Project Report: Evaluating Classifiers for Equity Performance Direction

Alexis Bruneau (1008704270)
Anton Korikov (1002223907)
Ramy ElMallah (1009480303)

CSC2515, University of Toronto
{alexis.bruneau, anton.korikov, ramy.elmallah}@mail.utoronto.ca

1 Introduction

Machine Learning (ML) has been increasingly used as a powerful tool to model and analyze complex phenomena. One major area where ML has increasingly shown potential is the prediction of equity performance. Historically, various methods, such as fundamental and technical analysis, have been used for predictions, but they are subjective and can be difficult to apply consistently. On the other hand, ML algorithms can help uncover patterns in large amounts of data that are not immediately clear to human analysts.

In recent years, various works [1][2] have studied the use of ML in predicting equity performance. Most applications of ML for market prediction focus on time-series forecasting [3]; however, such fine-grained time-series analyses are difficult to interpret for everyday individual investors who often do not have the time to monitor their investments daily, weekly, or even monthly [4]. In fact, both Graham [4], and Buffet [5], two of the most revered investors of all time, recommend that the lay investor should forgo investment techniques requiring continual portfolio oversight and instead follow a simple *buy-and-hold* strategy in which they identify high-quality investments and hold them for long periods of time. This mismatch between ML time-series formalisms and the simple buy-and-hold preferred by everyday investors thus makes it difficult for laypeople to use financial ML as part of their investing strategy. To address this problem, we explore ML classification methods to distinguish between stocks that make gains from those that take losses over a period of least one year, with the goal of investigating methods better aligned with a human buy-and-hold strategy and thus more conducive for joint human-AI decision making. We also show that when these simple classifiers are augmented with explainability techniques such as SHAP [6], the potential for joint human-AI decision-making is increased even further.

Using US market data from a five-year period, we thoroughly evaluate and compare three classifiers: random forests (RF), support vector machines (SVM), and neural networks (NN). Our study begins with an exploratory data analysis (EDA) which motivates several feature engineering steps (Section 2), and we then perform hyperparameter tuning and a hyperparameter sensitivity analysis for each classifier (Section 3). We then perform a SHAP [6] feature importance study which succeeds in identifying important features consistent with features deemed important in other publications. Section 5 investigates the effects of dimensionality reduction, and Section 6 explores the effects of variation in the training and prediction time horizon on each model's generalization performance. A unified discussion of the results is provided in Section 7, and related work is reviewed in Section 8.

2 Data Preprocessing and EDA

Our experiments are based on a public dataset [7] of US stock performance from 2014 to 2018. For each of the five years, this dataset includes 200 financial indicators obtained from the annual reports of 4000 public companies. As part of preprocessing and EDA, we cleaned this dataset to remove

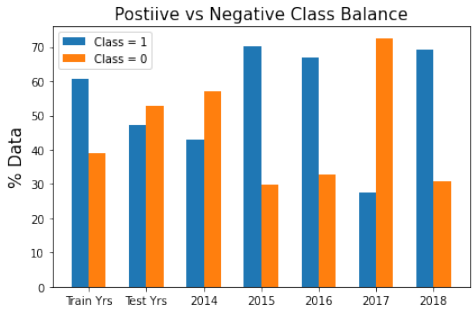


Figure 1: Class Balance

Figure 2: Sample Dendrogram Leaves

outliers and missing values, checked the balance of the target class and the cardinality of categorical features, dropped highly correlated features, and scaled and normalized the data. We also used a dendogram to check the semantic quality of the dataset and aggregated data across multiple years into a train and test set. Details for these steps are given below.

Multi-Year Aggregation Given financial data from year t as a set of features $x \in \mathcal{X}^t$, the target for the classification task is $y \in \mathcal{Y}^{t+1}$, $\mathcal{Y}^{t+1} \in \{0, 1\}$ which represents whether a stock's value increased or decreased in year $t+1$. This report explores several alternative ways to treat multi-year data (see Section 6), but the baseline approach uses the sets $\mathcal{X}^{TR} = \{\mathcal{X}^t\}_{t=2014}^{t=2016}$, $\mathcal{Y}^{TR} = \{\mathcal{Y}^{t'}\}_{t'=2015}^{t'=2017}$ for training and the sets $\mathcal{X}^{TE} = \{\mathcal{X}^t\}_{t=2017}^{t=2018}$, $\mathcal{Y}^{TE} = \{\mathcal{Y}^{t'}\}_{t'=2018}^{t'=2019}$ for testing. In this baseline method, predictions are always made given features from year t for stock performance in year $t+1$. No prediction or evaluation takes place for non-adjacent feature-prediction years.

Class Balance Figure 2 shows the balance of positive versus negative target classes in the dataset. Because the classes are not balanced, the F1 metric will be used for model evaluation. It is important to observe that the class balance is not the same between the training and test set, which will inherently limit generalization performance on the test set. However, it does not make sense to rebalance the train and test set, because real world data across multiple years will never maintain consistent class balances. The class imbalances also vary for each year which limits generalization performance across varying time horizons.

Dendrogram We use a dendrogram to check the semantic quality of the data. The dendrogram performs binary hierarchical clustering to produce a tree where each leaf is a feature, and features closer together in the tree are distributed more similarly than features which are far apart. Figure 1 shows a sample of leaves (features) resulting from our analysis, and it can be seen that semantically similar features are indeed close in the tree. For instance, Operating Cycle and Cash Conversion Cycle are adjacent leaves, and a set of five 10-year ratios are also a set of adjacent leaves.

Data Cleaning Our data cleaning involved removing outliers, imputing missing values, and removing features for which the percent of missing values exceeded a certain threshold. This threshold was set to 70% as suggested by Malakdar [8], and the remaining values were imputed using the median feature value. Given a feature distribution, outliers were defined as points outside the range $[Q1 - 3(IQR), Q3 + 3(IQR)]$, where $Q1, Q3$ are the first and third quartiles and IQR is the inter-quartile range, under the condition that the points outside this range made up fewer than 3% of the data. Figure 3 shows an example of outlier removal resulting in a more well-balanced feature distribution.

Correlation Analysis We performed a pairwise Pearson's coefficient correlation analysis on all numerical features. If two features had a correlation coefficient above a threshold of 0.9, the feature with the most missing values was removed. Figure 4 shows an example of two features, Net Income and Operating Income, with a correlation coefficient greater than 0.9. This example also provides a sanity check of the data quality since Net Income and Operating Income should indeed be strongly linearly correlated. A post-hoc analysis of model performance showed that dropping these correlated

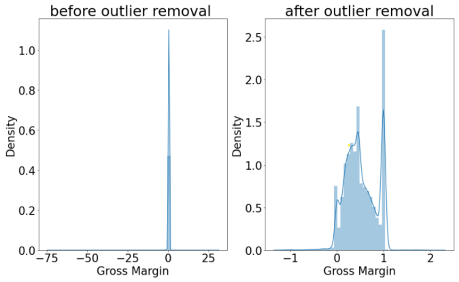


Figure 3: Example Effect of Outlier Removal on Feature Distribution

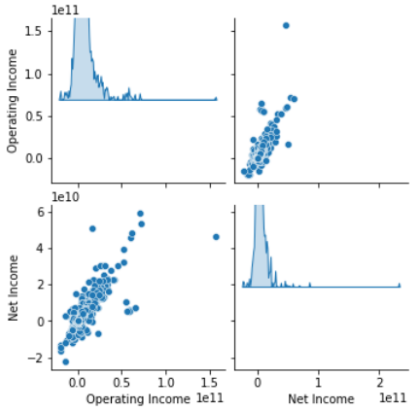


Figure 4: Joint Distributions of Net Income and Operating Income

features resulted in less than a 1% change in any metric while providing the benefit of producing smaller models.

Categorical Feature Cardinality The dataset includes one categorical feature: Sector (e.g. "Healthcare","Technology"). We confirmed that there were at least 10 observations for each category; indeed, the least frequent category occurred 445 times out of roughly 20,000 observations.

3 Classifiers Analysis

In this section, we present an analysis of the three baseline models selected, Random Forests (RF), Multi-Layer Perceptrons (MLP), and Support Vector Machines (SVM), which are trained and evaluated on the pre-processed data. Hyperparameter tuning has been conducted for each of the models to select the best hyperparameters, and a hyperparameter sensitivity study is presented.

3.1 Random Forests

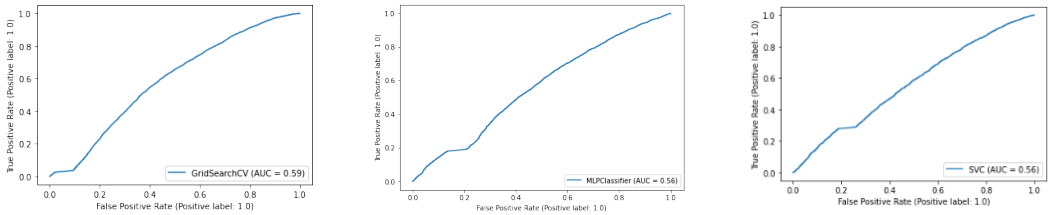
As Random Forest (RF) is known to be one of the good-performing ML algorithms for stock prediction tasks as demonstrated in [9, 10, 11], we include it as one of the algorithms in this study. The design choices we experiment with for RF include the number of trees in the forest and the number of randomly drawn candidate variables of each tree. These two hyperparameters are chosen as they are known to have the higher impact on RF performance [12] out of all RF tunable hyperparameters. We did a hyperparameter grid search using cross-validation on the training data with four folds to pick the number of trees out of the set of candidates{64, 128, 512, 1024, 2500} and to pick the maximum number of features per tree out of the set {4, 10, $\sqrt{\#features}$, 80, $\#features$)}. The objective of the hyperparameter tuning was chosen to be the F1-Score, as it is a more valuable metric for the problem at hand due to the imbalance of the dataset. The tuning resulted in a selection of 2500 trees in the forest with a maximum number of 4 features each. This combination resulted in the metrics values shown in Table 1. The Receiver Operating Characteristics (ROC) curve for the model is shown in Figure 5a, indicating that it is performing better than random chance.

3.2 Deep Learning (Multi-Layer Perceptron)

The second classifier explored was a multi-layer perceptron (MLP) classifier, for which an F1-based cross-validated gridsearch was performed for hyperparameter selection. Two activation functions were tested, logistic and rectified linear units (ReLU), with logistic activations performing best for all models. Several neural architectures were tested, each consisting of either three or four fully connected layers with between 30 and 100 neurons per layer; these architechtures followed either a uniform or an hourglass shape. The other hyperparameters tested were batch sizes in $\{50, 200\}$ and a constant versus adaptive learning rate, the later of which showed no effect. Finally, it was observed that too many training epochs resulted in overfitting, so early stopping was used. The best

Table 1: Baseline Models Performance Summary

Model	Accuracy	**F1-Score**	Precision	-ve Predictive Value	Recall	Specificity
Random Forest	0.549	**0.643**	0.513	0.685	0.861	0.270
MLP	0.541	**0.616**	0.509	0.624	0.777	0.330
SVM	0.472	**0.641**	0.472	NaN	1	0



(a) RF Model (AUC: 0.59) (b) MLP Model (AUC: 0.56) (c) SVM Model (AUC: 0.56)

Figure 5: ROC Curves for the baseline Models

hyperparameter values found as a result of the tuning process were a 4-layer uniform fully connected architecture of 50 neurons per layer with a batch size of 200, a constant learning rate, and logistic activation functions. The ROC curve resulting from the tuned MLP model had an AUC of 0.56 and is shown in Figure 5b.

3.3 Support Vector Machine

SVM has been a popular machine-learning algorithm method used to predict long-term stocks [13, 14, 15]. SVM is a classification technique that uses a decision boundary to separate both classes. That decision boundary is a hyperplane that tries to maximize the distance between the points of both classes. When the data is non-linear, the kernel method is used to classify nonlinear data by mapping the space into a higher dimension. The kernel trick uses dot products to modify the decision boundary. Some popular kernels are the following: linear, polynomial, Gaussian/radial basis function (RBF), and sigmoid.The type of kernel used is one of SVM's hyperparameters. Two other common ones are C (penalty parameter), and gamma. The penalty parameter penalizes misclassified points. A low value for C represents a low penalty assigned for misclassified points. This results in a larger decision boundary and more misclassification in the training set. A large value of C assigns a larger penalty to misclassified points. This results in a smaller decision boundary and fewer misclassified points in the training set. Gamma is used to tune the regions using RBF. As gamma decreases, the regions separating classes gets more generalized, and when it increases, the regions are more specific.

We wanted to test what was the combination of each parameter that resulted in the highest F1-score. We used the grid-search technique to find the best combination of those parameters. In practice, testing values between 10^{-3} to 10^{3} for C and gamma is sufficient [16]. For computational time reasons, we tested the following values: C and gamma values (0.1,1,100), Kernel function (RBF, polynomial, sigmoid). Using the F1-score, the hyperparameters that gave the best results were the following combination (C = 0.01, gamma = 0.01, and kernel = RBF). However, looking at 1, the recall obtained by SVM was 1, which means that it predicted all of the stock as a buy. Hence, the SVM method is not recommended.

3.4 Hyperparameter Sensitivity Analysis

Next, we conduct a study of the sensitivity of the models to the choice of hyperparameters. In this study, we reduce and increase each of the hyperparameters by 80% and re-train the model and evaluate it on the test data according to the F1-score. In Figure 6, we show that, in the case of the random forest model, these perturbations to the hyperparameter values cause changes to the F1-Score that does not exceed 0.5% of the previously selected optimal value (shown in the middle cell of the Figure). Similarly, a hyperparameter sensitivity analysis was conducted for the MLP and SVM models. For SVM, the change was exactly 0 for all perturbations indicating the model

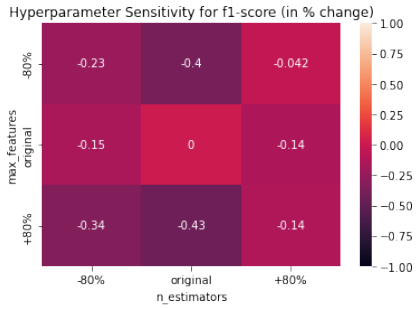


Figure 6: Random Forest Model Hyperparameter Sensitivity

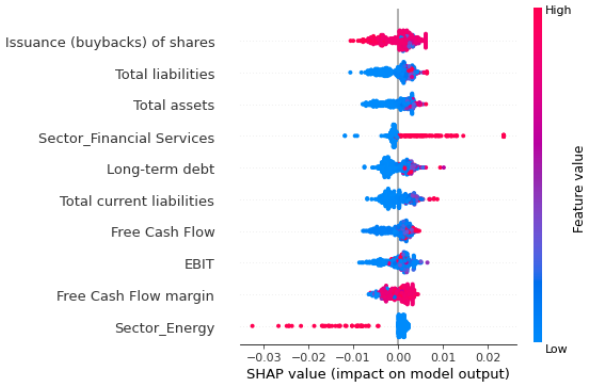


Figure 7: Top 10 most important features based on SHAP values analysis over 500 test data points

predictions on the test set did not change. In the case of MLP, the study was conducted for hidden layer sizes and batch sizes, and the change in F1-score was observed to vary up to 2.7% of the score for previously selected optimal values.

4 Feature Importance Study

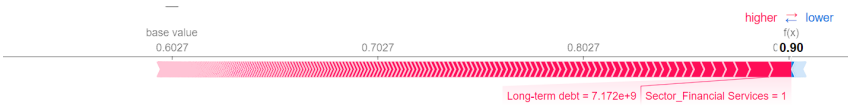
Due to the importance of explainability in ML, especially when it comes to making crucial decisions that could lead to detrimental consequences if wrong, we attempt to uncover feature importance used in our model with the help of SHAP values [6]. SHAP values are based on the concept of Shapley values, which measure the average marginal contribution of each feature to the overall score difference from the base expectation. SHAP values have been calculated for a subset of the test data using a trained RF model (to keep computational time under 1 hour), and a summary of feature contributions is shown in Figure 7. It could be seen that the financial services sector leads to higher predictability of stocks, which aligns with the conclusions in [2]. We present the results for RF only, as we ran into computational challenges and memory limitations in calculating SHAP values for both SVM and MLP.

To further explore the features that contributed most to making predictions, we also analyzed several individual samples in addition to the multi-sample aggregated results of Figure 7. For example, in Figure 8a, we show a visualization of feature contributions for one sample with positive ground truth and in Figure 8b, we show another for a sample with negative ground truth. We can see that both have been predicted correctly. In Figure 8c, we show a misclassified sample with negative ground truth. We can see that the model is drawing on meaningful patterns to make its predictions. For example, a high negative net income per share and a high negative free cash flow per share lead to a confident negative prediction. These explanations are important when such an ML model is used to guide decisions made by investors.

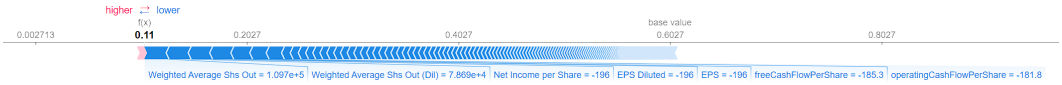
5 PCA Dimensionality Reduction Study

We explored the effect of using PCA dimensionality reduction on model performance. Given d initial features, PCA identifies $m < d$ principal components (linear combinations of features) to replace the initial features, with the goal of reducing the dimensionality and correlation of model inputs. Usually, the number of principal components used is the number required to represent either 90%, or 95% of the data variance, and in our case, we chose 95%. For 200 features, 107 principal components were required to describe 95% variance.

The effects of applying PCA are shown in Table 2: no significant changes from the baselines were observed. The F1-Score stayed the same for MLP and SVM and dropped for RF. All metrics for RF dropped: the F1-Score dropped by 0.014 from 0.643 to 0.629, and Specificity dropped by 0.045 from 0.27 to 0.225. Among the three models, the Recall for MLP was the only metric that increased, going up by 0.024 from 0.777 to 0.801. MLP's specificity dropped by 0.044 from 0.330 to 0.286, and no



(a) Most important features used to predict a correctly classified positive sample



(b) Most important features used to predict a correctly classified negative sample

(c) Most important features used to predict a misclassified test sample

Figure 8: Most influential features in making predictions based on SHAP values

Table 2: PCA-Based Model Performance

Model	Accuracy	F1-Score	Precision	Negative Predictive Value	Recall	Specificity
RF	0.524	0.629	0.497	0.638	0.857	0.225
MLP	0.530	0.616	0.501	0.617	0.801	0.286
SVM	0.472	0.641	0.472	NaN	1	0

changes were observed for the SVM model. Overall, the dimensionality reduction shows marginally worsened performance with no signs of any significant performance gains and is discussed further in Section 7.

6 Time Horizon Generalization Study

We conducted a time horizon study to investigate the generalization performance of our methods across time. The study is divided into two major parts: exploring the effect of varying training horizons and exploring the effect of varying prediction horizons. Note that precision is emphasized due to the nature of the application, where the selection of equities to buy is based on the model's positive predictions. The higher the true positives compared to all the predicted positives, the more likely an investor will be increasing their gains.

Varying Prediction Horizons with Fixed Training Horizons We explored the effect of varying the prediction horizon (i.e., the number of years predicted into the future) for a fixed training horizon. The classification task was predicting the direction of change in a stock's value at the end of the prediction horizon. As per the notation defined in Section 2 (see Multi-Year Aggregation), the training set was defined as $\mathcal{X}^{TR} = \{\mathcal{X}^t\}_{t=2014}^{t=2015}$, $\mathcal{Y}^{TR} = \{\mathcal{Y}^{t'}\}_{t'=2015}^{t'=2016}$. For the test sets, three prediction horizons were evaluated based on input features from 2016, $\mathcal{X}^{TE} = \{\mathcal{X}^t\}_{t=2016}^{t=2016}$, to make predictions one, two, and three years into the future. Here, a test target $y \in \mathcal{Y}^{t'}$ was defined as 1 if a stock's price in year t' was greater than its price in 2016 and 0 otherwise. This section of the study helps evaluate the ability of models to identify high-quality stocks for longer time horizons than one year. The results of the study are shown in Table 3. The results show that the stocks return direction performed similarly overall for the 1-Year and 2-Year horizons resulting in a relatively high and similar precision for all three models. However, performance fell for a three-year time horizon, likely due to changing market conditions and the fact that models did not have access to updated financial data. The three models behave in the same way in this study.

Varying Training Horizons with Fixed Prediction Horizons In this part, training is done on different time horizons while testing is done on fixed data. In other words, each model will have access to only the most recent past year of data in one case, the last two years of data in another case, and the last three years of data in the third case. The model is then used to make future predictions on the same two years of data. Thus, the test data is $\mathcal{X}^{TE} = \{\mathcal{X}^t\}_{t=2017}^{t=2018}$, $\mathcal{Y}^{TE} = \{\mathcal{Y}^{t'}\}_{t'=2018}^{t'=2019}$ and

Table 3: Prediction Quality for Different Future Horizons

	RF		MLP		SVM	
Prediction Horizon	F1-Score	Precision	F1-Score	Precision	F1-Score	Precision
1 Year	0.693732	0.694488	0.720512	0.724474	0.802994	0.670836
2 Years	0.693947	0.694488	0.720738	0.724474	0.802696	0.670419
3 Years	0.584361	0.49455	0.579619	0.492303	0.633528	0.463623

Table 4: Effect of Training Horizon on Prediction Quality

	RF		MLP		SVM	
Training Horizon	F1-Score	Precision	F1-Score	Precision	F1-Score	Precision
1 Year	0.643909	0.500881	0.624365	0.493744	0.641488	0.472198
2 Years	0.645223	0.498886	0.631468	0.492763	0.641488	0.472198
3 Years	0.643262	0.513437	0.615509	0.509423	0.641488	0.472198

the training data for the two-year training horizon is $\mathcal{X}^{TR} = \{\mathcal{X}^t\}_{t=2015}^{t=2016}$, $\mathcal{Y}^{TR} = \{\mathcal{Y}^{t'}\}_{t'=2016}^{t'=2017}$. This study helps uncover the sensitivity of the model to the amount of data it is trained on and the recency of that data. The study results are shown in Table 4. The results show that the SVM model has the same predictions regardless of the training data. The one and two years horizons act very similarly, indicating that the stock indicators and performance are similar over these two years. So, we mainly consider the difference between 2-year and 3-year training horizons. For RF and MLP, as the model trains on more data from the past, its precision improves, returning a larger number of good recommendations, although the F1-score drops slightly. This indicates that having access to older data does not guarantee an improvement in performance predicting future behaviour. In general, using training data from more years increases bias and reduces variance, which corresponds to models with higher precision and lower recall [17]: our results support these observations.

7 Discussion

Among the three classifiers compared, the best baseline model (Section 3) was RF (F1 = 0.643, Pr = 0.513). Though SVM achieved a similar F1 score to RF, it had a Recall of 1, meaning that it always predicted the positive class. Since such behaviour is unacceptable in real markets, we recommend against using our SVM model. Finally, the MLP classifier (F1 = 0.616, Pr = 0.509) achieved slightly worse but comparable results to RF. All models could be trained within 2.5 minutes (Table 5), with RF taking the most training time. For all three classifiers, our hyperparameter sensitivity analyses indicated little sensitivity to the hyperparameters values selected via grid search, with the F1-Score varying no more than 2.7% for $\pm 80\%$ variation in the hyperparameters tested. It was also found that PCA dimensionality reduction did not lead to significant changes. While dimensionality reduction produces simpler models that train faster, grouping features into principal components reduces explainability by making it more difficult to evaluate the importance of individual features. Thus, we do not recommend PCA for our stock classification models.

Since our primary motivation for taking a classifier approach versus a time-series approach was to better facilitate joint human-AI decision-making for everyday investors, the explainability analysis was crucial to our study. We showed that SHAP feature importance methods could be successfully applied to the RF classifier, producing results consistent with past work on feature importance in equity prediction [2]. Due to computational challenges, we were not able to generate explanations for SVM and MLP.

A key challenge for our models is generalizing across various training and prediction time horizons. For a fixed training horizon, we found that three year prediction horizon led to significantly degraded performance compared to one and two year prediction horizons. For fixed prediction horizons, increasing the training horizon improved precision but reduced the F1 score, which we attribute to a reduction in variance and an increase in bias resulting from the use of training data over longer time periods. All models had similar behaviour for the time horizon studies, indicating that the challenges of generalizing accross various time periods arise not from the assumptions underlying a given model, but from the temporal distribution of the data itself.

Table 5: Training Time complexity for Different Models

RF	SVM	MLP
138.08s	38.67s	92.19s

8 Related Work

Over the years, multiple machine-learning methods have been used for stock market forecasting. SVM, Random Forest, and Deep learning are among the methods used with Deep learning having an emergence in popularity in the last few years [13]. A publication from 2020 [14] compared the results of SVM, Random Forest, and ANN for long-term stock selection for China's Stock Market. Kernel, C, and gamma were tuned for SVM, and number of decision trees and the maximum number of features per tree were tuned for the Random Forest. Finally, the hyperparameters used for ANN were the number of hidden layer neurons, the optimization function, and the maximum number of iterations. Using accuracy as a metric, the following was their best model from best to worst: RF, ANN, and SVM with respective scores of (52.92%, 52.32%, and 51.73%). From their result, we can see that stocks are difficult to predict, and their results were not much higher than 50%. In another publication [18], the authors used Random Forest and SVM to predict whether a stock would be up by more than 10% over 1 year. Their data set consisted of S&P1000 stocks and used 24 common fundamental key metrics such as book value, Earnings Per Share as well as others. They achieved an F-score of 75.1% using Random Forest and 62.4% using SVM. From those papers, we can see that in general, Random Forest seems to a strong approach to predicting long-term stock performance. These papers also illustrate the difficulty of predicting stocks.

9 Conclusion and Future Work

We presented an EDA for the US stock performance dataset to clean and prepare the data for ML use. We demonstrated that financial data has many highly correlated features, such as Net Income and Operating Income, making EDA an important step for successful deployment to aid with explainability. We provided an analysis of three classifier models, namely RF, MLP, and SVM, to predict the equity performance direction to guide investors with stock-picking decisions for a buy-and-hold strategy. The analysis included model training with hyperparameter tuning, hyperparameter sensitivity study, and dimensionality reduction analysis. We have shown in the study of Feature Importance the applicability and potential of showing the features that steered the prediction the most, as well as generally ranking the features that increase the predictability when present. Additionally, the time horizon generalization study suggested that attention should be paid to the recency of the training data and the prediction horizons to ensure predictions are of higher quality.

Overall, we recommend RF as the best model architecture due to its overall performance and generalization abilities, in addition to being the best suited for the explainability techniques of the three methods while maintaining a reasonable training time complexity. Future work directions include analysis of more deep learning model architectures, studying additional market datasets to uncover patterns in the ease of predictability, and considering additional evaluation metrics to evaluate the variance in gains and losses. Another direction for future work is to study a regression problem which captures the magnitude of the change in stock value, not only the direction as done by our classification models.

10 Co-author Contributions Summary

The three co-authors have contributed equally to the project implementation and paper writing. The names are listed alphabetically. The baseline model analysis for RF, MLP, and SWM was conducted by Ramy, Anton, and Alexis. respectively. EDA was implemented mainly by Anton and Alexis, while the Feature Importance and Time Generalization Studies were implemented mainly by Ramy.

References

[1] Xiao Zhong and David Enke. Predicting the daily return direction of the stock market using hybrid machine learning algorithms. *Financial Innovation*, 5(1), June 2019.

[2] Yang Jiao and Jérémie Jakubowicz. Predicting stock movement direction with machine learning: An extensive study on sp 500 stocks. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 4705–4713, 2017.

[3] Marcos Lopez De Prado. *Advances in financial machine learning*. John Wiley & Sons, 2018.

[4] Benjamin Graham and Jason Zweig. *The intelligent investor*. HarperBusiness Essentials New York, 2003.

[5] Warren Buffett and Lawrence A Cunningham. *The essays of Warren Buffett: lessons for corporate America*. HeinOnline, 2001.

[6] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[7] Nicolas Carbone. 200+ financial indicators of us stocks (2014-2018), Jan 2020.

[8] Kishan Maladkar. 5 ways to handle missing values in machine learning datasets, Oct 2021.

[9] Luckyson Khaidem, Snehanshu Saha, and Sudeepa Roy Dey. Predicting the direction of stock market prices using random forest, 2016.

[10] Perry Sadorsky. A random forests approach to predicting clean energy stock prices. *Journal of Risk and Financial Management*, 14(2):48, January 2021.

[11] Manish Kumar and Thenmozhi M. Forecasting stock index movement: A comparison of support vector machines and random forest. *SSRN Electronic Journal*, 2006.

[12] Philipp Probst, Marvin N. Wright, and Anne-Laure Boulesteix. Hyperparameters and tuning strategies for random forest. *WIREs Data Mining and Knowledge Discovery*, 9(3), January 2019.

[13] Mahinda Mailagaha Kumbure, Christoph Lohrmann, Pasi Luukka, and Jari Porras. Machine learning techniques and data for stock market forecasting: a literature review. *Expert Systems with Applications*, page 116659, 2022.

[14] Xianghui Yuan, Jin Yuan, Tianzhao Jiang, and Qurat Ul Ain. Integrated long-term stock selection models based on feature selection and machine learning algorithms for china stock market. *IEEE Access*, 8:22672–22685, 2020.

[15] Junyoung Heo, Jin Yong Yang, et al. Stock price prediction based on financial statements using svm. *International Journal of Hybrid Information Technology*, 9(2):57–66, 2016.

[16] Rbf svm parameters. https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html, 2022.

[17] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[18] Nikola Milosevic. Equity forecast: Predicting long term stock price movement using machine learning. *arXiv preprint arXiv:1603.00751*, 2016.