Erasmus University Rotterdam
Erasmus School of Economics
Master Thesis Quantitative Finance

Dynamic Autoregressive Tree Model

Ties Hille Ris Lambers (492791)

Supervisor:	Dr. AM Camehl
Second assessor:	Dr. R Paap
Date final version:	11th March 2024

Abstract

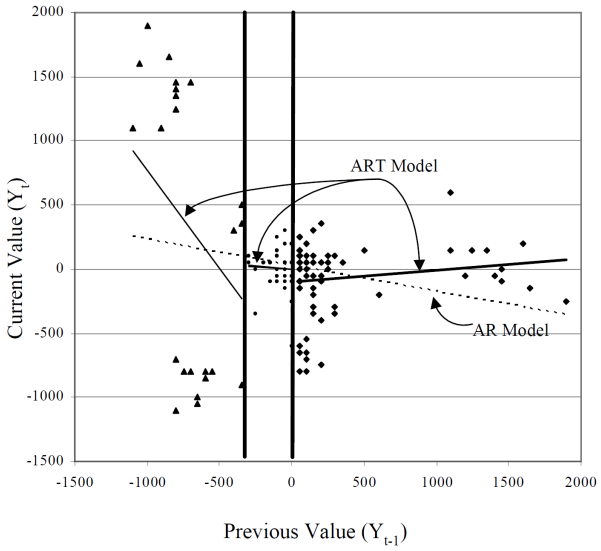
This paper presents innovative advancements to time series forecasting through the extension of the Autoregressive Tree (ART) model to include exogenous variables, thus evolving it into the Autoregressive Exogenous Tree (ARXT) model. The ART model, which combines decision trees with autoregressive models at the leaf nodes, has been pivotal in handling temporal dependencies within time series data. However, traditionally, it has only utilised the target variable with up to 'p' lags to construct its predictive framework. This paper contributes to this research by enabling the ART model to incorporate exogenous variables, both in data splits and as supplementary variables within node models, enhancing the interpretability and variance explanation of complex datasets. Furthermore, this paper delves into hyperparameter re-tuning in the presence of concept drift, where the underlying data distribution has changed. Utilising online changepoint detection and Bayesian optimisation, the efficacy of updating hyperparameters sequentially is assessed, leveraging Bayesian priors to inform the subsequent starting points and reduce computational costs. Empirical tests on forecasting the S&P500 showed ARXT models outperforming AR models in Directional Accuracy by up to 11%. ARXT's integration of exogenous variables significantly improved forecast precision. The tuning framework for ARXT, however, did not always enhance performance, suggesting a need for refinement. The findings underscore the ARXT model's efficacy in leveraging exogenous variables for enhanced forecasting, marking a significant advancement in econometric applications across diverse datasets.

Contents

1 Introduction

The foundations set by Meek, Chickering and Heckerman (2002) have allowed econometricians to use a new approach to forecasting time series. The Autoregressive Tree (ART) model represents the combination of a decision tree with different autoregressive models at the leaf nodes. By incorporating both features, this allows the decision trees to handle temporal dependencies in time series data. The ART(p) gives a clear depiction of the different 'p' order autoregressive models to be used for each split in the decision tree. Through this, an economist would easily be able to see that a time series is likely to revert back strongly, if the previous value is very low or have a low variance, in the case of previous values around zero. This reversal effect is proven through the paper by Da, Liu and Schaumburg (2014). The model is then able to predict future values using the relevant node of the decision tree in which the data at time t falls to.

Figure (1) *ART(1) example*



Note. The figure shows a figure from Meek et al. (2002), with a lag order, 'p', of 1 against a traditional AR(1) model.

The ART models interpretability is clearly shown in Figure 1, with the different data characteristics leading to distinctly different models from the ART(1) tree. However, the model only uses the target variable with up to 'p' lags to make tree splits and construct the autoregressive node models. In the years since the original ART(p) model, decision trees have evolved to split on different features, build to ensembles and use a host of boosting methods in order to efficiently make use of more elaborate datasets. This brings the **first contribution of this paper**: Expand the ART(p) model to incorporate exogenous variables in the leaf splits and as extra variables in node ARX models. This adaptation, named ARXT(p), while keeping the advantages of the ART(p) model, gives clear data splits from which simple models are formed, with extra variance being explained through the exogenous variables.

As with all machine learning models, the decision tree has multiple hyperparameters that need to be tuned. Examples, such as the optimal minimum size and maximum depth can have a big influence on the model outcomes (Probst, Wright & Boulesteix, 2019). Therefore, the **second contribution of this paper**: Test the effectiveness of re-tuning hyperparameters

when concept drift occurs in the data. Concept drift is when the underlying distribution of the data changes, $P(y_s|x_s) \neq P(y_t|x_t)$ where $P(x_s) = P(x_t)$. To test the effect of the change in distribution, an Online Changepoint detection Algorithm by Adams and MacKay (2007) is utilised to find the points in the data where the concept drift takes place. This algorithm seamlessly fits into the model forecasting program through use of Bayesian prior distributions. The hyperparameter training is done using the Bayesian optimisation process described in Snoek, Larochelle and Adams (2012), where Bayesian priors are updated using an optimisation function in order to sequentially tune the different hyperparameters. The advantage of using Bayesian priors is that when re-tuning the model, the previous priors and parameters provide a starting point from which the model can start, reducing the computational cost. Whereas the existing concept drift literature focuses on re-training models such as Gama, Žliobaitė, Bifet, Pechenizkiy and Bouchachia (2014), this paper looks at the improvement in forecasting when both hyper and model parameters are re-trained. This allows models to reconsider hyperparameters, given different data than that it was tuned with.

In order to test the our research questions, the models have been split up to include different elements. This includes either re-tuning or re-training after changepoints, allowing exogenous variables to determine split points and ignoring the changepoints completely. The effectiveness of the models is considered through their RMSE and Directional Accuracy (DA) scores from forecasting the S&P500 over a period of 16 years, from 2006. Eight additional variables have been added as exogenous information, with both Log-Normalisation and Differencing of the returns.

The RMSE values of the different models show that the ART model variants perform well across the dataset, keeping a consistent score across different datasets. The ARXT models without re-tuning or re-training, come second to the ART models in terms of the RMSE. When the DA is considered, the ARXT models, both with and without exogenous variables in the splits, without re-tuning or training achieved a DA of 60% throughout the data splits. This is 11% higher than the benchmark AR models and 22% higher than the original ART model. For the differenced data, the results are similar, with the ARXT train split model attaining the highest RMSE by (work out percentages), and the DA scores again being higher for the ARXT variants that are neither re-tuned nor re-trained. This is a good sign for the model, considering its ability to forecast stock movements, which has applications to much more uses outside of just financial time series.

While the performance of the ARXT model is good, the tuning framework proves not be as effective, with either worse or significantly similar performance against their untuned or trained counterparts. When considering the different split points, it is clear to see that the framework poorly fits the new data, seeing as the last 600 data points at the point of the changepoint are used. However, the framework set up is one that is easily adaptable and applicable to other models. Given minor adjustments and models who have more influential hyperparameters, this could make a large improvement to performance.

This introduction is followed by the relevant literature in Section 2. The methodology used is defined in Section 3. In Section 4 the data used is explained. Section 5 presents the empirical findings of the paper. Finally, Section 6 concludes the findings of the paper.

2 Literature

The original book by Breiman (2017) introduced the concept of Classification and Regression Trees, where regression trees are formed through a decision tree to predict continuous variables based on feature values. This method's relevance to our research lies in its foundational approach to tree-based modeling, which informs our exploration of advanced tree structures and their application in forecasting financial time series. Meek et al. (2002) adapt this approach by integrating autoregressive models at the root nodes, enhancing the model's ability to capture temporal dependencies. This modification is particularly pertinent for our work in financial forecasting, as it demonstrates the potential of combining tree-based models with time series analysis to improve prediction accuracy.

The Bayesian Additive Regression Trees (BART) model, as introduced by (Chipman, George & McCulloch, 2010), marks a significant departure from traditional Autoregressive Trees (ART) through its ensemble approach and Bayesian statistical framework. Unlike ART models that rely on a singular decision tree structure, BART employs a complex ensemble of decision trees, each contributing to the final prediction through a Bayesian additive process. This methodology allows BART to capture intricate, nonlinear relationships between variables with a higher accuracy and robustness against overfitting, leveraging the Markov chain Monte Carlo (MCMC) algorithm to iteratively refine predictions. While the methodology in forming the trees in BART is similar to the ART, there remains a distinct difference due to the less adaptable predictions at the nodes

The additive nature of BART, combined with its capacity to integrate multiple endogenous variables as demonstrated by Huber and Rossini (2022) in the Bayesian Additive Vector Autoregressive Tree (BAVART) model, significantly enhances its applicability to complex data structures like financial time series. This contrasts sharply with the more straightforward, but less flexible, ART models. Although BART's ensemble and Bayesian methodology introduce challenges in interpretability, the model's superior ability to model financial market dynamics, characterized by volatility and nonlinearity,underscores its relevance to our research in advanced statistical techniques for financial forecasting, emphasizing a critical trade-off between complexity and interpretability.

On a similar track to the ART(p) model, many ensemble papers have proposed different adaptations of the Random Forest by Breiman (2001). Tuncel and Baydogan (2018) propose their adaptation through the Multivariate-Autoregressive Random Forest (mv-ARF) model. This non-parametric, VAR-based approach is tailored to handle multivariate time series. Like the ART(p) model, mv-ARF chooses different models based on collective decision-making. The strength of mv-ARF lies in its ability to capture multivariate dependencies, although it can be computationally intensive. Du, Gao, Suganthan and Wang (2022) add to the family of autoregressive forest models by using ten disparate models to create the Bayesian optimisation-based dynamic ensemble (BODE) model. Dynamically adjusting the relative weighting by previous predictions errors and tuning the hyperparameters using Bayesian theory offers adaptability. However, the complexity of handling multiple models can be a drawback.

A key aspect of the ART model is its node Autoregressive (AR) models. The AR model is fundamental to time series analysis, capturing temporal dependencies by expressing a variable's

current value as a linear combination of its past values. The AR model, extensively discussed by (Box, Jenkins, Reinsel & Ljung, 2015), provides a robust framework for modeling and forecasting time series data, particularly where data points are closely time-correlated. Extending this concept, the ARX model incorporates exogenous inputs, creating a more versatile tool for multivariate time series analysis (Lütkepohl, 2005) and the incorporation of external factors into time series prediction.

Another important aspect of this papers tree building process, are the split points used for the tree building. The methods in D. M. Chickering, Meek and Rounthwaite (2001) address the challenge of finding fair and representative split points. They propose a feature space of 8 different fragments in which the data is equally represented, to which they then take the boundaries as the split points. This is critical to the splitting aspect of the tree building.

Thomas Bayes proposed and published his theorem on Bayesian statistics, Bayes (1763), to provide an alternative perception into standard probability theory. Many of the previous models make use of the principle, viewing parameters as random variables and estimating uncertainty through priors. One application of Bayesian mathematics is hyperparameter tuning. Snoek et al. (2012) introduce a Gaussian surrogate model to approximate the objective function, combined with an acquisition function that would use Bayesian priors to improve model parameters until a certain stopping criteria is met.

Following on from the Bayesian framework, this paper makes use of the Maximum a Posteriori Probability (MAP) parameters in calculating the node ARX models parameters. Gauvain and Lee (1994) introduce estimation of MAP parameters in the context of speech recognition, offering insights into the effective application of Bayesian estimation techniques in complex, real-world settings. Their methodology is relevant to any field where Bayesian approaches are employed for parameter estimation, including econometrics and financial modeling. As an extension to the original estimation methods, Heckerman and Geiger (2013) work on learning Bayesian networks proved a significant advancement in probabilistic modeling for networks. They provided a rigorous approach for learning both the structure and parameters of Bayesian networks, through analytical solutions, from which the application can be extended to tree structures.

Another focus of this paper is the behaviour of stock price time series. Many have shown evidence of concept drift in financial time series (Tsymbal, 2004)(Cavalcante, Minku & Oliveira, 2016). In the paper by Gama et al. (2014), the different learning methods used to find and deal with concept drift are investigated, with the revelation of reoccurring (through seasonality) or predictable concept drift being a key finding. The survey indicates that an adaptive learning process helps models significantly, with one example of an adaptive learning method being Online Model Learning. Adams and MacKay (2007) propose a method to incorporate a Bayesian distribution calculated from the last 'changepoint', a changepoint being an abrupt variations in the generative parameters of a data sequence after concept drift. Research has also been done into transfer learning, with tests performed across inductive, transductive and unsupervised transfer learning (Pan & Yang, 2009). While transfer learning offers the potential for leveraging knowledge across domains, it has shown sensitivity to the specific domains involved, sometimes resulting in detrimental outcomes.

3 Methodology

The methodology starts by introducing the formulation for the ARXT model in Section 3.1. This includes the Bayesian learning framework and the parameterisation of the leaf models. This is followed by the tuning framework we set up, explaining the choices in the Bayesian Hyperparameter Optimisation, Changepoint detection algorithm and how all aspects combine into the final model in Section 3.2. Finally, the evaluation methods and models used to compare the ARXT performance are explained in Section 3.3.

3.1 ARXT

3.1.1 Autoregressive Model

This section introduces the ARXT(p) model, a novel adaptation of the ART model introduced by Meek et al. (2002). The model is based on a decision tree with different nodes formed as ARX models with up to p lags. Each model has L leaves, each with their corresponding parameters θ. This makes leaf formulation 1:

$$f(y_t|y_{t-p},\dots,y_{t-1},\mathbf{z}_{t-p},\dots,\mathbf{z}_{t-1},\theta) = \mathcal{N}(m + \sum_{j=1}^{p}(\beta_j y_{t-j} + \sum_{q=1}^{Q}(\gamma_{qj} z_{q,t-j})), \sigma^2), \tag{1}$$

with $\theta = (m, b_1, \dots, b_p, z_1 1, \dots, z_Q p, \sigma^2)$ the model parameters and $\mathcal{N}(\mu, \sigma^2)$ the normal distribution, with mean μ and variance σ^2. The variable z_{qp}, $q = 1, \dots, Q$, has been introduced as the, with γ_{qj}, the parameters corresponding to the set of exogenous variables. For notational convenience, a vector of all z at time t is written as $\mathbf{z}_t$ in f. The model still remains linear and normally distributed with mean μ and variance σ^2. This leads to the model formulation 2:

$$\begin{aligned} f(y_t|y_{t-p},\dots,y_{t-1},\mathbf{z}_{t-p},\dots,\mathbf{z}_{t-1},\theta) &= \prod_{i=1}^{L} f_i(y_t|y_{t-p},\dots,y_{t-1},\mathbf{z}_{t-p},\dots,\mathbf{z}_{t-1},\theta_i)^{\phi_i} \\ &= \prod_{i=1}^{L} \mathcal{N}(m_i + \sum_{j=1}^{p}(\beta_{ij} y_{t-j} + \sum_{q=1}^{Q}(\gamma_{iqj} z_{q,t-j})), \sigma_i^2)^{\phi_i}, \end{aligned} \tag{2}$$

with L the number of leaves, and ϕ a boolean in the case of the data falling to that specific leaf. The data going to each leaf is determined through the tree structure.

This delineation of the ARX model allows the model to be specified to different sets of data dependent on the characteristics. Da et al. (2014) show that there is a clear reversal effect after big jumps. This would mean that a large drop would be more likely to see a positive return in the next period and that could be accounted for.

3.1.2 Leaf structure

Meek et al. (2002) makes use of the Bayesian score approach to learning for the ART(p) model. Considering S alternative model structures $s_1, \dots, s_S$, each with θ_s as their corresponding model parameters. The uncertainty of each of the different structures, $p(s)$ and $p(\theta_s|s)$, is combined with the data, d, to find the posterior distributions $p(s|d)$ and $p(\theta_s|d, s)$. The learning pro-

cess selects the model with the highest posterior probability, calculated through the Bayesian conditional probability formula:

$$p(s|d) = \frac{p(s)p(d|s)}{p(d)}. \tag{3}$$

Given that $p(d)$ is assumed constant at the time of learning, $p(s)p(d|s)$ can be taken as the Bayesian score. This approach can now be applied to the likelihood of the data as defined in equation 2. Meek et al. (2002) introduce a windowing approach to the data to reduce the computational cost to one of learning a linear regression model instead of a multiple variable regression. The windowing is applied to the data before model learning. For each sequences $y = (y_1, \ldots, y_T)$, the data is transformed as follows:

$$x^i = (x_1^i, \ldots, x_{p+1}^i), 1 < i < T - p, \text{ where } x_j^i = y_{i+j-1}, \tag{4}$$

and the same holds for the exogenous variables:

$$\mathbf{v}^i = (\mathbf{v}_1^i, \ldots, \mathbf{v}_{p+1}^i), 1 < i < T - p, \text{ where } \mathbf{v}_j^i = \mathbf{z}_{i+j-1}. \tag{5}$$

The subsequent transformed data set is now called the *length p transformation*. This allows equation 26 to be rewritten to:

$$p(y_{p+1}, \ldots, y_T|y_1, \ldots, y_p, \mathbf{z}_{t-p}, \ldots, \mathbf{z}_{t-1}, \theta, s) = \prod_{t=p+1}^{T} f_i(x_t|x_{t-p}, \ldots, x_{t-1}, \mathbf{v}_{t-p}, \ldots, \mathbf{v}_{t-1}, \theta, s), \tag{6}$$

where f_i is the normal distribution corresponding to the linear regression l_i for the data falling to split s, as in 2. This likelihood is the same as one for an ordinary regression model with target x_{p+1} and regressors $x_1, \ldots, x_p, \mathbf{v}_1, \ldots, \mathbf{v}_p$.

In order to make the model selections, the Bayesian score has to be calculated for each split. This score is constructed by the product of all the leaf scores. These can also be seen as the product of the prior probability of the leaf and the marginal likelihood that data falls to that leaf:

$$\text{Score(s)} = \prod_{i=1}^{L} \text{LeafScore}(l_i), \tag{7}$$

where:

$$\text{LeafScore}(l_i) = \kappa^{p+2} \int \prod_{x^t \text{ at } l_i} f_i(x_{p+1}^t|x_1^t, \ldots, x_p^t, \mathbf{v}_1^t, \ldots, \mathbf{v}_p^t, \theta_i, s)p(\theta_i, s)d\theta_i, \tag{8}$$

and x^t at l_i refers to the set of data at time period t corresponding to the specific leaf. The priori likelihood is given by $p(s) = \kappa^{|\theta|}$ with $0 < \kappa \leq 1$. Meek et al. (2002) set κ to 0.1, this value is canceled out in a later calculation. The last element, the parameter prior, is taken as the traditional conjugate prior for a linear regression. This is also referred to as the normal-gamma prior for θ_i.

The exact formulation of the leaf score is based on Maximum a Posteriori Probability (MAP) parameters. So given the transformation to the windowing approach, the exogenous variables

and the target variable are combined into the variable $a_i = [yi, z_{i1}, \ldots, z_{iQ}]$. The new variable can be used to set up the following relationship:

$$p(x_{p+1}^t | x_1^t, \ldots, x_p^t, \mathbf{v}_1^t, \ldots, \mathbf{v}_p^t, \theta, s) = \mathcal{N}(m + \sum_{j=1}^{p} b_j a_{t-j}, \sigma^2), \;\; t = 1, \ldots, N, \tag{9}$$

where:

$$m = \mu_{p+1} - \sum_{i=1}^{p} b_i \mu_i, \;\; b_j = \sum_{i=1}^{p} (W^{-1})_{p+1,i} \, (((W^{-1})^{p \times p})^{-1})_{i,j}, \;\; \sigma^2 = \frac{1}{W_{p+1,p+1}}. \tag{10}$$

The formulas that maximise $p(d|\theta, s)$, solved analytically by Heckerman and Geiger (2013) in the revised version of their 1995 paper, can be given as follows:

$$\widetilde{\mu} = \frac{\alpha_\mu \mu_0 + N \mu_N}{\alpha_\mu + N}, \;\; \widetilde{W}^{-1} = \frac{1}{\alpha_W + N - (p+1)} W_N, \tag{11}$$

where:

$$W_N = W_0 + S_N + \frac{\alpha_\mu N}{\alpha_\mu + N} (\mu_0 - \bar{\mu_N})(\mu_0 - \bar{\mu_N})', \tag{12}$$

and

$$\bar{\mu_N} = \frac{1}{N} \sum_{t=1}^{N} a_t, \;\; S_N = \sum_{t=1}^{N} (a_t - \bar{\mu_N})(a_t - \bar{\mu_N})', \tag{13}$$

with $\tilde{\mu}$ and $\bar{\mu}$ being the analytical and sample μ, respectively. Furthermore, $W_0 = I, \alpha_w = p + 2$ and $\alpha_\mu = p$.

The advantage of having these analytical solutions for the MAP parameters is the huge decrease in computational time. Traditionally, MAP parameters have to be found through simulation, which in this papers case, would mean simulations would have to be run for all possible different split points and parameter estimates. In Gauvain and Lee (1994), EMM estimation is also considered as an effective method to estimate MAP parameters, but at a much heavier computational cost than the direct calculation.

With these estimations and the assumptions from Heckerman and Geiger (2013), the marginal likelihood $p(d|s)$ can be formulated as:

$$p(d|s) = \pi^{-\frac{(p+1)N}{2}} (\frac{\alpha_\mu}{\alpha_\mu + N})^2 \frac{c(p+1, \alpha_W + N)}{c(p+1, \alpha_W)} |W_0|^{\frac{\alpha_W}{2}} |W_N|^{\frac{\alpha_W + N}{2}}, \tag{14}$$

where

$$c(l, \alpha) = \prod_{i=1}^{l} \Gamma(\frac{\alpha + 1 - i}{2}). \tag{15}$$

This leads to the final marginal likelihood being given as:

$$\int \prod_{t=1}^{N} p(x_{p+1}^t | x_1^t, \ldots, x_p^t, \mathbf{v}_1^t, \ldots, \mathbf{v}_p^t, \theta, s) p(\theta_i, s) d\theta_i = \frac{p(d|s)}{p(d^-|s)}, \tag{16}$$

Here, d^- is the windowed data set to $X_1, \ldots, X_p$, leaving out X_{p+1} meaning μ_0, W_0 and α_w get

replaced by μ_0^-, W_0^- and α_w^-. The parameters are fixed at $\mu_0 = 0, W_0 = I$ and $\alpha_w = p + 2$ for both d and d^-. These parameters are hence the assumptions needed for MAP parameters, this could prove to not hold in some cases leading to slight misspecification in the parameters.

Due to the value of α in the c function being heavily dependant on N, the values can explode when larger datasets being allocated to a certain split. This causes the value for the marginal likelihood to go to infinity, therefore, to handle these cases of large N, an approximation for $p(d|s)$ is derived:

$$p(d|s) \approx p(d|\hat{\theta}_s, s) - \frac{|\theta|}{2}\log(N) = \left(\frac{\alpha_u}{\alpha_u + N}\right)^{\frac{p+1}{2}} |W_0|^{\frac{\alpha_W}{2}} |W_N|^{-\frac{\alpha_W+N}{2}} - \frac{\alpha_W}{2}\log(N). \quad (17)$$

The final aspect to consider is the different split points when applying the above formulations to different splits. The same approach as in Meek et al. (2002) is used to define split points using the methodology by D. Chickering, Meek and Rounthwaite (2001). They propose a space of eight different equiprobable continuous regions, estimated from a normal distribution on the data split. The boundaries of the different spaces are then given as the split points. This paper uses eight different regions, leading to seven different boundaries and split points for each variable. This is applied to all exogenous and target variables.

3.1.3 Model Training

This section combines the previous section's algorithms to train the ARXT model. In the training process, all of the available data is used in order to build the tree structure. As will be further explained in Section 3.2.3, a lot of the training framework revolves around the hyperparameters.

The ARXT uses four distinct hyperparameters to optimise performance. The first hyperparameter, denoted as p, specifies the maximum number of lags in our autoregressive (ARX) models, effectively determining the extent of historical data points the model considers. The second hyperparameter, *max_depth*, is defined as the maximum number of consecutive child nodes that can branch off from the initial node, shaping the tree's complexity and depth.

The last two hyperparameters, *max_weight* and *min_size*, share a conceptual similarity. The hyperparameter *min_size*, sets the minimum data size required for a split within the model, ensuring a sufficient amount of data in each segment. Meanwhile, *max_weight* establishes the upper limit for the weight of the training data a child node can carry, balancing the distribution of data across the nodes.

The data splits with, which the ARXT model is made, are based on the standard 70 / 30 split. This allows the model to be trained for 70% of the data and then evaluated using the final 30% of the data. This is critical to the model evaluation in Section 3.2.3.

Algorithm 1 ARXT(p) Model Training

1. **Node Creation:** Start with the root node that includes the entire dataset.
2. **Feature Selection:** At each node, seven different split points are offered for all target and exogenous variables and their lags. The leaf scores are calculated as in (16). The split with the highest split score is chosen as the next splitting point.
3. **Recursive Splitting:**
   - For each split, repeat the above point to create new splitting points until a stopping criteria has been met.
   - Create child nodes and assign the respective subsets of data to these children.
   - Recursively apply the process to each child node.
4. **Stopping Criteria:** Determine when to stop splitting. This can be based on:
   - Maximum tree depth.
   - Minimum number of samples in a node.
   - Minimum weight of a sample.
5. **Terminal Node:** Once the stopping criteria are met, declare the node as a terminal node (leaf node). Determine the model parameters for the terminal node using the MAP parameters and go back to the branches for which the stopping criteria has not been met.

The computational costs of training the model come to $\mathcal{O}((p \times (Q+1))^2 \times p \times Q \times N)$. This is assuming the MAP parameters cost $\mathcal{O}((p \times (Q+1))^2 \times d)$ to calculate, and there are p lags, with Q exogenous variables. This is only not linear in p and Q, which realistically will stay relatively small, showing the advantages of training the model using MAP parameters and the windowing approach.

3.2 Bayesian Tuning

3.2.1 Bayesian Hyperparameter Tuning

The first choice that has to be made in the Bayesian optimisation process is the prior distribution on the optimisation functions. The most convenient and powerful distribution is a Gaussian Process (GP) (Snoek et al., 2012). The GP is defined by the property that any finite set of N points $\{z_n \in X\}_{n=1}^{N}$ induce a multivariate Gaussian distribution on $\mathbb{R}^N$. The GP offers the best flexibility and capacity for modeling complex functions. This leads to the function $f(\mathbf{x})$, drawn from a GP prior, with the assumption that the observations are of the form $\{\mathbf{x}_n, y_n\}_{n=1}^{\mathcal{N}}$ where $y_n \sim \mathcal{N}(f(\mathbf{x}_n), \nu)$, ν being the variance of the noise.

Secondly, the acquisition function has to be chosen. Again using the evidence from Snoek et

al. (2012) the Expected Improvement function gives a good closed form option to use:

$$a_{EI}(\mathbf{x}, \theta) = \sigma(\mathbf{x}, \theta)(\gamma(\mathbf{x})\Phi(\gamma(\mathbf{x})) + \mathcal{N}(\gamma(\mathbf{x}); 0, 1)), \tag{18}$$

where:

$$\gamma(\mathbf{x}) = \frac{f(\mathbf{x}_{best}) - \mu(\mathbf{x}, \theta)}{\sigma(\mathbf{x}, \theta)},$$

which subsequently has $\mu(\mathbf{x}, \theta)$ and $\sigma^2(\mathbf{x}, \theta)$, the predictive mean and variance functions respectively, with $f(\mathbf{x_{best}}) = \arg\min_{x_n} f(\mathbf{x_n})$. The advantage of the Expected Improvement function is its ability to both focus on areas of high uncertainty and feature spaces, with the highest predicted improvement through closed form solutions. When $\mu(\mathbf{x}, \theta)$ is larger than $f(\mathbf{x}_{best})$, this implies an improvement in the new feature space. When the uncertainty, $\sigma(\mathbf{x}, \theta)$, is high, this also leads to the best predicted improvement.

The final aspect of the Bayesian Optimisation, is the optimisation function. This paper trains the model, as in Subsection 3.1.3, with the following hyperparameters: *p*, *max_depth*, *min_size*, *max_weight* on the training set. In order to assess the models, the following two metrics are applied to the forecasting results in the testing set:

$$\text{Directional Accuracy} = \frac{1}{N-1} \sum_{i=1}^{N-1} \mathbb{I}\left(\text{sign}(y_{i+1} - y_i) = \text{sign}(\hat{y}_{i+1} - \hat{y}_i)\right), \tag{19}$$

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2}, \tag{20}$$

$$\text{Target} = \text{Directional Accuracy} * 2 - \text{RMSE} * 0.5. \tag{21}$$

This combination of the two metrics is novel, with a focus more towards RMSE than directional accuracy, scaling the two metrics to get a fair distribution between the two aspects. The standard of RMSE is most commonly used, but given the application to financial point forecasting, directional accuracy is an important factor to consider. The expected improvement algorithm will be evaluated using $f(\mathbf{x})$ as Target in 21.

Algorithm 2 Bayesian Hyperparameter Tuning

1. Set a starting point to determine the initial $f(\mathbf{x_{best}})$.
2. Calculate the expected improvement $a_{EI}(\mathbf{x}, \theta)$ across the feature space.
3. Calculate $f(\mathbf{x_n})$ chosen by the highest EI and set that to the value of $f(\mathbf{x_{best}})$ if there is an improvement.
4. Repeat steps 1 through 3 until the desired number of iterations has been reached.

The implementation is done through the package provided in Bayesian Optimization Contributors (2024).

3.2.2 Bayesian Changepoint Detection

The previous subsection describes how this paper uses Bayesian optimisation in order to find the best possible hyperparameters given a set of training data. Only the focus of this paper does not stop there. Namely, the second research question is if re-tuning hyperparameters after concept drift would significantly improve model performance.

There are many different ways to define concept drift. This paper decides to test for changepoints using the methodology in Adams and MacKay (2007). They propose the following algorithm to find the changepoints in the target variable. This is done with an online approach, meaning that the procedure is continually applied to new data. The online approach is essential to this papers model and usability, while the algorithm is flexible in allowing users to apply their own priors to the data, even if there is no information to be passed to the model. With this ability to set priors, there is room for error as the model continually updates its distribution to fit to the data.

For this algorithm, the same Gaussian distribution as in 3.2.1 is used as the prior distribution. This choice is motivated by the robustness of the model and conjugacy properties. This means the only priors that have to be defined are that of the mean and variance.

The Hazard function $H(\tau)$ plays a crucial role in the Bayesian Online Changepoint Detection algorithm. It quantifies the probability of a changepoint occurring at time τ, given that no changepoint has occurred until then. For a given run length τ, the Hazard function $H(\tau)$ is defined as:

$$H(\tau) = \frac{\mathrm{P}_{\mathrm{gap}}(g = \tau)}{\sum_{t=\tau}^{\infty} \mathrm{P}_{\mathrm{gap}}(g = t)}, \tag{22}$$

where $\mathrm{P}_{\mathrm{gap}}(g)$ is the probability distribution of the gaps between changepoints. A discrete exponential (geometric) distribution with timescale λ is used, with which the memoryless property can be exploited to simplify $H(\tau)$ to a constant $1/\lambda$.

In order to start the process, the initial run length, r_0, and hyperparameters, $\nu_1^{(0.00)}$ and $\boldsymbol{\chi}_1^{(0.00)}$, have to be defined for the exponential distribution. Considering there is a chance that either a changepoint happened right before the algorithm starts or that we have prior information that could influence the run length. The starting conditions are defined by the normalised survival function:

$$P(r_0 = \tau) = \frac{1}{Z} S(\tau), \tag{23}$$

where Z is a normalising constant and:

$$S(\tau) = \sum_{t=\tau+1}^{\infty} P_{gap}(g = t). \tag{24}$$

This leads to the online changepoint detection methodology, fully detailed in Appendix 3. Where a constant stream of data, in our case next day returns, is passed. The model is altered to return a boolean to the algorithm, if at any point a changepoint is detected and re-tuning needs to occur.

3.2.3 Tuning Framework

In order to effectively apply the methodology in Section 3.2 and Algorithm 3, a combined approach is taken. The initial tuning is done over the training set, the first 1000 observations in the time series. This gives a reasonable window in which the tuning can be done for the first 700 observations and the evaluation can be done out of sample on the remaining 300 observations. The feature space is defined as follows: $p \in [1, 20]$, $max\_depth \in [10, 150]$, $min\_size \in [1, 50]$, $max\_weight \in [0.01, 0.15]$. The training does an initial search over 20 parameter spots and then uses the expected improvement calculation to find next optima. Once a maxima has been found this is stored as the optimum, only if an improved maxima is found will the initial optimum be overwritten. This is done until convergence, defined as a marginal improvement of less than 0.001, is found.

Following the initial tuning, further tuning takes place after all subsequent changepoints. Based on the initial tuning, the optimal parameters are used as a starting points for the further training. This is defined by setting the optimisation bounds as : $p \in [0.7 \times p^*, 1.3 \times p^*]$, $max\_depth \in [0.7 \times max\_depth^*, 1.3 \times max\_depth^*]$, $min\_size \in [0.7 \times min\_size^*, 1.3 \times min\_size^*]$, $max\_weight \in [0.7 \times max\_weight^*, 1.3 \times max\_weight^*]$. This narrows the search of the hyperparameters to a smaller range based on previous optimisation, thereby reducing the need for a full tuning. Because of this, each re-tuning only uses an initial search of 10 iterations, followed by however many iterations are needed to converge.

Once the re-tuning has finished, the model is subsequently fit to the new data, based on the previous 600 observations.

3.3 Evaluation

3.3.1 Evaluation Models

In order to evaluate the effectiveness of the ARXT model, there are four different models considered. The first model is a step between the original ART model and the ARXT model. This is known as the ARXT target model, where the only difference to the ARXT model defined above being the splitting points. Whereas the ARXT model determines data splits based on the target value and all the exogenous variables and their respective lags, the splitting on the exogenous variables is left out for the ARXT target model. This means that the model training algorithm (1) changes in step 3 to only allow splits on the variable we are forecasting. For clarity, the model including exogenous splits is referred to as ARXT exog, and the new comparative model is named ARXT target.

The second model used to evaluate the ARXT is the original ART model introduced by Meek et al. (2002), provided by Nekoie (2024). The main difference is the conversion of the ARX node models to the AR models, as the ART model is a piece-wise AR model, and only the target variable is used in the model. This leads to the node equations to be defined as:

$$f(y_t|y_{t-p}, \ldots, y_{t-1}, \theta) = \mathcal{N}(m + \sum_{j=1}^{p} b_j y_{t-j}, \sigma^2), \tag{25}$$

with $\theta = (m, b_1, \ldots, b_p, \sigma^2)$ the model parameters and $\mathcal{N}(\mu, \sigma^2)$ the normal distribution with mean μ and variance σ^2.

Subsequently the model definition for the ART(p) can be written as:

$$f(y_t|y_{t-p}, \ldots, y_{t-1}, \theta) = \prod_{i=1}^{L} f_i(y_t|y_{t-p}, \ldots, y_{t-1}, \theta_i)^{\phi_i} = \prod_{i=1}^{L} \mathcal{N}(m_i + \sum_{j=1}^{p} b_{ij} y_{t-j}, \sigma_i^2)^{\phi_i}, \quad (26)$$

with L the number of leaves, each with their corresponding $\theta_i = (m_i, b_{i1}, \ldots, b_{ip}, \sigma^2)$. ϕ is again a boolean to determine if the tree is used for each leaf.

As this paper introduces ARX models into the nodes as opposed to the AR models, we use both as benchmarks to compare the performance to. Both models use least squares to determine the model parameters. The same amount of lags, p, are used for both models.

The AR model with p lags can be written as:

$$y_t = \alpha + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t, \quad (27)$$

where ϕ_i for $i = 1, \ldots, p$ are the coefficients that measure the impact of the i-th lagged value on the current value and ε_t is the error term.

The ARX(p) model (Lütkepohl, 2005) extends the AR model by incorporating external, or exogenous, variables. Therefore, the ARX(p) model with k exogenous variables is represented as:

$$y_t = \alpha + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_k x_{k,t} + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t, \quad (28)$$

where $x_{1,t}, \ldots, x_{k,t}$ represent the exogenous variables, with their corresponding lags, and $\beta_1, \ldots, \beta_k$ are coefficients for each exogenous variable.

The above models are used to evaluate the first contribution of this paper, the novel ARXT model. In order to test the second contribution, the improvement from re-tuning the model after concept drift is found, further alterations are made. These changes are to the tuning framework in Section 3.2.3. Instead of re-tuning the model at every changepoint, one alternative is to only retrain the model. Secondly, the changepoints are not considered and the initial model is used to forecast the time series from the beginning. These alterations are applied to the ARXT exog, ARXT target and ART model. An overview of all models models, with their different features and subsequent abbreviations is given in Table 1.

Table (1) Overview of Different Models Evaluated

Model	ARXT	ART	ARX	AR	Exog Split	Initial Tuning	Re-training	Re-tuning
ARXT x tu	x				x	x	x	x
ARXT x tr	x				x	x	x	
ARXT x	x				x	x		
ARXT t tu	x					x	x	x
ARXT t tr	x					x	x	
ARXT t	x					x		
ART tu		x				x	x	x
ART tr		x				x	x	
ART		x				x		
ARX tr			x				x	
ARX			x					
AR				x				

3.3.2 Evaluation Methods

In order to evaluate the performance of the models, two main metrics are used. The first is the directional accuracy, as in equation 19. This metric calculates the number of times the model correctly predicts the direction of the next movement. This is important when considering one step ahead forecasts as the direction can influence investing decisions made using the model. The second metric RMSE, as defined in 20, is an average over all squared errors. This efficiently shows the absolute forecasting power of each model variation. Although the target function in 3.2.1 is a combination of the two, we present the results separately.

The Diebold Mariano test, as given by Diebold and Mariano (2002), is used to measure the significance of the improvement over the benchmark. The Diebold-Mariano (DM) test statistic is computed as follows:

$$DM = \frac{\overline{d}}{\sqrt{\frac{\hat{\sigma}^2}{T}}}, \tag{29}$$

where $\overline{d}$ is the average of the pairwise forecast error differentials between two competing models over T forecast horizons, $\hat{\sigma}^2$ is an estimate of the long-run variance of the forecast error differentials and T is the number of forecasts.

The test is used to determine if the difference in forecast accuracy between two models is statistically significant. A large absolute value of the DM statistic indicates that one model has significantly higher predictive accuracy than the other. This is again used to compare the effectiveness of models to a benchmark and to assess the improvement of the re-tuning after changepoints.

All the models, evaluation methods and code can be found on the repository: `https://github.com/TiesHRL/DyART`.

4 Data

Three main indices are to be used: S&P 500, NASDAQ and the DJI. The forecasting will be on the S&P 500, with the other two being used as exogenous variables. Additionally, three further financial indicators: US inflation, unemployment and FOREX rates are used to give an impression of the global economy. The final indicator is the volatility measure, the VIX.

For inflation, we use a proxy in order to get daily data, like the stock return data. For this, a measure of expected inflation derived from 10-Year Treasury Constant Maturity Securities (BC_10YEAR) and 10-Year Treasury Inflation-Indexed Constant Maturity Securities (TC_10YEAR) is used. The latest value implies what market participants expect inflation to be in the next 10 years, on average. This is obtained from FRED (2023).

As unemployment figures are also only reported on a monthly basis, another proxy is used for this. The jobless claims, directly correlated to unemployment figures, poses as the effective proxy. This data is weekly, but can reasonably be extrapolated to daily data. A rolling 4 week average of the jobless claims is taken, with interpolation used to transform the weekly data into daily data. This is also taken from FRED (2023).

All data is subsequently transformed to returns through the formula: $\Delta\%x_t = \frac{x_t - x_{t-1}}{x_{t-1}}$. This allows the model assumptions in Equation 14 to hold.

Table (2) *Descriptive statistics of raw returns*

	S&P500	DJI	NASDAQ	EURUSD	JPYUSD	GBPUSD	VIX	INFLATION	ICSA
Mean	0.04	0.03	0.05	0.04	0.00	0.03	0.26	0.14	0.02
Median	0.04	0.03	0.06	0.00	-0.01	0.01	-0.34	0.00	-0.06
Maximum	11.58	11.37	11.81	25.01	18.35	37.47	115.60	380.00	283.73
Minimum	-11.98	-12.93	-12.32	-17.95	-15.03	-18.44	-29.57	-90.00	-8.26
Std. Dev.	1.19	1.13	1.35	2.73	0.73	2.80	7.55	6.80	4.16
Skewness	-0.25	-0.15	-0.21	0.74	1.86	2.33	2.12	37.45	61.45
Kurtosis	16.03	18.89	10.60	14.64	126.47	32.83	20.25	1979.88	4162.01
Jarque-Bera	36954	54934	12610	29933	3317109	198161	68575	8.51E+08	3.76+09
Probability	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

As seen from the results of the Jarque-Bera tests and the means, the data is not normally distributed. This is a problem for the AR(X(T)) model due to some of the underlying assumptions. For this reason, two different approaches are considered. The first approach is Log-Normalising the data. This can be done through the simple formula: $x_{LN} = ln(1 + x_i)$

Table (3) *Descriptive statistics of Log-Normalised data*

	S&P500	DJI	NASDAQ	EURUSD	JPYUSD	GBPUSD	VIX	INFLATION	ICSA
Mean	0.03	0.03	0.04	0.00	0.00	-0.01	-0.01	0.01	-0.01
Median	0.04	0.03	0.06	0.00	-0.01	0.01	-0.34	0.00	-0.06
Maximum	10.96	10.76	11.16	22.32	16.85	31.82	76.82	156.86	134.48
Minimum	-12.77	-13.84	-13.15	-19.78	-16.29	-20.38	-35.06	-230.26	-8.62
Std. Dev.	1.20	1.14	1.35	2.71	0.73	2.74	7.17	5.17	2.24
Skewness	-0.51	-0.46	-0.41	0.23	0.57	1.35	1.07	-10.02	43.26
Kurtosis	16.30	19.23	10.86	13.30	118.11	24.87	9.81	969.80	2501.23
Jarque-Bera	38690	57409	13558	23098	2880795	105527	11068	2.03E+08	1.36E+09
Probability	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

The values in this table show that the data has a lot more normal properties, but is far from normal. However, for our model assumptions this is sufficient as all they require is a zero mean and standard deviation of one, which the relevant predictors give. This approach however, does not consider the stationarity of the data. For this, a unit root test can be used on our target time series, S&P 500, to test for non stationarity. The Augmented Dickey-Fuller test by Fuller (1979) is used, resulting in a test statistic of -82.24. Considering this is tested against the null hypothesis of non-stationarity and a critical value of -2.86 at the 5% level, the data is clearly stationary.

As an alternative approach, differencing is also used for the data: $\Delta y_t = y_t - y_{t-1}$. This simple transformation leads to the following data:

Table (4) *Descriptive statistics of Differenced data*

	S&P500	DJI	NASDAQ	EURUSD	JPYUSD	GBPUSD	VIX	INFLATION	ICSA
Mean	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Median	-0.06	-0.05	-0.07	0.02	0.02	-0.01	0.40	0.00	0.00
Maximum	18.80	19.35	18.78	32.57	33.38	49.38	87.09	355.00	280.15
Minimum	-21.27	-22.29	-21.67	-24.94	-18.93	-38.06	-135.27	-346.67	-247.21
Std. Dev.	1.79	1.70	2.00	4.08	1.13	4.18	11.09	9.00	5.27
Skewness	0.49	0.45	0.30	0.10	4.28	0.65	-0.64	1.28	8.98
Kurtosis	19.84	23.54	13.96	10.26	190.16	22.56	13.26	974.95	2459.23
Jarque-Bera	61842	91891	26186	11467	7628821	83518	23216	2.05E+08	1.31E+09
Probability	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

By using a different approach, we are able to test the robustness of the data to different characteristics, while keeping to the model assumptions.

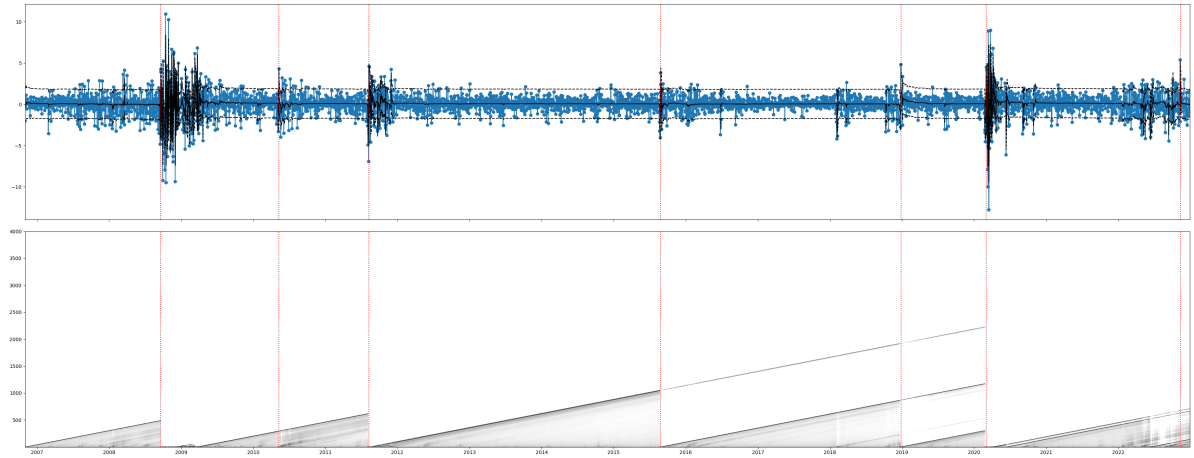
Given a time period of 20 years, January 2003 to December 2022, this leads to 5218 observations for the raw and Log-Normalised data and 5217 points for the differenced data. All of the necessary data can be collected using the appropriate tickers and the use of the Yahoo Finance API (`finance.yahoo.com`) or through the FRED (2023).

5 Results

From the changepoint detection algorithm explained in Section 3.2.2, Figure 2 was generated. The grey shading shows the likelihood of the underlying distribution of the data being the same as $t-1$. This can also be interpreted as a darker shading meaning a higher chance of a run length y at time t. As shown in Figure 2, this leads to triangular shaped shadings, showing the different run lengths,and the probability of them occurring. The top of these triangles are darker indicating the most probable run length at that time period. This is displayed under a depiction of the log normalised returns throughout the entire forecasting period. For the points where the probability of a changepoint occurring being maximal, over 0.9 in our case, red dotted lines are shown. With the exception of two changepoints, these lines are all found at the end of the shaded triangles. These two exceptions are found where the probability of the run continuing is lower than a changepoint occurring, causing the changepoints to be found in the middle of the shaded triangles. For the second changepoint, there is a clear darker shading at the bottom of the larger triangle, which is why the algorithm determines that a changepoint at that point is more probable than continuing the run.

5.1 Changepoint Detection

Figure (2) Results From Changepoint Detection Algorithm



Note. This figure is split into two parts. The top figure is the stock returns across the time series. The bottom figure shows the changepoint detection. The y axis is the run length plotted against time. The darker shading shows a higher probability of a run length of that magnitude at that time. The red dotted lines are plotted at the points where changepoints are found and drawn across to the top figure.

Looking at both the top and bottom plot, it is easy to see why the changepoints for the different sections are chosen. High periods of volatility in the data change the behaviour of the returns, which in turn should be considered when using our market models. Each of the points accurately reflect external influences to the S&P 500, as detailed below:

1. 2008-09-19: 2008 global financial crisis. Major financial institutions were failing, and there was extreme volatility in global markets. The U.S. government proposed a significant bailout plan for banks around this time, which likely caused shifts in market dynamics

and the reason for the changepoint to be chosen later in the extensive global financial crisis.

2. 2010-05-11: European sovereign debt crisis. Greece was particularly affected, and there were concerns about the stability of the Eurozone. These international events likely influenced the U.S. stock market, as investors reacted to the uncertainty in Europe.

3. 2011-08-10: Standard & Poor's downgraded the United States' credit rating from AAA to AA+ for the first time, citing concerns about the rising debt burden and the political environment. This was a significant event, leading to increased market volatility.

4. 2015-08-27: A period of significant volatility in the global markets, partly due to concerns about the Chinese economy. China's stock market experienced a major crash in mid-2015, leading to global repercussions, as shown in the increased volatility in an otherwise relatively stable period of 9 years.

5. 2018-12-27: A sharp downturn in the stock market, attributed to various factors including trade tensions between the U.S. and China, concerns about global economic slowdown, and the Federal Reserve's interest rate hikes.

6. 2020-03-03: The onset of the COVID-19 pandemic. In early 2020, as the severity of the pandemic became clear, global markets experienced significant declines due to the uncertainty and expected economic impact.

7. 2022-11-11: Concerns over inflation, rising interest rates, geopolitical tensions, and the ongoing impact of the COVID-19 pandemic. While markets recovered in the years after the initial pandemic, this was seen as a period of correction.

Table (5) Data Characteristics after splitting data

	Split 1	Split 2	Split 3	Split 4	Split 5	Split 6	Split 7	Split 8
count	491	427	326	1056	870	308	703	36
mean	-0.03	-0.01	0.00	0.05	0.03	0.07	0.04	-0.08
std	1.16	2.24	1.13	0.92	0.84	0.88	1.62	1.13
min	-4.83	-9.47	-6.90	-4.56	-4.18	-4.52	-12.77	-2.52
max	4.24	10.96	4.63	4.53	4.84	4.50	8.97	3.05

Note. The table shows key statistics from all different splits in the Log-Normalised data chosen by the changepoints.

Table 5 shows the different characteristics of the data after defining the different splits. The returns demonstrate varying data counts across splits, with Split 4 containing the largest dataset (1056 data points) and Split 8 the smallest (36 data points), coming to an average of 527 points per split. For the Log-Normalised returns, the mean values are relatively close to zero, with some splits like Split 6 (0.07) and Split 8 (-0.08) showing slight deviations, indicative of the subtle but distinct impacts of the respective market events on average returns. The standard deviation is the most distinctive difference across the splits. Notably, Split 2 has a higher standard deviation (2.24), potentially mirroring the market turmoil during the European sovereign debt crisis. The range of returns, captured by the minimum and maximum values, also varies significantly, with

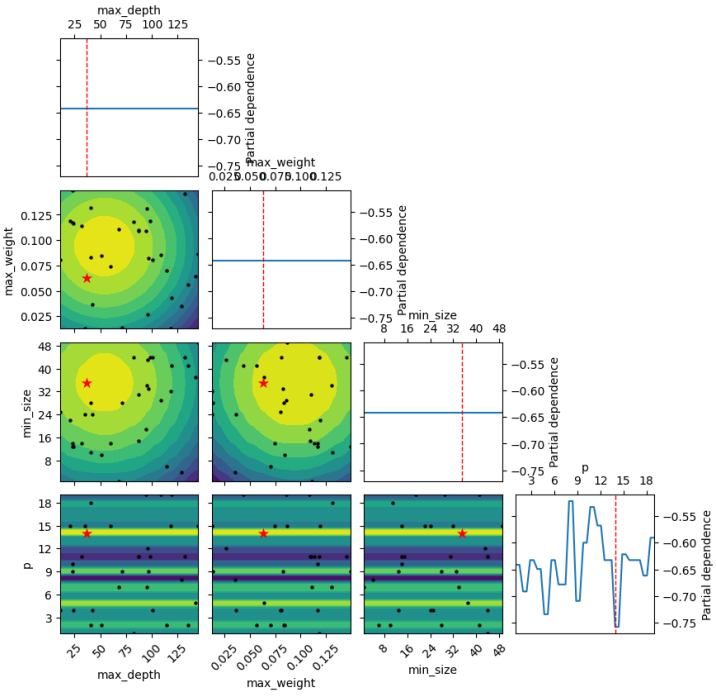
extreme values like in Split 7 (-12.77, 8.97) likely reflecting the market's response to events such as the late 2018 downturn or the beginning of the corona pandemic.

These results show the ability of the changepoint algorithm to spot changes in the underlying distribution of the data. These all clearly align with real life events, leading to seven points where the re-tuning and re-training framework can be applied to. Even if the effect of the re-tuning is not evident these results can be of use to explain different model performances across different splits, with clear explanations to the behaviour of the market through those periods.

5.2 Hyperparameter Tuning

Figure 3 presents a comprehensive visualisation of the optimisation landscape for our hyperparameters. The line plots depict the relationship between the model's performance metric and the respective hyperparameters. The series of contour plots, under the diagonal, offer insight into the exploration between pairs of hyperparameters. These heatmaps are colour-coded to represent different performance levels, with lighter tones indicating superior model performance. The red crosses mark the final outcome of specific hyperparameter combinations tested during the optimisation process.

Figure (3) Bayesian Dependence Plot ARXT Log-Normalised



Note. The figure shows the cross dependence of each hyperparameter on each other and the respective partial dependence to the target metric for the initial tuning of the ARXT model with Log-Normalised data. The target variable is reversed in this plot.

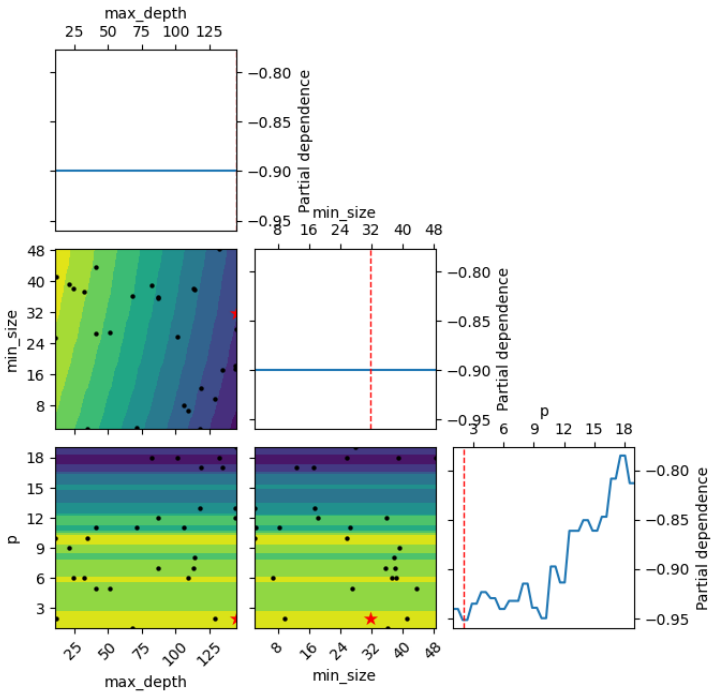
Looking more sharply into the meaning of the Figure, we see for the *min_size*, *max_depth* and *max_weight*, ring shaped contours. These are consistent across all three and contrast starkly to the lines found in the the bottom row, corresponding to p. This can be explained by the partial dependence plots, as seen in the Figure, the only non horizontal partial dependence is that of p. This indicates that the effect of changing the other hyperparameters is going to be

minimal, yet there is still a correlation between higher values of the target function and certain parameter pairs, leading to the clear contour lines. The areas with lighter shading should be more populated than the rest considering Algorithm 2 focuses on areas of maximal improvement. The darker shaded areas are also explored, yet to a lesser extent as it is often the case that the algorithm thinks that there is still a possibility of improvement in certain areas.

When the results in Figure 4 are considered, a big difference is seen to that of the Log-Normalised data. Instead of the much clearer congregation of the parameters, the spread is much greater. This is due to the tuning process being less clear on the optimisation across different feature spaces. The rings seen in the Log-Normalised plot are reversed in this case, as the algorithm finds clear areas where the optimum is not found. Again the partial dependence on p is much more significant than that of the other hyperparameters.

Figure B2 gives an insight into the simpler ART model, given that it does not have the additional parameter of *max_weight*. Similar results are again seen, with clear contouring on the relative dependence between *max_weight* and *min_size*, leading to a more distinct clustering of parameters for the Log-Normalised dataset. Again the importance of p over the rest is evident by the partial dependence plots.

Figure (4) Bayesian Dependence Plot ARXT Differenced



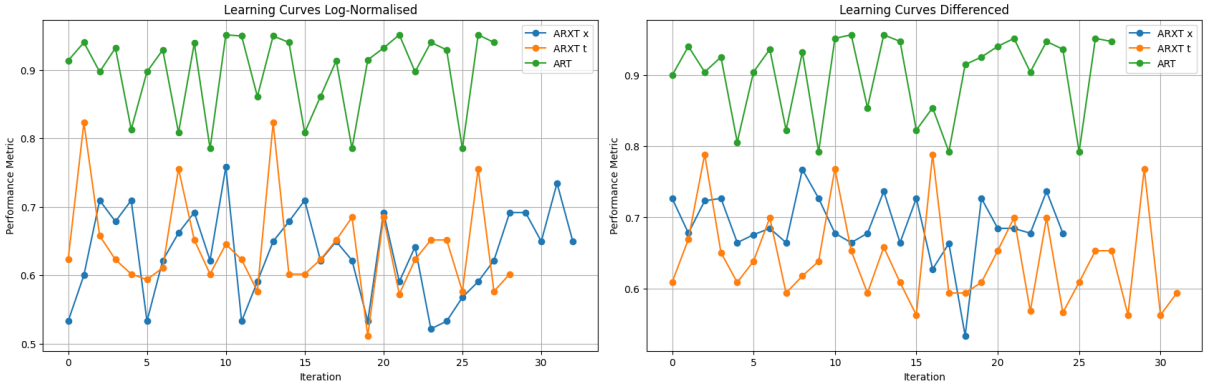
Note. The figure shows the cross dependence of each hyperparameter on each other and the respective partial dependence to the target metric. for the initial tuning of the ARXT model with Log-Normalised data.

Having seen the specific hyperparameters chosen and their optimums, it is interesting to see the development of the model performance over iterations. In Figure 5, the learning curves for the two datasets are given. Here the development of the target metric, as defined in equation 21, is plotted against the iterations of the initial hyperparameter tuning. This is done for the ARXT exog, ARXT train and ART models.

As we are looking to maximise the target metric, the initial search of 20 iterations gives an

optimum of 0.76 which is then not improved upon by the further iterations, with the closest being 0.73. As there are no improvements found over the first optima, the corresponding hyperparameters are found to be: $max\_depth = 37$, $max\_weight = 0.06$, $min\_size = 35$, $p = 14$. Looking at the learning curve for the ARXT t model, there are similar properties, with a maximum found at the second initial search. This same maximum is found again, with the only constant hyperparameter across the maxima being p. This leads to results of: $max\_depth = 68$, $max\_weight = 0.11$, $min\_size = 1$, $p = 7$. The model learning curve also shows how the addition of exogenous variables in the split decisions attains slightly forecasting results for the out of sample test with which the target metric is decided. As for the ART model, clearly higher results are attained in terms of the target metric. The optimum is found on the 11th iteration with a corresponding metric of 0.95 and parameters $max\_depth = 145$, $min\_size = 32$, $p = 2$.

Figure (5) *Learning Rates for ARXT*



Note. The Figure shows the different learning rates for the tuned and trained variations of the ARXT x and t model. The left figure gives the learning rate for the Log-Normalised data and the right gives the learning rate for the Differenced data.

The behaviour of the learning curves for the Differenced data exhibits similar properties. The maxima for ARXT x is found at the first iteration, with the deviations to the hyperparameters proving not to be effective in improving the results of the parameters. The optimal hyperparameters chosen are: $max\_depth = 36$, $max\_weight = 0.06$, $min\_size = 35$, $p = 14$. For the ARXT t model a slightly higher optima is seen, with $max\_depth = 68$, $max\_weight = 0.11$, $min\_size = 1$, $p = 7$. Again the ART model attains a higher metric with results of $max\_depth = 11$, $min\_size = 26$, $p = 10$, corresponding to a optima of 0.96.

Table (6) Lengths of re-tunings

	Log-Normalised			Differenced		
Split	ARXT x	ARXT t	ART	ARXT x	ARXT t	ART
2	16	15	13	17	16	13
3	14	13	12	19	13	15
4	15	14	11	12	16	15
5	12	17	11	15	12	15
6	17	17	13	14	16	12
7	12	15	10	15	15	13
8	13	17	10	13	15	14

Note. This Table presents the different iterations needed for each model, for each dataset across each split.

While the actual target scores are presented above, it is also of interest to look into the number of iterations required to converge. Figure 5 shows how the ART model only needs 27 and 28 iterations to converge, whereas the ARXT models vary a lot more in the number of iterations. This ranges from 24 to 33. There are more hyperparameters to search through, even if the partial dependence of the parameters outside of p are minimal. Table 6 gives the number of iterations needed for the re-tunings with both Differenced and Log-Normalised datasets. On average the number of iterations needed for the Log-Normalised data is less than that of the Differenced data by one iteration. This goes as well for the spread of the Differenced data, showing that the data is harder to fit and there is more room for the model to improve its hyperparameters.

This section concludes the hyperparameter tuning through the methods introduced in Section 3.2. Of the hyperparameters used, the only one with a large effect is the value p. So when testing the effectiveness of the improvement from tuning it should be considered that this tuning framework may be more effective for a deep learning model such as a Neural Network, where the partial dependence of each hyperparameter is higher. This sections results hint at the performance of the different models, further evaluation and testing on the framework is given in the following section.

5.3 Model Forecasting

5.3.1 Full Dataset

For this results section, the table in Section 3.3.1 displays the different methods tested for the the data. There are eleven different methods, of which six are are different versions of this papers ARXT model, allowing for the addition of exogenous variables in splits and the effectiveness of re-tuning or re-training the model. The performance of the models is considered through the Root Mean Squared Error (RMSE), Directional Accuracy (DA) and improvement with respect to the AR(p) model, tested by the Diebold Mariano (DM) test.

The top four lines of Table 7 present the forecast results of all eleven models over the full forecasting period. The ARXT models exhibit varied performance, their relative RMSE values range from 0.08 to 0.22, while the ART models show slightly lower RMSE values, indicating a better fit with values of -0.04 to 0.35. Notably, the untouched ART model, with a relative RMSE of -0.07, shows an even stronger fit. In terms of Directional Accuracy (DA) with normalised data, the ART models, after tuning and re-training, do not consistently outperform the AR model, with relative DA's as low as -0.11. The ARXT models with exogenous variables (ARXT x) and target (ARXT t) variations have relative DA scores spanning from 0.02 to 0.17, with the highest DA of 0.17 belonging to the base ART train model. This shows that while the RMSE may not necessarily be better than that of the AR model, the ARXT models are able to leverage information better in predicting the direction of price changes. The ARX models are evidently affected by noise too much by the results, with both variants underperforming against the rest of the models. This could also be due to the Least Squares estimation used in both the ARX and AR models.

When examining the differenced dataset, the relative RMSE for ARXT models show a significant spread from -0.31 to 0.11, while the ART models maintain a smaller range of -0.15 to -0.18.

This may indicate a greater consistency in the ART forecasts models compared to the ARXT variants. The DA scores for the ART model range from -0.01 to 0.07, which is comparable to or worse than the ARXT models, signifying reliable predictive performance. Again, it can be seen that the models that have not been retuned or trained after changepoints seem to perform better in forecasting accuracy and direction. The ARX model underperforms even more when applied to the differenced dataset with the lowest RMSE and DA of all the models.

The last two lines of Table 7 details the Diebold-Mariano (DM) test statistics, evaluating the forecasting accuracy of the models against the benchmark AR model. The Normalised DM and Differenced DM values for the ARXT models range dramatically, with some models showing negative values indicating improvement relative to the AR model, and others showing high positive values, particularly in normalised data, suggesting improved performance. Focusing on the normalised data, the ARXT models all underperform relative to the AR model, with only the ARXT t model being seen as statistically similar. When regarding the differenced data, all ARXT models are either statistically similar or have better performance than the AR model, a reflection of the RMSE results. For both datasets, the ART model exhibits consistently negative DM values, reflecting a robustness to different datasets. The ARX models show very positive DM statistics, especially in differenced data, with values around -1.33 to -6.72, pointing to a substantial deviation from the AR model's performance.

These findings suggest that while the ARXT models have some strengths, particularly in directional accuracy, the AR model generally provides a robust baseline. The variability in the ARXT models' performance across different data transformations underscores the complexity of model selection for forecasting purposes. The second conclusion that can be drawn from these forecasting results is the initial sign of better performance of the models without re-tuning and training.

Table (7) *Forecast Results RMSE and Directional Accuracy Full Dataset Log-Normalised and Differenced data*

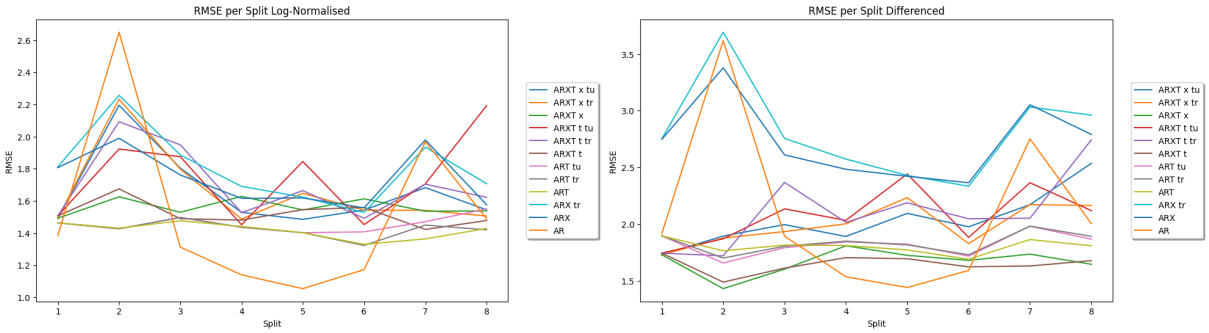
	ARXT x tu	ARXT x tr	ARXT x	ARXT t tu	ARXT t tr	ARXT t	ART tu	ART tr	ART	ARX tr	ARX
Norm RMSE	0.20	0.20	0.08	0.21	0.22	0.03	-0.04	-0.05	**-0.07**	0.35	0.29
Norm DA	0.02	0.03	0.08	0.02	0.00	**0.11**	-0.11	-0.12	-0.11	-0.15	-0.14
Diff RMSE	-0.01	0.01	-0.31	0.11	0.00	**-0.34**	-0.16	-0.15	-0.18	0.83	0.75
Diff DA	0.02	0.02	0.14	0.01	-0.01	**0.17**	0.00	-0.01	0.07	-0.31	-0.31
Norm DM	3.60 (0.00)	3.87 (0.00)	2.20 (0.03)	4.56 (0.00)	3.61 (0.00)	**0.88 (0.38)**	**-1.33 (0.18)**	**-1.68 (0.09)**	-2.44 (0.01)	6.06 (0.00)	5.29 (0.00)
Diff DM	**-0.26 (0.80)**	**0.26 (0.79)**	-8.99 (0.00)	2.49 (0.01)	**0.07 (0.94)**	-9.40 (0.00)	-5.48 (0.00)	-5.26 (0.00)	-6.72 (0.00)	10.92 (0.00)	10.81 (0.00)

Note. The top four lines of the table show the relative RMSE and Directional accuracy compared to the AR model for the entire forecasting period. The best performers (to 4 d.p.) for each metric are given in bold. The last two lines give the results from the Diebold Mariano tests, using the AR(p) model as a benchmark. The values are the test statistic from the test with the corresponding p-value in brackets. If the respective model is significantly similar to the AR(p) at the 5% level, then this is indicated in bold text.

5.3.2 Split Dataset

In the previous section the results show a superiority of the ART models and the ARXT models without tuning or training. The performance is further evaluated by looking into the results through the different splits. To start, the different RMSE values for the models is plotted in Figure 6. This gives a quick overview of the performance across the different splits.

Figure (6) RMSE across splits



Note. This figure is split into two parts. The left plots all the different RMSE values for the different models across all splits with Log-Normalised data, while the right does the same for the Differenced data.

Differing behaviour is clear across the samples, with a bigger range for the AR model compared to the rest. Looking at the left Figure of 6, the results from Table 7 can be deciphered. The ART models give a consistent RMSE across the splits with slight deviations after tuning and training. Conversely, the ARXT models have a much larger range of results, the same starting points diverging majorly throughout the dataset. Looking closer into the ARXT models, the tuned and trained models see a distinct increase in performance across splits 3 to 6, this can be explained by the lower volatility across those splits. The same holds for the AR model, with an RMSE as low as 1.1 for split 5.

When considering the differenced data, a similar patter of the performance is seen across the models. Across the board, models have aligned performance for the different splits, only differing in magnitude, regardless of the data processing method. Narrowing our focus onto the best performers, the raw ARXT t and ARXT x models show how they consistently beat all other models, with the exception of splits 4 through 6 to the AR model. This benchmark AR model is comparatively worse than for the normalised data, with big jumps seen in the second and seventh split. The ART models again perform well, slightly worse than the ARXT t and ARXT x model, and show consistency across different data sections.

Looking into the directional accuracy scores of all models in Table C1, there is a distinct drop in the scores from the second split, with the exception of the ARXT t and ARXT x models. These keep a DA score above 0.55 up to the last split, which only has 36 values. For the differenced data, in Table C2, a similar pattern is again seen. The untuned and untrained models keeping a consistently high score across the splits, while the tuning and training has an adverse effect.

In Tables C1 and C2 the above is again seen. A clear positive begin from the ARXT t and ARXT x models, with the remaining splits having either significantly better results for the remaining splits or significantly similar results to the benchmark AR model. The differenced data reflects the comparatively worse AR model, with results significantly close to the ARXT

tu and ARXT tr models for both variants. The ART model see's very similar results to the AR model for all variants in the normalised dataset, with very negative values in the differenced data, up to the last two splits.

From the results in this section so far, we can clearly see a sensitivity to the changing distribution of the data. This could be due to the way that the models are retrained. The model looks at the past 600 observations before the changepoint, when training for the next split of data. This indicates that a sub sample of the data under the new distribution should be used to train the model again to fully encapsulate the effects we search for. However, it is positive to see the ability of the two ARXT models leveraging the exogenous variables, in its ability to predict correct price changes better than its simpler variant and the benchmark models.

5.3.3 re-tuning evaluation

In order to effectively analyse the effect of the hyperparameter tuning and training after changepoints take place, a further test is taken in which each retuned and retrained model is compared to its model without extra training/tuning. In Table 8, the Diebold Mariano test is used to test for significant difference in forecasts, with the first split being ignored as the models have not been retuned or trained at that point. For the full dataset, none of the retuned or retrained models show improvement, as seen in previous sections. However, when looking into the individual splits we see either significant similarity, weighing to improvement, or negative score on the splits 4 through 6 in the ARXT x model. For the ARXT t model, the effectiveness of re-tuning is not seen, with only split 4 and 6 giving negative results, even if the re-training model shows significant similarity from split 4. When considering the ART model, a model which performed slightly worse than the untuned and untrained ARXT variants, improvements can be seen from the first re-tuning. It is clear to see here how the re-tuning has less of an effect as both the tuned and trained models result in the same statistics for all but the last three splits. This is due to the same hyperparameters being chosen and both models going through the same re-training. As for the results themselves, the effectiveness varies again, with the third split suffering from bad training data.

Considering Table 8's counterpart, Table B5 in the Appendix , we see that the effectiveness of re-tuning is negative across the board. Only in two splits, 2 and 3, is there an improvement seen by the ART model. This shows how the re-tuning is not holding up to the differenced data.

Overall, the results show how the re-tuning and re-training of the model is most effective when we don't see big variations in the data. This effect is converse when the methodology is applied to differenced data, and the method of re-tuning should be considered. This does not take away from the results of the ARXT x and ARXT t models, given their superior DA scores, robust over different data sets, which is very useful to people looking to use the model in predicting financial movements. Given more testing into the re-tuning framework, these results could also be seen to be enhanced further

Table (8) *Relative Results Diebold Mariano Test Log-Normalised*

	ARXT x tu	ARXT x tr	ARXT t tu	ARXT t tr	ART tu	ART tr
full	2.11 (0.04)	2.42 (0.02)	3.27 (0.00)	3.17 (0.00)	**1.82 (0.07)**	2.66 (0.01)
2	2.16 (0.03)	2.98 (0.00)	**1.06 (0.29)**	**1.96 (0.05)**	**-1.06 (0.29)**	**-1.06 (0.29)**
3	2.74 (0.01)	2.82 (0.01)	3.98 (0.00)	4.49 (0.00)	2.14 (0.03)	2.14 (0.03)
4	**-1.86 (0.06)**	-2.67 (0.01)	**-0.76 (0.45)**	**1.18 (0.24)**	**-1.63 (0.10)**	**-1.63 (0.10)**
5	**-1.49 (0.14)**	2.20 (0.03)	3.59 (0.00)	**1.85 (0.06)**	**-0.02 (0.98)**	**-0.02 (0.98)**
6	**-0.72 (0.47)**	**-0.72 (0.47)**	**-1.24 (0.22)**	**-0.72 (0.47)**	2.58 (0.01)	**-0.46 (0.65)**
7	**1.01 (0.31)**	**0.04 (0.97)**	2.17 (0.03)	**1.34 (0.18)**	**1.52 (0.13)**	2.83 (0.00)
8	**0.04 (0.97)**	**-0.16 (0.87)**	2.62 (0.01)	**0.57 (0.57)**	**0.86 (0.39)**	**-0.98 (0.34)**

Note. The Table shows the results from the Diebold Mariano tests, using the respective models without re-tuning or re-training as a benchmark for the different splits and the full dataset. The values are the test statistic from the Diebold Mariano test with the corresponding p-value in brackets. If the respective model is significantly similar to its basic model at the 5% level then this is indicated in bold text. Negative values indicate an improvement over the comparative model.

6 Conclusion

This paper presents a novel approach to forecasting financial market data, employing advanced econometric techniques such as changepoint detection, hyperparameter re-tuning, and advancements to Meek et al. (2002) ART model. This presents the two aims of the paper: will incorporating exogenous variables in the ART model improve its performance and if re-tuning hyperparameters following changepoints is effective.

The changepoint detection algorithm accurately highlighted key moments of market shifts, such as the 2008 financial crisis and the onset of the COVID-19 pandemic. These moments were not only pivotal from a financial perspective, but also marked significant changes in market dynamics and investor behavior. The alogrithm proved to be flexible in the priors it needed, while adapting well to changing input data. This complemented the tuning framework well with its ability to update values based on a steady data stream.

The ARXT models showed that incorporating the exogenous variables into the model leads to improvements. Using both Log-Normalised and Differenced stock returns, and a host of different indicators the models were used to predict stock returns over a period of sixteeen years, with approximately four years to train and tune the original models. In order to tune the hyperparameters of the model, Bayesian Optimisation was utilised. The results from the optimisation indicated that the model clearly found the best set of hyperparameters, yet the relevance of all but the lag order p was marginal. With the importance of the sole hyperparameter p, the effectiveness of implementing the advanced Bayesian Optimisation over another method, such as a random search, was limited.

However, given the initial training, all 11 model variations were tested against the benchmark AR model. Results show that the models without tuning or training scored best. With the ART model attaining a better RMSE than the benchmakr AR across both datasets. The ARXT models that did not have further tuning and training performed remakeably well. They comfortably beat the benchmakr and the ART model with their Differenced RMSE and achieved a Directional Accuracy (DA) score of over 60%. This shows the inert ability of the model to harness exogenous factors into model predictions and predict price changes better than any other

of the tested models. This performance did not carry over to the retuned and retrained models, with the initial performance dropping after the first data split made by the changepoints. The reason for this is due to the training dataset used for the models, with the last 600 observations being used to train at the point where the changepoint is found. Given this further testing may be useful in the best dataset to use when re-tuning the models in order to optimise the models to the changing underlying distribution of the data.

As with any paper, this research is not without its limitations. The sole focus on S&P500 data limits the exploration into alternative datasets. This could be expanded outside of US stocks, or even into datasets such as retail, commodities, traffic or sentiment, as influential external factors can be exploited by the model. The use of the ARX model in leaf nodes may be changed to a different model such as a ARIMA or VAR. This could then be further expanded by allowing feature selection in node models, in both the lag order used and the variables chosen. Another approach could be to use an information criteria such as the AIC to evaluate the relevance of certain factors to the model predictions. The tuning framework could also be enhanced by taking even more information into the model, such as passing on the expected improvement values to the next tuning, instead of using the optimal hyperparameters as a starting point.

In conclusion, this paper contributes to the field of econometrics by introducing the ARXT model which excels in predicting price changes, with easy application to different datasets. The re-tuning framework, based on changepoint detection, is one that can easily be applied to any model. Using models with more influential hyperparameters, the framework could drastically increase performance at a relatively cheap computational cost.

References

Adams, R. P. & MacKay, D. J. (2007). Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*.

Bayes, T. (1763). Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*(53), 370–418.

Bayesian Optimization Contributors. (2024). *Bayesian optimization.* `https://github.com/bayesian-optimization`. GitHub.

Box, G. E., Jenkins, G. M., Reinsel, G. C. & Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Breiman, L. (2017). *Classification and regression trees*. Routledge.

Cavalcante, R. C., Minku, L. L. & Oliveira, A. L. (2016). Fedd: Feature extraction for explicit concept drift detection in time series. In *2016 international joint conference on neural networks (ijcnn)* (pp. 740–747).

Chickering, D., Meek, C. & Rounthwaite, R. (2001). Efficient determination of dynamic split points in a decision tree. In *Proceedings 2001 ieee international conference on data mining* (p. 91-98). doi: 10.1109/ICDM.2001.989505

Chickering, D. M., Meek, C. & Rounthwaite, R. (2001). Efficient determination of dynamic split points in a decision tree. In *Proceedings 2001 ieee international conference on data mining* (pp. 91–98).

Chipman, H. A., George, E. I. & McCulloch, R. E. (2010). BART: Bayesian additive regression trees. *The Annals of Applied Statistics*, *4*(1), 266 – 298. Retrieved from `https://doi.org/10.1214/09-AOAS285` doi: 10.1214/09-AOAS285

Da, Z., Liu, Q. & Schaumburg, E. (2014). A closer look at the short-term return reversal. *Management science*, *60*(3), 658–674.

Diebold, F. X. & Mariano, R. S. (2002). Comparing predictive accuracy. *Journal of Business & economic statistics*, *20*(1), 134–144.

Du, L., Gao, R., Suganthan, P. N. & Wang, D. Z. (2022). Bayesian optimization based dynamic ensemble for time series forecasting. *Information Sciences*, *591*, 155-175. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0020025522000135` doi: https://doi.org/10.1016/j.ins.2022.01.010

FRED. (2023). *Federal reserve economic data.*

Fuller, D. A. D. . W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, *74*(366a), 427-431. doi: 10.1080/01621459.1979.10482531

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, *46*(4), 1–37.

Gauvain, J.-L. & Lee, C.-H. (1994). Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING*, *2*(2).

Heckerman, D. & Geiger, D. (2013). Learning bayesian networks: a unification for discrete and gaussian domains. *arXiv preprint arXiv:1302.4957*.

Huber, F. & Rossini, L. (2022). Inference in bayesian additive vector autoregressive tree models. *The Annals of Applied Statistics*, *16*(1), 104–123.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media.

Meek, C., Chickering, D. M. & Heckerman, D. (2002). Autoregressive tree models for time-series analysis. In *Proceedings of the 2002 siam international conference on data mining* (pp. 229–244).

Nekoie, H. (2024). *Autoregressive tree model*. `https://github.com/hnekoeiq/autoregressive-tree`. GitHub.

Pan, S. J. & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

Probst, P., Wright, M. N. & Boulesteix, A.-L. (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: data mining and knowledge discovery*, *9*(3), e1301.

Snoek, J., Larochelle, H. & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, *25*.

Tsymbal, A. (2004). The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*, *106*(2), 58.

Tuncel, K. S. & Baydogan, M. G. (2018). Autoregressive forests for multivariate time series modeling. *Pattern recognition*, *73*, 202–215.

A Bayesian Optimisation Algorithm

Algorithm 3 Bayesian Online Changepoint Detection

1. **Initialise:**

$$P(r_0) = \tilde{S}(r) \text{ or } P(r_0 = 0) = 1$$
$$\nu_1^{(0.00)} = \nu_{\text{prior}}, \boldsymbol{\chi}_1^{(0.00)} = \boldsymbol{\chi}_{\text{prior}}$$

2. **Observe New Datum:** x_t
3. **Evaluate Predictive Probability:** $\pi_t^{(r)} = P(x_t|\nu_t^{(r)}, \boldsymbol{\chi}_t^{(r)})$
4. **Calculate Growth Probabilities:** $P(r_t = r_{t-1}+1, \mathbf{x}_{1:t}) = P(r_{t-1}, \mathbf{x}_{1:t})\pi_t^{(r)}(1-H(r_{t-1}))$
5. **Calculate Changepoint Probabilities:**

$$P(r_t = 0, \mathbf{x}_{1:t}) = \sum_{r_{t-1}} P(r_{t-1}, \mathbf{x}_{1:t})\pi_t^{(r)} H(r_{t-1})$$

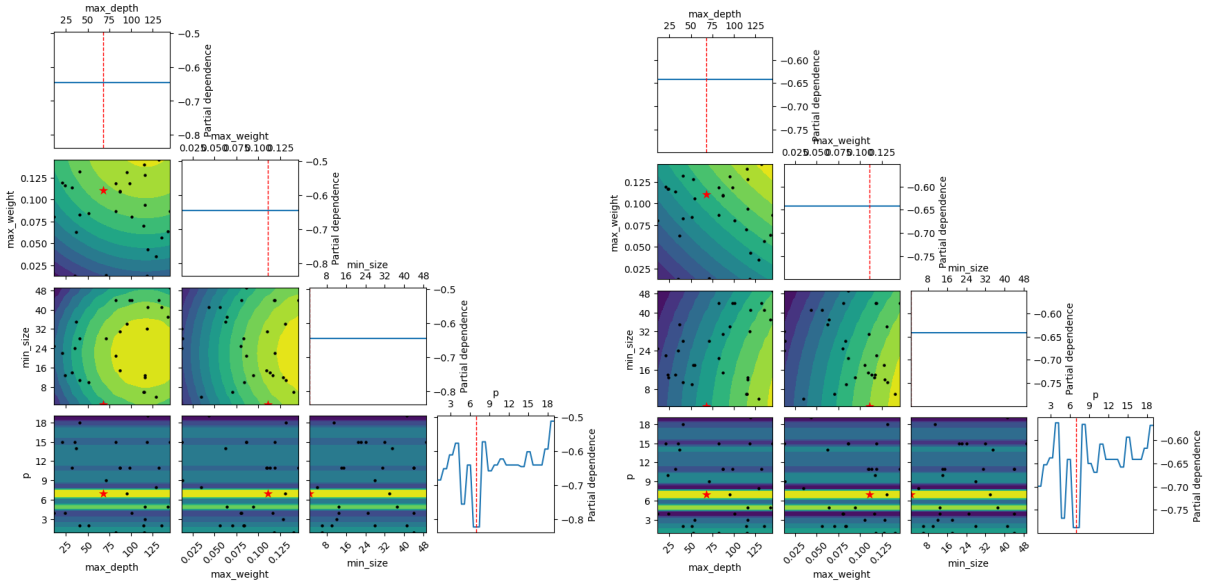
6. **Calculate Evidence:** $P(\mathbf{x}_{1:t}) = P(r_t, \mathbf{x}_{1:t})$
7. **Determine Run Length Distribution:** $P(r_t|\mathbf{x}_{1:t}) = P(r_t, \mathbf{x}_{1:t})/P(\mathbf{x}_{1:t})$
8. **Update Sufficient Statistics:**

$$\nu_{t+1}^{(0.00)} = \nu_{prior}$$
$$\boldsymbol{\chi}_{t+1}^{(0.00)} = \boldsymbol{\chi}_{prior}$$
$$\nu_{t+1}^{(r+1)} = \nu_t^{(r)} + 1$$
$$\boldsymbol{\chi}_{(t+1)} = \boldsymbol{\chi}_t^{(r)} + \mu(x_t)$$

9. **Perform Prediction:** $P(x_{t+1}|\mathbf{x}_{1:t}) = \sum_{r_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t^{(r)}, r_t)P(r_t|\mathbf{x}_{1:t})$
10. **Return information:** If the probability of a changepoint is over 90% then pass this information back.
11. **Return to Step 2**

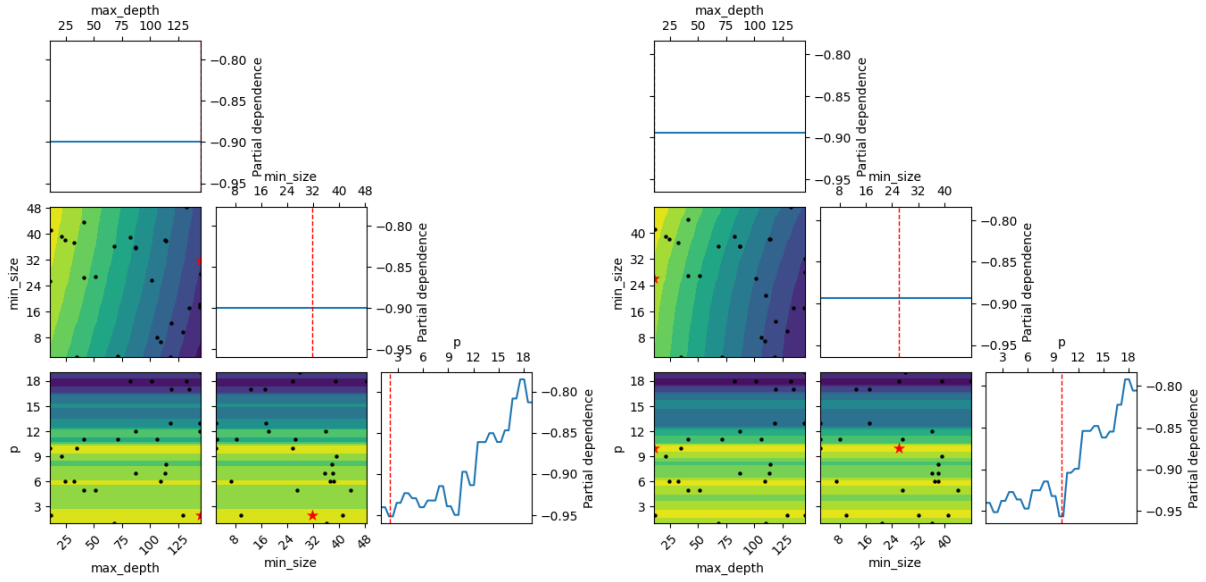
B Hyperparameter Tuning

Figure (B1) *Dependence Plots ARXT t Hyperparameters*



Note. The figure shows the cross dependence of each hyperparameter on each other and the respective partial dependence to the target metric for the initial tuning of the ARXT t model. The left figure uses Log-Normalised data, while the right figure is using Differenced data.

Figure (B2) *Dependence Plots ART Hyperparameters*



Note. The figure shows the cross dependence of each hyperparameter on each other and the respective partial dependence to the target metric for the initial tuning of the ART model. The left figure uses Log-Normalised data, while the right figure is using Differenced data.

C Model Forecasting

Table (C1) *Forecast Results Relative RMSE and Directional Accuracy Log-Normalised*

	ARXT x tu	ARXT x tr	ARXT x	ARXT t tu	ARXT t tr	ARXT t	ART tu	ART tr	ART	ARX tr	ARX
1 RMSE	0.00	0.00	0.00	0.02	0.02	0.02	**-0.03**	**-0.03**	**-0.03**	0.32	0.32
1 DA	0.06	0.06	0.06	**0.10**	**0.10**	**0.10**	-0.15	-0.15	-0.15	-0.20	-0.20
2 RMSE	0.71	0.74	0.14	0.43	0.60	0.19	**-0.06**	**-0.06**	-0.06	0.77	0.50
2 DA	0.08	**0.10**	0.09	0.09	0.09	0.09	-0.08	-0.08	-0.09	-0.12	-0.13
3 RMSE	0.32	0.31	0.04	0.39	0.46	0.00	0.01	0.01	**-0.01**	0.40	0.27
3 DA	0.10	0.11	0.09	0.07	0.05	**0.11**	-0.06	-0.06	-0.07	-0.12	-0.12
4 RMSE	0.04	0.00	0.14	-0.04	0.04	-0.01	**-0.05**	**-0.05**	-0.05	0.20	0.13
4 DA	0.04	0.04	0.07	0.05	-0.02	**0.11**	-0.10	-0.10	-0.10	-0.15	-0.15
5 RMSE	0.00	0.16	0.06	0.36	0.18	0.06	**-0.09**	**-0.09**	**-0.09**	0.14	0.13
5 DA	-0.07	-0.07	0.06	-0.09	-0.07	**0.12**	-0.15	-0.15	-0.12	-0.17	-0.17
6 RMSE	0.05	0.05	0.12	-0.04	0.00	0.07	-0.08	**-0.17**	-0.16	0.04	0.07
6 DA	0.04	0.04	0.11	0.01	-0.02	**0.15**	-0.07	-0.06	-0.05	-0.09	-0.08
7 RMSE	0.19	0.05	0.05	0.22	0.22	-0.07	-0.02	-0.04	**-0.12**	0.45	0.49
7 DA	0.01	0.03	**0.11**	0.01	-0.01	0.08	-0.11	-0.15	-0.10	-0.14	-0.12
8 RMSE	0.06	0.02	0.05	0.70	0.13	-0.01	0.06	**-0.07**	-0.06	0.22	0.09
8 DA	0.03	0.00	-0.03	-0.09	-0.09	**0.14**	-0.14	-0.20	-0.20	-0.06	-0.06

Note. The Table shows the relative RMSE and Directional accuracy (DA) for each split for all 11 models to the AR(p) model. The best performers (to 4 d.p.) for each split are given in bold. The results are from the Log-Normalised dataset. After the ARXT, x means that it considers exogenous variables as a splitting point, whereas t is only considering the target value as a splitting point. Tune means that the model is retuned after each changepoint and train indicates re-training the model after changepoints.

Table (C2) *Forecast Results Relative RMSE and Directional Accuracy Differenced*

	ARXT x tu	ARXT x tr	ARXT x	ARXT t tu	ARXT t tr	ARXT t	ART tu	ART tr	ART	ARX tr	ARX
1 RMSE	**-0.27**	**-0.27**	**-0.27**	-0.26	-0.26	-0.26	-0.11	-0.11	-0.11	0.75	0.74
1 DA	0.11	0.11	0.11	**0.17**	**0.17**	**0.17**	0.02	0.02	0.02	-0.35	-0.35
2 RMSE	-0.11	-0.13	**-0.57**	-0.13	-0.28	-0.52	-0.35	-0.30	-0.24	1.69	1.37
2 DA	0.18	0.15	0.15	0.12	0.13	0.15	**0.19**	0.15	0.08	-0.29	-0.31
3 RMSE	-0.01	-0.07	**-0.40**	0.13	0.36	-0.39	-0.21	-0.20	-0.19	0.75	0.61
3 DA	0.10	0.14	0.15	0.10	0.09	**0.16**	0.06	0.05	0.08	-0.29	-0.30
4 RMSE	-0.11	0.00	-0.20	0.03	0.01	**-0.30**	-0.16	-0.15	-0.19	0.57	0.48
4 DA	0.04	0.00	0.13	0.01	-0.07	**0.16**	0.02	-0.01	0.07	-0.31	-0.31
5 RMSE	0.09	0.23	-0.28	0.44	0.18	**-0.31**	-0.18	-0.19	-0.23	0.43	0.42
5 DA	-0.11	-0.11	0.10	-0.13	-0.10	**0.17**	-0.07	-0.06	0.04	-0.33	-0.34
6 RMSE	-0.03	-0.18	-0.32	-0.12	0.04	**-0.38**	-0.29	-0.27	-0.31	0.33	0.36
6 DA	-0.04	0.06	0.18	0.06	-0.04	**0.21**	-0.01	-0.02	0.12	-0.25	-0.24
7 RMSE	0.17	0.17	-0.27	0.36	0.05	**-0.37**	-0.02	-0.02	-0.14	1.03	1.05
7 DA	-0.02	0.01	**0.17**	-0.05	-0.07	0.15	-0.10	-0.10	0.11	-0.29	-0.28
8 RMSE	0.53	0.16	**-0.36**	0.11	0.74	-0.33	-0.14	-0.11	-0.20	0.96	0.79
8 DA	-0.06	-0.14	0.14	0.03	-0.11	**0.23**	-0.09	0.00	0.06	-0.26	-0.26

Note. The Table shows the relative RMSE and Directional accuracy (DA) for each split for all 11 models to the AR(p) model. The best performers (to 4 d.p.) for each split are given in bold. The results are from the Differenced dataset. After the ARXT, x means that it considers exogenous variables as a splitting point, whereas t is only considering the target value as a splitting point. Tune means that the model is retuned after each changepoint and train indicates re-training the model after changepoints.

Table (C3) *Forecast Results Diebold Mariano Test Log-Normalised Splits*

	ARXT x tu	ARXT x tr	ARXT x	ARXT t tu	ARXT t tr	ARXT t	ART tu	ART tr	ART
1	**0.06 (0.95)**	**0.06 (0.95)**	**0.06 (0.95)**	**0.29 (0.77)**	**0.29 (0.77)**	**0.29 (0.77)**	**-0.57 (0.57)**	**-0.57 (0.57)**	**-0.57 (0.57)**
2	2.67 (0.01)	3.61 (0.00)	**1.13 (0.26)**	2.25 (0.02)	2.68 (0.01)	**1.13 (0.26)**	**-0.65 (0.52)**	**-0.65 (0.52)**	**-0.6 (0.55)**
3	3.17 (0.00)	3.26 (0.00)	**0.49 (0.62)**	4.11 (0.00)	5.11 (0.00)	**0.01 (0.99)**	**0.11 (0.91)**	**0.11 (0.91)**	**-0.17 (0.86)**
4	**0.89 (0.37)**	**-0.02 (0.98)**	2.74 (0.01)	**-0.88 (0.38)**	**0.86 (0.39)**	**-0.18 (0.86)**	**-1.27 (0.20)**	**-1.27 (0.2)**	**-1.13 (0.26)**
5	**-0.08 (0.93)**	3.51 (0.00)	**1.17 (0.24)**	4.90 (0.00)	3.26 (0.00)	**1.04 (0.30)**	-2.87 (0.00)	-2.87 (0.00)	-2.81 (0.01)
6	**0.59 (0.55)**	**0.59 (0.55)**	**1.08 (0.28)**	**-0.37 (0.71)**	**0.04 (0.97)**	**0.58 (0.56)**	**-1.40 (0.16)**	-2.08 (0.04)	-2.2 (0.03)
7	**1.71 (0.09)**	**0.36 (0.72)**	**0.41 (0.68)**	**1.92 (0.05)**	**1.04 (0.30)**	**-0.9 (0.37)**	**-0.16 (0.88)**	**-0.36 (0.72)**	**-1.34 (0.18)**
8	**0.26 (0.80)**	**0.10 (0.92)**	**0.23 (0.82)**	**1.95 (0.06)**	**0.68 (0.50)**	**-0.04 (0.97)**	**0.29 (0.77)**	**-0.44 (0.66)**	**-0.38 (0.70)**

Note. The Table shows the results from the Diebold Mariano tests, using the AR(p) model as a benchmark for the different Log-Normalised splits. The values are the test statistic from the Diebold Mariano test with the corresponding p-value in brackets. If the respective model is significantly better than the AR(p) at the 5% level then this is indicated in bold text.

Table (C4) *Forecast Results Diebold Mariano Test Differenced Splits*

	ARXT x tu	ARXT x tr	ARXT x	ARXT t tu	ARXT t tr	ARXT t	ART tu	ART tr	ART
1	-4.49 (0.00)	-4.49 (0.00)	-4.49 (0.00)	-4.09 (0.00)	-4.09 (0.00)	-4.09 (0.00)	-2.69 (0.01)	-2.69 (0.01)	-2.69 (0.01)
2	**-0.53 (0.59)**	**-0.73 (0.47)**	-4.59 (0.00)	**-0.8 (0.42)**	**-1.33 (0.18)**	-3.72 (0.00)	-3.99 (0.00)	-3.75 (0.00)	-3.11 (0.00)
3	**-0.06 (0.95)**	**-0.53 (0.6)**	-4.14 (0.00)	**1.02 (0.31)**	**1.51 (0.13)**	-4.72 (0.00)	-3.81 (0.00)	-3.61 (0.00)	-3.70 (0.00)
4	-2.18 (0.03)	**-0.04 (0.97)**	-3.97 (0.00)	**0.46 (0.64)**	**0.22 (0.83)**	-6.43 (0.00)	-3.61 (0.00)	-3.52 (0.00)	-4.22 (0.00)
5	**1.91 (0.06)**	5.14 (0.00)	-5.18 (0.00)	5.83 (0.00)	3.11 (0.00)	-5.41 (0.00)	-4.97 (0.00)	-5.15 (0.00)	-5.95 (0.00)
6	**-0.31 (0.76)**	**-1.75 (0.08)**	-2.66 (0.01)	**-1.03 (0.30)**	**0.55 (0.58)**	-2.90 (0.00)	-3.10 (0.00)	-2.92 (0.00)	-3.41 (0.00)
7	**1.44 (0.15)**	**1.56 (0.12)**	-2.85 (0.00)	2.81 (0.01)	**0.25 (0.81)**	-3.72 (0.00)	**-0.23 (0.82)**	**-0.23 (0.82)**	**-1.56 (0.12)**
8	**1.59 (0.12)**	**0.90 (0.37)**	-2.05 (0.05)	**0.46 (0.65)**	2.34 (0.03)	**-0.96 (0.34)**	**-1.05 (0.30)**	**-0.89 (0.38)**	**-1.71 (0.10)**

Note. The Table shows the results from the Diebold Mariano tests, using the AR(p) model as a benchmark for the different Differenced splits. The values are the test statistic from the Diebold Mariano test with the corresponding p-value in brackets. If the respective model is significantly similar to the AR(p) at the 5% level then this is indicated in bold text.

Table (B5) *Relative Results Diebold Mariano Test Differenced*

	ARXT x tu	ARXT x tr	ARXT t tu	ARXT t tr	ART tu	ART tr
full	6.27 (0.00)	7.41 (0.00)	8.52 (0.00)	5.70 (0.00)	**1.59 (0.11)**	2.53 (0.01)
2	2.35 (0.02)	2.69 (0.01)	2.16 (0.03)	**1.19 (0.23)**	-2.90 (0.00)	-2.78 (0.01)
3	3.47 (0.00)	3.04 (0.00)	3.87 (0.00)	2.86 (0.00)	**-1.46 (0.15)**	**-0.87 (0.39)**
4	**1.44 (0.15)**	3.70 (0.00)	6.05 (0.00)	7.25 (0.00)	3.21 (0.00)	5.43 (0.00)
5	6.68 (0.00)	8.74 (0.00)	7.64 (0.00)	6.61 (0.00)	5.67 (0.00)	7.35 (0.00)
6	4.15 (0.00)	2.23 (0.03)	3.02 (0.00)	4.29 (0.00)	**1.35 (0.18)**	2.22 (0.03)
7	3.45 (0.00)	3.47 (0.00)	4.25 (0.00)	2.05 (0.04)	2.43 (0.02)	2.43 (0.02)
8	2.18 (0.04)	2.24 (0.03)	**1.52 (0.14)**	2.24 (0.03)	**1.26 (0.21)**	2.13 (0.04)

Note. The Table shows the results from the Diebold Mariano tests, using the respective models without re-tuning or re-training as a benchmark for the different Log-Normalised splits and the full dataset. The values are the test statistic from the Diebold Mariano test with the corresponding p-value in brackets. If the respective model is significantly better than its basic model at the 5% level then this is indicated in bold text. Negative values indicate an improvement over the comparative model.