Selection of Sustainable Dividend Stocks
Combining XGBoost with Genetic Algorithm

Francisco José Couto da Silva Instituto Superior Técnico
Lisbon, Portugal
francisco3597@gmail.com

Abstract—This work presents an approach which combines Extreme Gradient Boosting (XGBoost) with Genetic Algorithm (GA) to create a novel dividend stock selection system, capable of identifying the safest dividend growth stocks and avoid dividend cuts, using real regularly updated financial data of firms with ongoing dividend streaks from the S&P500. The proposed implementation defines new training windows yearly, on which XGBoost learns to classify how likely it is that a company ends their dividend streak in the space of one year, based on quarterly observations. The probability scores are then used to rank the companies under evaluation. For each window, the GA is used to find the best set of hyperparameters, based on the performance on a defined validation period of observations, in terms of ROC and PR AUC scoring. The system is able to yearly update the rankings and stock selections by generating new models and combining the classification scores with prior ones, in a sliding window fashion. The results were evaluated by analyzing the performance of the top ranking stocks in the year following scoring, using one or several combined models to generate scores. In this last case, an elitism parameter is introduced to the system, reducing the number of stocks returned by dropping the worst ranked stocks every year. For 2019, the system was able to select as much as one half of S&P500 stocks defined as having ongoing dividend streaks, while avoiding more than 80% of the companies with unsustainable dividends. Furthermore, the top performing stocks consistently generated annual total returns that outperformed those of the S&P500.

Index Terms—Dividend Investing; Dividend Policy; XGBoost; Genetic Algorithm; Fundamental Analysis; S&P500.

I. INTRODUCTION

In the world of financial markets, there are a large number of popular strategies investors use to generate profit. These strategies are adopted and used by different types of investors, according to their preferences in terms of risk tolerance, return timeframe and available funds. Among the most notorious investment strategies are growth, value and income investing.

Growth strategies revolve around purchasing more volatile stocks, often overpriced, stocks with high future growth perspectives, in hopes of profiting of said growth. On the other hand, value oriented approaches are long term strategies which consist on buying stocks with strong fundamentals, in theory underpriced, so that when the market catches up with real value the investor can profit on this adjustment. Lastly, income investing relies on building investment portfolios which generate regular streams of income, suited for those looking to accumulate wealth over time.

Dividend stocks are one the most popular sources of passive income for investors. Companies that pay dividends typically do so in a regular fashion and tend to increase the payouts at a steady rate. In particular, some dividend paying firms from the S&P500 have steadily increased their dividend for many years and decades, described as having ongoing dividend streaks. This historical track record makes them highly popular for dividend investors looking to compound their returns.

While a large portion of dividend stocks are seen as relatively safe and less volatile options by most investors, dividend investing is not as straightforward as it may look. This is mostly due to the sheer diversity of dividend paying firms progressing through different financial stages, shifting their dividend policies accordingly. The key to the success of a dividend stock portfolio lies in the selection of dividend stocks, in which the investor aims to include those which will keep on growing their dividend at a sustainable rate and avoid those which can in the near future cut their payouts. In particular, dividend cuts are among of the most troublesome events for investors since, in addition of reducing the level of passive income, drive stock prices down – specially in companies with long streaks of dividend increase or maintenance. This selection process therefore involves in-depth analysis of the firm’s fundamentals – making the act of predicting unsafe dividends non-trivial.

In recent years, with the increasing availability of financial data and the growth of the machine learning field, much work has been put into creating systems which use fundamental and technical analysis to help investors take profit in financial markets and improving their decision process. Other predictive systems have been used as pre-emptive tools, such as in the wide bankruptcy prediction field, which help investors identify poor fundamental structures before-hand and minimize loss.

In the same vein, this work proposes a new approach for a learning system based on XGBoost combined with Genetic Algorithm optimization, with the main goal of aiding the stock selection process made by dividend stock investors. The implemented system uses regularly updated fundamental and market based data to periodically evaluate and rank stocks, in terms of their future dividend sustainability and growth potential. This is done by yearly training and optimizing XGBoost models with past data from dividend paying firms of the S&P500, and using the probability scores to rank the firms, allowing investors to select the safest dividend stocks at all times.


II. RELATED WORK

A. Dividend Policy

Dividend policy is the set of decisions and rules taken by the board of directors with respect to dividend distribution. In order to find what drives managers to change their dividend policies, early studies made by Lintner [1] conducted interviews, finding that they mostly look at current earnings and to the target level of dividend payout when making the decisions, which were made conservatively. Brav [2] furthered this study in a more modern setup, conducting interviews and surveys on public firms from several sectors. The study finds that maintaining the historic track record, stability of future earnings projections and sustainable changes in earnings are among the key factors in the decision process. Brav [2] also asserts that managers tend to avoid reducing the dividend yield, and for that look at recent values from previous quarters, while also trying to maintain a smooth stream from year to year. Lintner [1] argued in favor of this smoothing in dividends, stating that it was an attempt to separate earnings volatility from the payout.

With respect to financial condition, other studies by Grullon [3] found statistically significant declines in leverage for dividend raising firms. DeAngelo [4] conducted a study on dividend payers which cut or omitted their dividend, finding that more than a half of the sample firms had eventually binding debt covenants in the year of the first dividend reduction, whereas others in the years of later reductions. These previous studies highlight the importance of leverage on dividend sustainability.

Due to the importance of earnings as a determinant for dividend policy, other studies focused on assessing earnings quality in regular dividend payers. Tong [5] measured quality in earnings with regard to discretionary accruals and in the mapping of accruals into cash flows, which are argued to reflect the current operating performance accurately. These enforce the previous studies, meaning that earnings and their quality are relevant determinants of dividend sustainability, since it is costly to to support cash dividends that do not reflect underlying performance and cash flows [5].

B. Machine Learning and Dividend Prediction

Machine Learning is one of the largest fields in Artificial Intelligence, which focuses on developing systems that learn and improve without being explicitly programmed to do so. It is a statistical data driven approach, since it uses large datasets to identify complex patterns and dependencies which are then used to build models that can be applied future data observations. Supervised learning algorithms are, in particular, one of the single largest families of machine learning algorithms and a popular field of research. The applicability of the “learn by example” framework and the increase in access to big structured datasets explain the recent rise in popularity. Supervised learning algorithms have been applied with success to several problems in the field of finance, ranging from stock market time-series prediction to dividend and bankruptcy prediction [6], [7].

Most machine learning studies so far on the subject of dividends were focused on dividend prediction and dividend policy change prediction. One of the first models developed for predicting dividends was introduced by Marsh & Merton [8]. This was a simple yet effective regressive model for predicting future dividends, which used past and present stock price and dividend historic data to predict dividends. Also using price and dividend data, more recently Kim [9] used Classification and Regression Trees (CART) with Knowledge Integration (KI) for predicting future dividends. This approach had the upside of deriving decision rules which could be interpreted, contrasting with other algorithms where knowledge is buried in weights and parameters. This work showed significant results, for different tolerance levels of accuracy. Subsequent studies by Won [10] used Genetic Algorithm based Knowledge Refinement (GAKR), which consisted on running multiple rule based algorithms (CART, CHAID, C5.0 and QUEST) and using GA to generate the optimal subset of rules for dividend policy prediction, using binary classification between dividend maintenance, if $D_t \leq D_{t+1}$, or reduction, if $D_{t+1} < D_t$. The results showed that the accuracy of the GAKR was on average higher than the underlying rule generating algorithms, with less generalization errors.

Other studies predicted dividends based on fundamentals instead of sole price and dividend history. These include Laoh [11], which used Neural Networks with GA parameter optimization in order to predict dividends, using financial ratios of the dividend payment year. Luebke [12] tested a set of machine learning algorithms to predict dividend change patterns of maintenance, increase and decrease. In particular, they compare the accuracy of the SVM, CART and Random Forests with traditional methods such as Linear Discriminant Analysis. The CART models allowed to evaluate feature importance, which showed that net income, market-to-book value and turnover growth rate had the strong predictive power. Hobbs [13], on the other hand, studied predictability of long term dividend streak maintenance. The study aimed to group dividend initiators in short term and long term payers, respectively those that are able to sustain their dividend for less than 3 years or more than 7 years. They were able to correctly label 150 out of 243 initiators, based on comparison between pre-initiation performance, using the volatility of profitability (risk adjusted Return-on-Assets (ROA) [3]) and logistic regressions using financial features such as firm size, dividend yield and debt ratios. Remarkably, none of the previous studies on dividend policy focused on the use of predictions to select stocks, from an investing and sustainability standpoint.

Some studies point out that dividend cuts and omissions are the first state in a continuum of financial distress steps, being followed by defaults on loan payments and bankruptcy [14]. In this context, many state-of-the-art algorithms have been applied to fundamental financial datasets and bankruptcy prediction became one of the most researched topics during the last years.

In this field, some of these recent works have found success by using Gradient Boosting algorithms, specially XGBoost. Le [6] applied XGBoost with GPU integration to bankruptcy forecasting, using two balanced and one imbalanced financial
ratio based datasets, comparing the proposed solution with other state-of-the-art gradient boosting algorithms. The results showed that XGBoost both consistently outperformed the other approaches in terms of ROC AUC score and in terms of computation time. Huang [15] tested 6 state-of-the-art supervised and unsupervised algorithms for overall financial distress prediction based on financial ratio data and found XGBoost to be the overall better performing classifier in terms of accuracy and type I and II errors.

III. IMPLEMENTATION

The implemented system has the goal of selecting S&P500 dividend stocks with ongoing dividend streaks by iteratively training, optimizing and combining XGBoost models, using real regularly updated financial statement data. These models are then used to assign probability scores correspondent to the firm’s likelihood of dividend increase or cut their in the following year, which creates the base of the stock selection.

Fig. 1 describes the high level architecture of the system, including the modules and the flow of data communication between them.

Fig. 1: Summarized high level description of the implemented system, including the modules and the flow of data communication between them.

The retrieval of historical price and dividend data from the web is performed by the market scraper module of the system. It retrieves closing price data according to the selected historic period and firm ticker, specified as parameters. The scraping process itself is performed with YahooQuery Python library, which provides fast and simple access to Yahoo Finance data. For dividends, the web scraping process was implemented manually, using Python Requests module. Stock split data was also scraped, in order to correct dividends, as they affect the real value of payouts – for example, in a 2:1 stock split the dividend is likely to be roughly halved past the split. To mitigate this problem, which is bound to affect the process of evaluating dividend streaks, the dividends are adjusted prior to the split by dividing them by the split fraction.

C. Financial Ratio Generation

The ratio generation module of the system takes the quarterly dataset and produces financial ratios and other derived data items from the financial statement data. It is responsible for building the features to be used by the algorithm in later steps of the data flow.

The choice of features was inspired on the studies described in Section II, conditioned by the availability of the items necessary to compute them. The features are presented in Table I, grouped by the performance measure they evaluate. Following the addition of all data items the data is cleaned, removing unnecessary columns, and returned.

D. Data Update

In order to update the data used by the system, the system retrieves new financial statements from the web and appends them to the existing data.

The web scraping sources used were Macrotrends and Yahoo Finance, due to the availability and reliability of their financial statement data. The simplified flowchart of the data update process is represented by Fig. 2.

1Stock splits divide existing shares by a given ratio, improving liquidity.
TABLE I: Selected financial features and respective computation methods. The ratios are grouped by the performance measure evaluated, together with some of the studies that inspired the feature selection.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Profitability</strong></td>
<td></td>
</tr>
<tr>
<td>Firm Size</td>
<td>$\log(\text{Assets})$</td>
</tr>
<tr>
<td>Firm Size</td>
<td>$\log(\text{Assets})$</td>
</tr>
<tr>
<td>Market-to-Book</td>
<td>$\frac{\text{Shares Outstanding \cdot Price}}{\text{(Assets - Liabilities)}}$</td>
</tr>
<tr>
<td>Dividend Yield</td>
<td>$\frac{\text{Dividend-per-share}}{\text{Price}}$</td>
</tr>
<tr>
<td>Accruals Ratio</td>
<td>$\frac{\text{NOA(t) - NOA(t-1)}}{\text{Mean[NOA(t), NOA(t-1)]}}$</td>
</tr>
<tr>
<td>Sloan Ratio</td>
<td>$\frac{\text{([Earnings - OCF - ICF])}}{\text{Assets}}$</td>
</tr>
<tr>
<td>Growth Rate</td>
<td>$\frac{\text{[Revenue(t) - Revenue(t-4)]}}{\text{Revenue(t-4)}}$</td>
</tr>
<tr>
<td>Revenue Growth</td>
<td>$\frac{\text{[Revenue(t) - Revenue(t-4)]}}{\text{Revenue(t-4)}}$</td>
</tr>
<tr>
<td>Earnings Growth</td>
<td>$\frac{\text{[Earnings(t) - Earnings(t-4)]}}{\text{Earnings(t-4)}}$</td>
</tr>
<tr>
<td>Dividend Growth</td>
<td>$\frac{\text{Dividend(t) - Dividend(t-1)}}{\text{Dividend(t-1)}}$</td>
</tr>
<tr>
<td>Sustainable Growth</td>
<td>$\text{ROE} : (1 - \text{DPR})$</td>
</tr>
<tr>
<td>Debt Ratio</td>
<td>$\frac{\text{Liabilities}}{\text{Assets}}$</td>
</tr>
<tr>
<td>Debt-to-Equity (DER)</td>
<td>$\frac{\text{Liabilities}}{\text{Shareholders' Equity}}$</td>
</tr>
<tr>
<td>Debt-to-EBITDA</td>
<td>$\frac{\text{Liabilities}}{\text{EBITDA}}$</td>
</tr>
<tr>
<td>LTD-to-Assets</td>
<td>$\frac{\text{LTD}}{\text{Assets}}$</td>
</tr>
<tr>
<td><strong>Liquidity</strong></td>
<td></td>
</tr>
<tr>
<td>Current Ratio</td>
<td>$\frac{\text{Current Assets}}{\text{Current Liabilities}}$</td>
</tr>
<tr>
<td>Quick Ratio</td>
<td>$\frac{\text{Net Income}}{\text{Current Liabilities}}$</td>
</tr>
<tr>
<td>Cash Ratio</td>
<td>$\frac{\text{Cash and Equivalents}}{\text{Current Liabilities}}$</td>
</tr>
<tr>
<td><strong>Leverage</strong></td>
<td></td>
</tr>
<tr>
<td>Asset Turnover</td>
<td>$\frac{\text{Revenue}}{\text{Mean[Assets(t), Assets(t-1)]}}$</td>
</tr>
<tr>
<td>Receivables Turnover</td>
<td>$\frac{\text{Revenue}}{\text{Mean[Receivables(t), Receivables(t-1)]}}$</td>
</tr>
<tr>
<td>Inventory Turnover</td>
<td>$\frac{\text{Revenue}}{\text{Mean[Inventory(t), Inventory(t-1)]}}$</td>
</tr>
<tr>
<td>Earnings Quality</td>
<td>$\frac{\text{([Earnings - OCF - ICF])}}{\text{Assets}}$</td>
</tr>
<tr>
<td>Derivative Growth</td>
<td>$\frac{\text{[DER(t) - DER(t-4)]}}{\text{DER(t-4)}}$</td>
</tr>
<tr>
<td>Turnover</td>
<td>$\frac{\text{Revenue}}{\text{Mean[Assets(t), Assets(t-1)]}}$</td>
</tr>
</tbody>
</table>

**E. Data Preprocessing**

In this step, a set of preprocessing transformations are applied to the data, which are necessary to train and build the predictive model, taking in the fully assembled and validated dataset from the data layer. The main operations performed in this step are:

1) Filtering the records by their dividend paying status, such that only observations of firms with ongoing dividend streaks of 3 years are kept;

2) The prediction label is defined, such that each observation is classified as positive if dividend streak is maintained in the following year, or negative if not;

3) The sector categorical variable is converted into a feature, with the one-hot encoding scheme, due to incapability of XGBoost of handling purely categorical variables as is

4) Removal of observations containing more than 20% of NaN from the data.

**1) Data Partitioning and Validation:** After processed, the data is split in different folds. The splits are done accordingly to the release date of the financial report that originated the data observations, such that a 2018 financial report is considered as such if released during that year. The data is first split into the following folds:

- **Training set**, used to fit an initial model on the financial data;
- **Validation set**, used to evaluate the performance of the model for different hyperparameter setups;
- **Test set**, or evaluation set, which comprises the data observations to be scored and evaluated by the final model.

Since the goal of the system is to use historic data of time-series nature to predict future dividend policy, the data splitting process is non-trivial as $n$-fold cross-validation cannot be used. Instead, the “walk-forward” approach is taken.

Fig. 2 represents the data window splitting process. The definition of the window is based on the testing period, given by $t_{\text{evaluation}}$. Following the example, setting $t_{\text{evaluation}}$ to 2018 uses data observations from 2009 to 2014 for the initial training, from 2016 to 2017 for validation and scores observations from 2018. The system leaves a 1 year gap between sets in order to increase the independence between folds. As described in Fig. 3, an additional fold is generated, which aggregates the initial training and validation sets. This set is used to retrain the model after the model parameters are learned.
F. XGBoost

Extreme Gradient Boosting (XGBoost) [21], is an ensemble based supervised learning algorithm derived from the original Gradient Boosting Machine (GBM) algorithm [22]. The GBM combines traditional boosting with the Gradient Descent algorithm, by connecting stepwise additive expansions to the steepest descent minimization [22]. XGBoost adds new improvements to this implementation, such as support for ridge and lasso regularization and performance improvements by parallel tree splitting.

The instantiation and training of XGBoost models is handled by the XGBoost module. The module takes as input the training data, the evaluation data and the model parameters, building the binary logistic XGBoost classifier and returning the list of scores given to each financial observation to score.

XGBoost supports several hyperparameters which control the general behaviour of the model, the boosting process or the learning task. While the general and learning setup is usually predefined, tree booster parameters should be fine tuned in order to adapt the model to the particular problem and data. These parameters significantly change the behaviour of the algorithm, for example by controlling the sequential weights given by base learning trees during the boosting process. The set of parameters to be defined, as well as their respective role, are represented in Table II.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>n_estimators</td>
<td>Number of weak learners used</td>
</tr>
<tr>
<td>eta</td>
<td>Learning rate</td>
</tr>
<tr>
<td>min_child_weight</td>
<td>Minimum sum of instances weight in a child</td>
</tr>
<tr>
<td>max_depth</td>
<td>Maximum depth of the tree</td>
</tr>
<tr>
<td>gamma</td>
<td>Minimum loss function reduction required for splits</td>
</tr>
<tr>
<td>subsample</td>
<td>Subsample ratio of training observations</td>
</tr>
<tr>
<td>colsample_bytree</td>
<td>Subsample ratio of training features</td>
</tr>
<tr>
<td>alpha</td>
<td>L1 regularization term</td>
</tr>
<tr>
<td>lambda</td>
<td>L2 regularization weight</td>
</tr>
</tbody>
</table>

When running the model, the class imbalance is evened by using the scaling parameters scale_pos_weight. This parameter balances the data patterns by changing the weights applied to the positive instances during training, which correspond to the minority class in the system. As suggested by Chen [21], this value is set to the ratio between negative instances and positive instances in the initial training set, during validation.

G. Genetic Algorithm

Finding the best set of machine learning inner model parameters, or hyperparameters, can be a slow and hard process. This is partially caused by the time overhead involved in evaluating a solution, which involves training a model and evaluating performance on a validation set. In particular, due to the large number of parameters to be tuned in XGBoost, this problem is exacerbated.

In problems with wide parameter search spaces the GA has proven to show the best results, both in terms of time and quality of solutions [23]. As such, it has been recently used in the optimization of neural networks and gradient boosting algorithms parameters [11] [24].

The GA is a search optimization algorithm which draws inspiration from the “survival of the fittest” principle. It uses evolutionary operators such as selection, crossover, mutation and fitness to produce the solution to optimization and search problems [25]. The GA module is implemented with the DEAP library, which provides the base functionalities for Python evolutionary algorithm prototyping, such as out of the box support for most crossover and selection functions. The flow of the module is described by the diagram in Fig. 4.

![Flowchart for the GA execution, as implemented by the GA module.](image-url)

The specific chromosomes of the individuals, which correspond to the parameters defined in Section III-F, must first be defined. The set of parameters to be optimized are encoded in...
a chromosome as a fixed length array of size 9, composed of floats and ints, as represented by Fig. 5.

At each iteration, the newly created offspring are evaluated in terms of fitness. The fitness function is obtained by fitting an XGBoost model using the parameters encoded in an individual. This process is handled by the XGBoost module, described in Section III-F. The model is fit to the initial training data and the respective performance is evaluated on the validation set, from which the fitness values are obtained for a particular setup of parameters. The performance score reflects the extrapolation capabilities of the model under a set of parameters, diagnosing underfitting and overfitting of the model. The system implements two distinct evaluation metrics, which take in the probability scores given to validation set observations and return a single score to be maximized. The two scoring metrics are the ROC AUC and the Precision-Recall AUC (P-R AUC), which were employed due to their applicability to the specific problem. Both of them take in the probability scores of the model and not the prediction itself, evaluating the discrimination between classes of observations, the key difference being that the P-R AUC gives more importance to the positive class.

After the fitness is evaluated, the most fit individuals are sequentially stored - as described in the diagram illustrated in Fig. 4. The design choices for selection and crossover methods were based on a batch of trial and error tests performed, in which tournament selection and uniform crossover yielded the best validation results. The mutation step was implemented by resampling the encoded value from an uniform distribution. Finally, the process stops when either the maximum number of iterations, set to 50, is reached or there are no score improvements for the past 5 consecutive iterations.

The module outputs the best parameters found for the training window, which can now be used to refit the model.

**H. Stock Ranking**

The stock ranking module takes in the probability scores given by the model to the test observations and ranks the dividend stocks.

This is done by first grouping the model scores by company and averaging the values. The averaged scores are then listed and sorted by company, in ascending order, with lower scores signalling lower averaged probability of dividend cuts in the following year and higher scores unsustainability of the dividend. After the stocks are ranked, the fraction $p$ of top ranking companies are returned, where $p$ is input as a parameter. The ranking process is described by Fig. 6.

**I. Sliding Window Setup**

The predictive layer modules described above implement the system features needed to score and rank the firms based on a single train-test window period. Each iteration of the sliding window can be summarized as follows:

1) The evaluation year $t_{evaluation}$ is retrieved and the training and validation periods are defined;
2) The GA is used to search for the best XGBoost hyperparameters, according to validation performance;
3) XGBoost is trained with the obtained parameters
4) The probability scores given to evaluation period observations are normalized and stocks are ranked;
5) The rankings are filtered, such that only $p$ percent of the top ranking stocks are kept;
6) The scores are then updated according the moving average, so that the current score of a company is averaged with the past scores given to that same company.

**IV. Results**

This chapter has the goal of presenting and evaluating the results obtained from the developed system, in different scenarios.
A. Single Window Dividend Increase Stock Ranking

In this case study, the system was set up to score the dividend stocks based on a single training window, scoring financial observations issued during 2018, while performing training and validation on the preceding years - as in Fig. 3. The stocks were then evaluated for 2019, both in terms of dividend policy change and stock returns. The evaluation sample included the total of 313 firms which paid increasing annualized dividends 3 years up to the end of 2018. From this set of firms, a subset of 15 did not manage to increase their dividend in 2019.

Two distinct models were obtained, trained using the best hyperparameters returned by each of the two GA fitness metrics, and then used to score the stocks. In order to evaluate the quality of the results, the ranking positions of the 15 firms that broke their dividend streak in the following year is assessed. The placement of these stocks is key to the performance evaluation of the system, as the goal is to rank these stocks as low as possible.

Fig. 8 shows the distribution of the streak breakers among the rankings. The results notably show that, for the P-R AUC objective, only 2 dividend streak breakers rank in the top 50%, which corresponds to 13% of the total number for that year. The results are closely similar to those obtained with the ROC AUC objective, which place one more dividend streak breaker in the top 50%. Furthermore, the bottom 25% includes 9 out of the 15 dividend streak breakers in both cases. With the ROC AUC objective, it is noticeable that in the tail end there is a larger concentration of streak breaking firms, with approximately half of them ranking in the lower 10%. In general, the results show that the model developed by the system has a strong discriminative power, being able to preemptively identify which firms are going to end their dividend streak.

The total price and dividend Return-on-Investment (ROI) of the top ranking stocks was also assessed and compared with the S&P500 index annual returns. In order to do so, the price weighted average value of several portfolios composed of the top ranking stocks was computed, for $n = 10$, 25, 50 and 100. The portfolios were generated following the evaluation period, and the adjusted ROI was computed throughout the following year. Fig. 9 shows the returns of the defined portfolios using the top ranking stocks, using both ROC AUC and P-R AUC scoring metrics. On the one hand, Fig 9(a) shows that the least diversified portfolios with the top 10 and 25 ranking firms generated returns as much as 10% above S&P500 compound returns. On the other, Fig. 9(b) shows that the top 10 and 25 returns underperform those generated by the top 50 and 100.

As a way to identify the financial determinants of dividend sustainability, the feature importance rankings were extracted from the XGBoost module, for both models trained. The average gain and weight of the 15 top ranking features is displayed in Fig. 10, for one the models trained. The plot represents the ranked features by F-Score, which is essentially defined as the average error decrease attained by adding tree splits on the specific features.

The results show that the sector features rank consistently among the most predictive features in terms of average gain. Even though their usage frequency in the trees is low, the sectors features seem to be predictive of future dividend policy changes when used. The other financial features consistently identified as strong predictors of dividend policy were the dividend yield, revenue and earnings growth, firm size, debt-to-EBITDA and current ratio.

B. Sliding Window Dividend Increase Stock Ranking

The case study discussed in this section uses the sliding window module, described in Section III-I, to generate rankings based on multiple predictive windows of data.

The sliding window stock ranking was executed with 3 evaluation years ($t_{evaluation} \in [2016, 2018]$) and $p$ initially
set to 1. The P-R AUC score was selected as the GA fitness metric for the sliding window study cases. This scenario can be interpreted as an historical simulation of performance, in which the system is first executed in the end of 2016, ranking the best stocks for the following year, then repeating the process one year after and then another, readjusting the ordering of stocks along the way.

The results obtained during the execution of the sliding window are presented in Table III. They show, for each iteration, the number of sample firms with ongoing streaks, the total number of streak breakers in the following year and the performance of the ranking system each year at identifying and ranking those firms.

To evaluate the returns of the top ranking stocks obtained with the sliding window, the ROI on the year following the final rankings was computed for several portfolios containing those stocks, similarly to Section IV-A. The resulting returns for 2019 are represented in Fig. 11 showing that again, the selected groups were able to again beat the compound S&P500 returns.

In order to test the impact of $p$ in the results, the system was further tested with a range of increasing $p$ thresholds. The performance of the final rankings generated by the system was evaluated for 2019, similarly to Section IV-A, by checking the number and distribution of dividend streak breakers. Table IV shows the number of companies included in the final rankings, as well as the number and distribution of those which break their dividend streak in 2019, for increasing values of $p$, which progressively decrease the elitism of the selection system. Furthermore, as the rankings of each year are intercepted, setting $p = 1$ still selects less than the overall 313 firms due to the increase of dividend streak requirements. The results suggest that dropping 30% to 50% of the bottom ranking firms at every iteration, such that $p \in [0.5, 0.7]$, would allow to select up to a half of the firms, while avoiding 80% of those which will cease increasing their payouts.

The stock rankings produced by the sliding window setup were also benchmarked against the standard single window setup from Section IV-A, by assessing the cumulative distribution of firms that break the dividend streak among the $n$ best, for 2019. This comparison allows to study if it is preferable to select the $n$ top ranking stocks using a single training window or combining multiple predictive windows, for different values of $n$. From the results depicted in Fig. 12:

- Using the sliding window with $p = 0.6$, for $n = 100$, a single non-increase is selected by the system;
- Using the sliding window with $p = 0.8$, for $n < 120$, only 1 non-increase is selected by the system, which is also a slight improvement over the single window. For $120 < n < 150$ the results worsen, since 3 of these firms are selected by the sliding window. For $n > 150$, the results show that picking the $n$ best stocks for $p = 0.8$ or from the single window returns similar results;
- The sliding window with $p = 1$ yields the larger number of ranked stocks for this setup. The results are similar to those obtained with $p = 0.8$ for $n < 180$, but short after there is a spike in the number of streak breakers selected.

On the one hand, the sliding window combination achieves some improvements by intersecting the rankings and increasing the dividend streak requirements, which decreases the number of non-increasers included in the rankings. On the other hand, the results show that, when selecting up to 200 stocks, the moving average of model scores does not provide significant improvements over using a single model to rank the stocks. This supports that, for the evaluation period tested, a single window of training is able to achieve very similar positive results in identifying the unsustainable dividend stocks.

C. Sliding Window Dividend Maintenance Stock Ranking

To further test the capabilities of the system, the definition of dividend streak for the sample observations was modified to the maintenance of the current dividend. This imposes that every observation included in the data has instead a 3-year
streak of dividend maintenance. Additionally, each observation was also labeled in terms of future dividend maintenance.

Similarly to the previous section, the sliding window setup is employed with 3 evaluation years, such that \( t_{\text{evaluation}} \in [2016, 2018] \), using the P-R AUC GA fitness scoring. In order to first test the baseline performance of the system, \( p \) was initially set to 1, similarly to the previous case study. Table V shows results obtained during execution of the sliding window.

The results generated by the model show a significant discriminative power, as more than half of the the dividend cutters land on the bottom 50%. In particular, looking at the final rankings generated after the last evaluation year \( t_{\text{evaluation}} = 2018 \), the 4 dividend cutters classified in the bottom 25% include the only firm which would end up ceasing payouts in the following year (GME), as well as the two larger cutters of 2019 (NLSN and PBI). This shows the system was able to detect beforehand the worst dividend traps, scoring them among the unsafest.

Similarly to the previous section, the price weighted returns of the top ranking firms were computed for the periods following each evaluation year. The results for 2019 (Fig. 13) show that the selected dividend stocks largely outperform both the index and the returns obtained in Sections IV-A and IV-B during the same period, with the top 25 and 50 groups generating annual returns above 40%.

Table VI shows the number of companies included in the final rankings, as well as the number and distribution of those which cut their dividend in 2019, for increasing values of \( p \).

![Fig. 13: Returns of top ranking dividend stocks in terms of dividend maintenance, benchmarked against S&P500 in 2019 and using the sliding window setup with \( p = 1 \). The returns are adjusted for dividends and splits](image)

Overall, the results have shown very positive stock selection outcomes, in particular for \( p \leq 0.8 \), which offered a good tradeoff between number of stocks selected and risk of experiencing a dividend cut in the following year. This means that the system was able to select up to 226 out of the total 348 stocks of firms with ongoing dividend maintenance streaks, excluding 75% of the firms that would end up cutting their dividend in the following year.

V. Conclusion

The results obtained from the developed system lead to the conclusion that XGBoost can achieve outstanding results in the field of dividend policy prediction.

First, using a single model to rank stocks with respect to sustainability of future dividend increase, the results obtained were very positive for both of the two GA objective functions tested. While in terms of returns the ROC AUC yielded the best results, the P-R AUC metric returned a smaller number of streak breakers for a larger percentage of top ranking stocks, placing 80% of these firms under 50%.

Furthermore, the sliding window setup combined the rankings generated by each XGBoost model iteratively for 2016, 2017 and 2018. The intersection of the top \( p \) ranking firms for each evaluation year made the system selections more elitist, reducing the overall number of firms ranked and filtering some of the firms with stale dividends - which brought by itself some improvement. The final stock rankings were evaluated by assessing performance of the scored stocks in terms of dividends and overall returns, throughout 2019. Overall, selecting up to 200 stocks, the results were very similar between the setups tested, with small differences in terms of streak breakers selected, which gave in some cases advantages to the combined use of XGBoost models. In some other cases,
TABLE V: Results obtained by the rankings iteratively generated by the sliding window setup from 2016 to 2018, with $p = 1$. The table shows the number of firms with an ongoing streak of dividend maintenance at year $t_{evaluation}$, the number of those firms which cut the dividend in $t_{evaluation} + 1$ and the performance of the system in predicting them, as measured by how those firms were ranked.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$t_{evaluation}$</th>
<th>Nr. of Firms Included</th>
<th>Dividend Breakers in $t_{evaluation} + 1$</th>
<th>Distribution of Breakers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Top 25%</td>
</tr>
<tr>
<td>1</td>
<td>2016</td>
<td>382</td>
<td>19</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2017</td>
<td>359</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2018</td>
<td>348</td>
<td>8</td>
<td>1</td>
</tr>
</tbody>
</table>

when selecting a large number of stocks, it was preferable to use a single training window if selecting stocks. The fact that a single training window suffices to provide safe stock selections supports the argument that the most recent scores are the most relevant, as more signals of financial distress can be observed in the year preceding dividend cuts. This suggests future approaches where it might be more valuable to add more weight to recent scores.

Finally, the sliding window setup was also adapted for dividend maintenance streaks, predicting only dividend cuts. The results showed that after running all evaluation years, most firms which end up cutting their dividend in the following year are not selected by the system. In particular, by setting the threshold $p = 0.8$, 65% of the total maintaining firms with a 3-year history of stable dividends would be selected, while more than 75% of the cutting firms would be excluded.

In general, the approach taken of combining binary multiple classification scores to rank stocks showed robust results and the developed system proved to be a reliable tool to select safe dividend stocks.

REFERENCES


