ARCADA

Explainability of time series models

Helena Aito

Master's Thesis
Master of Engineering - Big Data Analytics
December 27, 2022

MASTER'S THESIS	
Arcada University of Applied Sciences	
Degree Programme:	Master of Engineering - Big Data Analytics
Identification number:	8982
Author:	Helena Aito
Title:	Explainability of time series models
Supervisor (Arcada):	Leonardo Espinosa-Leal
Commissioned by:	Arcada University of Applied Sciences

Abstract:
The lack of interpretability of machine learning models is a drawback of their use. To better understand how the model works, how data affects its performance, how the model could be improved, and to gain trust in the model, investigating the model in more detail is necessary.

This thesis consists of two parts, model development and explainability analysis using S&P 500 index data. To find out what the baseline prediction accuracy is for S&P 500 index forecasting and to keep data preparation works as simple as possible, models were kept simple and features and responses were derived from S&P index data only. In total, five time series regression models were developed to predict S&P 500 index. Explainability of all five models was investigated both at a global and local level by using permutation importance, local interpretable model-agnostic explanations (LIME), and Shapley additive explanation (SHAP).

The best model was gradient boost. The prediction accuracy of the best model was considered sufficient both for a baseline version and explainability analysis. Model explainability was investigated using permutation importance, LIME, and SHAP. Structures of selected models were also visualized. Results from permutation importance, LIME, and SHAP were also compared to find out what the most important features are both on average and at a specific timestep. Potential next steps of this thesis could focus on using deep learning, such as long short-term memory network as well as investigating the usability and necessity of additional explainability methods.

Keywords:	time series forecasting, machine learning, financial data, S&P 500 index, explainable AI
Number of pages:	79
Language:	English
Date of acceptance:	December 27, 2022

CONTENTS

FIGURES

TABLES

ABBREVIATIONS

AI	Artificial Intelligence
ALE	Accumulated Local Effect
CNN	Convolutional Neural Network
CV	Cross-Validation
DL	Deep Learning
EBLR	Explainable Boosted Linear Regression
GRU	Gated Recurrent Unit
GB	Gradient Boosting
ICE	Individual Conditional Expectation
LIME	Local Interpretable Model-Agnostic Explanation
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MDA	Mean Decrease Accuracy
ML	Machine Learning
PDP	Partial Dependence Plot
PI	Permutation Importance
R^2	Coefficient of Determination, R^2 score
RF	Random Forest
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SHAP	Shapley Additive Explanations
SMAPE	Symmetric Mean Absolute Percentage Error
S&P	Standard and Poor's
STEL	Simplified Tree Ensemble Learner
XAI	Explainable Artificial Intelligence

1 INTRODUCTION

Artificial intelligence (AI) is used for different prediction tasks in various fields, such as finance and healthcare (Arrieta et al. 2020). Applications can also affect human lives, such as COVID-19 identification and self-driving cars (Angelov et al. 2021). Because AI can also relate to life or death decisions, it is essential to interpret and communicate about it properly (Rothman 2020). Governments have also started to implement laws relating to explainable models (Goodman & Flaxman 2017, Freeborough & van Zyl 2022). However, the interpretation of complex model structures and predictions has been found challenging because model explainability and predictive accuracy are inversely proportional (Angelov et al. 2021, Freeborough & van Zyl 2022).

Because the models are commonly carrying out critical tasks that can affect human lives, the lack of interpretability is a significant drawback (Arrieta et al. 2020, Rojat et al. 2021). Therefore, it is critical to aim to find the balance between model performance, structure, and use. To better understand how the model works, how different data and structures affect its performance, how its performance could be improved, and to gain confidence in the model predictions and model usage on a specific problem, investigating the model further is necessary. Model behavior can be investigated, for example, by studying how model inputs affect outputs and how the model works at certain levels of its structure. Although explainable AI, or XAI, is a relatively new term, it is evident that it greatly impacts the overall usability of a model. However, to ensure XAI establishes and maintains its position as good practice in model development, the tools for investigating explainability need to be, for example, easily available, easy to use, and as intuitive as possible.

In this thesis, selected time series regression models were developed with Python by using S&P 500 index data. The explainability of models was also investigated by utilizing selected methods found in the literature. Financial data was selected because S&P 500 data is easily available and well-known. Explainability analysis was included in this thesis mainly because of two reasons. Firstly, explainable artificial intelligence has increased interest in research in recent years (Arrieta et al. 2020, Angelov et al. 2021). Secondly, explainability methods available for financial forecasting, or broadly regression, problems have been found rather limited in comparison to those available for classification problems

(Freeborough & van Zyl 2022).

The work is based on three research questions: 1) What is meant by the explainability of AI?, 2) How explainability of AI can be evaluated in practice?, and 3) What evaluation methods can be used for investigating the explainability of time series forecasting models? The first research question is about familiarizing with the term and, for example, summarizing the pros and cons of XAI. The second research question is more practical than the first one by summarizing what methods can be used to investigate model explainability. The purpose of the third question is to show what selected explainability methods output and how the outputs can be interpreted. As it is possible to understand from the research questions, the main goal of this thesis is to show in practice how different explainability methods could be used and how well the selected methods can explain the model behavior. On the side, to investigate model explainability, selected machine learning models are developed for a time series problem.

The thesis consists of seven sections. After the introduction to the thesis, Section 2 covers topics related to the thesis, such as XAI, time series forecasting and S&P 500 index. Section 3 is reserved for summarizing the content of data used in this thesis. Methods used for processing data, developing and testing models, and investigating explainability are covered in Section 4. The remaining Sections 5–7 are reserved for results, discussion, and conclusions, respectively.

2 BACKGROUND

2.1 Artificial intelligence

The term artificial intelligence was already introduced in the 50s (Mehrotra 2019). In practice, AI can be used to refer to a human-like behavior of a machine or system. Thus, a machine that can recognize, learn, reason, or solve problems holds AI (Mehrotra 2019). The behavior has been adapted from an extensive number of examples that define what to detect and how to react to detections.

The relations between AI, machine learning (ML) and deep learning (DL) are shown in Figure 1. On the hierarchy level, AI is at the top, whereas ML follows AI and DL follows ML. ML utilizes statistics to build algorithms that can constantly learn and improve using historical data (Mehrotra 2019). It can be classified into three categories: supervised, unsupervised (Hastie et al. 2009), and reinforcement learning (Sutton & Barto 2018). In supervised learning, the model inputs and outputs are known and the model is trained so that the predicted outputs are close to the true outputs. Supervised learning can be further divided into classification and regression. There are several supervised learning methods available, such as linear regression, decision trees, support vector machines, k-nearest neighbors, tree ensembles, and artificial neural networks. In unsupervised learning, the model outputs are unknown, and the purpose is to detect patterns from the data in use and group data accordingly. Unsupervised learning methods include, for example, different clustering algorithms, principal component analysis, and singular value decomposition. Lastly, reinforcement learning means iterative learning with reinforcement. Depending on the action, correct or incorrect, the reinforcement is either a reward or punishment.

DL is an ML technique. It is based on similar operation principles as a human brain filters information. Deep learning methods typically use neural network architectures and they have several adjustable parameters and layers, resulting in complex structures. The number of layers included in a model can also be used to explain the depth of the model (Mehrotra 2019). There are several architectures of DL available, such as convolutional neural networks (CNN), recurrent neural networks (RNN), and RNN-based long short-term memory (LSTM) and gated recurrent unit (GRU) (Goodfellow et al. 2016).

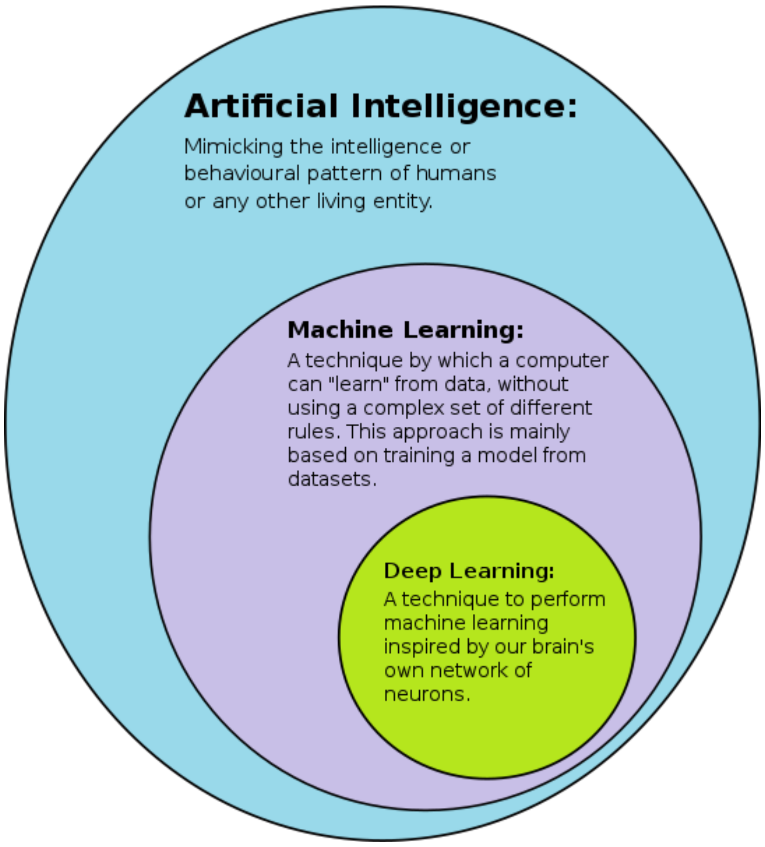

Artificial Intelligence:
Mimicking the intelligence or behavioural pattern of humans or any other living entity.
Machine Learning:
A technique by which a computer can "learn" from data, without using a complex set of different rules. This approach is mainly based on training a model from datasets.
Deep Learning:
A technique to perform machine learning inspired by our brain's own network of neurons.

Figure 1. Differences between AI, ML and DL. Retrieved from Wikimedia Commons (with Creative Commons Licence CC BY-SA 4.0): https://commons.wikimedia.org/wiki/File:AI-ML-DL.svg, author Tukijaaliwa. Accessed 14.11.2022.

2.2 Time series forecasting

2.2.1 Usage of time series data

Time series data represent variables that vary over time. Consequently, time series data is available everywhere. It can be utilized in different industries, such as in the financial and medical fields. By utilizing machine learning, it is possible to train models for various prediction problems in which temporal changes and time dependency matter.

Several prediction tasks are possible with time series data, such as classification, regression, and clustering. Classification means that data is arranged in groups or categories according to learned criteria. In regression, the relation between selected input and output variables is estimated. Regression can also be used for forecasting, in which historical data is used as input to estimate the future trend. Lastly, clustering is used to group similar data points in the same groups.

2.2.2 Examples of forecasting methods

Linear regression

Linear regression is based on the assumption that variables X and Y are linearly connected. The relation between the variables Y and X is visualized by a linear equation

$$Y = A + BX, \tag{1}$$

where A is the intercept between the linear regression and the y-axis. Linear regression example is shown in Figure 2. Blue dots represent the data and the red line represents the fitted line. In Scikit-learn (2022c), a linear model with coefficients $w = w1, ..., wp$ is fitted by minimizing the sum of squared errors between the data points and linear approximation.

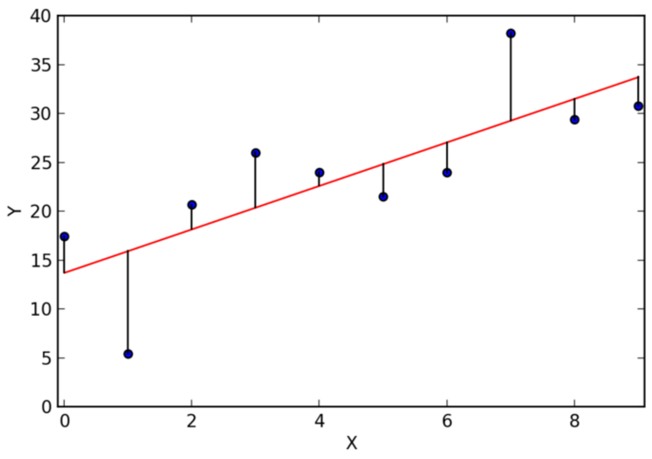


Figure 2. Linear regression example. Retrieved from Wikimedia Commons (with Creative Commons Licence CC BY-SA 3.0): https://commons.wikimedia.org/wiki/File:Residuals_for_Linear_Regression_Fit.png, author Thomas Haslwanter. Accessed 14.11.2022.

Decision tree regressor

Decision tree (Breiman et al. 2017) is a model that is based on a recursive if-else-structure. Due to if and else logic, which utilizes features and corresponding thresholds, the decision tree always splits into two parts, either to another if-else-structure or outputs. The starting point of the tree is called a root node. Regions where the tree splits, are called internal nodes or decision nodes and regions, where responses are defined, are called leaves or terminal nodes. Nodes are connected with branches.

When training a regression decision tree, the aim is to optimize the number of splits by creating branches with similar responses. Different metrics can be used to calculate the quality of a split. In Scikit-learn (2022a), the default is mean squared error. The error is based on the difference between responses in the terminal nodes and the mean of responses in the terminal nodes. It can be calculated according to Equation 2 (Scikit-learn 2022a)

$$H(Q_m) = \frac{1}{n_m} \sum_{y \in Q_m} (y - \bar{y}_m)^2, \tag{2}$$

where the mean of responses in terminal nodes is

$$\bar{y}_m = \frac{1}{n_m} \sum_{y \in Q_m} y \tag{3}$$

and Q_m is data in a terminal node m, n is the number of samples in a terminal node, and y is the response in a terminal node.

K-nearest neighbors regressor

K-nearest neighbors (KNN) method predicts the grouping of a data point. The prediction is based on k nearest known data points or in other words k nearest neighbors. The known data points are stored from training data. Different distance metrics can be used to calculate the distance between the query point and known data points, such as Euclidean distance. In Scikit-learn (2022b), the default distance metric is Minkowski and k is 5. The distance between two data points x and y can be calculated according to Equation 4

$$\text{Minkowski distance} = (\sum_{i=1}^{N} |x_i - y_i|^p)^{\frac{1}{p}}. \tag{4}$$

Tree ensembles

Ensemble methods have become popular in machine learning applications during the last decade (Raschka & Mirjalili 2019). According to Arrieta et al. (2020), tree ensembles are nowadays among the most accurate machine learning modes. Tree ensembles consist of different trees, also called weak learners or base models, to obtain predictions. This structure helps to tackle overfitting, which is a common challenge with decision trees.

The structure of tree ensembles tends to be rather complex due to the numerous decision trees included in the final model. Also, ensembles are computationally more demanding and thus, it is worth comparing computational costs and prediction accuracy critically (Raschka & Mirjalili 2019). Different tree ensembles exist, such as random forest (RF) (Breiman 2001) and gradient boosting (GB) (Friedman 2002). These are briefly covered next.

Random forest regressor

Random forest model consists of multiple decision trees that are averaged. By using multiple decision trees, it is possible for example to generalize the model more, decrease overfitting, and make the model more robust to noise than individual decision trees. (Raschka & Mirjalili 2019)

According to Raschka & Mirjalili (2019), creating a random forest model consists of four main steps: 1) take random n samples from the training data with replacement, 2) train a decision tree with the data extracted in step 1, 3) repeat steps 1 and 2 k times, and 4) collect decision tree predictions to make random forest prediction. Step 2 can be further divided into two parts; at each decision tree node, first, select random d features without replacement and then, split the node using the feature resulting in the best split. With replacement means that the selected sample is returned to the pool of samples, and without replacement means that the sample is excluded from the pool of samples. In step 4, prediction can be made based on majority voting (Breiman 2001) or averaging tree-specific predictions (Scikit-learn 2022e).

Gradient boosting regressor

Like random forest model, gradient boosting model consists of base models, which are trained using random subsets of training data. Gradient boosting can decrease both bias and variance. Creating a gradient boosting model consists of four main steps: 1) train base model C_1 by using random d_1 samples extracted from the training data without replacement, 2) train another base model C_2 by using random d_2 samples extracted from the training data without replacement and 50% of previously incorrectly predicted examples, 3) train third base model C_3 by using data set d_3 consisting of examples that base models C_2 and C_3 were not able to predict correctly, and 4) make gradient boosting prediction by using predictions from base models C_1, C_2 and C_3. (Raschka & Mirjalili 2019)

The concept of gradient boosting is similar to that of AdaBoost (Schapire 1990). According to Raschka & Mirjalili (2019), the way how weights are updated and how the base models are combined differ, though. In Adaboost, complete training data is used to train base models and subsets are reweighted based on prediction error. A new, more computationally efficient version of the gradient boosting method called XGBoost (Chen & Guestrin 2016) exists, too.

Deep learning

As stated by Rojat et al. (2021), state-of-the-art time series forecasting methods have been commonly based on deep learning, especially on recurrent neural networks. RNNs can adapt to the time series trend by learning relations from the trend and updating the network state at each timestep. In addition to RNN, CNNs have been popular, too, due to the possibility of learning relations and extracting features from raw data. In general, deep learning methods have both helped to increase prediction accuracies as well as to reduce the need for data preprocessing.

2.3 Explainable artificial intelligence

2.3.1 Purpose and goals

The purpose of explainable AI, or XAI, is to explain reliably and understandably how models work. Explainability analysis is especially useful for those models that are otherwise challenging to understand as they are. By clarifying how the model works, it is also possible to improve the robustness, confidence, and stability of the model (Rojat et al. 2021). There are several goals for reaching and improving explainability, such as *trustworthiness*, *causality*, *transferability*, *informativeness*, *confidence*, *fairness*, *accessibility*, *interactivity*, and *privacy awareness* (Arrieta et al. 2020).

2.3.2 Terminology

Various terms have been used together explainable AI. Consequently, in addition to becoming familiar with the various tools available for investigating explainability, it is useful to know the related terminology. Some common terms related to explainability are *interpretability*, *comprehensibility*, *understandability*, and *transparency*. These are covered briefly next.

Explainability as a concept is used to clarify and detail model actions and procedures so that the explanation acts as an interface between code and a human. *Interpretability* is, in turn, used to characterize a model at a certain level so that it is possible to explain the model or provide meanings that can be understood by a human. Thus, interpretability is also linked to model transparency. (Arrieta et al. 2020)

Understandability is used to describe whether a human can understand how the model works without the need for clarifying the model structure and how the model handles data. The level of understandability can also be used for evaluating the transparency of the model. (Arrieta et al. 2020)

Comprehensibility can be used to describe the ability of a model to show the learned knowledge in such a way that a human understands it. In practice, the learned knowledge could be shown in a similar way as humans would show it. This could mean, for example, dividing data into smaller chunks of information that are directly interpretable in natural language as well as analyzable both quantitatively and qualitatively. (Arrieta et al. 2020)

Transparency can be used used to describe a model that can be understood as it is. Because there are various models available, the transparency of a model can be further categorized into three different groups: decomposable models, simulatable models, and algorithmically transparent models. (Arrieta et al. 2020)

2.3.3 Challenges, benefits, and opportunities

Explainable artificial intelligence is nowadays considered an important feature for the practical deployment of AI models (Arrieta et al. 2020). Although explainability and interpretability differ from each other slightly, the terms are often mixed. Consequently, as stated by Arrieta et al. (2020), it has been challenging to establish the concept of XAI in the literature. However, as observed in the article, the number of articles related to XAI has increased roughly from 2017 onwards. Thus, it could be expected that more and more people will become familiar with the concept in the near years.

Findings of explainability analysis provide useful information about the model for people working in different positions (Rojat et al. 2021, Arrieta et al. 2020). By better explaining how the model works in a way that humans understand, XAI is also expected to help in reaching more stakeholders. For example, according to Arrieta et al. (2020), the findings of explainability analysis can provide developers better insight into the performance, functionalities, and improvement needs of the model. For product users and domain experts, these findings can provide a better understanding of how the model works from input to output and how reliable the model is. For managers and other company representatives, the findings can provide useful information about the maturity of the product being developed, the operation principles of the product, and how to assess regulatory compliance. Lastly, the findings could also help regulatory representatives in certifying model compliance with the country-specific legislation.

By utilizing the findings of explainability analysis, XAI is also expected to help engineers in reducing the malfunctions of a model (Arrieta et al. 2020). The information collected can also contribute to building more confidence in the models developed (Rojat et al.

2021, Arrieta et al. 2020). Furthermore, XAI could be used to create new metrics and training approaches to ensure that the confidence and robustness of even complex models have been reached (Rojat et al. 2021). XAI has also the opportunity to contribute to a bigger concept called responsible AI, which is based on fairness, ethics, privacy, transparency, security, safety, and accountability (Arrieta et al. 2020).

Explainable AI might not be, at the moment, so intuitive nor beneficial for a wider audience because of the abstract and various metrics that the existing evaluation methods output (Freeborough & van Zyl 2022). It could also be said that there is still a lack of established tools that objectively show the robustness of AI systems (Rojat et al. 2021). Consequently, it would be useful to establish some explainability-related requirements for ML development and improve the existing explainability tools to provide more intuitive results. The requirements could relate to 1) what is the minimum level of explainability analysis needed, 2) what specified methods should be at least used to analyze the model, and 3) what should be reported at least and how. Additional methods could also be used to better understand the model behavior – if needed and if possible.

2.4 Explainability methods

2.4.1 Taxonomy of explainability methods

Explainability methods can be divided into two groups, methods for transparent models and methods for trained models. The latter is also known as post-hoc methods, which can be further divided into model-specific and model-agnostic methods. Model-specific methods stand for methods that are only available to a specific model type, whereas model-agnostic methods mean general methods that can be applied to all models. (Arrieta et al. 2020)

In addition to transparent and post-hoc methods, explainability methods can also be divided into local and global methods. Local methods clarify how the model behaves at a specific instance, whereas global methods clarify how the model works in general (Rojat et al. 2021).

The groups of explainability methods are summarized in Table 1. The first column in the table defines the group and the second column defines the methods available for it. Methods for transparent models include running simulations, decomposing model structures into smaller parts, and inspecting the model at the algorithmic level. Post-hoc methods include text, visual and local explanations, feature relevance, explanations by example, and explanations by simplification. These methods are explained in more detail later.

Table 1. Explainability methods.

Group	**Explainability method**
Transparent	simulation
	decomposition
	algorithmic transparency
Post-hoc	text explanation
	visual explanation
	local explanation
	explanations by example
	explanations by simplification
	feature relevance

The differences between model-specific and model-agnostic methods are shown in Figure 3. The majority of the post-hoc methods are available for all models. Explanation by simplification, feature relevance, local explanation, and visual explanation are both model-specific and -agnostic explainability methods, whereas text explanation and explanation by example are model-specific explainability methods.

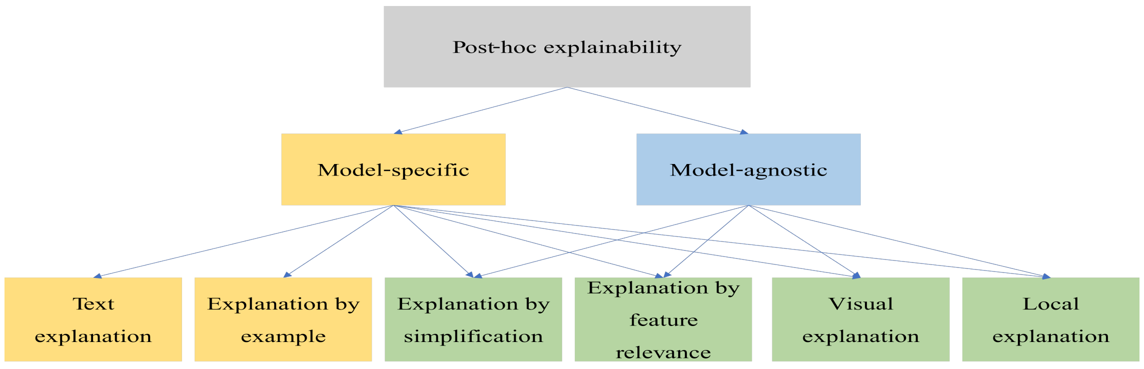


Figure 3. Taxonomy of post-hoc explainability methods. The content has been adapted from Arrieta et al. (2020) and Angelov et al. (2021).

2.4.2 Explainability of common machine learning models

Typical explainability methods used with common machine learning models are summarized in Table 2. The first column defines the model in question and the remaining columns define the typical explainability methods used. The usage of the methods is marked as yes, no, or partially. Yes means that the specific method can be at least be used, whereas partially means that the specific method can be applied for smaller or simplified model parts only. No has two meanings, either it is not needed (because other methods are sufficient) or it cannot be used (due to model complexity). Examples of applicable post-hoc methods are listed in the rightmost column.

As it is possible to see, the number of commonly used explainability methods depends heavily on the level of model complexity. The model is transparent if it is either simulatable, decomposable, or algorithmically transparent. The model is not transparent if, for example, its structure is complex. The behavior of opaque models can be investigated by using different post-hoc methods. (Arrieta et al. 2020)

Table 2. Commonly used explainability methods per model. Content adapted from Arrieta et al. (2020).

	Transparent models			**Trained models**	
Model	**Simulation**	**Decomposition**	**Algorithmic transparency**	**Post-hoc analysis**	**Examples**
Linear/Logistic Regression	yes	yes	no	no	
Decision Trees	yes	yes	yes	no	
K-Nearest Neighbors	yes	partially	no	no	
Rule Based Learners	yes	partially	no	no	
General Additive Models	partially	yes	no	no	
Bayesian Models	yes	yes	no	no	
Tree Ensembles	no	no	no	yes	Model simplification/Feature relevance
Support Vector Machines	no	no	no	yes	Model simplification/Local explanations
Multi–layer Neural Network	no	no	no	yes	Model simplification/Feature relevance/ Visualization
Convolutional Neural Network	no	no	no	yes	Feature relevance/ Visualization
Recurrent Neural Network	no	no	no	yes	Feature relevance

2.4.3 Explainability methods in general

In general, methods for transparent models include running simulations, decomposing model structures into smaller parts, and inspecting the model at the algorithmic level. Post-hoc methods include text, visual and local explanations as well as explanations by example and simplification. Indirect feature relevance analysis can also be used to explain how the model behaves. The meaning of each method is briefly explained next.

Simulatability can be understood as the possibility of explaining the model behavior step-by-step. Decomposability of a model means the possibility to easily explain what each model part does and what inputs are used. By being able to explain the roles of each model part, it can also be possible to better understand, interpret, and explain the model behavior. Algorithmic transparency can be understood as the possibility of explaining the process from model input to output by using mathematical analysis and methods. For example, a linear model is considered algorithmically transparent because its error surface is understandable and explainable and thus, it is possible to understand model actions. (Arrieta et al. 2020)

Text explanations can be used to clarify the structure and functioning of the model and explain results. Visual explanation techniques are commonly used for visualizing the model behavior in a simplified way, whereas local explanations can be used to explain model behavior by dividing the model into smaller parts and explaining these parts one by one. Explanation by example means extracting data examples that affect model output and using those to explain certain relationships and correlations within the model behavior. Furthermore, explanation by simplification means simplifying the trained model and explaining what happens in the simplified version of the model. Although the complexity is reduced, the behavior and performance are expected to be the same. Feature relevance explanations can be used to compute the relevance score of model variables. In practice, the score quantifies the sensitivity between a selected feature and model output. The most relevant features for the model output have the highest scores. (Arrieta et al. 2020)

To summarize, several explainability methods exist and the simpler the model, the easier it is to analyze the model. For example, model complexity affects simulatability greatly and thus, simulations are typically easier with simpler models. Similarly, the algorithmic visibility tends to decrease with respect to model complexity. Poor algorithmic visibility is, for example, common in deep architectures, where the model typically contains different layers and is fitted with data using optimization algorithms such as stochastic gradient descent. An efficient way to analyze a model is to calculate feature relevance, which can be used to measure how the data affects model predictions as well as rank and select features.

Practical examples of explainability methods are covered next. The main focus is on global and local model-agnostic methods because these are used in this thesis. Methods for transparent models and examples of model-specific methods are also briefly covered.

2.4.4 Methods for transparent models

Transparent models can be analyzed by running simulations and decomposing the model structure into smaller parts. In addition, if the structure of the model allows, the behavior of the model can be inspected at the algorithmic level. To gain a better understanding of how features affect predictions, other methods can be used. Examples of these are introduced next.

2.4.5 Examples of global model-agnostic methods

Global explainability methods are useful inspection techniques for every model, especially for those that are not transparent. These describe the model behavior in general and hence, these can help in understanding the model behavior and how data affects its performance. Several global model-agnostic methods exist, such as permutation feature importance (PI), partial dependence plot (PDP), accumulated local effect (ALE) plot, functional decomposition, feature interaction (H-statistic), and global surrogate models (Molnar 2022). These are covered in more detail next.

Permutation feature importance

Permutation importance (Breiman 2001) measures how much the changes in features affect model predictions, that is, how relevant each feature is for the model. The method is based on feature-specific scores that describe how much a feature replaced with noise affects the model predictions (Carta et al. 2022). The more the feature change increases the prediction error, the more important the particular feature is for the model. To calculate permutation feature importance, each feature is randomly shuffled one by one and at each update, prediction accuracy is calculated again and feature-specific importance is calculated using original and new prediction accuracy according to Equation 5 (Scikit-learn 2022d)

$$i_j = s - \frac{1}{K}\sum_{k=1}^{K} s_{kj}, \tag{5}$$

where j means feature index, k means repetition index, K means the number of features and thus repetitions, s means the reference prediction accuracy, and s_{jk} the feature change -specific prediction accuracy. Both training and testing data can be used, but using testing data has been recommended because the results can be too optimistic with data that the model has seen before and this can lead to misunderstanding on what features are the most important for the model (Molnar 2022).

Partial dependence plot

Partial dependence plot (Friedman 2001) shows how features affect the predictions on average and what the relationship between the feature and predictions is. By visualizing predictions versus a specific feature, PDPs are intuitive and easily interpretable. The partial dependence function for regression $\hat{f}_s$ is shown in Equation 6 (Molnar 2022)

$$\hat{f}_S(x_s) = E_{x_c}\left[\hat{f}(x_s, x_c)\right] = \int \hat{f}(x_s, x_c)\,d\mathbb{P}(x_c), \tag{6}$$

where x_s contains features under investigation, x_c contains the remaining features, and $\hat{f}$ represents the model. PDP function can be estimated using Equation 7 (Molnar 2022)

$$\hat{f}_S(x_s) = \frac{1}{N}\sum_{i=1}^{N} \hat{f}(x_s, x_c^{(i)}), \tag{7}$$

where N is the number of samples and $x_c^{(i)}$ represents the ith value in features from the set c. In practice, $\hat{f}_S(x_s)$ shows the average marginal effect of features from the set s on predictions. The calculation of PDP assumes that the features between sets s and c are not correlated (Molnar 2022). Consequently, in case the features between different sets are correlated, the averages can be irrational.

Accumulated local effect plot

Accumulated local effect plot (Apley & Zhu 2020), similar to PDP, can be used to visualize how features affect predictions on average. ALE is an alternative to PDP and it is claimed to work better with correlated features than PDP and be less computationally expensive than PDP (Apley & Zhu 2020). Molnar (2022) has gathered a comprehensive overview of the differences between PDP and ALE and related pros and cons. Also, in this work, using ALE instead of PDP is recommended because features are typically one way or another correlated.

Functional decomposition

Functional decomposition means simplifying something complex to ease interpretation. In practice, functional decomposition could mean simplifying a complex model by dividing it into smaller and more interpretable parts that produce the model output. An example of decomposing function $y = \hat{f}(x_1, x_2) = 2 + e^{x_1} - x_2 + x_1 \cdot x_2$ is shown in Equation 8 (Molnar 2022)

$$\hat{f}(x_1, x_2) = \hat{f}_0 + \hat{f}_1(x_1) + \hat{f}_2(x_2) + \hat{f}_{1,2}(x_1, x_2), \tag{8}$$

where $\hat{f}_1$ and $\hat{f}_2$ represent the main effect of features x_1 and x_2, respectively, $\hat{f}_{1,2}$ is the interaction between two features and $\hat{f}_0$ is the interception of the two features. Function decomposition is based on ALE (Molnar 2022).

Feature interaction

Feature interaction, or H-statistic, (Friedman & Popescu 2008) can be used to measure how much the interaction of the features affects the variation of the prediction. Interaction can be investigated from two perspectives: 1) are two features of the model interacting and if yes, how much; 2) is a feature interacting with the model and if yes, how much (Molnar 2022).

Interaction strength can be understood as the amount of variance explained by the interaction. Value 0 means that there is no interaction between the selected parts, whereas 1 means that all of the feature- or model-specific variance can be explained by the sum of partial dependence functions. Values greater than 1 can also exist if the 2-way interaction variance is high enough. Partial dependence plots can be then used to visualize what the selected interactions look like. (Molnar 2022)

For more details about the calculation of H-statistic, please for example see Friedman & Popescu (2008) and (Molnar 2022).

Global surrogate models

Global surrogate models are used to explain predictions of a complex, opaque model. In practice, predictions of the complex model are used to train an explainable model, and the resulting trained model can then be further investigated. The structure of the surrogate model, such as the decision tree, can be then visualized to show how the predictions are made. R^2 value, or the coefficient of determination (Equation 18), can be used to measure how well the surrogate model replicates the original model (Molnar 2022)

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}, \tag{9}$$

where N is the number of samples, y_i is the prediction of the original model at index i, $\hat{y}_i$ is the prediction of the surrogate model at index i, and $\bar{y}$ is the mean of the original model predictions. The closer R^2 value is to 1, the better the surrogate model approximates the behavior of the original model (Molnar 2022).

2.4.6 Examples of local model-agnostic methods

Local explainability methods can be used to explain selected predictions. By utilizing both global and local explainability methods, it is possible to get more a comprehensive overview of how the model behaves. Several local model-agnostic methods exist, such as individual conditional expectation (ICE) plot, local surrogate models, scoped rules (anchors), counterfactual explanations, Shapley values, and Shapley additive explanations (SHAP) (Molnar 2022).

Individual conditional expectation plot

Individual conditional expectation plot (Goldstein et al. 2015) visualizes how the feature changes affect predictions. ICE plot contains one line for each prediction, whereas PDP contains one line that shows how features affect predictions on average. Lines in ICE plot can be calculated by modifying only the values of the feature under investigation and taking predictions with the updated feature values (Molnar 2022). By visualizing the change in predictions, ICE can help in identifying interactions in the model and extrapolating in the predictor space (Goldstein et al. 2015). According to Molnar (2022), ICE plots are more intuitive than PDPs. However, the correlation between features can similarly distort visualizations as PDPs.

Local surrogate models

One approach for local surrogate models is a method called local interpretable model-agnostic explanation (LIME) (Ribeiro et al. 2016). LIME trains local interpretable models, such as linear regression, for a specific prediction by using features and responses from the region of interest (Molnar 2022).

In practice, creating a local surrogate model is about minimizing the loss $L(f, g, \pi_x)$ so that the level of surrogate model complexity $\Omega(g)$ is sufficiently low to ensure both local fidelity and interpretability (Ribeiro et al. 2016). Equation 10 shows the mathematical form of local surrogate models (Ribeiro et al. 2016, Molnar 2022)

$$\text{explanation}(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g), \tag{10}$$

where x is the prediction to be explained, L is the training loss between the prediction of the original model f and the prediction of the surrogate model g, $\Omega(g)$ is the model complexity (such as for linear models, the number of non-zero weights and for decision trees, the depth of the tree), π_x is the proximity measure to define locality around x, and G is the number of possible surrogate models (such as linear regression and decision trees).

LIME is as a useful tool for both expert and non-expert users to get a better understanding of and build more trust in the model predictions (Ribeiro et al. 2016). However, according to Molnar (2022), LIME has many disadvantages relating to the trustworthiness of the explanations and consequently, it should be carefully used and explanations should be critically analyzed.

Scoped rules

Scoped rules, or anchors, (Ribeiro et al. 2018) can be used to clarify how specific prediction is made. The researchers behind LIME developed anchors and consequently, these methods are somewhat similar. With LIME, predictions are explained using a local surrogate model, whereas with anchors, the predictions are explained with rules (Molnar 2022). In practice, the aim of the anchor method is to find specific if-then rules that sufficiently anchor the prediction locally (Ribeiro et al. 2018). More details about the equations used to generate anchors for specific instances are available in Ribeiro et al. (2018) and Molnar (2022).

The anchors are easy to interpret and they work even if model predictions are not linear. However, the method is highly configurable and thus, there are several parameters to be adjusted to get useful explanations. Sometimes, the explanations can be too specific and hence, it can be challenging to understand the model behavior without refining the results. (Molnar 2022)

Counterfactual explanations

Counterfactual explanations (Wachter et al. 2017, Dandl et al. 2020) are example-based explanations for individual predictions.

Counterfactual explanations summarize what factors and changes result in the specific prediction. Explanations can be generated via manual search or by utilizing a loss function that takes a specific feature as input, a counterfactual, and a feature-specific response. The counterfactual explanation is the one that minimizes the loss. Several benefits exist, such as ease of implementation and interpretation. However, usually, multiple explanations are found and thus, it can be challenging to select what explanations are the most suitable ones. (Molnar 2022)

Shapley values

Shapley values (Shapley 1953) were initially used in coalitional game theory to define player-specific payouts depending on the player's contribution to the payout.

In explainability analysis, the aim is to define how much each feature affects the prediction compared to the average prediction. Therefore, the prediction task is the game, features are players, the gain is the true prediction of a specific instance reduced by the average predictions, and Shapley value is the average effect of a feature on the prediction. The average effect is calculated from the differences between predictions with and without the feature under investigation. The number of comparisons is equivalent to the number of feature subsets (coalitions). To get model predictions, features not included in the subsets are replaced with random feature values. (Molnar 2022)

Shapley value calculation of a specific feature is shown in Equation 11 (Molnar 2022)

$$\phi_j(x) = \frac{1}{M} \sum_{m=1}^{M} \phi_j^m, \tag{11}$$

where $\phi_j(x)$ is the feature-specific Shapley value at instance x, M is the number of iterations, and m is the specific iteration (or a subset of features).

In comparison to LIME, Shapley values is an explainability method with solid theory. Also, the prediction is fairly distributed among the features with Shapely value, whereas with LIME it is not guaranteed. However, Shapley value calculation can be heavy - especially if there are several features. (Molnar 2022)

Shapley additive explanations

Shapley additive explanations (Lundberg & Lee 2017), based on Shapley values, can also be used to define a prediction-specific importance value for each feature. According to the developers, SHAP was developed by unifying existing methods to get desired properties and bringing clarity to the existing explainability methods.

SHAP provides tools for both global and local explanations. For example, KernelSHAP is a kernel-based model-agnostic approach inspired by LIME for estimating Shapley values. Model-specific explainability method for tree-based models, treeSHAP, exists, too. Global feature importances, similar to permutation feature importances, can also be calculated with SHAP. However, PI and SHAP are calculated differently and hence, the feature importances cannot be compared. (Molnar 2022)

SHAP explanation is defined in Equation 12 (Molnar 2022)

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j, \tag{12}$$

where g is the explanation model, z is the coalition vector (binary vector that defines what features are in use in the particular subset or coalition and what features are absent that should be resampled with random data samples), M is the maximum coalition size, and ϕ_j is the feature-specific Shapley value.

SHAP feature importance can be calculated by using Equation 13 (Molnar 2022)

$$I_j = \frac{1}{N} \sum_{i=1}^{N} |\phi_j^{(i)}|, \tag{13}$$

where N is the number of features and $\phi_j^{(i)}$ is the feature-specific Shapley value.

Different visualizations can be created by using SHAP. For example, feature importances can be normally visualized in a bar plot. The positive and negative effects of features can also be shown in a force plot with arrows. Other visualizations, such as summary and dependence plots can also be used. These visualize feature values and corresponding SHAP values so that the summary plot contains many features, whereas the dependence plot visualizes data only from a selected feature.

According to Molnar (2022), several benefits and drawbacks are related to SHAP. For example, the computation of global explanations is fast. Also, SHAP relates to LIME and Shapley values making it easier to understand its basic principles and how to use it. In addition, most likely due to the similarity with other explainability methods, SHAP has become popular in the field of XAI. However, as mentioned by Molnar, it is possible to create misleading interpretations from SHAP results. Also, kernelSHAP calculation can be slow if many instances are included in Shapley value calculation. Similarly, in case several Shapley values are calculated, the calculation of SHAP feature importances can also be slow. Furthermore, because SHAP is based on Shapley values, it has the same benefits and drawbacks as Shapley values. Despite some drawbacks, SHAP and LIME are currently the primary model-agnostic tools used (Freeborough & van Zyl 2022).

2.4.7 Examples of model-specific methods

To analyze and compare models developed systematically, this thesis focuses on model-agnostic methods available. Hence, model-specific methods are covered only briefly.

Model-specific explainability methods are typically used to provide more details about the model behavior and to improve the level of explainability of the model. Model-specific explanations can also be used to support the findings from other explainability methods. Several model-specific explainability methods exist. For example, methods for tree ensembles such as mean decrease accuracy (MDA) can be used to describe feature relevance, whereas simplified tree ensemble learner (STEL) can be used to explain models by simplification (Arrieta et al. 2020). Derivatives of model-agnostic methods also exist, such as model-specific counterfactual explanations and treeSHAP for tree-based models (Molnar 2022).

Summaries of model-specific explainability methods are, for example, available in the articles by Arrieta et al. (2020) and Rojat et al. (2021). Methods specific to deep learning models are for example covered in the works by Rojat et al. (2021) and Molnar (2022).

2.5 Index investing

In this thesis, financial data from S&P 500 index is used for developing time series models and investigating different explainability methods. General information about the index is covered next, whereas the data content used for developing models is covered in more detail in Section 3.

2.5.1 Pros and cons

The idea of index investing is well summarized by Chen (2022). It is a passive investing method that aims to replicate the performance of a benchmark index. Index investing is based on the expectation that the market will outperform any stock picker in the long run. Depending on the investment strategy, index investing is carried out by purchasing components according to the index. To invest according to the selected index cost-efficiently, it is also possible to, for example, purchase only the most weighted components or a specific ratio of components included in the index.

Index investing contains several benefits. For example, because index investing is based on an example index, it does not require active management of investment strategies. Consequently, it commonly requires less frequent trades reducing costs. In addition, typically index investing utilizes several components from a broad region and thus, the investments are more diversified and protected against risks than single stocks. Hence, index investing is a cost-efficient approach for gaining rather stable returns. Collecting all index-specific components at specific weights can be, however, time-consuming and costly. Also, the index can be greatly affected by changes in stock prices of highly weighted components.

2.5.2 S&P 500 index

S&P 500 index stands for standard and poor's. It was created in 1957 and it monitors the performance of 500 publicly-traded large companies that are listed on the stock in the United States. The index was the first market-capitalization-weighted stock market index and it has become a well-known financial benchmark worldwide. (Indices 2022b)

The equation used for calculating S&P 500 index is shown in Equation 14

$$S\&P500 = \frac{\sum_{i=1}^{N} P_i * Q_i}{Divisor}, \tag{14}$$

where P_i stands for the price of a company-specific stock, Q_i stands for the number of publicly available shares, and *divisor* is a normalization factor.

In practice, the index value is calculated by dividing the sum of the market capitalization of companies included in the index by the index divisor. The divisor is an arbitrary number and by adjusting the divisor value, it is possible to maintain the continuity of the index despite changes in index components. The divisor is updated after the close of trading. More details about the divisor adjustment can be found on S&P 500 homepage. (Indices 2022a)

Market capitalization is calculated by multiplying the current stock price by the outstanding shares of a company. The index weight per company is, in turn, done by dividing the market capitalization of a company by the total market capitalization of the index. (Kenton 2022)

Due to the different share classes of certain companies, such as Alphabet's Class A (GOOGL) and Class C (GOOG), the index consists of more than 500 stocks. In November 2022, the top three companies, according to the index weight, were Apple, Microsoft, and Amazon.com. Also, the top three represented sectors were information technology, health care, and financials. (Indices 2022b)

Unlike stocks, it is not possible to invest directly in the S&P 500 index. Instead, it is, for example, possible to invest in a fund that uses the index as a benchmark and that tracks its performance and content (Kenton 2022). At the end of the year 2020, more than $5.4 trillion was invested in the performance of this index (Indices 2022b). Furthermore, the performance of S&P 500 index is commonly used to indicate the health of current and future U.S. markets (Chen 2022).

2.6 Earlier work

In this thesis, the aim is to develop models for forecasting time series data and to investigate selected methods for analyzing model explainability. Consequently, a brief literature search was done to find earlier studies that combine financial data forecasting and model explainability analysis. Summaries of forecasting financial data using machine learning were easily found, but the number of studies reporting the usage of XAI alongside financial prediction model development was rather limited and they were typically published in recent years. Findings are briefly covered next.

2.6.1 Financial time series forecasting models

Krollner et al. (2010) did a literature search to summarize what approaches have been used to develop machine learning models to predict stock index. The reason for the study was two-sided. Firstly, stock index prediction is a challenge in financial time series forecasting, thanks to the large price volatility in the stock market. Secondly, the aim of the study was to help other researchers to better know what has been done before in the field and what potential future studies could be. The following information was collected: machine learning method used, forecasting time-frame, what features were used, and how the models were evaluated. Based on the results, a trend of using artificial neural networks (ANN) with new training algorithms or combining those with new machine learning methods was seen. Also, the most common timeframe was one day ahead. However, it was expected that one-day ahead predictions would not bring much value for an investor. Most commonly, features contained lagging and the values were derived from established technical indicators. As an improvement opportunity, in addition to developing accurate time series models, future studies could examine how the models developed could improve the risk-return tradeoff of an investor in a real-life scenario.

A newer literature search was done by Sezer et al. (2020). In this study, the extensive search focused on studies that used deep learning for financial time series forecasting and that were published in the years 2005–2019. Based on the findings, the introduction of deep learning for financial time series forecasting gave a new boost to the field studied for more than 40 years. According to Sezer et al. (2020), RNN-based models, especially LSTM, are the most commonly used models for financial time series forecasting. However, although DL models were commonly reported as better than traditional machine learning models, many studies also noted similar performances. Improved development environments, computing power, data availability, better performance, and feature learning possibilities have contributed to DL models becoming more common.

Fischer & Krauss (2018) focused on developing LSTM networks to a S&P 500-related prediction task. In addition to developing a highly accurate model, the study aimed to provide an in-depth guide to the model development, feature preparation, and utilization of the model predictions in the form of a trading strategy. LSTM was found to be superior to random forest and logistic regression both from the prediction accuracy and daily returns after transaction costs point-of-view.

2.6.2 Explainability of financial time series forecasting models

In work by Carta et al. (2022), permutation-based feature relevance analysis was utilized in automatic feature selection to gather data for models that predict next-day returns of selected stocks. This approach was found useful both in selecting important features as well as improving both prediction accuracy and model interpretability. The usage of random forest model was explained by its explainability and because it has been found to work well with financial prediction tasks. The capability of finding irrelevant features of the method developed in the study was also compared to the state-of-the-art approaches, including random forest feature importance and local interpretable models. The new method introduced was found to be superior to the state-of-the-art approaches. As stated by the authors, explainability is generally speaking challenging and it is especially challenging with financial data due to the typical low signal-to-noise ratio and hard interpretability of trends. Utilizing the method for automatic feature selection could help in reaching more robust and reliable explainable models.

Freeborough & van Zyl (2022) investigated the transferability of existing XAI methods to financial time series prediction. According to the authors, although explainable systems are becoming more desired, only a little development has been seen in the utilization of XAI with financial data. This was claimed to be apparent due to the limited number of XAI methods published for regression problems. In the study, deep learning models RNN, LSTM, and GRU were trained to investigate explainability with S&P 500 stocks data. The explainability analysis focused on investigating the importance of features, single data points, and different model parts. Four XAI methods were used: ablation, permutation, added noise, and integrated gradients. Based on the results of the explainability analysis, GRU was seen to retain long-term information the best and RNN the worst. In addition, although XAI methods used were found usable for financial time series models and robust in clarifying how each model ends up to the prediction, the need for less abstract, more accurate, and more context-specific XAI methods to describe model behavior in a practical way would be needed.

Ilic et al. (2021) developed an explainable boosted linear regression (EBLR) and investigated its usage for time series forecasting in comparison to other established models such as random forest and gradient boosting regression. EBLR is based on iterative model updates that utilize the error between the true and predicted value, decision tree, and baseline linear model. Explainability analysis focused on feature importance. Three different datasets were used in the study: synthetic dataset that mimics the sales of a store, true sales dataset of stores, and electricity consumption dataset. Based on the results, EBLR was found to perform similarly in comparison to other approaches. Due to the interpretability of the model predictions and prediction accuracy, EBLR was claimed as a promising alternative method for time series forecasting.

3 MATERIALS

3.1 Raw data

S&P 500 stocks data with Creative Commons License 1.0 (no copyrights) from Kaggle (Larxel 2022) was used in this thesis. This data is updated every day, and at the time of writing this thesis, the data contained daily stock data from 2010 to 2022 and index data from 2012 to 2022. The raw data contained daily S&P 500 index value and company-specific stock data.

The company data included trading day-specific close, adjusted close, high, low, open, and volume information for each company included in the index. Close means the price of the last stock traded at the end of a trading period, whereas open means the price of the stock at the beginning of a trading period. Adjusted close is similar to close, but it takes into account company actions such as splits and dividends (Balasubramaniam 2022). High and low mean a stock's maximum and minimum prices during a trading period, respectively. Lastly, volume defines the number of shares traded.

Company details, such as sector and industry classification, market capitalization, revenue, and the number of full-time employees, were also available in a separate file. Because the index includes multiple classes of stock of some constituent companies, such as class A (GOOGL) and class C (GOOG), the current index consists of data from 503 stocks in total.

The closing prices from 2010 to 2022 of S&P 500 companies with the index from 2012 to 2022 are shown in Figure 4. Closing prices are shown as gray lines, whereas the index is shown as a black line. Although the trends contain occasional drops from time to time, both closing prices and index have increased from 2012 to 2022. The fall of 2020 occurred around the same time coronavirus became a pandemic. Luckily, as it is possible to see from the trends, the companies have recovered from the pandemic well. Now in 2022, the trends have started to decrease again, though.

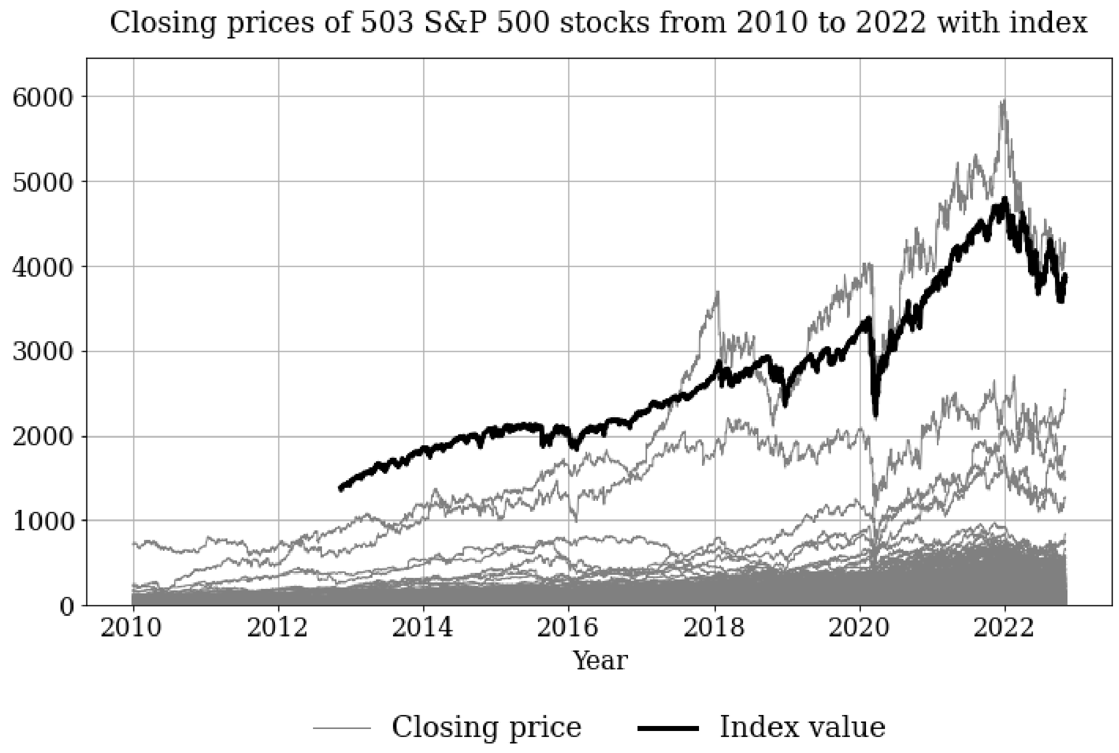


Figure 4. Closing prices and S&P 500 index.

3.2 Data content

Data from companies included in the S&P 500 index is summarized in Figures 5–8. These summarize sectors, continents, number of full-time employees, and market capitalization of the companies.

As shown in Figure 5, there are altogether 11 different sectors represented, and the top three sectors are technology (14.2%), industrials (14.6%), and financial services (13.8%). Based on Figure 6, the majority of the companies included in the index come from North America (96.4%), and others come from Europe (3.4%) and Asia (0.2%). Most commonly, there are ≥25 thousand full-time employees (Figure 7). From Figure 8 (market capitalization versus the number of full-time employees), it is possible to see that companies included in the index are similar from this perspective. Only a few companies have more employees and/or higher market capitalization than the majority. Variation of market capitalization value between the companies could be utilized, though, by, for example, grouping companies per market capitalization value and using this group info as part of the data fed to the model.

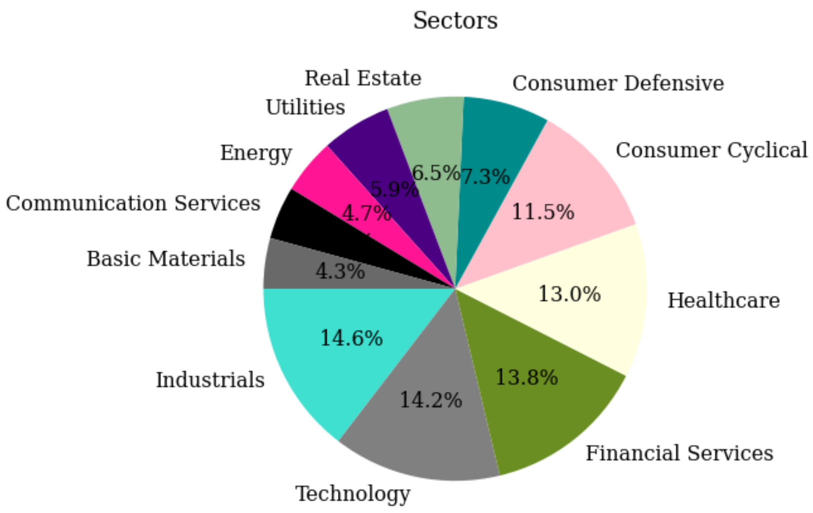


Figure 5. Sectors represented in S&P 500 companies.

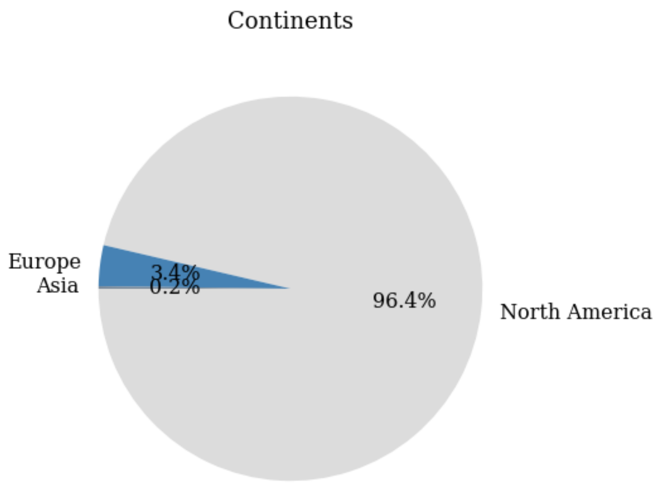


Figure 6. Continents represented in S&P 500 companies.

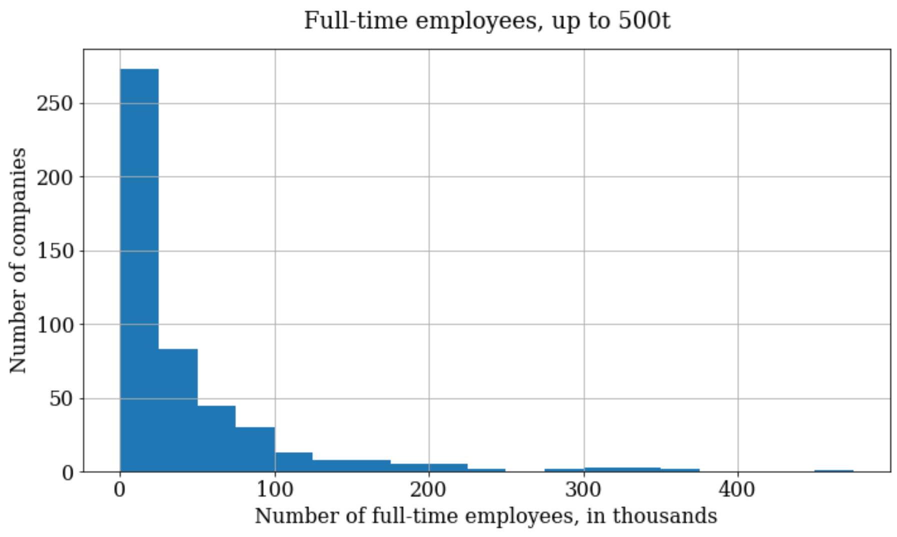


Figure 7. Number of full-time employees in S&P 500 companies.

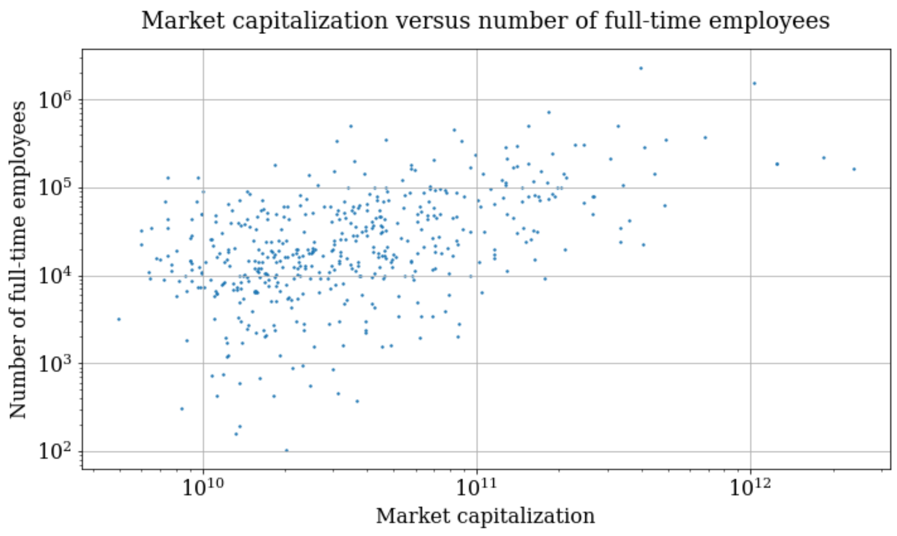


Figure 8. Market capitalization with the number of full-time employees.

3.3 Data sets

Raw index data was limited up to the end of October 2022. The data was divided into separate data sets for training and testing purposes with a 70:30 ratio. Data preparation for model training is described in more detail in Section 4.

The division between training and testing data is shown in Figure 9. Note that although this visualization contains both closing prices and S&P 500 index, the development data is based on S&P index only. To separate trends more efficiently, percentual change in price or logarithmic scaling in the y-axis could have been used instead. However, the purpose of this visualization is to show how the stocks in general have affected the index value, how the index has evolved through the years, and what the raw index trend in training and test set looks like. As it is possible to see, few stocks have affected the index greatly.

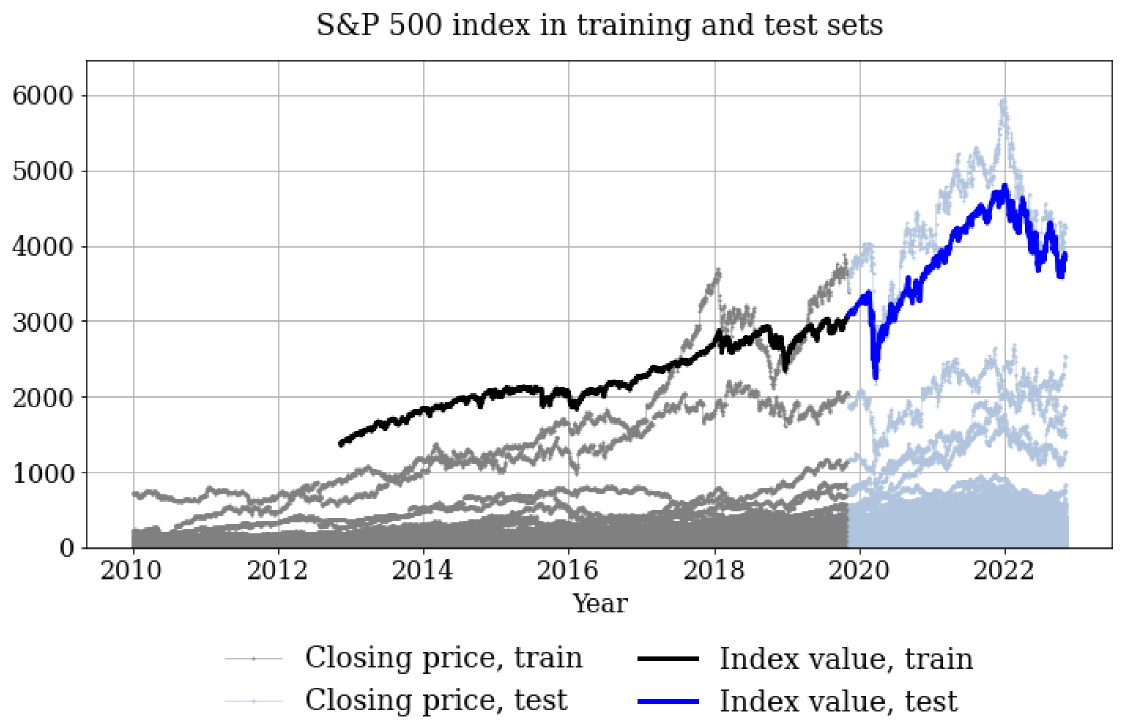


Figure 9. Closing prices and S&P 500 index with training and test sets.

4 METHODS

4.1 Prediction models

Different time series models were developed with Python for predicting S&P 500 index at each timestep. To find out what the baseline prediction accuracy is for S&P 500 index forecasting, models were kept simple. The following five relatively simple models were trained using sklearn library: linear regression, decision tree, K-nearest neighbors, random forest, and gradient boost. Deep learning models, such as LSTM, were also considered, but these were left out of the scope because five models are already sufficient for comparing models and testing explainability methods.

4.2 Feature and response selection

4.2.1 Feature-response pairs

Different feature-response pairs were investigated and the most suitable feature-response pair was selected for final model development. Feature-response pairs that were under consideration are listed in Table 3. For simplicity, feature IDs are used later in this thesis to refer to the corresponding features.

To find out what the baseline prediction accuracy is for S&P 500 index forecasting and to minimize data preparation work, features and responses were derived from S&P index data. The first feature-response pair consisted of S&P 500 index data only and the second pair included also additional derivatives of S&P 500 index. Features included moving S&P 500 index mean of the previous 5, 30, and 365 days; moving S&P 500 index standard deviation of the previous 5, 30, and 365 days; ratios between moving 5- and 365-day means; and ratios between moving 5- and 365-day standard deviations. S&P 500 index was used as response. The third pair was similar to the second one, but all values were ratios. Features included ratios between S&500 index and moving S&P 500 index mean of the previous 5, 30, and 365 days; ratios between S&500 index and moving S&P 500 index standard deviation of the previous 5, 30, and 365 days; ratios between moving 5- and 365-day means; and ratios between moving 5- and 365-day standard deviations. The response was in the format of S&P 500 index(t)/S&P 500 index(t-1), where t is time.

Table 3. Feature-response pairs.

Pair	Feature ID	Features	Responses
1	NA	S&P 500 index(t-1)	S&P 500 index(t)
2	F1	5-day average of S&P 500 index	S&P 500 index(t)
	F2	30-day average of S&P 500 index	
	F3	365-day average of S&P 500 index	
	F4	$\frac{\text{5-day average of S\&P 500 index}}{\text{365-day average of S\&P 500 index}}$	
	F5	5-day standard deviation of S&P 500 index	
	F6	30-day standard deviation of S&P 500 index	
	F7	365-day standard deviation of S&P 500 index	
	F8	$\frac{\text{5-day standard deviation of S\&P 500 index}}{\text{365-day standard deviation of S\&P 500 index}}$	
3	F1	$\frac{\text{S\&P 500 index}}{\text{5-day average of S\&P 500 index}}$	$\frac{\text{S\&P 500 index(t)}}{\text{S\&P 500 index(t-1)}}$
	F2	$\frac{\text{S\&P 500 index}}{\text{30-day average of S\&P 500 index}}$	
	F3	$\frac{\text{S\&P 500 index}}{\text{365-day average of S\&P 500 index}}$	
	F4	$\frac{\text{5-day average of S\&P 500 index}}{\text{365-day average of S\&P 500 index}}$	
	F5	$\frac{\text{S\&P 500 index}}{\text{5-day standard deviation of S\&P 500 index}}$	
	F6	$\frac{\text{S\&P 500 index}}{\text{30-day standard deviation of S\&P 500 index}}$	
	F7	$\frac{\text{S\&P 500 index}}{\text{365-day standard deviation of S\&P 500 index}}$	
	F8	$\frac{\text{5-day standard deviation of S\&P 500 index}}{\text{365-day standard deviation of S\&P 500 index}}$	

4.2.2 Best feature-response pair

When the first feature-response pair was used to train and test models, the models were overfitted with training data and performed poorly with test data. With the second feature-response pair, the models performed slightly better than with the first pair. However, as expected, models performed well only roughly up to the maximum value of index value in training data. The results with the third feature-response pair were the best, and no maximum value limitations were seen. Clearly, using features and responses with a more controlled value range improved the model performance.

Features and responses of the third pair were selected for model development and explainability analysis. The correlation between features and responses was also analyzed to get more insight into how features affect predictions and how features affect each other.

4.3 Data processing

Because of the moving window up to 365-days, the first 365 days of training and test data were excluded from features and responses. This way, it was possible to mitigate including missing values in training and testing data. The excluded parts are shown in Figure 10.

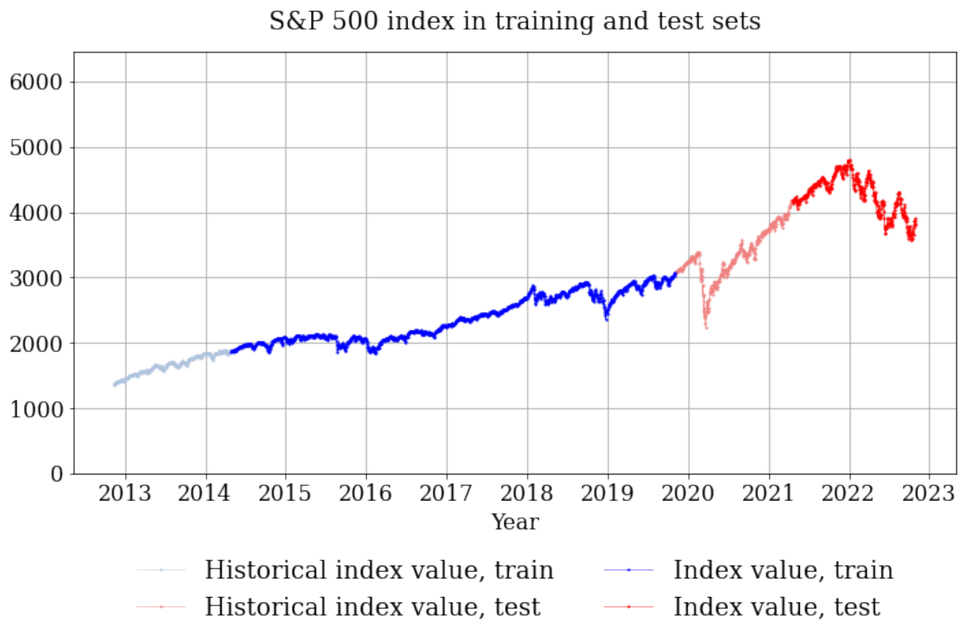


Figure 10. S&P 500 index with training and test sets and regions excluded due to moving window.

In addition to excluding missing values, data was normalized to value region 0–1 by using minMaxScaler from sklearn library. The scaler was included in the model pipeline, and thus both the scaler and model were trained using training data only. Normalization to value region 0–1 is shown Equation 15

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}, \tag{15}$$

where X is the value in the dataset, X$_{min}$ is the smallest value in the dataset, X$_{max}$ is the highest value in the dataset, and X' is the normalized value in the dataset.

To predict index values at time t, the ratios predicted by the model at time t were multiplied by the earlier value of the S&P 500 index(t-1).

4.3.1 Model parameter optimization

Selected model parameters were optimized by using randomizedGridSearchCV with 5-fold cross-validation (CV) from sklearn library. randomizedGridSearchCV was used instead of exhaustive gridSearchCV in order to expedite the optimization process. In CV 5-fold, training data is split into five subsets so that one subset is used for testing the model and the rest is used for training the model. Then, the sets are updated to train and test the model again. This process is repeated five times so that every subset has been used once for testing. Cross-validation is a useful and robust approach for testing the model performance. The optimized models were tested with separate test data.

4.4 Prediction accuracy

Common metrics were used to evaluate the prediction accuracy of models developed, both from the training and testing performance point-of-view. Prediction results with training and test data were also compared to see how well the model has learned from the training data. The functions used to calculate metrics for comparing models were mean absolute error (MAE, Equation 16)

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|, \tag{16}$$

root mean square error (RMSE, Equation 17)

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}, \tag{17}$$

and coefficient of determination (R^2, Equation 18)

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}, \tag{18}$$

where N is the number of samples, y_i is the true value at index i, $\hat{y}_i$ is the predicted value at index i, and $\bar{y}$ is the mean of values.

MAE and RMSE describe the overall prediction accuracy from two perspectives. MAE describes the general error level, whereas RMSE highlights the model performance at individual predictions by weighting high errors more than small ones. R^2 represents the goodness of the fit of the model. In practice, it describes how well previously unseen data will be predicted by the model based on the proportion of explained variance. The perfect fit between the true and predicted values is indicated by a value of 1.

4.5 Explainability methods

To systematically investigate and compare models developed and explainability methods used, selected model-agnostic methods were preferred. Methods to be used and analyzed are presented next.

In the study of Carta et al. (2022), permutation feature importance and LIME were used as standard explainability methods for a random forest time series model. Because financial data was used in the study, these methods were also used in this thesis. SHAP was also used to analyze models because of its popularity.

As introduced in Section 2, model behavior can be investigated both globally and locally. Permutation feature importance can be used to calculate global, whereas LIME can be used to calculate local feature importance values. SHAP can be used to investigate both global and local behavior. Hence, in addition to defining feature importance both at a global and local level, SHAP can be used to investigate the global and local behavior of the best model. Permutation importance function can be loaded from the sklearn library, LIME from LIME library, and SHAP from SHAP library.

To better understand how the model works and be able to investigate the logic behind the model, structure visualizations can also be used to explain the model behavior. According to Table 2, all three methods for transparent models; simulations, decompositions, and inspection at the algorithmic level; are available for decision trees. However, tree ensembles are not transparent and thus, simulatability and decomposability apply for linear regression and K-nearest neighbors only.

5 RESULTS

5.1 Correlation between selected features and responses

Correlation analysis was run with training data to better understand how features affect model responses. The correlation values are shown in Figure 11. As it is possible to see, small positive and negative correlations between S&P 500 index ratio and features exist, whereas the correlation is stronger between the features. The highest positive correlations relate to feature pairs F4-F3, F2-F1, and F3-F2, and the highest negative correlations relate to feature pairs F7-F4, F7-F3, and F8-F3.

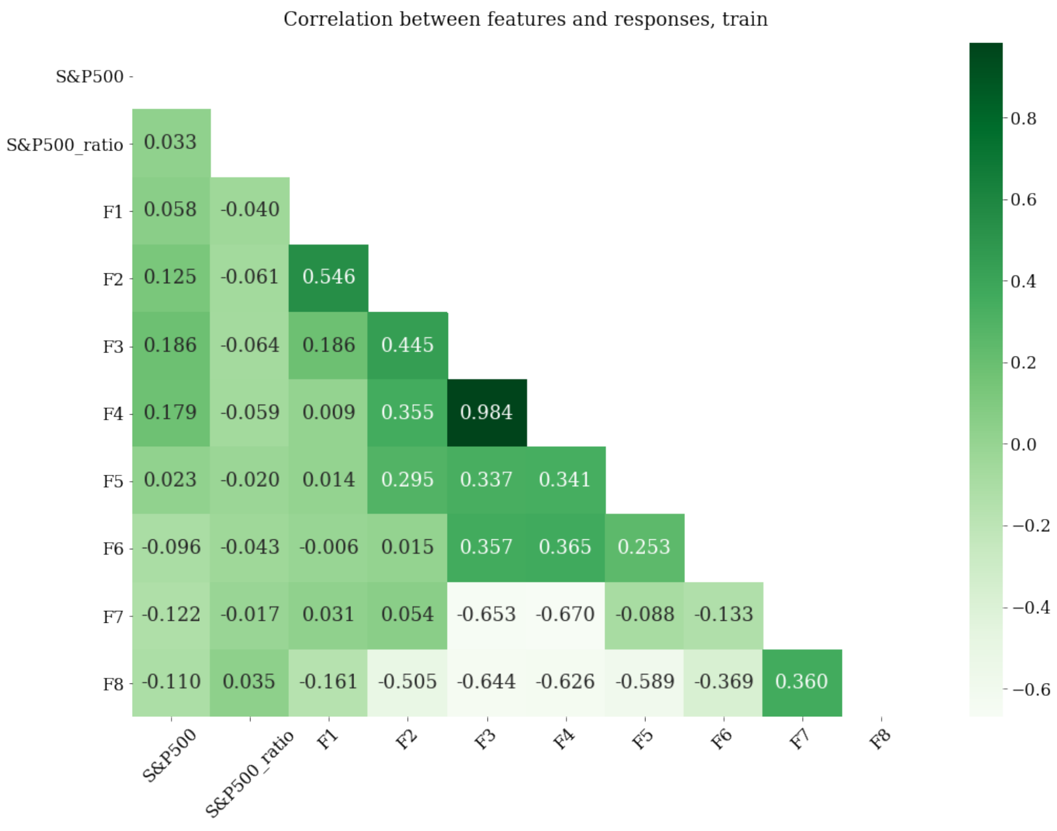


Figure 11. Correlation between features and responses of pair 3, training data.

5.2 Optimized parameters

The selected parameters and optimization results are shown in Table 4.

Table 4. Model parameter optimization results.

	Model	Parameters to optimize	Best value
1	Linear regression	normalize: False, True	False
2	Decision tree	max_depth: 10, 20, 50, 100	50
		max_leaf_nodes: 10, 20, 50, 100	50
		min_samples_leaf: 10, 20, 50, 100	100
		min_samples_split: 10, 20, 50, 100	50
3	KNN	n_neighbors: 5, 10, 20, 50	50
		weights: uniform, distance	uniform
		algorithm: auto, ball_tree, kd_tree, brute	kd_tree
4	Random forest	n_estimators: 10, 20, 50, 100, 150	50
		max_depth: 10, 20, 50, 100	100
		max_leaf_nodes: 10, 20, 50, 100	20
		min_samples_leaf: 10, 20, 50, 100	100
		min_samples_split: 10, 20, 50, 100	10
5	Gradient boost	n_estimators: 10, 20, 50, 100, 150	20
		learning_rate: 0.1, 0.01, 0.001, 0.0001	0.0001
		criterion: friedman_mse, squared_error, mse	friedman_mse
		max_depth: 10, 20, 50, 100	100
		max_leaf_nodes: 10, 20, 50, 100	50
		min_samples_leaf: 10, 20, 50, 100	100
		min_samples_split: 10, 20, 50, 100	50

5.3 Prediction accuracy

Training and test results of optimized models are summarized in Table 5. Model-specific results are also visualized in Figures 12–16. Prediction accuracy metrics include MAE, RMSE, and R^2 score. Clearly, training MAE and RMSE are smaller than test MAE and RMSE indicating that models overfit with training data. R^2 score varies only slightly, and it is above 0.9 with all models. The best performance was obtained with model 5, gradient boost. Next explainability of all models is investigated. The best model is used to visualize model-specific explanations as an example.

Table 5. Training and test results.

		Train			Test		
	Model	MAE	RMSE	R^2	MAE	RMSE	R^2
1	Linear regression	13.528	19.815	0.997	40.996	53.479	0.967
2	Decision tree	13.423	19.564	0.997	39.486	52.407	0.968
3	KNN	13.478	19.758	0.997	39.518	51.799	0.969
4	Random forest	13.325	19.594	0.997	39.306	52.027	0.968
5	Gradient boost	13.579	19.917	0.997	39.059	51.722	0.969

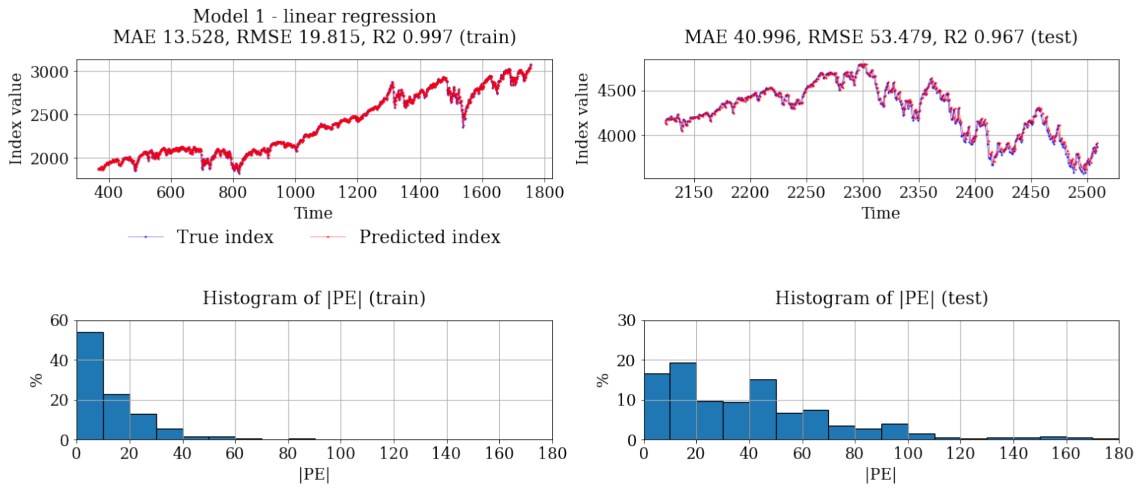


Figure 12. Predictions with model 1.

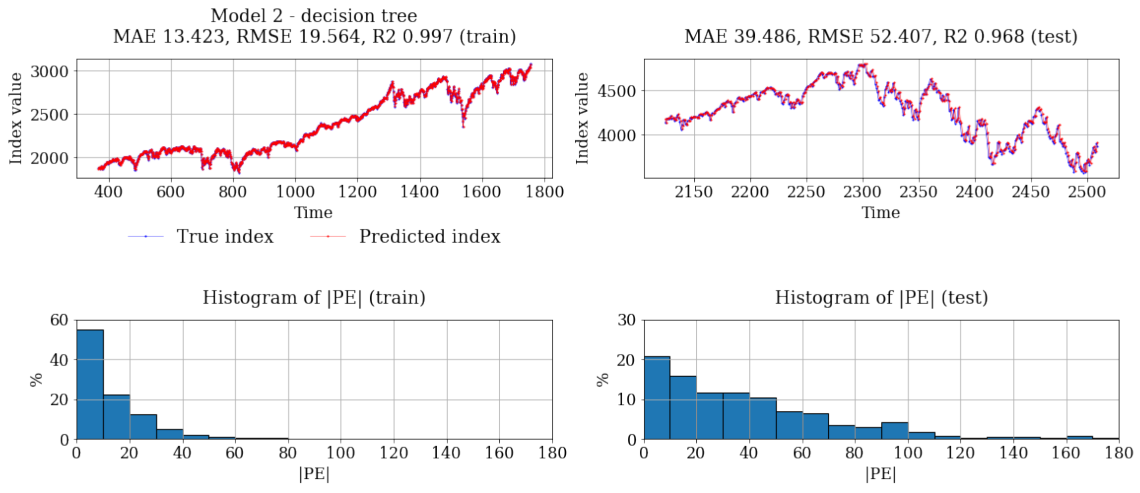


Figure 13. Predictions with model 2.

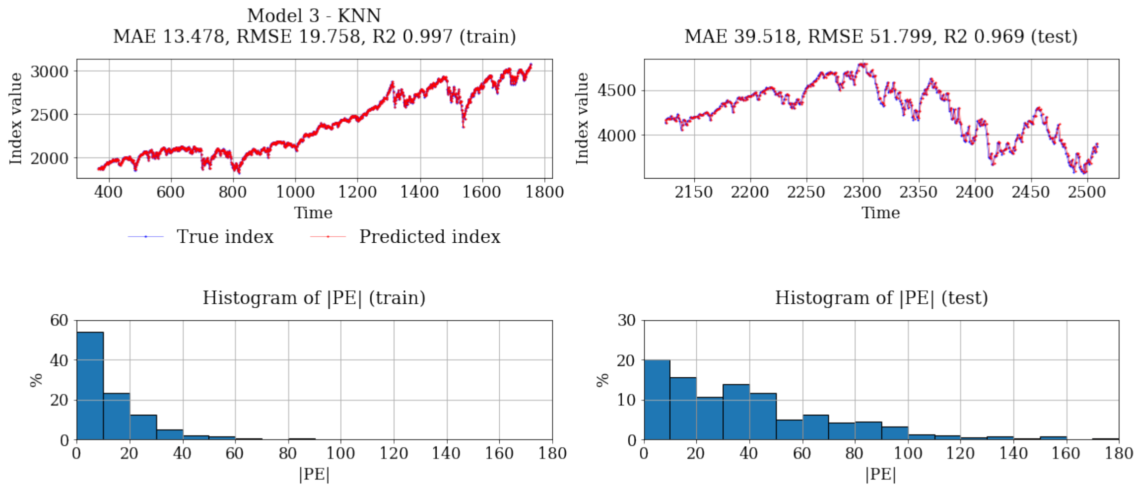


Figure 14. Predictions with model 3.

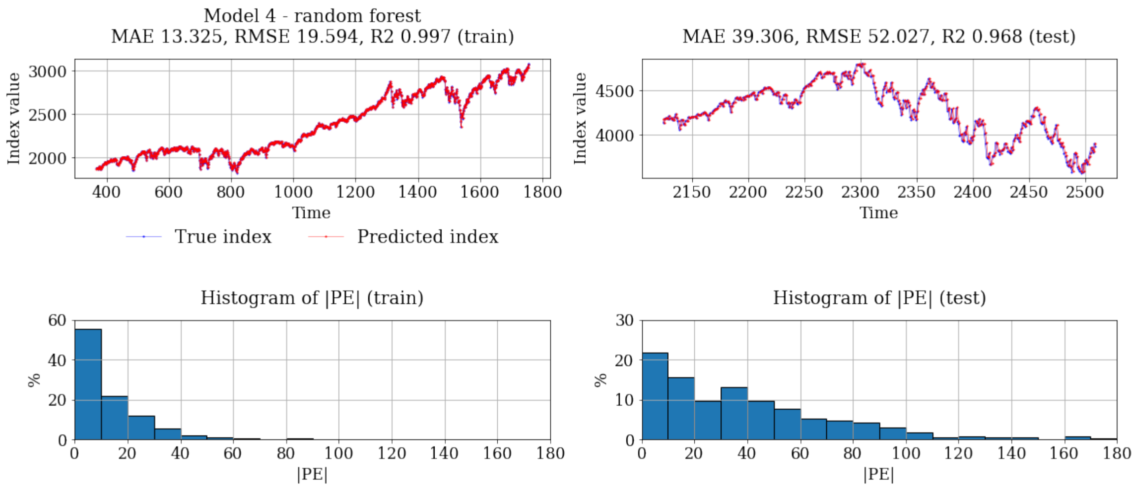


Figure 15. Predictions with model 4.

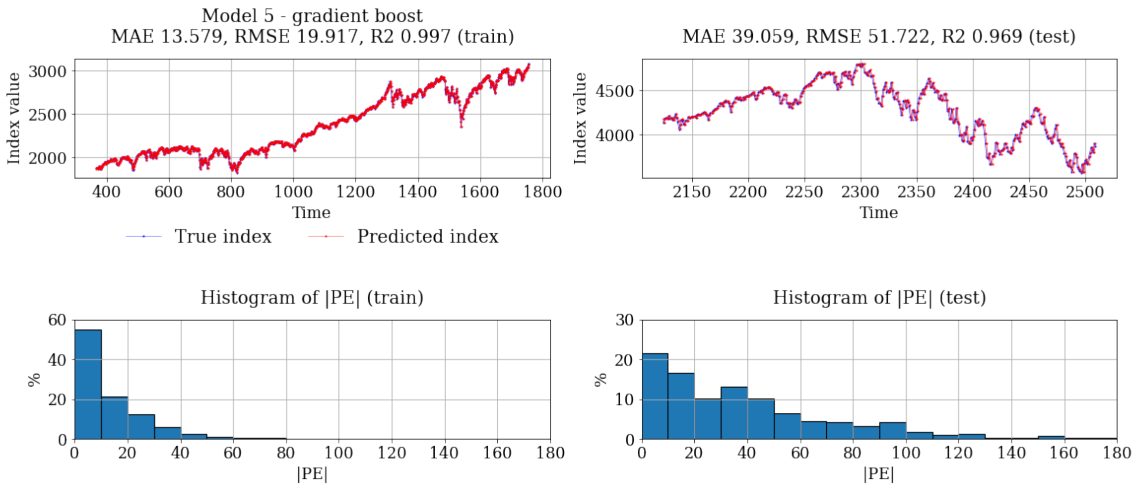


Figure 16. Predictions with model 5.

5.4 Global explainability

5.4.1 Permutation feature importance

Global model-agnostic explainability was investigated by calculating permutation feature importance values from test data. The results without and with normalization of absolute values are shown in Table 6. Note that the results are rounded, and for example values of gradient boost are small but still other than 0. To ease the comparison between features and models, the normalized absolute values are visualized in a heatmap in Figure 17. Also, an example of the model-specific results is shown in Figure 18. Features are listed on the y-axis, and related importance is shown on the x-axis. Based on normalized values in Table 6, the top three most important features on average are F2, F3, and F4. Note that the heatmap contains the normalized absolute values, and thus, the negative and positive effects are no longer visible. Hence, the table should only be used for identifying which features have a negative and which positive effect on the predictions.

Table 6. Permutation feature importances.

	Model	Values							
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**
1	Linear regression	1.277	0.002	116.252	114.151	0.001	-0.003	0.020	0.006
2	Decision tree	0.002	0.018	0.000	0.007	0.002	0.000	0.000	0.002
3	KNN	0.006	0.007	0.006	0.000	0.000	0.001	0.005	0.002
4	Random forest	0.000	0.007	-0.001	-0.003	0.000	0.000	0.000	-0.002
5	Gradient boost	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	AVG	0.257	0.007	23.251	22.831	0.001	0.000	0.005	0.002
		Normalized absolute values							
1	Linear regression	0.011	0.000	1.000	0.982	0.000	0.000	0.000	0.000
2	Decision tree	0.122	1.000	0.000	0.397	0.128	0.000	0.000	0.132
3	KNN	0.825	1.000	0.913	0.000	0.018	0.132	0.743	0.337
4	Random forest	0.005	1.000	0.218	0.375	0.000	0.051	0.021	0.230
5	Gradient boost	0.356	1.000	0.000	0.073	0.203	0.000	0.000	0.278
	AVG	0.264	0.800	0.426	0.365	0.070	0.037	0.153	0.195

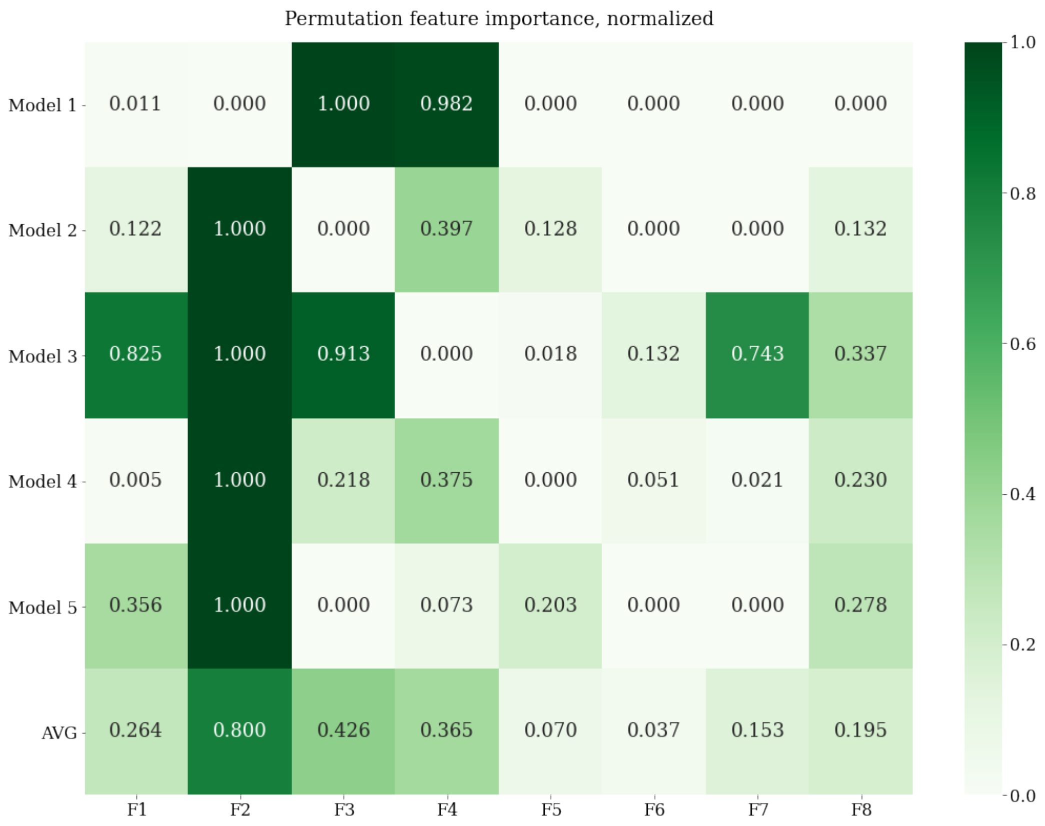


Figure 17. Heatmap of normalized absolute permutation feature importances.

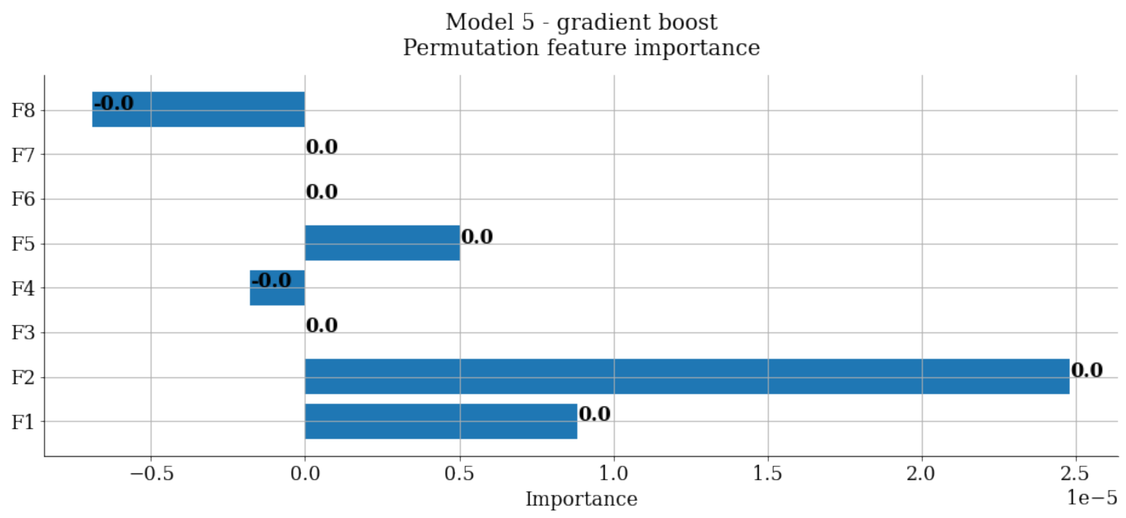


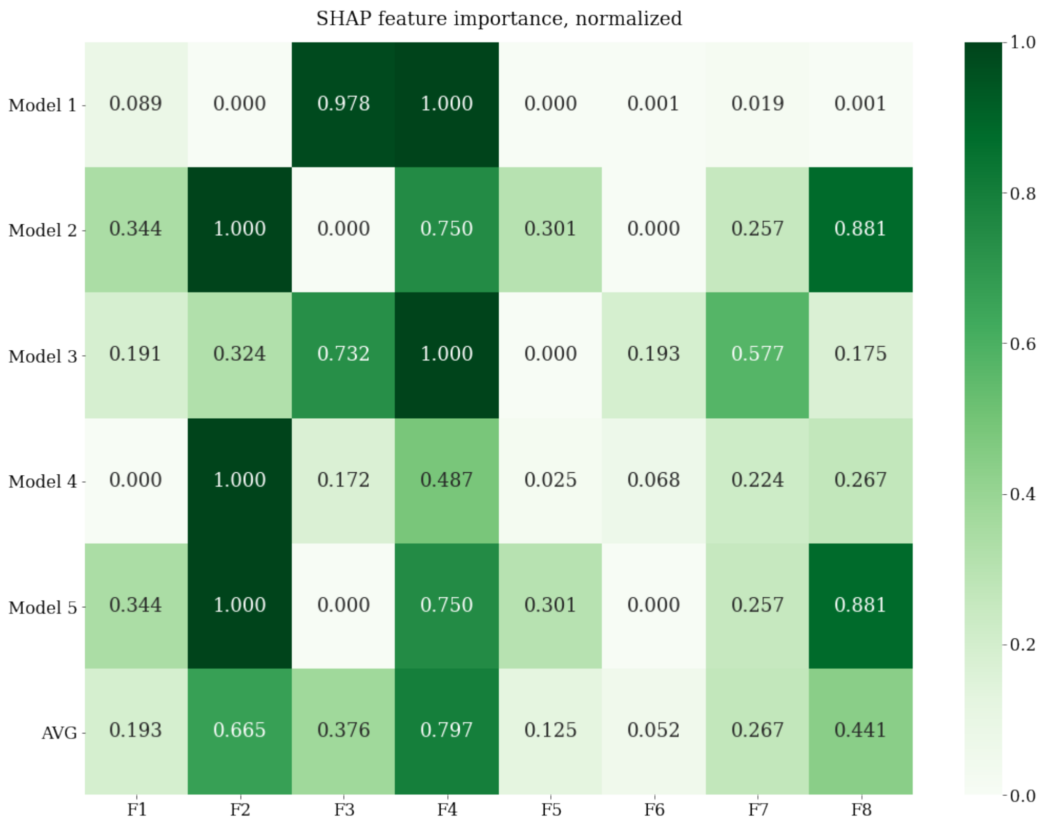
Figure 18. Permutation feature importances of model 5.

5.4.2 SHAP feature importance

Feature importances from test data were also calculated by using SHAP. The results without and with normalization of absolute values are shown in Table 7 and the normalized absolute SHAP feature importance values are visualized in a heatmap in Figure 19. An example of the model-specific results, visualized the same way as before, is shown in Figure 20. Based on normalized values in Table 7, the top three most important features on average are F4, F2, and F8.

Table 7. SHAP feature importances.

	Model	**Values**							
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**
1	Linear regression	0.008	0.000	0.086	0.088	0.000	0.000	0.002	0.000
2	Decision tree	0.000	0.001	0.000	0.001	0.000	0.000	0.000	0.001
3	KNN	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000
4	Random forest	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
5	Gradient boost	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	AVG	0.002	0.000	0.017	0.018	0.000	0.000	0.000	0.000
		Normalized absolute values							
1	Linear regression	0.089	0.000	0.978	1.000	0.000	0.001	0.019	0.001
2	Decision tree	0.344	1.000	0.000	0.750	0.301	0.000	0.257	0.881
3	KNN	0.191	0.324	0.732	1.000	0.000	0.193	0.577	0.175
4	Random forest	0.000	1.000	0.172	0.487	0.025	0.068	0.224	0.267
5	Gradient boost	0.344	1.000	0.000	0.750	0.301	0.000	0.257	0.881
	AVG	0.193	0.665	0.376	0.797	0.125	0.052	0.267	0.441



	F1	F2	F3	F4	F5	F6	F7	F8
Model 1	0.089	0.000	0.978	1.000	0.000	0.001	0.019	0.001
Model 2	0.344	1.000	0.000	0.750	0.301	0.000	0.257	0.881
Model 3	0.191	0.324	0.732	1.000	0.000	0.193	0.577	0.175
Model 4	0.000	1.000	0.172	0.487	0.025	0.068	0.224	0.267
Model 5	0.344	1.000	0.000	0.750	0.301	0.000	0.257	0.881
AVG	0.193	0.665	0.376	0.797	0.125	0.052	0.267	0.441

Figure 19. Heatmap of normalized absolute SHAP feature importances.

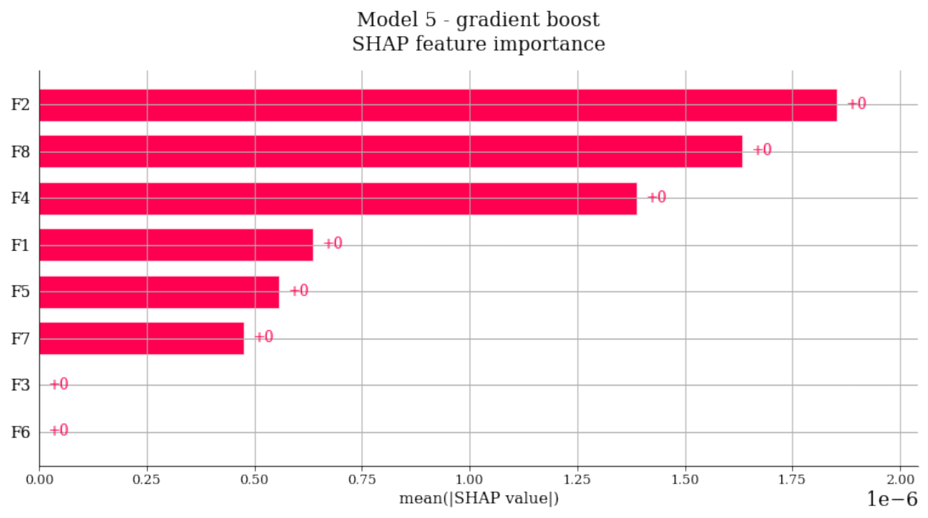


Figure 20. SHAP feature importances of model 5.

5.5 Local explainability

Local model-agnostic explainability was investigated by utilizing LIME. The effects of features on model-specific predictions at timesteps 1 and 100 are presented next.

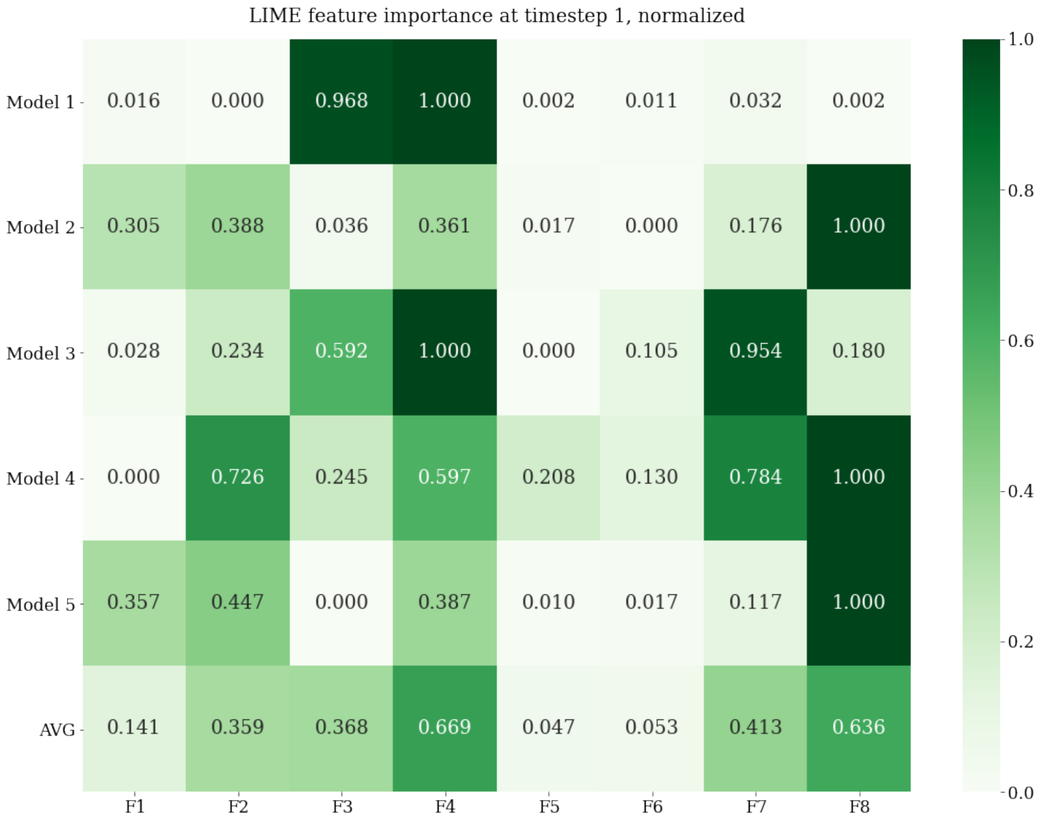
5.5.1 LIME feature importance

Prediction at timestep 1

The results of LIME feature importances at timestep 1 without and with normalization of absolute values are summarized in Table 8 and the normalized absolute LIME feature importance values are visualized in a heatmap in Figure 21. An example of the model-specific results is shown in Figure 22. These results are visualized almost the same way as before, features are shown on the y-axis and the feature effects on prediction are shown on the x-axis. In addition, the red bars indicate negative and the green ones positive effect on the prediction. Note that the order of features on the y-axis is based on the feature effect value. According to Table 8, the top three most important features on average are F4, F8, and F7.

Table 8. LIME feature importances at timestep 1.

	Model	**Values**							
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**
1	Linear regression	-0.001	0.000	0.057	-0.059	0.000	0.001	0.002	0.000
2	Decision tree	0.000	-0.001	0.000	-0.001	0.000	0.000	0.000	0.001
3	KNN	0.000	0.000	0.000	-0.001	0.000	0.000	0.001	0.000
4	Random forest	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	Gradient boost	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	AVG	0.000	0.000	0.011	-0.012	0.000	0.000	0.001	0.000
		Normalized absolute values							
1	Linear regression	0.016	0.000	0.968	1.000	0.002	0.011	0.032	0.002
2	Decision tree	0.305	0.388	0.036	0.361	0.017	0.000	0.176	1.000
3	KNN	0.028	0.234	0.592	1.000	0.000	0.105	0.954	0.180
4	Random forest	0.000	0.726	0.245	0.597	0.208	0.130	0.784	1.000
5	Gradient boost	0.357	0.447	0.000	0.387	0.010	0.017	0.117	1.000
	AVG	0.141	0.359	0.368	0.669	0.047	0.053	0.413	0.636



	F1	F2	F3	F4	F5	F6	F7	F8
Model 1	0.016	0.000	0.968	1.000	0.002	0.011	0.032	0.002
Model 2	0.305	0.388	0.036	0.361	0.017	0.000	0.176	1.000
Model 3	0.028	0.234	0.592	1.000	0.000	0.105	0.954	0.180
Model 4	0.000	0.726	0.245	0.597	0.208	0.130	0.784	1.000
Model 5	0.357	0.447	0.000	0.387	0.010	0.017	0.117	1.000
AVG	0.141	0.359	0.368	0.669	0.047	0.053	0.413	0.636

Figure 21. Heatmap of normalized absolute LIME feature importances at timestep 1.

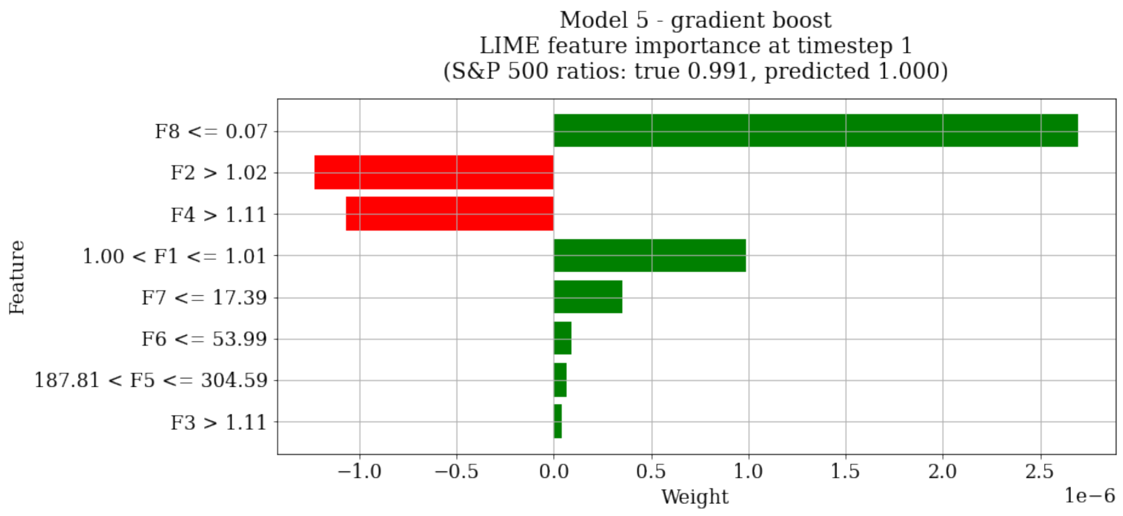


Figure 22. LIME feature importances of model 5 at timestep 1.

Prediction at timestep 100

The results of LIME feature importances at timestep 100 without and with normalization of absolute values are summarized in Table 9 and the normalized absolute LIME feature importance values are visualized in a heatmap shown in Figure 23. An example of the model-specific results is shown in Figure 24. According to Table 9, the top three most important features on average are F4, F8, and F2.

Table 9. LIME feature importances at timestep 100.

	Model	Values							
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**
1	Linear regression	0.013	0.000	0.057	-0.057	-0.001	0.001	0.003	0.002
2	Decision tree	0.000	-0.001	0.000	-0.001	0.000	0.000	0.000	0.001
3	KNN	0.000	0.000	0.000	-0.001	0.000	0.000	0.001	0.000
4	Random forest	0.000	-0.001	0.000	0.000	0.000	0.000	0.000	0.000
5	Gradient boost	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	AVG	0.002	0.000	0.011	-0.012	0.000	0.000	0.001	0.001
		Normalized absolute values							
1	Linear regression	0.218	0.000	1.000	0.997	0.005	0.007	0.037	0.020
2	Decision tree	0.264	0.457	0.009	0.440	0.117	0.000	0.197	1.000
3	KNN	0.173	0.113	0.831	0.970	0.000	0.436	1.000	0.175
4	Random forest	0.048	1.000	0.204	0.420	0.032	0.000	0.454	0.535
5	Gradient boost	0.310	0.416	0.010	0.434	0.143	0.000	0.144	1.000
	AVG	0.203	0.397	0.411	0.652	0.059	0.089	0.366	0.546

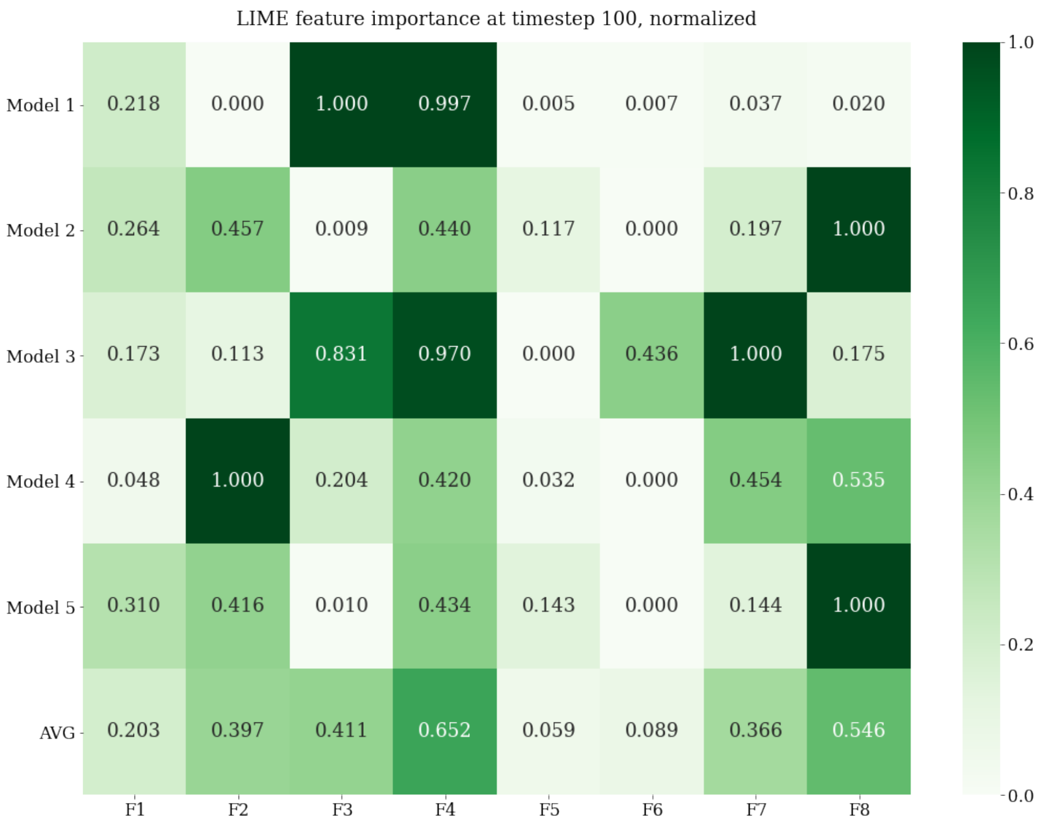


Figure 23. Heatmap of normalized absolute LIME feature importances at timestep 100.

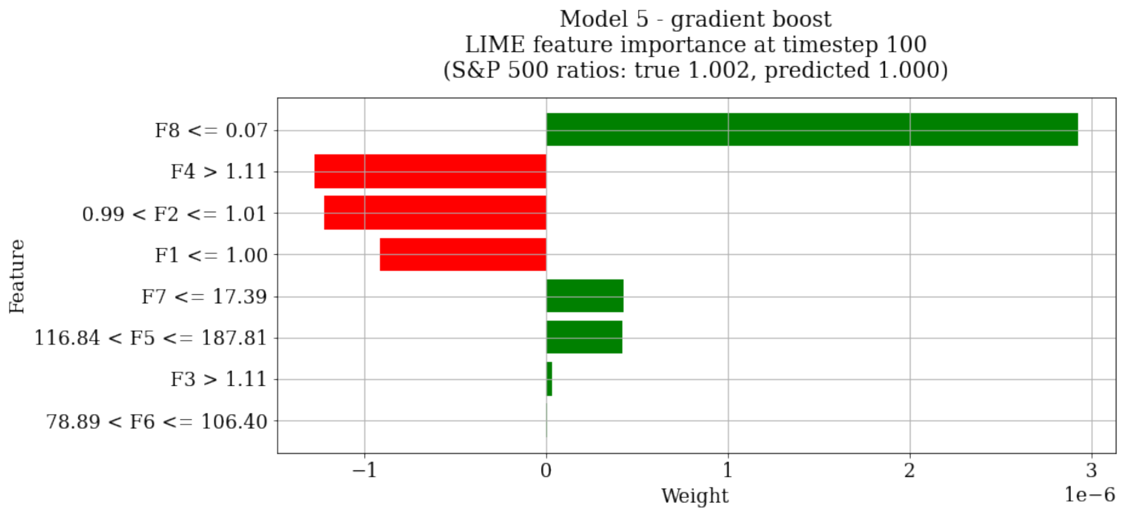


Figure 24. LIME feature importances of model 5 at timestep 100.

5.5.2 SHAP feature importance

Prediction at timestep 1

The results of SHAP feature importances at timestep 1 without and with normalization of absolute values are shown in Table 10 and the normalized absolute SHAP feature importance values are visualized in a heatmap in Figure 25. An example of the model-specific results is shown in Figure 26. Based on normalized values in Table 10, the top three most important features on average are F4, F8, and F3.

Table 10. SHAP feature importances at timestep 1.

	Model	**Values**							
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**
1	Linear regression	0.001	0.000	0.120	0.123	0.000	0.000	0.002	0.000
2	Decision tree	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001
3	KNN	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000
4	Random forest	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	Gradient boost	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	AVG	0.000	0.000	0.024	0.025	0.000	0.000	0.000	0.000
		Normalized absolute values							
1	Linear regression	0.009	0.001	0.977	1.000	0.000	0.001	0.015	0.001
2	Decision tree	0.073	0.202	0.000	0.110	0.084	0.000	0.065	1.000
3	KNN	0.000	0.009	0.314	1.000	0.021	0.486	0.099	0.167
4	Random forest	0.571	0.548	0.539	0.814	0.588	0.000	1.000	0.773
5	Gradient boost	0.073	0.202	0.000	0.110	0.084	0.000	0.065	1.000
	AVG	0.145	0.192	0.366	0.607	0.155	0.097	0.249	0.588

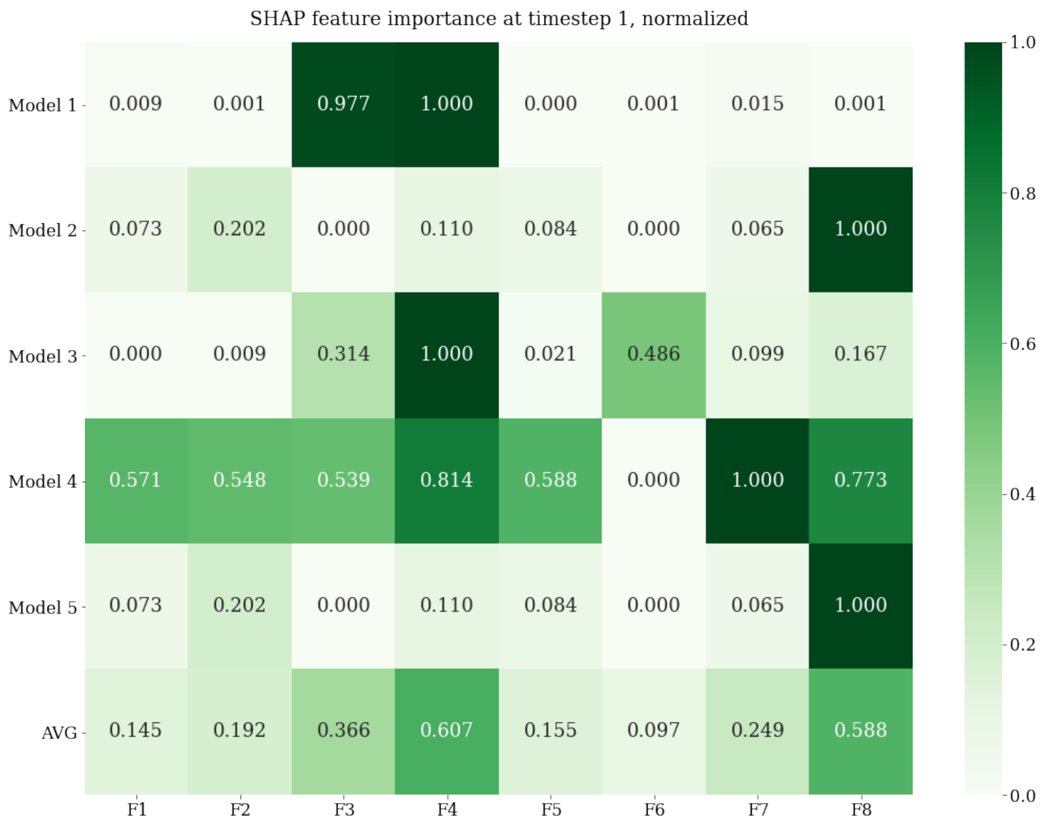


Figure 25. Heatmap of normalized absolute SHAP feature importances at timestep 1.

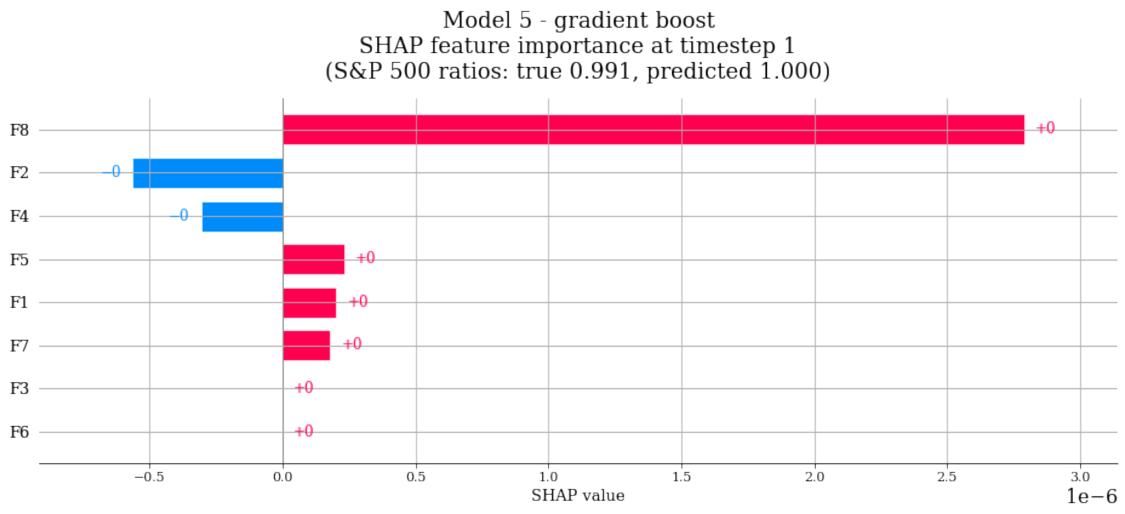


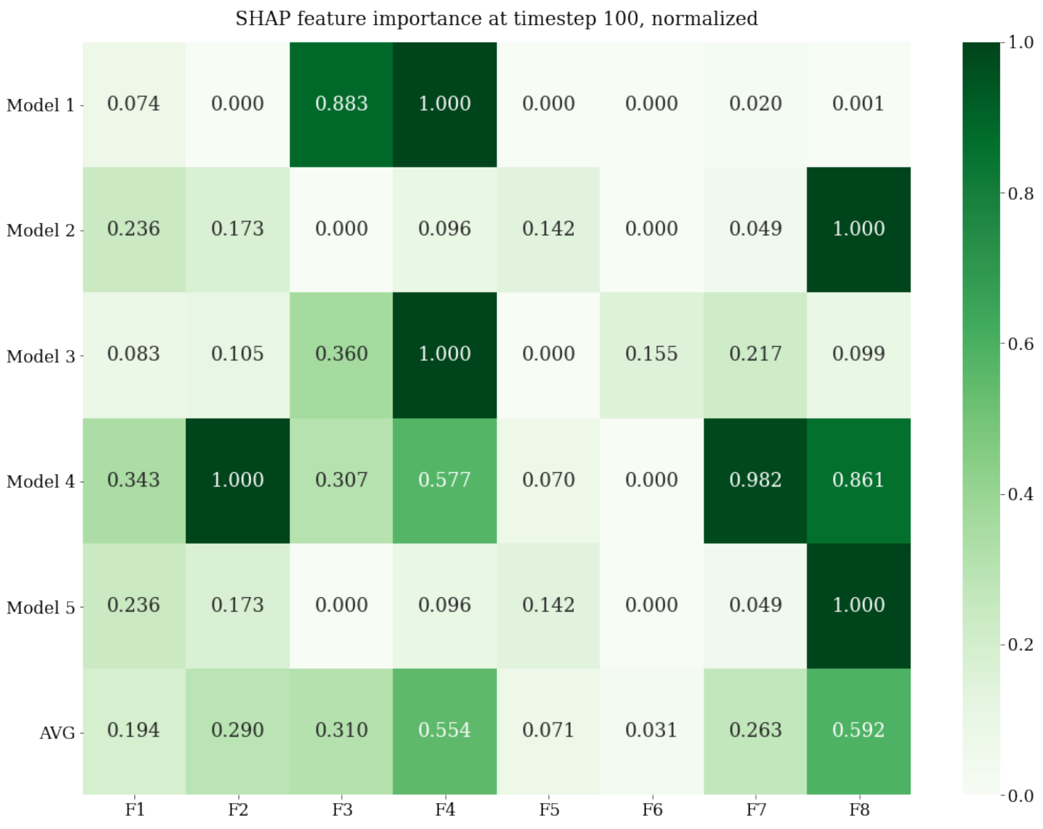
Figure 26. SHAP feature importances of model 5 at timestep 1.

Prediction at timestep 100

The results of SHAP feature importances at timestep 100 without and with normalization of absolute values are shown in Table 11 and the normalized absolute SHAP feature importance values are visualized in a heatmap in Figure 27. An example of the model-specific results is shown in Figure 28. Based on normalized values in Table 11, the top three most important features on average are F8, F4, and F3.

Table 11. SHAP feature importances at timestep 100.

	Model	**Values**							
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**
1	Linear regression	0.008	0.000	0.096	0.109	0.000	0.000	0.002	0.000
2	Decision tree	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002
3	KNN	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000
4	Random forest	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	Gradient boost	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	AVG	0.002	0.000	0.019	0.022	0.000	0.000	0.001	0.000
		Normalized absolute values							
1	Linear regression	0.074	0.000	0.883	1.000	0.000	0.000	0.020	0.001
2	Decision tree	0.236	0.173	0.000	0.096	0.142	0.000	0.049	1.000
3	KNN	0.083	0.105	0.360	1.000	0.000	0.155	0.217	0.099
4	Random forest	0.343	1.000	0.307	0.577	0.070	0.000	0.982	0.861
5	Gradient boost	0.236	0.173	0.000	0.096	0.142	0.000	0.049	1.000
	AVG	0.194	0.290	0.310	0.554	0.071	0.031	0.263	0.592



	F1	F2	F3	F4	F5	F6	F7	F8
Model 1	0.074	0.000	0.883	1.000	0.000	0.000	0.020	0.001
Model 2	0.236	0.173	0.000	0.096	0.142	0.000	0.049	1.000
Model 3	0.083	0.105	0.360	1.000	0.000	0.155	0.217	0.099
Model 4	0.343	1.000	0.307	0.577	0.070	0.000	0.982	0.861
Model 5	0.236	0.173	0.000	0.096	0.142	0.000	0.049	1.000
AVG	0.194	0.290	0.310	0.554	0.071	0.031	0.263	0.592

Figure 27. Heatmap of normalized absolute SHAP feature importances at timestep 100.

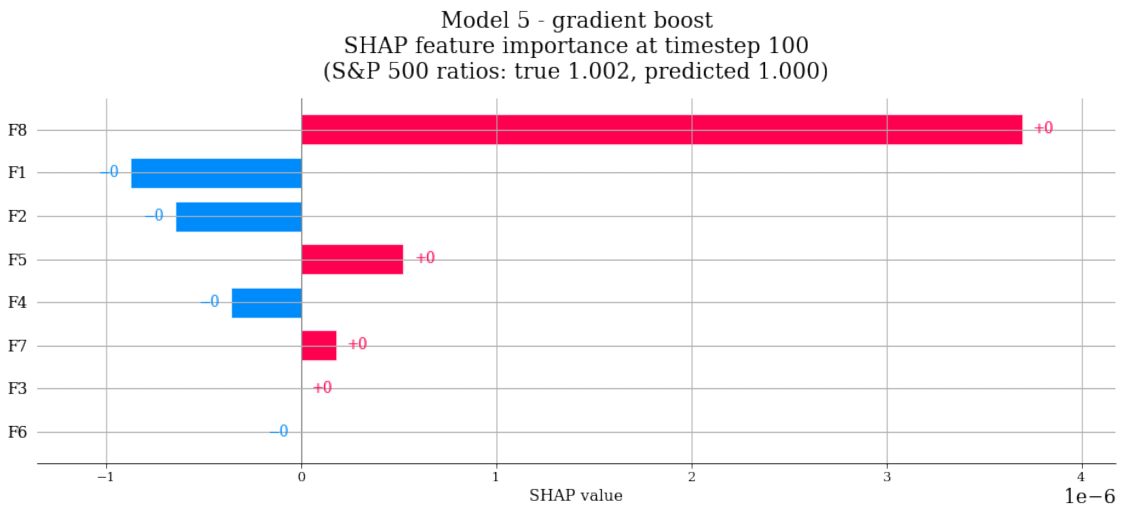


Figure 28. SHAP feature importances of model 5 at timestep 100.

5.6 Additional SHAP analysis with best model

The best model, gradient boost, was investigated further by using SHAP. Visualizations of SHAP summary plot, heatmap, and force plots at timesteps 1 and 100 are shown in Figures 29–32.

In Figure 29, SHAP values are shown on the x-axis, features are shown on the y-axis on the left, and feature values are shown on the y-axis on the right. Values are color-coded, red indicates high and blue low feature value. Values on the left side of the horizontal line have a negative effect, whereas values on the right side of the horizontal line have a positive effect on the prediction. For example, high and low values of F2 seem to have a great effect on predictions, whereas F3 and F6 seem to have the lowest effect on predictions in general.

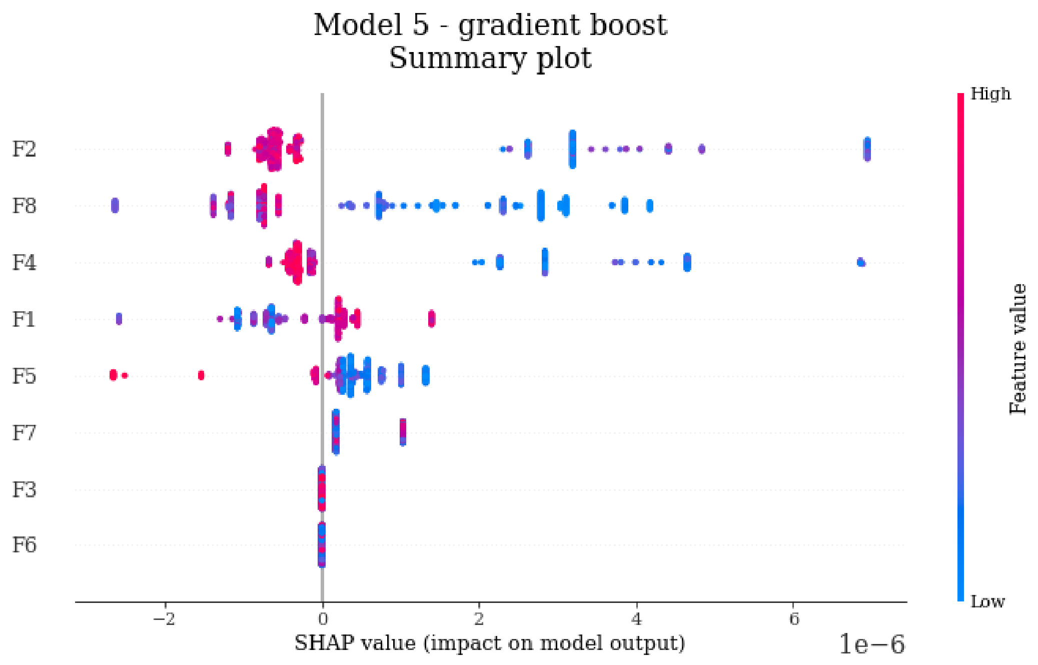


Figure 29. SHAP summary plot of best model.

In Figure 30, features are shown on the left and SHAP values are shown on the right. Predictions are visualized with respect to instances on top of all feature-specific information. In addition, global feature importance of each feature is visualized in a bar plot on the right. As interpreted from Figure 29, high F2 values affect prediction greatly. The same is observable from the heatmap, but now the effect of features on predictions is more clearly visualized.

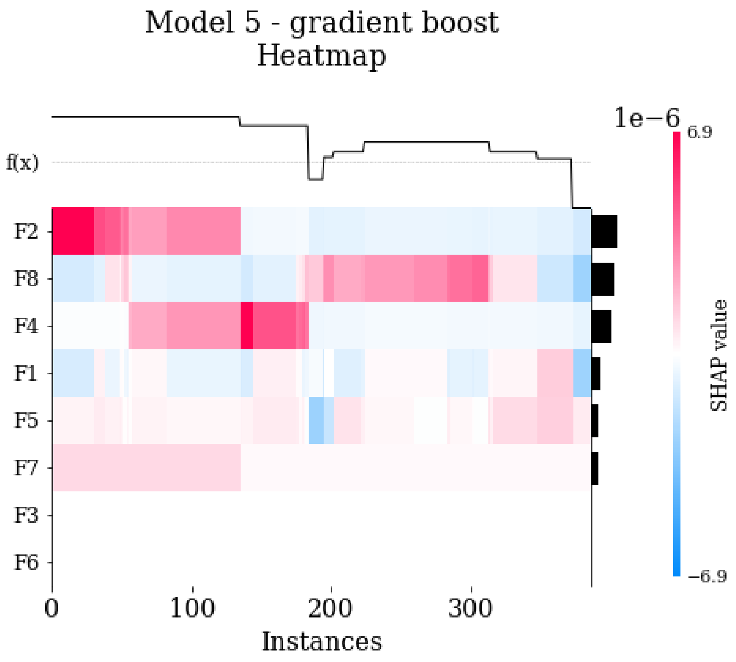


Figure 30. SHAP heatmap of best model.

The effect of the most important features on predictions at timesteps 1 and 100 are shown in Figures 31 and 32. Again, color-coding is used; blue indicates a negative and red positive effect on prediction. The bold number stands for the timestep-specific prediction. The base value represents the average of predictions in the dataset. For example, at timestep 1, F2 has a negative effect on the prediction, whereas at timestep 100, both features F1 and F2 have a negative effect on the prediction. Other features have a positive effect on the predicted value. In comparison to earlier permutation, LIME, and SHAP feature importance visualizations, the force plot is a more compact visualization because the effect of each feature on prediction is shown on the same line.

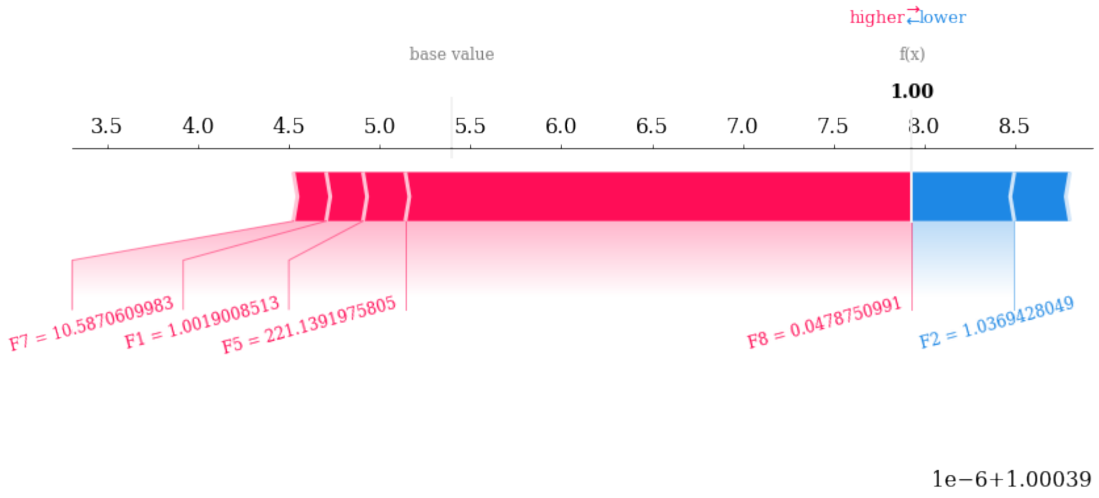


Figure 31. SHAP force plot of best model at timestep 1.

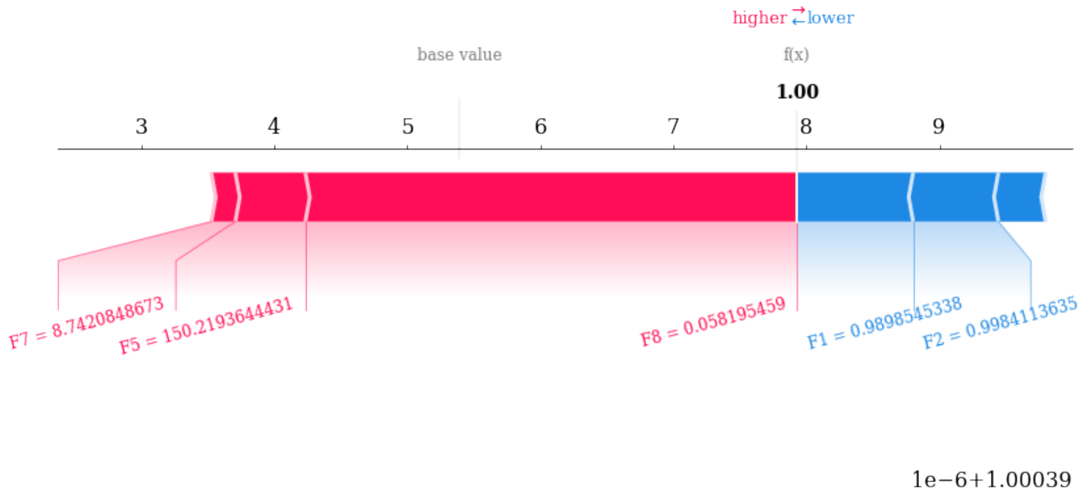


Figure 32. SHAP force plot of best model at timestep 100.

5.7 Model-specific explainability

5.7.1 Structure of decision tree

Model 2, a decision tree, contains six layers. It is transparent in all three categories of model transparency. Consequently, it is possible to visualize its structure, and by following the if-else logic from the top to the bottom of the decision tree, it is possible to understand how the model works. The structure is partially visualized in Figures 33–35. The visualizations contain the first three layers of the decision tree. As seen from these visualizations, the first layer is based on F2, the second on F7 and F4, and the third one on F8.

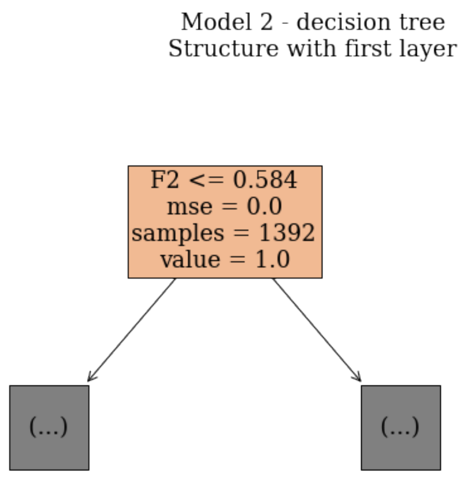


Figure 33. Model 2 structure with the first layer.

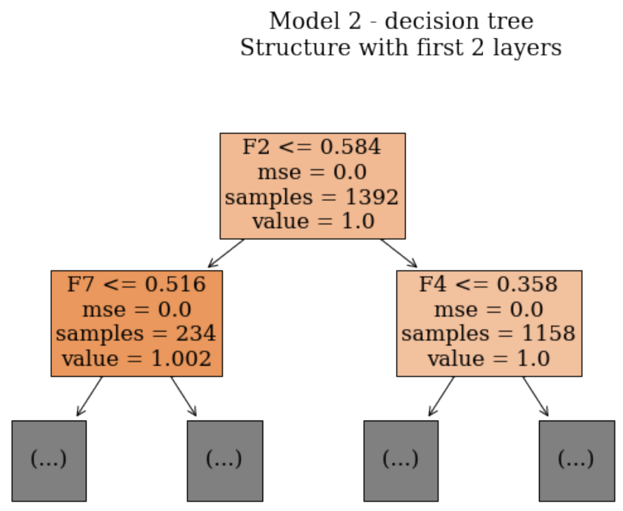


Figure 34. Model 2 structure with first two layers.

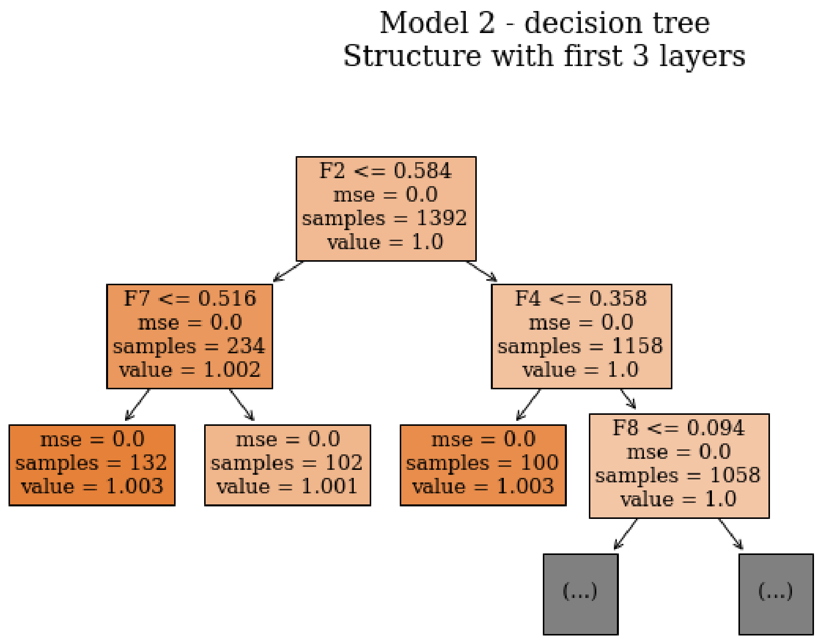


Figure 35. Model 2 structure with first three layers.

5.7.2 Structure of best model

Model 5, based on gradient boost, is an ensemble of 20 decision trees. Because its structure cannot be directly visualized, it can be indirectly visualized by training a decision tree with the same features and original predictions as responses and visualizing the structure of the trained model. By following the if-else logic of the decision tree trained, it is possible to better understand how the original model works.

The decision tree contains 39 layers. The structure is partially visualized in Figures 36–38. The visualizations contain the first three layers of the decision tree the same way as before. As seen from Figure 36, the first layer is based on F2, the second one on F4 and F7, and the third one on F7, F4, F2, and F8.

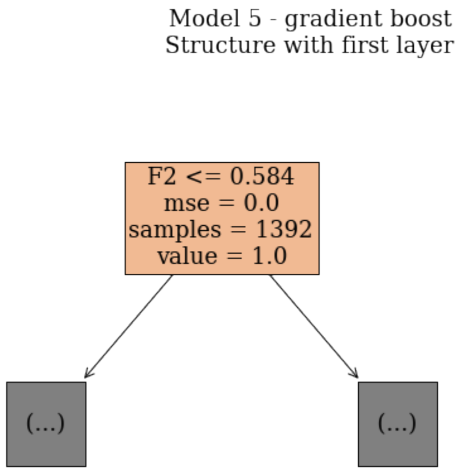


Figure 36. Model 5 structure with the first layer.

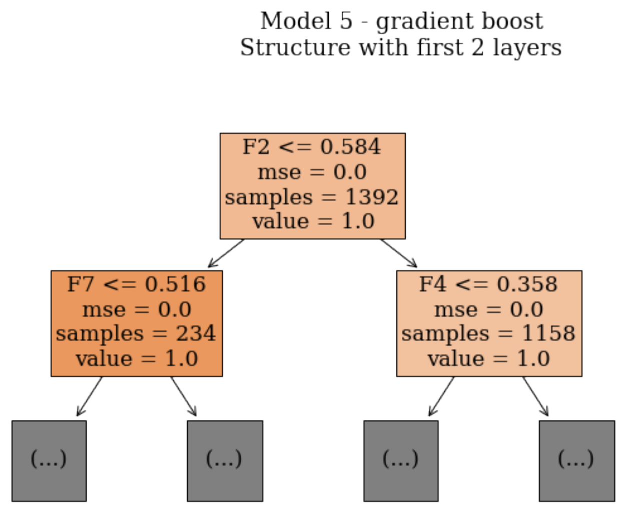


Figure 37. Model 5 structure with first two layers.

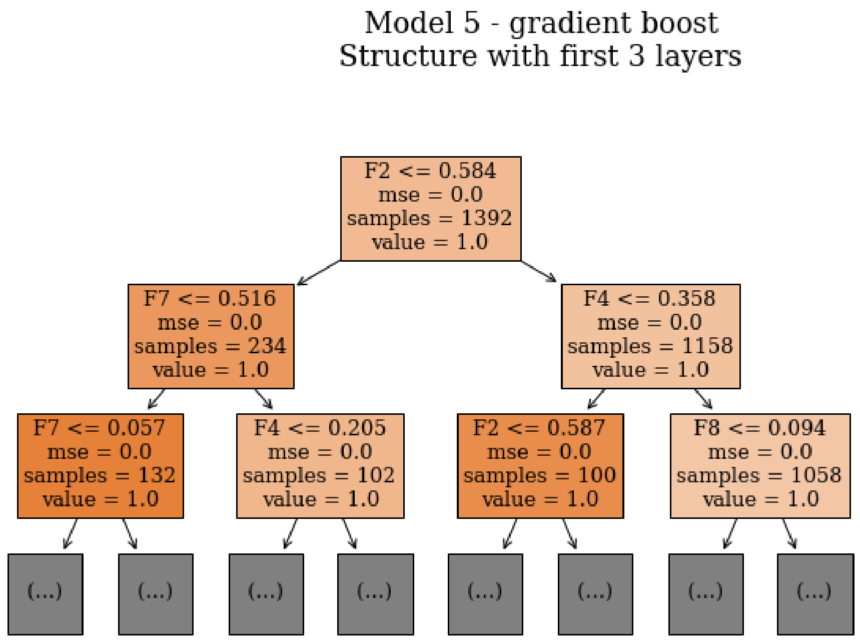


Figure 38. Model 5 structure with first three layers.

6 DISCUSSION

6.1 Results

6.1.1 Prediction accuracy

Five different models were trained. Training and test set prediction accuracy was evaluated by using MAE, RMSE, and R^2 score. With training data, MAEs were all roughly 13, but with test data MAEs were close to 40. The models may overfit with training data because prediction accuracies with training data were better than with test data. The best prediction accuracy with test data was obtained with model 5, gradient boost.

6.1.2 Explainability methods

Both global and local model-agnostic explainability of five different models were investigated. In practice, permutation feature importance values were calculated for each feature to globally explain how features affect predictions. In addition, LIME was used to explain how features affect predictions at specific timesteps. Permutation importance and LIME were selected according to the study of Carta et al. (2022). In addition to permutation importance and LIME, SHAP was also used to investigate model behavior both at a global and local level. SHAP was also used to investigate the best model in more detail.

In general, the level of explainability of the five models included in this thesis was good. The level of explainability results mainly from the simplicity of the model structures. Permutation importance, LIME, and SHAP are useful tools to get explanations efficiently and to find out whether other methods or deeper explainability analysis are needed. An easy way to investigate models more would be to simply continue using SHAP - as illustrated in this thesis.

The results between the models can also be easily compared for example by using normalized absolute values and a heatmap. This way it is easy to see what the most important features are and how these differ between 1) the models, 2) instances, and 3) between explainability methods. Thus, by comparing global and local explainability results between the models, it can also be possible to identify what features are generally useful with selected models and what features may help in fine-tuning the predictions. In the majority of the feature importance analyses, F4 was on average the most important feature. The second and third most important features varied commonly between F2 and F8.

In addition to using permutation importance, LIME, and SHAP, decision tree and best model were investigated further by visualizing the model structures. Because the best model consists of several decision trees, its structure is more complex than a single decision tree. Consequently, its structure was indirectly visualized by using a surrogate model. In practice, a decision tree was first trained with the features and predictions of the original model, and then the structure of the decision tree was visualized to explain how the best model makes predictions.

By using these explainability methods, it was possible to better understand how the models could be analyzed, how features affect model-specific predictions, what information the selected methods provide, and how this information can be utilized. In addition to the methods used in this thesis, several other post-hoc explainability methods were covered in Section 2. Based on the literature search, there are several relatively easy-to-use and interpretable explainability methods available. Of course, each method has its benefits and drawbacks, and thus, it is important to get started with selected methods, analyze results critically, and deepen the explainability analysis as per need.

6.1.3 Prediction accuracy versus level of explainability

Model interpretability and prediction accuracy indirectly proportional (Arrieta et al. 2020): the better the prediction accuracy, the poorer the model interpretability. A similar phenomenon was found in this thesis: model 1, linear regression, had the highest MAE with test data, whereas model 5, gradient boost, had the smallest MAE with test data.

6.2 Improvement opportunities

6.2.1 Prediction accuracy

This thesis work could be continued by testing how alternative features, model hyperparameter optimization, data windowing, and deep learning affect prediction accuracy. Especially the effect of other seasonal features, such as public holidays and weekdays, on S&P 500 index predictions could be investigated. In addition, because according to the literature search, LSTM has been found a useful model in time series forecasting and efficient to learn long-term dependencies in data, the usability of LSTM could also be investigated.

Some lag between true and predicted index values was observed. The lag in test data was investigated in more detail by calculating the absolute delay between the time of true and the time of the next closest predicted index value. The median and mode of delays were both one day. Hence, the typical lag was one day and longer delays were temporary. Most likely, the earlier index value (t-1) and lack of features describing day-level changes in the index contribute to the lag. Smoothing features before taking predictions as well as using alternative features and deep learning could help in reducing the small lag in the forecast. These could reduce the number of outliers in features and help the model and predictions to adapt to the true trend better.

Prediction accuracies could also be compared to results reported in similar studies. Based on the brief literature search, the work by Freeborough & van Zyl (2022) was found to be close to this thesis. However, some differences exist. For example, in this study, deep learning models LSTM, GRU, and RNN were used to predict S&P 500 index by using features derived from S&P 500 stock data. Also, symmetric mean absolute percentage error (SMAPE, Equation 19) was used by Freeborough & van Zyl (2022) to evaluate prediction accuracy instead of MAE, RMSE, and R^2 score

$$\text{SMAPE} = \frac{2}{N}\sum_{i=1}^{N}\frac{|Y_i - \hat{Y}_i|}{|Y_i| + |\hat{Y}_i|}, \tag{19}$$

where N is the number of instances, Y_i are true, and $\hat{Y}_i$ are predicted values. Hence, if this thesis topic is investigated further, SMAPE could also be calculated. Because SMAPE limits the effect of individual high errors on prediction accuracy, it could ease the comparison between selected studies and provide a more robust view of the model performance.

6.2.2 Explainability methods

Permutation importance, LIME, and SHAP provided sufficient information about how the features affect predictions and what features are the most important ones. If wanted, models could be further investigated by using selected model-specific methods to get more insight into how features interact with a specific model. Additional analysis was not, however, included in this thesis because the research questions narrowed the work to a more generic level of explainable AI concept.

To keep both model development and explainability analysis as simple as possible, the level of explainability analysis should be adjusted according to the development phase, needed level of explainability, and expected audience. In other words, several explainability methods are available, and there is no point in using all methods to both improve the model and explain how it works. This type of approach would easily leave the developer(s) and other audience overwhelmed by various colorful graphs. Instead, it would be better to first narrow the number of methods to be used to the minimum, and then widen the scope of explainability analysis as per need.

For example, if the model development has just started, first it could be most useful to investigate how features affect the predictions in general. This analysis could be then used to improve features and exclude irrelevant ones if there are any. Correlation analysis between features and responses can also provide useful information for data preparation. Eventually, the model development reaches a point in which the model works well with specific instances. Hence, to generalize the model more, it could be useful to deepen the explainability analysis. This could mean investigating how features affect predictions at specific instances, as well as identifying how and why the effect of each feature at specific instances varies. Again, this information could be then utilized to improve the model. Model-specific explainability methods could also be used if needed. Lastly, once the model has been developed, the most appropriate explainability methods, such as SHAP, could be used to provide a comprehensive but compact and interpretable summary from XYZ perspectives to explain how the model developed works.

7 CONCLUSIONS

This thesis consisted of two parts, model development and model explainability analysis. To find out what the baseline prediction accuracy is for S&P 500 index forecasting and to keep data preparation works as simple as possible, models were kept simple and features and responses were derived from S&P index data only. Selected time series regression models; linear regression, decision tree, k-nearest neighbors, random forest, and gradient boost; were developed to predict S&P 500 index. Prediction accuracy was evaluated by mean absolute error, root mean square error, and R^2 score. Selected explainability methods found in the literature were used to investigate models. The main focus was on model-agnostic post-hoc methods that can be applied to any trained model. Explainability of all five models was also investigated both at a global and local level by using permutation importance from sklearn library, LIME from LIME library, and SHAP from SHAP library. Also, the explainability of the best model was further investigated by using SHAP.

The best model was model 5, gradient boost. The prediction accuracy of the best model was considered sufficient both for a baseline version and explainability analysis. As expected, the level of explainability of each model was good, mainly due to the relatively simple structures of the models analyzed. The structures of decision tree and best model (indirectly) were also visualized. Permutation, LIME, and SHAP feature importances were also compared to find out what the most important features are. The explainability methods used in this thesis were found sufficient for these types of models.

Three research questions were answered in this thesis. The first research question, what is meant by the explainability of AI, is about a new concept that has become popular in recent years. Explainable AI, or XAI, is about methods that can be used to explain reliably and understandably how machine learning models work. Also, it aims to improve other aspects of model development, such as the trustworthiness, confidence, and informativeness of models. Findings from explainability analysis can also be utilized to improve models even more. The second research question, how the explainability of AI can be evaluated in practice, was answered by summarizing existing methods available. Taxonomy of different explainability methods and summaries of methods at a more general level were also covered to help in understanding how all the terms relate to each other. The third research question, what evaluation methods can be used for investigating the explainability of time series forecasting models, was answered by showing how selected global and local model-agnostic methods can be used in practice with the models trained and what kind of results each method outputs.

The potential next steps of this thesis could focus on using deep learning and additional explainability methods. For example, LSTM could be trained with similar features, and the prediction accuracy could be compared to those reported in this thesis. By comparing the results, it could be possible to see whether a complex model brings additional value to the prediction problem. Additional explainability methods, such as those specific to certain models, could also be used to investigate model behavior in more detail. The necessity of additional explainability methods could also be investigated.

To conclude, this thesis clarified what is meant by explainable AI, what existing explainability methods are available, and how the explainability of common time series models could be investigated in practice. The concept of XAI is relatively new. Because of its benefits in improving models and clarifying model behavior to a wide audience, in near future, XAI can be expected to 1) become a more popular concept and 2) an important part of model development. Several model-agnostic and -specific explainability methods for investigating model behavior both at a global and local level exist. Some of the most desired characteristics of explainability methods include interpretability, intuitiveness, robustness, easy-to-use, and comprehensiveness. The last one, comprehensiveness, a characteristic of a method that can be used to investigate any model at different levels, still seems to be underway.

REFERENCES

Angelov, Plamen P.; Soares, Eduardo A.; Jiang, Richard; Arnold, Nicholas I. & Atkinson, Peter M. 2021, Explainable artificial intelligence: an analytical review, *WIREs Data Mining and Knowledge Discovery*, vol. 11, no. 5, p. e1424.

Apley, Daniel W & Zhu, Jingyu. 2020, Visualizing the effects of predictor variables in black box supervised learning models, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 82, no. 4, pp. 1059–1086.

Arrieta, Alejandro Barredo; Díaz-Rodríguez, Natalia; Ser, Javier Del; Bennetot, Adrien; Tabik, Siham; Barbado, Alberto; Garcia, Salvador; Gil-Lopez, Sergio; Molina, Daniel; Benjamins, Richard; Chatila, Raja & Herrera, Francisco. 2020, Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI, *Information Fusion*, vol. 58, , pp. 82–115.

Balasubramaniam, Kesavan. 2022, *How to calculate a stock's adjusted closing price*. Available: https://www.investopedia.com/ask/answers/06/adjustedclosingprice.asp.

Breiman, Leo. 2001, Random forests, *Machine Learning*, vol. 45, no. 1, pp. 5–32.

Breiman, Leo; Friedman, Jerome H.; Olshen, Richard A. & Stone, Charles J. 2017, *Classification and regression trees*, Routledge.

Carta, Salvatore; Podda, Alessandro Sebastian; Reforgiato Recupero, Diego & Stanciu, Maria Madalina. 2022, Explainable AI for financial forecasting, vol. 13164 LNCS, Springer Science and Business Media Deutschland GmbH, pp. 51–69.

Chen, James. 2022, *Index investing*. Available: https://www.investopedia.com/terms/i/index-investing.asp.

Chen, Tianqi & Guestrin, Carlos. 2016, XGBoost, Association for Computing Machinery, pp. 785–794.

Dandl, Susanne; Molnar, Christoph; Binder, Martin & Bischl, Bernd. 2020, Multi-objective counterfactual explanations, Springer International Publishing, pp. 448–469.

Fischer, Thomas & Krauss, Christopher. 2018, Deep learning with long short-term memory networks for financial market predictions, *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669.

Freeborough, Warren & van Zyl, Terence. 2022, Investigating explainability methods in recurrent neural network architectures for financial time series data, *Applied Sciences*, vol. 12, no. 3, p. 1427.

Friedman, Jerome H. 2001, Greedy function approximation: a gradient boosting machine, *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232.

Friedman, Jerome H. 2002, Stochastic gradient boosting, *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378.

Friedman, Jerome H & Popescu, Bodgan E. 2008, Predictive learning via rule ensembles, *The Annals of Applied Statistics*, vol. 2, no. 3, pp. 916–954.

Goldstein, Alex; Kapelner, Adam; Bleich, Justin & Pitkin, Emil. 2015, Peeking inside the black box: visualizing statistical learning with plots of individual conditional expectation, *Journal of Computational and Graphical Statistics*, vol. 24, no. 1, pp. 44–65.

Goodfellow, Ian; Bengio, Yoshua & Courville, Aaron. 2016, *Deep learning*, MIT Press.

Goodman, Bryce & Flaxman, Seth. 2017, European union regulations on algorithmic decision making and a “right to explanation”, *AI Magazine*, vol. 38, no. 3, pp. 50–57.

Hastie, Trevor; Tibshirani, Robert & Friedman, Jerome H. 2009, *The elements of statistical learning: data mininig, inference, and prediction*, 2 edn., Springer.

Ilic, Igor; Görgülü, Berk; Cevik, Mucahit & Baydoğan, Mustafa Gökçe. 2021, Explainable boosted linear regression for time series forecasting, *Pattern Recognition*, vol. 120, , pp. 108–144.

Indices, S&P 500 Dow Jones. 2022a, *S&P Dow Jones Indices: index methodology index mathematics methodology*. Available: https://www.spglobal.com/spdji/en/documents/methodologies/methodology-index-math.pdf.

Indices, S&P Dow Jones. 2022b, *S&P 500 factsheet*. Available: http://www.spglobal.com/spdji/en.

Kenton, Will. 2022, *The S&P 500 Index: Standard & Poor's 500 Index*.

Krollner, Bjoern; Vanstone, Bruce J & Finnie, Gavin R. 2010, Financial time series forecasting with machine learning techniques: a survey, pp. 25–30.

Larxel. 2022, *S&P 500 stocks (daily updated)*. Available: https://www.kaggle.com/andrewmvd/sp-500-stocks?select=sp500_stocks.csv.

Lundberg, Scott M & Lee, Su-In. 2017, A unified approach to interpreting model predictions, Curran Associates, Inc., pp. 4768–4777.

Mehrotra, Dheeraj. 2019, *Basics of artificial intelligence & machine learning*, Notion Press.

Molnar, Christoph. 2022, *Interpretable machine learning: a guide for making black box models explainable*, 2 edn.. Available: https://christophm.github.io/interpretable-ml-book/.

Raschka, Sebastian & Mirjalili, Vahid. 2019, *Python machine learning*, 3 edn., Packt Publishing.

Ribeiro, Marco Tulio; Singh, Sameer & Guestrin, Carlos. 2016, "Why should I trust you?" Explaining the predictions of any classifier, vol. 13-17-August-2016, Association for Computing Machinery, pp. 1135–1144.

Ribeiro, Marco Tulio; Singh, Sameer & Guestrin, Carlos. 2018, Anchors: high-precision model-agnostic explanations, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1.

Rojat, Thomas; Puget, Raphaël; Filliat, David; Ser, Javier Del; Gelin, Rodolphe & Díaz-Rodríguez, Natalia. 2021, Explainable artificial intelligence (XAI) on timeseries data: a survey, *arXiv:2104.00950*.

Rothman, Denis. 2020, *Hands-on explainable AI (XAI) with Python: interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps*, Packt Publishing Ltd.

Schapire, Robert E. 1990, The strength of weak learnability, *Machine Learning*, vol. 5, no. 2, pp. 197–227.

Scikit-learn. 2022a, *Decision tree regressor*. Available: https://scikit-learn.org/stable/modules/tree.html.

Scikit-learn. 2022b, *K-nearest neighbors regressor*. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.

Scikit-learn. 2022c, *Linear regression*. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html.

Scikit-learn. 2022d, *Permutation feature importance*. Available: https://scikit-learn.org/stable/modules/permutation_importance.html.

Scikit-learn. 2022e, *Random forest regressor*. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.

Sezer, Omer Berat; Gudelek, Mehmet Ugur & Ozbayoglu, Ahmet Murat. 2020, Financial time series forecasting with deep learning: a systematic literature review: 2005–2019, *Applied Soft Computing*, vol. 90, , p. 106181.

Shapley, Lloyd S. 1953, A value for n-person games, *Contributions to the Theory of Games 2.28*, pp. 307–317.

Sutton, Richard S. & Barto, Andrew G. 2018, *Reinforcement learning: an introduction*, MIT Press.

Wachter, Sandra; Mittelstadt, Brent & Russell, Chris. 2017, Counterfactual explanations without opening the black box: automated decisions and the GDPR, *Harvard Journal of Law & Technology*, vol. 31, no. 2.