when there is only one class present in a node (when the node is pure).

In most cases, the Gini impurity and the Shannon entropy can be used interchangeably.

2.2. Advantages of CART

Some of the main advantages of using random forests of CART are:

1. CART-based random forests are *non-metric*. This means that there are no inherent assumptions of distributions in data, and neither are there any parameters which need to be tweaked in order to obtain optimal performance.
2. Random Forests are recommended to be used with *bagging*, i.e., test data is sampled with replacement. Bagging ensures that with the increase in the number of tree estimators in a random forest, the chance of error decreases. A simple assertion of this is made (using the binomial distribution stuff) and is proved using Chebyshev's Inequality in (enter section number).
3. CART can handle categorical and continuous values naturally.
4. It is easy to understand and quick to fit, even if the data is cluttered and/or if the problem is inherently large.
5. The accuracy of random forests compares well with traditional methods in ML, such as Naïve Bayes', etc.
6. Random forests are *stable*. A slight change in the input data may affect individual trees, but the characteristics of the forest remains largely unchanged.

2.3. OOB error and Convergence of the Random Forest

Given an ensemble of decision trees $h_1(X), h_2(x), h_3(x), ..., h_k(x)$ as in (Breiman, 2001), we define margin function as:

$$mg(X,Y) = av_k I(h_k(X) = Y) - max_{j \neq Y} av_k I(h_k(X) = j) \tag{3}$$

where X, Y are randomly distributed vectors from which the training set is drawn. Here, $I(\cdot)$ is the indicator function. The generalization error is given by:

$$PE^* = P_{X,Y}(mg(X,Y) < 0) \tag{4}$$

The X and Y subscripts indicate that probability is calculated over X, Y space. In random forests, the k^{th} decision tree $h_k(x)$ can be represented as $h(x, \theta_k)$ where x is the input vector and θ_k is the bootstrapped dataset which is used to train the k^{th} tree. For a sequence of bootstrapped sample sets $\theta_1, \theta_2, ..., \theta_k$ which are generated from the original dataset θ, it is found that PE^* converges to:

$$P_{X,Y}(P_\theta(h(X,\theta) = Y) - max_{j \neq Y} P_\theta(h(X,) = j) < 0) \tag{5}$$

The proof can be found in Appendix I in (Breiman, 2001). To practically prove this theorem with respect to our dataset, the generalization error is estimated using out of bags estimates (Bylander & Hanzlik, 1999). The out of Bag (OOB) error measures the prediction error of Random forests algorithm and other machine learning algorithms which are based on Bootstrap aggregation.

Note: The average margin of the ensemble of classifiers is the extent to which the average vote count for the correct class flag exceeds the count for the next best class flag.

2.4. Random Forest as Ensembles: An Analytical Exploration

As defined earlier, a Random Forest model specifies θ as classification tree marker for $h(X|\theta)$ and a fixed probability distribution for θ for diversity determination in trees is known. The margin function of an RF is:

$$margin_{RF}(x,y) = P_\theta(h(x|\theta) = y) - max_{j \neq y} P_\theta(h(x|\theta) = j) \tag{6}$$

The strength of the forest is defined as the expected value of the margin:

$$s = E_{x,y}(margin_{RF}(x,y)) \tag{7}$$

The generalization error is bounded above by Chebyshev's inequality and is given as:

$$Error = P_{x,y}(margin_{RF}(x,y) < 0) \leq P_{x,y}(|margin_{RF}(x,y) - s| \geq s) \leq \frac{var(margin_{RF}(x,y))}{s^2} \tag{8}$$

Remark: We know that the average margin of the ensemble of classifiers is the extent to which the average vote count for the correct class flag exceeds the count for the next best class flag. The strength of the forest is the expected value of this margin. When the margin function gives a negative value, it means that an error has been made in classification. The generalization error is the probability that the margin is a negative value. Since margin itself is a random variable, Equation 8 shows that it is bounded above by its variance divided by the square of the threshold. As the strength of the forest grows, error in classification decreases.

We present below the Chebyshev's inequality as the inspiration for the error bound.

2.5. Chebyshev's Inequality

Let X be any random variable (not necessarily non-negative) and C > 0. Then,

$$P(|X - E(X)| \geq c) \leq \frac{var(x)}{c^2} \tag{9}$$

It is easy to relate the inequality to the error bound of the Random Forest learner.

2.6. Proof of Chebyshev's Inequality:

We require a few definitions before the formal proof.

A) Indicator Random Variable

$$I(X \geq c) = \begin{cases} 1 & \text{if } X \geq c \\ 0 & \text{Otherwise} \end{cases} \tag{10}$$

B) Measurable space

$$A = \{x \in \Omega | X(x) \geq c\} \tag{11}$$

$$E(X) = \sum_{x \in A} P(x)X(x) = \mu \tag{12}$$

Proof:

Define $A = \{x \in \Omega | X(x) - E(x) \geq c\}$

Thus,

$$\begin{aligned} var(X) &= \sum_{x \in \Omega} P(X = x)(X(x) - E(x))^2 \\ &= \sum_{x \in A} P(X = x)(X(x) - E(x))^2 + \sum_{x \notin A} P(X = x)(X(x) - E(x))^2 \geq 0 \\ &\geq \sum_{x \in A} P(x = x)(X(x) - E(x))^2 \\ &\geq P(X = x)c^2 \quad \text{since,} X(x) - E(x) \geq C; \forall x \in A \\ &= c^2 P(A) = c^2 P(|X - E(X)| \geq c) \\ \Rightarrow \frac{var(X)}{c^2} &\geq P(|X - E(X)| \geq c) \end{aligned} \tag{13}$$

Remark: This means that the probability of the deviation of a data point from its expected value being greater than c, a threshold, is bounded above by the variance of the data points divided by the square of the threshold, c. As c increases, the upper bound decreases which implies the probability of a large deviation of a data point from its expected value is less likely.

2.7. Algorithms

Algorithm 1: GiniGain

input : $(x_i, y_i)_1^n$ is the labeled training data
c_L is the left child node
c_R is the right child node

output: The Gini gain of the current split

$G_N \leftarrow$ Gini impurity of root node;
$G_L \leftarrow$ Gini impurity of left child;
$G_R \leftarrow$ Gini impurity of right child;
$L_n \leftarrow$ number of samples in c_L;
$R_n \leftarrow$ number of samples in c_R;
$P_L \leftarrow L_n/n$;
$P_R \leftarrow R_n/n$;
return $G_N - (P_L \times G_L) - (P_R \times G_R)$;

Algorithm 2: DecisionTree

input : $X = (x_i, y_i)_1^n$ is the labeled training data
l is the current level of the tree
M is the set of features used to grow a tree

output: A tree which is configured to predict the class label of a test sample

---

$l \leftarrow l + 1$;
$C_L \leftarrow$ null;
$C_R \leftarrow$ null;
; `/* ` C_L, C_R `are the left and right children of this node respectively */`
$MaxGain \leftarrow 0$;
; `/* ` $MaxGain$ `stores the value of the maximum possible gain that can be achieved from splitting a node - based on this, the final split is determined */`
for j *in* M **do**
  Sort $(x_i, y_i)_1^n$ in the increasing order of the j^{th} feature
  for $i := 1$ *to* n **do**
    $c_L \leftarrow X[0 : i]$;
    $c_R \leftarrow X[i + 1 : n]$;
    $Gain \leftarrow$ **GiniGain**$((x_i, y_i)_1^n[j], c_L, c_R)$;
    if $Gain > MaxGain$ **then**
      $MaxGain \leftarrow Gain$;
      $C_L, C_R \leftarrow c_L, c_R$;
  end
end
if C_L ***does not satisfy the desired level of purity*** **then**
  DecisionTree(C_L, l, M);
if C_R ***does not satisfy the desired level of purity*** **then**
  DecisionTree(C_R, l, M);
; `/* A node is not pure when it has samples belonging to more than one class. In such a case, it needs to be split in turn. In this way, through successive function calls, the feature space is recursively partitioned. */`

3. eXtreme Gradient Boosting (XGBoost)

3.1. Gradient Boosted Decision Trees: An Analytic Exploration

The tree approximation by aggregating many functions is done by *additive learning*. Each node is hence built sequentially, with each successive approximated function trying to better classify the residuals of the

previous learner. XGBoost employs an additive strategy wherein every subsequent approximated function optimizes an objective function. Hence, a series of functions are built such that the node is ultimately approximated using an aggregate of all the functions and is gradually optimized. The objective function may be represented as:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^t) + \sum_{k=1}^{t} \Omega(f_k) \tag{14}$$

where l is a loss function and Ω is the regularization term, which is used to measure the complexity of the model. Ω is of the form: $\Omega(f) = \gamma T + \lambda \|w\|^2$, thus making use of an L2 regularization.

Additive learning over t iterations for the i^{th} tree happens as follows:

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
\hat{y}_i^{(3)} &= f_1(x_i) + f_2(x_i) + f_3(x_i) = \hat{y}_i^{(2)} + f_3(x_i) \\
&\dots \\
\hat{y}_i^{(t)} &= f_1(x_i) + ... + f_n(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned} \tag{15}$$

And hence, the expression:

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{16}$$

Hence, the objective function may be expanded as:

$$\begin{aligned}
\mathcal{L}^{(t)} &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{t} \Omega(f_k) \\
&= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constants
\end{aligned} \tag{17}$$

The constants arise because the regularization till step $t-1$ is considered to be a constant at step t. Hence, Equation 17 represents the loss function in it's general form. This loss function can be approximated using the Taylor Series approximation till the n^{th} order term. The approximation till the 2^{nd} order is considered. The Taylor Series expansion is done as follows.

Let us consider a function $F(u) = l(X(u), Y(u)) = l(X + u\Delta X, Y + u\Delta Y)$. At $u = 1$, the function becomes: $F(1) = l(X + \Delta X, Y + \Delta Y)$. Hence, in the current context, at $u = 1$, the following considerations can be made: $X \equiv y_i$, $\Delta X = 0$, $Y \equiv \hat{y}_i^{(t-1)}$, $\Delta Y \equiv f_t(x_i)$. Applying the chain rule, we can find the n^{th} order derivatives of $F(u)$. The expression for the first order derivative is derived in Equation 18.

$$\begin{aligned}
&\frac{d}{du}\big[F(u)\big] = \frac{\partial}{\partial X}\big[l(X(u), Y(u))\big]\frac{d}{du}(X(u)) + \frac{\partial}{\partial Y}\big[l(X(u), Y(u))\big]\frac{d}{du}(Y(u)) \\
\Rightarrow &\frac{d}{du}\big[F(u)\big] = \frac{\partial}{\partial X}\big[l(X(u), Y(u))\big]\Delta X + \frac{\partial}{\partial Y}\big[l(X(u), Y(u))\big]\Delta Y \\
\Rightarrow &\frac{d}{du}\big[F(u)\big] = \frac{\partial}{\partial X}\big[l(X(u), Y(u))\big] \cdot 0 + \frac{\partial}{\partial Y}\big[l(X(u), Y(u))\big] \cdot f_t(x_i) \\
\Rightarrow &\frac{d}{du}\big[F(u)\big] = \frac{\partial}{\partial \hat{y}_i^{(t-1)}} l(\hat{y}_i, \hat{y}_i^{(t-1)}) \cdot f_t(x_i) \\
&\Rightarrow F'(u) = g_i f_t(x_i)
\end{aligned} \tag{18}$$

where

$$g_i = \frac{\partial}{\partial \hat{y}_i^{(t-1)}} l(\hat{y}_i, \hat{y}_i^{(t-1)})$$

The expression for the second order derivative is derived in Equation 19:

$$\begin{aligned}
&\frac{d^2}{du^2}\big[F(u)\big] = \left\{ \frac{\partial}{\partial X}\Big[\frac{\partial}{\partial Y}\big[l(X(u),Y(u))\big]\Big] + \frac{\partial}{\partial Y}\Big[\frac{\partial}{\partial Y}\big[l(X(u),Y(u))\big]\Big] \right\} \cdot \frac{d}{du}(Y(u)) \\
\Rightarrow &\frac{d^2}{du^2}\big[F(u)\big] = \left\{ \frac{\partial^2}{\partial X \partial Y}\big[l(X(u),Y(u))\big] \cdot \frac{d}{du}(X(u)) + \frac{\partial^2}{\partial Y^2}\big[l(X(u),Y(u))\big] \cdot \frac{d}{du}(Y(u)) \right\} \cdot \frac{d}{du}(Y(u)) \\
\Rightarrow &\frac{d^2}{du^2}\big[F(u)\big] = \left\{ \frac{\partial^2}{\partial X \partial Y}\big[l(X(u),Y(u))\big] \cdot \Delta X + \frac{\partial^2}{\partial Y^2}\big[l(X(u),Y(u))\big] \cdot \Delta Y \right\} \cdot \frac{d}{du}(Y(u)) \\
\Rightarrow &\frac{d^2}{du^2}\big[F(u)\big] = \left\{ \frac{\partial^2}{\partial X \partial Y}\big[l(X(u),Y(u))\big] \cdot 0 + \frac{\partial^2}{\partial Y^2}\big[l(X(u),Y(u))\big] \cdot \Delta Y \right\} \cdot \Delta Y \\
\Rightarrow &\frac{d^2}{du^2}\big[F(u)\big] = \left\{ \frac{\partial^2}{\partial Y^2}\big[l(X(u),Y(u))\big] \cdot \Delta Y \right\} \cdot \Delta Y \\
\Rightarrow &\frac{d^2}{du^2}\big[F(u)\big] = \frac{\partial^2}{\partial \big(\hat{y}_i^{(t-1)}\big)^2} l(y_i, \hat{y}_i^{(t-1)}) \cdot \Delta Y \cdot \Delta Y \\
\Rightarrow &\frac{d^2}{du^2}\big[F(u)\big] = \frac{\partial^2}{\partial \big(\hat{y}_i^{(t-1)}\big)^2} l(y_i, \hat{y}_i^{(t-1)}) \cdot f_t^2(x_i) \\
&\Rightarrow F''(u) = h_i f_t^2(x_i)
\end{aligned} \tag{19}$$

where

$$h_i = \frac{\partial^2}{\partial \big(\hat{y}_i^{(t-1)}\big)^2} l(y_i, \hat{y}_i^{(t-1)})$$

The Taylor series approximation till the second order is then given as:

$$\begin{aligned}
l\big[y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\big] &= F(0) + F'(0) + \frac{1}{2}F''(0) \\
&= l(\hat{y}_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)
\end{aligned} \tag{20}$$

Thus the objective function from Equation 14 can be approximated as:

$$\begin{aligned}
\mathcal{L}^{(t)} &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{t} \Omega(f_k) \\
&= \sum_{i=1}^{n} \big[l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i))\big] + \Omega(f_t) + constants \\
&= \sum_{i=1}^{n} \big[l(\hat{y}_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\big] + \Omega(f_t) + constants
\end{aligned} \tag{21}$$

As constants are not required in optimization, all the constant terms (note that the term $l(\hat{y}_i, \hat{y}_i^{(t-1)})$ can be considered to be constant in the t^{th} iteration) may be removed, thus consolidating the objective function further to:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{22}$$

The optimization, hence, only depends on g_i and h_i. XGBoost can support any loss function by taking g_i and h_i as input. These are known as the *gradient statistics* of the structure.

Each tree $f(x)$ is defined as:

$$f_t(x) = w_{q(x)}, w \in R^T, q : R^d \to 1, 2, ..., T \tag{23}$$

Here, w is the vector of scores on leaves q is a function assigning each data point to the corresponding leaf T is the number of leaves.

The regularization function Ω is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \tag{24}$$

Here, γ is the minimum gain a branch needs to contribute to the entire learner in order to be added to the overall structure. The idea of γ becomes clearer later in the derivation. There are multiple ways of defining the regularization function, but the one used here is an L2 regularization.

From equations (previous two), the objective value with the t^{th} iteration is as follows:

$$\begin{aligned} Obj^{(t)} &\approx \sum_{i=1}^{n} \left[g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2 \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T} \left[\Big(\sum_{i \in I_j} g_i \Big) w_j + \frac{1}{2} \Big(\sum_{i \in I_j} h_i + \lambda \Big) w_j^2 \right] + \gamma T \end{aligned} \tag{25}$$

where $I_j = i | q(x_i) = j$ is the set of indices of data points assigned to the j^{th} leaf, and λ is the *learning rate*. If $G_j = \sum\limits_{i \in I_j} g_i$ and $H_j = \sum\limits_{i \in I_j} h_i$, then Equation 25 becomes:

$$Obj^{(t)} = \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2} \big(H_j + \lambda \big) w_j^2 \right] + \gamma T \tag{26}$$

The part of Equation 39 within the summation is in a quadratic form. The optimal condition of a second degree objective function is at the point where the first derivative is zero. Hence, substituting the value of w_j at the point where the slope of the objective function is zero gives the optimal value of the objective function.

$$\begin{aligned} &\frac{d}{dw_j} \left[G_j w_j^* + \frac{1}{2} \big(H_j + \lambda \big) (w_j^2)^* \right] = 0 \\ &\Rightarrow G_j + 2 \cdot \frac{1}{2} \big(H_j + \lambda \big) w_j^* = 0 \\ &\Rightarrow \big(H_j + \lambda \big) w_j^* = -G_j \\ &\Rightarrow w_j^* = \frac{-G_j}{\big(H_j + \lambda \big)} \end{aligned} \tag{27}$$

Thus, substituting w_j^* in the objective function, we get the optimal value of the objective (from Equation 39):

$$
\begin{aligned}
Obj^* &= \sum_{j=1}^{T} \left[G_j \cdot \left(\frac{-G_j}{H_j + \lambda} \right) + \frac{1}{2}(H_j + \lambda) \left(\frac{-G_j}{H_j + \lambda} \right)^2 \right] + \gamma T \\
&= \sum_{j=1}^{T} \left[\frac{-G_j^2}{H_j + \lambda} + \frac{1}{2} \cdot \frac{G_j^2}{H_j + \lambda} \right] + \gamma T \\
&= -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T
\end{aligned} \tag{28}
$$

Now we have an objective function which can tell us about the goodness of a tree. But how do we decide what is the best split? Analogous to the GiniGain measure in the case of random forests, the Gain in XGBoost is used to split a node. The form of the gain function is similar to GiniGain; the best split at a node results in the best Gain.

$$
Gain = -\frac{1}{2} \cdot [Gain_L + Gain_R + Gain_{Root}] - \gamma \tag{29}
$$

The loss function commonly used in XGBoost is the squared loss function, which is given by:

$$
\mathcal{L} = \left[y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)) \right]^2 \tag{30}
$$

For which the gradient scores are calculated as:

$$
\begin{aligned}
g_i &= \frac{\partial}{\partial \hat{y}_i^{(t-1)}} \left[y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)) \right]^2 \\
&= 2 \cdot \left[y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)) \right] \cdot \left[0 - (1 + 0) \right] \\
&= -2 \left[y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)) \right] \\
\Rightarrow h_i &= \frac{\partial}{\partial \hat{y}_i^{(t-1)}} \Big\{ -2 \left[y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)) \right] \Big\} \\
&= -2 \left[0 - (1 + 0) \right] \\
&= 2
\end{aligned} \tag{31}
$$

The algorithm of GBDTs are explained in Algorithms 3 and 4.

3.2. Algorithms

Algorithm 3: GradientBoostedTree

input : $(x_i, y_i)_1^n$ is the labeled training data
γ is the minimum required structure score
L is the maximum number of levels in the tree
l is the current level of the tree
λ is the learning rate
N is the number of training steps

output: A tree which is configured to predict the class label of a test sample

---

$l \leftarrow l + 1$;
if $l \leq L$ **then**
  $t \leftarrow 0$;
  $f \leftarrow 0$;
  while $t < N$ **do**
    estimate f_t as a regressor function, density, or distribution;
    $f \leftarrow f + f_i$;
  end
  initialize array of scores $S[:]$;
  for $j := 1$ *to* n **do**
    $G_j \leftarrow 0$;
    $H_j \leftarrow 0$;
    for $i := 1$ *to* N **do**
      $G_j \leftarrow G_j + g_{ji}$;
      $H_j \leftarrow H_j + h_{ji}$;
      /* g_{ji} and h_{ji} are the gradient statistics of the j^{th} sample for the i^{th} function */
    end
    $S[j] \leftarrow -0.5 \cdot G_j^2 \div (H_j + \lambda)$;
  end
  $C_L, C_R \leftarrow$ **MaxGain**$((x_i, y_i)_1^n, S)$; // C_L, C_R are the left and right children of this node respectively
  if C_L ***does not satisfy the desired level of purity*** **then**
    GradientBoostedTree$(C_L, \gamma, L, l, \lambda, N)$;
  if C_L ***does not satisfy the desired level of purity*** **then**
    GradientBoostedTree$(C_R, \gamma, L, l, \lambda, N)$;

Algorithm 4: MaxGain

input : $(x_i, y_i)_1^m$ is the labeled training data
$S[:]$ is the list of structure scores

output: Left and right child nodes, ensuring maximum purity after split

---

Sort $(x_i, y_i)_1^m$ and $S[:]$ in the increasing order of $S[:]$;
$Gain_{split} \leftarrow -\infty$ /* $Gain_{split}$ stores the value of the gain after each split */
$Gain_N \leftarrow \sum S[:]$ /* the overall score of the parent node */
$SplitPoint \leftarrow -1$ /* stores the best split point */
$f \leftarrow 0$;
for $j := 1$ *to* $m - 1$ **do**
  $N_L \leftarrow S[0 : j]$ /* list of scores of left child */
  $N_R \leftarrow S[j + 1 : m]$ /* list of scores of right child */
  $G_L \leftarrow \sum N_L$ /* overall gain of left child */
  $G_R \leftarrow \sum N_R$ /* overall gain of right child */
  $Gain \leftarrow G_L + G_R - G_N$ /* gain calculated in the j^{th} iteration */
  /* the best gain is selected iteratively */
  if $Gain > Gain_{split}$ ***and*** $Gain > \gamma$ **then**
    $Gain_{split} \leftarrow Gain$;
    $SplitPoint \leftarrow j$;
end
return $(x_i, y_i)_1^j, (x_i, y_i)_{j+1}^m$;

4. Exposition of Random Forests and XGBoost

4.1. Random Forest

4.1.1. Decision Trees: Background

Decision trees (Geurts & Louppe, 2014; Breiman, 2001) can be used for various machine learning applications. But trees that are grown really deep to learn highly irregular patterns tend to over-fit the training sets. Noise in the data may cause the tree to grow in a completely unexpected manner. Random Forests overcome this problem by training multiple decision trees on different subspaces of the feature space at the cost of slightly increased bias. This means that none of the trees in the forest sees the entire training data. The data is recursively split into partitions. At a particular node, the split is done by asking a question on an attribute. The choice for the splitting criterion is based on some impurity measures such as Gini impurity or Shannon Entropy.[1]

Gini impurity is used as the function to measure the quality of split in each node. Gini impurity at node N is given by:

$$G(N) = 1 - (P_1)^2 - (P_{-1})^2 \tag{32}$$

where P_i is the proportion of the population with class label i. The obvious heuristic approach to choose the best splitting decision at a node is the one that reduces the impurity as much as possible. In order words, the best split is characterized by the highest gain in information or the highest reduction in impurity. The information gain due to a split can be calculated as follows:

$$Gain = I(N) - p_L I(N_L) - p_R I(N_R) \tag{33}$$

where $I(N)$ is the impurity measure (Gini, or any other) of node N, $I(N_L)$ is the impurity in the left child of node N after the split and similarly, $I(N_R)$ is is the impurity in the right child of node N after the split; N_L and N_R are the left and right children of N respectively' p_L and p_R are the proportions of the samples in the left and right children nodes with respect to the parent node.

Note that Equations 2, 32, and 33 can be used when there are only two splits at each node. However, depending on the algorithm used, there may be more than two splits and hence these formulae may be generalized as:

$$\begin{aligned} G(N) &= 1 - \sum_{j=1}^{n} P_j \\ Gain &= I(N) - \sum_{j=1}^{n} p_j I(N_j) \end{aligned} \tag{34}$$

In our work, we have used the CART (Classification and Regression Trees) algorithm, which splits every node into only two children nodes.

Bootstrap aggregating, also known as *bagging*, is an ensemble learning method that improves the stability and accuracy of learning algorithms while reducing variance and over-fitting, which are common problems while constructing decision trees. Given a sample dataset D of shape $n \times m$, bagging generates B new sets of shape $n' \times m'$ by sampling uniformly from D with replacement, where $n < n'$ and $m < m'$.

4.1.2. A Relevant Example Based on the Data

In random forests, decision tree learners are constructed by randomly selecting m out of M features and n out of N samples. Here, we illustrate the working of random forests by randomly considering 20 samples from the data set as the training set and 5 samples as the test set; the training and test sets are mutually exclusive. They are shown in Tables 1 and 2 respectively. Let us take $m = \sqrt{M}$, so that in each tree constructed, $\sqrt{M}$ features are randomly considered. In the data set, since there are 6 features, $\sqrt{M} = \sqrt{6} \approx 2.45$, hence, in each tree, 2 features will be considered. We shall take the subsample size to be equal to the sample size, i.e., 20 in each tree constructed here.

[1]Preprint is available in Arxiv and ResearhGate. Plagiarism similarity score may be misleading.

S.Id	Class Label	RSI	SCH	W%R	MACD	PROC60	OBV
1	-1	91.02	74.0	-54.07	1.12	0.03	259336600.0
2	1	11.09	17.12	-35.42	-0.8	-0.05	23149700.0
3	1	75.95	77.76	-54.6	0.64	-0.02	263929400.0
4	-1	88.25	85.8	-50.08	1.02	0.01	369776000.0
5	-1	16.71	14.19	-35.08	-0.67	0.04	1434446700.0
6	1	9.55	11.48	-35.28	-1.5	-0.02	1264042100.0
7	1	66.21	77.38	-48.78	-0.23	-0.02	198021100.0
8	-1	67.82	77.98	-50.25	0.36	0.02	276163600.0
9	-1	71.88	49.71	-50.82	0.58	0.08	560679400.0
10	1	4.66	11.6	-44.17	-1.56	-0.06	135643900.0
11	1	1.52	16.01	45.27	-0.33	-0.13	-2263609600.0
12	-1	17.41	24.08	-44.97	-0.12	0.02	-3417917300.0
13	1	22.27	31.57	-40.62	-0.22	-0.03	-3265559400.0
14	-1	70.79	77.25	-45.93	0.09	0.02	-2846701500.0
15	1	77.74	84.71	-72.5	0.36	-0.08	-3516514200.0
16	-1	57.5	73.5	-44.46	-0.12	0.09	-2926120100.0
17	1	42.81	54.22	-52.63	-0.26	-0.02	-3213555400.0
18	1	61.83	72.97	-66.66	-0.09	-0.1	-3904360800.0
19	-1	18.86	16.0	-36.27	-0.13	0.03	-3645587700.0
20	-1	63.08	52.02	-50.11	-0.03	0.02	-2182828500.0

Table 1: Sample Training Set

S.Id	Class Label	RSI	SCH	W%R	MACD	PROC60	OBV
1	-1	24.97	15.54	-43.74	-0.57	0.15	-1373395100.0
2	-1	85.06	75.77	-59.75	0.37	0.08	-1711049700.0
3	-1	86.88	70.44	-66.76	0.78	0.03	-1910292300.0
4	1	0.69	13.22	-36.48	-1.35	-0.19	-2573216900.0
5	1	78.9	80.95	-47.66	1.71	-0.01	-2371874000.0

Table 2: Sample Test Set

S.Id	Class Label	RSI	SCH
11.0	1	1.52	16.01
10.0	1	4.66	11.6
6.0	1	9.55	11.48
2.0	1	11.09	17.12
5.0	-1	16.71	14.19
12.0	-1	17.41	24.08
19.0	-1	18.86	16.0
13.0	1	22.27	31.57
17.0	1	42.81	54.22
16.0	-1	57.5	73.5
18.0	1	61.83	72.97
20.0	-1	63.08	52.02
7.0	1	66.21	77.38
8.0	-1	67.82	77.98
14.0	-1	70.79	77.25
9.0	-1	71.88	49.71
3.0	1	75.95	77.76
15.0	1	77.74	84.71
4.0	-1	88.25	85.8
1.0	-1	91.02	74.0

Table 3: Training set sorted in increasing order of RSI.

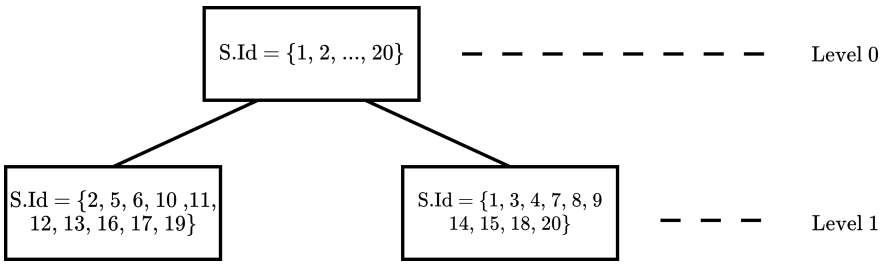


Figure 1: Tree creation by arbitrarily using the first 10 samples from Table 3 as the left child and the last 10 samples as the right child

The best *split* for a node in a decision tree is determined by sorting the samples in a node with respect to a feature (considered by the respective tree) and then partitioning it into two parts: the left and right children of that node. The partitioning is done by selecting the best threshold (hence, the sorting) for the corresponding feature and calculating the Gini gain (insert equation number). For example, let us consider a tree constructed by using the first two features in the training set, RSI and Stochastic Oscillator (SCH), the training set as seen by the decision tree, after being sorted based on RSI, appears as shown in Table 3.

The root node of the decision tree will perform the first partition of the data in Table 3. Hence, the constituents of the root node of the tree (or any tree) will have all the samples in the training set with 2 features selected at random. Let us consider arbitrarily the first 10 samples in Table 3 to constitute the *left child* and the last 10 samples to constitute the *right child* of the root node. The tree, after just the first split, will appear as shown in Figure 1.

Here, the impurity of the parent node (in this case, at level 0; the root node) is 0.5. This is easily understandable as there are 10 samples belonging to Class 1 and 10 samples belonging to Class -1. So the Gini Impurity is calculated as shown in Equation 35.

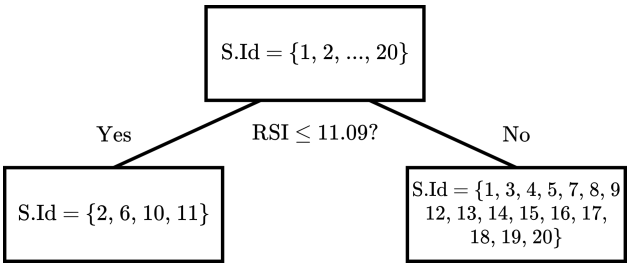


Figure 2: The best split for the root node when considering RSI and SCH.

$$\begin{aligned} G(N) &= 1 - (P_1)^2 - (P_{-1})^2 \\ &= 1 - \left(\frac{10}{20}\right)^2 - \left(\frac{10}{20}\right)^2 \\ &= 1 - 0.25 - 0.25 \\ &= 0.5 \end{aligned} \tag{35}$$

However, as the quality of the split depends on the *GiniGain*, the Gini impurities of the left and right children also need to be calculated. They are calculated in a manner similar to Equation 35.

$$\begin{aligned} G(N_L) &= 0.48 \\ G(N_R) &= 0.48 \end{aligned} \tag{36}$$

Note that the left child has 6 samples belonging to Class 1 and 4 samples belonging to Class -1, and the right child has 6 samples belonging to Class -1 and 4 samples belonging to Class 1. Hence, in this case, $G(N_L)$ and $G(N_R)$ are equal. It is not always necessary for them to be equal. The *Gain*, is calculated as in Equation 37.

$$Gain = 0.5 - \frac{10}{20} \times 0.48 - \frac{10}{20} \times 0.48 = 0.02 \tag{37}$$

This is how a node in a tree is split. At the next level of the tree, in each child node, the best split is determined again in a similar way. But now the partitions of the training sample space belonging to the children nodes are considered for partitioning each respective child node. This kind of partitioning is done recursively till *pure* nodes are reached. In decision trees, all leaf nodes need to be pure nodes (a leaf node is a node that is not split further; see 1 for more definitions and details). A node is said to be *pure* if all the samples in that node belong to the same class such that it does not require any further splitting. When an element needs to be classified, through a series of partitioning criteria, it reaches a leaf node and is assigned the class label of all the elements in the respective leaf node.

Returning to the illustration, it should be noted that a 50:50 split on the root node, with the first 10 samples from Table 3 constituting the left child node and the last 10 samples constituting the right child node is not the best split for that node. In fact, the best split is when the left child node has the first 4 samples and the right child node has the last 16 samples, such that the split as shown in Figure 2. For this split, the *GiniGain* is 0.125, which is greater than that for the split shown in Figure 1.

Now, how is this number 11.09, as the splitting threshold, determined? Looking at Table 3, we see that it is the value of the RSI attribute of the fourth sample. Reiterating, the samples in this table are sorted in the increasing order of RSI. As the partition till the 4^{th} sample for the left child node and the last 16 samples for the right child node results in the best GiniGain, the RSI value of the last sample comprising the left partition is used as a boundary value, or the split value (notice how the first four samples all belong to Class 1: this should help develop the readers' intuition on the working of *GiniGain* for determining the best split).

S.Id	RSI	SCH	Class Label	Predicted Label
1	24.97	15.54	-1	1
2	85.06	75.77	-1	-1
3	86.88	70.44	-1	-1
4	0.69	13.22	1	1
5	78.9	80.95	1	-1

Table 4: Results of classification the sample test set using **Tree 1**

Thus, the split point for the root node becomes $(RSI, 11.09)$. When an element is being classified using this tree, it essentially needs to reach a leaf node by going through a series of splits. Hence, the first split will be done using $(RSI, 11.09)$: if the value of RSI is ≤ 11.09) for an element to be classified, it will traverse the left branch of the root node (which, in this case, is a pure node), and if the value of RSI > 11.09, it will traverse the right branch, and consequently, a series of more splits, till it reaches a leaf node. The complete tree, built recursively such that all leaf nodes are pure nodes, is shown in Figure 3.

The geometric interpretation of a decision tree is that of partitions in the feature space corresponding to each class. Using the thresholds determined using the *Gain* for each split, the feature space can be considered to be divided into partitions or *pockets*, with each partition corresponding to one class. The partitions of the feature space for Tree 1 are shown in Figure 4. In this graph, all the partitions in blue color correspond to Class 1, and all the partitions in red color correspond to Class 2. On this graph, if we were to plot each sample in the test set (Table 2) by considering only (RSI, SCH), then the *color* of the partition a sample belonged to would indicate which class the respective sample is a part of (as the color of the partition represents the Class label)! Thus, random forests can be used to effectively handle non-linear trends in data or separability of data.

Now that a tree is constructed using the training set, the obvious task at hand is to classify the samples in the test set using this tree. Reiterating, in order to classify a test sample, we need to trace a test sample down the tree until a leaf node is reached. As leaf nodes are pure, the class of the leaf node will be assigned to the test sample being classified.

From Table 4, we can see that **Tree T1** misclassifies 2 out of 5 test samples, samples 1 and 5. Hence, it can be said that **T1** has a classification accuracy of 60%. Generally, a 60% accuracy for a classifier is not considered to be good enough. So in order to have a complete understanding of how a random forest might work, we shall construct two more trees and consider the majority vote for each test sample.

Let us construct the next tree in the forest by considering the features W%R and SCH. Notice that we are re-sampling the values of SCH. This is the principle of bootstrap aggregation, more commonly known as *bagging*: to sample data with replacement. Bagging has certain advantages. While growing a tree, a feature may or may not be chosen to be a feature using which the tree grows. This is especially relevant for high-dimensional datasets. Since each feature has a certain probability of success or failure of being chosen when a tree is being grown, it can be said that the probability of a feature being used in a random forest depends on the number of trees being constructed, with the probability being *binomially distributed*.

In the dataset used for the current work, as we have already mentioned, there are 6 features and for each tree being constructed, 2 features are considered. This implies that the probability of *success* (success being the condition that a feature is chosen to grow a decision tree) is $P(success) = 2/6 = 1/3 = 0.33$. The probability of *failure* (failure being the condition that a feature is not chosen to grow a decision tree) is $P(failure) = 1 - P(success) = 0.67$. Let us represent the probability of success as p and the probability of failure as q. Then the probability that a feature x_a is used in growing k out of n trees is given by Equation 38.

$$P(k \ successes) = \binom{n}{k} p^k q^{n-k} \tag{38}$$

It is easy to see that the LHS of Equation 38 increases as the value of n increases. This can be used to understand the idea of *stability* of a random forest: that upon increasing the number of trees in a random

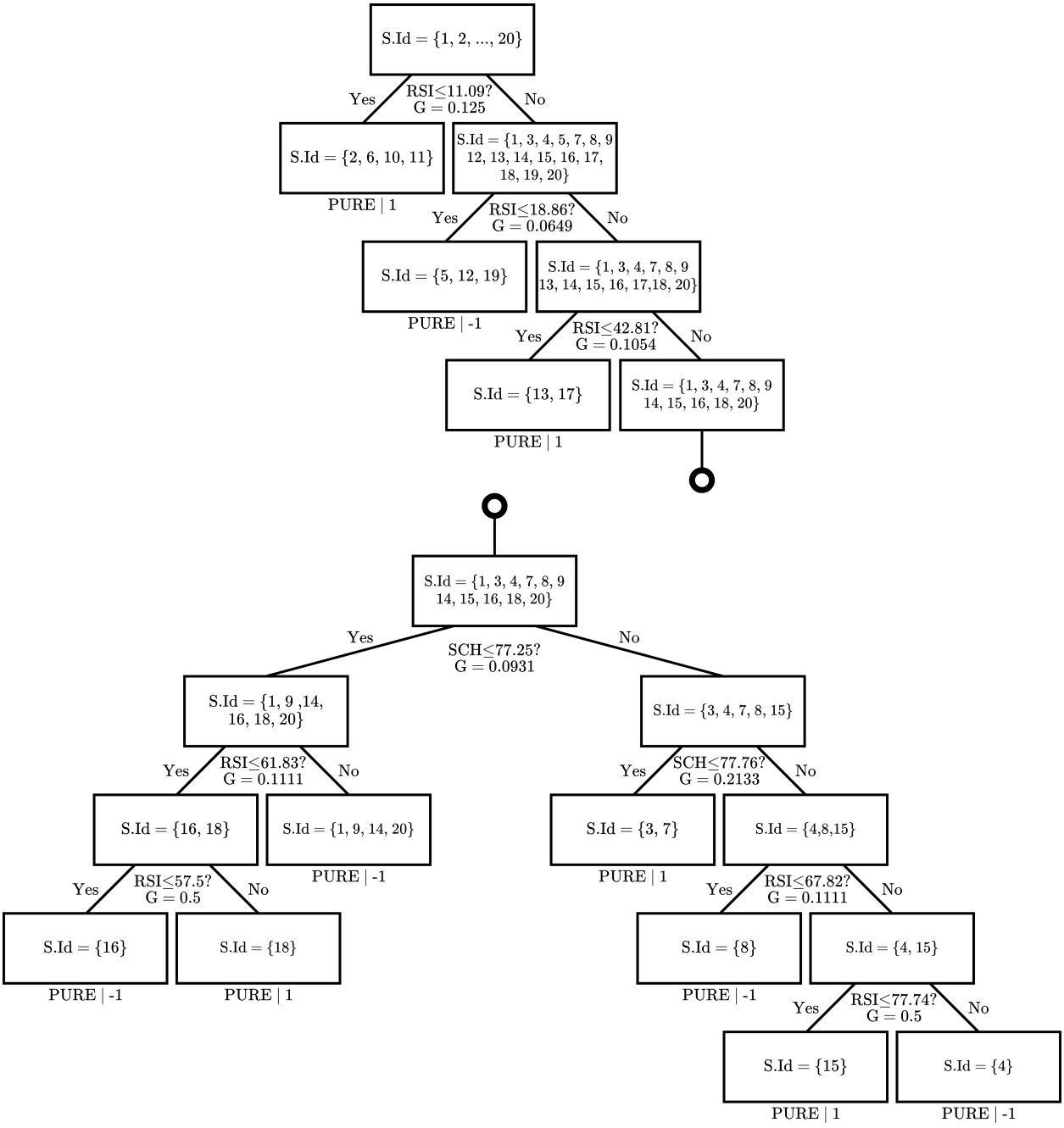


Figure 3: **Tree 1**: Random tree constructed considering RSI and SCH

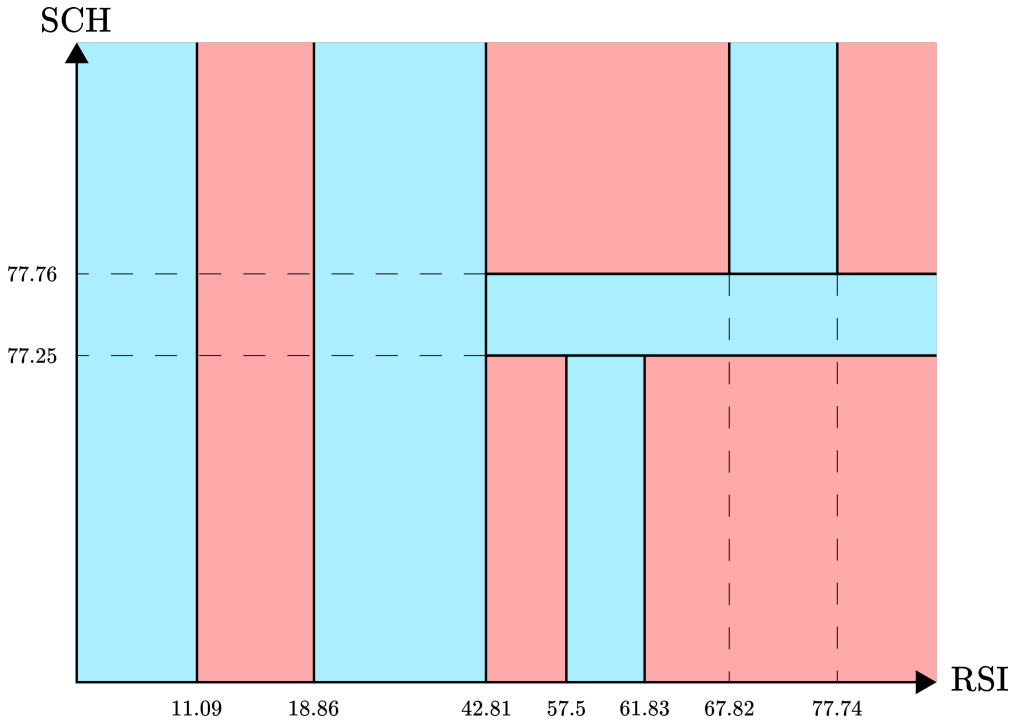


Figure 4: Partitions of sample space of Tree 1 (graph not drawn to scale)

S.Id	W%R	SCH	Class Label	Predicted Label
1	-43.74	15.54	-1	-1
2	-59.75	75.77	-1	1
3	-66.76	70.44	-1	1
4	-36.48	13.22	1	-1
5	-47.66	80.95	1	1

Table 5: Results of classification the sample test set using **Tree 2**

forest, the expected results are stabilized by continuous re-sampling of data so that the forest as a whole is representative of the class to which a sample to be classified belongs.

From Table 5, we can see that 2 out of 5 samples are classified correctly. This performance, as an individual tree, is not good. Upon comparing the predicted results from **Tree 1** and **Tree 2**, we see that the results from both trees combined are accurate only 50% of the time. The system as a whole is not accurate enough to predict the class of a previously unobserved sample accurately. To move a step closer towards an accurate system, let us construct another tree. We shall construct **Tree 3** using SCH and PROC60.

The decision tree constructed using SCH and PROC60 is a rather interesting one: there's only one split in the tree and the children of the root nodes are pure! The results of classification using just this tree is given in Table 6. This single tree gives us a 100% classification accuracy! The question arises: *why not use just this tree to classify the sample?*. The reasons go back to the notions of stability: that if there's some change in the data, some of the trees might not perform well, but the results of the forest should largely remain unchanged. In general, a forest classifier can be used to reduce the effect of muddy data. As this is an illustrative example, we are dealing with only 20 samples. However, the trend in millions of samples may not be as easy to deal with a single threshold. Besides, more recent innovations in tree based classifiers (such as AdaBoost, XGBoost) can discern or decide how to grow trees based on the performance of previously grown trees in the forest.

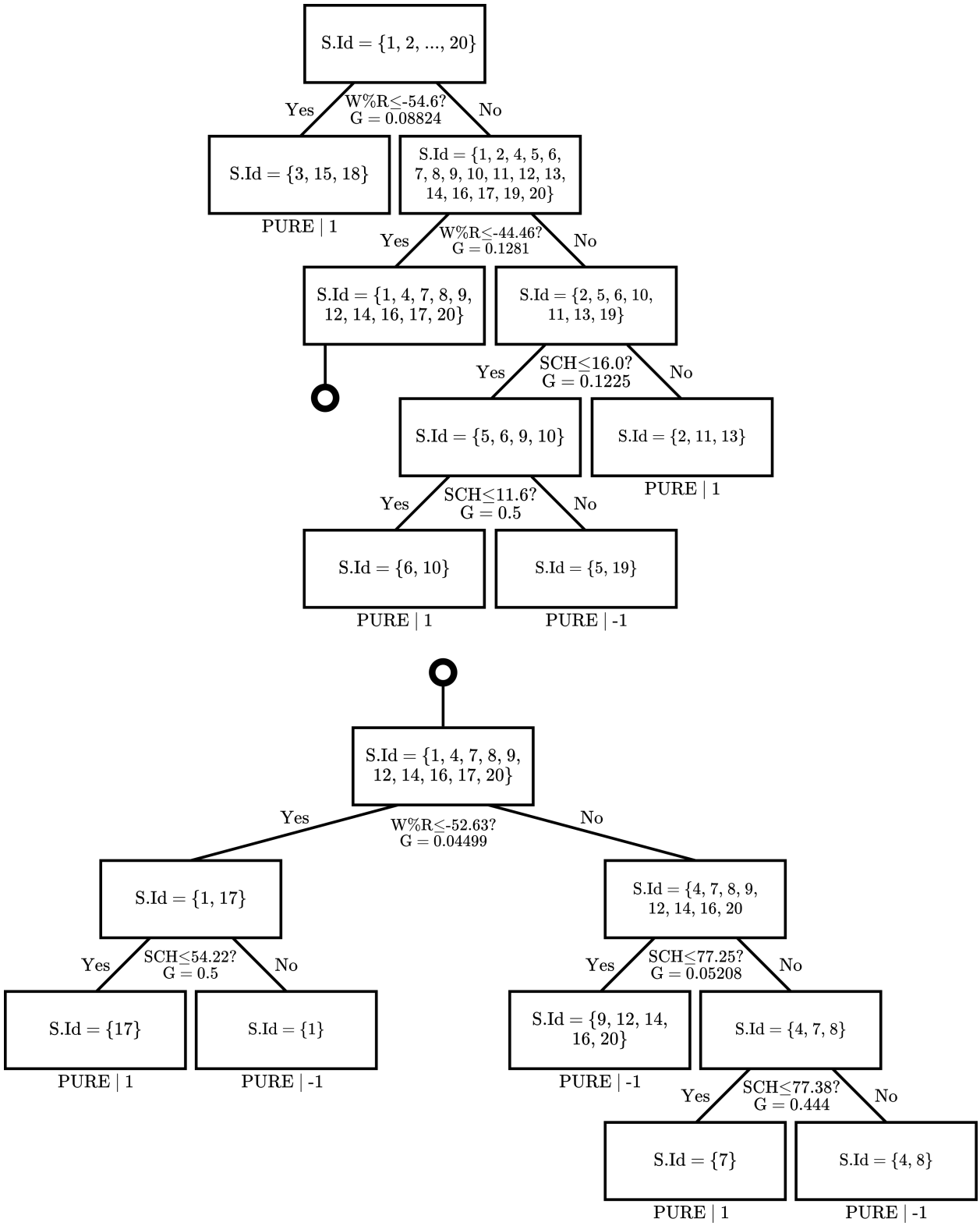


Figure 5: **Tree 2**: Random tree constructed considering W%R and SCH

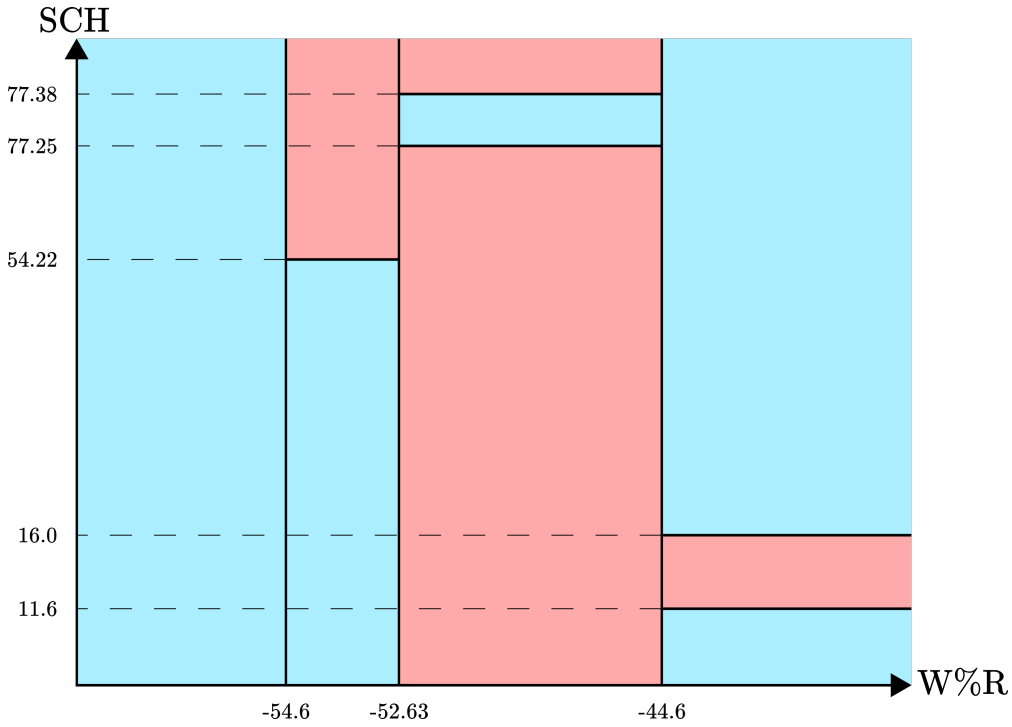


Figure 6: Partitions of sample space of Tree 2 (graph not drawn to scale)

Now that there are three trees in our forest, we can perform a majority voting. The result of the entire tree using **Tree 1**, **Tree 2** and **Tree 3** is given in Table 7. By combining the performance of all the trees in the forest, the results are perfect, and also stable.

The algorithm used to build decision trees is described in Algorithms 2 and 1.

4.1.3. Out-of-Bag (OOB) Error Visualization

After creating all the decision trees in the forest, for each training sample $Z_i = (X_i, Y_i)$ in the original training set T, we select all bagged sets T_k which does not contain Z_i. This set contains bootstrap datasets which do not contain a particular training sample from the original training dataset. These sets are called out of bags examples. There are n such sets for each n data samples in the original training dataset. OOB error is the average error for each Z_i calculated using predictions from the trees that do not contain z_i in their respective bootstrap sample. OOB error is an estimate of generalization error which measures how accurately the random forest predicts previously unseen data. We plotted the OOB error rate for our random forest classifier using the AAPL dataset.

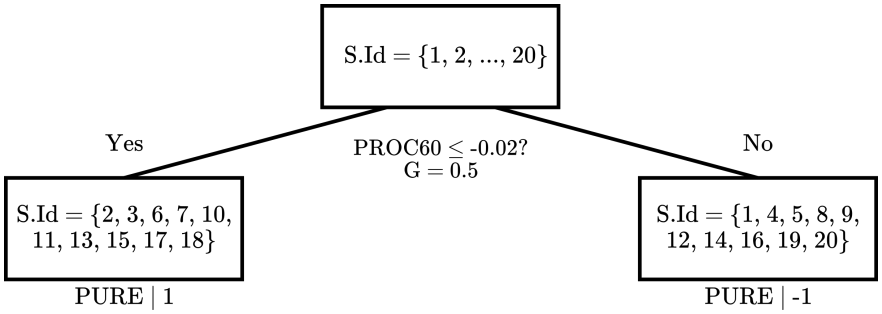


Figure 7: **Tree 3**: Random tree constructed considering SCH and PROC60

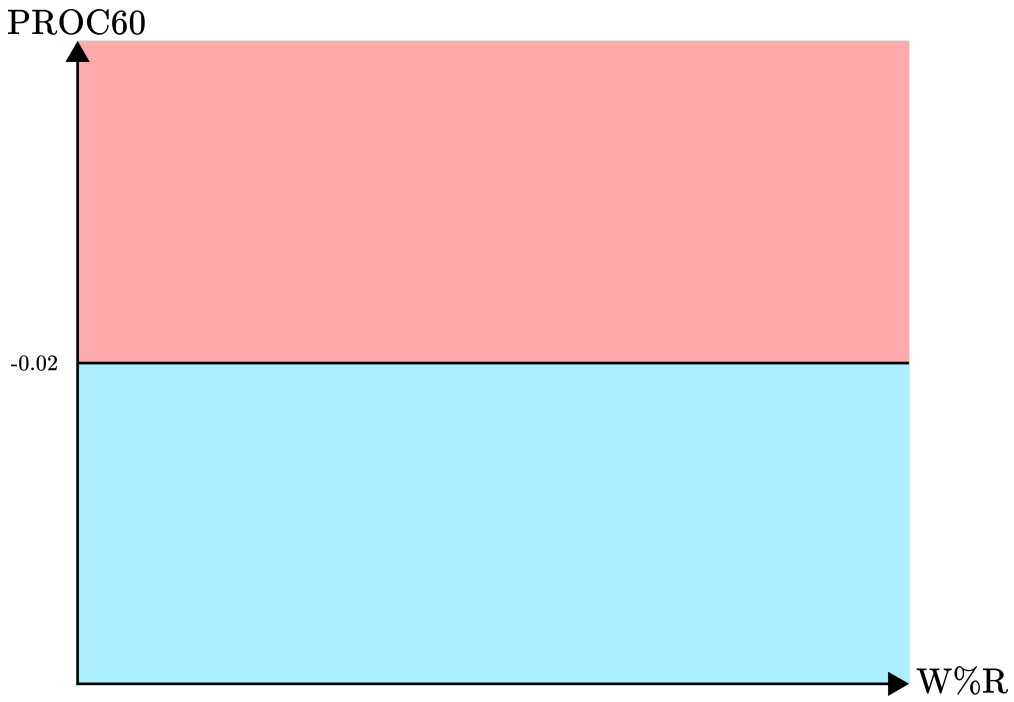


Figure 8: Partitions of sample space of Tree 3 (graph not drawn to scale)

S.Id	SCH	PROC60	Class Label	Predicted Label
1	15.54	0.15	-1	-1
2	75.77	0.18	-1	-1
3	70.44	0.03	-1	-1
4	13.22	-0.19	1	1
5	80.95	-0.10	1	1

Table 6: Results of classification the sample test set using **Tree 3**

S.Id	Class Label	Tree 1	Tree 2	Tree 3	Predicted Label
1	-1	1	-1	-1	-1
2	-1	-1	1	-1	-1
3	-1	-1	1	-1	-1
4	1	1	-1	1	1
5	1	-1	1	1	1

Table 7: Results of classification of the entire forest using majority voting

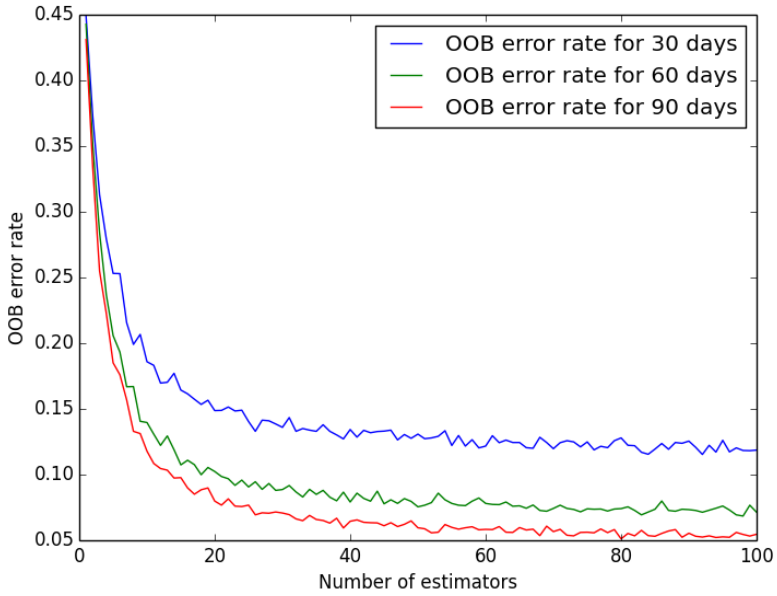


Figure 9: OOB error rate vs Number of estimators

From Figure 9 OOB plot, we can see that the OOB error rate decreases dramatically as more number of trees are added in the forest. However, a limiting value of the OOB error rate is reached eventually. The plot shows that the Random Forest converges as more number of trees are added in the forest. This result also explains why random forests do not over-fit as more number of trees are added into the ensemble.

$L(y, \gamma)$ is the differentiable loss function and $h_m(x)$ is the base learner, connected by the following relation: $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$.

4.2. eXtreme Gradient Boosting (XGBoost)

4.2.1. Gradient Boosted Decision Trees: A Convex Optimization Approach to Tree Classifiers

The tree approximation in gradient boosted trees is done by aggregating many functions by *additive learning*. Each node is hence built sequentially, with each successive approximated function trying to better classify the residuals of the previous learner. XGBoost employs an additive strategy wherein every subsequent approximated function optimizes an objective function. Hence, a series of functions are built such that the node is ultimately approximated using an aggregate of all the functions and is gradually optimized. The objective function may be represented as:

$$Obj^{(t)} = \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2}\big(H_j + \lambda\big) w_j^2 \right] + \gamma T \tag{39}$$

where $I_j = i|q(x_i) = j$ is the set of indices of data points assigned to the j^{th} leaf, λ is the *learning rate*, $G_j = \sum_{i \in I_j} g_i$ and $H_j = \sum_{i \in I_j} h_i$, and g_i and h_i are the first and second partial derivatives of the loss function respectively.

The part of Equation 39 within the summation is in a quadratic form. The optimal condition of a second degree objective function is at the point where the first derivative is zero. Hence, substituting the value of w_j at the point where the slope of the objective function is zero gives the optimal value of the objective function.

$$w_j^* = \frac{-G_j}{\big(H_j + \lambda\big)} \tag{40}$$

S. Id	g_i	h_i	$\hat{y}_i^{(t-1)}$	y_i
2	0	2	0	1
6	0	2	0	1
10	0	2	0	1
11	0	2	0	1

Table 8: Gradient statistics for left child node of root node of Tree 1.

S. Id	g_i	h_i	$\hat{y}_i^{(t-1)}$	y_i
1	0	2	0	-1
3	-2	2	0	-1
4	0	2	0	-1
5	0	2	0	-1
7	-2	2	0	-1
8	0	2	0	-1
9	0	2	0	-1
12	0	2	0	-1
13	-2	2	0	-1
14	0	2	0	-1
15	-2	2	0	-1
16	0	2	0	-1
17	-2	2	0	-1
18	-2	2	0	-1
19	0	2	0	-1
20	0	2	0	-1

Table 9: Gradient statistics for right child node of root node of Tree 1.

Thus, substituting w_j^* in the objective function, we get the optimal value of the objective (from Equation 39):

4.2.2. A Relevant Example for XGBoost

Consider the split made in the root node in Tree 1 (Figure 4). Utilizing the definitions laid down of g_i, h_i, Gain, λ, and γ, it can be considered from the perspective of gradient boosted trees that the left child node of the root node estimates $\hat{y}_i$ as Class 1 and the right node estimates it as -1, as the majority of the samples used for building the tree belong to classes $+1$ and -1 in the left and right children nodes, respectively. Thus, the structure score of the left branch may be calculated as (from Table 8):

From Table 9, G_j and H_j may be calculated as: $G_j = \sum g_i = 0$ and $H_j = \sum h_i = 8$. All the entities in this node truly belong to Class 1, and by the definition of g_i, the value of G_i sums up to 0. As the root node is the first in the series, it does not classify any of the entities and hence $\hat{y}_i^{(t-1)}$ for this split is 0 (note that the class labels are $+1$ and -1, signifying an increase and decrease in prices respectively). Consider the value of learning rate, λ, to be 1, arbitrarily. The structure score can thence be calculated as:

$$obj^* = -\frac{1}{2} \cdot \frac{G_L^2}{H_L + \lambda} = -\frac{1}{2} \cdot \frac{0}{8+1} = 0 \tag{41}$$

Similarly, for the right child node, the gradient statistics may be calculated using Table 9, in the following manner:

Here, $G_j = -12$ and $H_j = 32$. The structure score then becomes:

$$obj^* = -\frac{1}{2} \cdot \frac{G_R^2}{H_R + \lambda} = -\frac{1}{2} \cdot \frac{-12^2}{32+1} = -4.36 \tag{42}$$

Next, we shall see how the Gain of the split is calculated. For this, let us consider the value of γ to be 0.3, arbitrarily:

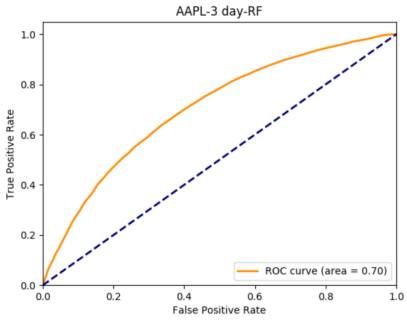
$$\begin{aligned} Gain &= -\frac{1}{2} \cdot \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{G_{Root}^2}{H_{Root} + \lambda} \right] - \gamma \\ &= -\frac{1}{2} \cdot \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma \\ &= -\frac{1}{2} \cdot \left[0 + \frac{144}{32 + 1} - \frac{(0 - 12)^2}{8 + 32 + 1} \right] - 0.3 \\ &= 0.8514 - 0.3 \\ &= 0.5514 \end{aligned} \tag{43}$$

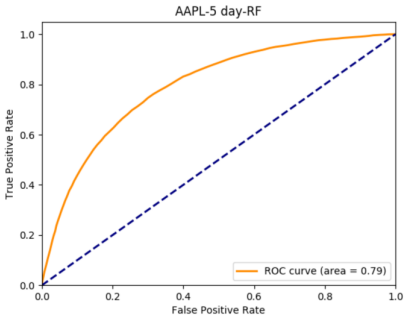
This is how the gain is calculated. In a similar fashion, the remaining levels of this tree are constructed. The parameters λ and γ can be tuned for optimal performance. Once this tree is complete, the next tree is built by bagging. Thus, with every level and every tree in sequence, the forest of trees is able to reduce the error in classification.

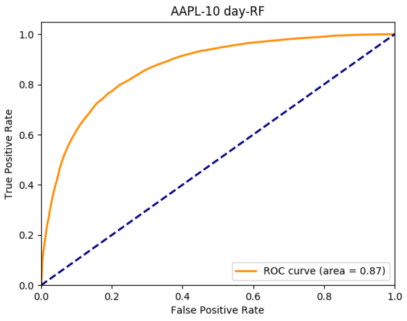
5. Receiver Operating Characteristic Plots

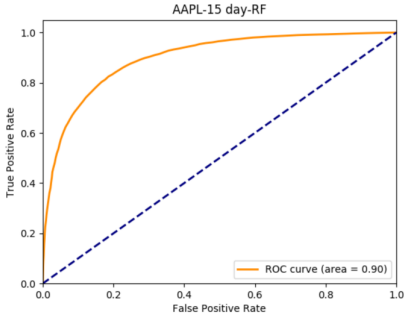
In this section, we present the ROC curves of the classifiers' outputs for the results presented and discussed in the main paper in Section 4.

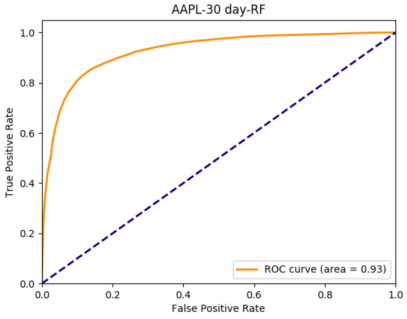
AAPL

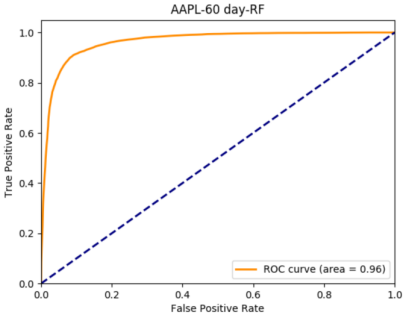

AAPL-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.70)

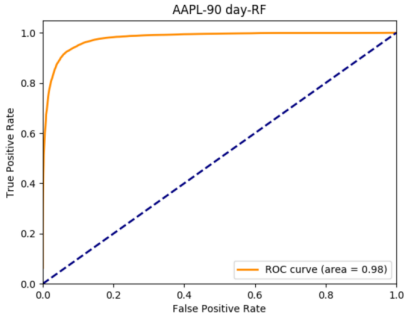

AAPL-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

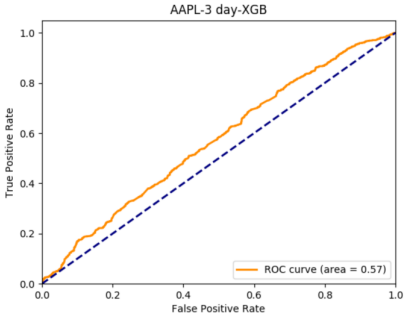

AAPL-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.87)

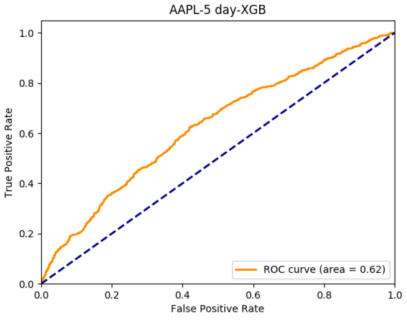

AAPL-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.90)

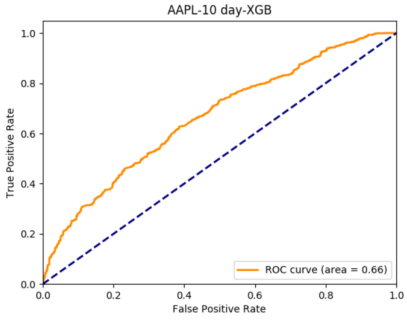

AAPL-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

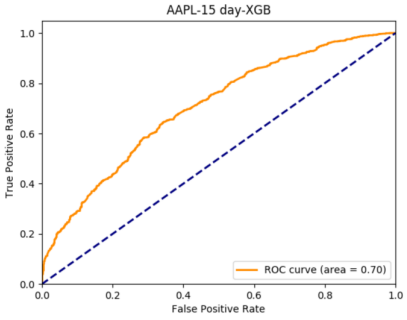

AAPL-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)

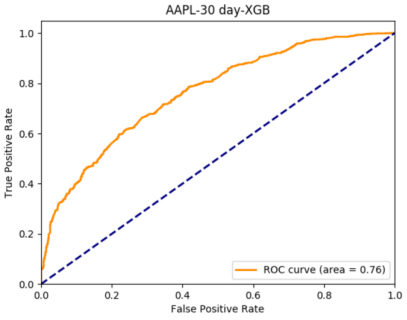

AAPL-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)

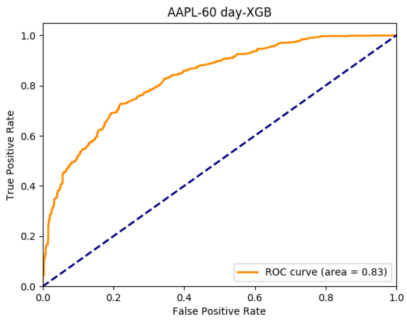

AAPL-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

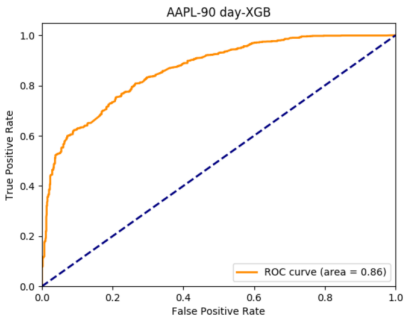

AAPL-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)


AAPL-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

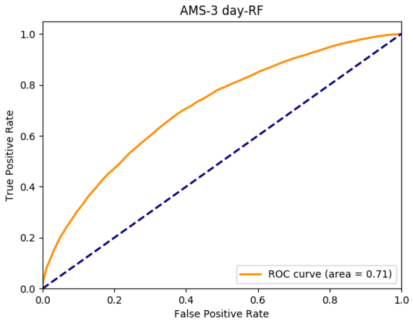

AAPL-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.70)

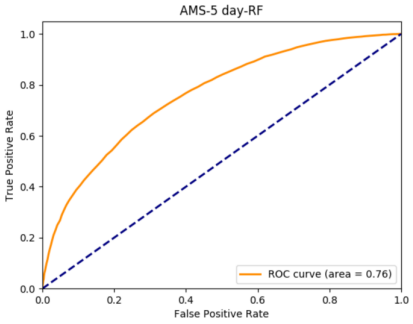

AAPL-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.76)

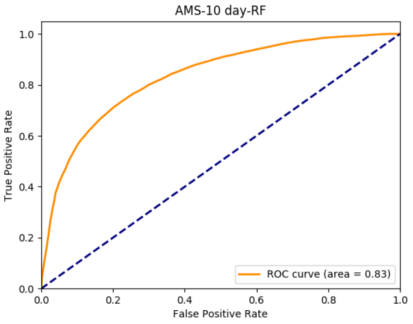

AAPL-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.83)

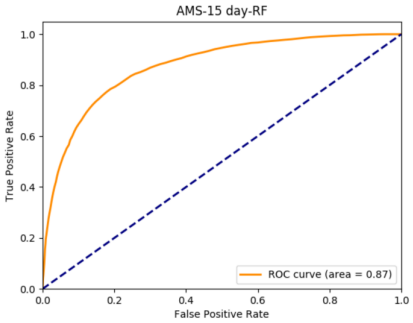

AAPL-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)

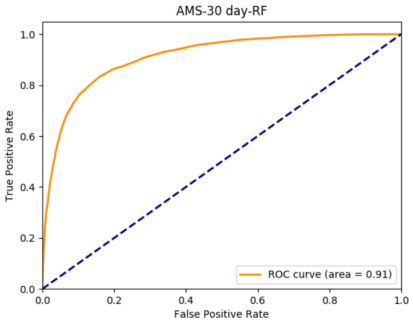
AMS

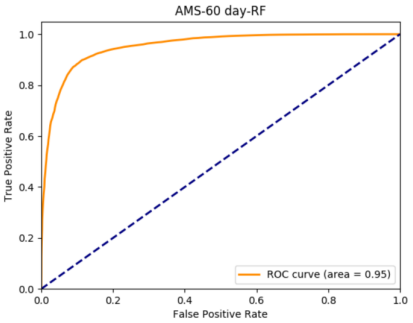

AMS-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

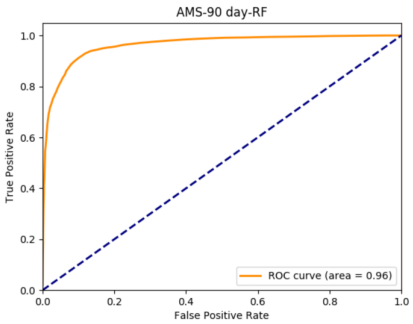

AMS-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.76)

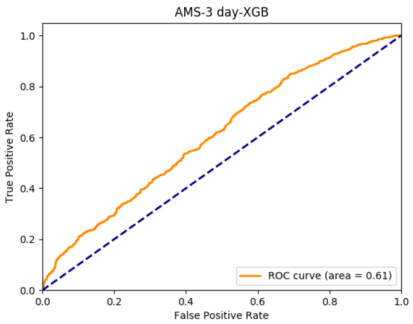

AMS-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.83)

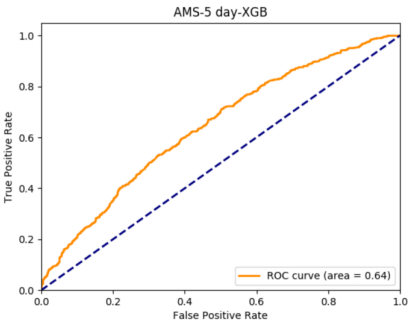

AMS-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.87)

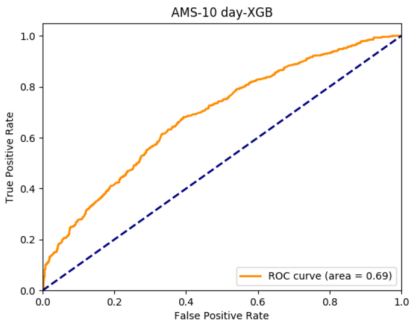

AMS-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)

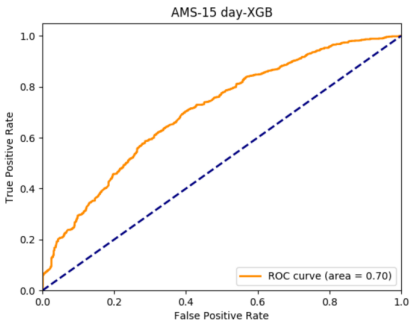

AMS-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

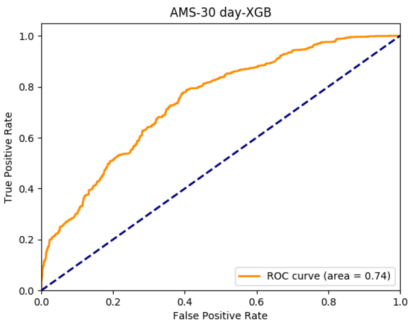

AMS-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)

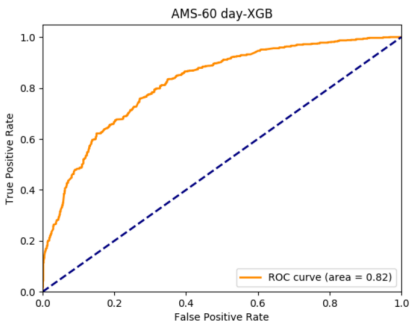

AMS-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

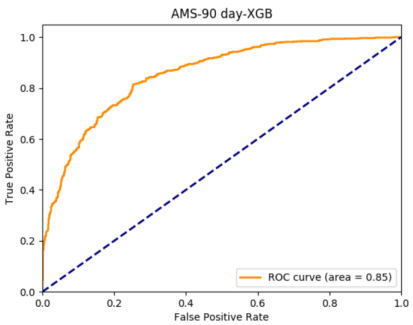

AMS-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)


AMS-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

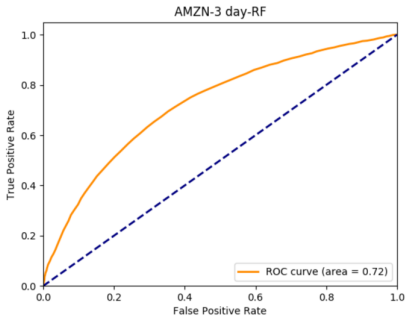

AMS-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.70)

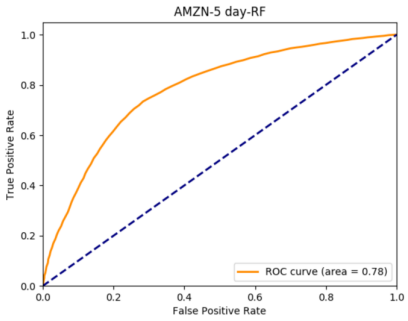

AMS-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

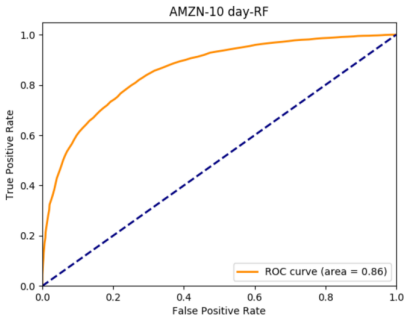

AMS-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.82)

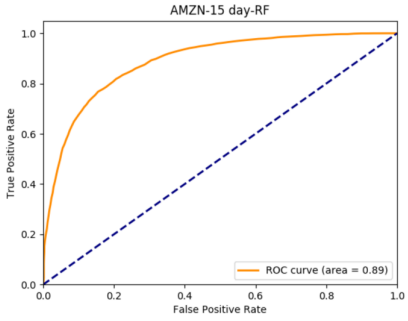

AMS-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.85)

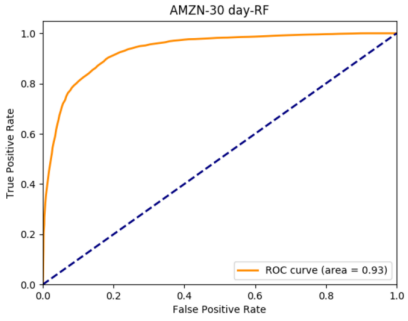
AMZN

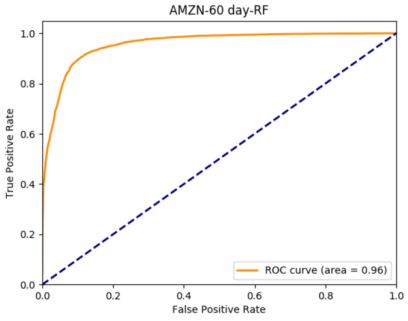

AMZN-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

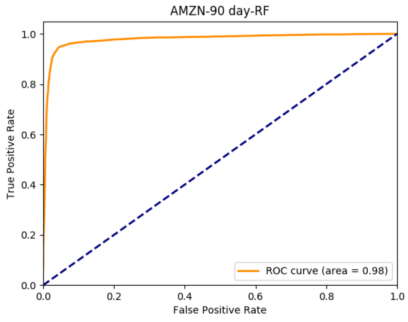

AMZN-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.78)


AMZN-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)


AMZN-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)

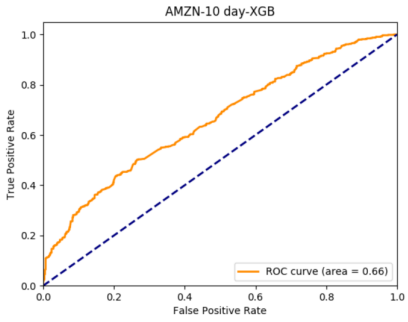

AMZN-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

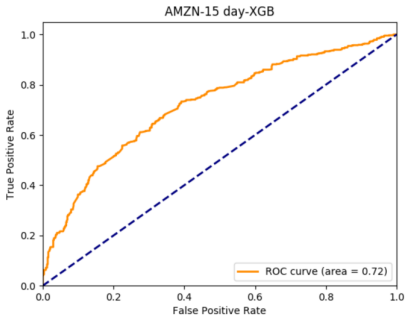

AMZN-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)

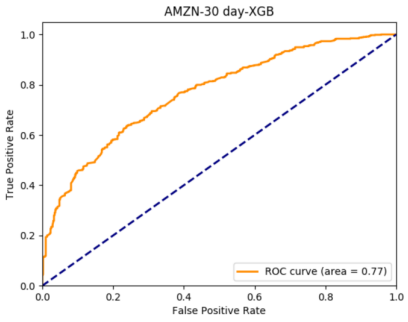

AMZN-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)

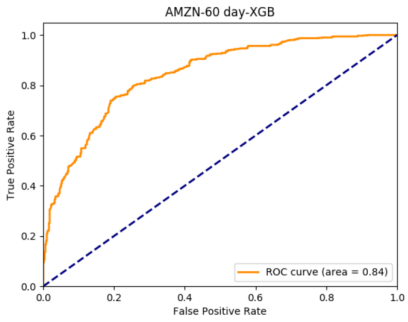

AMZN-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

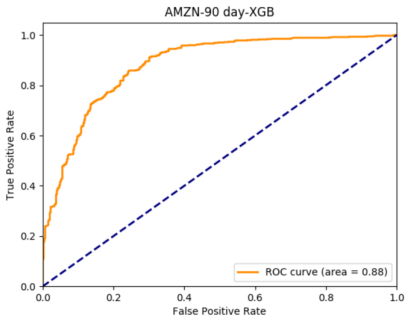

AMZN-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)


AMZN-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

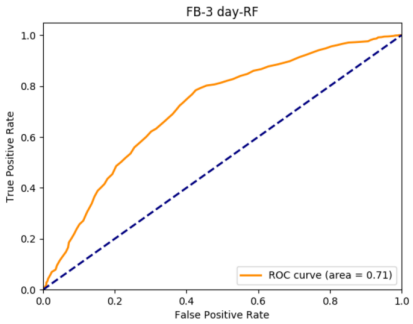

AMZN-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

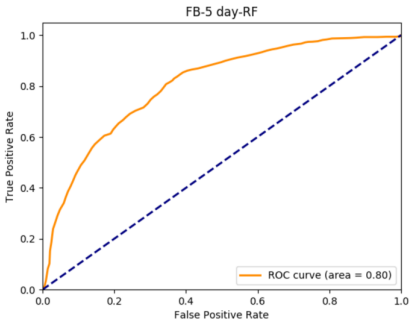

AMZN-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)

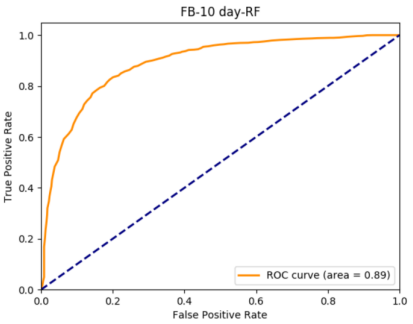

AMZN-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.84)

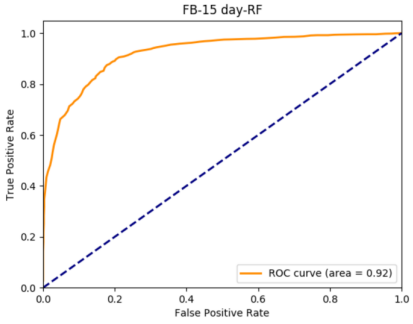

AMZN-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.88)

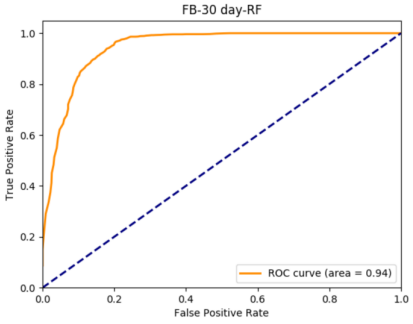
FB

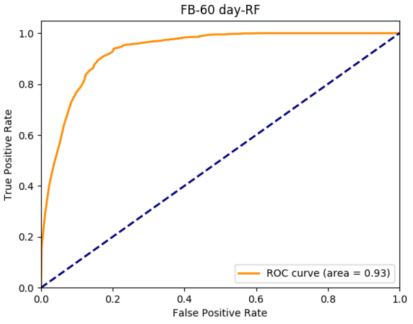

FB-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

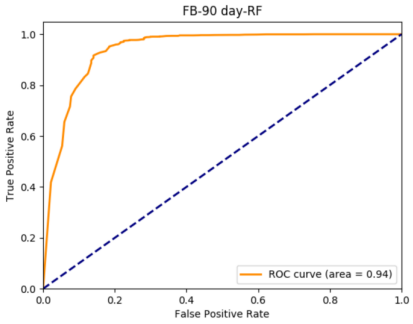

FB-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

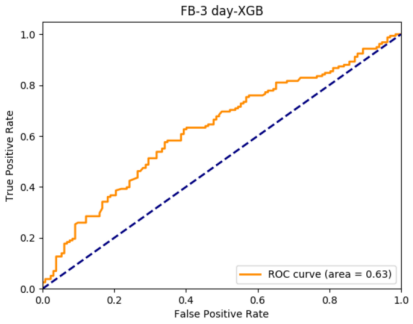

FB-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)

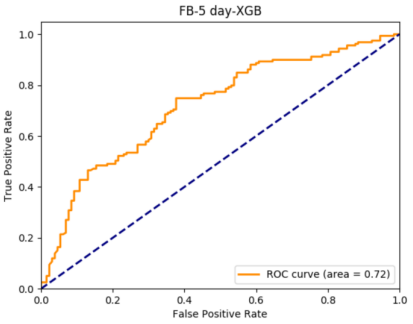

FB-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.92)

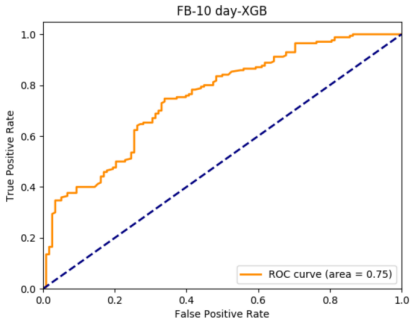

FB-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

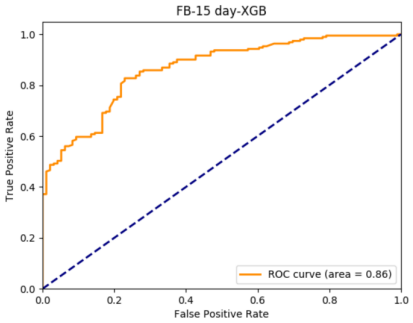

FB-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

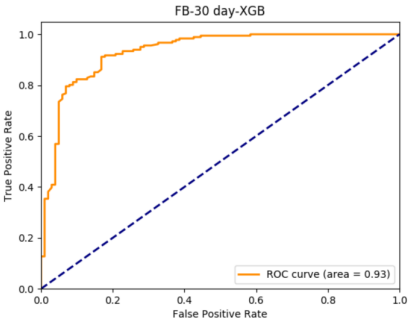

FB-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

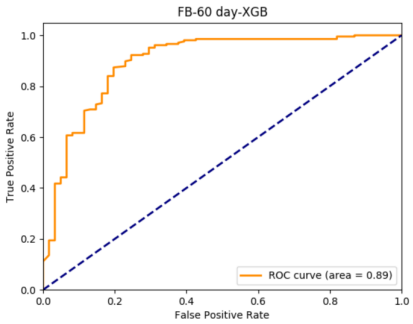

FB-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

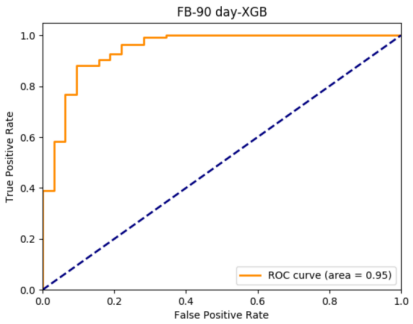

FB-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)


FB-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)

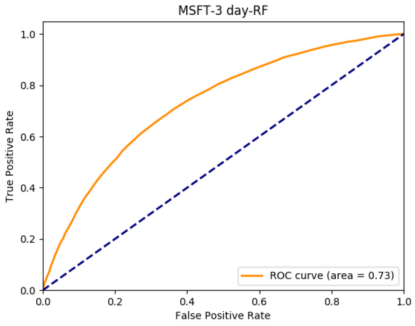

FB-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)

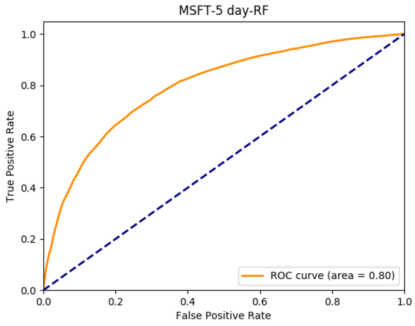

FB-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

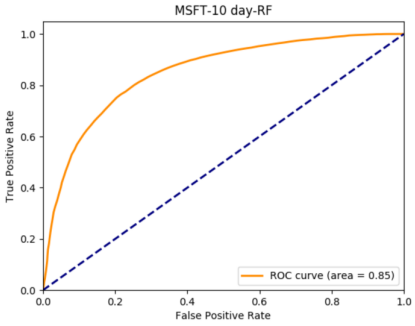

FB-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)

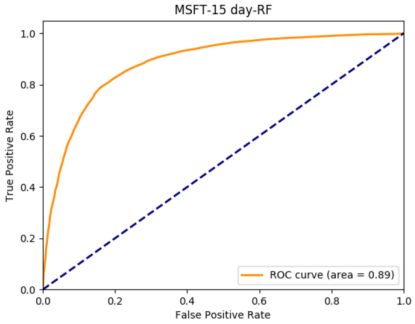

FB-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

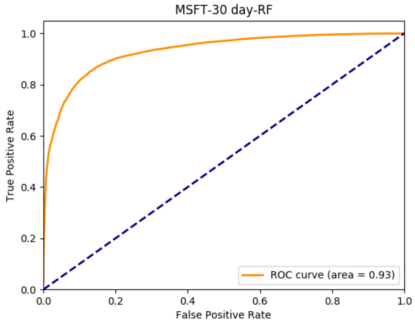
MSFT

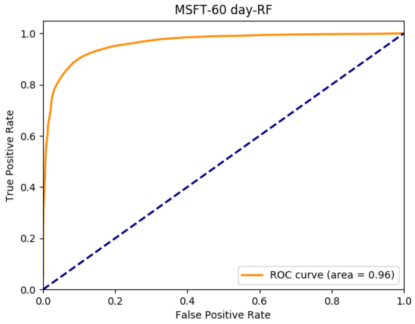

MSFT-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.73)

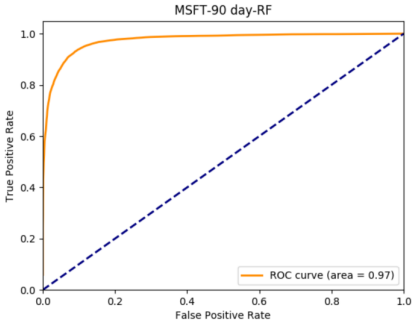

MSFT-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

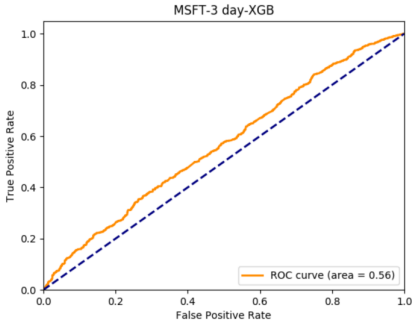

MSFT-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.85)

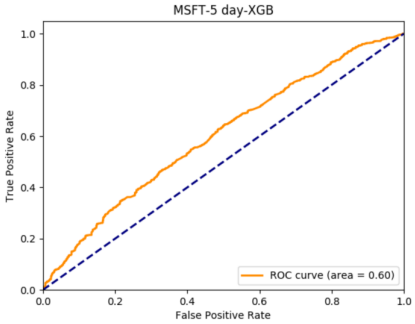

MSFT-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)

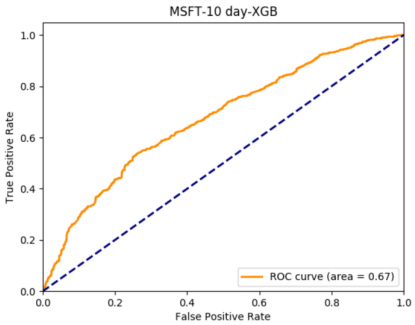

MSFT-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

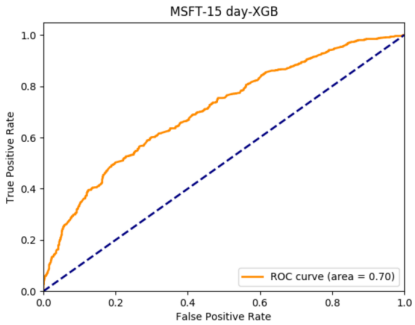

MSFT-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)

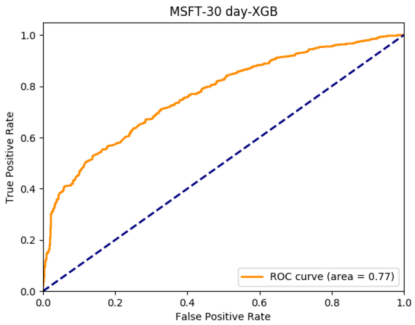

MSFT-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.97)

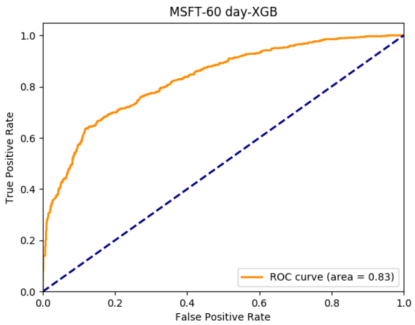

MSFT-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

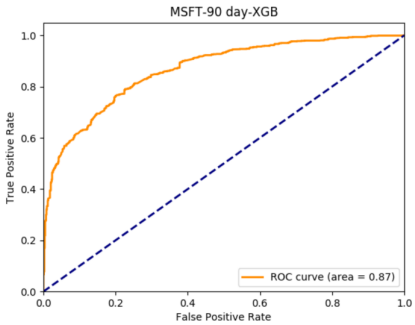

MSFT-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)


MSFT-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

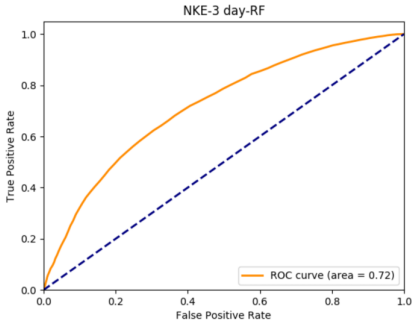

MSFT-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.70)

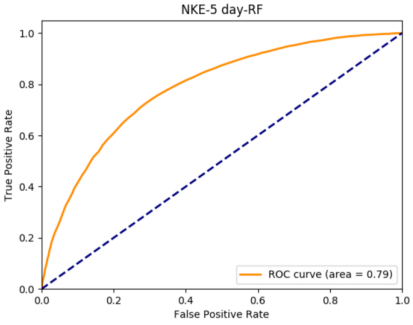

MSFT-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)

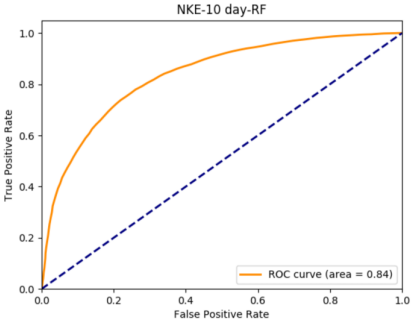

MSFT-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.83)

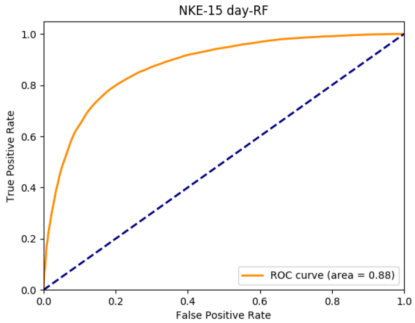

MSFT-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.87)

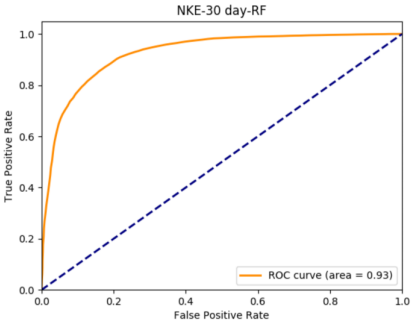
NKE

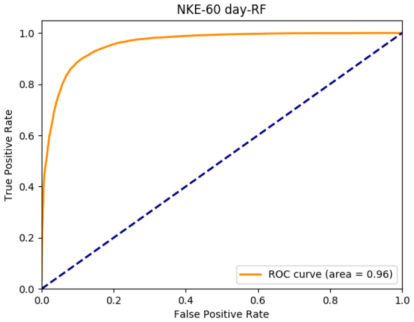

NKE-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

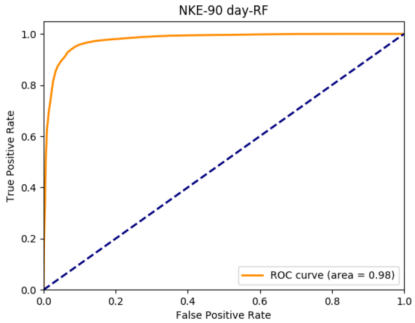

NKE-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

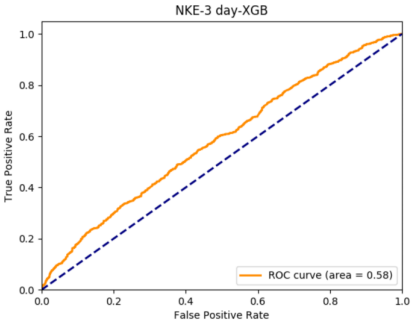

NKE-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.84)

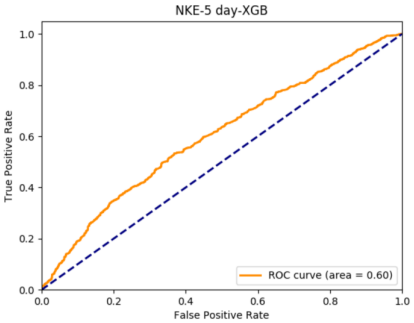

NKE-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.88)

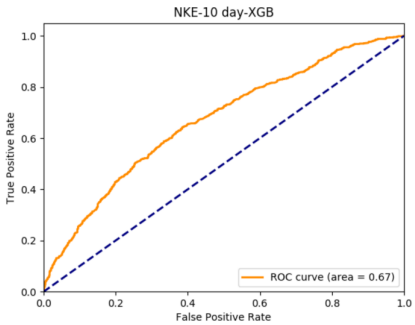

NKE-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

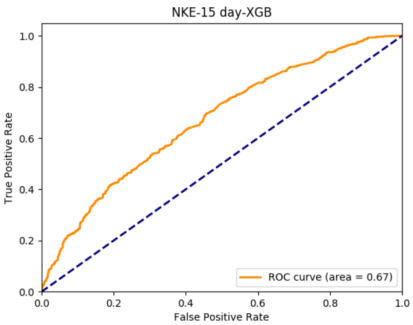

NKE-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)

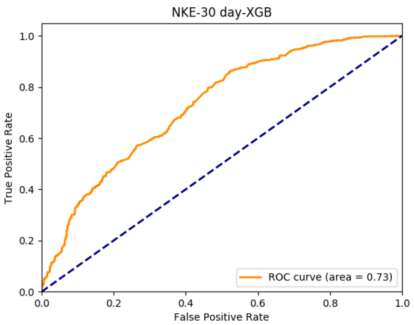

NKE-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)

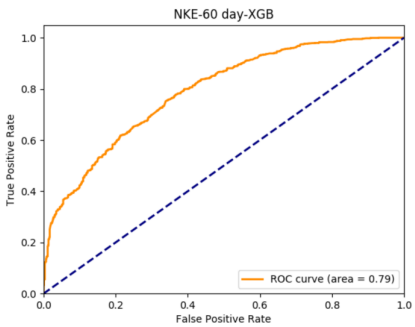

NKE-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

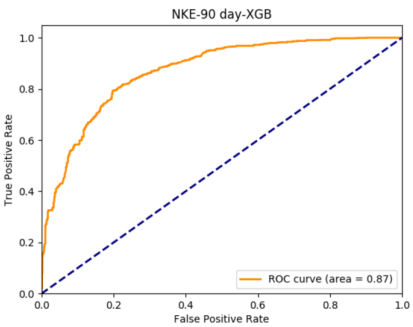

NKE-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)


NKE-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

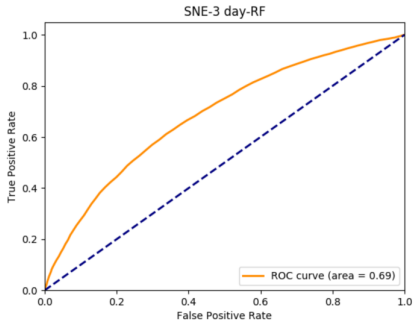

NKE-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

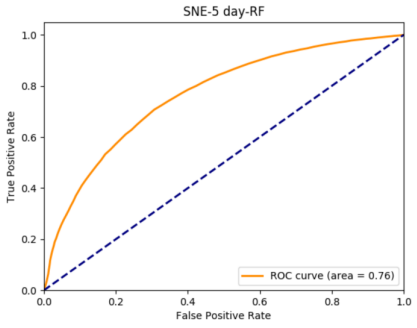

NKE-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.73)

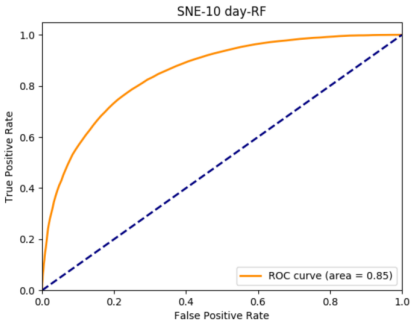

NKE-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

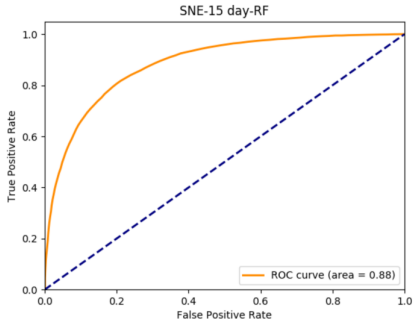

NKE-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.87)

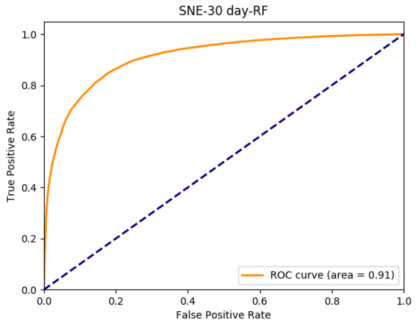
SNE

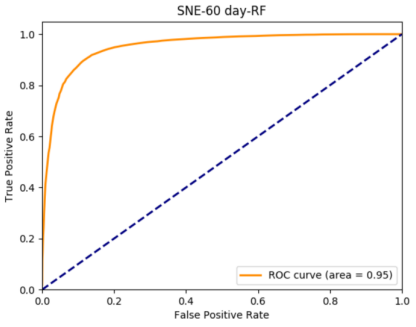

SNE-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

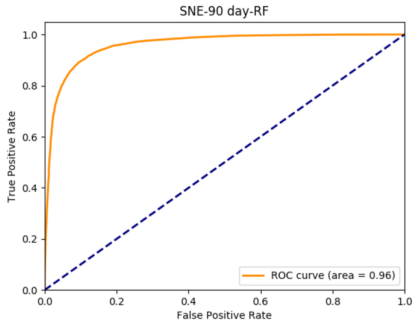

SNE-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.76)

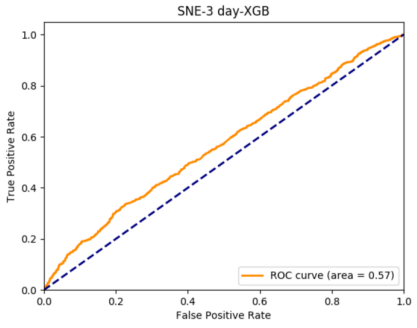

SNE-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.85)

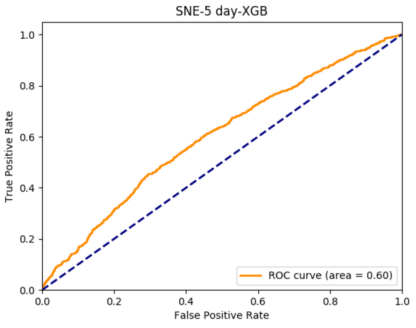

SNE-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.88)

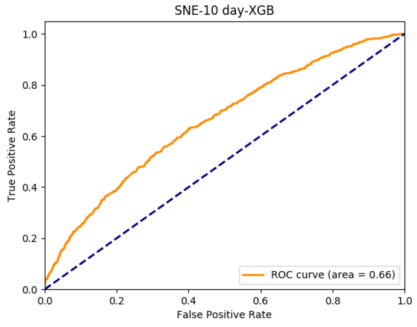

SNE-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)

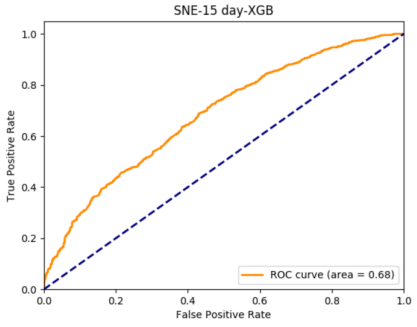

SNE-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

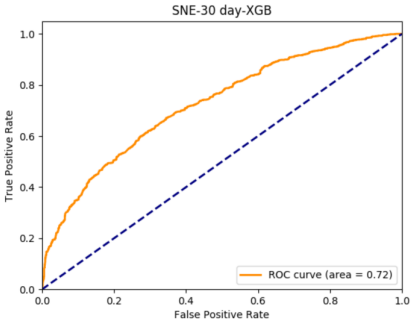

SNE-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)

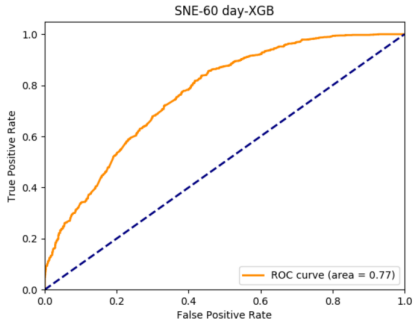

SNE-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

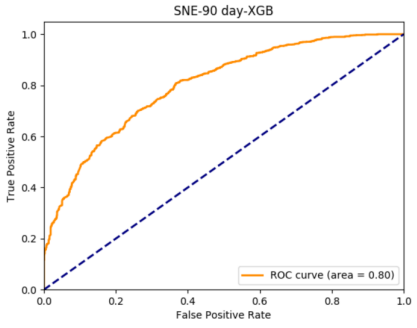

SNE-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)


SNE-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

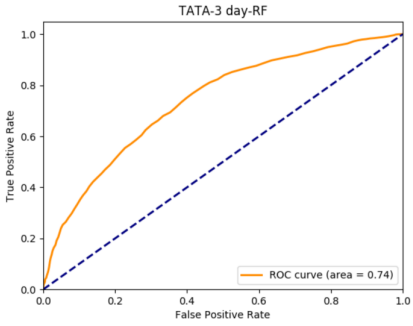

SNE-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.68)

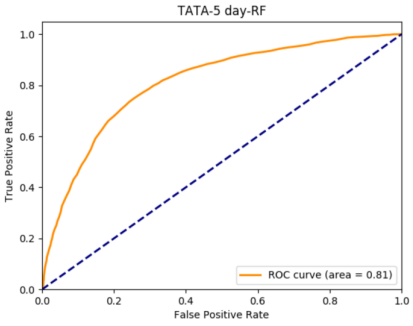

SNE-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

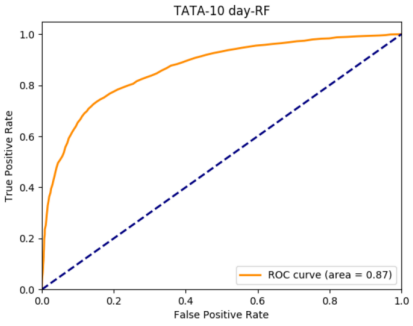

SNE-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)

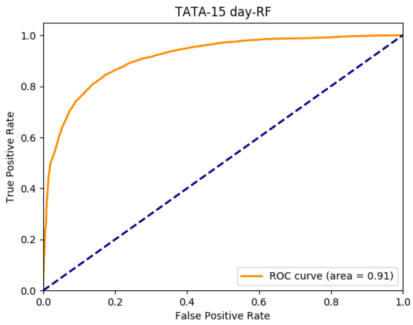

SNE-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

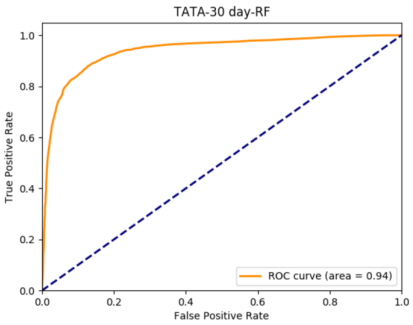
TATA

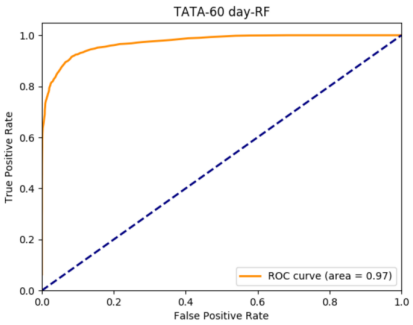

TATA-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

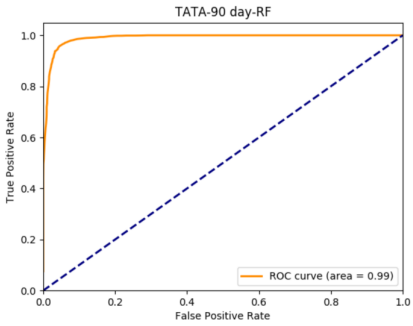

TATA-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.81)

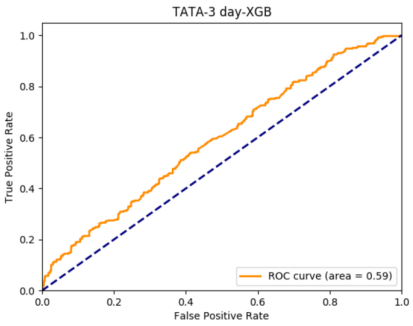

TATA-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.87)

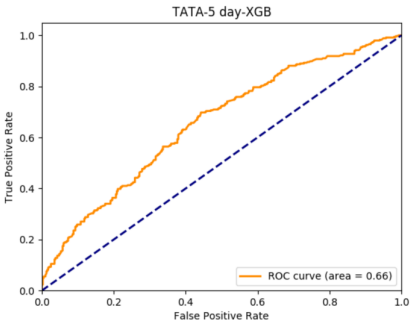

TATA-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)

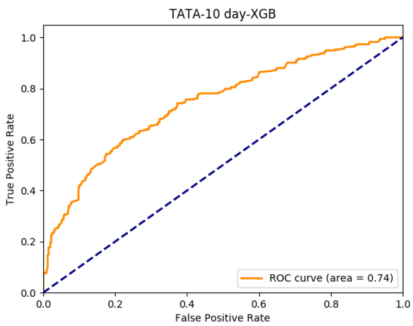

TATA-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

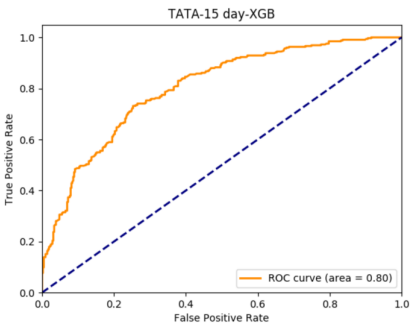

TATA-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.97)

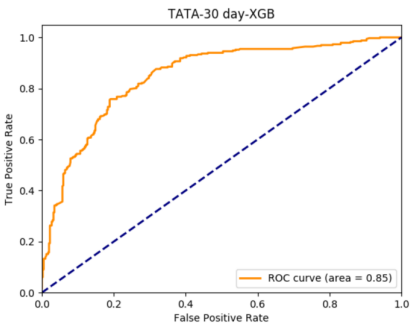

TATA-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.99)

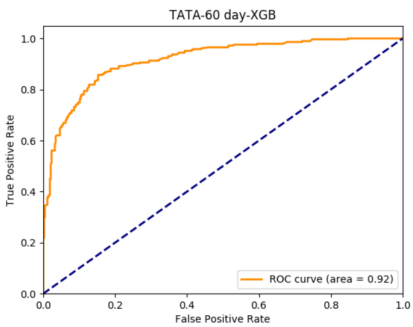

TATA-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

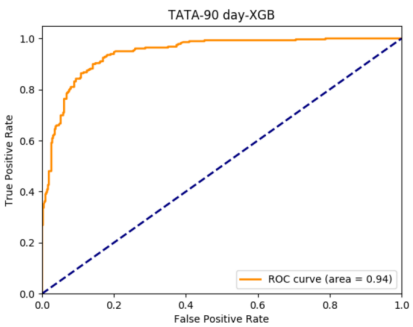

TATA-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)


TATA-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

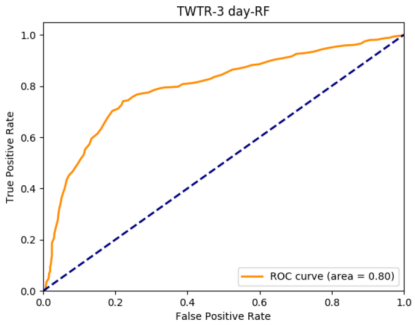

TATA-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

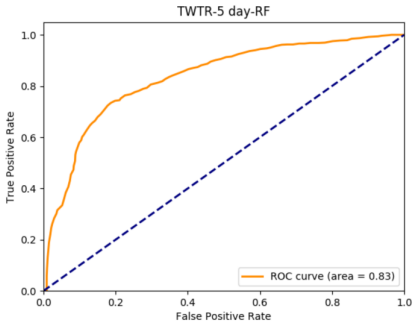

TATA-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.85)

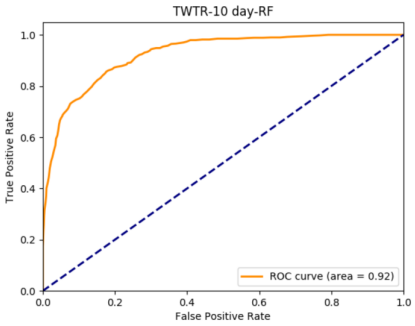

TATA-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.92)

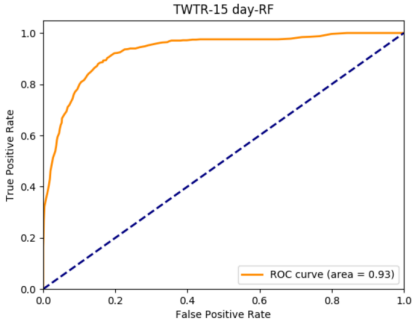

TATA-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

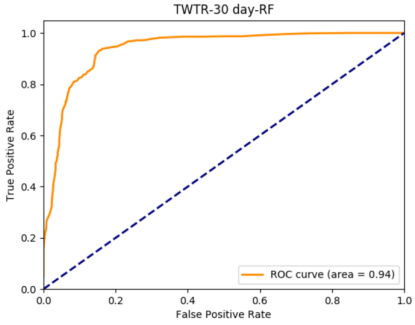
TWTR

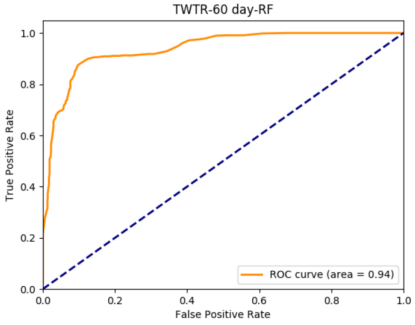

TWTR-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

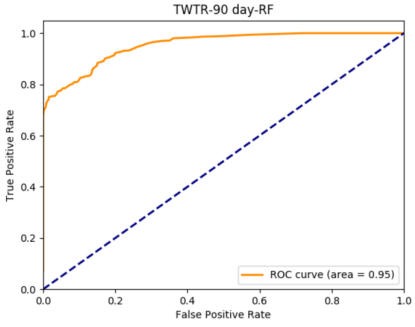

TWTR-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.83)

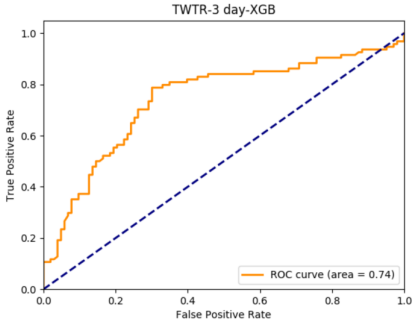

TWTR-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.92)

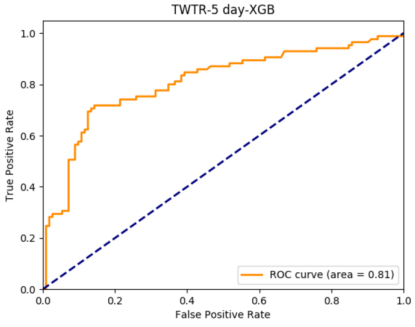

TWTR-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

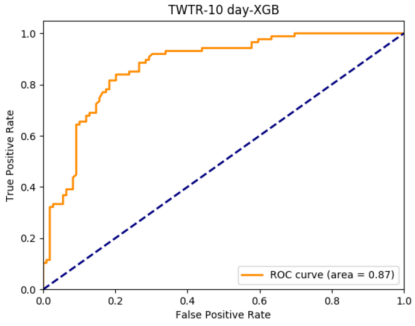

TWTR-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

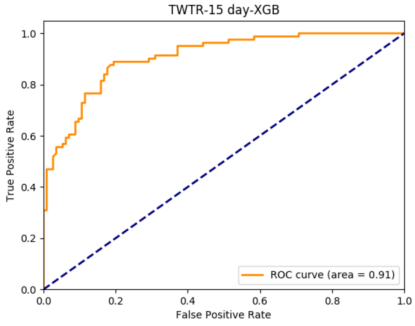

TWTR-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

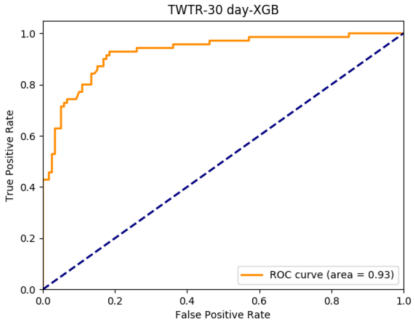

TWTR-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

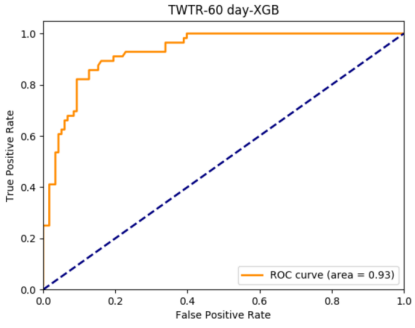

TWTR-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

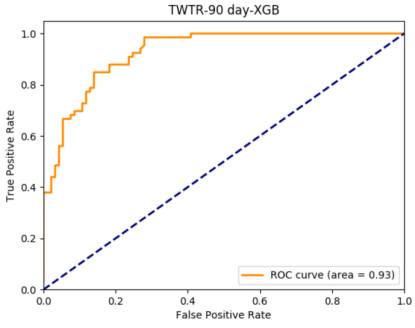

TWTR-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.81)


TWTR-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.87)

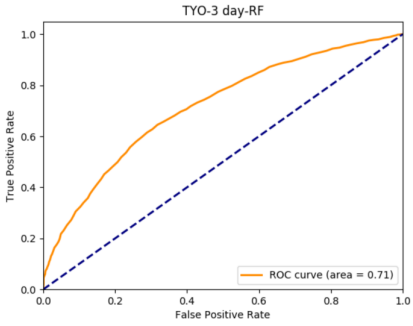

TWTR-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)

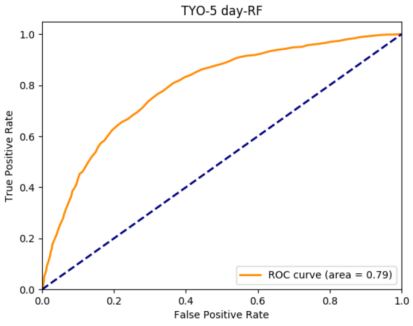

TWTR-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

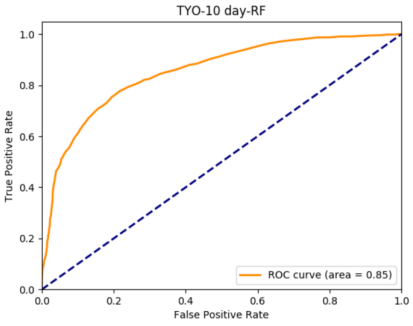

TWTR-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

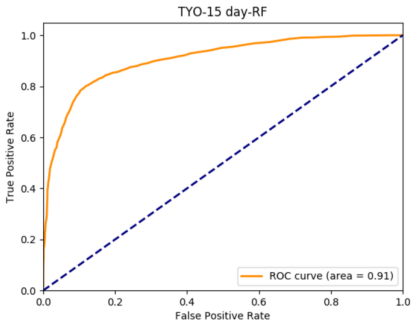

TWTR-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

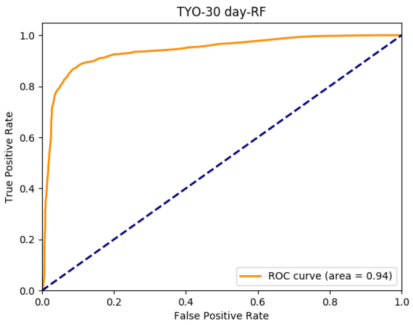
TYO

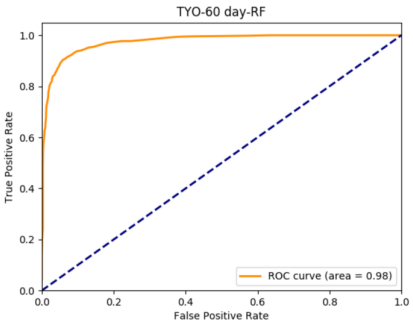

TYO-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

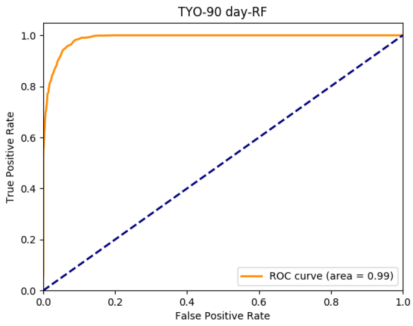

TYO-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

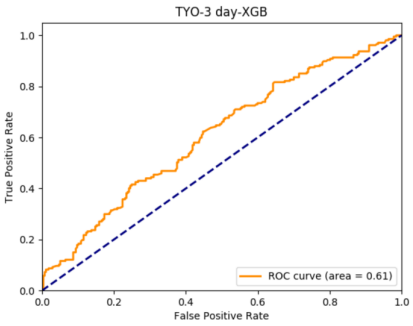

TYO-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.85)

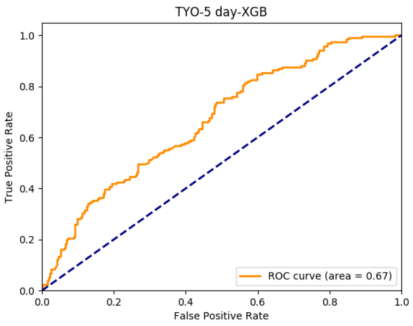

TYO-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)

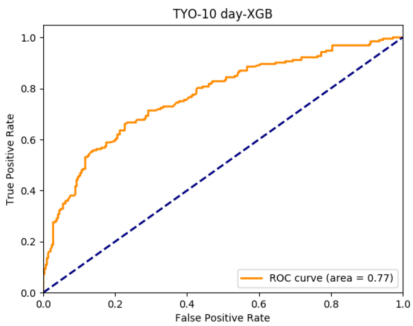

TYO-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

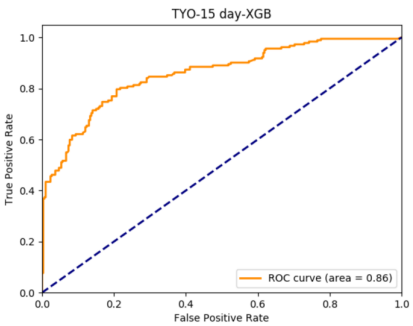

TYO-60 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)

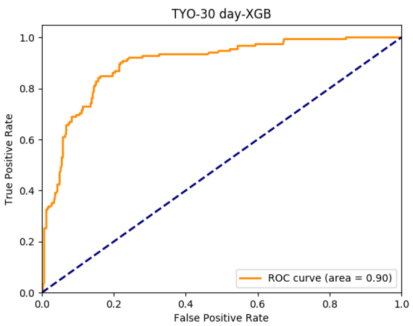

TYO-90 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.99)

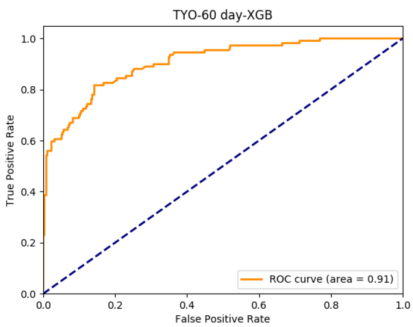

TYO-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

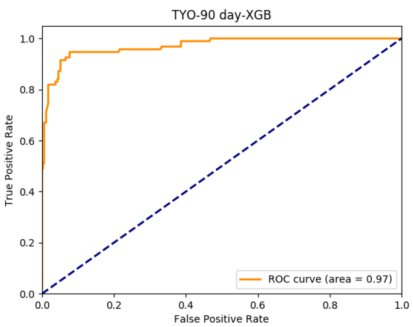

TYO-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)


TYO-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)


TYO-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)


TYO-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.90)


TYO-60 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)


TYO-90 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.97)

6. An Application of the ML Methods to Pharmaceutical Stocks: A Case Study

It may be an interesting exercise to investigate why certain stocks did not succumb to the aggravated economic crisis, even during the last round of global meltdown. The relevant candidates could easily be found among the pharmaceutical companies. Generally speaking, the health care industry comprising of the life science sectors such as pharmaceutical, biotechnology, and medical devices do not respond significantly to crisis or boom. These are well-known providers of inelastic commodities and services, that do not undergo visible changes in demand when subjected to income and price effects. These products are less exposed to fluctuations typically because health care requirements continue to remain unchanged regardless of economic prosperity or downturns. Indeed, during economic crisis, it is quite possible that people may be even more susceptible to stress and depression, leading to other forms of ailments. However, it cannot be neglected that during economic crisis, the internal readjustments within such companies generating purely supply side effects could lead to loss of production. It is also expected that research and development, which is at the core of innovations and inventions for the pharmaceutical industry get negatively affected. This may have considerable implications for profit and the outcome of stock prices. The principal conjecture is that the randomness or volatility does not affect the prediction accuracy significantly when the time window is expanded. Pharmaceutical stocks may indeed be resilient to shocks (Behner et al., 1998). It is therefore an empirically open question that we attempt to answer through the general procedures followed above.

From a machine learning and methodology point of view, it is an interesting problem to address. The reasons we want to address stocks of pharmaceutical companies are two. First, it is for the purpose of diversification – we wanted to make sure that we address stocks of different types of companies. And second, the low fluctuations in the stocks provides an interesting premise for automation methods for stock trading suggestions and strategies. The results confirm our choice of methods and the higher-than-usual accuracy in turn verifies the nature of the stocks of pharmaceutical companies. The direction predictability, is related to the stocks' variance over time. Naturally, the gains or losses of stocks which are stable would be easier to predict than stocks that are relatively noisy. A lower variance in the data implies a better predictive capability of ML classifiers, and from an economic point of view, it accounts for the stability. For these reasons, we have selected data for one year and have presented the results till $t = 30$ days. Our presumption prior to investigating this was that the convergence would be appreciable and that we'd be able to achieve good results for a shorter trading window. Our results confirm our hypothesis.

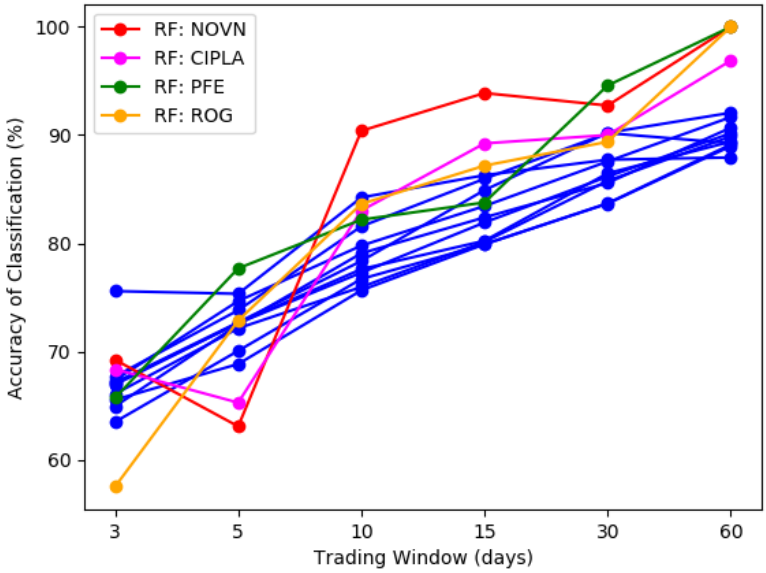


Figure 10: Comparison of the performance of the algorithm on stock data of pharmaceutical companies. The plots in blue represent the accuracies of ten other companies (the results of whose classification are presented in Appendix 1 of the main paper. Prediction accuracy in pharmaceutical stocks is distinctively better.

From Figure 10, it is observed that in the trading window between 10 and 60 day period, the method exhibits better accuracy. This is noteworthy as it establishes the resistance of pharmaceutical companies to market fluctuations. More specifically, the prediction accuracy achieved for the non-pharma organizations (10 data sets considered in our analysis) requires at least 60-day window to match the accuracy in prediction, as compared to the pharmaceutical companies that take only 10-15 day window to match the prediction. Typically, the exercise of predictive analysis in pharmaceutical stocks does not seem to require wide training windows. This is clearly in agreement with our conjecture.

As reported by Behner et al. (1998) (only a few such studies are available since the crisis), unless major policy changes are implemented or completely exogenous factors are in play (such as budget cuts in health care), the pharmaceutical stock prices are less exposed to financial crisis or elasticity. Therefore, understanding and accounting for volatility in pharmaceutical stocks should be easier as compared to other stocks we considered in this paper. Consequently, it is not too hard to comprehend the reason for the remarkable prediction accuracy achieved using shorter time windows. We have presented the results in Tables 11 and 11 and the ROC plots of the pharmaceutical companies after that.

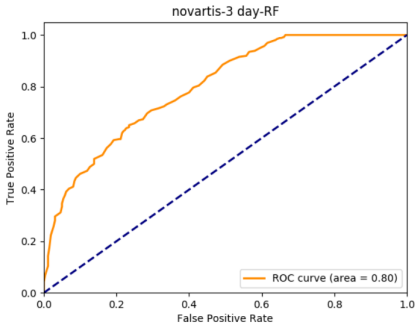
Table 10: RF Results Table for Pharmaceutical Stocks: results of Random Forests implemented on stocks of novartis, cipla, pfizer and roche.

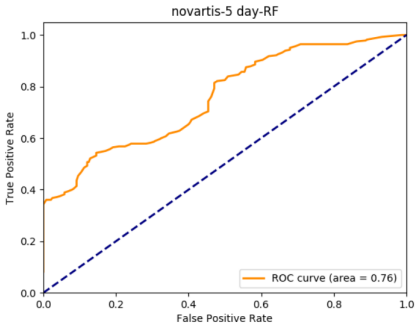
Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
novartis	3	68.46	0.75	0.66	0.62	0.70	0.33	0.79
	5	63.65	0.59	0.69	0.69	0.64	0.38	0.76
	10	90.00	1.00	0.81	0.83	0.89	0.10	0.99
	15	93.88	0.89	0.94	0.97	0.92	0.06	0.95
	30	92.73	0.92	0.83	0.93	0.87	0.07	0.97
cipla	3	69.27	0.72	0.72	0.66	0.72	0.29	0.73
	5	63.45	0.76	0.50	0.56	0.60	0.36	0.76
	10	82.64	0.93	0.79	0.70	0.85	0.19	0.92
	15	89.23	0.91	0.89	0.87	0.90	0.12	0.95
	30	90.43	0.91	0.91	0.90	0.91	0.11	0.92
pfizer	3	66.43	0.50	0.72	0.82	0.59	0.39	0.75
	5	79.11	0.77	0.74	0.81	0.75	0.21	0.91
	10	82.36	0.74	0.87	0.90	0.80	0.18	0.89
	15	83.02	0.71	0.95	0.96	0.81	0.17	0.95
	30	94.38	0.96	0.93	0.93	0.94	0.04	0.99
roche	3	55.00	0.60	0.61	0.48	0.60	0.45	0.61
	5	75.54	0.85	0.78	0.60	0.81	0.25	0.77
	10	83.09	0.93	0.84	0.63	0.88	0.15	0.89
	15	86.42	0.92	0.90	0.67	0.91	0.11	0.89
	30	90.00	1.00	0.90	0.20	0.95	0.10	0.80

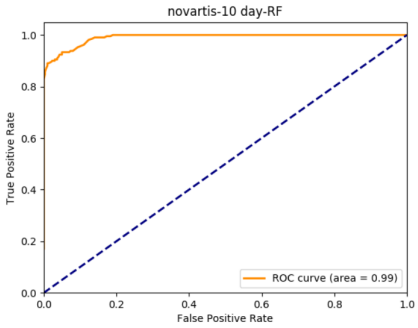
Table 11: XGB Results Table for Pharmaceutical Stocks: results of XGBoost implemented on stocks of novartis, cipla, pfizer and roche.

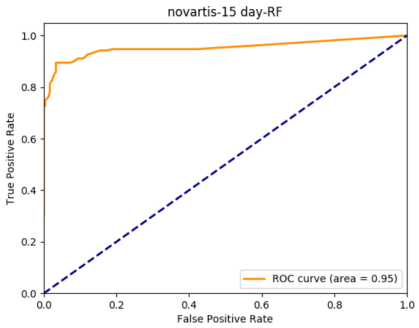
Company Name	Trading Window	Accuracy	Recall	Precision	Specificity	F-Score	Brier Score	AUC
novartis	3	63.46	0.73	0.61	0.54	0.67	0.37	0.74
	5	63.46	0.57	0.70	0.71	0.63	0.37	0.69
	10	90.20	0.95	0.83	0.87	0.89	0.10	0.97
	15	95.92	0.95	0.95	0.97	0.95	0.04	0.99
	30	90.91	0.92	0.79	0.91	0.85	0.09	0.98
cipla	3	65.45	0.67	0.69	0.64	0.68	0.35	0.74
	5	61.82	0.75	0.48	0.54	0.59	0.38	0.65
	10	84.91	0.93	0.82	0.75	0.87	0.15	0.95
	15	86.54	0.93	0.84	0.79	0.88	0.13	0.93
	30	89.36	0.88	0.92	0.91	0.90	0.11	0.91
pfizer	3	66.07	0.56	0.68	0.76	0.61	0.34	0.67
	5	76.79	0.74	0.71	0.79	0.72	0.23	0.83
	10	80.00	0.73	0.83	0.86	0.78	0.20	0.86
	15	84.91	0.74	0.95	0.96	0.83	0.15	0.94
	30	93.75	0.92	0.96	0.96	0.94	0.06	0.98
roche	3	58.93	0.72	0.62	0.42	0.67	0.41	0.61
	5	76.79	0.86	0.79	0.62	0.82	0.23	0.75
	10	85.45	0.95	0.85	0.67	0.90	0.15	0.94
	15	86.79	0.93	0.90	0.67	0.92	0.13	0.88
	30	89.58	1.00	0.89	0.17	0.94	0.10	0.78

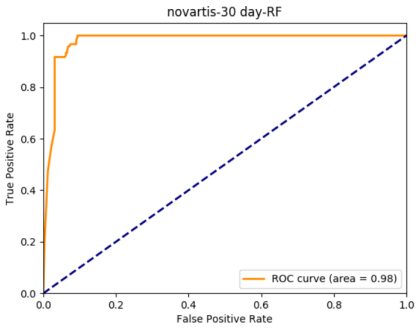
NOVN

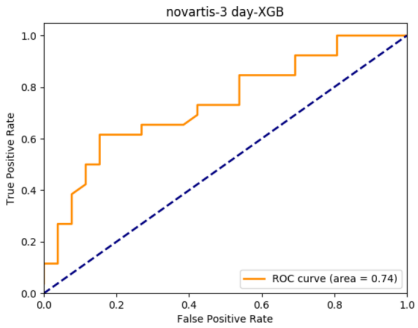

novartis-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

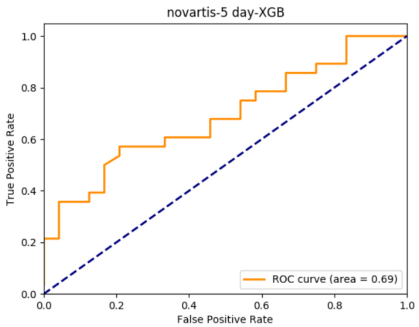

novartis-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.76)

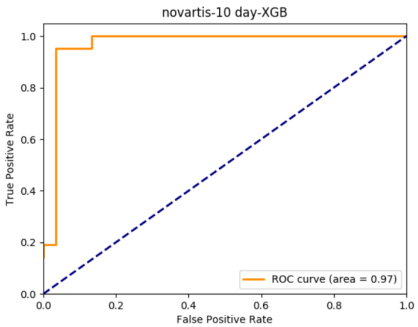

novartis-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.99)

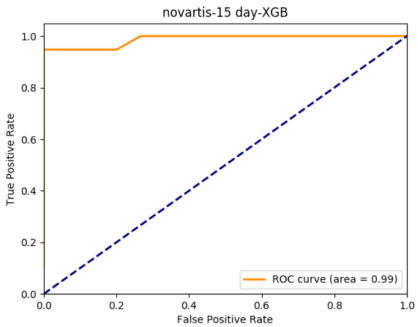

novartis-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

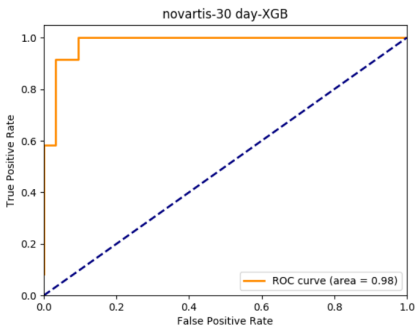

novartis-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)


novartis-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

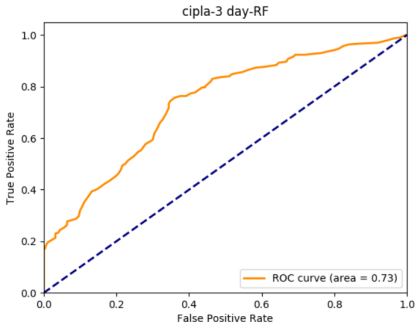

novartis-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

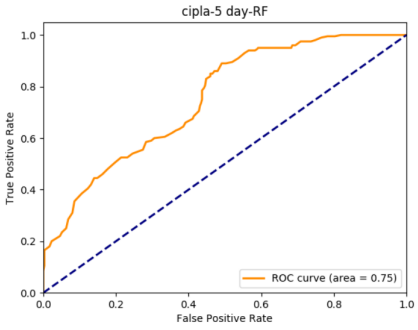

novartis-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.97)

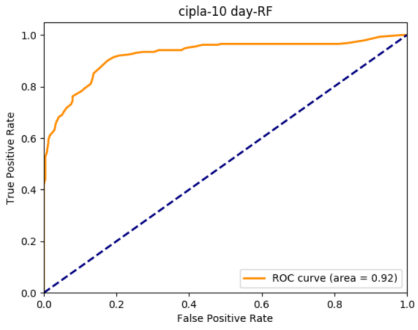

novartis-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.99)

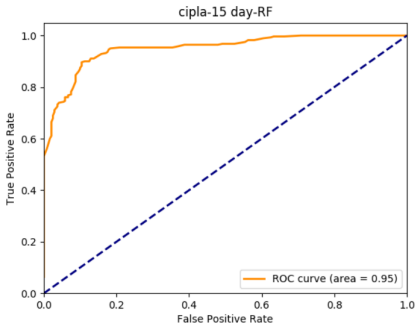

novartis-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)

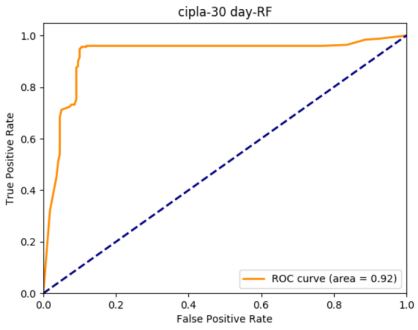
CIPLA

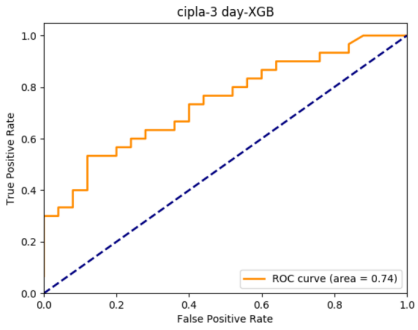

cipla-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.73)

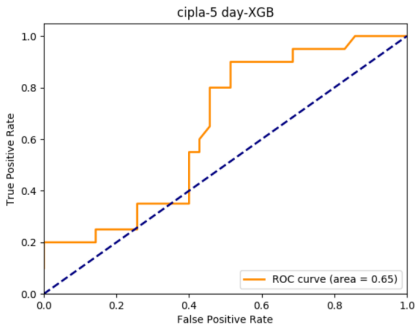

cipla-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)

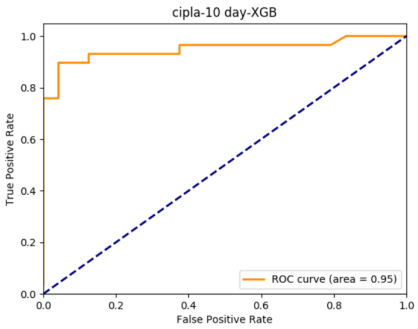

cipla-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.92)

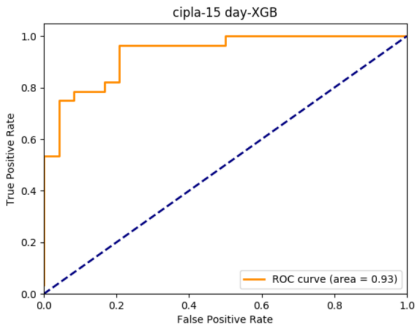

cipla-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

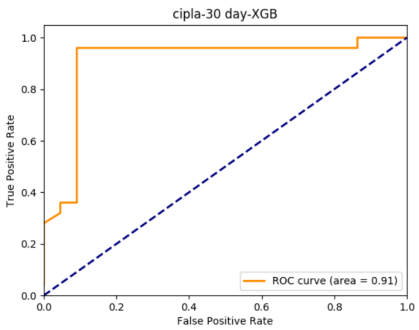

cipla-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.92)


cipla-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

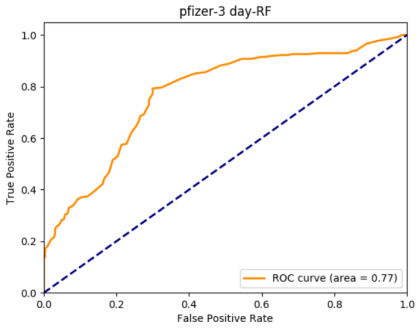

cipla-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

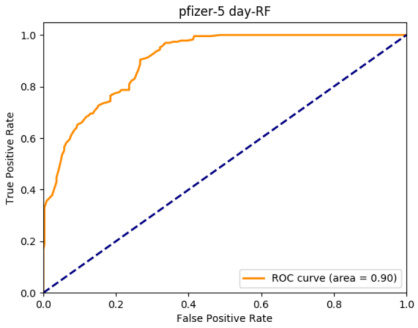

cipla-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

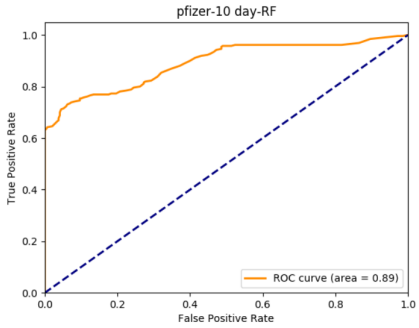

cipla-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.93)

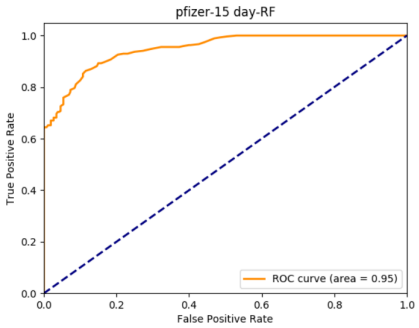

cipla-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.91)

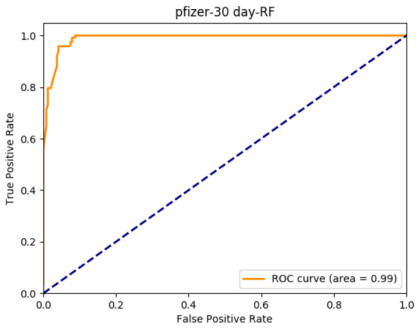
PFE

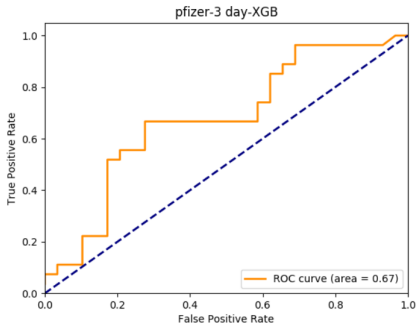

pfizer-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)

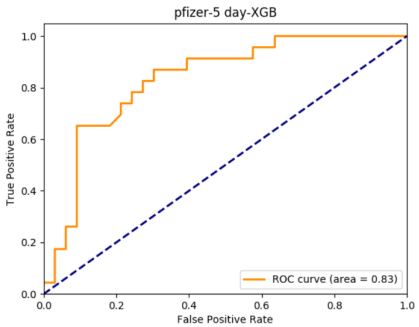

pfizer-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.90)

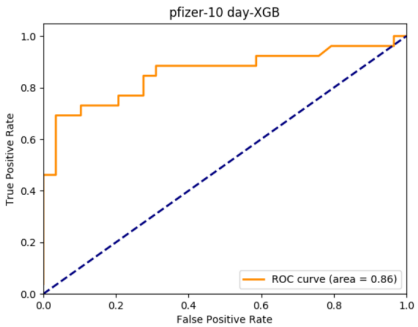

pfizer-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)

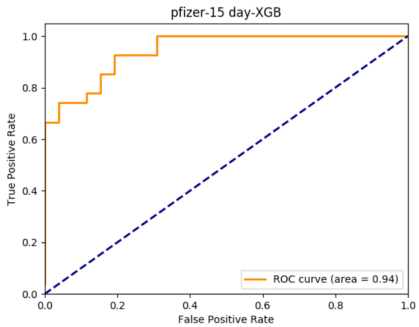

pfizer-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.95)

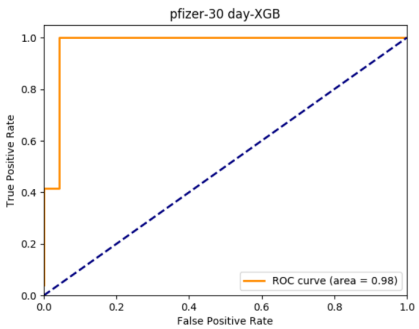

pfizer-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.99)


pfizer-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

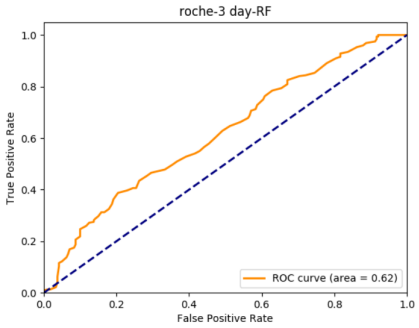

pfizer-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.83)

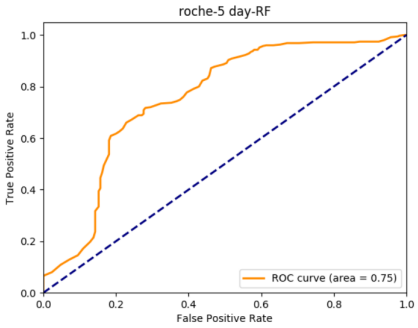

pfizer-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)

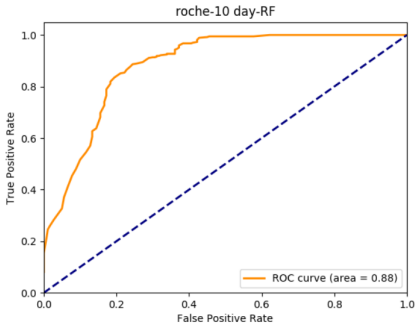

pfizer-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)

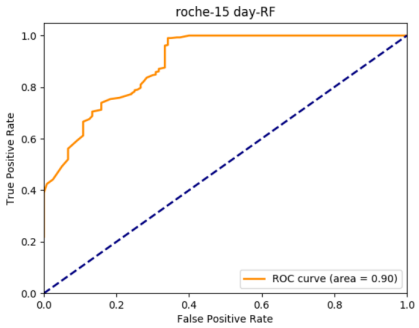

pfizer-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.98)

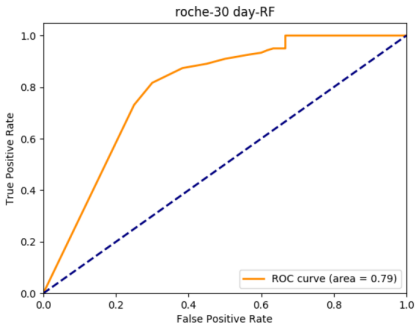
ROG

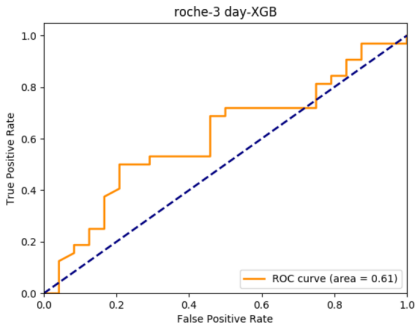

roche-3 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

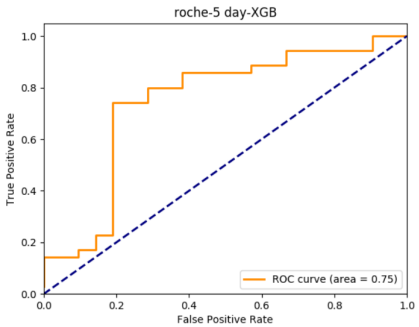

roche-5 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)

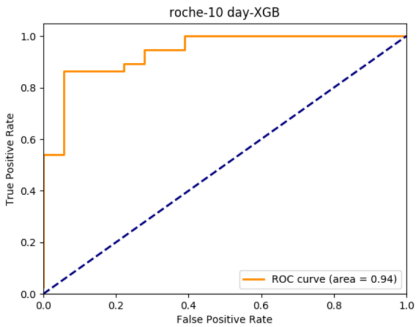

roche-10 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.88)

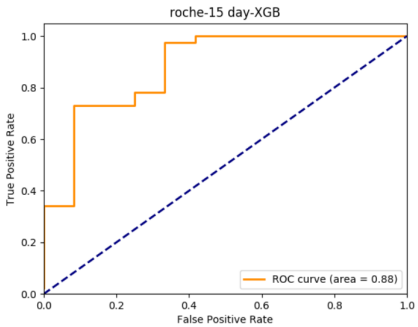

roche-15 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.90)

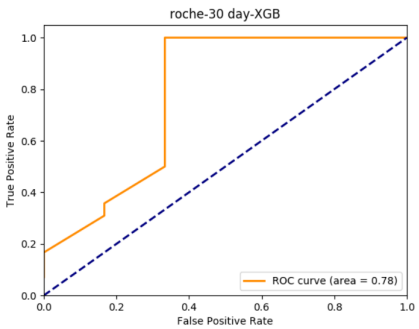

roche-30 day-RF
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)


roche-3 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)


roche-5 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)


roche-10 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.94)


roche-15 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.88)


roche-30 day-XGB
True Positive Rate
False Positive Rate
ROC curve (area = 0.78)

7. Comparison of Performance

In this section, we present the ROC plots of another popular classifier used in literature: support vector machines (SVM). We present a representative sample to facilitate the understanding of the efficacy of SVM in Figure 11, which is not an exhaustive set of ROCs; we see that unlike the case of RF and XGB, SVM (with a linear kernel) does not exhibit an improvement in performance with the trading window. Furthermore, the ROC curves are very close to the 50% line.

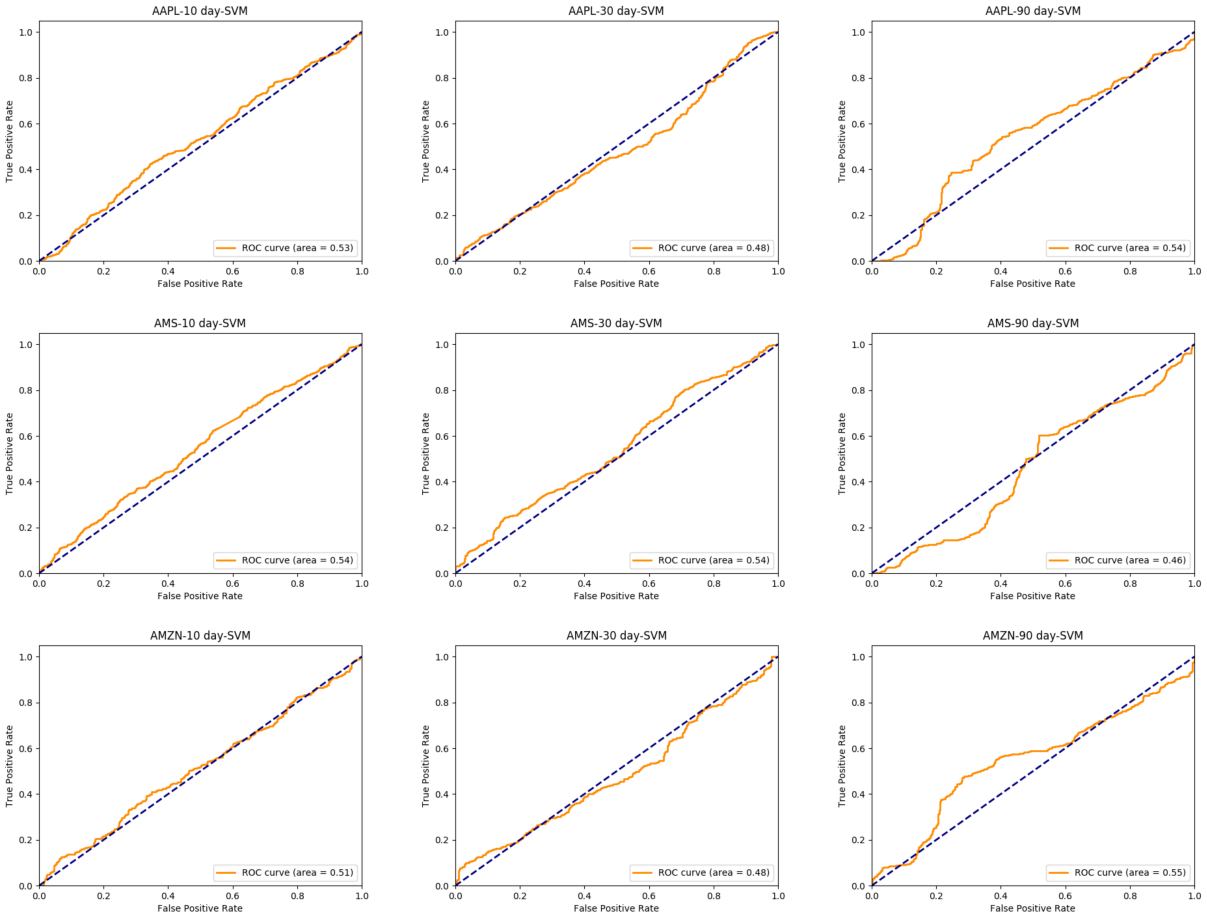


Figure 11: ROC curves of SVM applied to the dataset after preprocessing. From the graphs, it is evident that generally, the efficacy of SVM with a linear kernel is lower than that of RF and XGBoost.

8. References

Breiman, L. (2001), Statistics Department, University of California Berkeley, CA 94720. Random Forests.

Bylander, T. & Hanzlik, D. (1999) Estimating Generalization Error Using Out-of-Bag Estimates. AAAI-99 Proceedings.

Behner, P., Schwarting, D., Vallerien, S., Ehrhardt, M., Beever, C. & Rollmann, D. (2009). Pharmaceutical Companies in the Economic Storm Navigating from a Position of Strength. Technical Report: BooZ & Co Analysis.

Geurts, P., & Louppe, G. (2011). Learning to rank with extremely randomized tree. JMLR: Workshop and Conference Proceedings, 14, 49–61.

Horowitz, E., Sahni, S. & Anderson-Freed, S. (1992). Fundamentals of Data Structures in C, W. H. Freeman & Co., New York, NY, USA, ISBN = 0716782502

Quinlan, J. R. (1986). Induction of Decision Trees, MACH. LEARN 1, pp 81–106

Highlights

- Application of Machine learning to diverse portfolio of stocks
- Rich knowledge discovery from pharmaceutical stocks
- Reformulation of traditional forecasting problem as classification problem
- Classification models with average case prediction accuracy of above 90%
- Recursive error minimization by parameter tuning