UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Lucas Rabelo de Araujo Morais

Bitcoin and Cryptocurrencies: COMTE-LEFTIST Hybrid Explanations and Time-Series Classification.

Recife

2025

Lucas Rabelo de Araujo Morais

Bitcoin and Cryptocurrencies: COMTE-LEFTIST Hybrid Explanations and Time-Series Classification.

Dissertação de mestrado apresentada ao Programa de Pós-Graduação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Inteligência Computacional

Orientador (a): Teresa Bernarda Ludermir

Recife

2025

.Catalogação de Publicação na Fonte. UFPE - Biblioteca Central

Morais, Lucas Rabelo de Araujo.
Bitcoin and Cryptocurrencies: COMTE-LEFTIST Hybrid Explanations and Time-Series Classification / Lucas Rabelo de Araujo Morais. - Recife, 2025.
80f.: il.

Dissertação (Mestrado) - Universidade Federal de Pernambuco, Centro de Informática, Programa de Pós-Graduação em Ciência da Computação, 2025.
Orientação: Teresa Bernarda Ludermir.
Inclui referências.

1. Explainable AI; 2. Time-Series Classification; 3. Hybrid Explanations. I. Ludermir, Teresa Bernarda. II. Título.

UFPE-Biblioteca Central

Lucas Rabelo de Araujo Morais

“Bitcoin and Cryptocurrencies: COMTE-LEFTIST Hybrid Explanations and Time-Series Classification”

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Aprovado em: 17/07/2025.

BANCA EXAMINADORA

_____

Profa. Dra. Teresa Bernarda Ludermir
Centro de Informática / UFPE
(**orientadora**)

_____

Prof. Dr. Sergio Fernandovitch Chevtchenko
MARCS Institute / University of Western Sydney

_____

Profa. Dra. Gecynalda Soares da Silva Gomes
Departamento de Estatística /UFBA

To my beloved and caring father, Arquimedes Rabelo de Morais (1951-2024), who worked hard for my education.

ACKNOWLEDGEMENTS

To my father, an intelligent man who came from a small town in the interior of Brazil, traveled through South America, lived in the largest city in the country, and always valued and invested in my education. Even during difficult times in his final days, he never stopped praying the rosary, never lost his faith in God, and always woke up giving thanks for a new day. Without his efforts and prayers, this work would not have been possible.

To my mother, a noble woman who took care of our family through the most challenging times we faced, always encouraging me to aim higher in both academic and professional life.

To my sister and brother, two strong individuals who also motivated and supported me during difficult moments.

To all my family, on both my father's and mother's sides, who helped us through hard times and helped us keep hope alive. Their support was crucial to the completion of this work.

To my academic advisor, Dr. Teresa Bernarda Ludermir, for her professionalism, support and for motivating me to explore new academic horizons and aim high in my academic pursuits.

To the examining board, Dr. Sergio Fernandovitch Chevtchenko and Dr. Gecynalda Soares da Silva Gomes, for their valuable contributions to this work.

To all those who, knowingly or unknowingly, contributed to my academic journey and made this work possible.

RESUMO

A "Corrida Global pela IA" incentivou uma estratégia conhecida como "IA para a sociedade". Um dos principais resultados dessa estratégia foi o Regulamento Geral de Proteção de Dados (GDPR), uma regulamentação europeia aplicada em 28 de maio de 2018, que estabeleceu o "direito à explicação". Essa regulamentação contribuiu significativamente para o avanço da Inteligência Artificial Explicável (XAI). Em meio a essas inovações tecnológicas, o mercado de ativos digitais, conhecidos como criptomoedas, se beneficiaram de pesquisas sobre sistemas de trade com Inteligência Artificial (IA) e Aprendizado de Máquina (AM). No entanto, esses sistemas frequentemente dependem de modelos caixa-preta, tornando a explicabilidade um aspecto crucial. Nesse contexto, este trabalho aplica modelos de Aprendizado de Máquina especificamente desenhados para Classificação de Séries Temporais (CST) e propõe um novo método híbrido que fornece explicações baseadas em séries temporais. Após a coleta de dados de Bitcoin e outras criptomoedas de uma exchange, os dados são processados e treinados utilizando modelos de AM tabular, modelos de AM para séries temporais e modelos de Aprendizado Profundo (AP). O estudo avalia incerteza, performance dos modelos e a explicabilidade por meio de um modelo híbrido de explicabilidade, que combina COMTE (método contrafactual de explicação para CST) e LEFTIST (método baseado em ondaletas que fornece a importância de cada janela de tempo). Os resultados mostram que o modelo de CST MRSQM (Multiple Representations Sequence Miner) obteve um desempenho robusto, enquanto os modelos AM tabular não apresentaram diferenças significativas em relação aos modelos de CST. No entanto, os modelos de AP tiveram um desempenho fraco, especialmente no segundo experimento. A análise de incerteza revelou diferenças notáveis na estimativa de incerteza dentre os modelos, e o modelo híbrido de explicabilidade COMTE-LEFTIST conseguiu fornecer explicações híbridas com sucesso. O modelo híbrido teve um desempenho particularmente bom no primeiro experimento, que focou em séries temporais univariadas, já no segundo experimento, envolvendo múltiplas séries temporais em formato tabular, apresentou desafios adicionais. Em conclusão, este trabalho está entre os primeiros a aplicar métodos de CST ao Bitcoin e a diferentes criptomoedas, além de propor um método híbrido de explicação para CST, incentivando pesquisas e desenvolvimentos adicionais na área.

Palavras-chaves: IA Explicável. Classificação de Séries Temporais. COMTE. LEFTIST. Hybrid XAI.

ABSTRACT

The "Global Race For AI" has driven the pursuit of a strategy known as "AI for society". One of the key outcomes of this strategy was the General Data Protection Regulation (GDPR), an European regulation enforced on May 28, 2018, which established the "right to explanation". This regulation significantly contributed to the rise of Explainable AI (XAI). Amidst this wave of technological innovation, the market around digital assets, commonly known as the cryptocurrency market has benefited from research into Artificial Intelligence (AI) and Machine Learning (ML) based trading systems. However, these systems often rely on black-box models, making explainability crucial. In this context, this work applies Machine Learning models specifically designed for Time-Series Classification (TSC) and proposes a novel hybrid method that provides time-series-based explanations. After collecting Bitcoin and cryptocurrency data from a crypto exchange, the data is processed and trained using ML tabular models, ML TSC models, and Deep Learning (DL) models. The study evaluates uncertainty, performance, and explainability through a hybrid explainability model, which merges COMTE (a counterfactual TSC explanation method) and LEFTIST (a time-point-based method that provides feature importance for each timestep). The results show that the Multiple Representations Sequence Miner (MRSQM) TSC model achieved a strong performance, while ML tabular models did not differ significantly from TSC models. DL models, however, performed poorly, particularly in the second experiment. Uncertainty analysis revealed notable differences in uncertainty estimation, and the COMTE-LEFTIST hybrid explainability model successfully provided hybrid explanations. The hybrid model performed particularly well in the first experiment, which focused on univariate time-series data, while the second experiment, involving multiple time-series in a tabular format, presented additional challenges. In conclusion, this is among the first works to apply TSC methods to Bitcoin and other cryptocurrencies, while also proposing a novel hybrid explainability approach for TSC, encouraging further research and development in the field.

Keywords: Explainable AI. Time-Series Classification. COMTE. LEFTIST. Hybrid XAI.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ADA	Cardano
AI	Artificial Intelligence
ANN	Artificial Neural Network
AVAX	Avalanche
BCH	Bitcoin Cash
BiLSTM	Bidirectional Long Short-Term Memory
BTC	Bitcoin
CNN	Convolutional Neural Network
COMTE	Counterfactual Explanations for Multivariate Time Series
DARPA	Defense Advanced Research Projects Agency
DL	Deep Learning
ETH	Ethereum
EU	European Union
GB	Gradient Boosting
GDPR	General Data Protection Regulation
GRU	Gated Recurrent Unity
HCTSA	Highly Comparative Time-Series Analysis
KNN	K-Nearest Neighbors
LEFTIST	Agnostic Local Explanation for Time Series Classification
LIME	Local Interpretable Model-Agnostic Explanations
LSTM	Long Short-Term Memory
LTC	Litecoin
MCP	Maximum Class Probability
ML	Machine Learning
MLP	Multi Layer Perceptron

MRSQM	Multiple Representations Sequence Miner
RNN	Recurrent Neural Network
SAX	Symbolic Aggregate Approximation
SFA	Symbolic Fourier Approximation
SOL	Solana
SVC	Support Vector Classification
SVM	Support Vector Machine
TSC	Time-Series Classification
TSF	Time-Series Forest
UE	Uncertainty Estimation
UFPE	Federal University Of Pernambuco
USDT	Tether
XAI	Explainable AI
XGBoost	Extreme gradient boosting
XLM	Stellar
XRP	Ripple

LIST OF SYMBOLS

$\bigoplus$	Concatenation
λ	Lambda
$\in$	In
δ	Delta
Ω	Omega
μ	Mi
τ	Tau
ϕ	Phi
π	Pi

SUMMARY

1 INTRODUCTION

Machine Learning, Deep Learning, and Artificial Intelligence have become increasingly important fields and are now considered major assets that companies and nations strive to develop as of 2025. Reflecting this trend, the Joint Research Centre website of the European Commission, which is an official European Union (EU) platform refers to this current scenario as the "Global Race for AI" [1]. In this competition, China, the EU, and the United States are the primary players, each pursuing three competing AI strategies: AI for profit, AI for control, and AI for society. The EU, for instance, prioritizes the development of fair AI systems that are secure and ethical by design. A key example is the european regulation named General Data Protection Regulation (GDPR), which establishes the "right to explanation" (HOLZINGER et al., 2018), making black-box solutions such as Machine Learning (ML) and Deep Learning (DL) challenging to deploy in practice. Since its enforcement on May 28, 2018, the GDPR has significantly elevated the importance of Explainable AI (XAI), transforming it from a niche research area into a critical field of study.

In this context, as stated by Saranya and Subhashini (2023), the term "Explainable AI" was coined by the Defense Advanced Research Projects Agency (DARPA) agency. The right to explanation, as mandated by the GDPR, primarily focuses on providing justifications for a model's decisions and encompasses key aspects such as responsibility, transparency, and accountability. XAI has been applied across various domains, including finance, time-series analysis, computer vision, and healthcare. Its applications range from image classification and churn prediction models to recommendation systems and EEG signal classification. By ensuring the right to explanation, XAI fosters collaboration between AI developers, academia, and stakeholders, ultimately enhancing both model transparency and usability.

In parallel with the growing interest in XAI, the 2008 financial crisis led to the emergence of a new class of primarily digital assets, the first one introduced was the Bitcoin (NAKAMOTO, 2008). These digital assets, known as cryptocurrencies, have driven extensive research into Artificial Intelligence (AI) and ML-based trading systems. However, to foster trust in these systems, XAI can be applied. Nevertheless, adaptations may be necessary, as these systems typically involve dealing with time-series data.

[1] <https://joint-research-centre.ec.europa.eu/jrc-mission-statement-work-programme/facts4eufuture/artificial-intelligence-european-perspective/global-race-ai_en>, accessed on March 20, 2025

1.1 GOALS AND MOTIVATIONS

The main objective of this work is to evaluate whether explainability can be improved by combining two XAI algorithms specifically designed for Time-Series Classification (TSC) problems. This involves the development and assessment of a hybrid method that merges different explanatory strategies.

In addition to this central goal, the study sets out to achieve the following specific aims:

1. Assess the performance of machine learning algorithms tailored for TSC tasks. These models are tested on datasets with different structures, including univariate Bitcoin time series and tabular cryptocurrency data containing multiple time series.

2. Highlight the importance of interpretability and accountability in crypto recommendation systems. By exploring a trending academic topic, the study emphasizes the ethical challenges of deploying AI in high-risk environments.

3. Compare traditional ML and DL models for tabular data with models designed specifically for time-series classification. This comparison helps determine which approaches are most effective and interpretable for financial time-series tasks.

1.2 RELATED PUBLICATIONS

MORAIS, Lucas R. A. ; Teresa B. Ludermir . A Hybrid COMTE-LEFTIST Time-Series Explanation Method For a Time-series Classification Bitcoin Recommendation System. In: Latinx in AI @ NeurIPS 2024, 2024, Vancouver. Proceedings Latinx in AI @ NeurIPS 2024, 2024. v. 1. p. 1-8.

1.3 DISSERTATION STRUCTURE

Besides the first introductory chapter, the work is structured as follows:

- A literature review on relevant topics for Time-Series Classification and explainability is presented in Chapter 2. The models for Time-Series Classification are introduced in Section 2.1, while the taxonomy and key concepts of Explainable AI, along with traditional explainability methods such as SHAP and LIME, are described in Section 2.2.

Additionally, the concept of explainability is extended to Time-Series Classification tasks in Section 2.3. A brief review of the Deep Learning and Machine Learning models used in this work is provided in Section 2.4.

- Chapter 3 introduces Bitcoin, its development, and its significance, along with other cryptocurrencies analyzed in this work. The core concepts are presented in Section 3.1. Section 3.2 reviews the application of ML to cryptocurrency data and explores the role of explainability in this field.

- The hybrid method is introduced in Chapter 4, where Section 4.1 reviews existing hybrid methods, and Section 4.2 details the proposed approach.

- The materials and methods used in the experiments are described in Chapter 5. Key aspects, such as computing resources and libraries, are discussed in Section 5.1. ETL processes and model training are covered in Section 5.2, while models and the hybrid explanation algorithms are presented in Section 5.3.

- The experimental results are presented in Chapter 6. Section 6.1 evaluates the performance of the classification models, Section 6.2 analyzes the uncertainty over the Bitcoin data comparing estimation from different classes of models, Section 6.3 presents the outcomes of the hybrid explanation model, and Section 6.4 provides a critical evaluation of the findings.

- The work concludes in Chapter 7. The final remarks are presented in Section 7.1, limitations are discussed in Section 7.2, key findings are summarized in Section 7.3, and future research directions are outlined in Section 7.4.

2 THEORETICAL FRAMEWORK

In this chapter we describe the theoretical basis of the used models and concepts in this work, further details on used libraries, hyperparameters and arguments used in the functions are detailed in Chapter 5, furthermore this chapter is summarized as follows:

- In section 2.1 the main concepts of TSC are introduced together with some of its main fields of applications. Furthermore in subsection 2.1.1 ML models specifically designed to deal with TSC problems are briefly detailed.
- In section 2.2 topics relevant to Explainable AI are defined, important explainability models such as LIME and SHAP are described in subsection 2.2.1, whereas subsection 2.2.2 defines the concept of surrogate models and subsection 2.2.3 describes concepts relative to uncertainty estimation and fairness.
- In section 2.3 a brief overview is given on the growth of interest in the topic of Explainable Time-series Classification and its core concepts. Subsection 2.3.1 defines LEFTIST and subsection 2.3.2 defines COMTE, two explainability models tailored specifically for for TSC.
- In section 2.4 concepts relative to the ML and DL models used in this work are defined.

2.1 TIME-SERIES CLASSIFICATION

The widespread use of sensors to monitor human activity, coupled with the proliferation of internet connectivity, has resulted in an exponential increase in the daily collection of time-series data. Time-series data can be broadly defined as a sequence of ordered values, where each element in the series is associated with a timestamp t. If for a time-series s_i present in a set of time-series $S = \{s_1, ..., s_m\}$, each timestamp corresponds to only one value, the series is classified as univariate. Conversely, if two or more values are associated with each timestamp in s_i, the series is classified as multivariate.

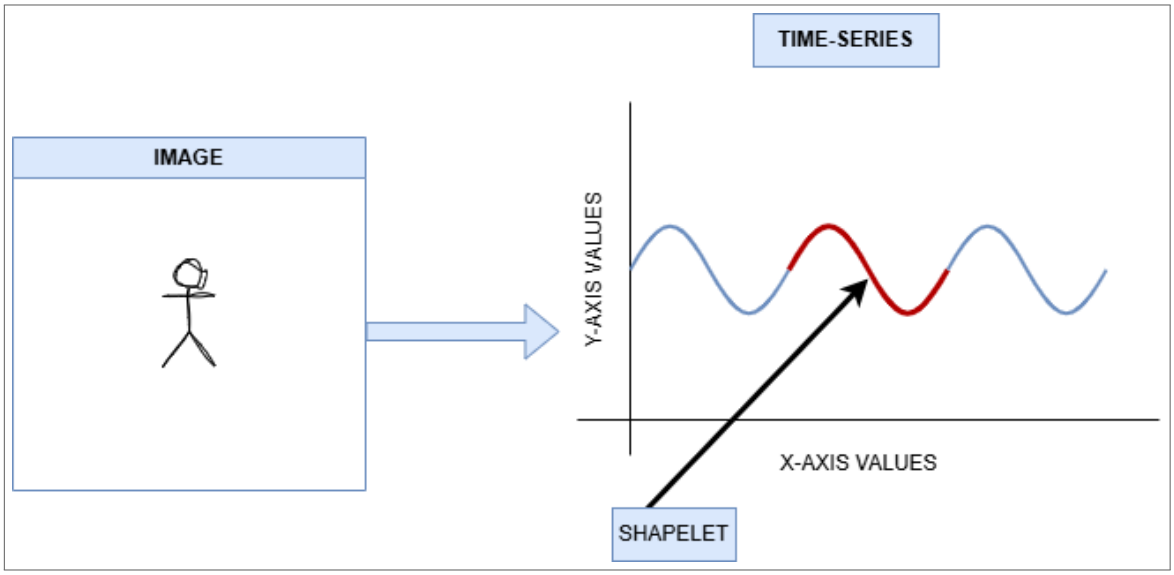
Traditionally, supervised machine learning commonly deals with assigning labels to observations in tabular or image data and training models to learn patterns that lead to final classification. In contrast, TSC involves observations comprising univariate or multivariate time-series,

which are labelled so that deep learning models, time-series-specific machine learning models, or tabular machine learning models can detect patterns within each series.

Indeed, Faouzi (2024) conducted a comprehensive review on TSC, highlighting applications ranging from food spectrograph analysis in chemometry to kinematic data used to improve surgical practices. The author explores a wide variety of methods and terminology used in the field and emphasizes that standard machine learning classification algorithms are not always well-suited for time-series data, as the order of values is a crucial aspect of time series. Similarly, Bagnall et al. (2017) support this view, noting that TSC algorithms can be categorized into different types based on the discriminatory features employed by each technique.

Regarding these methods, shapelet-based techniques are among the most traditional approaches in the field, first introduced by Ye and Keogh (2009). According to the authors, shapelets are defined as time points that are subsequences of a time series that are representative of a specific class, serving as key features to differentiate between classes. Figure 1 illustrates how an image, when transformed into a time-series, by the use of signal processing techniques, where the time series is labelled based on the object in the figure, contains a shapelet that maximizes the discriminative power for this class.

Figure 1 – Time-Series Shapelet Pipeline Representing The Image of a Person



Source: Author, 2024. The pipeline represents an image of the class "person" with its corresponding time series, where the shapelet is highlighted in red.

As noted by Bagnall et al. (2017), other types of algorithms for TSC include whole series-based methods, where two series are compared either as vectors or using a distance measure; interval-based methods, which involve feature selection based on a specific interval of the

time series; dictionary-based methods, which build classifiers by generating histograms from frequency counts of recurring patterns in time series; combination-based methods, which integrate two or more approaches; and model-based methods, which fit a generative model to each series and measure similarity between series through the similarity of their models. Model-based methods, in particular, perform well on time series of unequal lengths.

In this context Susto, Cenedese and Terzi (2018) divides these methodologies into two main branches: feature-based methods and distance-based methods. In summary, feature-based methods involve extracting features before classification, using signal statistics to identify the class to which a time series belongs. In contrast, distance-based methods bypass feature extraction and instead rely on distance metrics, as feature extraction can be time-consuming and may lead to information loss.

2.1.1 Time-Series Classification Specific Models

Standard ML techniques often fail to account for the temporal dependencies inherent in time-series data, leading to the development of specialized methods for TSC. This section provides an overview of the algorithms discussed in the Experiments and Methods chapter that were applied in this work. The dummy classifier, used as a baseline for prediction, is not included in this overview, as it will be defined exclusively in Chapter 5.

2.1.1.1 Support Vector Machines

The use of Kernel methods for Time Series Classification is highlighted by Faouzi (2024) as one of the most used algorithms in ML. For instance, Abade et al. (2015) employed Support Vector Machine (SVM) for TSC to classify satellite imagery time-series collected via sensors. While TSC algorithms require adaptations to handle time-series input, the most common used kernel functions themselves remain unchanged, Abade et al. (2015) also summarized the most commonly used kernel functions in Support Vector Classification (SVC), with the sigmoid kernel, used in this work, described by the following equation:

$$K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r), \quad \gamma > 0 \tag{2.1}$$

Where γ represents the smoothing parameter, and r denotes the bias term. In summary,

SVC methods are designed for pattern recognition, relying fundamentally on the choice of a kernel function. This function maps the training data into a hyperplane (a higher-dimensional space) that enables the optimal separation of the data into distinct classes.

2.1.1.2 CATCH22 - Canonical Time-series Characteristics

Subjective feature selection may leave out if a different feature could have achieved the optimal performance on a TSC task, to tackle this limitation, data-driven approaches for feature selection were developed. In this context, the Highly Comparative Time-Series Analysis (HCTSA) toolbox was designed to compare thousands of time-series features for feature selection, however given some limitations of this approach, such as being computationally expensive and requiring a matlab license to run, Lubba et al. (2019) developed CATCH22 which is the acronym for Canonical Time-series Characteristics. CATCH22 is a feature selector for time-series, that comprises a set of 22 high performance features, so that these features can be later fed into a classification model.

CATCH22 comprises a set of 22 features distilled from the 4791 features in HCTSA. These approaches were tested on a wide variety of time-series datasets. While HCTSA, with its full set of 4791 features, achieved a mean class-balanced accuracy of 77.2% across all tasks, CATCH22, with only 22 features, achieved a competitive 71.7%. To derive the 22 features, the authors reduced redundancy in HCTSA by applying hierarchical complete linkage clustering on the correlation distances of 710 high-performing features (selected from the initial set of 4791), using a distance threshold of $\gamma = 0.2$. This process resulted in the concise set of 22 features that define CATCH22. In addition to being interpretable, CATCH22 is approximately a thousand times faster than HCTSA and provides an efficient summary of time-series data. The specific features included in CATCH22 are detailed in Table 1.

Table 1 – CATCH22 Feature set

Feature Name	Description
DN_HistogramMode_5	Mode of z-scored distribution (5-bin histogram).
DN_HistogramMode_10	Mode of z-scored distribution (10-bin histogram).
SB_BinaryStats_mean_longstretch1	Longest period of consecutive values above the mean.
DN_OutlierInclude_p_001_mdrmd DN_OutlierInclude_n_001_mdrmd	Time intervals between successive extreme events above the mean.
CO_f1ecac	First $\frac{1}{e}$ crossing of autocorrelation function.
CO_FirstMin_ac	First minimum of autocorrelation function.
SP_Summaries_welch_rect_area_5	Total power in lowest fifth of frequencies in the Fourier power spectrum.
SP_Summaries_welch_rect_centroid	Centroid of the Fourier power spectrum.
FC_LocalSimple_mean3_stderr	Mean error from a rolling 3-sample mean forecasting.
CO_trev_1_num	Time-reversibility statistic.
CO_HistogramAMI_even_2_5	Automutual information.
IN_AutoMutualInfoStats_40_gaussian_fmmi	First minimum of the automutual information function.
MD_hrv_classic_pnn40	Proportion of successive differences.
SB_BinaryStats_diff_longstretch0	Longest period of successive incremental decreases.
SB_MotifThree_quantile_hh	Shannon entropy of two successive letters in equiprobable 3-letter symbolization.
FC_LocalSimple_mean1_tauresrat	Change in correlation length after iterative differencing.
CO_Embed2_Dist_tau_d_expfit_meandiff	Exponential fit to successive distances in 2-d embedding space.
SC_FluctAnal_2_dfa_50_1_2_logi_prop_r1	Proportion of slower timescale fluctuations that scale with DFA.
SC_FluctAnal_2_rsrangefit_50_1_logi_prop_r1	Proportion of slower timescale fluctuations that scale with linearly rescaled range fits.
SB_TransitionMatrix_3ac _sumdiagcov	Trace of covariance of transition matrix between symbols in 3-letter alphabet.
PD_PeriodicityWang_th0_01	Periodicity measure.

Source: Table adapted from (LUBBA et al., 2019).

2.1.1.3 Composable Time Series Forest

Decision Trees have historically been applied to interval features of time-series data for TSC. However, a significant limitation was the lack of robust measures to effectively distinguish between candidate splits. To address this issue, Deng et al. (2013) introduced a tree-ensemble

classifier named Time-Series Forest (TSF), one of the first TSC algorithms based on random forests (FAOUZI, 2024), which incorporates a novel measure called Entrance. This measure improves the identification of high-quality candidate splits, thereby enhancing the classifier's overall performance. Time-series Forests are composed of a collection of time-series trees. These trees utilize interval features such as the average, or the standard deviation value over a specified time interval (e.g., timestamps 10 to 30) to represent the data effectively and train the classification model.

For the root node, given a set of K features and $f_k(t_1, t_2)$ denoting the k_{th} interval feature (e.g., a metric such as the mean or standard deviation) calculated between t_1 and t_2, a candidate split S must satisfy the condition $f_k(t_1, t_2) \leq \tau$. Instances meeting this condition are directed to the left node, while those that do not are sent to the right node. For child nodes, where the optimal split criterion is represented as $S^* = f_k(t_1^*, t_2^*) \leq \tau^*$, the Entrance measure is used as the split criterion. This measure combines entropy gain with a distance metric to identify high-quality splits. The entropy component of the measure is defined as:

$$\text{Entropy} = -\sum_{c=1}^{C} \gamma_c \log \gamma_c \tag{2.2}$$

Where $\{\gamma_1, \ldots, \gamma_c\}$ represent the proportions of instances belonging to classes $\{1, \ldots, c\}$. The entropy gain, ΔEntropy, is defined as the difference between the entropy at the parent node and the weighted sum of the entropies at the child nodes, where the weights correspond to the proportions of instances assigned to each child node. Additionally, the $Margin$ measures the distance between a candidate threshold and its nearest feature value and is defined as:

$$Margin = \min_{n=1,2,\ldots,N} |f_k^n(t_1, t_2) - \tau| \tag{2.3}$$

Where $f_k^n(t_1, t_2)$ is the value of f for the n_{th} instance at the node. With $Margin$ and the entropy gain ΔEntropy it's possible to define the Entrance metric as:

$$Entrance = \triangle\text{Entropy} + \alpha \cdot \text{Margin} \tag{2.4}$$

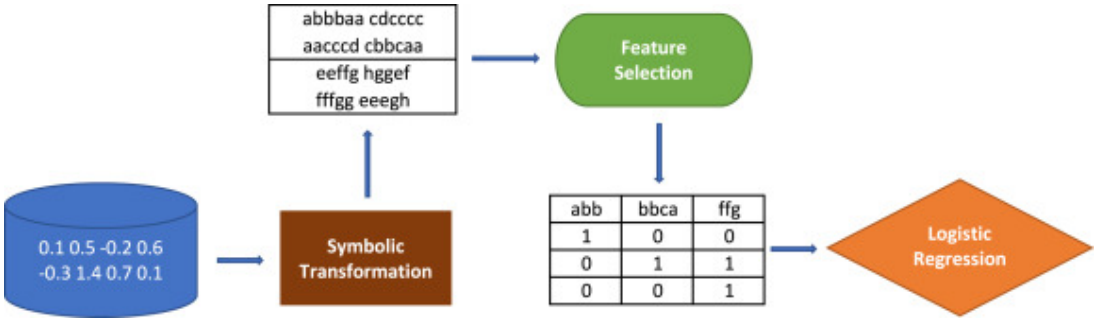
Where α is a parameter with a small value, that breaks ties that occur only from ΔEntropy. The TSF is simply an ensemble of time-series trees that use the metrics previously defined, and predicts the test instance by using majority class voting (e.g. if 2 trees predict class A and another predicts class B, then class A is the final prediction).

2.1.1.4 MRSQM

In TSC methods, a trade-off between computational cost and accuracy is frequently observed, particularly in models that leverage symbolic representation. To address this challenge, Nguyen and Ifrim (2023), Nguyen and Ifrim (2022) introduced MRSQM, a symbolic time-series classifier specifically designed to achieve high accuracy while minimizing computational expense. Symbolic representation techniques convert numeric time-series data into symbolic sequences. Among the most common methods in the literature are Symbolic Aggregate Approximation (SAX) and Symbolic Fourier Approximation (SFA). Both methods are based in three principles, they utilize a sliding window to extract segments of the time-series, approximate each segment with a vector of equal or smaller length, and discretize the approximation to produce a symbolic word.

The first step in MRSQM involves generating symbolic representations of the time-series using either SFA or SAX. Once the symbolic representations are obtained, random subsequences are sampled and converted into binary features: a value of 1 indicates the presence of a subsequence in the sample, while 0 indicates its absence. This process effectively transforms the time-series data into a tabular format, which is then used in the final step to train a classifier based on logistic regression. MRSQM has two variants: MRSQM-R and MRSQM-RS. In this work, only the MRSQM-R variant was applied. Figure 2 illustrates the three steps of MRSQM-R, as described in this paragraph.

Figure 2 – Workflow for the MRSQM time series classifier.



Source: (NGUYEN; IFRIM, 2022).

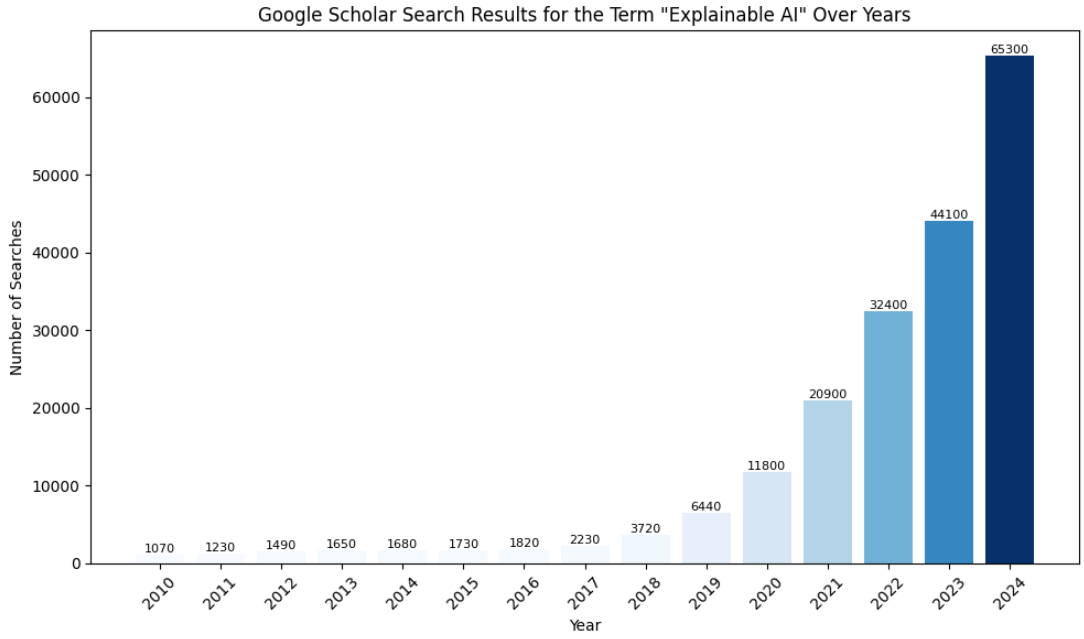
2.2 EXPLAINABLE AI

XAI serves as the foundation for Responsible AI, which encompasses a set of principles that must be upheld when deploying AI applications, including fairness, accountability, and transparency. In their comprehensive review, Arrieta et al. (2020) argue that any method aimed at reducing the complexity of a model or simplifying its outputs can be considered an XAI approach. The authors also provide taxonomies and terminologies for the field, emphasizing that terms such as "interpretability" and "explainability" are often used interchangeably, despite notable distinctions between them. In this work, we adopt the concepts outlined in their study; some of these will be defined throughout the text, while the main ones are defined as follows:

- **Explainability:** Refers to the explanation as an interface between humans and a decision-maker, often associated with post-hoc methods that clarify the reasoning behind a model's decisions.
- **Post-Hoc Explainability:** Focuses on providing understandable information about how a pre-existing model produces its predictions for any given input. Post-hoc techniques can be categorized as either global or local.
- **Local Explanation:** Focuses on explaining specific instances, providing insights into individual predictions.
- **Global Explanation:** Aims to explain the overall behavior of a model, offering insights into its functioning across the entire dataset.
- **Interpretability:** Describes the ability of a system to explain or convey its operations in a manner that is understandable to humans.
- **Fairness:** Aims to identify and mitigate bias in the data used to train models, ensuring the ethical and equitable application of AI algorithms.
- **Transparency:** A model is considered transparent if its structure and operations are inherently understandable without requiring additional explanations.
- **Accountability:** Closely tied to the auditability of algorithms and data, emphasizing the need to minimize and report negative impacts or limitations.

The field of XAI has been growing exponentially each year since 2010, as evidenced by the approximate results of the Google Scholar search queries data (Figure 3). These results show an approximate percentual growth of 6002.80% over this period. From 2023 to 2024 alone, the growth was approximately 48.07%, highlighting the hotness of the field and its recent importance in academia.

Figure 3 – Google Scholar Search Results for "Explainable AI" since 2010.



Source: Author, 2025. Data collected from Google Scholar Search Results on 2025-01-05.

In this context, Saranya and Subhashini (2023) reviewed recent advancements and outlined emerging trends within the field. The authors identified that the application of XAI to TSC was initially introduced in a study focused on analyzing clinical gait through time-series data, which is often gathered via sensors. Concerning post-hoc techniques for achieving explainability, they observed that LIME remains the most widely adopted approach for interpreting black-box models. Nevertheless, they highlighted SHAP as a more reliable method, capable of delivering both local and global explanations across diverse datasets.

2.2.1 Post-Hoc Explainability with LIME and SHAP Values

SHAP was introduced by Lundberg and Lee (2017), the authors proposed the use of Shapley Values as a unified measure of feature importance, and highlighted that its explanations closely align with human perception, offering a more intuitive and reliable understanding of

model outputs. Rooted in cooperative game theory, Shapley Values are defined based on the concept of marginal contribution. As summarized by Fryer, Strümke and Nguyen (2021), these values are estimated using a model-averaging approach. This procedure computes the weighted average of each feature's marginal contribution.

Classic Shapley value estimation, using Shapley regression values, is illustrated by equation 2.5. Here, F denotes the set of all features. To compute a feature's importance, a model $f_{S\cup\{i\}}$ is trained with the feature i included, while another model f_S is trained without it. The importance is determined by comparing their predictions: $f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)$, where $S \subseteq F$ and x_S are the input feature values in S.

$$\phi_i = \sum_{S\subseteq F\setminus\{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!}\left(f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)\right) \tag{2.5}$$

As noted by Lundberg and Lee (2017), the exact computation of Shapley values is computationally challenging. Instead, the authors propose approximations based on insights from additive feature attribution. They describe two model-agnostic approximation methods: Shapley Sampling Values, an established technique that applies sampling approximations to equation 2.5 by integrating samples from the training dataset to estimate the effect of removing a feature; and Kernel SHAP, a novel approach requiring fewer evaluations of the original model (but that is fairly good at approximating accuracy). To formulate Kernel SHAP, we need the concept of LIME, which is going to be introduced in the next paragraph. Additionally, we do not delve into model-specific Shapley value estimation methods in this work.

LIME was proposed by Ribeiro, Singh and Guestrin (2016) as a method to produce faithful local explanations for any classifier or regression model. The method is defined by equation 2.6, where G represents the class of interpretable models, and $g \in G$. Since not all g are inherently interpretable, $\Omega(g)$ measures model complexity (e.g., the depth of a Decision Tree). $\pi_x(z)$ quantifies the proximity between an instance z and x. For classification models, $f(x)$ denotes the predicted probability score. $\mathcal{L}(f, g, \pi_x)$ measures how poorly g approximates f around x. To ensure both interpretability and local fidelity, $\mathcal{L}(f, g, \pi_x)$ must be minimized, while $\Omega(g)$ remains low enough for human comprehension.

$$\xi(x) = \underset{g\in G}{\operatorname{argmin}}\left(\mathcal{L}(f, g, \pi_x) + \Omega(g)\right) \tag{2.6}$$

Kernel Shap is tied to the properties of Local Accuracy (equation 2.7), Missingness (equation 2.8) and Consistency (equation 2.9) of Additive Feature Attribution methods, where the

equations of these properties are defined such as:

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i \tag{2.7}$$

The property of Local Accuracy (equation 2.7) states that $f(x)$ represents the original model, $g(x')$ is the explanation model, and $\phi_0 = f(h_x(0))$ corresponds to the model output when the simplified inputs (x') are missing.

$$x_0^i = 0 \implies \phi_i = 0 \tag{2.8}$$

For the property of Missingness (equation 2.8), a constraint is put on features where $x'_i = 0$.

$$f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i) \tag{2.9}$$

The property of Consistency states that, given $f_x(z') = f(h_x(z'))$ and $z' \setminus i$ where $z'_i = 0$, for models f and f', if (equation 2.9) holds $\forall z' \in \{0,1\}^M$, then $\phi_i(f', x) \geq \phi_i(f, x)$.

Shapley Values are the only solution to the linear LIME equation (equation 2.6) that satisfies the properties of Local Accuracy, Missingness, and Consistency. The solution to equation 2.6 depends on the choice of parameters: L (loss function), $\pi_{x'}$ (weighting kernel), and Ω (regularization term). However, LIME selects these parameters heuristically, which is insufficient to recover Shapley Values. By applying the Shapley Kernel Theorem (Theorem 1), heuristic choices can be avoided, enabling the recovery of Shapley Values.

Theorem 1 (Shapley Kernel Theorem) *In order to guarantee the properties of Local Accuracy, Missingness and Consistency the parameters (parameters) need to be set as*

$$\Omega(g) = 0,$$

$$\pi_{x'}(z') = \frac{(M-1)}{(M \; choose \; |z'|)|z'|(M-|z'|)},$$

$$\mathcal{L}(f, g, \pi_{x'}) = \sum_{z' \in Z} \left[f(h_x(z')) - g(z')\right]^2 \pi_{x'}(z')$$

where $|z'|$ represents the number of non-zero elements in z' and the $choose$ function is the combinatorial choice function $\binom{M}{|z'|}$.

2.2.2 On the Use of Surrogate Models

According to Molnar (2022), in the field of engineering, if an outcome of interest is costly to compute, the theory of surrogate models suggests using a cheaper model as an alternative to complex computer simulations. For interpretable ML, the surrogate model g must be a ML model and not only provide accurate approximations to the black box model f but also be interpretable.

To train a surrogate model, a dataset X must be selected, then the predictions y of the black box model on X must be gathered, the surrogate model is supposed to be trained on X and y, producing predictions y', for the final step it's necessary to measure if the surrogate can replicate the underlying black box model, for this, metrics such as R^2 or accuracy can be used when comparing the predictions y and y', the higher the accuracy or the R^2 the better the approximation of the surrogate model and in consequence the explanation, in case the surrogate model is also interpretable.

The authors also highlight that it's important to bear in mind that there are some limitations regarding interpretable surrogate models. Some authors challenge the concept of intrinsically interpretable models, arguing that their use can be misleading and may provide only an illusion of interpretability when used as surrogates. In our experiments we have used agnostic explanation models, which use the concept of interpretable surrogate models (based on SHAP and LIME techniques). Additionally, we fitted a non-interpretable surrogate model as an intermediate step to apply explanation methods (Section 4.2).

2.2.3 On Uncertainty and Fairness

Explainable AI is the foundation for the topic of Responsible AI, and entails the concept of transparency. Transparency in explanations can be further enhanced by studying the uncertainty of predictions. According to Bhatt et al. (2021), uncertainty is defined as the lack of knowledge about a particular outcome, while confidence represents its opposite. Some authors, however, do not differentiate between confidence and uncertainty, maintaining a similar interpretation. Alonso (2024), in his dissertation, evaluated various uncertainty quantification methods and described uncertainty as the inverse of confidence, a high degree of uncertainty implies low confidence (CATTELAN; SILVA, 2022). Various metrics exist to convey uncertainty, with summary statistics of the predictive distribution being commonly employed.

Uncertainty can be categorized into two types: aleatoric (data uncertainty) and epistemic (model uncertainty). Aleatoric uncertainty reflects the inherent stochastic nature of an event and is considered irreducible, although it can be mitigated by altering the source of data. In contrast, epistemic uncertainty arises from insufficient data used to train the model and is reducible through additional data collection. Softmax output probabilities are commonly employed as an uncertainty metric and are regarded by Cattelan and Silva (2022) as the most natural method for uncertainty estimation. Gawlikowski et al. (2023) suggests using the Maximum Class Probability (MCP) for uncertainty estimation. Similarly, Judge (2021) proposes flipping the MCP of a probability vector (e.g., obtained from softmax outputs) $max(p)$ to quantify uncertainty. Another widely used approach for Uncertainty Estimation (UE) is the use of deep ensembles, which aggregate predictions from different models and improve UE.

It is important to note that, while softmax output probabilities may not directly quantify uncertainty, they can be interpreted as representing data uncertainty (aleatoric). However, they cannot be associated with epistemic uncertainty (or model uncertainty) (ALONSO, 2024) (GAWLIKOWSKI et al., 2023). In this study, we focus solely on data uncertainty resulted from softmax outputs, by comparing UE in different classes of ML models, as suggested by Holm, Wright and Augenstein (2023). Further details on the approach to studying it are provided in Chapter 5 under Subsection 5.3.2.

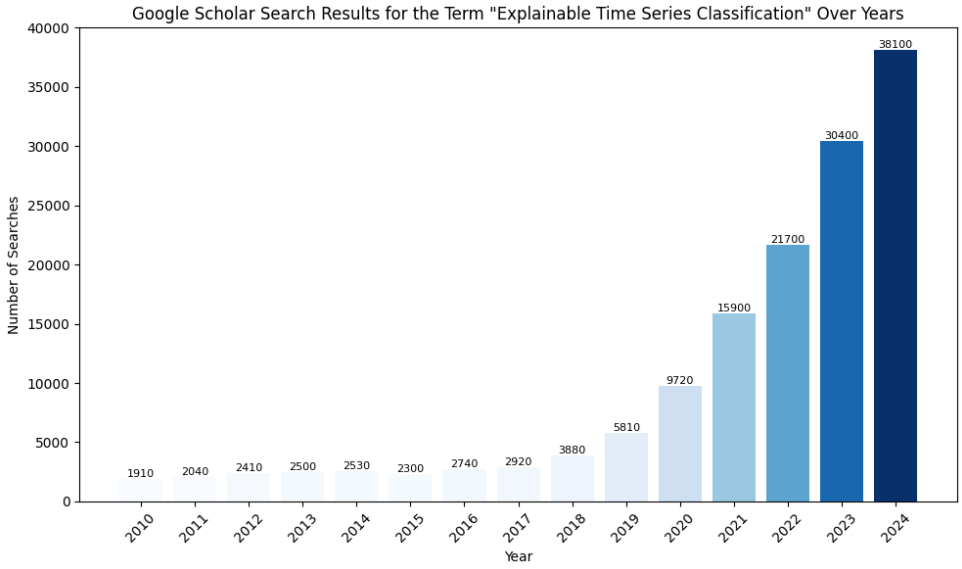
Fairness is a crucial aspect of Responsible AI. Identifying uncertainty can contribute to improving fairness in model decisions by revealing potential biases in data or models. In the experiments, in order to mitigate bias in the models, a stratified train-test split was employed. Minatel et al. (2023) demonstrated that even a simple stratification method by class and group can effectively incorporate fairness into ML models. Fairness can also be linked to the evaluation and comparison of different classification models, by ensuring that the same model-tuning is used (KWON et al., 2019). This perspective was considered during the formulation of the proposed experiments.

2.3 EXPLAINABLE AI FOR TIME-SERIES CLASSIFICATION

As of 2024, Explainability for Time Series Classification has emerged as a prominent research topic (Figure 4 - the search query returns approximate results) in academia. An exploratory Pearson correlation analysis between search results for “Explainable AI” and “Explainable Time Series Classification” (Figure 5) reveals a correlation coefficient of 0.99, indicating a strong

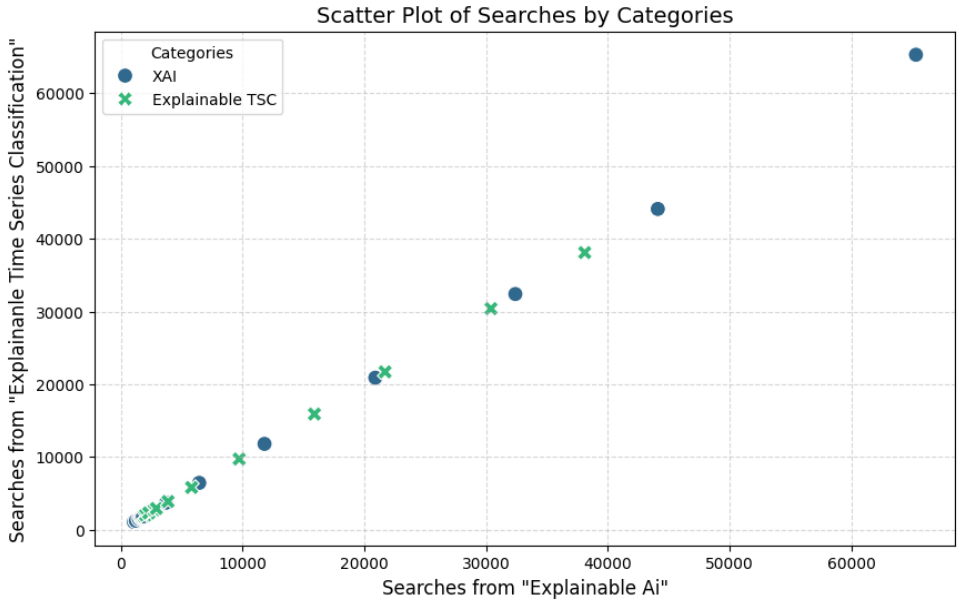
linear relationship between the two topics. The growing interest in Explainable AI (XAI) has naturally driven increased attention to Explainability in TSC, as TSC-specific XAI represents a subdomain within the broader field of XAI.

Figure 4 – Google Scholar Search Results for "Explainable Time Series Classification"since 2010.



Source: Author, 2025. Data collected from Google Scholar Search Results on 2025-01-05.

Figure 5 – Scatter Plot Comparing The Terms Explainable Ai and Explainable Time Series Classification



Source: Author, 2025. Data collected from Google Scholar Search Results on 2025-01-05.

Theissler et al. (2022) provided a comprehensive taxonomy and reviewed advancements in the field of Explainability for TSC. The authors identified three primary categories of explana-

tion methods: Time Point-based explanations, Subsequence-based explanations, and Instance-based explanations. These methods are defined as follows:

- **Time Point-based:** These explanations are analogous to feature importance, as they assign a score or weight to each time point in a time-series. There are two main approaches within this category: attribution-based explanations and attention-based explanations. Attribution-based methods attribute output predictions to input variables (e.g., LIME and SHAP, mainly used in computer vision tasks, which can be applied to time-series data). In contrast, attention-based methods leverage internal mechanisms of the TSC model to generate explanations.

- **Subsequence-based:** These explanations focus on identifying the most critical subparts of a time-series that contribute to its classification. Among subsequence-based methods, shapelets are the most widely used. Shapelets represent the most representative sequence of values within a time-series and are considered powerful discriminative tools for time-series data.

- **Instance-based:** These explanations leverage the entire time-series instance for providing explanations. The most common approach involves counterfactual explanations, where a counterfactual time-series x' illustrates how specific changes in the input time-series x lead to a different classification outcome. To ensure meaningful explanations, it is crucial that the difference between x and x' is minimal while maintaining the plausibility of x'.

While there are additional explanation methods that do not fall into these categories, this work does not explore them. For time-series explanations, we focused on Agnostic Local Explanation for Time Series Classification (LEFTIST), a model-agnostic local explainer that leverages both SHAP and LIME to determine the importance of time-series segments. This method fits both in the Time Point-based and Subsequence-based categories. Additionally, we employed Counterfactual Explanations for Multivariate Time Series (COMTE), a technique that utilizes counterfactual explanations derived from time-series instances in the training set.

2.3.1 LEFTIST

LEFTIST was introduced as the first model-agnostic local explainer specifically designed for time-series classification (GUILLEMÉ et al., 2019). It draws inspiration from the idea that the shapelets of a time-series often represent behaviors that are intuitive and easily interpretable for humans. The interpretable components of a time-series t can be divided into segmentations $S(t)$, defined as $S(t) = \{S_1^t, ..., S_m^t\}$, where:

$$S_1^t = \langle (t_{s_1}, v_1), ..., (t_{s_{t^1}}, v_{t^1}) \rangle, \ldots, S_m^t = \langle (t_{s_{t^{m-1}+1}}, v_{t^{m-1}+1}), ..., (t_{s_p}, v_p) \rangle \tag{2.10}$$

Here, each pair (t_i, v_i) represents a timestamp t_i and its corresponding value v_i, with $i \in [1, p]$. Using the concatenation operator $\bigoplus$, the time-series t can be expressed as $t = \bigoplus_{i=1}^{m} S_i^t$, meaning that t comprises all segments S. As such, t can be equivalently represented as $m_t = (1, ..., 1)$. Given m_t^j as the j^{th} neighbor of m_t, it represents cases where some segments are omitted by flipping an arbitrary number of ones to zeros. The proxy classifier for the black-box model f is expressed as:

$$g_t(m_j^t) = \phi_0 + \phi_1 m_t^{j,1} + \cdots + \phi_m m_t^{j,m} \tag{2.11}$$

Where $m_j^t = (m_t^{j,1}, ..., m_t^{j,m})^T$, $m_{j,i}^t \in \{0, 1\}$, and the coefficients (importance of each interpretable component) $\phi_i \in R$ are learnt by least squares. To construct this proxy model, which incorporates a learning process utilizing LIME or SHAP, a function h_t is defined (as shown in equation 2.12). This function serves as a mechanism to compute the class probabilities of the neighbors of the explained instance. It achieves this by mapping the masks m_t back to the original data space, generating a new training set, and then calculating the class probabilities, which are subsequently fed into the local model g_t, which is used to build the local explanation.

$$h_t(m_j^t) = \bigoplus_{i=1}^{m} \begin{cases} S_i^t & \text{if } m_t^{j,i} = 1 \\ \text{transform}_t(S_i^t) & \text{if } m_t^{j,i} = 0 \end{cases} \tag{2.12}$$

The developers of LEFTIST have proposed three main transform functions to be used along with it:

- **Linear Interpolation:** $\text{transform}_t(S_t^i) = \langle (t_{s_i}, d_{t,i}(t_{s_i})), ..., (t_{s_{t^i}}, d_{t,i}(t_{s_{t^i}})) \rangle$, where $d_{t,i}(x) = a_{t,i}x + b_{t,i}$ is defined as a line that passes by $(t_{s_{t^{i-1}}}, v_{t^{i-1}})$ and $(t_{s_{t^{i+1}}}, v_{t^{i+1}})$

- **Constant:** transform$_t(S_t^i) = \langle (t_{s_i}, d_{t,i}(t_{s_i})), ..., (t_{s_{ti}}, d_{t,i}(t_{s_{ti}})) \rangle$, where $d_{t,i}(x) = constant$, the constant may be computed by the time-series average or can be a parameter.
- **Random BackGround:** transform$_t(S_t^i) = S_{tr}^i$, where tr is a random time-series that belongs to a set of data BGS (BackGround Set) that needs to be available.

2.3.2 COMTE

COMTE, developed by Ates et al. (2021), is regarded by its authors as the first counterfactual explanation method designed specifically for multivariate time-series, though it also supports univariate time-series when the number of features m equals 1. The method involves selecting time-series instances from the training set that are highly similar to the time-series sample under investigation and using them to generate different classification outcomes. Counterfactual explanations have a wide range of applications, including their use in dashboards and for extracting knowledge about system behavior. While some methods rely on synthetic data for counterfactual explanations, COMTE enhances the meaningfulness and reliability of its explanations by exclusively utilizing data from the training set.

The first step took by the developers of COMTE, was to define the problem of counterfactual explanations for multivariate time-series, in which given a probability $f_c(x)$ for the class $c \in [1, k]$, where k represents the total number of classes, the optimal counterfactual explanation x' is a modified sample derived from the test sample x_{test}, where the difference between x' and x_{test} must be minimized while simultaneously maximizing $f_c(x')$. A distractor sample x_{dist}, selected from the training set, is used in this process to construct the optimal explanation. The distractor is chosen to minimize equation 2.14, where a tuning parameter λ, an identity matrix I_m, and a binary diagonal matrix A are defined. The rule governing $A_{j,j}$ is expressed as follows:

$$A_{j,j} = \begin{cases} 1 & \text{if variable } j \text{ of } x_{test} \text{ is replaced by the corresponding value in } x_{dist}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.13}$$

The distractor x_{dist} must minimize the following equation:

$$L(f, c, A, x') = (1 - f_c(x'))^2 + \lambda \|A\|_1 \tag{2.14}$$

x' is given by:

$$x' = (I_m - A)x_{\text{test}} + Ax_{\text{dist}} \tag{2.15}$$

However, due to its complexity, the authors adopted a heuristic approach for COMTE, modifying the loss function in equation 2.14 as follows:

$$L(f, c, A, x') = \left((\tau - f_c(x'))^+\right)^2 + \lambda \left(\|A\|_1 - \delta\right)^+ \tag{2.16}$$

In equation 2.16, τ represents the target probability, and δ is the threshold below which reducing the number of variables does not enhance explanations. The term $x^+ = \max(0, x)$ corresponds to the ReLU function, which penalizes explanations falling below δ. After selecting the candidate x_{dist} samples, the optimal choice is determined as the one whose A matrix yields the lowest loss value. The authors proposed finding these matrices using either a greedy algorithm or a random-restart hill-climbing method. Further details on the implementation of the algorithms can be found in the original work (ATES et al., 2021).

2.4 DEEP LEARNING AND TABULAR MACHINE LEARNING MODELS

Tabular ML and DL models can also be employed for TSC, although criticism has been made for standard/tabular ML methods (FAOUZI, 2024). There exists extensive literature on standard ML methods, and detailed descriptions of the methods utilized in this work can be found in the `scikit-learn`[1] and xgboost[2] libraries. Additionally, Sarker (2021b) provided a comprehensive review summarizing the primary techniques employed in ML. Based on the author's review, the methods applied in this work are summarized below:

- **SVM:** Builds a hyperplane that is used for either classification or regression tasks, its behavior depends on the choice of a kernel function.
- **K-Nearest Neighbors (KNN):** The model classifies data points based on a similarity measure, typically a distance metric. Classification is determined through a majority vote among the k nearest neighbors of each data point.

[1] <https://scikit-learn.org/stable/>, accessed on January 27, 2025
[2] <https://xgboost.readthedocs.io/en/stable/>, accessed on January 27, 2025

- **Random Forest:** Is an ensemble classification method that is built from several decision tree classifiers, the model uses majority voting or averages for the final classification result.
- **Extreme gradient boosting (XGBoost):** Is an ensemble classification algorithm that generates a final model based on a series of individual models, typically decision trees, the algorithm uses gradient to minimize a loss function and determine the best model, the algorithm is fast and efficient in handling large amounts of data.
- **Logistic Regression:** Is a parametric-based statistical model that uses a logistic function to estimate probabilities.

Deep Learning models have the capacity of learning complex non-linear patterns in data. In their comprehensive review of DL methods for TSC, Fawaz et al. (2019) highlighted that end-to-end deep learning architectures consistently achieve state-of-the-art performance in this domain. However, Nguyen and Ifrim (2023) pointed out that, deep learning approaches for TSC remain relatively recent and computationally intensive. This explains the current lack of dedicated libraries specifically designed for TSC in DL.

Neural networks are the core foundation for DL, many online resources are available on explaining the subject, the `pytorch`[3] documentation details it briefly while also explaining how to use the module in python, Sarker (2021a) conducted a review on DL methods covering the main architectures of neural networks, such as Multi Layer Perceptron (MLP), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM) and Gated Recurrent Unity (GRU). The MLP is the simplest architecture and is a feedforward network that contains an input layer, an output layer and may contain one or more hidden layers, using the notation in Fawaz et al. (2019) in the context of TSC non-linearity of a MLP can be expressed as:

$$A_{l_i} = f(\omega_{l_i} * \mathbf{X} + b) \tag{2.17}$$

Where ω_{l_i} represents the weights, b is the bias term, and A_{l_i} denotes the activation of the neurons in layer l_i. The symbol $*$ denotes a matrix multiplication between the weights and the input X's. The activation function, which determines the neuron's output, can be ReLU, Tanh, Sigmoid, or Softmax. The number of neurons, layers, and iterations in the network

[3] <https://pytorch.org/tutorials/beginner/basics/buildmodel_tutorial.html>, accessed on January 29, 2025

are hyperparameters, while the weights are learned automatically using a backpropagation algorithm with optimization such as SGD or Adam to minimize a loss function.

Another widely used neural network architecture is the LSTM, a variant of Recurrent Neural Network (RNN) designed to address the vanishing gradient problem. LSTMs incorporate three specialized gates: an input gate that filters incoming information, a forget gate that retains relevant information while discarding unnecessary details, and an output gate that controls the final output. These mechanisms allow LSTMs to effectively capture long-term dependencies in sequential data. An extension of the LSTM is the Bidirectional Long Short-Term Memory (BiLSTM), which enhances classification performance by employing two LSTM layers—one processing inputs in the forward direction and the other in the backward direction. Alternatively, the GRU architecture operates similarly to LSTMs but employs only two gates a reset gate and an update gate, thus making it faster to compute.

Attention layers were deemed by the developers of the Transformer architecture (VASWANI et al., 2017) as extremely important in its development, since the model is entirely based on attention mechanisms. The Transformer fully relies on attention to capture global dependencies between input and output. The attention mechanism focuses on the most relevant parts of an input sequence. In this context, self-attention is a specific type of attention mechanism that computes representations of a sequence by considering dependencies across different positions and can yield more interpretability in models, making self-attention widely used in tasks such as reading comprehension, abstractive summarization, and sentence representation.

3 BITCOIN AND CRYPTOCURRENCIES

This chapter describes the core concepts of Bitcoin and other cryptocurrencies available in the market, as well as how ML and DL have been used to aid decision-making for traders and other types of investors in the financial market. In this context, it also explores how XAI has contributed to more transparent models and decisions, ensuring compliance with industry regulations and enhancing accountability. Furthermore, this chapter is structured as follows:

- In section 3.1 the main concepts of Bitcoin are introduced together with important descriptions of other cryptocurrencies available in the market.
- Section 3.2 discuss works that have applied classification models to assist in decision making with cryptocurrencies.
- Section 3.2.1 brings an overview of works that have used XAI in cryptocurrencies and Bitcoin.

3.1 OVERVIEW AND CONTEXTUALIZATION

Bitcoin was created by Nakamoto (2008), a pseudonym whose true identity remains unknown as of 2025. The main philosophy behind Bitcoin's development was to create an electronic payment system based on cryptographic proof rather than trust. This approach aimed to eliminate the need for third-party financial institutions, thereby reducing transaction costs associated with mediation.

However, a key challenge in digital currencies is the double-spending problem since it would not be possible to verify if the coin was spent twice. To address this, the paper proposed a peer-to-peer electronic cash system that maintains a public record of all transactions, known as the Blockchain, in which it is computationally impractical for attackers to alter past transactions. In practical terms, according to Morisse (2015), each node in the Bitcoin network verifies transactions. The node that successfully solves a cryptographic puzzle is rewarded with Bitcoins and records the transaction in the Blockchain. As highlighted by Hellani et al. (2018), while Bitcoin relies on blockchain technology for its existence, blockchain itself is independent from it.

After Bitcoin's growth, a new market emerged around blockchain, which led to the development of other cryptocurrencies like Dash and Ethereum. Morisse (2015) reviewed Bitcoin and other cryptocurrencies, highlighting their upsides such as the low inflationary risks and lower transaction costs, however, also tackling its suspicions such as being widely used for money laundering and illicit trade. A promising global market has emerged around cryptocurrencies, offering opportunities due to their fast and low-cost international transactions.

The following table describes the main cryptocurrencies used in the experiments available in the Experiments and Methods chapter, with descriptions from the Bitstamp crypto exchange market [1]:

Table 2 – Cryptocurrencies' Description

Crypto Name	Description
Bitcoin (BTC)	Bitcoin is the coin unit available in the Bitcoin blockchain. It is divided into millibitcoins (0.001 BTC), and satoshis (0.00000001 BTC).
Ethereum (ETH)	Ethereum is the first network based on blockchain that supported smart contracts, the coin of Etheureum network is labelled ETH (or Ether).
Litecoin (LTC)	Litecoin was created to facilitate smaller transactions that may not be economically viable in Bitcoin. It shares many similarities with Bitcoin, and its native unit is called LTC.
Ripple (XRP)	Ripple is a financial institution that provides cryptocurrency-based solutions to customers, primarily through use of the XRP digital asset.
Tether (USDT)	Tether manages multiple stablecoins (cryptocurrencies that are programmed to maintain a value approximately equal to another asset) tokens, the one used in this work is called USDT which is pegged to the US dollar.
Bitcoin Cash (BCH)	BCH is one of the largest cryptocurrencies on the market. It is the result of a hard fork from the original Bitcoin blockchain.
Solana (SOL)	SOL is the native cryptocurrency of Solana, a platform whose goal is to maximize transaction speeds through a novel computational mechanism called Proof of History.
Cardano (ADA)	Founded with a focus on sustainability, Cardano is a carbon neutral blockchain, thanks to its low emission mechanism called Ouroboros. Its native token is called ADA.
AVAX	Avalanche is a smart contract-capable platform built to maximize blockchain scalability, its native utility coin is called AVAX.
Stellar (XLM)	Stellar aims to be blockchain solution in nations where the majority of the population remains unbanked. If a party was sending USD to Germany, Stellar would adjust the amount from USD to XLM and then to EUR in the recipient's wallet.

Source: Author, 2025. Description collected from Bitstamp Crypto Definition Search Results on 2025-02-03.

3.2 WORKS ON BITCOIN AND CRYPTOCURRENCY CLASSIFICATION

Several studies have analyzed the use of classification approaches for Bitcoin and cryptocurrencies. However, comparisons should account for methodological differences and the specific outcome variable being predicted. Ranjan, Kayal and Saraf (2023) applied a machine learning approach to classify Bitcoin price increases and decreases using daily and 5-minute granularity

[1] <https://www.bitstamp.net/learn/crypto-definitions/>, accessed on Febrary 03, 2025

data. Their results indicated that logistic regression performed best for daily price predictions. Qian and Qi (2022) explored the application of machine learning models for Bitcoin price prediction and identified SVM, LSTM, and MLP as the most frequently used models, while most approaches focused on regression, some were designed for classification, with all SVM-based methods being used exclusively for classification.

Given the lack of a cryptocurrency prediction framework to forecast the short to medium and long-term price that considers instant volatility, Iqbal et al. (2024) proposed a framework based on regression and classification models to assist in Sell-or-HODL recommendations for Bitcoin. (HODL stands for "hold on for dear life", meaning investors intend to retain their cryptos for an extended period). To achieve this, the authors employed SVM, LSTM, and an Artificial Neural Network (ANN), their findings indicate that although it is possible to accurately predict the present BTC price, predicting price increases and decreases remains challenging. Regarding different cryptocurrencies, Kwon et al. (2019) compared Gradient Boosting (GB) and LSTM for TSC across various cryptocurrencies to classify price trends (up or down). Their results showed that LSTM performed best, though they also found BCH particularly difficult to predict. As of the writing of this work, at the best of knowledge, only a limited number of studies, such as Yamak et al. (2024), have applied TSC-specific models to Bitcoin and cryptocurrency data.

3.2.1 Explainability on Cryptocurrencies

Previous studies on Bitcoin and cryptocurrencies have incorporated explainability into various tasks. Gupta et al. (2023) used SHAP to identify the features contributing to the forecasting of SOL and ETH using a regression approach based on GRU, LSTM, Random Forest, and SVM. Motivated by the need for improved interpretability in cryptocurrency trading, Fior, Cagliero and Garza (2022) developed CryptoMLE, a dashboard that leverages SHAP to support decision-making. This tool assists cryptocurrency investors by enabling them to compare rules inferred by ML algorithms with domain knowledge. Additionally, recognizing the lack of explainability in portfolio management, Babaei, Giudici and Raffinetti (2022) utilized SHAP to develop an explainable cryptocurrency portfolio.

Due to strict regulations in the financial industry, it is essential to understand how ML and DL models make decisions, in this context, SHAP was employed by Morais (2022), the author aimed to use XAI to identify the key determinants for predicting BTC value, and to

the best of the author's knowledge, this work is deemed to be the first one to propose such an application for XAI in the context of cryptocurrencies. A common denominator between all of these studies is the use of SHAP. Although SHAP and LIME contribute to decision making, these methods do not handle univariate time series directly. Instead, they rely on additional features to represent temporal patterns, and as regarded by Ates et al. (2021) both LIME and SHAP assume that all of the features are independent variables, however time-series are a set of ordered values, often exhibiting temporal dependencies (CUOMO et al., 2023), hence these XAI algorithms may lack the power of capturing temporal complexity, and also, do not provide a time-series as an explanation. These limitations motivate the hybrid approach proposed in this dissertation, which, to the best of our knowledge, is the first study to apply the combined use of COMTE and LEFTIST to cryptocurrency time-series data. The proposed method will be presented in the following chapter.

4 THE PROPOSED APPROACH

This chapter presents the proposed explanation approach developed in this work. It may be the first hybrid explanation method that combines two TSC-specific, model-agnostic explanation techniques. Furthermore, this chapter is structured as follows:

- Section 4.1 discusses the application of hybrid methodologies in the field of XAI.
- Section 4.2 introduces COMTE-LEFTIST, the hybrid explanation model proposed in this work, which integrates counterfactual (instance-based) and time point-based explanations.

4.1 HYBRID EXPLANATIONS

Hybrid approaches have already been studied in the field of XAI, the subject was one of the topics in the work of Arrieta et al. (2020) that traces important concepts, taxonomies and applications in XAI, the authors highlight different types of hybrid explanations, such as the use of surrogate interpretable model such as using an interpretable surrogate model, like KNNs or decision trees, after applying a DL model, or enriching black-box models with knowledge derived from transparent ones. The work of Álvarez, Díaz and García (2024) has also reviewed hybrid XAI modelling, but focused on merging aspects of interpretable models together with black-box models, the authors realized that research in this field can lead to more transparency in ML systems.

However this work proposes a different view on hybrid XAI, which is similar to the one proposed by Tahir et al. (2024), who proposed a hybrid XAI framework that merges SHAP and LIME. This LIME-SHAP approach was aimed to be applied to autonomous vehicle systems, inasmuch the system also makes use of sensor data, which can often be represented as time-series data. Their framework has three modules, a perception, a decision-making and an explanation generation module, the latter one utilizes the LIME-SHAP, where LIME is used for low-risk scenarios and SHAP for higher-risk (e.g. higher speed), for the LIME-SHAP method, the initial feature importance is computed using LIME and estimates are refined by using SHAP, the following equation summarizes the hybrid framework:

$$\phi_t^{\text{Hybrid}} = \alpha \cdot \hat{f}_t^{\text{LIME}} + (1 - \alpha) \cdot \phi_t^{\text{SHAP}} \tag{4.1}$$

Where $\hat{f}_t^{\text{LIME}}$ is the LIME equation (equation2.6), ϕ_t^{SHAP} refers to the SHAP values that can be computed by using the principles set by Theorem 1, α is a a weighting factor that balances the contributions of LIME and SHAP.

The authors consider this hybrid method to be more consistent than LIME in estimating feature importance while leveraging SHAP to optimize computational cost. This approach serves as an inspiration for the method proposed in this work. Although initially applied to different datasets—LIME-SHAP for autonomous vehicles (which may include sensor data) and COMTE-LEFTIST for cryptocurrency data, they share similarities in concept and formulation. The next section describes the method proposed in this work.

4.2 COMTE-LEFTIST A HYBRID EXPLANATION METHOD FOR TIME-SERIES CLASSIFICATION

COMTE-LEFTIST is a hybrid XAI method tailored for TSC, it may be applied to sensor data (e.g. ECG), finances, longitudinal studies, etc. Summarizing the whole proccess, the LEFTIST and COMTE approaches are combined to emphasize the impact of the most important shapelets from the counterfactual explanation on class predictions. The basic idea behind the hybrid framework is that LEFTIST captures the most important time-windows of the time-series and COMTE is responsible for generating the counterfactual examples.

To generate explanations with COMTE-LEFTIST, a mask needs to be built, by using the positive LIME or SHAP values derived from the LEFTIST approach. This mask filters out only the shapelets that contribute positively to the class prediction, so that we can apply it to COMTE and find the most relevant shapelet for the counterfactual explanation. The process can be summarized by the following steps:

- Generate a mask of positive LEFTIST values, estimated using either LIME or SHAP, where Positive values are marked as `True` and Negative or zero values are marked as `False`.
- Apply the mask to the COMTE values. Positive values highlight the relevant shapelets in the counterfactual explanation, while non-positive values are discarded from the predicted time series.
- Concatenate the shapelets of predicted and counterfactual values found by the mask.

- Feed the concatenated values into the prediction model.

After applying these steps it's necessary to compute the effect of the counterfactual explanation on the prediction score, as represented by the following equation:

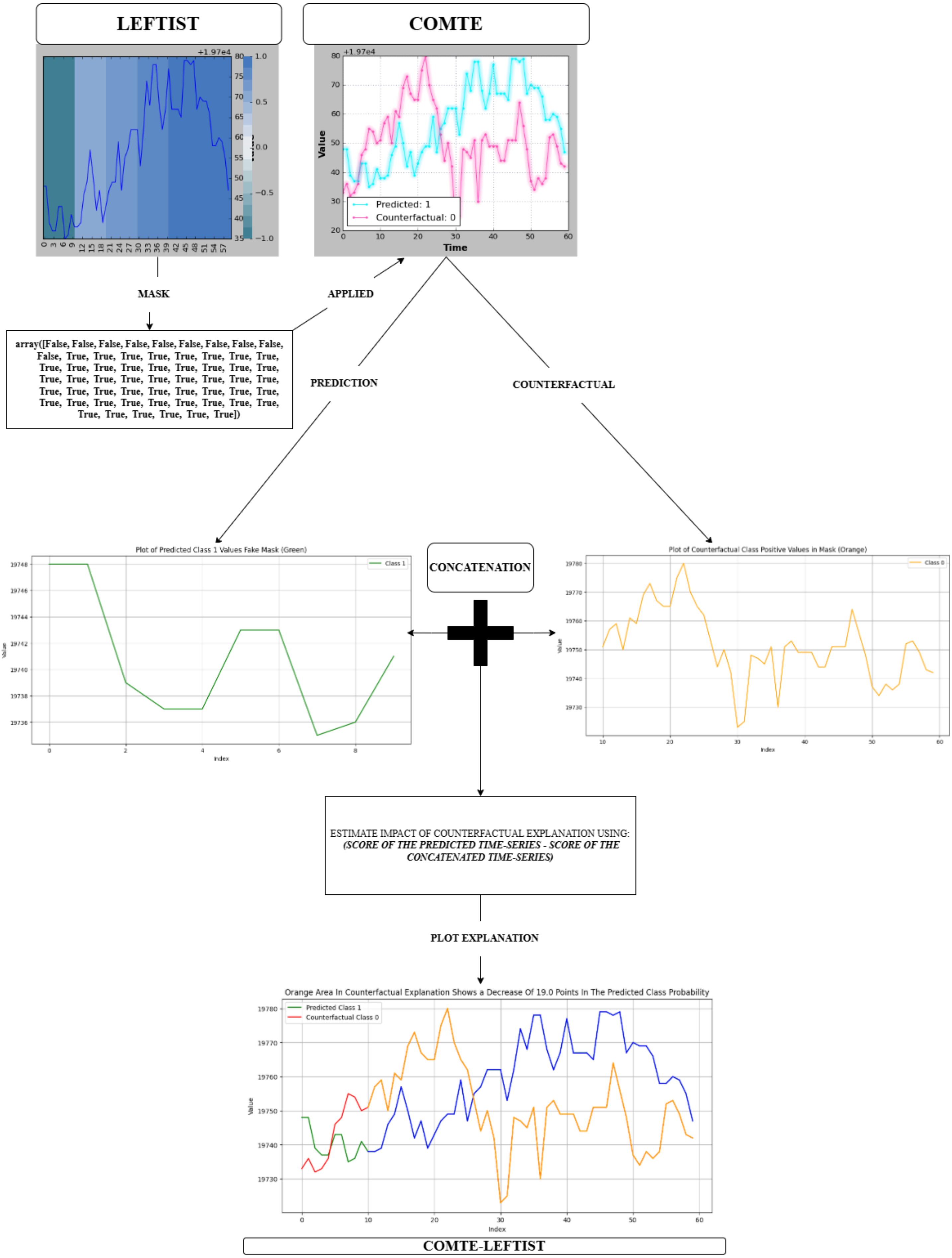
$$h(f, x, x') = f(x) - f\left(\bigoplus_{i=1}^{m} \begin{cases} x_i & \text{if } mask^i = False \\ x'_i & \text{if } mask^i = True \end{cases}\right) \tag{4.2}$$

Where mask $=$ (True, . . . , False) refers to the mask built by LEFTIST, $i = 1, \ldots, m$ refers to the position of the time-series values. The variable x represents the predicted time-series values, while x' corresponds to the counterfactual values, $\bigoplus$ is the concatenation operator, and f is the prediction function, which may or may not include standardized values for the predicted input, depending on the model's training configuration. The training and test dataset are standardized before being fed into the COMTE-LEFTIST hybrid model. However, the impact of standardizing the instance being predicted before feeding it to the explanation model, is observed and detailed in Chapter 6, as results vary depending on whether the instance is standardized or not, together with other aspects such as stability of explanations and the experiment being evaluated.

For regular ML models to work with COMTE-LEFTIST they need to be adapted, since LEFTIST requires that the training and instance arrays need to have a 3 dimensional shape, and COMTE also requires the instance to have the same shape. Given these specificities, the hybrid framework only works with ML TSC-specific models or DL models, ML models would need to be adapted to handle 3 dimensional shaped arrays. Furthermore when applying the COMTE-LEFTIST model on the cryptocurrencies dataset, the framework makes use of a surrogate model to communicate between the Tabular dataset with dummy variables and the framework.

The hybrid explanation is then visualized by plotting the most important shapelets of both the counterfactual and predicted series, each highlighted in different colors. This visualization shows the impact of the highlighted counterfactual shapelets on the final prediction score, offering a clearer understanding of how these counterfactual components influence the model's decision-making process. The steps to build COMTE-LEFTIST explanations can be summarized by the following figure:

Figure 6 – COMTE-LEFTIST Pipeline



Source: Author, 2025.

5 EXPERIMENTS AND METHODS

This chapter outlines the materials used in the experiments conducted in this study. It details key steps in data preparation, model training, uncertainty estimation, and explanation generation. The chapter is structured as follows:

- Section 5.1 provides an overview of the libraries used in this study and details the computing cluster, which played a crucial role in the experiments.
- Section 5.2 outlines the data acquisition process and the preprocessing steps applied before training (see subsection 5.2.1).
- Section 5.3 presents the classification models used in this study, categorizing them into tabular ML, TSC-ML, and DL. Subsection 5.3.1 details the hybrid XAI algorithms, highlighting key considerations for interpreting their results. Additionally, subsection 5.3.2 discusses the uncertainty estimation process and its evaluation in this work.

5.1 LIBRARIES AND COMPUTING RESOURCES

All experiments were conducted by using Jupyter Notebooks in Python (version 3.10.6), together with the Apuana Cluster provided by the Federal University Of Pernambuco (UFPE). The Cluster's architecture[1] contains 10 processing nodes, integrating 11 GPUs RTX3090, 5 GPUs A100 and 5TB of RAM. In the experiments, we allocated computational resources using the `salloc` command with the following parameters: `–mem=64G` to request 64GB of RAM, `-c 32` to allocate 32 CPU cores, `–gpus=2` to utilize two GPUs, and `-p short` to specify the short partition (No access was provided for the long partition since there was no need for it). The computations were often executed by specifying cluster nodes that were not being much used, ensuring efficient parallel processing and GPU acceleration for model training and evaluation.

Besides the common libraries used for handling tasks relating to data manipulation and visualization in Python, such as Pandas, Numpy, Matplotlib, etc. To train the ML tabular models the libraries used were: sklearn (Version 1.3.0), xgboost (Version 2.1.0). For the DL models pytorch (Version 2.4.1+cu121) was used. For ML models designed specifically for TSC, sktime (Version 0.30.2) developed by Löning et al. (2019) was used, the library provides

[1] <https://helpdesk.cin.ufpe.br/servicos/cluster-apuana>, accessed on February 15, 2025

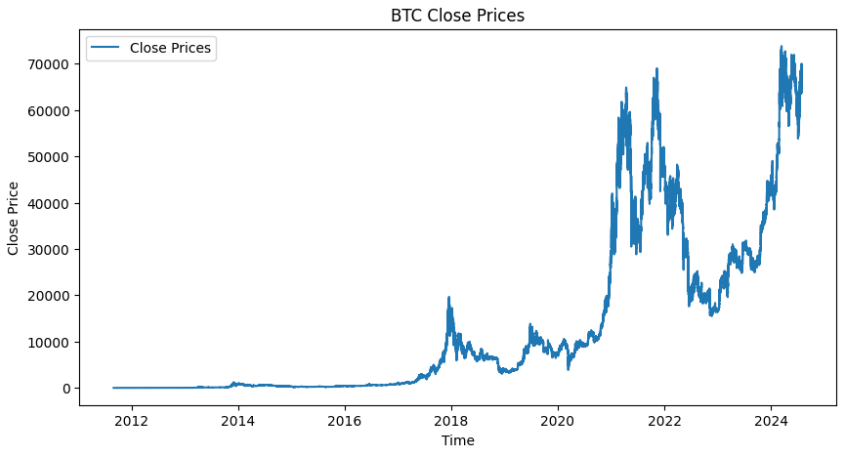
a unified interface, that consists of techniques used for feature extraction, robust regression or classification with the use of ML and DL on time-series. Another important library for TSC was the tsLearn (Version 0.6.3), introduced by Tavenard et al. (2020) specifically designed to implement advanced techniques, in this work the library was used to apply the time-series SVC model.

Regarding explainability for TSC, the TSInterpret library (Version 0.4.5) developed by Höllig, Kulbach and Thoma (2023) played a key role in this work. Its core philosophy is to provide a unified interface to state-of-the-art XAI algorithms. Among its various functions and explainability techniques, the library facilitates the application of COMTE and LEFTIST explanations. For the experiments, data from Bitcoin and various cryptocurrencies was sourced from the Bitstamp cryptocurrency exchange using the `ccxt` library (Version 4.3.58). The data processing and transformation steps will be discussed in the next section.

5.2 DATA PREPARATION

Two sets of experiments were conducted. The first set focused solely on Bitcoin data. For this experiment, closing price data with one-minute granularity (in USD), spanning from 2011-09-01 to 2024-08-01, was used to train the models and evaluate explainability. The Bitcoin data consists of a univariate time-series, that can be represented as an 1-d array (the same would be applied to every crypto in the experiments). Figure 7 illustrates the full extent of the Bitcoin data before it was processed into different time windows to align with the TSC perspective.

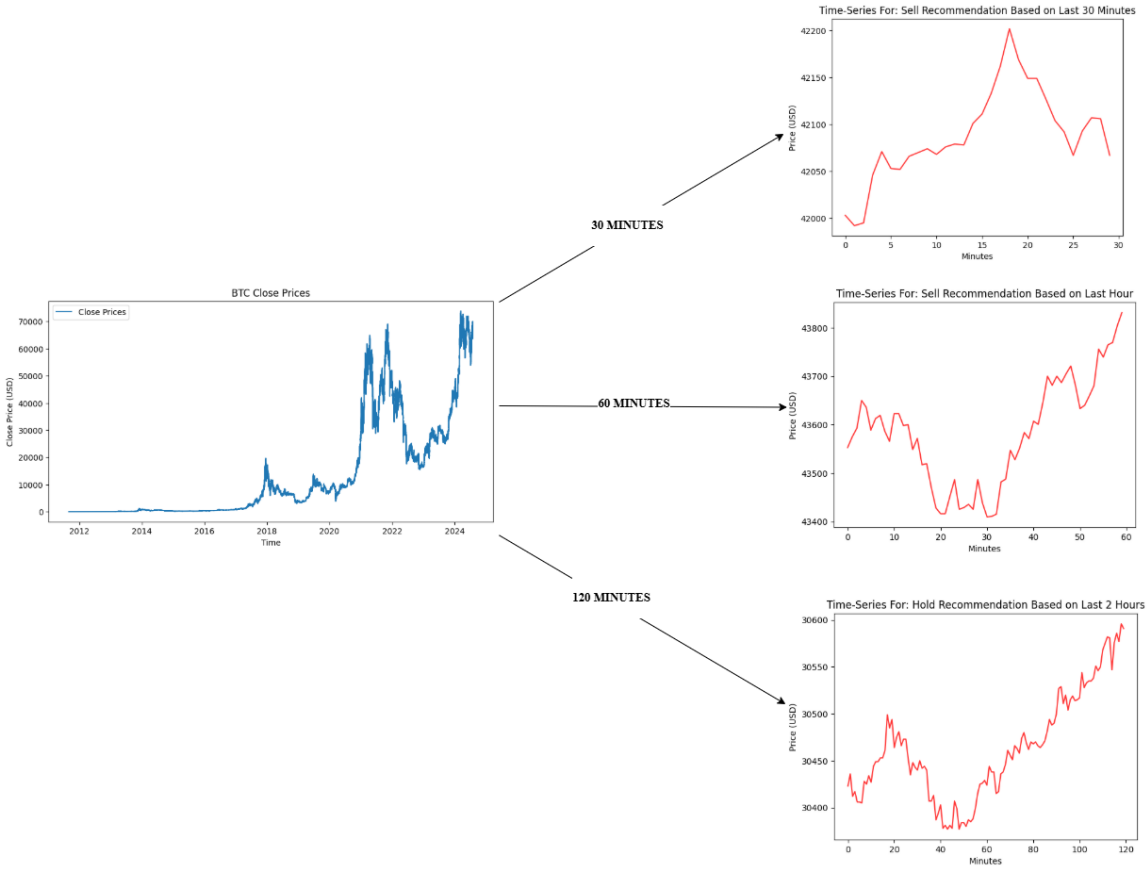
Figure 7 – Bitcoin Closing USD Prices 1-minute granularity



Source: Author, 2025.

To align with the TSC perspective, the Bitcoin data was segmented into three distinct time windows: 30 minutes, 1 hour, and 2 hours. Each time window corresponds to a time-series, where the number of timesteps matches the duration of the window in minutes (e.g., a 1-hour window consists of 60 timesteps). This process is illustrated in Figure 8. Each window was labeled as either "1" or "0", where "1" indicates a sell recommendation, meaning that if a certain amount of Bitcoin is purchased during the current time window, the average Bitcoin price is expected to rise in the next window. Conversely, "0" represents a hold recommendation, suggesting that the average Bitcoin price is expected to decline, making the market unfavorable for selling Bitcoin at that time.

Figure 8 – Bitcoin Time-series Data Transformation Pipeline



Source: Author, 2025.

The same labeling process and steps outlined in Figure 8 were applied to the cryptocurrency dataset, with one key difference. Instead of creating separate datasets for each cryptocurrency, all cryptocurrencies were combined into a single dataset and distinguished using dummy columns, meaning that in the end a tabular datasets for each time window containing time-series for each crypto were created. This approach aimed to accumulate sufficient data to train the DL models, with the expectation that the models would learn the distinct behavioral charac-

teristics of each cryptocurrency. The proportion of each cryptocurrency in the cryptocurrency datasets is as follows: BTC 24%, ETH 13%, LTC 13%, XRP 14%, USDT 5%, BCH 13%, SOL 3%, ADA 5%, AVAX 3%, and XLM 7%. The respective sizes of each dataset, including the ones used exclusively for BTC, are detailed in the following table:

Table 3 – Length of different Datasets

Crypto Name	Length
Bitcoin 30 Minutes	226,472
Bitcoin 1 Hour	113,235
Bitcoin 2 Hours	56,617
30 Minutes Cryptocurrencies	914,579
1 Hour Cryptocurrencies	457,286
2 Hour Cryptocurrencies	228,642

Source: Author, 2025.

5.2.1 Training Pipeline

The first step for both datasets was the train-test split, where 80% of the data was used for training and 20% for testing. The split was stratified by label, ensuring that the proportions of classes 0 and 1 remained approximately between 45%–55% in both the training and testing datasets across the 30-minute, 1-hour, and 2-hour windows. For the cryptocurrency dataset, the split was also stratified by cryptocurrency to ensure that each cryptocurrency was also represented in both the training and test sets, preventing the scenario where all data from a specific cryptocurrency would be confined to a single set, additionally, the stratification preserved label proportions around 50% across different cryptocurrencies. The `train_test_split` function from the `sklearn.model_selection` module was used for the split, with `random_state=42` applied to all experiments to ensure reproducibility, and that different models would be working with the same train-test split.

Data was also standardized, by using the MinMax standardization, which was fitted to the training data, and then used to transform the test data, in order to prevent information leakage (preventing the model to learn the minimum and maximum from the test set, which could potentially risk the reliability of the models). After the training process the ML, DL models, and agnostic explanation methods were applied to enhance interpretability and support decision-making. Furthermore, the Kruskal-wallis test, created by Kruskal and Wallis (1952), was used to verify if there was any statistically significant difference between different classes of models,

given that not much experimental data was available and normality assumptions couldn't be fulfilled, a statistical significance level of 5% was chosen.

5.3 MODELS AND EXPLANATIONS

This study evaluated three categories of models: ML tabular models, ML time-series models (designed specifically for TSC), and DL models. While each model was trained on different time windows, identical hyperparameters were used across all models to ensure a fair performance comparison, considering the distinct characteristics of the cryptocurrency and Bitcoin-only datasets. For the TSC-specific ML models implemented using the sktime and TsLearn libraries, most models were applied to both the cryptocurrency and Bitcoin datasets. The only exception was the time-series SVC, which was excluded from the cryptocurrency dataset due to excessive computation time, making it unfeasible given the available resources.

For the ML tabular models, computation time was excessive for the cryptocurrency dataset. As a result, instead of using SVC, a logistic regression model was adopted. Regarding DL approaches, all methods were applied in both experiments, but with different training configurations. After applying the standard pipeline described in subsection 5.2.1, a random seed of 2 was set using `torch.manual_seed`, `random.seed`, and `numpy.random.seed`. The train and test dataloaders used a batch size of 32, with training batches shuffled while test batches remained in order. In the two hour cryptocurrencies dataset, most DL models required dropping the last batch, except for the Fully Connected MLP. This was handled by using the parameter last_batch=True, which according to Pytorch's documentation[2] ignores the last batch when the dataset size is not divisible by the batch size. The following table details the parameters and arguments of the functions used in the classification models:

[2] <https://pytorch.org/docs/stable/data.html>, accessed on February 18, 2025

Table 4 – Summarization of Models

Model	Parameters	Experiment Applied
MRSQM	`MrSQM(strat='R', random_state=42)`	Bitcoin / Cryptocurrencies
Catch22	`Catch22Classifier(random_state=3)`	Bitcoin / Cryptocurrencies
Dummy Classifier	`DummyClassifier(strategy='prior', random_state=3)`	Bitcoin / Cryptocurrencies
Time-series SVM Classifier	`TimeSeriesSVC(kernel="sigmoid", gamma="auto", probability=True, random_state=42)`	Bitcoin
Composable Time-series Forest	`ComposableTimeSeriesForestClassifier(RocketClassifier(num_kernels=100), n_estimators=10, random_state=4)`	Bitcoin / Cryptocurrencies
KNN	`KNeighborsClassifier(n_neighbors=5)`	Bitcoin / Cryptocurrencies
SVM Classifier	`SVC(random_state=42, probability=True, kernel='sigmoid', gamma="auto")`	Bitcoin
Logistic Regression	`LogisticRegression(random_state=42, max_iter=1000)`	Cryptocurrencies
XGBoost Classifier	`XGBClassifier(objective='binary:logistic', random_state=42)`	Bitcoin / Cryptocurrencies
Random Forest Classifier	`RandomForestClassifier(random_state=5)`	Bitcoin / Cryptocurrencies
CNN-GRU with Attention	Conv1D; MaxPooling1D; GRU Layer; Attention Layer; GlobalAveragePooling1D; BatchNormalization; Output Layer	Bitcoin / Cryptocurrencies
CNN-LSTM	Conv1D Layer; AdaptiveMaxPooling1D; Flatten Layer; Fully Connected Layer; BatchNormalization; Output Layer	Bitcoin / Cryptocurrencies
Simple DNN	Flatten Layer; Fully Connected Layer; Fully Connected Layer	Bitcoin / Cryptocurrencies
BiLSTM	Bidirectional LSTM; BatchNormalization; Output Layer	Bitcoin / Cryptocurrencies

Source: Author, 2025.

About the DL models, it's also important to highlight that the same hyperparameters were used in both experiments, models were trained with 1000 epochs, the learning rate chosen was of 10^{-5} to be as close as possible to 0 to improve convergence in a local minima, following an strategy of early stopping based on the validation loss, meaning that, if the loss did not improve for 50 epochs given a Δ parameter, then the training had to stop.

5.3.1 Explanations

Hybrid explanations were generated using LEFTIST and COMTE, both available in the TSInterpret library. For LEFTIST, the chosen learning process was LIME, combined with the uniform transform (also known as the mean transform, similar to the constant transform described in subsection 2.3.1). Since the explanation process exhibits stochastic behavior, a random seed was set to ensure reproducibility. The same approach used for the DL models was applied here, with fixed random seeds of 2.

For the COMTE explanation, which is considered an instance-based method, brute-force computation (`brute`) was chosen as the optimization approach over the optimized calculation. When implementing the COMTE-LEFTIST technique, the selected argument values were combined with the default parameters of the respective functions. Additionally, when evaluating the impact of the most relevant shapelet in the counterfactual explanation on score probability, it is essential to distinguish between standardizing or not standardizing the instance under

evaluation. Standardization can influence the effect of the counterfactual component on score probability. For the Bitcoin experiment, this process is illustrated in algorithm 1.

Algorithm 1: COMTE-LEFTIST Explanation

```
Input: instance, model, trainX, predicted label, trainy, prob_estimation
Output: Counterfactual explanation and probability shift
Initialize LEFTIST explainer;
Get explanation for instance;
if prob_estimation in ['standardized', 'not-standardized'] then
    Standardize trainX;
    Predict probabilities;
    Initialize COMTECF explainer with trainX not standardized;
    Get counterfactual explanation;
    Compute counterfactual label;
    Identify positive indices in explanation;
    Modify instance with counterfactual values;
    if prob_estimation == 'standardized' then
        Apply standardization on instances before compute impact on probability;
    Compute impact on probability score;
Plot instance, counterfactual, and highlighted areas;
```

Source: Author, 2025.

For the cryptocurrency dataset, the following algorithm 2 incorporates an intermediate surrogate model to communicate with the explanation generation process, given the structure of data. Explanations with surrogate models are always standardized, as experimental results showed issues when estimating their impact on score probability.

Algorithm 2: COMTE-LEFTIST with Surrogate

Input: instance, model, trainX, predicted label, trainy, prob_estimation, surrogate

Output: Counterfactual explanation and probability shift

Initialize LEFTIST explainer;

Get explanation for instance;

if *prob_estimation == 'standardized'* ***or*** *surrogate == True* **then**

- Standardize trainX;
- Predict probabilities;
- Initialize COMTECF explainer with trainX not standardized;
- Get counterfactual explanation;
- Compute counterfactual label;
- Identify positive indices in explanation;
- Modify instance with counterfactual values;
- Apply standardization on instances before compute impact on probability;
- Compute impact on probability score;

else

- Standardize trainX;
- Predict probabilities;
- Initialize COMTECF explainer with trainX not standardized;
- Get counterfactual explanation;
- Modify instance with counterfactual values;
- Compute impact on probability score;

Plot instance, counterfactual, and highlighted areas;

Source: Author, 2025.

5.3.2 Uncertainty estimation

Although uncertainty estimation was not the primary focus of this work, it is important to address the topic, particularly in the context of responsible AI, which is associated with this work. Aleatoric uncertainty was assessed in the Bitcoin experiment to investigate potential differences in how various model classes (DL, tabular ML, and TSC-specific ML) generate estimates. Initially, aleatoric uncertainty was computed for each time window and analyzed within

each one using the Kruskal-Wallis test, followed by the Conover post-hoc test (CONOVER; IMAN, 1979), which is designed to compare different groups after a significant Kruskal-Wallis result. This procedure was also extended to examine whether uncertainty differed across the three model classes when considering all time windows collectively.

To calculate aleatoric uncertainty (or data uncertainty), the maximum score probability was flipped. Since the classification was binary, aleatoric uncertainty was estimated as 1 minus the maximum predicted probability, which is equivalent to the minimum class probability. Although there are tons of ways to estimate aleatoric uncertainty, this approach was chosen due to its simplicity, ease of computation, and intuitive interpretation. Given the exploratory nature of this analysis, this method was deemed sufficient for the study.

6 RESULTS

This chapter presents and discusses the results of the experiments conducted in this study. It covers key topics such as uncertainty estimation, model performance, and interpretability. The chapter is structured as follows:

- Section 6.1 presents the classification model results for both the Bitcoin experiment (subsection 6.1.1) and the Cryptocurrency experiment (subsection 6.1.2).
- Section 6.2 discusses the uncertainty estimation results for the Bitcoin experiment.
- The primary findings of the hybrid COMTE-LEFTIST explanation method are presented in Section 6.3.
- Section 6.4 reflects on the implications of the findings and addresses limitations related to the explanation method and uncertainty estimation.

6.1 MODEL RESULTS

In both experiments, the primary evaluation metric was accuracy. Precision was also considered, as it is important to assess whether the models exhibit higher error rates in sell or hold recommendations. First, the results of the Bitcoin experiment will be presented, followed by the results of the Cryptocurrency experiments.

6.1.1 Bitcoin Model Results

First and foremost given the results of the Kruskal-Wallis test, there is not enough evidence at the 5% level of statistical significance to reject the hypothesis that the three different model classes (Tabular-ML, TSC-ML and DL) are equal (P-value = 0.8386), the results imply that if the focus is mainly on accuracy of the models, it's important to test between different model classes before selecting one. However, these results should be taken carefully, given the non-parametric nature of the test, and the small amount of data regarding the results provided in the experiment. Under different circumstances results may differ, as it will be seen in Subsection 6.1.2.

Evaluating strictly the accuracy, for the DL models, the CNN-GRU with attention mechanism was deemed to be the best model in all time-windows. For ML tabular models the Random Forest achieved the best metric results in all time-windows, whereas KNN showed a sharp decline of 10.76% in accuracy from the 1 hour to 2 hour time-window. For the TSC-specific ML models the MRSQM had the best metrics, it also surpassed the accuracy values of models in different classes. The Dummy Classifier is the baseline model, since it simply classifies as the most proeminent class, the SVC models (both tabular and TSC-specific) weren't able to overcome the baseline values.

Table 5 – Prediction Accuracy for All Model Types In BTC Experiment

Model	30 minutes	1 Hour	2 Hours
Deep Learning			
Attention CNN-GRU	0.64	0.64	0.64
CNN-LSTM	0.64	0.63	0.63
MLP	0.56	0.56	0.58
BiLSTM	0.63	0.61	0.59
Machine Learning Tabular			
Random Forest	0.68	0.68	0.60
KNN	0.65	0.65	0.58
SVC (Tabular)	0.53	0.52	0.53
XGB	0.57	0.58	0.57
Time Series Specific			
Dummy Classifier	0.55	0.54	0.54
Catch22	0.64	0.63	0.64
MRSQM	**0.70**	**0.69**	**0.68**
SVC (Time Series)	0.48	0.48	0.51
TS Forest	0.66	0.66	0.66

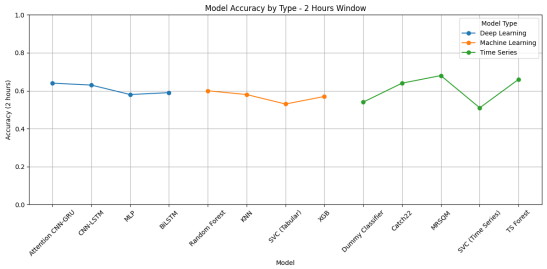
Source: Author, 2025.

The impact of each model can be visualized through the comparisons in Figure 9. In the two-hour time window, DL models appear to outperform tabular ML models, as the Attention CNN-GRU model (referred to in the figure as `attention_model`) and the CNN-LSTM model (denoted as cnn) achieved higher accuracy than both the Random Forest classifier (`forest_classifier`) and the KNN model. Additionally, the BiLSTM model (`lstm`) slightly outperformed the Random Forest classifier. When considering the three best-performing models in terms of accuracy across all three time windows, their improvements over the Dummy Classifier were as follows:

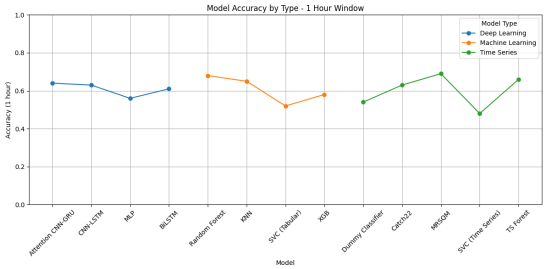
- In the 30-minute window, the accuracy increase was of 27.27% (MRSQM), 23.64% (Random Forest), and 16.36% (Attention CNN-GRU).
- In the one-hour window, the improvements were 27.78% (MRSQM), 25.93% (Random Forest), and 18.52% (Attention CNN-GRU).
- In the two-hour window, the gains were 25.93% (MRSQM), 11.11% (Random Forest), and 18.52% (Attention CNN-GRU).

Among these models, MRSQM consistently demonstrated the highest accuracy improvement over the baseline.

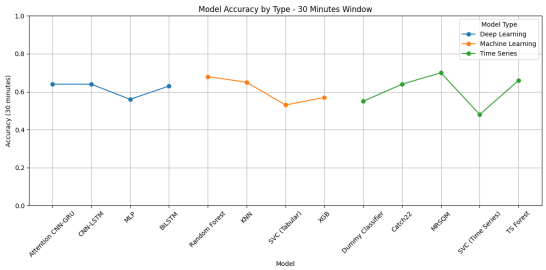
Figure 9 – Comparison of BTC accuracy at different time intervals



(a) Accuracy - 2 Hours



(b) Accuracy - 1 Hour



(c) Accuracy - 30 minutes

Source: Author, 2025.

When analyzing the precision of Selling (Class 1) and Holding (Class 0) recommendations, the MRSQM model, which achieved the highest accuracy, also demonstrated the highest precision for Class 1 across all time windows. This indicates that MRSQM is the most reliable model for issuing Sell recommendations, possibly making it the most effective at maximizing possible gains. The MLP model, on the other hand, achieved the highest precision for Class 0, meaning it was more accurate in identifying when holding is the optimal decision. While avoiding losses is generally more critical than missing out on gains (as an incorrect Sell recommendation is more detrimental than an incorrect Hold recommendation), a model that

exclusively recommends holding would not be practical. Given MRSQM's balanced and consistent performance in both Sell and Hold recommendations, together with its accuracy, it was deemed the best-performing model in this experiment.

Table 6 – Prediction Precision for All Model Types (Class 0) (Class 1)

Model	30 minutes	1 Hour	2 Hours
Deep Learning			
Attention CNN-GRU	(0.70)(0.62)	(0.73)(0.61)	(0.75)(0.61)
CNN-LSTM	(0.71)(0.62)	(0.72)(0.61)	(0.71)(0.61)
MLP	(0.64)(0.55)	(0.83)(0.55)	(0.76)(0.56)
BiLSTM	(0.65)(0.63)	(0.75)(0.59)	(0.62)(0.58)
Machine Learning Tabular			
Random Forest	(0.65)(0.71)	(0.66)(0.70)	(0.66)(0.68)
KNN	(0.62)(0.67)	(0.62)(0.66)	(0.62)(0.64)
SVC (Tabular)	(0.48)(0.57)	(0.48)(0.56)	(0.50)(0.53)
XGB	(0.59)(0.57)	(0.59)(0.58)	(0.62)(0.61)
Time Series Specific			
Dummy Classifier	(0.00)(0.55)	(0.00)(0.54)	(0.00)(0.54)
Catch22	(0.61)(0.68)	(0.61)(0.66)	(0.62)(0.66)
MRSQM	**(0.65)(0.74)**	**(0.66)(0.73)**	**(0.65)(0.71)**
SVC (Time Series)	(0.43)(0.52)	(0.44)(0.52)	(0.47)(0.54)
TS Forest	(0.61)(0.70)	(0.62)(0.71)	(0.62)(0.70)

Source: Author, 2025.

6.1.2 Cryptocurrencies Model Results

For the cryptocurrency experiments, results were analyzed both at the individual cryptocurrency level and across the entire dataset. In the latter case, the Kruskal-Wallis test was applied to determine whether there were significant differences in accuracy among the three model classes. The test yielded a P-value < 0.01, indicating strong evidence at the 5% significance level to reject the hypothesis that all model classes perform equivalently. Furthermore, pairwise comparisons using the Conover post-hoc test revealed that DL models were significantly different from both tabular ML models and TSC-specific ML models (P-value $<$ 0.01 in both comparisons). However, no statistically significant difference was found between TSC-specific and tabular ML models (P-value $= 0.8072$). It is important to note that the results of the Kruskal-Wallis test in this experiment differ from those obtained in the Bitcoin-

only experiment. Given the non-parametric nature of the test and the fact that identical DL model parameters were used in both experiments, these discrepancies may be attributed to differences in the TSC and tabular ML models used.

The accuracies for each model and time-window are detailed in Table 7. The MRSQM model achieved the highest accuracy across all timeframes. Within the DL category, the CNN-LSTM model performed best across all time-windows. However, its accuracy improvements over the baseline were relatively modest. Overall, DL models underperformed expectations, yielding the lowest accuracy results among all model classes. Regarding tabular ML models, KNN and Random Forest were among the top-performing models.

Table 7 – Cryptocurrencies Prediction Accuracy for All Model Types

Model	30 minutes	1 Hour	2 Hours
Deep Learning			
Attention CNN-GRU	0.53	0.53	0.53
CNN-LSTM	0.55	0.58	0.58
MLP	0.54	0.53	0.54
BiLSTM	0.52	0.51	0.51
Machine Learning Tabular			
Random Forest	0.64	0.63	0.63
KNN	0.64	0.64	0.62
Logistic Regression	0.54	0.54	0.54
XGB	0.55	0.56	0.57
Time Series Specific			
Dummy Classifier	0.52	0.52	0.51
Catch22	0.62	0.60	0.58
MRSQM	**0.69**	**0.69**	**0.68**
TS Forest	0.61	0.61	0.61

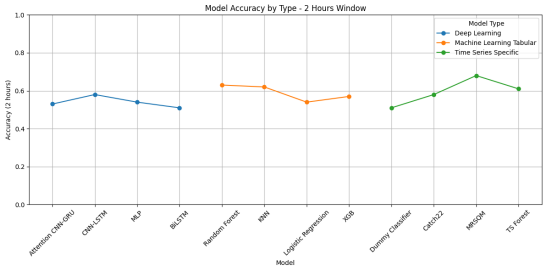
Source: Author, 2025.

Figure 10 illustrates the impact of each model compared to the baseline (Dummy Classifier). There was a slight decrease of 1.92% in accuracy performance when comparing the BiLSTM with the baseline for the 1 hour time-window. Furthermore, the increase in accuracy for the top-performing models in each class, across different time windows, is detailed as follows:

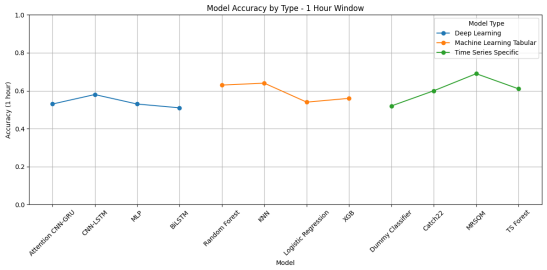
- In the 30-minute window, the accuracy increase was of 32.70% (MRSQM), 23.10% (Random Forest and KNN), and 6% (CNN-LSTM).

- In the 1-hour window, the accuracy increase was of 32.70% (MRSQM), 23.10% (KNN), and 11.54% (CNN-LSTM).
- In the 2-hour window, the accuracy increase was of 33.33% (MRSQM), 23.53% (Random Forest), and 13.73% (Attention CNN-GRU).

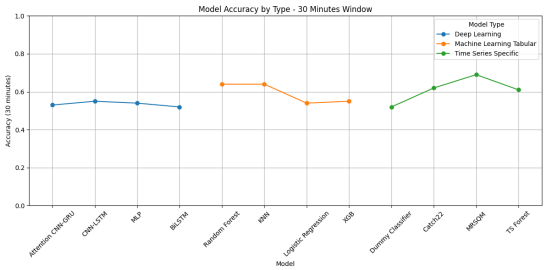
Figure 10 – Comparison of cryptocurrencies accuracy at different time intervals



(a) Accuracy - 2 Hours



(b) Accuracy - 1 Hour



(c) Accuracy - 30 minutes

Source: Author, 2025.

Regarding the precision of both classes (selling and holding), the MRSQM maintained balanced and strong results (Table 8). For the 30-minute and 1-hour time-windows, the model achieved a Sell recommendation precision of 70%, meaning that 7 out of 10 Sell predictions were true positives. In the 2-hour window, this precision slightly decreased to 69%. For the Hold recommendation, MRSQM performed consistently well, with precision scores only slightly lower than logistic regression in the 30-minute and 1-hour windows. For instance, in the 1-hour timeframe, logistic regression correctly predicted approximately 8 out of 10 Hold recommendations (75% precision). However, in the 2-hour window, MRSQM outperformed logistic regression, achieving nearly 7 out of 10 correct Hold predictions, effectively minimizing potential losses for investors, since it's able to better capture when the market will remain stable or go down on average.

Table 8 – Cryptocurrencies Prediction Precision for All Model Types (Class 0) (Class 1)

Model	30 minutes	1 Hour	2 Hours
Deep Learning			
Attention CNN-GRU	(0.52)(0.54)	(0.52)(0.53)	(0.53)(0.53)
CNN-LSTM	(0.53)(0.57)	(0.59)(0.57)	(0.58)(0.57)
MLP	(0.63)(0.53)	(0.54)(0.53)	(0.57)(0.53)
BiLSTM	(0.53)(0.52)	(0.50)(0.52)	(0.50)(0.52)
Machine Learning Tabular			
Random Forest	(0.67)(0.62)	(0.68)(0.61)	(0.65)(0.61)
KNN	(0.62)(0.65)	(0.63)(0.64)	(0.62)(0.63)
Logistic Regression	(0.70)(0.53)	(0.75)(0.53)	(0.56)(0.53)
XGB	(0.60)(0.55)	(0.59)(0.55)	(0.59)(0.57)
Time Series Specific			
Dummy Classifier	(0.00)(0.52)	(0.00)(0.52)	(0.00)(0.51)
Catch22	(0.60)(0.63)	(0.59)(0.60)	(0.57)(0.60)
MRSQM	**(0.67)(0.70)**	**(0.68)(0.70)**	**(0.67)(0.69)**
TS Forest	(0.62)(0.60)	(0.62)(0.60)	(0.62)(0.61)

Source: Author, 2025.

Accuracy was also evaluated by cryptocurrency type (Table 9), as the model may better capture the behavior of certain cryptocurrencies while struggling with others. In this experiment, the Kruskal-Wallis test yielded a P-value < 0.0001, indicating significant differences in accuracy across model types. The Conover post-hoc test also produced a P-value < 0.0001 when comparing DL models with both tabular and TSC-specific ML models. However, consistent with previous findings, no significant differences were observed between TSC-specific and tabular ML models at the 5% level of significance (P-value $= 0.3319$).

The AVAX achieved the highest accuracy across all models for the 30-minute time-window, reaching 74%, while among the top-performing models, the lowest accuracy was observed for USDT. For the 1-hour time-window, ADA achieved the highest accuracy (72%), whereas USDT remained the lowest-performing cryptocurrency. In the 2-hour time-window, only SOL and AVAX managed to reach 70% accuracy. Notably, USDT consistently had the lowest accuracy among the best-performing models. Across all time-windows, MRSQM demonstrated strong performance, particularly in the 1-hour timeframe, where it dominated the top results. The Random Forest model also performed well, while DL models fell below expectations, even though the CNN-LSTM model showed improved performance specifically for BTC.

Table 9 – Cryptocurrencies Prediction Accuracy for All Model Types By Crypto

(a) 30 Minutes Time-Window

Model	ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
Dummy Classifier	0.51	0.51	0.51	0.53	0.51	0.53	0.55	0.56	0.51	0.55
Catch22	0.61	0.60	0.61	0.55	0.62	0.66	0.70	0.69	0.62	0.61
MRSQM	**0.68**	0.68	**0.67**	**0.62**	**0.69**	**0.72**	**0.72**	**0.74**	**0.67**	**0.70**
Time-Series Forest	0.65	0.63	0.51	0.53	0.65	0.63	0.55	0.65	0.51	0.64
Random Forest	**0.68**	**0.69**	0.51	0.53	**0.69**	0.70	0.56	0.71	0.51	0.68
KNN	0.64	0.65	0.65	0.51	0.65	0.66	0.66	0.67	0.61	0.65
Logistic Regression	0.53	0.51	0.51	0.53	0.51	0.53	0.55	0.56	0.51	0.58
XGBoost	0.53	0.54	0.54	**0.62**	0.55	0.56	0.59	0.56	0.55	0.56
Attention CNN-GRU	0.51	0.51	0.51	0.53	0.51	0.53	0.55	0.56	0.51	0.58
CNN-LSTM	0.53	0.50	0.51	0.53	0.54	0.53	0.55	0.57	0.49	0.64
MLP	0.51	0.51	0.51	0.53	0.52	0.53	0.55	0.57	0.51	0.59
BiLSTM	0.51	0.51	0.51	0.53	0.51	0.53	0.55	0.56	0.51	0.55

(b) 1 Hour Time-Window

Model	ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
Dummy Classifier	0.51	0.50	0.51	0.51	0.50	0.51	0.52	0.52	0.51	0.54
Catch22	0.61	0.57	0.60	0.58	0.58	0.62	0.62	0.60	0.62	0.59
MRSQM	**0.69**	**0.69**	**0.69**	**0.62**	**0.69**	**0.71**	**0.72**	**0.71**	**0.70**	**0.70**
TS Forest	0.65	0.65	0.51	0.51	0.65	0.63	0.52	0.64	0.51	0.65
Random Forest	**0.69**	**0.69**	0.51	0.51	**0.69**	0.69	0.52	0.68	0.51	0.69
KNN	0.65	0.64	0.65	0.53	0.64	0.62	0.65	0.64	0.64	0.64
Logistic Regression	0.54	0.51	0.51	0.51	0.56	0.51	0.52	0.52	0.51	0.59
XGB	0.53	0.55	0.56	0.61	0.56	0.54	0.58	0.53	0.56	0.56
Attention CNN-GRU	0.51	0.49	0.51	0.51	0.51	0.51	0.52	0.52	0.51	0.58
CNN-LSTM	0.66	0.52	0.51	0.51	0.60	0.54	0.52	0.53	0.51	0.65
MLP	0.51	0.49	0.51	0.51	0.53	0.51	0.52	0.53	0.51	0.59
BiLSTM	0.51	0.50	0.51	0.51	0.50	0.51	0.52	0.52	0.51	0.54

(c) 2-Hour Time-Window

Model	ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
Dummy Classifier	0.51	0.50	0.50	0.49	0.50	0.51	0.50	0.50	0.51	0.54
Catch22	0.60	0.57	0.60	0.57	0.58	0.57	0.58	0.58	0.59	0.59
MRSQM	0.68	0.68	**0.68**	0.59	**0.68**	**0.70**	**0.69**	**0.70**	**0.69**	**0.68**
TS Forest	0.66	0.66	0.51	0.51	0.66	0.65	0.51	0.65	0.51	0.65
Random Forest	**0.69**	**0.69**	0.51	0.51	**0.68**	0.68	0.51	0.67	0.51	0.67
KNN	0.64	0.63	0.63	0.56	0.63	0.62	0.62	0.63	0.63	0.62
Logistic Regression	0.56	0.51	0.50	0.51	0.53	0.51	0.50	0.50	0.51	0.59
XGB	0.56	0.58	0.57	**0.62**	0.59	0.56	0.57	0.52	0.57	0.56
Attention CNN-GRU	0.53	0.51	0.50	0.51	0.53	0.48	0.50	0.50	0.51	0.59
CNN-LSTM	0.66	0.54	0.50	0.49	0.60	0.54	0.50	0.50	0.49	0.65
MLP	0.56	0.50	0.50	0.51	0.55	0.51	0.50	0.51	0.51	0.60
BiLSTM	0.51	0.50	0.50	0.51	0.50	0.49	0.50	0.50	0.51	0.54

Source: Author, 2025.

Considering Table 7 and Table 9, the precision of each class was highlighted for the most promising models in each category based on accuracy. The MRSQM demonstrated consistent and strong precision for both sell and hold recommendations, particularly excelling in AVAX. Additionally, some models with lower overall accuracy exhibited higher precision for one of the classes. For instance, the CNN-LSTM model in the 2-hour time-window for AVAX achieved an accuracy of only 50%, yet its precision for selling recommendations was 1.0. This suggests that the model was likely unbalanced, predicting the selling class more frequently while failing to properly learn the holding recommendation. Furthermore, Table 10 illustrates the precision of each class by cryptocurrency for these models, which were deemed the best in each category.

Table 10 – Prediction Precision for All Best Performing Models By Cryptocurrency (Class 0) (Class 1)

(a) 30 Minutes Time-Window

Model	ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
MRSQM	(0.67)(0.70)	(0.67)(0.69)	(0.66)(0.68)	(0.61)(0.62)	(0.69)(0.69)	(0.71)(0.73)	(0.74)(0.71)	(0.75)(0.74)	(0.65)(0.68)	(0.67)(0.72)
Random Forest	(0.68)(0.69)	(0.68)(0.70)	(0.59)(0.51)	(1.00)(0.53)	(0.69)(0.70)	(0.71)(0.70)	(0.78)(0.55)	(0.71)(0.70)	(0.75)(0.51)	(0.66)(0.70)
KNN	(0.63)(0.64)	(0.64)(0.65)	(0.64)(0.65)	(0.49)(0.54)	(0.65)(0.65)	(0.65)(0.66)	(0.64)(0.67)	(0.66)(0.68)	(0.61)(0.62)	(0.62)(0.67)
CNN-LSTM	(0.51)(0.80)	(0.49)(0.52)	(0.51)(0.51)	(1.00)(0.53)	(0.52)(0.57)	(0.50)(0.55)	(0.00)(0.55)	(0.52)(0.61)	(0.49)(0.00)	(0.66)(0.63)

(b) 1 Hour Time-Window

Model	ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
MRSQM	(0.69)(0.70)	(0.68)(0.69)	(0.67)(0.70)	(0.60)(0.66)	(0.68)(0.70)	(0.71)(0.71)	(0.72)(0.71)	(0.73)(0.70)	(0.68)(0.71)	(0.68)(0.72)
Random Forest	(0.69)(0.70)	(0.69)(0.69)	(0.57)(0.51)	(0.00)(0.51)	(0.68)(0.69)	(0.69)(0.70)	(0.87)(0.52)	(0.67)(0.68)	(0.88)(0.51)	(0.67)(0.71)
KNN	(0.64)(0.65)	(0.64)(0.64)	(0.65)(0.66)	(0.52)(0.53)	(0.63)(0.64)	(0.62)(0.62)	(0.65)(0.65)	(0.63)(0.65)	(0.63)(0.65)	(0.62)(0.66)
CNN-LSTM	(0.64)(0.70)	(0.51)(0.87)	(0.00)(0.51)	(0.00)(0.51)	(0.57)(0.68)	(0.55)(0.54)	(0.00)(0.52)	(0.94)(0.52)	(0.00)(0.51)	(0.76)(0.62)

(c) 2-Hour Time-Window

Model	ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
MRSQM	(0.67)(0.69)	(0.68)(0.68)	(0.67)(0.69)	(0.58)(0.60)	(0.68)(0.69)	(0.69)(0.71)	(0.68)(0.71)	(0.70)(0.70)	(0.68)(0.70)	(0.66)(0.70)
Random Forest	(0.69)(0.69)	(0.69)(0.69)	(0.69)(0.51)	(0.51)(1.00)	(0.68)(0.68)	(0.68)(0.69)	(0.67)(0.51)	(0.66)(0.67)	(0.53)(0.51)	(0.65)(0.69)
KNN	(0.63)(0.64)	(0.62)(0.63)	(0.62)(0.64)	(0.55)(0.57)	(0.63)(0.63)	(0.62)(0.62)	(0.62)(0.62)	(0.63)(0.62)	(0.63)(0.64)	(0.60)(0.64)
CNN-LSTM	(0.63)(0.70)	(0.61)(0.53)	(0.00)(0.50)	(0.00)(0.49)	(0.59)(0.63)	(0.66)(0.53)	(0.50)(0.00)	(0.50)(1.00)	(0.49)(0.00)	(0.70)(0.63)

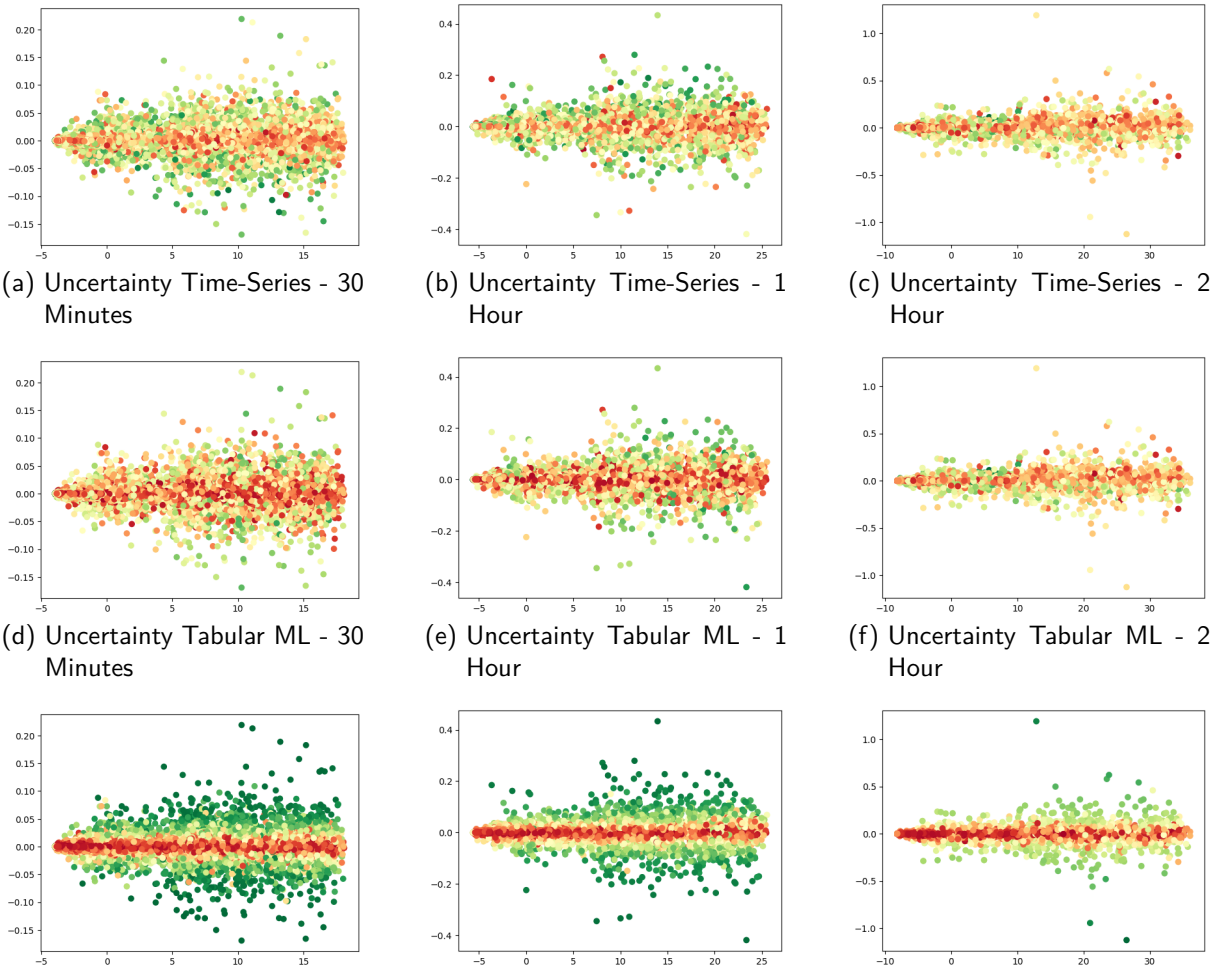
Source: Author, 2025.

6.2 UNCERTAINTY

The analysis of uncertainty was brief, as the experiment was conducted using only Bitcoin data. The results showed no evidence that the uncertainty in any model class or time-window followed a Gaussian distribution. The Shapiro-Wilk test yielded P-values$<$0.0001, thus rejecting the null hypothesis of normality. Additionally, the Kruskal-Wallis test across all models and time-windows produced a p-value $<$ 0.0001, supporting the hypothesis that at least one model class differs in estimating aleatoric uncertainty at the 5% of statistical significance level. The Conover post-hoc test yielded p-values $<$ 0.0001 when comparing all three model classes, indicating significant differences in their uncertainty estimations.

Figure 11 exhibits the uncertainty captured across different time-windows and model types. Red indicates higher levels of uncertainty, while greener shades represent lower uncertainty. For the TSC-specific ML models (Figures 11a, 11b, and 11c), more green points were observed compared to Tabular ML models, suggesting that these models are more confident considering their UE. In contrast, for Tabular models (Figures 11d, 11e, and 11f), green points were less prevalent in comparison to TSC-specific ML models, while red points were more prominent, indicating a higher frequency of low-confidence predictions in this category. For DL models (Figures 11g, 11h, and 11i), both extremes were frequently captured, as this class of models tended to overestimate confidence in certain time-series while capturing lower confidence in others.

Figure 11 – Comparison of Uncertainty At Different Time-Windows By Model Class



(a) Uncertainty Time-Series - 30 Minutes
(b) Uncertainty Time-Series - 1 Hour
(c) Uncertainty Time-Series - 2 Hour
(d) Uncertainty Tabular ML - 30 Minutes
(e) Uncertainty Tabular ML - 1 Hour
(f) Uncertainty Tabular ML - 2 Hour
(g) Uncertainty Deep Learning - 30 Minutes
(h) Uncertainty Deep Learning - 1 Hour
(i) Uncertainty Deep Learning - 2 Hour

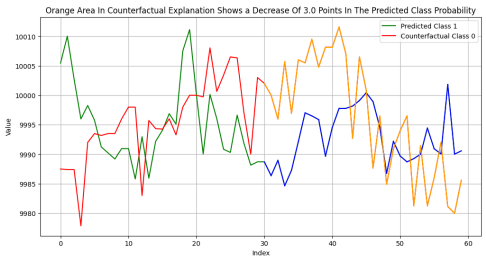
Source: Author, 2025.

6.3 COMTE-LEFTIST EXPLANATIONS

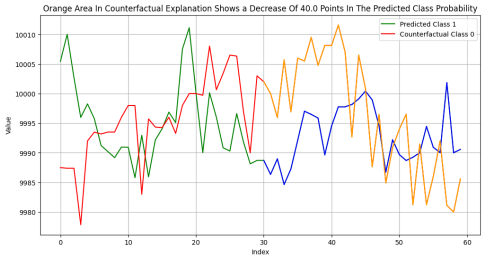
Explanations were generated using the MRSQM model on the 1-hour time-window data, as this model was identified as the most stable and best-performing. For Bitcoin data, two random instances were selected (no differences were found between LIME and SHAP, hence why LIME was chosen in the following experiments), one corresponding to a selling recommendation in the predicted class (Class 1) and the other to a holding recommendation (Class 0). Figure 12 illustrates the COMTE-LEFTIST explanation. In these figures, the orange highlighted line in the counterfactual series marks the most significant time-window in the explanation, that indicates the behavior most likely to alter the classification. The blue highlighted line represents the most influential segment of the predicted series that affects the class prediction.

Some differences appear in the estimation of the impact on probability when considering standardization. When analyzing the selling recommendation time-series, a comparison between the standardized (Figure 12a) and non-standardized (Figure 12b) versions, the impact on the score probability changes, since more weight is put on the highlighted shapelets of the counterfactual explanation. A similar effect is observed in the holding recommendation case (Figure 12c vs. Figure 12d), demonstrating that standardization influences the computation of probability impact, leading to variations in the explanation results.

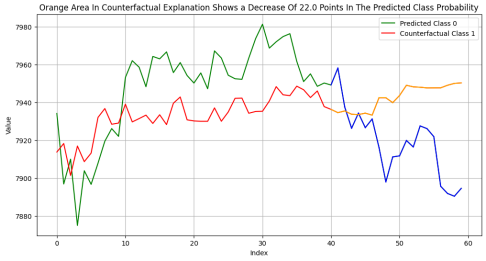
Figure 12 – COMTE-LEFTIST Explanations for Bitcoin In 1-hour Time-Window



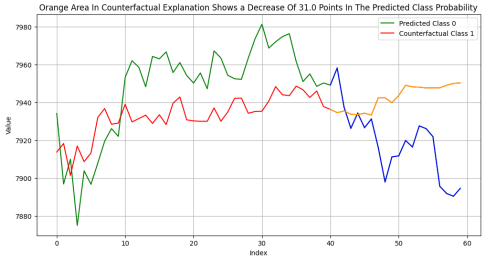
(a) Selling Recommendation Standardized



(b) Selling Recommendation Not Standardized



(c) Holding Recommendation Standardized

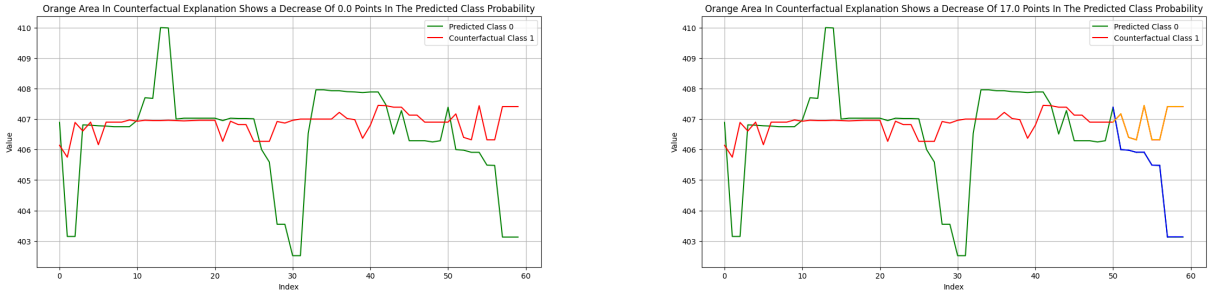


(d) Holding Recommendation Not Standardized

Source: Author, 2025.

It is important to highlight that the explanations are sensitive to the random seed, which is why seed 2 was used consistently throughout the explanation results. In Figure 13, the same time-series was analyzed under different random seeds. A comparison between Figure 13a, which uses a random seed of 2 and detected no or minimal impact on the score probability from the counterfactual explanation (hence the 0 impact on probability), and Figure 13b, which was generated with a random seed of 23, shows that the change in seed allowed the model to detect and attribute importance to a shapelet in the time-series. This shapelet was identified as responsible for a 17-point decrease in the score probability for the predicted class.

Figure 13 – Stochastic Behavior of COMTE-LEFTIST Explanations



(a) Explanation using Seed 2

(b) Explanation using Seed 23

Source: Author, 2025.

Regarding the cryptocurrency data, the first MRSQM model, fitted to obtain the classification results, is used to generate classification labels for each time series. However, for the counterfactual COMTE plots and LEFTIST importance analysis, a separate MRSQM model must be trained as a surrogate for the time-series data, excluding the dummy columns that identify which cryptocurrency the time-series belongs to. Therefore this surrogate model is fitted for each cryptocurrency, with accuracy measured relative to the predictions of the original model being used as an indicator of "goodness-of-fit". The lowest accuracy observed was 0.90 for BTC, suggesting that the surrogate model is a reliable approximation of the original. Cryptocurrencies with lower representation in the dataset typically exhibited higher accuracy values, such as AVAX, SOL, and ADA, as shown in Table 11.

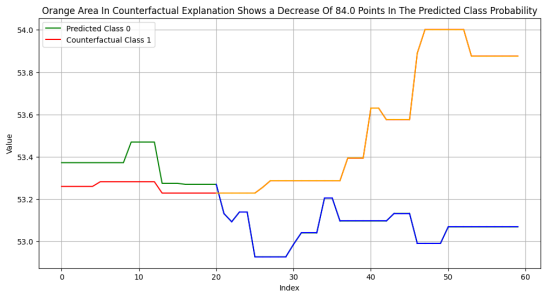
Table 11 – Cryptocurrency Surrogate Model Accuracy Values

ETH	LTC	XRP	USDT	BCH	SOL	ADA	AVAX	XLM	BTC
0.9277	0.9269	0.9218	0.9333	0.9380	1.0000	0.9951	1.0000	0.9795	0.8954

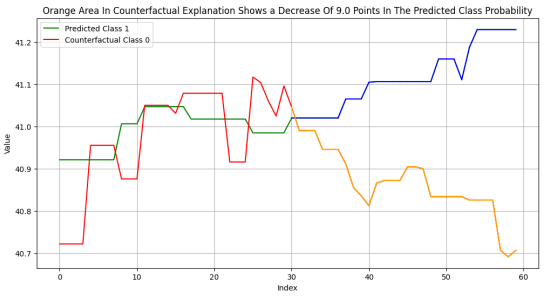
Source: Author, 2025.

In Figure 14, COMTE-LEFTIST explanations are presented for AVAX data. Both explanations rely on standardization, which is crucial when employing a surrogate model. Without standardization, the explanations were often suboptimal and did not yield reliable results.

Figure 14 – COMTE-LEFTIST Explanations For the Cryptocurrencies Data (AVAX)



(a) COMTE-LEFTIST Explanations For AVAX

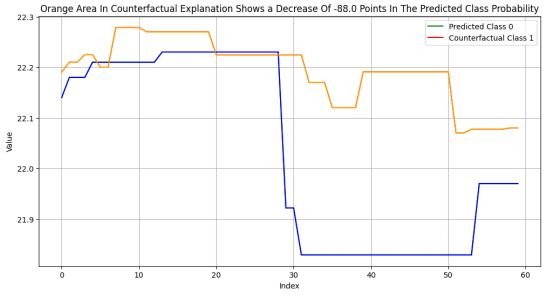


(b) COMTE-LEFTIST Explanations For AVAX

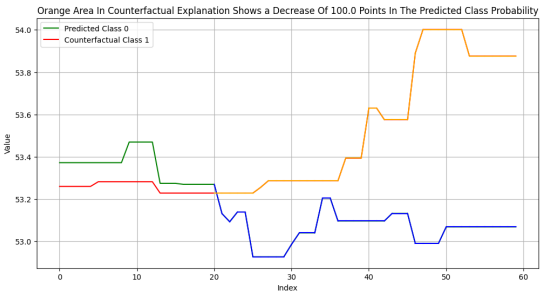
Source: Author, 2025.

Explanations for the cryptocurrency dataset were significantly more unstable compared to Bitcoin data, which affects their reliability (Figure 15). Figures 15a and 15b illustrate common issues that arise when standardization is not applied in the surrogate model strategy. In the former, a negative impact appears, this pattern frequently occurred when generating explanations without standardization. In the latter, an impact of 100% was attributed to the highlighted shapelet, which is an extreme and questionable result.

Figure 15 – Problems with COMTE-LEFTIST Not Standardized Explanations for the Crypto Experiment



(a) Unreliable Not Standardized With Negative Impact

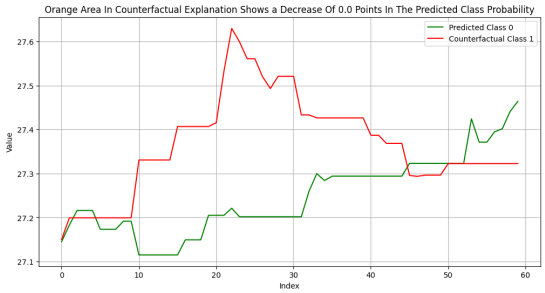


(b) AVAX Not Standardized With 100% Impact

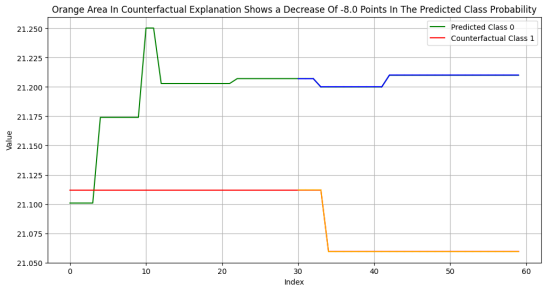
Source: Author, 2025.

Regarding standardization, although it improved the stability of explanations, odd behaviors still appeared as shown in Figure 16, although less frequently than when standardization is not applied. Figure 16b shows that negative impact was still a constraint, while Figure 16a demonstrates a case where the explanation resulted in null impact. While null impact does not necessarily indicate a problem, it was observed frequently during experiments.

Figure 16 – Problems with COMTE-LEFTIST Standardized Explanations for the Crypto Experiment



(a) Standardized Explanations For AVAX With Null Impact



(b) Standardized Explanations For AVAX With Negative Impact

Source: Author, 2025.

6.4 CRITICAL EVALUATION

DL models achieved the worst performance in both experiments, while it wasn't possible to state with statistical significance that the performance of this specific class wasn't much worse than TSC-specific and ML-tabular models, for the Bitcoin experiment, the opposite happened in the Cryptocurrencies dataset, which is counter-intuitive, since more data was available, and may have happened because the same parameters from the Bitcoin experiment were kept. When including more cryptocurrency data, it appears that TSC-specific and ML-tabular models, more specifically the MRSQM had a slight increase in accuracy of 0.01 points, for BTC both in the 1 hour and 2 hour time-windows and Random Forest had an increase of 0.01 point for 1 hour and 0.07 points for the 2-hour time-window.

Another key difference lies in the Kruskal-Wallis test results. In the Bitcoin experiment, the p-value was above the 5% threshold. However, in the Cryptocurrency experiment, the opposite occurred. This discrepancy may be attributed to the underperformance of DL models in the latter experiment, since it's not possible statistically distinguish TSC-specific from ML-tabular models at the 5% level. Furthermore, regarding the Cryptocurrency experiment, it is challenging to determine whether the inclusion of additional data improved model learning, as each cryptocurrency exhibits distinct behavior, some being more chaotic while others are more predictable.

Furthermore, given that each crypto asset exhibits distinct behavior, Bitcoin follows cycles of growth and decline, perhaps influenced by a periodic event known as halving, which results from the fixed supply limit of 21 million Bitcoins. Fabus et al. (2024) analyzed this event and reported that nearly 90% of BTC has already been mined. To maintain scarcity, the halving

process reduces mining rewards by decreasing the number of BTC issued per block. This cycle occurs approximately every four years and, despite the influence of random market events, contributes to a relatively predictable price increase. In contrast, cryptocurrencies such as USDT, which is pegged to the US dollar, are more susceptible to external and unpredictable factors, such as political and/or economic trade conditions. Consequently, disregarding factors such as the sample size of each asset in the cryptocurrency experiment, these differences in behavior may explain why some cryptocurrencies, like BTC, exhibited higher predictability and classification accuracy, whereas others with more chaotic and less predictable price movements, such as USDT, yielded lower accuracies.

Comparing the findings of this work with other studies on cryptocurrency prediction and classification is challenging due to methodological differences. However, Ranjan, Kayal and Saraf (2023) employed a ML approach to classify Bitcoin price movements (increase or decrease) using daily and 5-minute granularity data. Their best performance was achieved with logistic regression, obtaining an accuracy of 64.8% for daily price predictions, while for 5-minute granularity, their best model reached 59.4% accuracy.

Similarly, Iqbal et al. (2024) explored longer time windows of 90, 30, and 7 days to classify Bitcoin price increases and decreases using LSTM, ANN, and SVC. While LSTM achieved the highest accuracy at 74%, SVC outperformed deep learning models in two of the time-windows. This pattern, where ML models occasionally outperform DL models, was also observed in this study with models such as Random Forest and MRSQM surpassing DL models.

In contrast, Kwon et al. (2019) compared GB and LSTM for different cryptocurrencies, finding that LSTM achieved the best classification results for each cryptocurrency evaluated. As stated in the previous paragraph, methodological differences play a crucial role in the outcomes of each experiment, and DL methods can in fact be used for TSC. However, considering the results from these prior studies, the performance of MRSQM and TSC-specific methods appears promising and should be further explored in the field of cryptocurrency recommendation systems, since this category of models was specifically designed to tackle this class of problems.

Regarding uncertainty estimation, notable differences were observed across the three model classes. Lower levels of high uncertainty were found in TSC-specific and ML-tabular models, whereas DL models exhibited the opposite trend. The prevalence of overconfident and underconfident predictions in DL approaches aligns with critiques of using softmax for UE. For instance, Klaß et al. (2022) and Alonso (2024) argue that raw softmax outputs are unreliable,

often leading to poor calibration in ANNs, which results in over- or under-confident predictions.

Additionally, Alonso (2024) suggests that averaging softmax outputs across an ensemble, as implemented in this work, captures aspects of both aleatoric and epistemic uncertainty, in this regard Gawlikowski et al. (2023) also states that even though softmax outputs should represent data uncertainty, it is not possible no tell the amount of model uncertainty that affects a specific prediction. Consequently, epistemic uncertainty may be influencing the final uncertainty estimates, contributing to the discrepancies observed between model classes. These factors, including model uncertainty and potential miscalibration, should be carefully considered when interpreting UE results.

In the Bitcoin experiment, explanations generated using the hybrid COMTE-LEFTIST approach demonstrated significantly greater stability compared to those in the Cryptocurrency experiment. The primary concern in this experiment is not the occurrence of null impact, since this simply indicates that no specific shapelet was identified as the most influential in class prediction, but rather the stochastic nature of the explanations. A simple change in the random seed can lead to variations in the generated explanations, highlighting potential instability. Regarding the standardization of the predicted and concatenated instances, if the trained model was also fitted on standardized data, it is considered good practice to standardize the instances as well, given the gap that was identified when calculating the impact of the explanation on the score probability.

In contrast, the explanations in the Cryptocurrency experiment were considerably less reliable, likely due to the use of an intermediate surrogate model, which possibly propagates additional sources of error throughout the explanation process (causing negative impact and often 100% or null impact). As a result, applying COMTE-LEFTIST in this context requires caution, and its use is more advisable for univariate time-series, such as the Bitcoin time-series in the first experiment.

7 FINAL REMARKS

This final chapter concludes the study by presenting final remarks, detailing current limitations, summarizing key findings, and outlining future research directions in the field of XAI for TSC.

- The final conclusion of this research is presented in Section 7.1, summarizing the discussion on the employed models, the uncertainty study, and the COMTE-LEFTIST hybrid method.
- The limitations related to computational resources and employed methods encountered in this study are discussed in Section 7.2.
- Section 7.3 is structured as a bulleted list to summarize the key findings and main contributions of this study.
- Section 7.4 outlines future research directions based on the study's limitations and the current advancements in TSC, highlighting promising new methods that have yet to fully benefit from XAI frameworks.

7.1 CONCLUSION

The MRSQM achieved 70% accuracy in generating Sell and Hold recommendations for Bitcoin in the 30-minute time-window, outperforming other models in both experiments when considering BTC. For other cryptocurrencies, its performance remained strong, though occasionally slightly lower than models such as Random Forest or KNN. Additionally, it demonstrated high precision for the Selling and Holding classes. Furthermore, the Kruskal-Wallis test results were non-significant for the Bitcoin experiment, whereas in the Cryptocurrency experiment, the results at a 5% significance level suggested a rejection of the hypothesis of equality among model classes. This indicates that, in certain cases, TSC-specific models and tabular ML algorithms can outperform DL models, as the post hoc test did not find significant differences between Tabular and TSC ML models. Therefore, it is essential to experiment with different model classes to determine the most suitable approach, as the optimal choice depends on the problem's nature and the complexity of the time-series.

Uncertainty was assessed using the averaged inverse of the MCP across ensembles of the different model classes. The Kruskal-Wallis test results indicated that, at the 5% significance level, it would be reasonable to reject the assumption of equal uncertainty across model classes. Additionally, post hoc analysis using the Conover Test revealed significant differences among all three model classes. Furthermore, DL models have exhibited both higher uncertainty and confidence compared to the others model classes. Even though Holm, Wright and Augenstein (2023) supports using softmax in UE when resource efficiency is a concern, highlighting that softmax can be as good at UE as Monte Carlo dropout in some cases, they only recommend using it in low-risk applications. Furthermore, given the criticisms of the employed method (KLAß et al., 2022; ALONSO, 2024), these results should be interpreted with caution, and further improvements and research with more advanced UE methods is needed to assess uncertainty with more reliability.

Despite certain limitations, COMTE-LEFTIST emerges as one of the first attempts to generate instance- and subsequence-based explanations in TSC. Notably, neither SHAP nor LIME incorporates time-series representations within their explanations. Furthermore, previous explainability approaches in Bitcoin and cryptocurrency experiments, such as those proposed by Fior, Cagliero and Garza (2022), Babaei, Giudici and Raffinetti (2022), Gupta et al. (2023), and Morais (2022), have not focused on providing visual time-series explanations within their frameworks. Instead, these studies pursued different objectives, such as identifying important features for predicting cryptocurrency value or recommending asset allocation, which explains their predominant reliance on SHAP and LIME for model interpretability.

In contrast, the hybrid COMTE-LEFTIST method not only provides time-series as part of the explanation but also generates counterfactual examples and identifies the most important timestamps. As a result, COMTE-LEFTIST consolidates the key elements of an explanation. It offers users valuable insights, helping them understand the model's reasoning behind sell or hold recommendations, and assess whether it behaves as expected. Although certain limitations were encountered during the experiments, the contribution of hybrid explanations is crucial for advancing the field of explainability in TSC.

7.2 LIMITATIONS

The SVC model could not be used in the Cryptocurrency experiment due to excessive computation time. Given the available resources, it was not feasible to keep the machine

running for several days. This limitation also influenced model selection, as overly complex DL architectures required significant training time, with some models taking over six hours to train. Additionally, for TSC-specific models, due to the same reason of the SVC, it was not possible to use HIVE-COTE 2.0, which is the latest extension of the first ensemble-based algorithm for TSC (FAOUZI, 2024), being frequently updated, and is also available in sktime [1].

Uncertainty estimation used the softmax score probabilities and in general for all model classes involved averaging the inverse of the MCP, which often suffers from criticism since more advanced and reliable methods to investigate uncertainty are available for UE. The assessment of uncertainty was also only explored in the Bitcoin experiment, however it would also be useful to understand how uncertainty behaves among different crypto assets.

One of the major concerns regarding COMTE-LEFTIST is that it has a tendency to exhibit stochastic behavior, primarily due to the inherent randomness of the LEFTIST component. As a result, the hybrid method becomes highly dependent on the random seed used for reproducibility. This randomness can influence the calculation of the counterfactual explanation's impact on the predicted class probability, since it's dependent on LEFTIST values, leading to greater variability in results. In some cases, particularly observed in the cryptocurrency experiment, rather than reducing the predicted class probability, the explanation exhibited a negative impact. This led to the decision to avoid standardization when using surrogates in this experiment, since most cases happened when it was applied. Additionally, instances where no effect was observed were also noted; however, this was considered a less concerning issue, as no particular shapelet appeared to significantly alter the predicted class probability.

7.3 SUMMARIZATION

- The field of Explainable AI applied to TSC is an emerging and actively researched topic, as illustrated in Figure 4. This work contributes to this trend by proposing a hybrid explanation method. To the best of knowledge, it is the first study to introduce a hybridization of explainability models specifically tailored for TSC, with a particular focus on cryptocurrency data.

- The MRSQM achieved the best performance across both experiments, standing out for its accuracy and consistently strong precision in the holding and selling classes. Its

[1] <https://www.sktime.net/en/stable/api_reference/auto_generated/sktime.classification.hybrid.HIVECOTEV2.html#sktime.classification.hybrid.HIVECOTEV2>, accessed on March 18, 2025

stability makes it a relatively "safe" model for recommendation tasks.

- While uncertainty estimation in this experiment requires further refinement, the findings suggest differences in the UE of aleatoric uncertainty across ensembles of the three model classes.
- COMTE-LEFTIST has been fully implemented, despite some limitations. This contribution aims to introduce hybrid XAI modeling to TSC and encourage further research in this area.

7.4 FUTURE WORK AND RESEARCH HORIZON

This work is among the first ones to explore hybrid XAI methods for TSC, highlighting the need for further research into the hybridization of different XAI techniques in this domain. Additionally, the frequent instability of COMTE-LEFTIST in the Cryptocurrency experiment was not fully examined in this study and warrants further investigation before it can be reliably applied to datasets with multiple time-series, such as multivariate time-series data.

Emerging methods like Wide-TSNet, proposed by Yamak et al. (2024), have demonstrated promising results, achieving 94% accuracy in Bitcoin price prediction by integrating time-series and image classification via Markov Transition Fields. These models excel at identifying key regions within a sequence and capturing temporal correlations, making them a compelling direction for further research. Future studies could explore training surrogate models on these advanced architectures that merge image classification with TSC, by enhancing explainability through COMTE-LEFTIST and other TSC XAI techniques.

BIBLIOGRAPHY

ABADE, N. A.; JÚNIOR, O. A. de C.; AES, R. F. G.; OLIVEIRA, S. N. de. Comparative analysis of modis time-series classification using support vector machines and methods based upon distance and similarity measures in the brazilian cerrado-caatinga boundary. *Remote Sensing*, v. 7, n. 9, p. 12160–12191, 2015. Available at: <https://doi.org/10.3390/rs70912160>.

ALONSO, A. G. *Uncertainty of deep learning classifiers: A comparative study across different architectures and quantification methods*. Master's Thesis (Master's Dissertation), 2024. Available from: <https://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-24581>. Available at: <https://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-24581>.

ÁLVAREZ, G. Q.; DÍAZ, M. J. del J.; GARCÍA, P. G. Explainable artificial intelligence: An overview on hybrid models. In: *Proceedings of the First Multimodal, Affective and Interactive eXplainable AI Workshop (MAI-XAI24 2024), co-located with 27th European Conference On Artificial Intelligence (ECAI 2024)*. CEUR Workshop Proceedings (CEUR-WS.org), 2024. v. 3803, p. 49–60. ISSN 1613-0073. Available at: <https://ceur-ws.org/Vol-3803/paper4.pdf>.

ARRIETA, A. B.; DÍAZ-RODRÍGUEZ, N.; SER, J. D.; BENNETOT, A.; TABIK, S.; BARBADO, A.; GARCÍA, S.; GIL-LÓPEZ, S.; MOLINA, D.; BENJAMINS, R.; CHATILA, R.; HERRERA, F. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, v. 58, p. 82–115, 2020. ISSN 1566-2535. Available at: <https://www.sciencedirect.com/science/article/pii/S1566253519308103>.

ATES, E.; AKSAR, B.; LEUNG, V. J.; COSKUN, A. K. Counterfactual explanations for multivariate time series. In: *2021 International Conference on Applied Artificial Intelligence (ICAPAI)*. [S.l.: s.n.], 2021. p. 1–8.

BABAEI, G.; GIUDICI, P.; RAFFINETTI, E. Explainable artificial intelligence for crypto asset allocation. *Finance Research Letters*, v. 47, p. 102941, 2022. ISSN 1544-6123. Available at: <https://www.sciencedirect.com/science/article/pii/S1544612322002021>.

BAGNALL, A.; LINES, J.; BOSTROM, A. et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, v. 31, n. 3, p. 606–660, 2017.

BHATT, U.; ANTORÁN, J.; ZHANG, Y.; LIAO, Q. V.; SATTIGERI, P.; FOGLIATO, R.; MELANcON, G.; KRISHNAN, R.; STANLEY, J.; TICKOO, O.; NACHMAN, L.; CHUNARA, R.; SRIKUMAR, M.; WELLER, A.; XIANG, A. Uncertainty as a form of transparency: Measuring, communicating, and using uncertainty. In: *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. New York, NY, USA: Association for Computing Machinery, 2021. (AIES '21), p. 401–413. ISBN 9781450384735. Available at: <https://doi.org/10.1145/3461702.3462571>.

CATTELAN, L.; SILVA, D. On the performance of uncertainty estimation methods for deep-learning based image classification models. In: *Anais do XIX Encontro Nacional de Inteligência Artificial e Computacional*. Porto Alegre, RS, Brasil: SBC, 2022. p. 532–543. ISSN 2763-9061. Available at: <https://sol.sbc.org.br/index.php/eniac/article/view/22810>.

CONOVER, W. J.; IMAN, R. L. *On Multiple-Comparisons Procedures*. [S.l.], 1979. Available at: <https://doi.org/10.2172/6057803>.

CUOMO, J.; HOMAYOUNI, H.; RAY, I.; GHOSH, S. Detecting temporal dependencies in data. In: *Proceedings of the British International Conference on Databases*. [s.n.], 2023. Available at: <https://par.nsf.gov/biblio/10340373>.

DENG, H.; RUNGER, G.; TUV, E.; VLADIMIR, M. A time series forest for classification and feature extraction. *Information Sciences*, v. 239, p. 142–153, 2013. ISSN 0020-0255. Available at: <https://www.sciencedirect.com/science/article/pii/S0020025513001473>.

FABUS, J.; KREMENOVA, I.; STALMASEKOVA, N.; KVASNICOVA-GALOVICOVA, T. An empirical examination of bitcoin's halving effects: Assessing cryptocurrency sustainability within the landscape of financial technologies. *Journal of Risk and Financial Management*, v. 17, n. 6, p. 229, 2024. Available at: <https://doi.org/10.3390/jrfm17060229>.

FAOUZI, J. Time series classification: A review of algorithms and implementations. In: ROCHA, J.; VIANA, C. M.; OLIVEIRA, S. (Ed.). *Time Series Analysis*. Rijeka: IntechOpen, 2024. chap. 2. Available at: <https://doi.org/10.5772/intechopen.1004810>.

FAWAZ, H. I.; FORESTIER, G.; WEBER, J. et al. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, v. 33, p. 917–963, 2019. Available at: <https://doi.org/10.1007/s10618-019-00619-1>.

FIOR, J.; CAGLIERO, L.; GARZA, P. Leveraging explainable ai to support cryptocurrency investors. *Future Internet*, v. 14, n. 9, p. 251, 2022. Available at: <https://doi.org/10.3390/fi14090251>.

FRYER, D.; STRÜMKE, I.; NGUYEN, H. Shapley values for feature selection: The good, the bad, and the axioms. *IEEE Access*, v. 9, p. 144352–144360, 2021.

GAWLIKOWSKI, J.; TASSI, C. R. N.; ALI, M. et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, v. 56, n. Suppl 1, p. 1513–1589, 2023. Available at: <https://doi.org/10.1007/s10462-023-10562-9>.

GUILLEMÉ, M.; MASSON, V.; ROZÉ, L.; TERMIER, A. Agnostic local explanation for time series classification. In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. [S.l.: s.n.], 2019. p. 432–439.

GUPTA, A.; VYAS, D.; NALE, P.; JAIN, H.; MISHRA, S.; BIDWE, R. V.; ZOPE, B.; BUCHADE, A. Cryptocurrency prediction and analysis between supervised and unsupervised learning with xai. In: *2023 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*. [S.l.: s.n.], 2023. p. 1–7.

HELLANI, H.; SAMHAT, A. E.; CHAMOUN, M.; GHOR, H. E.; SERHROUCHNI, A. On blockchain technology: Overview of bitcoin and future insights. In: *2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)*. [s.n.], 2018. p. 1–8. Available at: <https://doi.org/10.1109/IMCET.2018.8603029>.

HÖLLIG, J.; KULBACH, C.; THOMA, S. Tsinterpret: A python package for the interpretability of time series classification. *Journal of Open Source Software*, The Open Journal, v. 8, n. 85, p. 5220, 2023. Available at: <https://doi.org/10.21105/joss.05220>.

HOLM, A. N.; WRIGHT, D.; AUGENSTEIN, I. Revisiting softmax for uncertainty approximation in text classification. *Information*, v. 14, n. 7, 2023. ISSN 2078-2489. Available at: <https://www.mdpi.com/2078-2489/14/7/420>.

HOLZINGER, A.; KIESEBERG, P.; WEIPPL, E.; TJOA, A. M. Current advances, trends and challenges of machine learning and knowledge extraction: From machine learning to explainable ai. In: HOLZINGER, A.; KIESEBERG, P.; TJOA, A. M.; WEIPPL, E. (Ed.). *Machine Learning and Knowledge Extraction*. [S.l.]: Springer, Cham, 2018, (Lecture Notes in Computer Science, v. 11015).

IQBAL, M.; IQBAL, A.; ALSHAMMARI, A.; ALI, I.; MAGHRABI, L. A.; USMAN, N. Sell or hodl cryptos: Cryptocurrency short-to-long term projection using simultaneous classification-regression deep learning framework. *IEEE Access*, v. 12, p. 118169–118184, 2024.

JUDGE, T. *Uncertainty Estimation Review*. 2021. <https://vitalab.github.io/blog/2021/06/17/uncertainty.html>. Reviewed on Jun 17, 2021.

KLAß, A.; LORENZ, S. M.; LAUER-SCHMALTZ, M. W.; RÜGAMER, D.; BISCHL, B.; MUTSCHLER, C.; OTT, F. Uncertainty-aware evaluation of time-series classification for online handwriting recognition with domain shift. In: *Proceedings of the 1st International Workshop on Spatio-Temporal Reasoning and Learning (STRL 2022)*. CEUR-WS.org, 2022. (CEUR Workshop Proceedings, v. 3190). ISSN 1613-0073. Available at: <https://ceur-ws.org/Vol-3190/paper3.pdf>.

KRUSKAL, W. H.; WALLIS, W. A. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, v. 47, n. 260, p. 583–621, 1952.

KWON, D.-H.; KIM, J.-B.; HEO, J.-S.; KIM, C.-M.; HAN, Y.-H. Time series classification of cryptocurrency price trend based on a recurrent lstm neural network. *Journal of Information Processing Systems*, v. 15, n. 3, p. 694–706, 2019. Available at: <https://doi.org/10.3745/JIPS.03.0120>.

LÖNING, M.; BAGNALL, A.; GANESH, S.; KAZAKOV, V.; LINES, J.; KIRÁLY, F. sktime: A unified interface for machine learning with time series. In: . [S.l.: s.n.], 2019.

LUBBA, C. H.; SETHI, S. S.; KNAUTE, P. et al. catch22: Canonical time-series characteristics. *Data Mining and Knowledge Discovery*, v. 33, p. 1821–1852, 2019. Available at: <https://doi.org/10.1007/s10618-019-00647-x>.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017. (NIPS'17), p. 4768–4777. ISBN 9781510860964.

MINATEL, D.; SILVA, A. C. M. D.; SANTOS, N. R. D.; CURI, M.; MARCACINI, R. M.; LOPES, A. d. A. Data stratification analysis on the propagation of discriminatory effects in binary classification. In: *Symposium on Knowledge Discovery, Mining and Learning (KDMILE)*. Sociedade Brasileira de Computação, 2023. p. 73–80. ISSN 2763-8944. Available at: <https://doi.org/10.5753/kdmile.2023.232582>.

MOLNAR, C. *Interpretable Machine Learning*: A guide for making black box models explainable. 2. ed. [s.n.], 2022. Available at: <https://christophm.github.io/interpretable-ml-book>.

MORAIS, S. *Application of Explainable AI in Machine Learning Models to Identify the Main Determinants of Bitcoin Price*. Master's Thesis (Master's Dissertation) — Universidade Católica Portuguesa, December 2022.

MORISSE, M. Cryptocurrencies and bitcoin: Charting the research landscape. 2015.

NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system. 2008. Available at: <http://dx.doi.org/10.2139/ssrn.3440802>.

NGUYEN, T. L.; IFRIM, G. A short tutorial for time series classification and explanation with mrsqm. *Software Impacts*, v. 11, p. 100197, 2022. ISSN 2665-9638. Available at: <https://www.sciencedirect.com/science/article/pii/S2665963821000865>.

NGUYEN, T. L.; IFRIM, G. Fast time series classification with random symbolic subsequences. In: . Berlin, Heidelberg: Springer-Verlag, 2023. p. 50–65. ISBN 978-3-031-24377-6. Available at: <https://doi.org/10.1007/978-3-031-24378-3_4>.

QIAN, S.; QI, Y. How machine learning methods unravel the mystery of bitcoin price predictions. In: *Proceedings of the 2022 International Conference on mathematical statistics and economic analysis (MSEA 2022)*. Atlantis Press, 2022. p. 381–389. ISBN 978-94-6463-042-8. ISSN 2352-538X. Available at: <https://doi.org/10.2991/978-94-6463-042-8_56>.

RANJAN, S.; KAYAL, P.; SARAF, M. Bitcoin price prediction: A machine learning sample dimension approach. *Computational Economics*, v. 61, p. 1617–1636, 2023. Available at: <https://doi.org/10.1007/s10614-022-10262-6>.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "why should i trust you?": Explaining the predictions of any classifier. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 1135–1144. ISBN 9781450342322. Available at: <https://doi.org/10.1145/2939672.2939778>.

SARANYA, A.; SUBHASHINI, R. A systematic review of explainable artificial intelligence models and applications: Recent developments and future trends. *Decision Analytics Journal*, v. 7, p. 100230, 2023. ISSN 2772-6622. Available at: <https://www.sciencedirect.com/science/article/pii/S277266222300070X>.

SARKER, I. H. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, v. 2, p. 420, 2021. Available at: <https://doi.org/10.1007/s42979-021-00815-1>.

SARKER, I. H. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, v. 2, p. 160, 2021. Available at: <https://doi.org/10.1007/s42979-021-00592-x>.

SUSTO, G. A.; CENEDESE, A.; TERZI, M. Chapter 9 - time-series classification methods: Review and applications to power systems data. In: ARGHANDEH, R.; ZHOU, Y. (Ed.). *Big Data Application in Power Systems*. Elsevier, 2018. p. 179–220. ISBN 978-0-12-811968-6. Available at: <https://www.sciencedirect.com/science/article/pii/B9780128119686000097>.

TAHIR, H. A.; ALAYED, W.; HASSAN, W. U.; HAIDER, A. A novel hybrid xai solution for autonomous vehicles: Real-time interpretability through lime–shap integration. *Sensors*, v. 24, n. 21, p. 6776, 2024. Available at: <https://doi.org/10.3390/s24216776>.

TAVENARD, R.; FAOUZI, J.; VANDEWIELE, G.; DIVO, F.; ANDROZ, G.; HOLTZ, C.; PAYNE, M.; YURCHAK, R.; RUSSWURM, M.; KOLAR, K.; WOODS, E. Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, v. 21, n. 118, p. 1–6, 2020. Available at: <http://jmlr.org/papers/v21/20-091.html>.

THEISSLER, A.; SPINNATO, F.; SCHLEGEL, U.; GUIDOTTI, R. Explainable ai for time series classification: A review, taxonomy and research directions. *IEEE Access*, v. 10, p. 100700–100724, 2022.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L. u.; POLOSUKHIN, I. Attention is all you need. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. v. 30. Available at: <https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.

YAMAK, P. T.; LI, Y.; ZHANG, T.; GADOSEY, P. K. Wide-tsnet: A novel hybrid approach for bitcoin price movement classification. *Applied Sciences*, v. 14, n. 9, p. 3797, 2024. Available at: <https://doi.org/10.3390/app14093797>.

YE, L.; KEOGH, E. Time series shapelets: a new primitive for data mining. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2009. (KDD '09), p. 947–956. ISBN 9781605584959. Available at: <https://doi.org/10.1145/1557019.1557122>.