Pontifícia Universidade Católica
do Rio de Janeiro

Flavio Sérgio da Silva

Enhancing Asset Price Prediction: Conformal Prediction Ensembles

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Informática, do Departamento de Informática of PUC-Rio, in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor: Prof. José Alberto Rodrigues Pereira Sardinha

Rio de Janeiro
May 2025

Pontifícia Universidade Católica
do Rio de Janeiro

Flavio Sérgio da Silva

Enhancing Asset Price Prediction: Conformal Prediction Ensembles

Disse rtation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee:

Prof. José Alberto Rodrigues Pereira Sardinha
Advisor
Departamento de Informática – PUC-Rio

Prof. Markus Endler
Departamento de Informática – PUC-Rio

Prof. Edward Hermann Haeusler
Departamento de Informática – PUC-Rio

Rio de Janeiro, May 19th, 2025

Flavio Sérgio da Silva

Undergraduate in Computer Science from the Universidade Pontificia Católica do Rio de Janeiro - PUC-Rio.

Bibliographic data

Silva, Flavio Sergio da

Enhancing Asset Price Prediction: Conformal Prediction Ensembles / Flavio Sérgio da Silva; advisor: José Alberto Rodrigues Pereira Sardinha. – 2025.

106 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2025.

Inclui bibliografia

1. Informática – Teses. 2. Previsão de Preços de Ativos. 3. Conjunto de Previsões Conformes. 4. Gestão de Riscos. 5. Mercados Financeiros. 6. Modelos de Aprendizado de Máquina. I. Sardinha, José Alberto Rodrigues Pereira. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

To my mother and family
for their support and encouragement.

Acknowledgments

To my advisor, for the knowledge transfer, empathy, and support throughout this partnership in developing this work.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Silva, Flavio Sergio da; Sardinha, José Alberto Rodrigues Pereira (Advisor). **Enhancing Asset Price Prediction: Conformal Prediction Ensembles**. Rio de Janeiro, 2025. 106p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The financial market is widely recognized as a central indicator of a nation's economic vitality, providing essential credit and liquidity to support investment and capital allocation. It plays a dual role by enabling the growth of corporate capital and enhancing investor wealth. Asset Price Prediction (APP) has been approached through a range of techniques, including Conventional Statistics (CS), Fundamental Analysis (FA), Technical Analysis (TA), Heuristic Rules (HR), and, more recently, Machine Learning (ML). Despite considerable advancements in computational power and algorithmic design, APP remains a complex challenge due to the inherently stochastic and nonlinear behavior of financial markets. Recent state-of-the-art (SOTA) studies report trend prediction accuracy near 79% and price prediction accuracy around 27%. However, a key limitation of many existing approaches is their inability to provide reliable estimates of predictive uncertainty, which is critical for informed risk management. This work addresses this gap by proposing a Conformal Prediction Ensemble (CPE) framework that incorporates Conformal Prediction (CP) techniques to calibrate the outputs of ML-based APP models. The proposed methodology consists of four sequential steps: ML models generate Close value predictions, which are then calibrated using CP. Next, the Conformal Prediction Intervals (CPIs) are intersected to enhance reliability. Finally, a Random Approach (RA) is used to sample Close values from the resulting intersection set uniformly. Model performance is assessed with and without the application of CP, using the Symmetric Mean Absolute Percentage Error (sMAPE) as the evaluation metric. Empirical validation is carried out on two benchmark indices: the Standard & Poor's 500 (SPX) and the Bovespa Index (IBOV). The CPE framework demonstrates improved predictive robustness by explicitly incorporating uncertainty estimation, thus contributing to a practical and empirically grounded strategy for risk-aware APP in financial markets.

Keywords

Asset Price Prediction; Conformal Prediction Ensemble; Risk Management; Financial Markets; Machine Learning Models.

Resumo

Silva, Flavio Sergio da; Sardinha, José Alberto Rodrigues Pereira. **Aprimorando a Previsão de Preços de Ativos: Conformal Prediction Ensembles**. Rio de Janeiro, 2025. 106p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

O mercado financeiro é um indicador fundamental da saúde econômica de um país, promovendo crédito e liquidez para sustentar investimentos e o fluxo de capital. Ele facilita o crescimento das empresas e contribui para a geração de riqueza dos investidores. A Previsão de Preços de Ativos (APP) tem sido abordada por meio de diversas metodologias, como Estatísticas Convencionais (CS), Análise Fundamentalista (FA), Análise Técnica (TA), Regras Heurísticas (HR) e, mais recentemente, Aprendizado de Máquina (ML). Apesar dos avanços computacionais, a APP continua sendo uma tarefa complexa devido à natureza estocástica e caótica dos mercados financeiros.

Atualmente, os melhores resultados (SOTA) apresentam acurácia de cerca de 79% para previsão de tendência e aproximadamente 27% para previsão de preço. No entanto, a maioria das abordagens carece de métodos robustos para quantificar a incerteza das previsões, o que limita sua aplicação prática na gestão de riscos. Este estudo propõe um modelo baseado em Conformal Prediction Ensemble (CPE), integrando Conformal Prediction (CP) à calibragem dos resultados obtidos por ML para APP. A metodologia em cinco etapas inicia com o uso de HR para simular cenários realistas de APP. Em seguida, modelos de ML predizem o valor de fechamento (*Close*), que é calibrado com CP. Um Random Approach (RA) seleciona novos valores de *Close* de forma uniforme a partir do conjunto de previsões CP. Os resultados com e sem CP são comparados por meio do *Symmetric Mean Absolute Percentage Error (sMAPE)*. Os dados utilizados são os índices Standard and Poor's 500 (SPX) e Bovespa (IBOV). A proposta visa superar o desempenho de modelos ML isolados, incorporando estimativas de incerteza, e contribui com uma estratégia empírica e prática de gestão de risco baseada em CP.

Palavras-chave

Previsão de Preços de Ativos; Conjunto de Previsões Conformes; Gestão de Riscos; Mercados Financeiros; Modelos de Aprendizado de Máquina.

Table of contents

1 Introduction **16**
1.1 Problem 16
1.2 Context 17
1.3 Importance 18
1.4 Difficulty 18
1.5 Prediction Techniques 20
1.6 Uncertainty Quantification 21
1.7 Gap 23
1.8 Structure 24

2 Literature Review **25**
2.1 Prediction Models 25
2.2 Calibration Techniques 29

3 Methodology **31**
3.1 ML Layer 33
3.2 CP Layer 35
3.3 HR Layer 36
3.4 Ensemble 39
3.5 Analysis 40

4 Experiment **42**
4.1 ML Layer 44
4.2 CP Layer 49
4.3 Ensemble Method 51
4.4 Heuristic 54
4.5 Analysis 57

5 Conclusion **61**
5.1 Future Works 61

6 Bibliography **63**

Glossary **73**

A Appendix **75**
A.1 Background 75
A.2 Review 86
A.3 Proposal 101
A.4 Additional Information 105

List of figures

Figure 3.1 Illustration of our multi-model ensemble pipeline 31
Figure 3.2 Intersection of CPI. 38
(a) Intersection of the Conformal Prediction Interval (CPI). 38
(b) Possible Scenarios Cases of the Intersection of the Conformal Prediction Interval (CPI). 38

Figure 4.1 SPX dataset with and without shifting. 43
(a) SPX without any shifting. 43
(b) SPX with shifting. 43
Figure 4.2 BVSP dataset with and without shifting. 43
(a) BVSP without any shifting. 43
(b) BSVP with shifting. 43
Figure 4.3 LSTM Prediction over the *test* subset. 47
(a) BVSP Real vs Predicted. 47
(b) SPX Real vs Predicted. 47
Figure 4.4 SVM Prediction over the *test* subset. 48
(a) BVSP Real vs Predicted. 48
(b) SPX Real vs Predicted. 48
Figure 4.5 XGBoostPrediction over the *test* subset. 49
(a) BVSP Real vs Predicted. 49
(b) SPX Real vs Predicted. 49
Figure 4.6 CQR: Conformal Results over the BVSP dataset. 50
(a) LSTM - BVSP. 50
(b) SVM - BVSP. 50
(c) XGBoost - BVSP. 50
Figure 4.7 CQR: Conformal Results over the SPX dataset. 50
(a) LSTM - SPX. 50
(b) SVM - SPX. 50
(c) XGBoost - SPX. 50
Figure 4.8 Mondrial CP: Conformal Results over the BVSP dataset. 51
(a) LSTM - BSVP. 51
(b) SVM - BVSP. 51
(c) XGBoost - BVSP. 51
Figure 4.9 Mondrial CP: Conformal Results over the SPX dataset. 51
(a) LSTM - SPX. 51
(b) SVM - SPX. 51
(c) XGBoost - SPX. 51
Figure 4.10 Intersection of six BVSP's *CPI*. 52
(a) BVSP without zoom. 52
(b) BVSP with zoom on the last 10%. 52
Figure 4.11 Intersection of six SPX's *CPI*. 53
(a) SPX without zoom. 53
(b) SPX with zoom on the last 10%. 53
Figure 4.12 Heuristic - Final Result of BVSP using Ensemble-M. 56
(a) BVSP without zoom. 56

(b) BVSP with zoom on the Last 10%. 56
Figure 4.13 Heuristic - Final Result of SPX using Ensemble-M. 56
(a) SPX without zoom. 56
(b) SPX with zoom on the Last 10%. 56
Figure 4.14 Heuristic - Final Result of BVSP using Ensemble-R. 57
(a) BVSP without zoom. 57
(b) BVSP with zoom on the Last 10%. 57
Figure 4.15 Heuristic - Final Result of SPX using Ensemble-R. 57
(a) SPX without zoom. 57
(b) SPX with zoom on the Last 10%. 57

Figure A.1 *The architecture of LSTM unit (STAUDEMEYER; MORRIS, 2019).* 76
Figure A.2 Approaches for Comformal Prediction.(SEEDAT et al., 2023) 86
Figure A.3 *Traditional pre-processing code (authorship based on (SHAH, 2022) sample).* 102
Figure A.4 Prediction result of the calibration step (authorship based on (SHAH, 2022) sample 102
(a) Code 102
(b) Plot 102
Figure A.5 Result of the *prediction absolute error (PAE)* and the quantile value threshold (authorship based on (SHAH, 2022) sample). 103
(a) Code 103
(b) Plot 103
Figure A.6 Code and resulting table of the prediction values and the CPI's bounds values (authorship based on (SHAH, 2022) sample). 104
(a) Code 104
(b) Table 104
Figure A.7 Code and resulting scatter plot of the prediction values and the CPI's bounds values (authorship based on (SHAH, 2022) sample). 105
(a) Code 105
(b) Table 105

List of tables

Table 4.1 Prediction Error Metrics 59
Table 4.2 Resume CPI Metrics 60

List of Abreviations

ARCH – Auto-Regressive Conditional Heteroscedasticity

APP – Asset Price Prediction

ARMA – Auto-Regressive Moving Average

BVSP – Bolsa de Valores de São Paulo

CP – Conformal Prediction

CPE – Conformal Prediction Ensemble

CPI – Conformal Prediction Interval

CQR – Conformalized Quantile Regression

CS – Conventional Statistics

EMH – Efficient Market Hypothesis

FA – Fundamental Analysis

GAN – Generative Adversarial Network

HFT – High-frequency trading

HR – Heuristic Rules

HLPV – High-and-Low Price Values

IBOV – Bovespa Index

LSTM – Long Short Term Memory

MEP – Market Entry Price

MLP – Market Leave Price

ML – Machine Learning

NProhet – Neural Prophet

OHLC – Open-high-low-close

QT – Quantitative trading

RA – Random Approach

SOTA – State-of-the-Art

SPX – Standard and Pool Exchange

ST – Swing Trading

SVM – Support Vector Machine

TA – Technical Analysis

XGBoost – Extreme Gradient Boosting

Science without religion is lame, religion without science is blind.

Albert Einstein, .

1
Introduction

According to An, Sun e Wang (2022), the global financial market exceeded $90 trillion in capitalization by the year 2021, reflecting the immense scale and complexity of this economic domain. This market involves millions of investors worldwide, each contributing to the dynamic interplay of asset valuation, investment strategies, and financial forecasting. As such, the development of accurate predictive models for asset prices is not only a scientific challenge but also of significant economic relevance.

1.1
Problem

The issue pertains to *assets* transactions within the *financial market*, where investors aim to capitalize on the discrepancy between buying and selling values. The most common *market operations*, or *financial market trades*, are *buy* and *sell*. These trades can be executed regardless of the prevailing *market trends*, which are typically classified as either an *Uptrend* or a *Downtrend*. The entry into a position occurs at a specific *market entry price (MEP)*, while the exit is made at a defined *market leave price (MLP)*. The most widespread strategy for generating *returns* (i.e., making a profit) is to *buy low and sell high*, referred to as *going long*. However, more experienced investors also use the inverse strategy—*going short*—which consists of *selling high and buying back low*, allowing profits in declining markets.

At each time step, traders must decide whether to buy, hold, or sell to maximize their net returns. In this context, the *asset return* is defined as the difference between the *current asset price* and its previous price. From an operational perspective, it represents the realized gain or loss resulting from an asset's purchase and subsequent sale.

Various strategies are employed to support such trading decisions, individually or in combination. These include Conventional Statistics (CS), Fundamental Analysis (FA), Technical Analysis (TA), Heuristic Rules (HR), and Machine Learning (ML).

CS makes use of traditional statistical models, such as Auto-Regressive Moving Average (ARMA) (BOX; PIERCE, 1970), and Auto-Regressive Conditional Heteroscedasticity (ARCH) (ENGLE, 1982). The latter addresses issues where the *variance of the time series is not constant*, a phenomenon known as *Heteroscedasticity* (or *Heteroskedasticity*).

FA entails analyzing a company's internal and external performance indicators, including management strategies and financial reports such as income statements, balance sheets, assets, liabilities, and cash flows.

TA identifies patterns and trends through visual and statistical analysis of price and volume data. Standard tools include Simple Moving Averages (SMA), Support and Resistance levels, and trend lines.

HR involves rule-based reasoning drawn from practical trading experience and subjective beliefs, often used by discretionary traders.

ML leverages data-driven algorithms to model complex relationships and patterns. These algorithms span several paradigms, including neuron-based (e.g., neural networks), rule-based (e.g., decision rules), kernel-based (e.g., SVM), regression-based, tree-based (e.g., random forests), curve-based (e.g., polynomial fitting), and generative-based (e.g., GANs or VAEs) models.

1.2 Context

This study utilizes real-life daily data from two prominent global stock market indexes (SMI)[1]: the Standard and Poor's 500, referred to as the S&P 500 (SPX), representing the performance of the top 500 companies listed on U.S. stock exchanges, and the Bovespa Index, known as IBOVESPA (IBOV), comprising nearly 100 of the most capitalized and actively traded stocks from Brazilian companies on the B3 S.A - Brasil, Bolsa, Balcão stock exchange. As of December 2023, IBOV included 86 assets from 83 companies.

Selvin et al.(2017) classified prediction problems based on trade duration into Short-term (ST), lasting seconds to months; Medium-term (MT), spanning one to two years; and Long-term (LT), exceeding two years. High-frequency trading (HFT), completed in milliseconds, achieves state-of-the-art (SOTA)[2]

[1]The *stock market index (SMI)*, or *index*, is a financial instrument (contract or security) that represents the performance of an asset basket (portfolio). The measurement result is given in points instead of prices, and the calculation is based on the weight of some indicators such as *market-cap*, *revenue*, *floating*, *fundamental*, among others.

[2]Aït-Sahalia et al. (2022) represents the *state of the art (SOTA)* in *High-Frequency Trading (HFT)*, having achieved a prediction accuracy of around 79% for trends and 27% for prices. results but is less popular due to its computational demands (AÏT-SAHALIA et al., 2022).

Day Trading (DT) involves buying and selling within a day, preventing prolonged market exposure. Swing Trading (ST) includes short-term or medium-term operations. The stock exchange provides historical asset price data in the open-high-low-close (OHLC) format, presented as a time series (TS) with specific time units, such as milliseconds, seconds, minutes, hours, or days. This research focuses on the short term, covering operations lasting a few minutes or hours.

1.3 Importance

The *financial market (FM)* serves as a fundamental infrastructure for capital flow and liquidity in modern economies. Its efficiency relies on seamless execution of transactions under varying market conditions, enabling investors to act on perceived opportunities without significant slippage or delay. During bullish or bearish cycles, strategic entry and exit points are essential for aligning investments with anticipated price movements, reinforcing the importance of market accessibility and timing.

In this context, investors aim to identify patterns and optimal market entry and exit points, making *asset price prediction (APP)* a crucial task for maximizing returns. Rather than focusing on the precise future asset price, the goal is to anticipate the price direction or trend reliably. Profits stem from maintaining a consistent strategy where the number of profitable transactions outweighs unprofitable ones over time, regardless of temporary fluctuations.

Given the highly competitive nature of the *FM*, where all participants strive to outperform each other, adopting robust and precise forecasting techniques becomes imperative. As noted by Hassan e Nath (2007), the growth of online trading platforms has lowered entry barriers, empowering even small investors to access markets and achieve significant profits through informed decision-making and timely strategy execution.

1.4 Difficulty

The primary challenge lies in asset prices' chaotic and unstable nature, as highlighted by Lawrence (1997a), making asset price prediction (APP) a highly complex task influenced by numerous observable and latent variables. This complexity arises from inherent characteristics of financial time series, such as high volatility, intrinsic noise, nonlinearity, and nonstationarity. The nonlinearity limits the applicability of traditional statistical techniques, while nonstationarity—manifested through time-varying statistical properties such as mean, variance, and autocovariance—complicates the development of consistent and robust predictive models. Moreover, financial data often exhibit heteroscedasticity, where the error terms or asset returns variance changes over time, further challenging model calibration and interval estimation. In addition, input features typically vary in scale, necessitating normalization or transformation to stabilize training processes, particularly in models based on neural networks.

According to Grandhmal and Patel Gandhmal e Patel (2019), high-frequency fluctuations and abrupt changes in market behavior further complicate trend forecasting. Market trend identification is typically derived from relationships among high, low, and closing price values. For example, if the high-minus-close value exceeds the close-minus-low value, the trend is considered upward; conversely, if the close-minus-low value exceeds the high-minus-close value, the trend is downward. Otherwise, the market is viewed as trending sideways.

Arslan (2022) pointed out that conventional statistical methods are inadequate for capturing the nonlinear structure of financial time series. He also highlighted that neural networks, while more expressive, often suffer from overfitting and require careful hyperparameter tuning, which can be both time-consuming and nontrivial.

The debate surrounding market efficiency remains a foundational topic in financial economics. According to the Efficient Market Hypothesis (EMH), proposed by Fama (1970), markets fully reflect all available information, making it impossible to consistently achieve above-average returns through arbitrage or technical analysis. However, several scholars and practitioners argue that markets are not always efficient, mainly due to behavioral biases, liquidity constraints, and persistent anomalies. In particular, Cornell (2013) presents a thorough critique of market efficiency, emphasizing that structural and informational frictions can sustain inefficiencies over time, offering opportunities for improved forecasting techniques.

Another intrinsic difficulty of APP is its online nature—decisions must be made sequentially as new data arrives, without access to future observations. This scenario falls under the category of online problems, where algorithms must operate without complete information. Therefore, investors must make real-time decisions based on partial and evolving knowledge, while asset prices exhibit behavior akin to a random walk. This randomness implies that past price movements provide only limited predictive power for future prices, reinforcing the challenge of accurate forecasting in financial markets.

The overarching challenge is to develop robust prediction mechanisms that support strategic decision-making to maximize returns while minimizing risk exposure. These mechanisms must address the complexities of chaotic dynamics, evolving data distributions, market inefficiencies, and financial systems' unpredictability in theoretical formulation and empirical implementation.

1.5 Prediction Techniques

In financial markets, all models that assist traders in making decisions, especially regarding asset prices and trend prediction, are called Quantitative Models (QM). These models are grounded in mathematics and statistics and aim to support informed, data-driven decision-making. It is important to note that "Quant" models, based on deterministic classical computing, should not be confused with "Quantum" models, which rely on stochastic principles derived from quantum mechanics. Although portfolio optimization is also relevant for QMs, this work focuses specifically on the prediction dimension. As noted by An, Sun e Wang (2022), the last decade has seen a significant rise in quantitative trading (QT), automatically generating trading signals using data-driven methodologies.

1.5.1 Heuristic Rules

Heuristic Rules (HR) refer to empirical strategies developed from a trader's experience, intuition, or personal beliefs. These rules typically lack statistical rigor or generalizability but are widely used by financial professionals. Traders often act on these heuristics with firm conviction, and when such rules are encoded into systems to assist decision-making, they effectively become Quantitative Models (QM). An, Sun e Wang (2022) highlights that traditional quantitative trading methods are often built upon either heuristic logic or human-guided predictive models, further validating the relevance of HR within the predictive modeling landscape.

1.5.2 Conventional Statistics

The prediction of asset prices was initially tackled using Conventional Statistics (CS) and econometrics. These methods rely on strong assumptions such as linearity, normality, and stationarity, which are often misaligned with the non-linear and dynamic nature of financial time series. In contrast, Machine Learning (ML) methods learn patterns directly from historical data without imposing strict assumptions about its distribution or behavior.

In the statistical modeling domain, the Auto-Regressive Moving Average (ARMA) framework serves as a classical linear technique for forecasting, including Asset Price Prediction (APP). Meanwhile, models like Auto-Regressive Conditional Heteroscedasticity (ARCH) and its extensions aim to predict the

variance of asset prices, capturing typical volatility clustering in financial markets.

Despite their historical importance, these techniques have key limitations, particularly their inability to effectively handle non-stationarity, where properties like the mean or variance change over time. According to Tambi (2005), such limitations hinder their practical predictive accuracy in financial environments characterized by volatility and complex dynamics, leading to a growing preference for ML-based alternatives.

1.5.2.1 Machine Learning

With the advancement of computing power over the last two decades, Machine Learning (ML) has gained prominence as a robust approach to Asset Price and Trend Prediction. Unlike Conventional Statistics, ML techniques are more adaptable and can achieve higher accuracy, especially when trained on large-scale datasets.

The most common ML methods include neuron-based models (such as neural networks), rule-based systems, neighbor-based algorithms (like K-Nearest Neighbors), kernel-based models (e.g., Support Vector Machines), regression-based approaches (such as linear or logistic regression), tree-based models (e.g., decision trees and random forests), curve-based techniques (e.g., spline regression), ensemble-based strategies (like boosting and bagging), and generative-based models (such as GANs and VAEs). Each paradigm offers unique strengths in capturing complex patterns inherent to financial time series.

1.6 Uncertainty Quantification

Machine Learning (ML) models, including state-of-the-art (SOTA) architectures, inherently yield uncertain predictions. This uncertainty arises from factors that degrade performance, such as noise, randomness, volatility, and data instability. In practical scenarios like asset price prediction (APP), it becomes essential to quantify this uncertainty to inform decision-making. In other words, beyond predicting outcomes, it is crucial to estimate the level of confidence or reliability associated with those predictions.

Uncertainty Quantification (UQ) measures the uncertainty associated with model outputs in a probabilistic context. It provides a way to assess how likely the predictions reflect actual outcomes. As emphasized by Kabir et al.

(2018), UQ plays a significant role across scientific and engineering applications and has led to the development of various calibration techniques.

1.6.1 Conventional Quantification

Several traditional approaches exist for Uncertainty Quantification (UQ), each with notable limitations. Standard methods include classification probabilities, which often suffer from poor calibration; Bayesian posterior intervals, which require strong assumptions about the underlying distribution—typically assuming normality in finance; and bootstrapping methods, which may misestimate variance during model re-estimation. Other calibration strategies include Platt Scaling, Isotonic Regression, Spline Calibration, Ensemble Methods, and Direct Interval Estimation, though they frequently lack rigorous probabilistic guarantees.

Furthermore, as noted by Romano (2022), many ML models assume that data is independent and identically distributed (i.i.d), implying mutual independence and identical probability distributions across instances. While this assumption simplifies modeling and inference, it rarely holds in real-world financial applications, where temporal dependencies, volatility shifts, and structural breaks are common. Consequently, even high-performing models can produce unreliable predictions when calibration is not explicitly addressed.

1.6.2 Conformal Prediction

Conformal Prediction (CP), introduced by Gammerman, Vovk e Vapnik (1998), offers a principled framework for generating prediction intervals with formal probabilistic guarantees. Unlike conventional methods, CP is model-agnostic and imposes no strict assumptions on the data distribution or the internal mechanics of the prediction model. This makes it highly versatile and applicable across various use cases.

CP benefits Black-Box models, such as deep neural networks, where internal interpretability is limited. It enables uncertainty quantification by constructing Conformal Prediction Intervals (CPI), providing reliable and statistically valid estimates of prediction confidence. Importantly, CP offers non-asymptotic coverage guarantees over finite samples without requiring the data to follow specific distributions like the normal distribution.

The method operates by splitting the original dataset into training and calibration sets. The ML model is trained only once on the training portion, while the calibration data is used separately to derive the prediction intervals.

This separation ensures that the uncertainty estimates are based on data not used during model fitting, supporting better generalization.

Ultimately, CP delivers rigorous and practical uncertainty quantification, supporting decision-making in high-risk domains such as finance. Although its guarantees are statistical rather than deterministic, they are sufficient to underpin confidence-based actions in real-world predictive modeling.

1.7 Gap

Manokhin (2022) emphasized that most Machine Learning (ML) models suffer from poor calibration. He warned that proper calibration poses significant risks in high-stakes domains such as healthcare, finance, autonomous driving, and pharmaceuticals. Uncalibrated models can lead to overconfident or misleading predictions in these contexts, thereby impairing critical decision-making processes.

Caruana e Niculescu-Mizil (2006), Guo et al. (2017), Johansson e Gabrielsson (2019), and Mukhoti et al. (2020). These studies have consistently shown that many ML models produce probabilistic outputs not aligned with actual outcome frequencies, leading to unreliable uncertainty quantification.

This gap is particularly concerning in the context of Asset Price Prediction (APP), where poorly calibrated ML models are frequently employed. Given the financial implications of misestimation, robust calibration methods are essential to improve reliability and risk assessment in APP scenarios.

1.7.1 Contribution

To address the calibration gap identified in predictive models, we propose the development of a Conformal Prediction Ensemble (CPE) aimed at calibrating the prediction outputs of *Machine Learning (ML)* models in the context of *Asset Price Prediction (APP)*. This approach seeks to improve the reliability of predictions, particularly in scenarios where poorly calibrated outputs may compromise financial decision-making.

Drawing inspiration from the concept of a methodological "melting pot" as advocated by Tibshirani e Hastie (2021) and echoed by Romano (2022), we adopt a hybrid strategy that harmonizes diverse modeling philosophies. Both authors suggest that integrating data-driven and model-based approaches can lead to more robust solutions, especially in complex real-world domains such as finance.

The ensemble framework is composed of the following four key methodological components:

- **Heuristic Rules (HR)**: Leveraging expert-driven strategies grounded in practical financial knowledge, we adopt heuristic reasoning to forecast the *close price value (CPV)* of assets. While not statistically formalized, practitioners widely use these heuristics and often reflect market dynamics effectively.
- **Machine Learning (ML)**: We incorporate three different ML models—eXtreme Gradient Boosting (XGBoost), Long-Short Term Memory (LSTM), and Support Verctor Machine (SVM), to generate the initial prediction outputs, referred to as *Original Asset Price Predictions (OAPP)*. These predictions, while accurate in many cases, lack reliable uncertainty quantification.
- **Conformal Prediction (CP)**: To calibrate the OAPP results, we apply two distinct techniques from the CP family: Conformalized Quantile Regression (CQR) (ROMANO; PATTERSON; CANDèS, 2019) and Mondrian Conformal Prediction (MCP) (VOVK; PAPADOPOULOS; GAMMERMAN, 2005). These methods transform the raw outputs into probabilistically valid *Conformal Prediction Intervals (CPI)*, addressing the known issue of poorly calibrated ML predictions.
- **Randomization Approach (RA)**: As a final step, we incorporate a randomization mechanism that selects a calibrated *CPV* from within the predicted CPI using a uniformly random sampling strategy. This adds a layer of diversity and robustness to the ensemble output.

Further methodological details, including the execution sequence and rationale for selecting or omitting each technique, are presented in Section 3.

1.8 Structure

The remaining part of this research is organized as follows: Section 2 presents the literature review for *asset price prediction (APP)* using *ML* and the variants related to this research; Section 3 provides the definitions of the main experiments and their sequence; Section 4 presents the model description and model formalization; Section 5 describes the experiment execution using different *ML* techniques and the *two* dataset benchmarks; Finally, Section 6 concludes this paper, adding considerations and future research.

2
Literature Review

Using numerous approaches, the *asset price prediction (APP)* is a vast research topic. For this reason, we narrowed the literature review to what is closer to and around the prediction models and calibration techniques we chose to use.

2.1
Prediction Models

This section details the relevant research related to both *asset price prediction (APP)* and *asset trend prediction (ATP)* using the models **LSTM**, **SVM**, and **XGBoost**.

The early foundations of asset price prediction (APP) were shaped by the random walk theory and the Efficient Market Hypothesis (EMH). Cootner (1964) observed that daily asset prices followed a *random-walk* behavior, suggesting unpredictability in financial markets. Subsequently, Fama (1965) posed a crucial question about the possibility of identifying predictive patterns in historical price data, a concern that continues to influence modern financial modeling. He argued that historical data would not offer a consistent advantage in forecasting if markets were truly efficient, as posited by the *Efficient Market Hypothesis (EMH)*.

Challenging this position, several studies published between 2003 and 2011 presented empirical evidence contradicting the EMH and the random walk hypothesis. These included works by Malkiel (2003), Smith (2003), Jr e Parker (2007), and Bollen, Mao e Zeng (2011), all of which highlighted anomalies, behavioral biases, and social sentiment as potential predictive features.

The exploration of artificial intelligence in finance began with White (1988), who appears to be the first to apply *Neural Networks (NN)* to predict daily asset returns for IBM. His optimistic results demonstrated the potential of NN for APP, marking a key turning point in the field. Building on this, Roman (1996) investigated the use of both backpropagation and *Recurrent Neural Networks (RNNs)* to predict asset trend prediction (ATP) across multiple international markets. Their results showed that while recurrent architectures captured temporal dependencies, the improvement over traditional feedforward models was not yet substantial, foreshadowing the later breakthroughs made possible by LSTM.

2.1.1 LSTM

Lawrence (1997) challenged the Efficient Market Hypothesis (EMH) by highlighting the difficulty of accurately predicting short-term stock movements, prompting interest in models capable of capturing the stochastic behavior of financial time series.

Hochreiter and Schmidhuber (HOCHREITER; SCHMIDHUBER, 1997) introduced the Long Short-Term Memory (LSTM) network to address limitations of standard RNNs, particularly vanishing gradients. Using input, output, and forget gates, their architecture enabled long-term sequence modeling and became foundational for time-dependent prediction tasks such as asset price and trend forecasting.

Subsequent studies expanded LSTM's role in financial modeling. Kara et al. (KARA; BOYACIOGLU; BAYKAN, 2011) demonstrated improved directional accuracy when combining LSTM with technical indicators. Chen et al. (CHEN; ZHOU; DAI, 2015) confirmed LSTM's effectiveness for intraday forecasting using high-frequency data, while Fischer and Krauss (FISCHER; KRAUSS, 2018) showed it outperformed MLPs by capturing nonlinear dependencies without manual feature engineering.

Hybrid approaches also gained traction. Nelson et al. (NELSON; PEREIRA; OLIVEIRA, 2017) integrated sentiment analysis into LSTM models to enhance predictions under volatility. Qiu et al. (QIU; SONG; AKAGI, 2020) implemented ensemble LSTMs with varied lag structures, improving generalization and robustness in trading environments. Cao et al. (CAO; LI; LI, 2021) introduced attention mechanisms, enabling LSTM to dynamically weight relevant time steps, which enhanced interpretability and accuracy.

These advances illustrate LSTM's adaptability for financial forecasting, from core sequence modeling to hybrid and attention-based strategies. Its capacity to capture temporal dependencies and integrate diverse inputs makes it a strong candidate for robust APP solutions under varying market conditions.

Additional information about LSTM can be found in Appendices A.2.1.2 and A.1.2.

2.1.2 SVM

Support Vector Machines (SVMs) have played a significant role in stock market forecasting since the early 2000s, offering robustness in modeling the nonlinear, noisy nature of financial time series. Tay and Cao (TAY; CAO,

2001) demonstrated that SVM outperformed traditional neural networks, particularly when working with limited and volatile data.

Kim (KIM, 2003) explored SVM's ability to classify stock index direction and emphasized the role of input feature selection in enhancing predictive performance. Huang et al. (HUANG; NAKAMORI; WANG, 2005) further improved results by integrating SVM with Genetic Algorithms (GA) for feature selection and model optimization, inspiring hybrid SVM frameworks in financial domains.

Cao and Tay (CAO; TAY, 2005) refined hyperparameter tuning for SVR models, proposing an optimization framework that yielded more stable predictions across indices. Thakur et al. (THAKUR; PADMANABHAN; GUPTA, 2011) introduced Particle Swarm Optimization (PSO) for parameter search, showing advantages over conventional methods.

Patel et al. (PATEL et al., 2015) proposed an ensemble combining SVM, ANN, and Random Forests, where SVM showed competitive accuracy when properly optimized. Ghanbari and Goldani (GHANBARI; GOLDANI, 2021) applied the Golden Sine Algorithm (GSA) for SVR tuning, reporting enhanced predictive accuracy and efficiency.

These developments underscore SVM's evolution from benchmark predictive models to sophisticated hybrid systems. Its continued relevance in financial forecasting stems from strong generalization, adaptability, and compatibility with modern optimization techniques.

Additional information about the *SVM* model can be found in Appendices A.2.1.3 and A.1.4.

2.1.3 XGBoost

XGBoost, developed by Chen and Guestrin (CHEN; GUESTRIN, 2016a), is a high-performance, regularized gradient boosting algorithm known for its scalability and predictive strength. Its adoption in financial forecasting has grown due to its robustness with structured, noisy datasets.

Ballings et al. (BALLINGS et al., 2015) demonstrated XGBoost's superior classification accuracy over logistic regression and neural networks. Patel et al. (PATEL et al., 2015) confirmed its regression accuracy using technical indicators, while their follow-up work (PATEL; SHAH; KOTECHA, 2019) showed enhanced performance by including macroeconomic variables during turbulent periods.

The model's flexibility has been extended with domain-specific features. Zhang et al. (ZHANG; XU; WANG, 2018) and Chakraborty et

al. (CHAKRABORTY; GHOSH, 2021) incorporated sentiment data, and Jabeur et al. (JABEUR; LAHMIRI; HUSSAIN, 2020) applied XGBoost to the volatile cryptocurrency market. Dong et al. (DONG; YU; LIU, 2022) introduced a multi-resolution approach combining short- and long-term temporal views, improving adaptability to nonstationary behavior. Januschowski et al. (JANUSCHOWSKI et al., 2022) further validated the dominance of boosted trees in time series forecasting competitions. Verma et al. (VERMA; AGRAWAL; SHARMA, 2023) contributed interpretability by integrating SHAP values to highlight influential variables such as volatility and volume.

Despite its predictive power, XGBoost lacks native support for uncertainty quantification—a limitation in risk-sensitive financial environments. To address this, we integrate Conformal Prediction (CP) into the model, enabling distribution-free calibration of predictive intervals.

Further implementation details of *XGBoost* are available in Appendices A.2.1.4 and A.1.6.

2.1.4 Addition Review

Huck (2009) proposed a pairs trading framework to identify undervalued and overvalued assets using multi-criteria decision techniques applied to S&P100 components, contributing early advances to directional forecasting. Jacobs (2015) cataloged 100 market anomalies, expanding the landscape of factors in financial modeling.

This work intentionally excludes several families of models based on empirical limitations. Conventional statistical methods such as ARIMA and GARCH (BOX et al., 2015; BOLLERSLEV, 1986) offer interpretability and volatility modeling but fall short under nonlinear dynamics and nonstationary conditions (Cont, 2001; Christoffersen, 2012). Their parametric assumptions and lack of flexible uncertainty modeling limit practical deployment in modern financial forecasting.

Convolutional Neural Networks (CNNs), although successful in image and short-term time series tasks (Borovykh, 2017), lack intrinsic mechanisms for long-range temporal dependency modeling. Hybrid architectures incorporating CNNs with LSTMs or attention layers (QIN et al., 2017) have been proposed, but pure CNNs struggle with trend detection, seasonality, and probabilistic forecasting—key to financial decision-making (ANGELOPOULOS; BATES; FANNJIANG, 2021).

Large Language Models (LLMs), while transformative in NLP, are not well-aligned with structured temporal prediction. Studies like (LUO et al.,

2023) and Zhang (2023) find specialized time series models like LSTM, PatchTST (LIU et al., 2023), and Temporal Fusion Transformers (LIM; ZOHREN, 2021) outperform LLMs in this domain. LLMs also lack native structures for autocorrelation, lagged dependencies, or calibrated uncertainty—critical aspects for financial risk-aware applications (ANGELOPOULOS; BATES; FANNJIANG, 2021).

Thus, while these excluded methods have theoretical and niche value, our focus remains on empirically robust, interpretable, and uncertainty-aware approaches that align with financial markets' stochastic and regime-shifting nature.

2.2 Calibration Techniques

In this session 2.2, we detail some relevant research related to model calibration using the *Conformal Prediction (CP)*. Although the *CP* can be applied to calibrate the prediction of various *Machine Learning (ML)* algorithms, we focus on just the regression case.

2.2.1 Conformal Prediction

Conformal Prediction (CP) was initially introduced by Vovk, Gammerman e Shafer (2005a) and formalized into a practical uncertainty quantification method by Shafer e Vovk (2008a), offering finite-sample validity guarantees. CP provides prediction sets that maintain user-specified coverage probabilities, regardless of the underlying data distribution, making it particularly attractive for financial time series where assumptions such as normality and independence are frequently violated.

In a pivotal contribution, Romano, Patterson e Candès (2019) proposed Conformalized Quantile Regression (CQR), which integrates CP with quantile regression to generate prediction intervals with formal coverage guarantees. CQR enables the model to adapt to heteroscedastic data, a common trait in financial time series, by directly estimating lower and upper quantiles and conformally adjusting them.

Later, Gibbs, Candès e Lei (2021) extended CP to handle non-exchangeable data through Mondrian Conformal Prediction (MCP). MCP partitions the data into groups (e.g., time buckets or volatility regimes) and applies CP within each group independently. This ensures coverage even under distributional shifts, a valuable enhancement for dynamic financial environments.

In the energy and price prediction context, Kath e Ziel (2021) compared CP with the state-of-the-art Quantile Regression Averaging (QRA) model. Their empirical results demonstrated that CP-based methods provided more robust coverage across error distributions and time scales.

Wiśniewski, Jastrzębski e Olszewski (2020) applied CP to financial time series forecasting, benchmarking it against classical models such as Moving Average (MA) and Quantile Regression (QR). Their findings showed that CP improved interval sharpness and maintained theoretical coverage under distributional uncertainty.

For volatility forecasting, Chernozhukov e Wüthrich (2021) used CP to construct predictive sets based on volatility-conditioned stock returns, demonstrating improved performance compared to traditional econometric models.

Romano, Sesia e Candes (2022) revisited CP in financial applications, providing theoretical arguments and experimental evidence in favor of its adoption in asset price prediction tasks. The authors emphasized the importance of designing informative nonconformity scoring functions (NCS) to obtain efficient and narrow predictive intervals, highlighting this as a central challenge in practical implementations.

These foundational and recent advancements underscore CP's growing relevance in finance, particularly for calibrating machine learning models and managing predictive uncertainty in volatile or non-stationary environments.

Further discussion and detailed reviews of calibration techniques, including CP, are available in Appendix A.2.2.

3
Methodology

This chapter presents the proposed methodology and outlines the rationale behind selecting and excluding specific techniques for asset price prediction (APP). We adopt a hybrid strategy inspired by the "melting pot" philosophy advocated by Tibshirani e Hastie (2021) and Romano (2022), which encourages the integration of complementary models and paradigms.

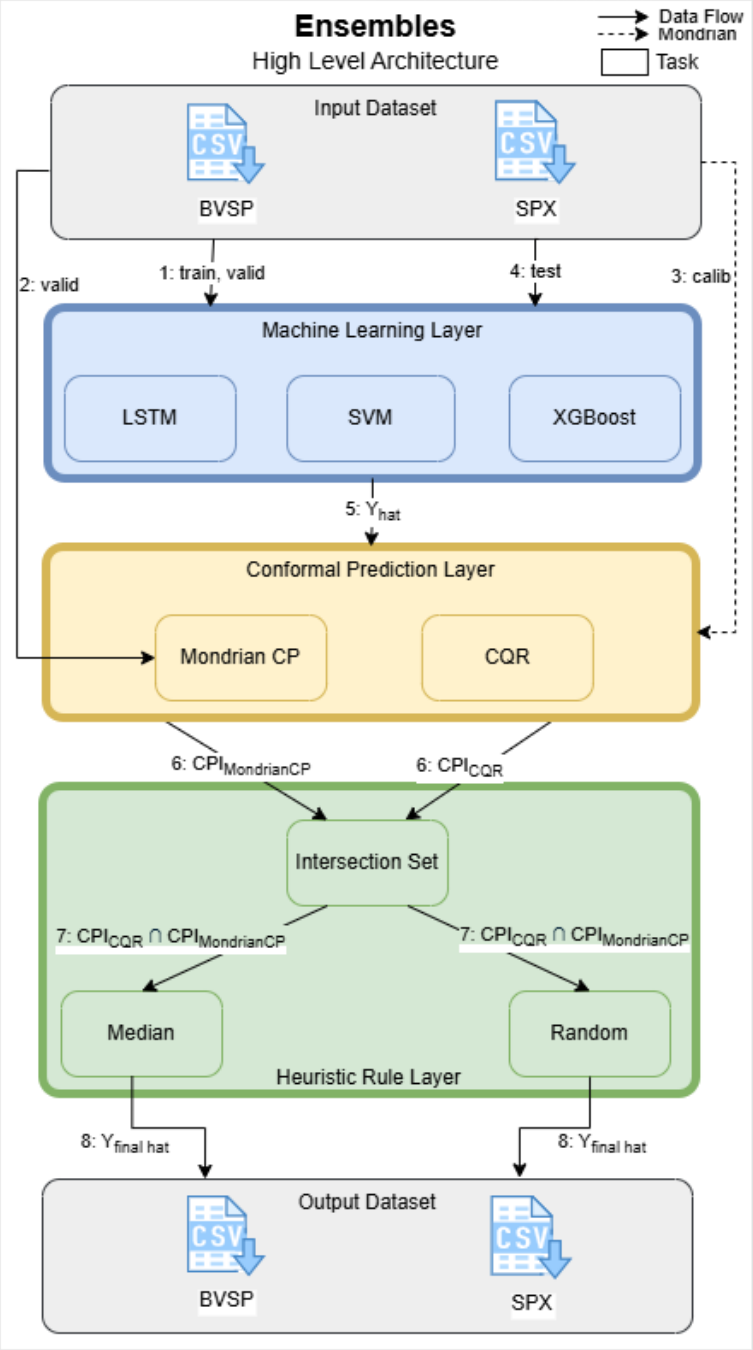


Figure 3.1: Illustration of our multi-model ensemble pipeline

Figure 3.1 illustrates the integrated architecture of our proposed ensemble, which guides our experimental design, combining three methodological pillars as the following layers:

- **Machine Learning Layer (MLL)** - (in blue)
- **Conformal Prediction Layer (CPL)** - (in yellow)
- **Heuristic Rules Layer (HRL)** - (in green)

Figure 3.1 illustrates the three architectural layers and the sequential flow of eight key steps, as detailed below:

1. **Training:** This is a standard task for any machine learning (ML) model, involving no modifications to the fitting process. The model is trained using the training data slice, while the validation data slice monitors generalization performance and avoids overfitting. Each asset is associated with a distinct input dataset, allowing the models to learn asset-specific patterns and dynamics independently. The input dataset is split into four slices for training (*train*), validation (*valid*), calibration (*calib*), and testing (*test*). The proportion of them are 70%, 10%, 10%, and 10%, respectively.

2. **Categorization:** This step is specific to the Mondrian Conformal Prediction (MCP) method. It utilizes the validation dataset to fit the structural partitions the Mondrian framework requires, effectively categorizing the input space. The term *fit* follows the `scikit-learn` API convention.

3. **Calibration:** In this step, both CP methods utilize the calibration data slice to determine the nonconformity scores required for interval generation. The quantile-based calibration is performed at a confidence level of $1 - \alpha$, where $\alpha = 5\%$.

4. **Test Ingestion:** This is a standard step in predictive modeling workflows, where the ML models ingest the test data slice in preparation for generating predictions.

5. **Initial Prediction:** At this stage, the ML models produce point predictions for the test data. This step is performed independently of the CP mechanisms and represents the raw, uncalibrated output.

6. **CPI Generation:** Specific to the conformal prediction framework, this step generates the Calibrated Prediction Intervals (CPI). It leverages the initial point predictions from Step 5 along with the categorization and calibration from Steps 2 and 3, respectively.

7. **Intersection Set:** This step initiates the proposed heuristic. It computes the intersection of all CPI intervals produced by the various CP techniques. The resulting interval tends to be narrower—or potentially null—as in any classical set intersection, as illustrated in Figures 3.2a and 3.2b.

8. **Final Prediction:** This is the concluding step of the proposed heuristic. Based on the interval produced by the intersection, two distinct ensemble strategies are applied:

   – **Ensemble-M (Median-based):** Computes the final prediction by calculating the median of the lower and upper bounds of all CPI intervals obtained in the previous step.
   – **Ensemble-R (Random-based):** Samples the final prediction uniformly at random from within the interval produced by the Intersection Set step.

3.1 ML Layer

This layer addresses the task of asset price prediction (APP), the close, through the use of three machine learning models, each representing a distinct class of modeling approaches, and different architectural paradigms:

– **Long Short-Term Memory (LSTM)**
– **Support Vector Machine (SVM)**
– **Extreme Gradient Boosting (XGBoost)**

Support Vector Machine (SVM), *Long Short-Term Memory (LSTM)*, and *eXtreme Gradient Boosting (XGBoost)*. This diversity ensures we capture temporal dependencies, non-linear patterns, and structural flexibility across different learning paradigms. For the SVM model, the configuration included a regularization parameter $C = 100.0$, an epsilon-tube of 0.01 for regression margin, a radial basis function (RBF) kernel, and a kernel coefficient $\gamma = 0.001$.

The Support Vector Machine was introduced by Cortes e Vapnik (1995a) as a robust model for classification and regression tasks. We adopt the Support Vector Regression (SVR) variant for our use case, which is well-suited for modeling noisy, non-linear relationships commonly observed in financial time series. Its kernel-based projection into high-dimensional spaces enables the model to identify optimal hyperplanes that generalize well, mitigating overfitting and improving forecast stability.

The Long Short-Term Memory network, developed by Hochreiter e Schmidhuber (1997), is a recurrent neural network (RNN) designed to learn long-term dependencies in sequential data. Its architecture—featuring input, forget, and output gates—effectively addresses the vanishing gradient problem and is particularly useful for time series tasks involving complex temporal dynamics. LSTM has become a benchmark model in financial contexts because it can model non-stationary and volatile data sequences. Our implementation follows best practices in the literature, leveraging LSTM's memory capabilities to forecast future asset values. The LSTM model was configured with a learning rate 0.001, trained for a single epoch, and employed two hidden layers of 50 units each. It used a batch size of 1 and an early stopping threshold of 15 epochs. The optimizer was set to stochastic gradient descent (SGD), and the loss function was Quantile Loss with quantile levels of 0.1, 0.5, and 0.9. The quantile error tolerance for conformal calibration was fixed at 0.05. Evaluation metrics included MAE, RMSE, MAPE, and SMAPE.

XGBoost, introduced by Chen e Guestrin (2016b), is a tree-based ensemble method that builds upon gradient boosting principles. Its scalability, sparsity-aware optimization, and ability to capture intricate data interactions make it highly effective for structured data. XGBoost has demonstrated superior performance in a range of predictive modeling competitions. Januschowski et al. (2022) confirmed the dominance of Gradient Boosted Decision Trees (GBDT) in international forecasting challenges, such as the M5 Competition, highlighting their suitability for time-series forecasting. The XGBoost model used a learning rate of 0.3, a maximum depth of 10, and 500 estimators. Additional hyperparameters included a minimum child weight of 1, L2 regularization λ set to 5, early stopping with a patience of 30 iterations, and full feature and row sampling (i.e., both 'colsample_bytree' and 'subsample' set to 1.0). Like LSTM, it used Quantile Loss at levels 0.1, 0.5, and 0.9 with an error tolerance of 0.05, and evaluation based on MAE, RMSE, MAPE, and SMAPE.

These models were selected not only for their strengths but also for their complementary characteristics. Together, they enable a diversified modeling strategy to enhance the robustness and accuracy of our asset price prediction ensemble.

This layer ingests only the training (*train*) and validation (*valid*) subsets of the data to perform model fitting.

Additional implementation details for each model can be found in the Appendices: A.1.4 for SVM, A.1.2 for LSTM, and A.1.6 for XGBoost.

3.2 CP Layer

To address the poor calibration of machine learning models in asset price prediction (APP), we adopt the Conformal Prediction (CP) framework introduced by Gammerman, Vovk e Vapnik (1998). CP offers model-agnostic, non-asymptotic prediction intervals with formal probabilistic guarantees, making it well-suited for financial applications where uncertainty quantification is critical.

Modern ML models typically yield point predictions, called Scored Prediction Results (SPR), which are often poorly calibrated and lack confidence bounds. To rectify this, we calibrate the original asset price prediction (OAPP) from each model using CP, producing Conformal Prediction Intervals (CPI) that represent calibrated asset price predictions (CAPP). These intervals quantify prediction uncertainty as a bounded range (multi-point set) or, in rare cases, an empty set.

We utilize two CP-based techniques to address the calibration as follows:

- **Conformalized Quantile Regression (CQR)**
- **Mondrian Conformal Prediction (MCP)**

The CQR proposed by Romano, Patterson e Candes (2019) combines quantile regression with CP guarantees, while Mondrian Conformal Prediction (MCP), introduced by Vovk, Papadopoulos e Gammerman (2005), is tailored for non-exchangeable data distributions. Both methods are used independently to calibrate the SPR output from each ML model.

Our three machine learning models—XGBoost, LSTM, and SVM—predict close price values (CPVs) for two distinct assets, resulting in six CPV predictions. These predictions are then individually calibrated using CQR and MCP, resulting in 12 CPI results (six per CP method).

The quality of a CPI is assessed based on two efficiency criteria: (i) confidence, which increases as the CPI becomes narrower; and (ii) credibility, representing the belief that the actual value falls within the predicted interval.

While CP offers strong theoretical guarantees, its application requires a hold-out calibration set, which reduces the data available for training the model. Despite this limitation, the interpretability and reliability of the resulting CPI make CP a valuable component of our prediction pipeline.

This layer ingests only the validation (*valid*) and calibration (*calib*) subsets of the data to perform model fitting. The Mondrian Conformal Prediction (MCP) method exclusively uses the validation dataset. In this context, the validation data is not used for performance validation, but rather

to fit and categorize the Mondrian structure, enabling conditional calibration based on data attributes.

3.3 HR Layer

This study incorporates *heuristic rules (HR)* through two complementary strategies: domain-informed modeling and randomized interval sampling. The first strategy draws from empirical trading practices, where experienced traders recognize that asset prices are more likely to traverse a price range during a trading session rather than settle at an exact target value. Based on this insight, we implement a heuristic rule that prioritizes the likelihood of intraday price occurrences over precise closing prices.

The second strategy introduces stochastic diversity into the prediction process by sampling uniformly within the bounds of the Conformal Prediction Interval (CPI), which is produced by conformal calibration. This randomization technique generates plausible point estimates inside the CPI, enabling the construction of diversified and statistically grounded predictions.

We employ a heterogeneous ensemble approach to enhance forecasting accuracy, combining outputs from three distinct machine learning models: XGBoost, LSTM, and SVM. Rather than relying on complex meta-modeling such as stacking, we use a naive ensemble strategy based on arithmetic aggregation. While stacking is often beneficial for classification tasks, it offers limited advantages for regression problems, which is this study's focus.

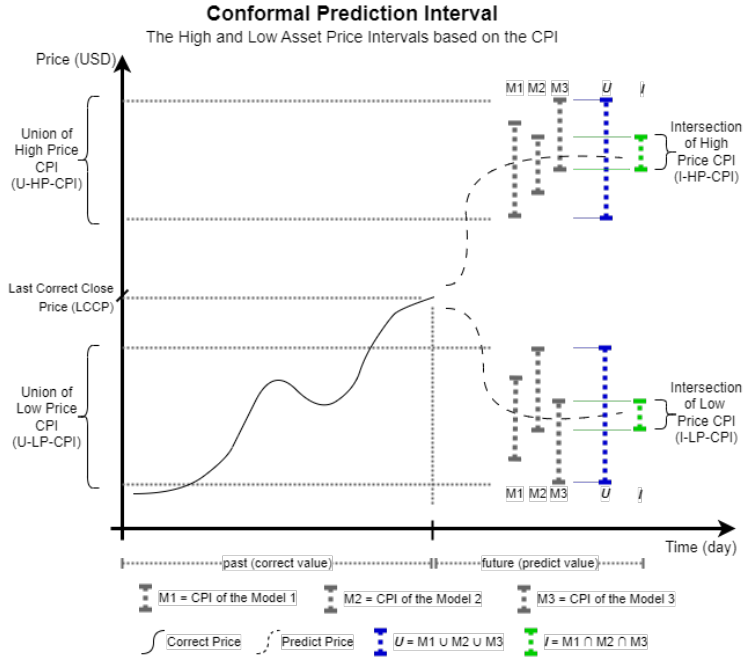
Each base learner's output is calibrated using Conformal Prediction (CP) methods, resulting in six CPIs—three models applied to each of two price types (high and low). We apply a set intersection operation across the CPIs to consolidate these intervals into a single prediction. This yields a narrower interval, referred to as the Subset Conformal Prediction Interval (Subset CPI), which reduces uncertainty and narrows the range of plausible outcomes.

Finally, we apply a uniform random sampling over the Subset-CPI to select the final point prediction. This technique is inspired by the randomization advantage explored by Gupta et. al. (2020) , who demonstrated that randomized selection can improve performance in online decision-making settings. Although our approach does not provide additional probabilistic guarantees beyond those offered by CP, it effectively combines robustness and simplicity to deliver high-quality predictive outcomes.

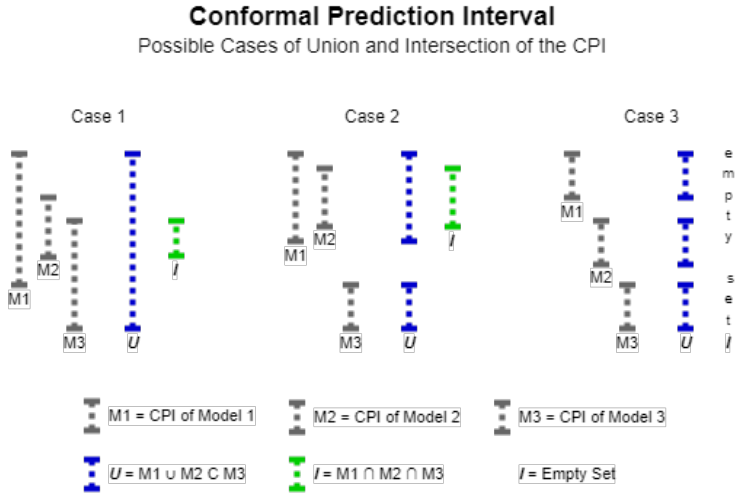
To enhance the reliability of our ensemble predictions, we adopt a set intersection strategy applied to the Conformal Prediction Intervals (CPI) produced by each of the three base Machine Learning (ML) models. This

procedure generates a refined interval, called the Subset Conformal Prediction Interval (Subset CPI), a narrower subset of the original CPI outputs. The goal is to improve prediction precision by minimizing both bias and covariance.

Unlike traditional approaches such as set union—which may inflate the prediction range—the intersection technique is designed to restrict it, resulting in sharper and more focused estimates. Although this method does not maintain the formal probabilistic guarantees provided by standard Conformal Prediction (CP) frameworks, it offers a pragmatic and empirically grounded calibration refinement.



(a) Intersection of the Conformal Prediction Interval (CPI).



(b) Possible Scenarios Cases of the Intersection of the Conformal Prediction Interval (CPI).

Figure 3.2: Intersection of CPI.

Figure 3.2a depicts the critical regions under consideration. The upper-bound corresponds to the high price value (HPV), while the lower-bound pertains to the close price value (CPV). This stratification conceptually parallels Bollinger Bands (BOLLINGER, 2002), which dynamically model market volatility using statistical boundaries.

In the diagram, CPI results from the three ML models, denoted as M1, M2, and M3, are shown in gray. Their intersections form two Subset-CPIs: one for HPV and one for CPV, indicated by green dashed lines. For contextual

reference, blue dashed lines indicate set unions, although this operation has not been adopted in our method due to its tendency to widen the predictive range.

Visual cues such as continuous and dashed curves are added to support interpretability. These highlight the last correct close price (LCCP), the open price value (OPV), and prediction flow, though they are not directly used in the CPI computation.

When CPI sets are disjoint and fail to intersect, we default to selecting the median CPI as the Subset-CPI.

Figure 3.2b outlines three scenarios: complete intersection, partial intersection, and empty set fallback. This intersection mechanism provides a robust strategy for synthesizing multiple conformal predictions into a single, statistically meaningful interval, despite the absence of formal guarantees.

Overall, this methodology allows us to empirically enhance asset price prediction by balancing the trade-off between interval width and coverage confidence, with results evaluated across HPV and CPV boundaries.

In the context of stock market prediction, a heuristic approach considers a prediction successful if the actual price trajectory intersects the predicted value at any point during the trading session, regardless of whether it crosses from above or below. This perspective aligns with practical trading scenarios where an order is deemed executed if the market price reaches the specified level, even if it does not close at that level.

While this heuristic is prevalent in trading practices, it is not typically incorporated into formal error metrics within predictive models. Instead, it serves as an operational criterion for evaluating the effectiveness of predictions in real-world trading environments.

Although this heuristic is widely recognized among practitioners, it is not often discussed in the academic literature. Therefore, while it offers valuable practical insights, it lacks formal theoretical backing in scholarly research.

3.4 Ensemble

As illustrated in Figure 3.1 and detailed in Section 3, both proposed ensemble methods share identical processes in the *Machine Learning (ML)* and *Conformal Prediction (CP)* layers. The distinction lies in the *Heuristic Rule (HR)* layer: *Ensemble-M* selects the final predicted value using the median statistic, whereas *Ensemble-R* selects the value through a random choice within the calibrated prediction interval.

In both ensembles, this selection is performed after applying a common *Intersection Set (IS)*, Figures 3.2a and 3.2b, procedure. This *IS* operation, executed within the *HR Layer*, refines the prediction interval by computing the intersection of all *Conformal Prediction Intervals (CPI)* produced by each CP method. The resulting narrowed interval enhances the precision of the final prediction while preserving the diversity introduced by the ensemble structure.

3.5 Analysis

This section outlines the methodology used to evaluate the performance of *Asset Price Prediction* (APP) and *Conformal Prediction Intervals* (CPI) across a range of experimental scenarios. These scenarios are generated from combinations of four methodological components:

- **Data Benchmark:** Two datasets derived from different stock market indices;
- **Machine Learning (ML):** Three predictive models—LSTM Hochreiter e Schmidhuber (1997), SVM Cortes e Vapnik (1995a), and XGBoost Chen e Guestrin (2016b);
- **Conformal Prediction (CP):** Two CP calibration strategies—CQR Romano, Patterson e Candes (2019) and Mondrian CP Vovk, Papadopoulos e Gammerman (2005);
- **Heuristic Rule (HR):** Two proposed ensemble methods named *Ensemble-M* and *Ensemble-R*.

The primary focus is to evaluate how *conformal prediction* (CP) methods improve the predictive performance of baseline machine learning (ML) models. To account for market dynamics, the experimental setup incorporates directional trends—*Uptrend* and *Downtrend*—and computes prediction error ϵ using the Euclidean norm, as $\epsilon = ||y - \hat{y}||_2$. For upward trends, the error is $||y_{\text{high}} - \hat{y}_{\text{high}}||_2$, and for downward trends, it becomes $||y_{\text{low}} - \hat{y}_{\text{low}}||_2$.

We evaluate the ML model outputs using two standard metrics: *Mean Absolute Percentage Error (MAPE)* and *Symmetric MAPE (sMAPE)*. The analysis compares:

- The three base ML models (LSTM, SVM, and XGBoost) applied to two different assets;
- Two ensemble strategies (Ensemble-M and Ensemble-R) applied to the same assets.

This results in a total of 10 scenarios: (3 ML models+2 ensembles) $\times$ 2 assets = 10.

To evaluate the quality of the CP-calibrated prediction intervals, we use:

- Coverage percentage: measuring how often the true value falls within the predicted interval;
- Interval size: average and median width of the CPI, reflecting informativeness and precision.

We compare two CP techniques (CQR and Mondrian CP) and the two ensemble strategies (Ensemble-M and Ensemble-R), each applied to two ML models over two assets, yielding 16 scenarios: (2 CP methods + 2 ensembles) $\times$ 2 ML models $\times$ 2 assets = 16.

While we use common benchmarks and widely recognized algorithms, our goal is not to rely on external benchmarking. Instead, we perform internal comparisons among baseline CP methods and proposed ensemble variants to provide a replicable foundation for future research.

Additionally, we introduce a secondary heuristic evaluation criterion: a prediction is considered successful if the real price crosses the predicted value during the time window, regardless of the direction. Although this heuristic is not reflected in MAPE or sMAPE metrics, its practical relevance is demonstrated through color-coded visualizations across scenarios.

4
Experiment

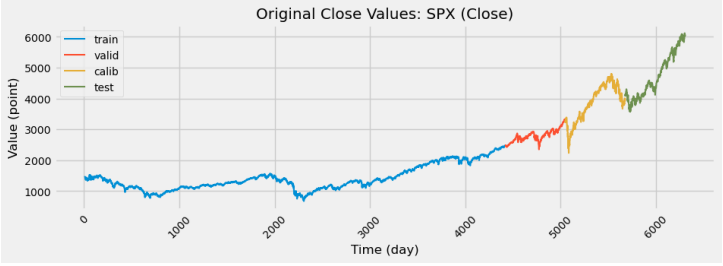
This section describes the experimental setup for evaluating asset price prediction and conformal prediction calibration techniques. All preprocessing procedures—such as data quality checks, normalization, and alignment—are detailed in the Appendix.

The *data partitioning* ensures fair and reproducible model evaluation, each dataset is partitioned into four disjoint subsets such as 70% for training, 10% for validation (used for the ML models and the Nondrian CP technique, 10% for conformal calibration, and 10% for testing. This configuration enables the separation of tasks: model fitting (train), hyperparameter tuning (valid), calibration (calib), and final evaluation (test).

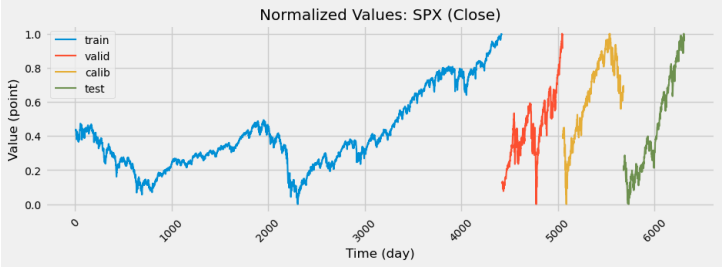
As the *forecasting Strategy*, we adopt a one-step-ahead prediction approach using a backward time window of five trading days. This decision is based on empirical evidence suggesting that, in short-term trading, the most predictive influence arises from the prior five business days. Influence rapidly decays beyond the 5-day mark and is often negligible by the 14th or 15th day, as noted in Suleiman (2023). This is consistent with financial trading heuristics and academic observations in time series forecasting.

We proceed with a *minimal feature set* to isolate the performance of conformal prediction techniques, and we constrain the feature space to a minimal configuration. This reduces confounding effects from extensive feature engineering, ensuring that performance improvements can be attributed primarily to the calibration strategy rather than feature complexity.

Tree-based models such as XGBoost are known to perform poorly when extrapolating beyond the training data range (LUNDBERG et al., 2020). To address this, we applied a value-shifting preprocessing step to align the distributions of all input subsets with the training set, improving prediction stability under distributional shifts. This procedure is detailed in the subsequent Subsection 4.1.



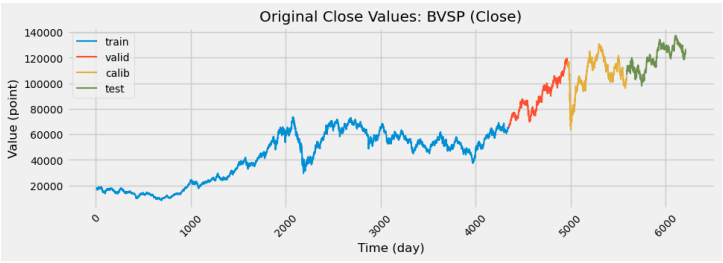
(a) SPX without any shifting.



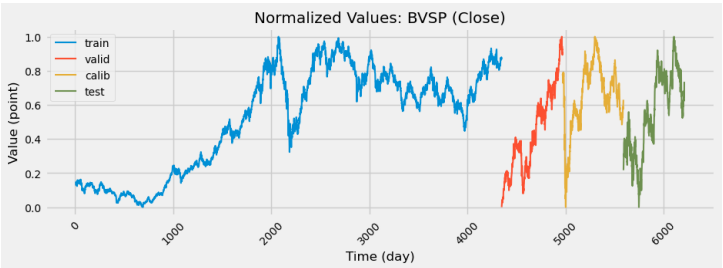
(b) SPX with shifting.

Figure 4.1: SPX dataset with and without shifting.

As the *visual diagnostic*, represented by a distinct colors, Figures 4.1a and 4.2a shows the original unaligned data inputs, while Figures 4.1b and 4.2b presents the post-shifted data input. This transformation ensures that each subset resides within a known operational domain, improving prediction consistency.



(a) BVSP without any shifting.



(b) BSVP with shifting.

Figure 4.2: BVSP dataset with and without shifting.

Both indices, the BVSP and the SPX, exhibit similar movements in their values, suggesting that these two stock markets are subtly correlated and similarly exposed to global influences that drive periods of depreciation and appreciation.

All model hyperparameters follow best practices established by previous literature on stock market forecasting, ensuring methodological rigor and alignment with validated configurations.

4.1 ML Layer

In this subsection, we evaluate the predictive performance of three distinct *machine learning (ML)* models applied to *close price value (CPV)* forecasting. Each model is trained and tested using historical asset price data from two benchmark financial indices: the S&P 500 (SPX) and the Bovespa Index (IBOV). The primary objective at this stage is to generate baseline predictions without conformal calibration, which will subsequently serve as a comparative benchmark for assessing the contribution of *Conformal Prediction* techniques to model performance and uncertainty quantification.

We adopt a one-step-ahead forecasting strategy, as this experiment does not require forecasts extending beyond a single time step into the future. A backward time window of five trading days is used as input for the models, based on empirical evidence suggesting that incorporating additional historical steps beyond this threshold yields diminishing predictive returns (Suleiman, 2023).

The data preprocessing involved comprehensive diagnostic assessments and corrective treatments to ensure data quality and consistency. For the sake of brevity, the detailed procedures are not presented in this work.

The selected machine learning models represent different methodological paradigms, each offering a unique approach to modeling the complex dynamics of financial time series. Specifically, we employ: (i) Long Short-Term Memory (LSTM), a recurrent neural network architecture designed to capture temporal dependencies; (ii) Support Vector Machine (SVM), a kernel-based model capable of handling non-linearities in multi-dimensional feature spaces; and (iii) eXtreme Gradient Boosting (XGBoost), a scalable decision-tree-based ensemble algorithm well-regarded for its high predictive accuracy.

Given the architectural and theoretical diversity among these models, we anticipate that at least one will effectively capture relevant patterns in the data, providing a robust predictive foundation. This diversity is also essential for evaluating the robustness and generalizability of the conformal calibration

procedure between different modeling approaches.

To ensure fair and consistent evaluation, the dataset is partitioned into four disjoint subsets: 70% for training, 10% for validation, 10% for conformal calibration, and 10% for testing. This division separates the processes of model fitting and hyperparameter optimization, prediction calibration, and final performance assessment. The introduction of a dedicated calibration subset is specific to the *Conformal Prediction (CP)* technique, while the remaining factions follow conventional practices commonly employed in Machine Learning (ML) applications.

All models were configured using hyperparameters informed by empirical findings from prior academic studies specifically focused on stock market prediction, such as Gandhmal e Patel (2019), Patel et al. (2015), ensuring alignment with best practices in the literature and maintaining methodological rigor and practical relevance.

Notably, while it is widely recognized that the performance of Conformal Prediction (CP) methods strongly depends on the predictive quality of the underlying base models (ZECCHIN et al., 2024; TENG et al., 2023), we deliberately constrained the set of features to a minimal configuration. This design choice enables a more focused evaluation of the CP calibration techniques by isolating their effects from the influence of complex feature engineering.

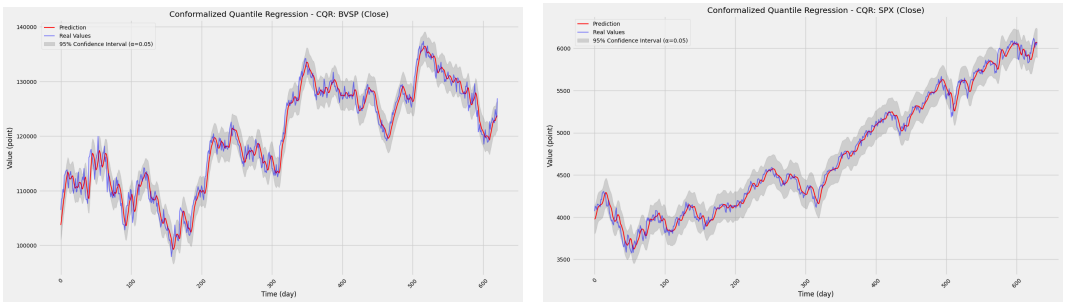
Although standard normalization was applied across all input subsets, we observed instability in the predictive performance of the *XGBoost* model when values in the validation, calibration, or test sets exceeded the numerical range encountered during training. The occurrence outside the training range is a characteristic of stock market behavior during specific time slots, observed in the exchanges analyzed in this study and reported in other markets worldwide. This behavior reflects a well-documented limitation of tree-based models: their poor ability to extrapolate when handling out-of-domain data. Unlike parametric models, decision tree-based algorithms partition the input space solely according to the distribution observed during training, lacking the capacity to generalize to unseen regions in the feature space. Consequently, any change in the input distribution, a common phenomenon in time series prediction, can lead to a degradation in predictive accuracy (LUNDBERG et al., 2020).

A preprocessing step was introduced before normalization involving a value-shifting technique to mitigate the extrapolation issue observed in the *XGBoost* model. Specifically, the validation, calibration, and test sets were transformed to align their respective maximum values with the maximum value observed in the training set. This domain alignment ensured that all

input subsets fell within a similar numerical range, reducing the likelihood of the model encountering out-of-distribution inputs. As a result, this procedure improved consistency across data partitions and enhanced the stability and robustness of *XGBoost* predictions under distributional shifts.

All figures presented in the following subsections show graphical results derived exclusively from the test subset.

The *LSTM* model demonstrated slower training and inference times than the other machine learning models (*SVM* and *XGBoost*), primarily due to its sequential architecture and higher computational complexity. However, it performed better than the *XGBoost* on the training data, capturing temporal dependencies more effectively. While beneficial for modeling complex patterns, this higher fitting capacity also increases the risk of overfitting, particularly in financial time series where noise and nonstationarity are prevalent. As a result, additional caution must be exercised when evaluating *LSTM* performance, ensuring that its predictive gains are not merely artifacts of memorizing training dynamics.

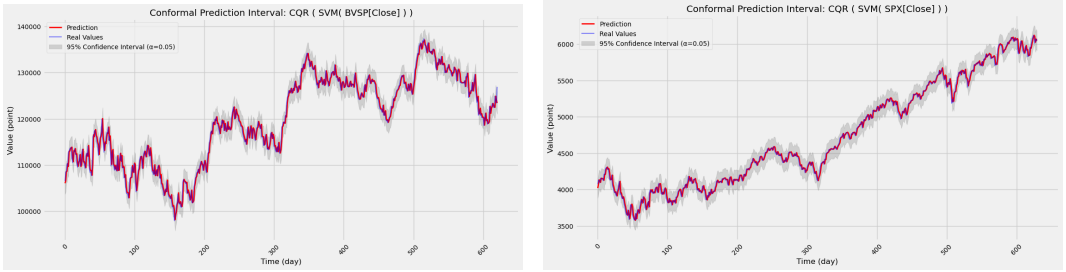


(a) BVSP Real vs Predicted. (b) SPX Real vs Predicted.

Figure 4.3: LSTM Prediction over the *test* subset.

As illustrated in Figures 4.3a and 4.3b, the *SPX* index exhibits a more monotonic, stable, and trend-oriented behavior compared to *BVSP*. This suggests that the Brazilian market (BVSP) is more volatile and often operates in a sideways (non-trending) regime.

The *Support Vector Machine (SVM)* model is straightforward to configure, requiring only a limited number of hyperparameters. Its simplicity and well-established theoretical foundation make it an attractive choice for baseline modeling tasks. In our experiments, SVM demonstrated high computational efficiency, particularly well suited for relatively short time series datasets. Given that the financial time series used in this study, based on daily intervals over the past 24 years, comprised slightly more than 6,200 observations, SVM could train and produce predictions with minimal computational overhead. This efficiency, combined with its compatibility with the `scikit-learn` framework, contributed to its selection as a replacement for the initially tested *Neural Prophet (NProphet)* model, which does not follow a `scikit-learn`-like structure.



(a) BVSP Real vs Predicted.

(b) SPX Real vs Predicted.

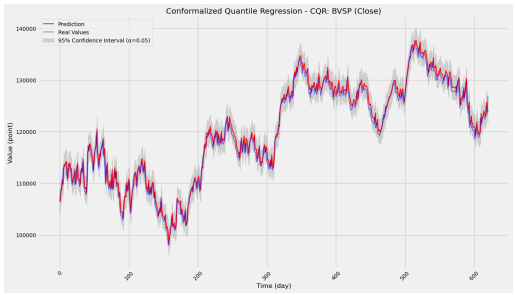
Figure 4.4: SVM Prediction over the *test* subset.

As illustrated in Figures 4.4a and 4.4b, the *SVM* model demonstrates superior predictive performance, with the predicted values (in red) closely tracking and visually overlapping the actual values (in blue). This high level of alignment causes the red prediction line to dominate the plot, making the blue line of the actual values difficult to distinguish. In contrast, the *LSTM* model, shown in Figures 4.3a and 4.3b, exhibits a visibly more significant divergence between predicted and true values, the blue line remaining visible. These observations suggest that the *SVM* achieves higher prediction accuracy and a better fit for our stock market dataset compared to the *LSTM* model.

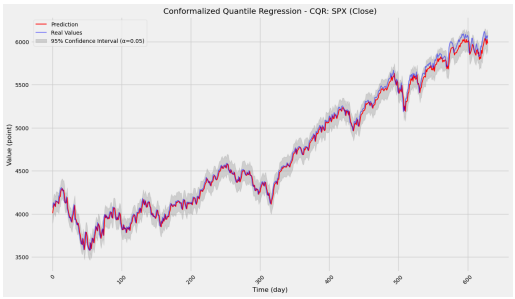
XGBoost is widely recognized for its computational efficiency and ability to deliver strong predictive performance with minimal data preprocessing. Its tree-based architecture, enhanced by gradients, enables it to robustly handle non-linear relationships and missing values, making it a popular choice for forecasting financial time series (CHEN; GUESTRIN, 2016a).

However, a notable limitation was observed during experimentation: *XGBoost* exhibited sensitivity to domain changes between training and evaluation datasets. Specifically, when the calibration and test datasets contained feature values outside the numerical range encountered during training and validation, such as when training values fell within a bounded interval but calibration and test values exceeded those bounds, *XGBoost* struggled to generalize effectively. This behavior stems from the fact that decision tree-based models, including *XGBoost*, partition the feature space based on thresholds derived from the training data and are inherently limited in their ability to extrapolate beyond observed regions (LUNDBERG et al., 2020).

As mentioned in Section 4.1, this issue is mitigated by applying the value-shifting technique.



(a) BVSP Real vs Predicted.



(b) SPX Real vs Predicted.

Figure 4.5: XGBoostPrediction over the *test* subset.

As illustrated in Figures 4.5a and 4.5b, the *XGBoost* model demonstrates satisfactory predictive performance throughout most of the time series, with a notable decline toward the end. This behavior is particularly evident in the *SPX* dataset, where the model's predictive accuracy deteriorates as the series approaches higher value levels. This degradation may be attributed to the limitations of extrapolation previously discussed at the beginning of Section 4.1. Interestingly, the loss of predictive power coincides with data points reaching values beyond those encountered during training. In addition to its predictive performance, *XGBoost* is also characterized by fast training execution, comparable to that of the *SVM* model.

4.2 CP Layer

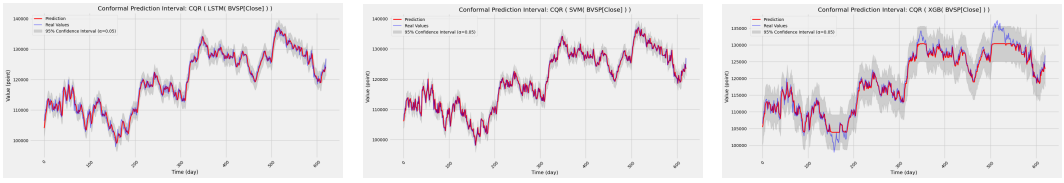
This section applies two Conformal Prediction (CP) techniques—Conformalized Quantile Regression (CQR) and Mondrian Conformal Prediction (MCP)—to calibrate machine learning models trained for asset price prediction. These methods produce statistically valid prediction intervals (PIs) for Close Price Values (CPVs), enabling uncertainty quantification and informed risk management.

Both techniques are evaluated on the IBOV and SPX indices to ensure methodological consistency. The analysis investigates each method's ability to enhance predictive robustness and empirical coverage, contributing to the practical reliability of CP in real-market scenarios.

Conformalized Quantile Regression (CQR) is used to construct distribution-free, feature-conditional prediction intervals by leveraging *quantile regression (QR)* within the *conformal prediction (CP)* framework. The procedure begins by training a model to estimate the conditional *lower* and *upper quantiles* of the response variable at a pre-specified miscoverage level. These quantile estimates are then calibrated using a *non-conformity score*

(NCS), typically defined as the difference between the predicted interval and the observed value, computed over a held-out *calibration* set.

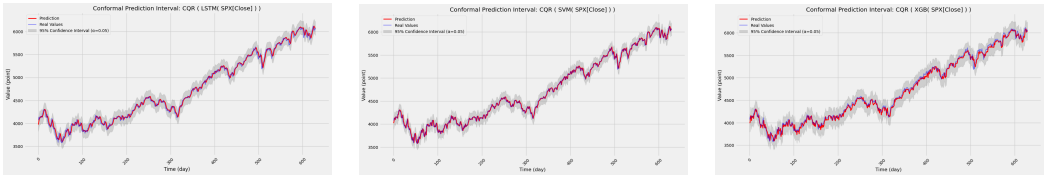
Under the assumption of data *exchangeability*, this calibration step guarantees marginal coverage of a finite sample without relying on distributional assumptions. *CQR* thus combines the flexibility of *QR*, which adapts to heteroscedastic and non-linear relationships, with the rigorous probabilistic guarantees of *CP*, making it well suited for uncertainty quantification in complex forecasting tasks (ROMANO; PATTERSON; CANDES, 2019).



(a) LSTM - BVSP. (b) SVM - BVSP. (c) XGBoost - BVSP.

Figure 4.6: CQR: Conformal Results over the BVSP dataset.

Figures 4.6b and 4.6c illustrate the *Conformal Prediction Interval (CPI)* bounds (in gray) constructed using *CQR* for two different machine learning models, SVM and XGBoost, applied to the same dataset (*BVSP*). The predicted values are colored red, while the actual ones are colored blue. This comparison highlights the *CP* calibration's sensitivity to the underlying model's predictive quality. In particular, in the final portion of the time series, the XGBoost model decreases the predictive precision, as evidenced by significantly wider *CPI* bands. This outcome reflects the dependence of conformal prediction on the reliability of the base model: poorer predictive performance leads to greater uncertainty and, consequently, broader intervals (ROMANO; PATTERSON; CANDES, 2019).

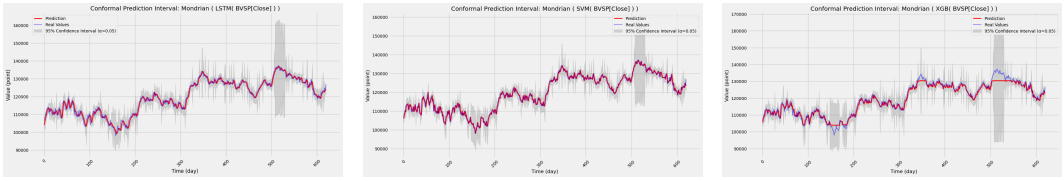


(a) LSTM - SPX. (b) SVM - SPX. (c) XGBoost - SPX.

Figure 4.7: CQR: Conformal Results over the SPX dataset.

Mondrian Conformal Prediction (Mondrian CP) extends the standard *CP* framework by incorporating localized calibration based on partitions of the input space. Instead of applying a single global *conformity score (CS)* throughout the entire dataset, *Mondrian CP* conditions the *nonconformity*

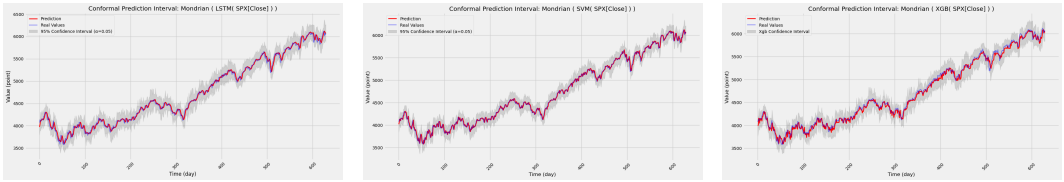
scores (NCS) within distinct subsets, often defined by discrete variables or *clustering* strategies, allowing for heterogeneity in the data. This stratified approach improves the efficiency and informativeness of the resulting intervals, particularly in domains like finance, where market dynamics can vary significantly between different regimes or asset types. By capturing such contextual variability, *Mondrian CP* offers more precise and relevant prediction intervals tailored to the specific behavior of each data segment (VOVK; GAMMERMAN; SHAFER, 2005b).



(a) LSTM - BSVP. (b) SVM - BVSP. (c) XGBoost - BVSP.

Figure 4.8: Mondrial CP: Conformal Results over the BVSP dataset.

Figures 4.8a, 4.8b, and 4.8c illustrate that the *Mondrian Conformal Prediction (MCP)* method struggled to maintain a reliable interval calibration in the same critical region where *XGBoost* previously failed. This convergence of errors across multiple models suggests the presence of a structurally challenging or chaotic regime within that time slot, which warrants deeper investigation in future studies.



(a) LSTM - SPX. (b) SVM - SPX. (c) XGBoost - SPX.

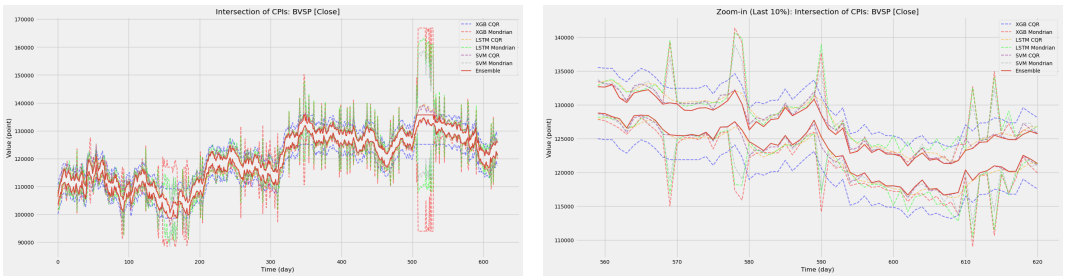
Figure 4.9: Mondrial CP: Conformal Results over the SPX dataset.

Figures 4.9a, 4.9b, and 4.9c demonstrate representative results of the experiments using *Mondrian CP*.

4.3 Ensemble Method

This section presents two ensemble strategies, *Ensemble-M* and *Ensemble-R*, developed to enhance the calibration of predictive uncertainty in financial time series forecasting by employing *Conformal Prediction (CP)*

techniques. Each ensemble strategy operates on a set of six predictive intervals, generated by applying two CP methodologies—*Conformalized Quantile Regression (CQR)* and *Mondrian Conformal Prediction (Mondrian-CP)*—to the outputs of three distinct machine learning models: *Long Short-Term Memory (LSTM)*, *Support Vector Machines (SVM)*, and *eXtreme Gradient Boosting (XGBoost)*. Both ensemble strategies utilize a standard mechanism for interval aggregation, which involves computing the pointwise numerical intersection of the six resulting *Conformal Prediction Intervals (CPIs)*. This intersection seeks to retain only those regions where the predictions from all models and calibration methods agree, resulting in a more robust and conservative predictive interval. The CPIs, by design, are valid and distribution-free, providing rigorous uncertainty quantification under minimal assumptions. The *Ensemble-M* strategy exemplifies this approach by aggregating the CPIs for each time step and producing a consensus region of confidence. This intersection-based mechanism inherently filters out noisy or extreme intervals influenced by outlier model behavior, producing narrower and more reliable bands. The underlying rationale is to leverage the strengths of model diversity while mitigating individual weaknesses, making a predictive region that is both empirically supported and statistically cautious. Intersection-based ensemble strategies offer substantial advantages in volatile financial environments, where achieving precision and reliability is critical for risk-sensitive decision-making. By aggregating conformal prediction intervals (CPIs) across diverse models and calibration schemes, such strategies enhance the robustness of predictive inference while mitigating the risk of overconfident or biased estimates. The theoretical and empirical benefits of ensemble conformal predictors are well-supported in recent literature. For example, Vovk et al. (2022) demonstrates that ensemble approaches can improve conditional coverage and sharpen predictive intervals, especially under model uncertainty and distributional shifts, conditions commonly encountered in financial time series forecasting.

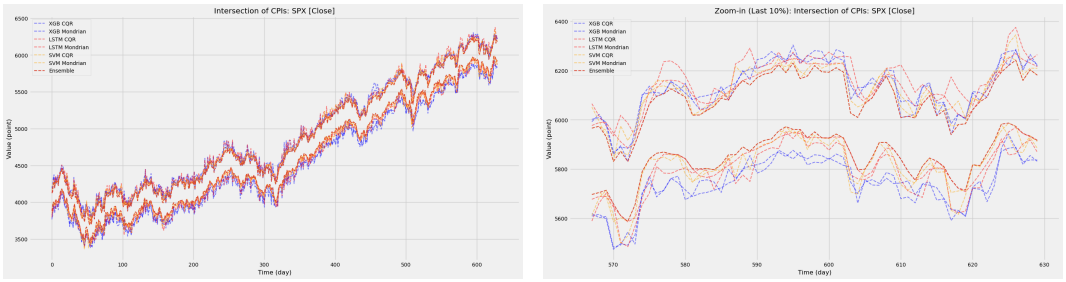


(a) BVSP without zoom.

(b) BVSP with zoom on the last 10%.

Figure 4.10: Intersection of six BVSP's *CPI*.

Figure 4.10a presents the results of the initial ensemble strategy, denoted as *Ensemble-M*, applied to each target financial asset. This strategy combines multiple *Conformal Prediction Intervals (CPIs)* derived from calibrated outputs of different *machine learning (ML)* models. Figure 4.10b provides a magnified view of the final 10% of the test subset, allowing for a more detailed examination of prediction interval behavior during this evaluation window.



(a) SPX without zoom.

(b) SPX with zoom on the last 10%.

Figure 4.11: Intersection of six SPX's *CPI*.

Figure 4.11 exhibits analogous behavior to that observed in Figure 4.10. To avoid redundancy, their detailed explanation is omitted here. The only notable distinction lies in the relative smoothness of the *SPX* Conformal Prediction Intervals (*CPIs*), which exhibit lower volatility compared to those obtained for the *BVSP* asset.

The primary distinction between *Ensemble-M* and *Ensemble-R* is the strategy used to extract the final point estimate from the unified interval. While *Ensemble-M* selects the median value within the intersected interval, *Ensemble-R* randomly samples a value from the same region. The subsequent sections further discuss the rationale and implications of these selection criteria.

4.3.1 Ensemble-M

The distinguishing feature of *Ensemble-M* is its selection strategy for the final point prediction. Specifically, the ensemble computes a representative point based on the *median* of the interval bounds, hence the character "*M*" from the name "*Ensemble-M*". To ensure robustness against outliers and extreme values, the median is calculated through a filtering process based on the *interquartile range (IQR)*. This involves aggregating all *upper* and *lower bounds* from the six *CPIs* and retaining only those values that fall within the *IQR*, defined between the first (*Q1*) and third (*Q3*) *quartiles*. The final prediction is then obtained by computing the *median* of these filtered values. This strategy

aims to balance the influence of all models while reducing sensitivity to anomalous interval bounds, thereby producing a stable and central prediction within the collective uncertainty space. Figures 4.10 and 4.11 illustrate the practical application of this method. At each time step, a point estimate is derived from the median of the CPI bounds, producing a consistent and resilient central prediction across the test dataset.

4.3.2 Ensemble-R

The distinguishing feature of the *Ensemble-R* strategy lies in the method used to select the final point prediction within the aggregated *Conformal Prediction Interval (CPI)*. Instead of relying on a central tendency measure such as the *median*, as in *Ensemble-M*, the *Ensemble-R* method selects a single point *randomly* from a uniform distribution defined over the bounds of the intersected *CPI*. This approach reflects a probabilistic selection mechanism, where each value within the interval has equal likelihood of being chosen, hence the *"R"* (random) designation in its name. This random selection technique aligns with principles from game theory, particularly under the assumption of a fully informed adversarial environment. In their seminal work, von Neumann et. al. (1944) demonstrated that, in such adversarial settings, randomized strategies can serve as optimal countermeasures to prevent deterministic predictability. By adopting a stochastic final prediction, *Ensemble-R* enhances robustness and avoids overfitting to deterministic strategies that may be exploitable in volatile or manipulated financial environments. Figures 4.10 and 4.11 also illustrate the practical implementation of this strategy. At each prediction time step, a point is sampled from a uniform distribution within the corresponding CPI. This ensures a realistic representation of intra-interval variability and aligns the output with the probabilistic interpretation of conformal prediction. The randomized final prediction provides a more nuanced tool for financial decision-making, especially under uncertain and dynamic trading conditions.

4.4 Heuristic

To more accurately capture realistic *intraday* trading dynamics, we adopt a heuristic that evaluates prediction success based on *intra-session* price behavior, rather than relying exclusively on proximity to the *end-of-day* closing price. Specifically, a prediction is considered successful if the actual market price crosses the predicted point at any time during the trading session, irre-

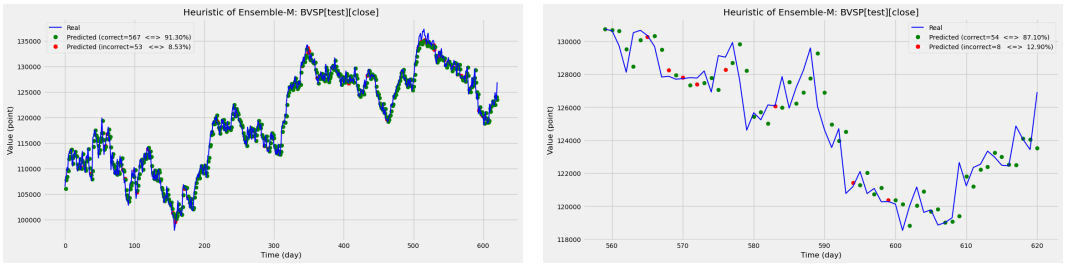
spective of the direction of the crossing. This approach, termed the *crossing-based heuristic (CBH)*, is rooted in practical trading logic, particularly the operation of *limit orders*, *stop-loss triggers*, and other threshold-based execution mechanisms commonly utilized by both institutional and retail traders. Such mechanisms activate when a specified price level is touched, not necessarily maintained, thereby making the moment of crossing more relevant than the closing position. The *CBH* is especially pertinent for high-liquidity, high-value financial instruments, where execution is typically feasible throughout the session. Although this heuristic is a practical convention in real-world trading, it has received limited formal treatment in the academic literature. Nonetheless, its foundations align with empirical findings on intraday price formation and execution timing. For example, Heston, Korajczyk e Sadka (2010) examines systematic patterns in intraday return behavior across a cross-section of stocks and supports using within-session price dynamics as a critical determinant of market activity. By adopting the CBH, we incorporate a more execution-aligned assessment of prediction quality, which is essential for evaluating model performance in real-time trading applications. Both ensemble models presented in this work, *Ensemble-M* and *Ensemble-R*, implement the *CBH* to assess whether a predicted value was practically reachable during the trading day. For interpretability and visual analysis, a color-coded representation is employed:

- *Green dots* denote successful predictions where the asset's price range crossed the forecasted price during the session.
- *Red dots* denote unsuccessful predictions, i.e., the forecasted value was not touched within the session.

This evaluation metric thus captures statistical accuracy and execution feasibility, a critical consideration for applications in real-time financial forecasting and decision support.

4.4.1 Ensemble-M

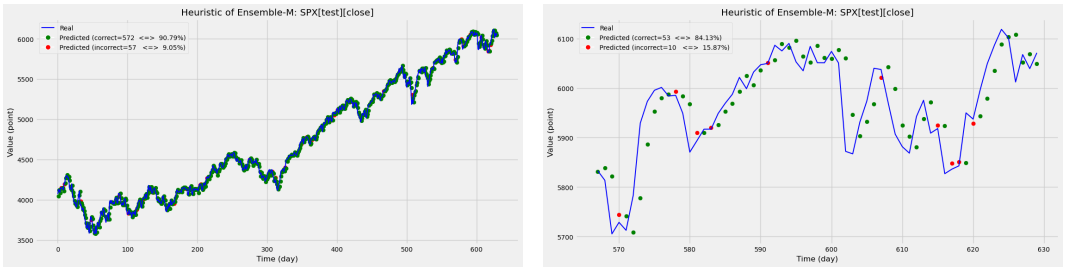
Figure 4.12a presents the prediction results of the *Ensemble-M* model applied to the test subset of the *BVSP* asset, with final point predictions evaluated according to the *crossing-based heuristic (CBH)*. Figure 4.12b provides a magnified view of the final 10% of this dataset, allowing for a closer inspection of prediction performance near the time horizon.



(a) BVSP without zoom. (b) BVSP with zoom on the Last 10%.

Figure 4.12: Heuristic - Final Result of BVSP using Ensemble-M.

Figures 4.13a and 4.13b exhibit analogous behavior to that observed in Figure 4.12. To avoid redundancy, their detailed explanation is omitted here. The only noteworthy distinction is the comparative effectiveness observed across datasets: the *Ensemble-M* method achieved a higher incidence of successful predictions under the *crossing-based heuristic (CBH)* when applied to the *BVSP* dataset than to *SPX*. This discrepancy may be attributed to differences in intraday volatility and market microstructure between the two assets, as suggested by empirical studies on high-frequency trading and return dynamics (HESTON; KORAJCZYK; SADKA, 2010).

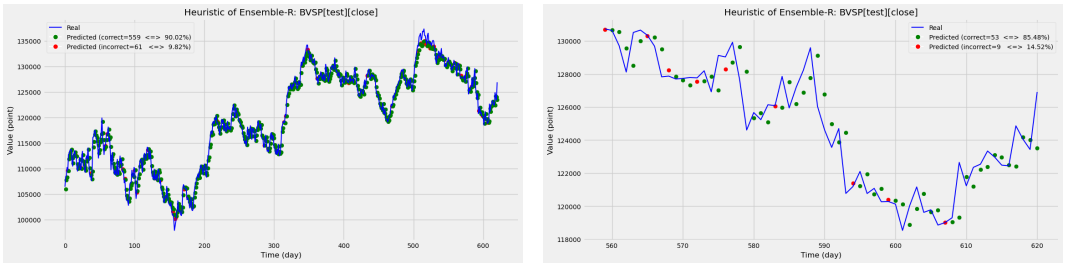


(a) SPX without zoom. (b) SPX with zoom on the Last 10%.

Figure 4.13: Heuristic - Final Result of SPX using Ensemble-M.

4.4.2 Ensemble-R

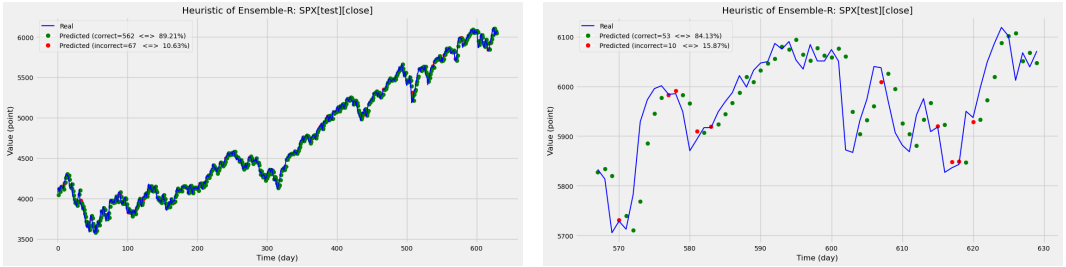
Figure 4.14a presents the prediction results of the *Ensemble-R* model applied to the test set of the *BVSP* asset, with final point predictions evaluated according to the *crossing-based heuristic (CBH)*. Figure 4.14b provides a magnified view of the final 10% of this dataset, allowing for a closer inspection of prediction performance near the time horizon.



(a) BVSP without zoom. (b) BVSP with zoom on the Last 10%.

Figure 4.14: Heuristic - Final Result of BVSP using Ensemble-R.

Figures 4.15a and 4.15b exhibit analogous behavior to that observed in Figure 4.14. To avoid redundancy, their detailed explanation is omitted here. Consistent with the behavior observed in the *Ensemble-M* configuration, the *Ensemble-R* strategy also demonstrated superior predictive performance on the *BVSP* (Bovespa) asset compared to the *SPX* (S&P 500). This discrepancy can also be attributed to market-specific characteristics, such as volatility profiles, liquidity distribution, and regional trading patterns, which influence the responsiveness of conformal prediction intervals and ensemble decision-making under uncertainty.



(a) SPX without zoom. (b) SPX with zoom on the Last 10%.

Figure 4.15: Heuristic - Final Result of SPX using Ensemble-R.

4.5 Analysis

This sub-section presents the evaluation methodology adopted to assess the performance of the predictive and calibration components of the proposed framework. We use error metrics to quantify *machine learning (ML)* predictions and the effectiveness of *conformal prediction (CP)* methods' effectiveness in generating reliable prediction intervals. The analysis includes point prediction errors and interval-based evaluations, providing a comprehensive view of how each component contributes to overall predictive performance and uncertainty quantification.

4.5.1 Prediction Metric

In stock market forecasting, percentage-based error metrics are generally considered more appropriate than absolute ones, as they normalize predictive performance across various price scales and market conditions. This study adopts the *Symmetric Mean Absolute Percentage Error (sMAPE)* as the principal evaluation criterion due to its robustness in treating both under and over-forecasting symmetrically (MAKRIDAKIS, 1993). sMAPE is particularly advantageous in financial contexts, as it accounts for bidirectional error dispersion, providing a balanced view of model accuracy over volatile time series. Nonetheless, the use of sMAPE is not without criticism. Goodwin e Lawton (1999) observed that sMAPE can introduce a subtle asymmetry by penalizing under-predictions more heavily than over-predictions. In contrast, Hyndman e Koehler (2006) argued that sMAPE tends to penalize over-forecasting, particularly in low actual values disproportionately. These divergent viewpoints underscore the need for caution when interpreting sMAPE in edge cases or near-zero series. To provide a comprehensive performance assessment, we also report additional error metrics, including *Mean Absolute Percentage Error (MAPE)*, *Mean Absolute Error (MAE)*, and *Root Mean Square Error (RMSE)*. While MAPE is a widely cited metric in forecasting literature, it is unstable when actual values approach zero, leading to inflated or undefined errors (MYTTENAERE et al., 2016). Consequently, MAPE is not employed as the primary evaluation metric in this study but is retained for completeness and comparability with prior research.

Across all experiments, as shown in Table 4.1, the individual machine learning models (LSTM, SVM, and XGBoost) generally outperformed the proposed ensemble models regarding point prediction accuracy. This outcome is expected, as the ensemble models are designed not for optimal point estimation but for generating robust and reliable prediction intervals in the face of uncertainty.

The performance of the models also varied depending on the dataset. In particular, models trained and tested on the SPX dataset consistently achieved lower error rates than those evaluated on the BVSP dataset. This empirical observation suggests that the SPX market may exhibit more stable or learnable patterns, whereas the BVSP data appears more volatile or structurally complex, making it relatively less predictable.

Table 4.1: Prediction Error Metrics

Model	Asset	SMAPE ↓	MAPE	MAE	RMSE
BVSP	SVM	0.8227	0.8229	969.3567	1250.9759
	Ensemble-M	0.8696	0.8689	1023.7192	1298.4379
	Ensemble-R	0.8804	0.8795	1037.8368	1314.8526
	LSTM	1.0175	1.0164	1192.6343	1494.4216
	XGBoost	1.1146	1.1104	1331.0614	1791.1499
SPX	SVM	0.7200	0.7204	32.9221	43.7233
	Ensemble-M	0.7739	0.7740	35.3620	46.2093
	Ensemble-R	0.7908	0.7906	36.1571	46.7404
	LSTM	0.8787	0.8797	40.2592	52.5583
	XGBoost	1.0247	1.0206	47.1508	59.5157

4.5.2 CPI Metric

This subsection analyzes key metrics related to Conformal Prediction Intervals (CPIs), which provide calibrated uncertainty estimates for high- and low-value price forecasts. The evaluation focuses on three core indicators: CPI Coverage, CPI Mean, and CPI Median.

CPI Coverage represents the proportion of instances where the actual observed value falls within the predicted interval. It is a measure of reliability, with higher values indicating that the interval successfully captures the proper market behavior.

In contrast, CPI Mean and CPI Median quantify the average and median length of the prediction intervals, respectively. These metrics reflect the informativeness and precision of the intervals, with shorter intervals being preferable for practical decision-making.

This study's ensemble methods aim to reduce the width of the *Conformal Prediction Interval (CPI)* by aggregating outputs from multiple conformal predictors. Although intersection-based strategies may slightly relax the formal statistical guarantees of individual conformal methods, empirical evidence indicates that coverage remains robust.

As shown in Table 4.2, both ensemble approaches—*Conformalized Quantile Regression (CQR)* and *Mondrian Conformal Prediction*—demonstrate strong calibration performance, maintaining coverage rates of at least 98% across both the SPX and BVSP datasets. These results highlight the effectiveness of the conformal framework in balancing reliability and interval compactness, reinforcing its robustness for uncertainty quantification in financial forecasting.

Table 4.2: Resume CPI Metrics

Dataset	**ML Model**	**CP Model**	**CPI Coverage (%) ↓**	**CPI Mean**	**CPI Median**
BVSP	LSTM, SVM, XGBoost	Ensemble-R	98	6956.78	6411.92
	LSTM, SVM, XGBoost	Ensemble-M	98	6956.78	6411.92
	XGB	CQR	98	10562.20	10562.20
	XGB	Mondrian	97	10753.32	5822.94
	SVM	Mondrian	95	8217.10	5254.84
	LSTM	Mondrian	95	9268.50	5842.10
	SVM	CQR	94	4724.71	4724.71
	LSTM	CQR	93	5407.06	5407.05
SPX	LSTM, SVM, XGBoost	Ensemble-R	100	360.10	367.75
	LSTM, SVM, XGBoost	Ensemble-M	100	360.10	367.75
	LSTM	Mondrian	100	365.70	346.86
	XGB	CQR	100	393.35	393.35
	XGB	Mondrian	100	378.12	377.34
	LSTM	CQR	99	300.02	300.02
	SVM	Mondrian	99	312.13	301.61
	SVM	CQR	99	269.51	269.51

Regarding sharpness, SVM-based models generate the narrowest mean and median CPI widths, reflecting high precision in uncertainty estimation. Ensemble strategies, particularly on the SPX dataset, also deliver compact intervals, closely matching the performance of XGBoost-based models.

Overall, the findings support ensemble-based conformal approaches as a practical means to enhance interval sharpness while preserving reliable coverage. They are well-suited for risk-aware decision-making in financial time series forecasting.

5
Conclusion

As noted by Zecchin et al. (2024) and Teng et al. (2023), the effectiveness of the calibration is inherently dependent on the quality of the underlying predictions. This dependency was observed in the *XGBoost* model, where poor predictive performance in specific time intervals negatively impacted the overall quality of the calibration.

The experimental results demonstrate that applying Conformal Prediction (CP) significantly contributes to risk management in asset price forecasting by quantifying prediction uncertainty through calibrated confidence intervals. These intervals provide an interpretable measure of money exposure, allowing traders to align their decisions with predefined risk thresholds. The analysis further indicates that most operations, exceeding 89%, are successfully accepted by the market under the heuristic evaluation, confirming the practical viability of the approach in high-liquidity scenarios.

Moreover, the proposed Conformal Prediction Ensemble (CPE), which is grounded in the CP technique, consistently improves average performance metrics compared to models without CP calibration. This validates the potential of the CPE framework to improve predictive reliability and operational applicability in real-world trading environments.

5.1
Future Works

An important avenue for future research is to investigate the *theoretical probabilistic guarantees* arising from the integration of *Conformal Prediction (CP)* methods with *Heuristic Rules (HR)*. This includes evaluating conformal intervals' validity, calibration, and conditional coverage when heuristic-driven decision layers are embedded into the predictive pipeline. Additionally, future work could benefit from employing more robust clustering strategies within the *Mondrian CP* approach to improve the granularity and effectiveness of data partitioning, directly impacting the quality of localized prediction intervals.

Disclaimer

It is academic research and should not be used in real-life trades without the necessary modifications to be more adherent to the investor context. All risks are borne by those who use the approaches or artifacts presented here.

Disclosure

The authors affirm that there is not any conflict of interest over this academic research.

Contribution

The relevance of this research is to experiment and quantify how much the *Conformal Prediction (CP)* improves the average prediction results of three classical and *benchmarks* predictor models, using 2 worldwide *financial indexes* (SPX (SP500) and IBOV) as *benchmarks* datasets.

Funding

This research was possible thanks to the financial support granted by "Coordenação de Aperfeiçoamento de Pesquisa de Nível Superior - Brazil (CAPES)" of the project number 88887.713278/2022-00.

Aknowledgement

The author thanks Professor José Alberto Sardinha for his transference of knowledge and kind support.

6
Bibliography

AÏT-SAHALIA, Y. et al. **How and when are high-frequency stock returns predictable?** [S.l.], 2022. Disponível em: <https://www.nber.org/papers/w30366>. Cited in page 17.

AN, B.; SUN, S.; WANG, R. Deep reinforcement learning for quantitative trading: Challenges and opportunities. **IEEE Intelligent Systems**, IEEE, v. 37, n. 2, p. 23–26, 2022. Disponível em: <https://doi.org/10.1109/MIS.2022.3165994>. Cited 2 times in pages 16 and 20.

ANGELOPOULOS, A. N.; BATES, S.; FANNJIANG, C. e. a. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. **arXiv preprint arXiv:2107.07511**, 2021. Cited 2 times in pages 28 and 29.

ANGELOPOULOS, A. N. et al. Uncertainty sets for image classifiers using conformal prediction. **arXiv preprint arXiv:2009.14193**, 2020. Disponível em: <https://doi.org/10.48550/arXiv.2009.14193>. Cited 2 times in pages 99 and 100.

ARSLAN, O. Forecasting the financial market using hybrid models: A review and research agenda. **PeerJ Computer Science**, PeerJ Inc., v. 8, p. e1001, 2022. Cited in page 19.

BALLINGS, M. et al. Evaluating multiple classifiers for stock price direction prediction. **Expert Systems with Applications**, Elsevier, v. 42, n. 20, p. 7046–7056, 2015. Cited in page 27.

BARBER, R. F. et al. Predictive inference with the jackknife+. **The Annals of Statistics**, JSTOR, v. 49, n. 1, p. 486–507, 2021. Cited in page 99.

BOLLEN, J.; MAO, H.; ZENG, X. Twitter mood predicts the stock market. **Journal of Computational Science**, Elsevier, v. 2, n. 1, p. 1–8, 2011. Cited in page 25.

BOLLERSLEV, T. Generalized autoregressive conditional heteroskedasticity. **Journal of Econometrics**, Elsevier, v. 31, n. 3, p. 307–327, 1986. Cited in page 28.

BOLLINGER, J. **Bollinger on Bollinger Bands**. [S.l.]: McGraw-Hill, 2002. Cited in page 38.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: **Proceedings of the fifth annual workshop on Computational learning theory**. [S.l.: s.n.], 1992. p. 144–152. Cited in page 91.

BOX, G. E.; PIERCE, D. A. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. **Journal of the American Statistical Association**, Taylor & Francis, v. 65, n. 332, p. 1509–1526, 1970. Cited in page 16.

BOX, G. E. P. et al. **Time Series Analysis: Forecasting and Control**. 5. ed. [S.l.]: John Wiley & Sons, 2015. Cited in page 28.

CAO, J.; LI, Z.; LI, Q. Financial time series forecasting model based on ceemdan and lstm-attention. **Expert Systems with Applications**, Elsevier, v. 169, p. 114481, 2021. Cited in page 26.

CAO, L.; TAY, F. E. Support vector machines for forecasting financial market indices. **Neurocomputing**, Elsevier, v. 70, p. 192–198, 2005. Cited in page 27.

CARUANA, R.; NICULESCU-MIZIL, A. An empirical comparison of supervised learning algorithms. In: ACM. **Proceedings of the 23rd international conference on Machine learning**. [S.l.], 2006. p. 161–168. Cited in page 23.

CHAKRABORTY, A.; GHOSH, T. Forecasting stock market using fusion of sentiment analysis and machine learning techniques. **Journal of Intelligent & Fuzzy Systems**, IOS Press, v. 41, n. 3, p. 3667–3675, 2021. Cited in page 28.

CHEN, K.; ZHOU, Y.; DAI, F. Lstm networks for financial time series prediction. In: IEEE. **10th International Conference on Intelligent Systems (IS)**. [S.l.], 2015. p. 507–512. Cited in page 26.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: ACM. **Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining**. [S.l.], 2016. p. 785–794. Cited 2 times in pages 27 and 48.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: **Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining**. [s.n.], 2016. p. 785–794. Disponível em: <https://arxiv.org/pdf/1603.02754.pdf>. Cited 2 times in pages 34 and 40.

CHERNOZHUKOV, V.; WüTHRICH, K. Distributional conformal prediction: Inference under model misspecification. **Journal of the Royal Statistical Society Series B**, Wiley, 2021. Cited in page 30.

CHO, K. et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. **arXiv preprint arXiv:1406.1078**, 2014. Cited 2 times in pages 89 and 90.

COOTNER, P. H. **The random character of stock market prices**. [S.l.]: MIT Press, 1964. Cited in page 25.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995. Cited 2 times in pages 33 and 40.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995. Cited in page 91.

DEWOLF, N.; BAETS, B. D.; WAEGEMAN, W. Valid prediction intervals for regression problems. **Artificial Intelligence Review**, v. 56, n. 1, p. 577–613, 2023. Disponível em: <https://link.springer.com/article/10.1007/s10462-022-10178-5>. Cited in page 101.

DONG, Y.; YU, T.; LIU, J. Multi-scale ensemble learning model for stock price prediction. **Expert Systems with Applications**, Elsevier, v. 187, p. 115906, 2022. Cited in page 28.

DRUCKER, H. et al. **Support vector regression machines**. [S.l.], 1997. Cited in page 91.

ENGLE, R. F. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. **Econometrica: Journal of the Econometric Society**, Wiley, v. 50, n. 4, p. 987–1007, 1982. Cited in page 16.

FAMA, E. F. The behavior of stock-market prices. **The Journal of Business**, University of Chicago Press, v. 38, n. 1, p. 34–105, 1965. Cited in page 25.

FAMA, E. F. Efficient capital markets: A review of theory and empirical work. **The Journal of Finance**, Wiley, v. 25, n. 2, p. 383–417, 1970. Cited in page 19.

FISCHER, T.; KRAUSS, C. Deep learning with long short-term memory networks for financial market predictions. **European Journal of Operational Research**, Elsevier, v. 270, n. 2, p. 654–669, 2018. Cited in page 26.

GAMMERMAN, A.; VOVK, V.; VAPNIK, V. Learning from examples using rough sets and statistical learning theory. In: **Proceedings of the 13th international conference on machine learning (ICML)**. [S.l.: s.n.], 1998. p. 106–114. Cited 3 times in pages 22, 35, and 97.

GANDHMAL, D. P.; PATEL, K. R. Systematic analysis and review of stock market prediction techniques. **Computer Science Review**, Elsevier, v. 34, p. 100190, 2019. Cited 2 times in pages 19 and 45.

GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. Learning to forget: Continual prediction with lstm. In: IET. **Proceedings of the 9th International Conference on Artificial Neural Networks**. [S.l.], 2000. p. 850–855. Cited in page 87.

GERS, F. A.; SCHRAUDOLPH, N. N.; SCHMIDHUBER, J. Learning precise timing with lstm recurrent networks. In: **Journal of Machine Learning Research**. [S.l.: s.n.], 2002. v. 3, p. 115–143. Cited in page 87.

GHANBARI, M.; GOLDANI, M. Support vector regression parameters optimization using golden sine algorithm and its application in stock market. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 100, p. 104207, 2021. Cited in page 27.

GIBBS, C.; CANDèS, E. J.; LEI, J. Adaptive conformal inference via data splitting. **Journal of the Royal Statistical Society Series B: Statistical Methodology**, Wiley, v. 83, n. 4, p. 727–747, 2021. Cited in page 29.

GOODWIN, P.; LAWTON, R. Rule-based forecasting: An empirical test of the theory. **Journal of Forecasting**, Wiley, v. 18, n. 1, p. 1–13, 1999. Disponível em: <https://doi.org/10.1002/(SICI)1099-131X(199901)18:1<1::AID-FOR719>3.0.CO;2-2>. Cited in page 58.

GRAVES, A. et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: ACM. **Proceedings of the 23rd International Conference on Machine Learning**. [S.l.], 2006. p. 369–376. Cited in page 88.

GRAVES, A. et al. Offline handwriting recognition with multidimensional recurrent neural networks. In: IEEE. **2009 10th International Conference on Document Analysis and Recognition**. [S.l.], 2009. p. 200–205. Cited in page 89.

GRAVES, A.; SCHMIDHUBER, J. Framewise phoneme classification with bidirectional lstm and other neural network architectures. **Neural networks**, Elsevier, v. 18, n. 5-6, p. 602–610, 2005. Cited in page 88.

GRAVES, A.; SCHMIDHUBER, J. Keyword spotting using time-delay neural networks. In: **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2007. v. 19, p. 1049–1056. Cited in page 88.

GREFF, K. et al. Lstm: A search space odyssey. **IEEE transactions on neural networks and learning systems**, IEEE, v. 28, n. 10, p. 2222–2232, 2016. Cited 2 times in pages 89 and 90.

GUO, C. et al. On calibration of modern neural networks. In: PMLR. **Proceedings of the 34th International Conference on Machine Learning**. [S.l.], 2017. p. 1321–1330. Cited 2 times in pages 23 and 96.

HASSAN, M. R.; NATH, B. A fusion approach of hmm, ann and ga for stock market forecasting. **Expert Systems with Applications**, Elsevier, v. 33, n. 1, p. 171–180, 2007. Cited in page 18.

HECHTLINGER, Y.; PóCZOS, B.; WASSERMAN, L. Cautious deep learning. **arXiv preprint arXiv:1805.09460**, 2018. Disponível em: <https://doi.org/10.48550/arXiv.1805.09460>. Cited in page 100.

HESTON, S. L.; KORAJCZYK, R. A.; SADKA, R. Intraday patterns in the cross-section of stock returns. **The Journal of Finance**, Wiley Online Library, v. 65, n. 4, p. 1369–1407, 2010. Cited 2 times in pages 55 and 56.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Cited 4 times in pages 26, 34, 40, and 87.

HOFF, P. D.; KULESHOV, V.; ERMON, S. Achieving calibrated regression with sharpness guarantees. In: **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2022. Cited in page 97.

HUANG, W.; NAKAMORI, Y.; WANG, S.-Y. Forecasting stock market movement direction with support vector machine. **Computers & Operations Research**, Elsevier, v. 32, n. 10, p. 2513–2522, 2005. Cited in page 27.

HYNDMAN, R. J.; KOEHLER, A. B. Another look at measures of forecast accuracy. **International Journal of Forecasting**, Elsevier, v. 22, n. 4, p. 679–688, 2006. Disponível em: <https://doi.org/10.1016/j.ijforecast.2006.03.001>. Cited in page 58.

JABEUR, S. B.; LAHMIRI, S.; HUSSAIN, A. A. Forecasting cryptocurrency price direction using machine learning techniques: The case of bitcoin and ethereum. **Entropy**, MDPI, v. 22, n. 5, p. 552, 2020. Cited in page 28.

JANUSCHOWSKI, T. et al. Forecasting with trees. **International Journal of Forecasting**, Elsevier, v. 38, n. 4, p. 1473–1481, 2022. Disponível em: <www.amazon.science/publications/forecasting-with-trees>. Cited 2 times in pages 28 and 34.

JOHANSSON, F.; GABRIELSSON, R. Calibration of probabilistic predictions. **arXiv preprint arXiv:1911.08534**, 2019. Cited in page 23.

JR, R. R. P.; PARKER, W. D. The financial/economic dichotomy in social behavioral dynamics: The socionomic perspective. **Journal of Behavioral Finance**, Routledge, v. 8, n. 2, p. 84–108, 2007. Cited in page 25.

KABIR, H. M. D. et al. Neural network-based uncertainty quantification: A survey of methodologies and applications. **IEEE Access**, v. 6, p. 36218–36234, jun. 2018. Disponível em: <https://doi.org/10.1109/ACCESS.2018.2836917>. Cited in page 22.

KARA, Y.; BOYACIOGLU, M. A.; BAYKAN, Ö. K. Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange. **Expert Systems with Applications**, Elsevier, v. 38, n. 5, p. 5311–5319, 2011. Cited in page 26.

KATH, L.; ZIEL, F. Conformal prediction for day-ahead electricity price forecasting. **Energy Economics**, Elsevier, v. 103, p. 105551, 2021. Cited in page 30.

KIM, K.-j. Financial time series forecasting using support vector machines. **Neurocomputing**, Elsevier, v. 55, p. 307–319, 2003. Cited in page 27.

KULESHOV, V.; FENNER, N.; ERMON, S. Accurate uncertainties for deep learning using calibrated regression. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2018. p. 2796–2804. Cited in page 96.

KUMAR, A. **Combining LSTM and Hidden Markov Models for Stock Price Prediction**. Dissertação (Master's thesis) — University of Victoria, 2023. Disponível em: <https://dspace.library.uvic.ca/handle/1828/14740>. Cited in page 90.

LAI, G. et al. Modeling long-and short-term temporal patterns with deep neural networks. In: **Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval**. [S.l.: s.n.], 2018. p. 95–104. Cited 2 times in pages 89 and 90.

LAWRENCE, R. D. Economic forecasting and neural networks: an application to tourism demand. **Journal of Applied Econometrics**, Wiley, v. 12, n. 5, p. 407–424, 1997. Cited in page 18.

LAWRENCE, R. J. Use of neural networks in forecasting stock market prices. **Proceedings of the South African Institute of Computer Scientists and Information Technologists**, p. 52–57, 1997. Cited in page 87.

LEI, J. et al. Distribution-free predictive inference for regression. **Journal of the American Statistical Association**, v. 113, n. 523, p. 1094–1111, 2018. Disponível em: <https://doi.org/10.1080/01621459.2017.1307116>. Cited 3 times in pages 85, 98, and 100.

LEI, J.; RINALDO, A.; WASSERMAN, L. A conformal prediction approach to explore functional data. **Annals of Mathematics and Artificial Intelligence**, v. 74, p. 29–43, 2015. Disponível em: <https://doi.org/10.1007/s10472-013-9366-6>. Cited in page 98.

LEVI, D. et al. Evaluating and calibrating uncertainty prediction in regression tasks. **arXiv preprint arXiv:2006.11220**, 2020. Cited in page 96.

LIM, B.; ZOHREN, S. Temporal fusion transformers for interpretable multi-horizon time series forecasting. **International Journal of Forecasting**, 2021. Originally NeurIPS 2019. Cited in page 29.

LIU, Y. et al. Patchtst: A patch-based transformer for multivariate time series forecasting. In: **International Conference on Learning Representations (ICLR)**. [S.l.: s.n.], 2023. Cited in page 29.

LUNDBERG, S. M. et al. From local explanations to global understanding with explainable ai for trees. **Nature Machine Intelligence**, Nature Publishing Group, v. 2, n. 1, p. 56–67, 2020. Cited 3 times in pages 42, 45, and 48.

LUO, Z. et al. Is llm all you need for time series forecasting? **arXiv preprint arXiv:2311.06625**, 2023. Cited in page 29.

MAKRIDAKIS, S. Accuracy measures: theoretical and practical concerns. **International Journal of Forecasting**, Elsevier, v. 9, n. 4, p. 527–529, 1993. Disponível em: <https://doi.org/10.1016/0169-2070(93)90079-3>. Cited in page 58.

MALKIEL, B. G. The efficient market hypothesis and its critics. **Journal of Economic Perspectives**, American Economic Association, v. 17, n. 1, p. 59–82, 2003. Cited in page 25.

MANOKHIN, V. **Machine learning for probabilistic prediction**. Tese (Doutorado) — Royal Holloway, University of London, 2022. Disponível em: <https://pure.royalholloway.ac.uk/ws/portalfiles/portal/46009689/2022manokhinvphd.pdf>. Cited in page 23.

MOGHAR, A.; HAMICHE, H. Stock market prediction using lstm recurrent neural network. **Procedia Computer Science**, Elsevier, v. 170, p. 1168–1173, 2020. Cited in page 90.

MUKHOTI, J. et al. Calibrating deep neural networks using focal loss. In: **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2020. v. 33, p. 15288–15299. Cited in page 23.

MYTTENAERE, A. D. et al. Forecasting performance measures: What do they mean? **Machine Learning**, Springer, v. 102, n. 1, p. 35–49, 2016. Disponível em: <https://doi.org/10.1007/s10994-015-5529-5>. Cited in page 58.

NELSON, D. M.; PEREIRA, A. C.; OLIVEIRA, R. A. de. Stock market's price movement prediction with lstm neural networks. **2017 International Joint Conference on Neural Networks (IJCNN)**, IEEE, p. 1419–1426, 2017. Cited in page 26.

OLIVEIRA, R. I. et al. **Split conformal prediction for dependent data**. 2022. Disponível em: <https://arxiv.org/pdf/2203.15885.pdf>. Cited in page 99.

PAPADOPOULOS, H. et al. Inductive confidence machines for regression. In: **Machine Learning: ECML 2002, 13th European Conference on Machine Learning, Proceedings**. Springer, 2002. (Lecture Notes in Computer Science, v. 2430), p. 345–356. Disponível em: <https://doi.org/10.1007/3-540-36755-1_29>. Cited 2 times in pages 97 and 98.

PATEL, J.; SHAH, S.; KOTECHA, K. Predicting stock market trends using machine learning algorithms via public sentiments and economic parameters. **International Journal of Intelligent Engineering and Systems**, v. 12, n. 4, p. 205–214, 2019. Cited in page 27.

PATEL, J. et al. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. **Expert Systems with Applications**, Elsevier, v. 42, n. 1, p. 259–268, 2015. Cited 2 times in pages 27 and 45.

QIN, Y. et al. A dual-stage attention-based recurrent neural network for time series prediction. **Proceedings of the 26th International Joint Conference on Artificial Intelligence**, p. 2627–2633, 2017. Cited 3 times in pages 28, 89, and 90.

QIU, M.; SONG, Y.; AKAGI, F. Forecasting stock prices using multi-input lstm networks. **International Journal of Computational Intelligence Systems**, Atlantis Press, v. 13, n. 1, p. 235–245, 2020. Cited in page 26.

ROMANO, J. V. **Conformal Prediction Methods in Finance**. IMPA, 2022. Disponível em: <https://impa.br/wp-content/uploads/2022/11/Projeto_Final_Joao-Vitor-Romano.pdf>. Cited 4 times in pages 22, 23, 31, and 100.

ROMANO, Y.; PATTERSON, E.; CANDES, E. Conformalized quantile regression. **Advances in Neural Information Processing Systems**, v. 32, 2019. Disponível em: <https://doi.org/10.48550/arXiv.1905.03222>. Cited 5 times in pages 35, 40, 50, 99, and 101.

ROMANO, Y.; PATTERSON, E.; CANDES, E. J. Calibrated prediction sets with a rejected option. **Journal of the Royal Statistical Society: Series B (Statistical Methodology)**, Wiley Online Library, v. 84, n. 4, p. 879–913, 2022. Cited in page 97.

ROMANO, Y.; PATTERSON, E.; CANDèS, E. J. Conformalized quantile regression. **Advances in Neural Information Processing Systems (NeurIPS)**, Curran Associates, Inc., v. 32, p. 3543–3553, 2019. Cited 2 times in pages 24 and 29.

ROMANO, Y.; SESIA, M.; CANDES, E. J. Classification with valid and adaptive coverage. **Journal of the American Statistical Association**, Taylor & Francis, v. 117, n. 537, p. 314–328, 2022. Cited in page 30.

SAUNDERS, C.; GAMMERMAN, A.; VOVK, V. Transduction with confidence and credibility. In: **Proceedings of the Sixteenth International Conference on Machine Learning (ICML)**. [s.n.], 1999. p. 722–726. Disponível em: <https://eprints.soton.ac.uk/258961/>. Cited in page 97.

SCHÖLKOPF, B. et al. Estimating the support of a high-dimensional distribution. **Neural computation**, MIT Press, v. 13, n. 7, p. 1443–1471, 2001. Cited in page 91.

SEEDAT, N. et al. Improving adaptive conformal prediction using self-supervised learning. In: **International Conference on Artificial Intelligence and Statistics**. PMLR, 2023. p. 10160–10177. Disponível em: <https://proceedings.mlr.press/v206/seedat23a.html>. Cited 2 times in pages 11 and 86.

SESIA, M.; CANDèS, E. J. A comparison of some conformal quantile regression methods. **Stat**, v. 9, n. 1, p. e261, 2020. Disponível em: <https://doi.org/10.1002/sta4.261>. Cited in page 100.

SHAFER, G.; VOVK, V. A tutorial on conformal prediction. **Journal of Machine Learning Research**, JMLR, v. 9, p. 371–421, 2008. Cited in page 29.

SHAFER, G.; VOVK, V. A tutorial on conformal prediction. **Journal of Machine Learning Research**, v. 9, n. 3, mar 2008. Disponível em: <https://www.jmlr.org/papers/volume9/shafer08a/shafer08a.pdf>. Cited in page 98.

SHAH, R. **Prediction Intervals with Conformal Inference: An Intuitive Explanation**. 2022. Accessed on September 23, 2022. Disponível em: <https://colab.research.google.com/drive/1bA_TrrmRpgJ0jasWBZCxkXSLePi8uWBx>. Cited 5 times in pages 11, 102, 103, 104, and 105.

SMITH, G. The intuitive investor: A behavioural finance explanation of value investing. **Journal of Financial Planning**, v. 16, n. 2, p. 56–63, 2003. Cited in page 25.

SONG, J.; ZHAO, S.; ERMON, S. Distribution calibration for regression and classification. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2021. p. 9847–9857. Cited in page 96.

STAUDEMEYER, R. C.; MORRIS, E. R. Understanding lstm–a tutorial into long short-term memory recurrent neural networks. **arXiv preprint arXiv:1909.09586**, 2019. Disponível em: <https://blog.mlreview.com/understanding-lstm-and-its-diagrams-37e2f46f1714>. Cited 2 times in pages 11 and 76.

SUYKENS, J. A.; VANDEWALLE, J. Least squares support vector machine classifiers. **Neural processing letters**, Springer, v. 9, n. 3, p. 293–300, 1999. Cited in page 91.

TAY, F. E.; CAO, L. Application of support vector machines in financial time series forecasting. **Omega**, Elsevier, v. 29, n. 4, p. 309–317, 2001. Cited in page 27.

TENG, Y.-L. et al. Predictive inference with feature conformal prediction. In: **Proceedings of the 40th International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2023. Cited 2 times in pages 45 and 61.

THAKUR, G.; PADMANABHAN, B.; GUPTA, M. Stock market prediction using svm and pso. In: IEEE. **2011 International Conference on Computer Applications and Industrial Electronics (ICCAIE)**. [S.l.], 2011. p. 486–489. Cited in page 27.

TIBSHIRANI, R.; HASTIE, T. A melting pot. **Observational Studies**, v. 7, n. 1, p. 213–215, 2021. Disponível em: <https://muse.jhu.edu/article/799737>. Cited 2 times in pages 23 and 31.

VERMA, A.; AGRAWAL, S.; SHARMA, A. Explainable ai for stock market prediction: An application of shap with xgboost. **Journal of King Saud University - Computer and Information Sciences**, Elsevier, 2023. In Press. Cited in page 28.

VOVK, V. Cross-conformal predictors. **Annals of Mathematics and Artificial Intelligence**, v. 74, p. 9–28, 2015. Disponível em: <https://doi.org/10.1007/s10472-013-9368-4>. Cited in page 99.

VOVK, V.; GAMMERMAN, A.; SAUNDERS, C. Machine-learning applications of algorithmic randomness. In: **Proceedings of the Sixteenth International Conference on Machine Learning (ICML)**. [s.n.], 1999. p. 444–453. Disponível em: <https://eprints.soton.ac.uk/258960/1/Random_ICML99.pdf>. Cited in page 97.

VOVK, V.; GAMMERMAN, A.; SHAFER, G. **Algorithmic learning in a random world**. [S.l.]: Springer, 2005. Cited in page 29.

VOVK, V.; GAMMERMAN, A.; SHAFER, G. **Algorithmic Learning in a Random World**. Springer, 2005. (Information Science and Statistics). ISBN 978-3-031-06649-8. Disponível em: <https://link.springer.com/book/10.1007/978-3-031-06649-8>. Cited in page 51.

VOVK, V. et al. Criteria of efficiency for conformal prediction. In: **Conformal and Probabilistic Prediction with Applications (COPA)**. PMLR, 2022. (Proceedings of Machine Learning Research, v. 152), p. 1–18. Disponível em: <https://proceedings.mlr.press/v152/vovk21a.html>. Cited in page 52.

VOVK, V. et al. Cross-conformal predictive distributions. In: **Conformal and Probabilistic Prediction and Applications**. PMLR, 2018. p. 37–51. Disponível em: <http://proceedings.mlr.press/v91/vovk18a.html>. Cited in page 99.

VOVK, V.; PAPADOPOULOS, H.; GAMMERMAN, A. Mondrian confidence machine. **Proceedings of the 19th Annual Conference on Neural Information Processing Systems (NIPS)**, MIT Press, p. 1385–1392, 2005. Cited 5 times in pages 24, 35, 40, 97, and 98.

VUONG, Q.-T. et al. Hybrid deep learning and machine learning model for financial time series prediction. **International Journal of Financial Studies**, MDPI, v. 10, n. 1, p. 15, 2022. Cited in page 90.

WIERSTRA, D.; GOMEZ, F.; SCHMIDHUBER, J. Evolino: Hybrid neuroevolution/optimal linear search for sequence learning. In: **Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)**. [S.l.: s.n.], 2005. p. 853–858. Cited in page 88.

WIśNIEWSKI, M.; JASTRZęBSKI, S.; OLSZEWSKI, D. Conformal prediction bands for time series. **Expert Systems with Applications**, Elsevier, v. 160, p. 113699, 2020. Cited in page 30.

ZADROZNY, B.; ELKAN, C. Transforming classifier scores into accurate multiclass probability estimates. In: ACM. **Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.], 2002. p. 694–699. Cited in page 96.

ZECCHIN, A. et al. Generalization and informativeness of conformal prediction. **arXiv preprint arXiv:2402.02497**, 2024. Cited 2 times in pages 45 and 61.

ZHANG, X.; XU, Y.; WANG, T. Stock market prediction via multi-source multiple instance learning. **IEEE Access**, IEEE, v. 6, p. 50720–50728, 2018. Cited in page 27.

Glossary

Asset Price Prediction (APP) The task of forecasting future prices of financial assets based on historical data and various modeling techniques.

Conformal Prediction (CP) A statistical framework that provides reliable prediction intervals with valid coverage guarantees regardless of the underlying distribution.

Conformal Prediction Ensemble (CPE) A hybrid framework that integrates machine learning models with conformal prediction techniques to enhance prediction reliability in asset pricing.

Conformal Prediction Interval (CPI) An interval output generated by conformal prediction methods that defines the range where the true value is expected to lie with a specified confidence.

Heuristic Rules (HR) Empirical rules derived from experience or intuition used in decision-making without formal statistical or mathematical foundations.

Machine Learning (ML) A field of artificial intelligence focused on building systems that can learn patterns from data and make decisions or predictions.

Long Short-Term Memory (LSTM) A type of recurrent neural network capable of learning long-term dependencies in sequential data, useful for time series forecasting.

Support Vector Machine (SVM) A supervised machine learning algorithm used for classification and regression, leveraging kernel methods to find optimal decision boundaries.

eXtreme Gradient Boosting (XGBoost) A powerful ensemble learning algorithm based on gradient boosted decision trees, known for high predictive accuracy and efficiency.

Random Approach (RA) A post-processing technique in which predictions are randomly sampled from the calibrated prediction interval to add stochastic diversity.

Quantitative Trading (QT) A method of trading that uses mathematical models and algorithms to make trading decisions based on quantitative analysis.

Symmetric Mean Absolute Percentage Error (sMAPE) A performance metric that calculates the percentage difference between predicted and actual values in a symmetric form.

Conformalized Quantile Regression (CQR) A method combining quantile regression with conformal prediction to produce valid prediction intervals under distributional uncertainty.

Mondrian Conformal Prediction (MCP) A variant of conformal prediction tailored for non-exchangeable data by conditioning predictions within partitions of the data.

Support Vector Regression (SVR) A regression variant of SVM that predicts continuous outcomes by minimizing error within an epsilon margin.

Open-high-low-close (OHLC) A format for representing financial time series using the open, high, low, and close prices for a given time interval.

High-frequency trading (HFT) A form of trading that uses powerful algorithms to execute a large number of orders at extremely high speeds.

Efficient Market Hypothesis (EMH) A financial theory stating that asset prices fully reflect all available information, making it impossible to consistently achieve superior returns.

Auto-Regressive Conditional Heteroscedasticity (ARCH) A time series model that describes changing variance over time, often used to model financial volatility.

Auto-Regressive Moving Average (ARMA) A classical time series model that combines autoregression and moving average components to model linear relationships in data.

A
Appendix

A.1
Background

This appendix aims to provide additional information about the background of the models and techniques in case the reader wants it.

A.1.1
Machine Learning

This appendix provides additional information about the technique's background if the reader wants it.

A.1.2
LSTM

The *Long-Short Term Memory (LSTM)* is a type of gated *Recurrent Neural Network (RNN)* with improvements. The primary capability of the *LSTM* is its memory pool to save information for further processing. The memory poll has two segments, the *Short-term state* and the *Long-term state*. The first saves the current result from processing, and the second handles the long-term information through processing.

The *LSTM* has a memory pool with two segments, the *Short-term state* and the *Long-term state*. The first saves the current result from processing, and the second handles the long-term information through processing. The *Long-term state* handles what to save, read, or reject based on an *Activation Function (AF)*. Figure A.1 shows one block (unit), similar to a small state machine.

The *gates* have weights that are fitted during the *train* process.

A.1.2.1
Activation Function

The standard *Activation Function (AF)* for the *LSTM* model is the *Hyperbolic Tangent (tanh)*, which is non-linear and has better predictive results than the *Logistic Sigmoid Function* [1] . Other *AF* could provide better results, such as *Rectified Linear Unit (ReLU)*, which is the most used *AF*. The *ReLu* is a *Piece-*

[1]The *Sigmoid Function* is any *strictly increasing*, *monotonic*, *continuous*, and *differentiable* in the complete of *real numbers set*. It has an '*S*' shape curve that is consequently *non-linear*. The most common *Sigmoid* is the *Logistic Sigmoid Function* that can map any *real number* to a value between 0 and 1.

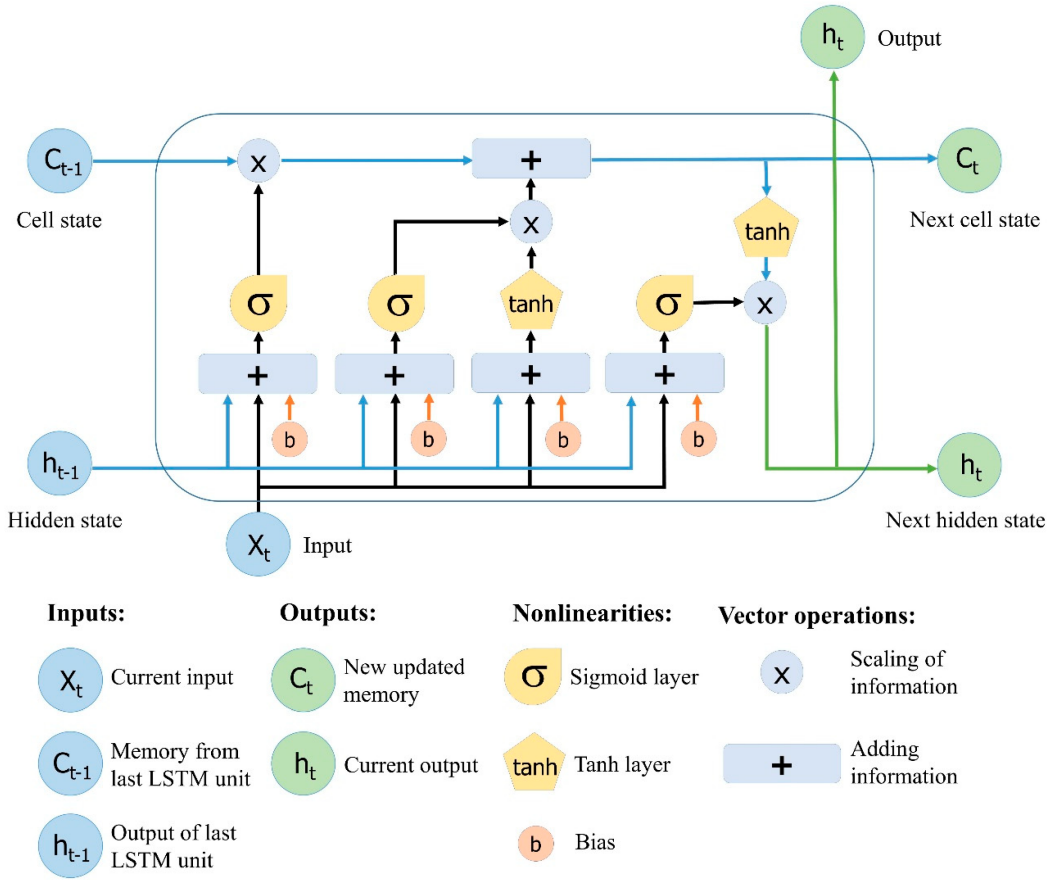


Figure A.1: *The architecture of LSTM unit (STAUDEMEYER; MORRIS, 2019).*

wise Linear Function [2] having two linear pieces, meaning more straightforward computation. Variations of the *ReLU* solve the problem of *dying ReLU*, or *dead neurons*, when it gets stuck, resulting in zero. The variations to be considered are the *Leaky ReLU* and *Randomized ReLU*. These both are monotonic functions, and their derivatives are monotonic, resulting in less computation effort during the optimization using the *Stochastic Gradient Descent (SGD)*.

A.1.3 Gate Function

The standard *Gate Function (GF)* for the *Long Short Term Memory (LSTM)* is the *Logistic Sigmoid Function* [1] , which is an S-shaped curve that is consequently non-linear.

The *LSTM* has three *GF* responsible for filtering information, such as the *Forget Gate*, *Input Gate*, and *Output Gate*. This filtering decides to let in, store, and throw the information away.

[2]The *Piece-wise Linear Function*, or *Hinge Function*, is a nearly linear function that provides a number of linear segments (pieces) over the same quantity of interval.

A.1.3.1 Input Gate

It conditionally decides whether to update the *Memory State* with values from the *Input Data* based on the *Previous Hidden State* and the *Current Input Data*. If the new value is accepted, it will be part of the *Long Term Memory (Cell State)* of the *Neural Network (NN)*.

A.1.3.2 Forget Gate

It conditionally decides whether to forget or not information, based on the new *Input Data* and the previous *Hidden State*. Both are inputs in the block (unit), and they are an output of a *Sigmoid Activation Function*[1], which results in a value between 0 and 1. The 0 means lower relevance, and the 1 means higher relevance.

The old memories will pass through to the next block (unit) if it is fully opened, and the old memories will be kept if it is shut off. subsubsectionOutput Gate

It conditionally decides which information will generate the output on the *Final Hidden State* of the *Neural Network (NN)*. The condition is based on the *Block Memory* and the *Input Data*. The *Block Memory* are the *Cell State (Long-Term Memory)* and the *Previous Hidden State*

The *Output Gate Function* is a *Logistic Sigmoid Function*[1], and it acts as *Activation Function*, which decides if the information will be in the *Updated Cell State*. And if it is relevant enough to result as an output of the *New Hidden State*.

A.1.3.3 Cell State

It has the dependency and relation of the current *Long-Term Memory* of the *Neural Network (NN)*. It avoids the *Gradient Vanishing* on the *LSTM*.

A.1.3.4 Hidden State

The *Previous Hidden State* has the *Previous Time Step* output.

A.1.3.5 Formulation

The main details of the *LSTM* formulation follow as the setup below:

$t = \{1, 2, \ldots, n\}$, data index time from the dataset.

$x =$ feature values.

$y =$ correct target (label) value.

$\hat{y} =$ predicted target (label) value.

$\hat{f}(x) =$ prediction function.

$i(t) = \sigma(W^{ix}.x^{(t)} + W^{ih}.h^{(t-1)} + b_i)$, input gate.

$f(t) = \sigma(W^{fx}.x^{(t)} + W^{fh}.h^{(t-1)} + b_f)$, forget gate.

$o(t) = \sigma(W^{ox}.x^{(t)} + W^{oh}.h^{(t-1)} + b_o)$, output gate.

$g(t) = \tanh(W^{gx}.x^{(t)} + W^{gh}.h^{(t-1)} + b_g)$, go into the cell (memory) state.

$c^{(t)} = f^{(t)} * c^{(t-1)} + i^{(t)} * g^{(t)}$, memory (cell) state.

$h^{(t)} = \tanh(c^{(t)} * o^{(t)}$, hidden state.

A.1.4 SVM

A.1.5 Description

Support Vector Machine (SVM) has undergone significant evolution since its inception, marked by theoretical advancements and diverse applications across various domains. The conceptual framework of SVMs was introduced by Vladimir N. Vapnik and Alexey Ya. Chervonenkis, in 1963, focused on developing linear classifiers capable of separating data points with maximal margin. This approach aimed to enhance generalization capabilities by identifying an optimal hyperplane that maximizes the margin between distinct classes.

In 1992, Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik extended this framework by incorporating the kernel trick, facilitating the creation of nonlinear classifiers. This innovation allowed the algorithm to operate effectively in high-dimensional feature spaces without explicit computation of the transformations, thus broadening the applicability of SVMs to more complex, nonlinearly separable datasets. Building upon the foundational SVM model, several variants have been developed to address specific challenges:

- ***Support Vector Regression (SVR):*** Introduced to adapt SVMs for regression tasks, SVR employs ?-insensitive loss functions to approximate

real-valued functions, enabling the prediction of continuous outcomes.

- ***Least Squares Support Vector Machines (LS-SVMs):*** Proposed by Johan A.K. Suykens and Joos Vandewalle in 1999, LS-SVMs reformulate the standard SVM optimization problem by utilizing a least squares cost function, resulting in a set of linear equations. This modification simplifies the computational complexity associated with traditional quadratic programming in SVMs.
- ***One-Class SVMs:*** Developed by Bernhard Scholkopf et al. in 1999, this variant is tailored for novelty or anomaly detection tasks. It constructs a decision boundary around most data, effectively identifying outliers or novel patterns.
- ***Incremental and Online SVMs:*** Addressing the need for real-time learning in dynamic environments, incremental SVM algorithms have been designed to update the model as new data becomes available, without retraining the entire dataset. This approach is particularly beneficial for applications involving large-scale or streaming data.

SVMs have found extensive applications in financial time series forecasting, attributed to their robustness in handling high-dimensional and nonlinear data. Notable applications include:

- ***Stock Price Prediction:*** SVMs have been employed to forecast stock prices by capturing complex patterns in historical data, demonstrating superior performance compared to traditional models.
- ***Volatility Modeling:*** The capability of SVMs to model nonlinear relationships has been leveraged to predict market volatility, aiding in risk management and derivative pricing.
- ***Credit Risk Assessment:*** SVMs contribute to more accurate credit scoring and risk evaluation by effectively classifying borrowers based on risk profiles.
- ***Algorithmic Trading:*** SVMs facilitate the development of trading strategies by identifying profitable opportunities through pattern recognition in market data.

Comparative Analysis of Advantages and Limitations is: **Advantages**

- ***Robustness to High-Dimensional Data::*** SVMs perform effectively in spaces where the number of dimensions exceeds the number of samples.
- ***Flexibility through Kernel Functions:*** Using various kernel functions allows SVMs to model complex, nonlinear decision boundaries.
- ***Strong Theoretical Foundations:*** Rooted in statistical learning theory, SVMs offer insights into their generalization capabilities.

A.1.5.1 Limitations

- ***Computational Intensity:*** Training SVMs can be resource-intensive, particularly with large datasets, due to the complexity of the optimization problem.
- ***Parameter Sensitivity:*** The performance of SVMs is contingent on the appropriate selection of kernel parameters and regularization terms.
- ***Interpretability:*** The resultant models can be less interpretable than other methods, such as decision trees, posing challenges in domains where model transparency is crucial.

Contemporary research endeavors aim to enhance the scalability, efficiency, and applicability of SVMs:

- ***Scalable Algorithms:*** Efforts are underway to develop algorithms capable of handling large-scale datasets more efficiently, including parallel processing techniques and approximation methods.
- ***Hybrid Models:*** Integrating SVMs with other machine learning models, such as neural networks, to capitalize on the strengths of each and improve predictive performance.
- ***Automated Parameter Selection:*** Research into automatically tuning hyperparameters seeks to alleviate the reliance on manual selection, thereby streamlining the modeling process.
- ***Enhanced Interpretability:*** Developing techniques to render SVM models more transparent and interpretable, facilitating their adoption in fields requiring explainable AI solutions.

Support Vector Machines (SVMs) have demonstrated significant efficacy in financial time series forecasting, particularly in stock market prediction. Their ability to model complex, nonlinear relationships makes them well-suited for the volatile nature of economic data. For instance, Tay and Cao (2001) applied SVMs to forecast financial time series, comparing their performance against traditional back-propagation neural networks. Their findings indicated that SVMs outperformed neural networks regarding predictive accuracy, highlighting their potential in financial forecasting applications.

Similarly, Kim (2003) investigated the application of SVMs in predicting stock price indices. The study concluded that SVMs provided a promising alternative to traditional methods, offering improved financial time series forecasting prediction performance.

In another study, Huang et al. (2005) explored the use of SVMs to predict the weekly movement direction of the NIKKEI 225 index. The results demonstrated

that SVMs outperformed other classification methods, reinforcing their applicability in stock market prediction.

These studies underscore the adaptability and robustness of SVMs in modeling the intricate dynamics of financial markets. By effectively capturing the nonlinear patterns inherent in economic data, SVMs have become a valuable tool in the arsenal of financial analysts and researchers aiming to enhance predictive accuracy in stock market forecasting. Another emerging direction is online and stochastic SVMs, designed to handle streaming data and massive datasets. Algorithms like LASVM (Bordes et al., 2005) and Pegasos (Shalev-Shwartz et al., 2011) introduced stochastic gradient descent-based training procedures for SVMs, enabling real-time learning and reducing memory footprints. These enhancements further expanded the practical utility of SVMs in high-velocity data environments. Finally, conformal prediction frameworks have been recently applied to SVMs to provide confidence sets around predictions. Although conformal prediction is model-agnostic, its application to SVMs has shown promising results in creating reliable and calibrated decision boundaries with provable guarantees. This reflects the continuous theoretical enrichment of SVMs toward interpretable and trustworthy AI systems. In summary, the evolution of Support Vector Machines (SVMs) has been marked by a continuous stream of innovations, ranging from architectural flexibility and kernel-based learning to probabilistic reasoning, computational scalability, and recent integrations with deep learning and uncertainty quantification frameworks. These advancements have preserved the theoretical rigor of the original formulation while significantly expanding the scope and expressiveness of SVM-based models across diverse and complex domains.

SVM remains a foundational algorithm in machine learning, striking a balance between robust theoretical grounding and practical effectiveness. Its adaptability to classification and regression tasks and its resilience in handling nonlinearity and high-dimensionality has sustained its relevance in academic and industrial applications–particularly in financial time series forecasting.

While newer approaches, such as deep learning, have gained prominence, SVM remains a reliable baseline and frequently plays a role in ensemble and hybrid architectures. The ongoing convergence of SVM with scalable computational techniques and interpretable machine learning methodologies ensures that it will remain a critical component of predictive modeling frameworks well into the foreseeable future.

A.1.5.2 Formalization

The Equation (A-1) section aims to describe the standard formulation of the Support Vector Machine (SVM) for binary classification in LaTeX, followed by a detailed description of each component.

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - \mathbf{w}^\top\phi(\mathbf{x}_i) - b \le \varepsilon + \xi_i \\ & \mathbf{w}^\top\phi(\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \ge 0, \quad i = 1, \ldots, n \end{aligned} \tag{A-1}$$

A.1.5.3 Variables and Constants:

- $\mathbf{x}_i \in \mathbb{R}^d$: Input feature vector of the i-th training sample.
- $y_i \in \mathbb{R}$: Real-valued target/output for the i-th sample.
- $\phi(\cdot)$: Optional mapping function to a high-dimensional feature space (used with the kernel trick).
- $\mathbf{w} \in \mathbb{R}^d$: Weight vector of the regression model.
- $b \in \mathbb{R}$: Bias (intercept) term of the model.
- $\varepsilon > 0$: Insensitivity margin; deviations within $\pm\varepsilon$ are not penalized.
- $\xi_i, \xi_i^* \ge 0$: Slack variables allowing deviations above and below the ε-tube, respectively.
- $C > 0$: Regularization parameter that controls the trade-off between the flatness of the function and the tolerance to errors.
- $K(\mathbf{x}_i, \mathbf{x}_j)$: Kernel function representing the inner product in the feature space, i.e., $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top\phi(\mathbf{x}_j)$.
- α_i, α_i^*: Lagrange multipliers from the dual optimization formulation.
- n: Total number of training samples.

The Equation (A-2) is the decision function in kernelized form.

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) + b \tag{A-2}$$

A.1.6 XGBoost

The *Decision Tree (DT)*, as the name suggests, is a *supervised learning algorithm* based on *Decision Rules (DR)* using a *tree data structure*. The intuition behind the *DT* is to create multiple criteria (rules) to split each *tree's node* into further two or more *sub-nodes* as if it was splitting the dataset among the *nodes*.

The *DT* main benefits are: It is *Non-Parametric*, so it does not require strong assumptions, mainly related to the *data distribution*; It is considered as *White-Box*[3] because it is easy to understand and interpret by *boolean logic*; It does not require *data normalization* and *scaling*[4]; Missing values do not impact it; It performs the automatic *features selection* that is important to reduce the *data noise*, and also provide the *feature* significance ranking; It is not impacted by *outliers*; it can be used for *categorical* and *numerical data* as well as *classification and regression problems*; It handles *non-linear* data correlation (pattern).

The *DT drawbacks* are: It memorizes the *train* data, *overfitting* its; It could have weak *generalization* to deal with *Zero-shot* data, even after the *overfitting* been addressed; It is unstable because a small data change results in a considerably different *tree structure*, which is known as *variance*, and it is mitigated by using *Ensemble Learning (EL)*; It tends to *bias* favoring the significant *target outcome* (class or label), especially in the face of *unbalanced classes*; It has *Non-Smoothness Decision Boundaries* (abrupt) because the *boundary decision rules* are discrete in the form of a stair (not a curve), and it could not be desirable in the *regression problems*; It has *greedy* approach to compound the *Decision Rules (DR)* in each tree's node, and it may not achieve the optimal tree structure, which is the case of *high-dimensional* data; And others.

The *EL* techniques are used by multiple *DT* to mitigate the *overfitting* problem and increase the prediction performance. The most common *EL* for *DT* is based on *Bagging* or *Boosting* approaches, such as *Random Forest* and *Gradient Boost*, respectively.

The *Bagging* executes different *training* to create several *Experts* [5] or *Weak Learners (WL)* using different *dataset's sub-set* (*bootstrapping)*, all *features*, and all *target outcome* (class or label). The *Bagging* flow to generate each *Expert* is

[3] A model is considered a *Black-Box* when it is difficult to explain and interpret, such as *Artificial Neural Network (ANN)*. By contrast, the *White-Box* is easier to explain and interpret.

[4] The *Scaling* is a preprocessing data transformation technique that takes one or more features (aleatory variable) to the same scale (order of magnitude). It is essential to stabilize (converge) the *Machine Learning (ML)* models, mainly for those based in *Neural Network (NN)*.

[5] The *Expert* is an entity with a set of rules enough to support the *decision-making* process.

parallel, which means that one *Expert* generation will not affect the other. In the end, it provides the *final prediction result* based on *majority voting* or *statistical calculation* over each generated *Expert*. The *Random Forest*, which uses *Bagging*, generates multiple *trees* based on *random* different *dataset's sub-set* and also the features. Each *DT* is considered as a *Expert* with the necessary rules to provide the *Intermediate Prediction Result (IPR)* for each specifically *DT*. The *final prediction result* for the *regression problems* mainly uses the *average*.

The *Boosting* executes *Weak Learners (WL)* and combines them, interactively, creating a *Strong Learners (SL)* . It uses the full dataset, all *features*, and just one *target outcome* (class or label). Each interaction creates a standalone composite model, which improves the *Intermediate Prediction Result (IPR)* of the *SL*. After each iteration, the *mispredicted* instances (data points) receive more attention until these instances are correctly *predicted*. The *Boosting* flow to generate each *SL* is sequential, which means that the previously generated *SL* will positively affect the next *SL*. In the end, the *final prediction result* is exactly the last generated *SL*, which has the better *accuracy*. The *Gradient Boost* generates multiples *trees* based on *random* different sub-sets of the dataset and also the features.

A.1.6.1 Formulation

The main details of the *XGBoost* formulation follow as the setup below:

$$
\begin{aligned}
i &= \{1, 2, \ldots, n\}, \text{data index from the train dataset.} \\
t &= \{1, 2, \ldots, m\}, \text{iteraction index.} \\
x &= \text{feature values.} \\
y &= \text{correct target (label) value.} \\
\hat{y} &= \text{predicted target (label) value.} \\
\hat{f}(x) &= \text{prediction function.} \\
l(y, \hat{y}) &= \text{Differentiable CART learners and loss function to be minimized.} \\
\mathcal{L}_t &= \sum_{i=1}^{n} l(y_t(x_i), \quad \hat{y}_{(t-1)}(x_i) + \hat{y}_t(x_i)) + \Omega(\hat{y}_t(x_i)) \\
\hat{f}_0(x) &= \arg\min_{\theta} \sum_{i=1}^{n} l(y(x_i), \theta), \text{initial constant value.}
\end{aligned}
$$

A.1.7 Conformal Prediction

This appendix provides additional information about the technique's back-

ground if the reader wants it.

A.1.7.1
CP - Conformalized Quantile Regression

The *Conformalized Quantile Regression (CQR)* is a method to *Quantify the Accuracy* of the predictions. It combines the *Split CP* technique and *Quantile Regression (QR)* method.

The key advantages of the *CQR* are *distribution-free* and adaptiveness on the *heteroscedastic*[6]data. Due to the adaptability of each event over time, it is fundamental in various real cases, such as the *asset´s price* in the financial market. It means the *Conformal Prediction Interval (CPI)* varies in length based on the previous event covariance. When the covariance increases, the *CPI* also increases, which is analogous to the covariance decreasing case.

The *QR* in detail has the key disadvantages of the *asymptotical coverage guarantee* and not be *model-free*.

A.1.7.2
CP - Non-Conformity Scores

The *Conformal Prediction (CP)* requires a function to return the *prediction score*. This score function is known as *Non-Conformity Scores (NCS)*, *Non-Conformity Measure (NCM)*, or just the *Conformity Scores (NCS)* function. The intuition behind the *NCS* is to know how far the *predicted value* is from the *correct value*.

The *coverage guarantees* of the *CP* are unrelated to the *NCS* function. However, the *NCS* function influences the size (width) of the *Prediction Interval (PI)* set. The size of the *PI* is associated with the *Efficiency Criteria (EC)*. Small *PI*'s size is related to high *EC* because it is a more concise result set, and large size is related to small *EC*.

The simplest *NCS* function is the *Regression Residual (RR)*, (LEI et al., 2018). It is the difference between the *correct value* and the *predicted value*. In math worlds, it is equal to the absolute error $\hat{\epsilon}_{train}(x, y) = ||y - \hat{y}(x)||_2$, or just the $L2 - Norm$[7] of the error.

[6]The *Heteroscedasticity*, or *Heteroskedasticity*, occurs when the *variance* is not constant over time.

[7]$||A||_2$ is the $L2 - Norm$ of the A, and the $L2 - Norm$ is the *Standard Euclidean Norm* or just *Euclidean Distance*. It is equivalent to $||A||_2 = \sqrt{A^2}$.

A.1.7.3
CP - Self-supervised Learning

Figure A.2, Seedat et al. (2023), shows the three different approaches to improve the adaptiveness of the *CP*. The *Self-supervised Learning* improves a signal for adaptiveness.

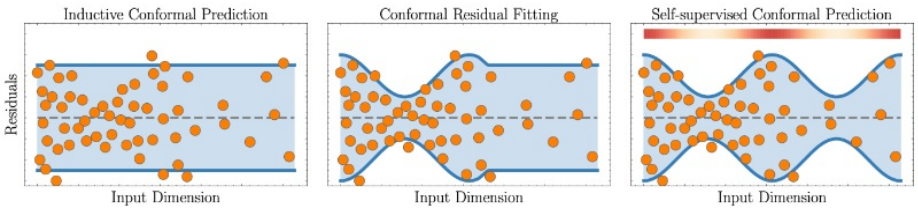


Figure A.2: Approaches for Comformal Prediction.(SEEDAT et al., 2023)

A.2
Review

This appendix aims to provide additional information about the review of this research in case the reader wants it.

A.2.1
Prediction

This appendix provides additional information from the literature review on *asset price prediction (APP)*, focusing on discovering predictive models and their enhancements over the years.

A.2.1.1
Before the LSTM

Cootner (1964) observed a *random walk* behavior on the daily *asset price*. Fama (1965) mentioned an intriguing question that was already tormenting the academy and the *financial market (FM)*, where it would be possible to identify patterns of behavior in prices. White (1988) seems to be the first to use the *Neural Networks (NN)* for daily prediction of the *asset returns*[16] of the IBM company with very optimistic results. S. Hochreiter (1991) discussed *Vanishing Gradient Problem (VGP)* [8] in his thesis, advised by J. Schmidhuber. Baba and Kozaki (1992), and Chenoweth and Obradovic (1996) argued about *Feed-forward Neural Network (FNN)*. Roman (1996) used *backpropagation* and *RNN* to predict the *ATP*. *asset price prediction (APP)* begins popularity with the *Artificial Stock*

[8]The *Vanishing Gradient Problem (VGP)* occurs when the *Descendent Gradient (DG)* goes to *zero*.

Market (ASM) [Arthur et al., 1997]. Schuster (1997) approached the *Birectional Recurrent Neural Network (Bi-RNN)* which was trained in both directions.

A.2.1.2 LSTM

Lawrence (1997b) conducted a survey on *asset price prediction (APP)* using *neural networks (NN)*. He challenged the validity of the *Efficient Market Hypothesis (EMH)* by presenting empirical evidence that contradicted its core assumptions and demonstrated the potential of data-driven approaches in financial forecasting.

Hochreiter e Schmidhuber (1997) introduced the *Long Short-Term Memory (LSTM)* model as a specialized form of *Recurrent Neural Network (RNN)* to mitigate the *Vanishing Gradient Problem (VGP)*. The architecture featured memory cells equipped with input, output, and forget gates, supported by *Constant Error Carousel (CEC)* units, enabling the model to retain information over extended sequences and overcome training difficulties in deep recurrent structures.

Gers, Schmidhuber e Cummins (2000) proposed a refinement to the original LSTM architecture by introducing an adaptive *forget gate*. This innovation enabled the model to learn when to reset its internal memory, improving adaptability in dynamic environments and non-stationary time series, features especially important for modeling financial markets.

Chan et al. (2000) proposed two enhancements for training neural networks in asset price prediction (APP): the *conjugate gradient learning (CGL)* algorithm and the *multiple linear regression weight initialization (MLRWI)* technique. These addressed the slow convergence and suboptimal initialization issues of the standard *steepest descent* algorithm in backpropagation, especially when applied to daily financial data.

Gers, Schraudolph e Schmidhuber (2002) demonstrated that LSTM models are capable of learning both *Context-Free Languages (CFL)* and, for the first time, *Context-Sensitive Languages (CSL)*, thereby extending their applicability beyond regular sequence tasks to formal language processing. This milestone established LSTM as a more general-purpose learner in structured sequential domains.

Hochreiter e Schmidhuber (1997) further positioned LSTM as an effective *meta-learner*, a model that can learn how to learn. Their theoretical and empirical results showed that LSTM could match the performance of *Hidden Markov Models (HMM)* in sequence modeling tasks, without requiring changes to the core LSTM algorithm.[9]

[9]The *Meta-Learn*, also known as *learning-to-learn*, is a machine learning approach where one learning algorithm is trained to support the learning process of another. Common examples include *transfer learning*, *hyperparameter optimization*, and *ensemble learning*.

Malkiel (2004) provided empirical evidence that asset prices do not always follow a *random walk*. He observed increased predictability during certain market phases, such as short-term momentum driven by positive serial correlation and long-term mean reversion caused by negative serial correlation. Additionally, he noted that valuation metrics like *price-to-earnings (P/E)* ratios tend to vary with interest rates and dividend yields, suggesting potential features for forecasting models in APP.

Graves e Schmidhuber (2005) introduced the *Bidirectional LSTM (Bi-LSTM)* for phoneme classification, showing that bidirectional architectures, trained using *Backpropagation Through Time (BPTT)*, outperformed standard RNNs and multilayer perceptrons in modeling sequential dependencies. Their results established Bi-LSTM—sometimes referred to as *Vanilla LSTM*—as a state-of-the-art model, widely adopted in domains requiring context-rich time series modeling, including financial forecasting.

Wierstra, Gomez e Schmidhuber (2005) presented *Evolino* (EVOlution of systems with LINear Outputs), a neuroevolution framework for training recurrent networks like LSTM on tasks that demand long-term memory, such as context-sensitive language processing and time series forecasting. Evolino optimizes internal state weights using evolutionary algorithms, avoiding overfitting and local minima issues. The framework performed well on benchmark datasets, including the *Mackey-Glass System (MGS)*,[10] by using linear regression or quadratic programming to map hidden states to outputs.

Graves et al. (2006) introduced the *Connectionist Temporal Classification (CTC)* loss to enhance LSTM's performance in unsegmented sequence learning. The CTC framework enabled end-to-end training by aligning predicted labels with temporal input data, removing the need for pre-segmented training examples or post-processing steps. This innovation allowed a single network to simultaneously handle alignment and recognition tasks, outperforming traditional HMM and hybrid HMM-RNN models in speech recognition.

Mayer et al. (2006) applied LSTM models trained with *Evolino* to improve robotic execution of surgical knot-tying. Their work demonstrated the architecture's effectiveness in learning continuous, precision-dependent movements, underlining LSTM's versatility in real-time control and robotics.

Graves e Schmidhuber (2007) extended the CTC-BLSTM framework with *policy gradients* for reinforcement learning tasks. They demonstrated a discriminative keyword spotting system trained without supervision, achieving 84.5% accuracy, significantly outperforming the HMM baseline (65.0%).

[10]The *Mackey-Glass System* is a nonlinear differential equation used to model chaotic biological processes. It is widely used as a benchmark for evaluating time series prediction algorithms due to its complex, non-periodic behavior.

Their results validated the architecture's capability to model long-range dependencies in bioinformatics.

Graves et al. (2009) demonstrated the practical superiority of LSTM models trained with Connectionist Temporal Classification (CTC) by winning two handwriting recognition categories at the ICDAR 2009 competition. This was the first time a recurrent neural network (RNN) achieved such recognition in an international benchmarking event, affirming LSTM's capability in real-world sequence classification tasks.

Cho et al. (2014) introduced the *Gated Recurrent Unit (GRU)* as a simplified RNN architecture. Designed initially for neural machine translation, GRU uses only two gates—reset and update—eliminating the need for a separate cell state and output gate. This streamlined design addresses the *Descendent Gradient Dissipation Problem (DGDP)*[11], reducing training complexity while maintaining performance in sequence modeling.

Samarawickrama (2017) compared several RNN-based architectures, including GRU, simple RNNs, and LSTM, on daily asset price prediction (APP). The study evaluated predictive accuracy and found that LSTM and GRU models outperformed basic RNNs in capturing short-term dependencies in financial data.

Greff et al. (2016) conducted a comprehensive analysis of LSTM variants, assessing the influence of key architectural components such as peephole connections and gate couplings. Their empirical results suggested that many commonly used components could be omitted with minimal loss in performance, guiding the design of more efficient LSTM-based models.

Qin et al. (2017) proposed the Dual-Stage Attention-Based Recurrent Neural Network (DA-RNN), which combines LSTM with input and temporal attention mechanisms. This hybrid framework improves interpretability and accuracy in multivariate time series forecasting, including applications in finance.

Lai et al. (2018) introduced the LSTNet architecture, integrating convolutional layers, LSTM modules, and autoregressive components. This hybrid model captures both short-term patterns and long-term dependencies in periodic financial time series, significantly improving predictive robustness.

Moghar (2020) explored the impact of training epochs on LSTM performance in stock market prediction. The findings underscored that tuning epoch count is critical for balancing learning sufficiency and overfitting risk in financial time series forecasting.

Graves et al. (2009) demonstrated the practical superiority of LSTM models trained with Connectionist Temporal Classification (CTC) by winning two hand-

[11]The *Descendent Gradient Dissipation Problem (DGDP)* refers to the progressive loss of signal in backpropagation through time, which limits effective learning across long sequences.

writing recognition categories at the ICDAR 2009 competition. This was the first time a recurrent neural network (RNN) achieved such recognition in an international benchmarking event, affirming LSTM's capability in real-world sequence classification tasks.

Cho et al. (2014) introduced the *Gated Recurrent Unit (GRU)* as a simplified RNN architecture. Designed initially for neural machine translation, GRU uses only two gates—reset and update—eliminating the need for a separate cell state and output gate. This streamlined design addresses the *Descendent Gradient Dissipation Problem (DGDP)* 11 , reducing training complexity while maintaining performance in sequence modeling.

Samarawickrama (2017) compared several RNN-based architectures, including GRU, simple RNNs, and LSTM, on daily asset price prediction (APP). The study evaluated predictive accuracy and found that LSTM and GRU models outperformed basic RNNs in capturing short-term dependencies in financial data.

Greff et al. (2016) conducted a comprehensive analysis of LSTM variants, assessing the influence of key architectural components such as peephole connections and gate couplings. Their empirical results suggested that many commonly used components could be omitted with minimal loss in performance, guiding the design of more efficient LSTM-based models.

Qin et al. (2017) proposed the Dual-Stage Attention-Based Recurrent Neural Network (DA-RNN), which combines LSTM with input and temporal attention mechanisms. This hybrid framework improves interpretability and accuracy in multivariate time series forecasting, including applications in finance.

Lai et al. (2018) introduced the LSTNet architecture, integrating convolutional layers, LSTM modules, and autoregressive components. This hybrid model captures both short-term patterns and long-term dependencies in periodic financial time series, significantly improving predictive robustness.

Moghar e Hamiche (2020) explored the impact of training epochs on LSTM performance in stock market prediction. The findings underscored that tuning epoch count is critical for balancing learning sufficiency and overfitting risk in financial time series forecasting.

Vuong et al. (2022) proposed a hybrid model integrating *XGBoost* for feature selection and *LSTM* for sequential learning in the context of stock and forex forecasting. Their results, tested on a Forex dataset from 2008 to 2018, demonstrated superior performance compared to the ARIMA baseline across MAE, MSE, and RMSE metrics, highlighting the synergy between gradient boosting and deep learning.

Kumar (2023) developed a model combining a four-layer LSTM architecture with a Hidden Markov Chain (HMC) to forecast stock prices. By leveraging

the temporal dynamics captured by LSTM and the probabilistic state transitions modeled by HMC, the hybrid approach provided robust predictions with enhanced interpretability, using RMSE and steady-state distribution analysis.

Together, these works illustrate the evolution of LSTM from a theoretical breakthrough to a cornerstone of sequential modeling, continuously extended by architectural refinements, integration with attention, and combination with other deep learning components. Its effectiveness in modeling complex temporal dynamics continues to make it a widely adopted approach for asset price prediction and other financial forecasting tasks.

Graves (2013) presented the *Stacked LSTM (S-LSTM)*, combining multiple levels of the *LSTM* for speech recognition.

A.2.1.3 SVM

The foundational work on Support Vector Machines (SVMs) was introduced by Cortes e Vapnik (1995b), where the authors formulated the problem of finding a hyperplane that maximizes the margin between two linearly separable classes. This seminal contribution established SVM as a powerful and theoretically grounded approach to binary classification. Building on Vapnik's statistical learning theory, this work laid the foundation for further developments, particularly in extending SVM to non-linear and more complex data structures.

A significant enhancement came with introducing the kernel trick, which allows SVMs to operate in high-dimensional feature spaces without explicitly computing the transformation. This extension, presented by Boser, Guyon e Vapnik (1992), enabled SVMs to learn non-linear decision boundaries by using kernel functions such as the radial basis function (RBF), polynomial, and sigmoid kernels.

To adapt SVMs for regression tasks, Drucker et al. (1997) proposed the Support Vector Regression (SVR) framework, which applies the same maximal margin principles to continuous-valued outputs. SVR introduced the concept of an ϵ-insensitive loss function, allowing the model to ignore minor deviations and focus on capturing significant trends, a beneficial property in financial time series forecasting.

The Least Squares SVM (LS–SVM) was another vital variation, introduced by Suykens e Vandewalle (1999), where the traditional convex quadratic programming problem was reformulated into a system of linear equations. This modification simplified the optimization process, making the algorithm more efficient and suitable for large-scale problems.

Further, Schölkopf et al. (2001) introduced the One-Class SVM, designed for unsupervised anomaly detection, a technique particularly relevant for identifying

rare events in financial markets, such as crashes or abrupt trend reversals. One-Class SVMs estimate the support of a high-dimensional distribution by separating normal data from outliers using an implicit kernel space.

Over the years, SVMs have been continually enhanced through algorithmic and computational innovations. These include online and incremental learning frameworks, developed to handle real-time data streams, and scalable solvers optimized for high-dimensional datasets. In recent years, SVMs have also been integrated with deep learning models to combine feature learning with robust decision boundaries. Though still under active research, these hybrid approaches aim to overcome the interpretability limitations of deep neural networks while maintaining the predictive strength of SVMs.

Despite the emergence of complex deep learning architectures, SVMs remain a strong benchmark due to their solid theoretical foundation, robust generalization properties, and broad applicability in domains such as bioinformatics, image processing, and especially financial forecasting. Their ability to manage nonlinear, high-dimensional data makes them well-suited for tasks like asset price prediction, where the signal-to-noise ratio is typically low and the underlying patterns are highly nonlinear and non-stationary.

Thus, the trajectory of SVM research reflects a balance between theoretical rigor and practical adaptability. The original principles laid out by Vapnik and colleagues have stood the test of time and evolved to meet the demands of contemporary machine learning tasks.

A.2.1.4 XGBoost

Morgan and Sonquist (1963) introduced the first decision tree (DT) model as part of the AID project, called Automatic Interaction Detection (AID), which pioneered binary regression tree structures.

Hunt (1966) published the first formal academic paper on decision trees, providing foundational theory for recursive partitioning.

Messenger and Mandell (1972) developed the first classification tree algorithm, THAID, tailored specifically for categorical target prediction.

Breiman et al. (1974) created the Classification and Regression Tree (CART) algorithm, which formalized binary tree structures for both regression and classification tasks.

Breiman et al. (1977) released the first software for implementing CART, enabling practical application of decision tree algorithms in data analysis.

Kass (1980) proposed the CHAID algorithm, using chi-squared statistics for splitting and Bonferroni corrections to control type I error, though often

conservatively.

Breiman et al. (1984) enhanced CART with pruning, tunneling, and subtree selection to avoid overfitting and improve model generalization.

Gordon and Olshen (1985) extended CART for survival analysis using Minimum Wasserstein Distance between Kaplan-Meier curves and mass points as node impurity measures (NIM).

Quinlan (1986) introduced ID3, a non-binary tree algorithm that uses information gain ratio to guide splits, enabling multi-branch partitions.

Ciampi et al. (1988) presented RECPAM, adapting CART to censored data through Proportional Hazards Likelihood Ratio tests.

Loh and Vanichsetakul (1988) proposed FACT, which implemented linear splits using Recursive Linear Discriminant Analysis.

Segal (1988) and Davis and Anderson (1989) independently extended CART to censored data with the Log-Rank Statistic Test as the NIM.

Ciampi (1991) integrated CART with Generalized Linear Models, increasing flexibility in handling different outcome distributions.

Segal (1992) adapted CART for longitudinal data analysis, enhancing its applicability for repeated measurements.

LeBlanc and Crowley (1992) applied Proportional Hazards Log-likelihood for splitting in survival trees.

Quinlan (1992) proposed M5, the first model tree with piecewise linear regression at the leaves.

Quinlan (1993) released C4.5 and C5.0, extending ID3 by introducing pruning, support for numeric features, and better efficiency.

Breiman (1996) introduced Bagging, an ensemble approach that averages predictions over bootstrapped samples to reduce variance.

Alexander and Grimshaw (1996) extended CART with Simple Linear Regression to improve regression accuracy.

Loh and Shih (1997) developed QUEST, a fast and unbiased splitting strategy that corrects selection bias.

Torgo (1997) extended M5 by using Kernel and Nearest-Neighbor models in terminal nodes.

Chipman et al. (1998) and Denison et al. (1998) developed Bayesian CART, introducing probabilistic modeling of tree structures.

Zhang (1998) extended CART to support multiple binary response variables.

Kim and Loh (2001) proposed CRUISE, improving unbiased split selection for both classification and regression.

Breiman (2001) developed Random Forests (RF), combining bagging and random feature selection to grow unpruned trees and introducing variable importance metrics.

De'ath (2002) created MVPART, extending CART for multivariate responses.

Chaudhuri and Loh (2002) enhanced GUIDE with quantile regression.

Loh (2002) broadened GUIDE to include bootstrap bias correction, bagging, and RF.

Chipman et al. (2002) further advanced Bayesian CART.

Kim and Loh (2003) improved CRUISE for performance and interpretability.

Chan and Loh (2004) extended GUIDE with logistic regression.

Dusseldorp and Meulman (2004) proposed RTA for adaptive spline modeling in trees.

Su et al. (2004) applied MVPART to multivariate applications.

Lee (2005) refined multivariate response modeling in CART.

Fan and Gray (2005) introduced TARGET, combining trees and genetic algorithms.

Loh (2006) extended GUIDE with Poisson regression.

Guerts et al. (2006) developed Extra Trees, an ensemble that uses full datasets with randomized splits.

Hothorn et al. (2006) proposed CTREE, using permutation tests to eliminate variable selection bias.

Zeileis et al. (2008) presented MOB, a permutation-based model tree method.

Fan and Gray (2008) refined TARGET's optimization.

Su et al. (2008, 2009) proposed Interaction Trees for treatment effect heterogeneity.

Loh (2009) enhanced GUIDE with kernel methods, deeper interactions, and robust handling of missing values.

Dusseldorp et al. (2010) introduced STIMA for subgroup analysis.

Chipman et al. (2010) proposed BARD, a Bayesian ensemble for uncertainty quantification.

Foster et al. (2011) introduced Virtual Twins for personalized treatment effect modeling.

Lipkovich (2011) developed SIDES for subgroup identification in trials.

Sela and Simonoff (2012) proposed Random Effect Trees to account for hierarchical data.

Loh and Zheng (2013) expanded GUIDE for longitudinal and multiresponse tasks.

Loh (2014) reviewed 50 years of decision tree developments.

Loh et al. (2015) incorporated proportional hazards regression in GUIDE.

Schapire (1990) formally introduced boosting under the PAC learning framework, forming the theoretical base of ensemble learning.

Friedman (1991) proposed MARS, using splines for nonlinear regression.

Jordan and Jacobs (1994) presented Hierarchical Mixtures of Experts, combining expert models with probabilistic gating.

Freund (1995) introduced a general boosting framework.

Freund and Schapire (1997) presented AdaBoost.M1, a discrete classification booster using reweighting.

Friedman and Fisher (1999) created PRIM for interpretable rule-based prediction.

Friedman et al. (2000) adapted AdaBoost for continuous outputs.

Friedman (2001) introduced Gradient Boosting Machine (GBM) and shrinkage to improve regularization.

Chen and Guestrin (2016) proposed XGBoost, a fast, scalable gradient boosting framework with second-order optimization and regularization.

Ke et al. (2017) introduced LightGBM, incorporating GOSS and EFB for scalability.

Prokhorenkova et al. (2018) proposed CatBoost for categorical feature support using ordered boosting.

He et al. (2020) presented asynchronous optimization for distributed XGBoost.

Januschowski et al. (2021) reviewed GBTs' dominance in M5 and other structured data competitions.

These contributions chronicle the evolution of boosting—from its PAC-learning roots and AdaBoost foundations, to powerful frameworks like XGBoost, LightGBM, and CatBoost—showing how regularization, scalability, and categorical feature modeling have shaped the state-of-the-art in gradient boosting for machine learning.

A.2.2 Calibration

This appendix aims to provide additional information about the literature review of the *prediction calibration* in case the reader wants it.

A.2.2.1 Full CP

The *Full CP* denomination comes from the idea of using the entire dataset, different from other *CP* variants that will be presented in the sequence. Because of its solid theoretical methodology, the *Full CP* is the basis of all further *CP* variations.

Platt (1999) laid the foundation for modern calibration techniques by introducing Platt Scaling, a logistic regression model trained on the outputs of a classification algorithm to transform uncalibrated scores into calibrated probabilities. Initially applied to Support Vector Machines (SVMs), this method remains widely used due to its simplicity and effectiveness in binary classification tasks.

Zadrozny e Elkan (2002) advanced prediction calibration by proposing Isotonic Regression as an alternative to Platt Scaling. This non-parametric approach proved more flexible, especially for models where the assumption of a sigmoid-shaped score-probability relationship does not hold. They also formalized a framework for evaluating probabilistic predictions and demonstrated the method's general applicability across classification models.

Guo et al. (2017) conducted a comprehensive empirical study on the calibration properties of modern deep learning models, revealing that despite improvements in accuracy, neural networks tend to be poorly calibrated. Their work emphasized the tradeoff between confidence and accuracy in over-parameterized models. It reintroduced Temperature Scaling as a simple yet effective post-hoc method to recalibrate softmax outputs in deep classifiers.

Kuleshov, Fenner e Ermon (2018) extended calibration techniques to regression models, a less explored domain. They developed a method based on conformal prediction and quantile regression to generate calibrated predictive intervals for continuous outputs. This represented a shift from calibrating class probabilities to producing reliable confidence intervals in regression tasks, with strong theoretical guarantees under mild assumptions.

Levi et al. (2020) introduced diagnostic tools and evaluation metrics tailored for probabilistic regression calibration, such as the Probability Integral Transform (PIT) histogram and empirical coverage plots. Their analysis explained when and how regression models produce miscalibrated predictions and how calibration can be corrected or diagnosed during model evaluation.

Song, Zhao e Ermon (2021) proposed Distribution Calibration as a novel technique for few-shot learning. By aligning the support and query sets distributions using calibrated scores, they demonstrated improved generalization in low-data regimes. This approach highlights the emerging intersection between calibration

and transfer learning, especially when confidence quantification is crucial.

Romano, Patterson e Candes (2022) unified probabilistic classification and regression calibration under the conformal prediction and conditional coverage framework. They provided rigorous formulations for calibrating predictive distributions and proposed efficient algorithms that ensure distributional guarantees. This work represents a significant theoretical advancement in general-purpose calibration strategies with valid statistical guarantees.

Hoff, Kuleshov e Ermon (2022) addressed the challenge of achieving both calibration and sharpness in regression. They analyzed the inherent tradeoff between producing narrow predictive intervals (sharpness) and maintaining empirical coverage (calibration), proposing loss functions that explicitly balance these objectives. Their contributions enhance the practicality of calibrated regression in risk-sensitive applications.

The *Conformal Prediction (CP)* starts with Gammerman, Vovk e Vapnik (1998), although without the explicit *CP* name. They described a procedure to quantify the degrees of confidence on the *Support Vector Machine (SVM)*'s predictions. Saunders, Gammerman e Vovk (1999) presented a new algorithm, based on the Gammerman, Vovk e Vapnik (1998), producing two *Efficiency Criteria (EC)* of a prediction. The *EC* is a reliability indicator composed by the measures of *confidence* and *credibility*. The first highest *p-value*, p_1, determines the *credibility* of the class predicted by the *ML* model. The second highest *p-value*, p_2, determines the *confidence* $1-p_2$ of the prediction. Vovk, Gammerman e Saunders (1999) presented a method to measure the confidence of *Support Vector Machine (SVM)* prediction on pattern recognition problem. Papadopoulos et al. (2002) approached of replacing the *Transductive Inference*[13] with the *Inductive Inference*[13] in regression problems based on *Ridge Regression (RR)* to deal with large datasets.

Vovk, Papadopoulos e Gammerman (2005) proved the validity of conformal sets given *Exchangeable* [12] data and a *miscoverage level* $\alpha \in (0,1)$ considering a unique dataset case. The confidence level is specified by the user as $1-\alpha$. They also formulate how to measure the *non-conformity*. The guarantees proved by Vovk, Papadopoulos e Gammerman (2005) are the baseline of most further research about *CP*.

As the *Full CP* uses the whole dataset, it turns computationally expensive in some real-world cases. Some research tackles this problem, such as Burnaev and Vovk (2014), Lei (2019), Ndiaye and Takeuchi (2019), Abad et al. (2022).

The use of a unique sample set led to some problems, such as: Computational

[12] "A sequence of random variables is *exchangeable* when its joint distribution is invariant to arbitrary permutations of the variables" (Nathanael L. Ackerman at el., 2017).

cost; Over-fits of the *Machine Learning (ML)* on the fit (train) step; The *Non-Conformity Scores (NCS)* function would equal zero. The *NCS* function aims to return the *prediction residue* (error), and the most naive *NCS* is the difference between the *correct value* and the *predicted value*.

A.2.2.2 Split CP

The *Split CP (SCP)*, also known as *Inductive CP (ICP)* [13] , is a particular case of the *Full CP (FCP)* by splitting the original set into two slices, one for *fitting* (train) and another for *calibration*. The *calibration* step uses a different dataset than the *Machine Learning (ML)* algorithm uses for *training* step. It reduces the computational cost because the *calibration* is executed in a shorter set. It also avoids the *Non-Conformity Scores (NCS)* going to zero because it uses a set not over-fitted by the *ML* algorithm. This usefulness has made the *Split CP* the most widely used *CP* method.

Papadopoulos et al. (2002) defined the *Inductive Confidence Machines (ICM)* replacing the *Transductive Inference*[13] with the *Inductive Inference*[13]. The *ICM* provides a measure of its own accuracy in regression problems based on the *Ridge Regression (RR)* model to deal with large datasets. Vovk, Papadopoulos e Gammerman (2005) proved the validity of conformal sets given *exchangeable*[12] data and a *miscoverage level* $\alpha \in (0, 1)$, considering the *Split CP* as a particular case of the *Full CP*. Shafer e Vovk (2008b) developed a complete theory about *CP* known as *Marginal Coverage CP*. The *Marginal Coverage* means the *CP* is fulfilled in the average case and not in each data sample. V. Fedorova et al. (2013) and U. Johansson et al. (2013) presented a new *Efficiency Criteria (EC)* of a prediction, known as being *probabilistic-based*. They do not have the issues found on the standard *EC*. Lei, Rinaldo e Wasserman (2015) applied the *Split CP* to *outlier detection* and *clusters detection* using several *Conformity Scores*. V. Vovk et al. (2016) presented two *probabilistic EC* for *classification*, having a finite set of labels. They affirmed that the *probabilistic* approach should replace the standard criterion and presented six other *EC*. They optimize ten *EC* when the distribution is previously known. Lei et al. (2018) proved that if the anti-conservativeness of conformal sets, the *Non-Conformity Scores (NCS)*, are almost surely distinct. The prediction set can be upper bounded by $1 - \alpha + \frac{1}{n+1}$, where n is the length of an *exchangeable*[12] data and $\alpha \in (0, 1)$ is the *miscoverage level*. It considers the *Split CP* as a particular case of the *Full CP*. Angelopoulos et

[13]The *Inductive Inference* is similar to regular *Supervised Learning* when it splits the dataset into *train* and *test*, and it *trains* general rules for unseen *test* cases. On the contrary, the *Transductive Inference* used the entire dataset beforehand, and it is a specific *training* for specific *test* cases.

al. (2020) discursed about the sizes of the *train* and *calibration* sets. They said the *coverage distribution* concentrates over $1 - \alpha$ by using more *calibration* data (sample). Oliveira et al. (2022) proved the *Split CP* is also applied to dependent and *Non-Stationary*[14] data, keeping the *probabilistic guarantees*. They also developed a general method, based on *concentration of measure* and *decoupling inequalities* instead of the *Exchangeability*, to analyze the *CP*. The *Non-Stationary* is precisely our case of *asset price prediction*, as well as the *Non-Exchangeability*[12].

A.2.2.3
Cross CP

The *Cross-Validation* is not useful for our case of *asset price prediction*, because it is a particular case of *Exchangeability*[12]. Vovk (2015) introduced the *Cross CP*, as a hybrid of *Inductive CP*[13] and *Cross-Validation*, to avoid the *Inductive CP* problem of losing parts of the dataset during the training. Vovk et al. (2018) compared the computational efficiency of *Split CP* and *Cross CP*, and their advantages and limitations.

A.2.2.4
Jackknife CP

The original technique of *Jackknife* is not new as the existence of review research by Miller and Rupert G. (1974). The combination of *Conformal Prediction (CP)* and the *Jackknife* generates an *CP Interval (CPI)* centered on the 'predicted value by the *Machine Learning (ML)* algorithm. The width of the *CPI* is obtained by the *quantiles* of the *Leave-One-Out Residuals* [15] .

Barber et al. (2021) introduced the *Jackknife+* method that also uses the *quantiles* of the *Leave-One-Out Residuals*, similar to the original *Jackknife*. It achieves a *rigorous coverage guarantee* through the assumption of *exchangeable*[12] *training* samples. It also extends the *Jackknife+* to *k-fold* and discourses the relation with the *Cross CP* proposed by (VOVK, 2015).

A.2.2.5
Conformalized Quantile Regression

Conformalized Quantile Regression (CQR) is a particular case of *Split CP* using *Quantile Regression (QR)* method.

Romano, Patterson e Candes (2019) first introduced the *Conformalized Qualite Regression (CQR)* method that inherited the benefits of both the *CP* and the *QR*. They showed the *Efficiency Criteria (EC)* of the *CQR* method tends

[14]The *Non-Stationarity* occurs when the time-series distribution has some statistics changing over time, such as *mean*, *variance*, and time-wise *covariance*.

[15]The *Leave-One-Out Residuals* is the resulting error in not using the altogether data.

to produce better results than it means shorter intervals. Sesia e Candès (2020) compared two *CQR* methods using simulated and *correct data* (real number set). They discuss the ideal proportion of *training* and *calibration* based on empirical observation, that is, between 70% and 90% for the *training* set. Romano (2022) said the *CQR* is good to generate variable intervals length because it uses the *quantiles* estimation. He affirmed all theorems applied to *Split CP* is valid for *CQR* as it is a particular case of the *Split CP*. He said the procedural difference is that *CQR* uses a specific *Non-Conformity Scores (NCS)* and two base models instead of just one.

A.2.2.6 Non-Conformity Scores

The *Non-Conformity Scores (NCS)* function is an important artifact because it influences the size of the *Conformal Prediction Interval (CPI)* set, and therefore the *Efficiency Criteria (EC)*.

Koenker and Basset (1978) first introduced the classical *Quantile Regression (QR)* method. It estimates *conditional quantiles* through *Quantile Least Squares (QLS)*.

Lei et al. (2018) used the *Regression Residual (RR)* as *NCS* function. It is given simplistically by the spread of the residual $\hat{\epsilon}_{train}(x, y)$ between the *correct value* y and the *fitted value* $\hat{y}(x)$ resulting by the *Regression Model (RM)* conditioned on x.

Lei et al. (2018) also used the *Weighted Regression Residuals (WRR)* as *NCS* function. It considers the *non-constant* residual variance that is very typical to the *asset price* data, which has *Non-Stationarity*[16] characteristics. It provides a *weight* to residual $\hat{\epsilon}_{train}(x, y)$ by dividing it per the *Mean Absolute Deviation (MAD)* $\hat{\mu}_{train}(x, y)$. The $\hat{\epsilon}_{train}(x, y)$ is calculated by the $\hat{\mu}_{train}(x, y)$ itself, conditioned on x. It estimates the *MAD* $|y - \hat{y}(x)|$ and the conditional *mean* separately, and it locally reaches adaptive *CPI*.

Hechtlinger, Póczos e Wasserman (2018) used the *Increasing Sets (IS)* as *NCS* function. They tackle classification problems approaching the *Conformal Prediction Interval (CPI)* based on $\hat{p}(x|y)$ instead of $\hat{p}(y|x)$. They argued it is useful when the *train* dataset will not fully describe the *test* dataset. This is exactly the case with many outlier or adversarial attacks, such as the *asset price* data. Angelopoulos et al. (2020) presented a method to stabilize the *Conformal Prediction Interval (CPI)* by regularization. They also used the *Increasing Sets (IS)* as *NCS* function in the image classification problem.

[16]The *asset return* is the difference between the *current asset price* value and the *previous asset price* value. From an operational point of view, it is the result (profit or loss) between the purchase and sale of the *asset*.

Romano, Patterson e Candes (2019) proposed a new adaptive method *Conformalized Quantile Regression (CQR)* combining the *Split Conformal Prediction (SCP)* and the classical *Quantile Regression (QR)*. They used the *Plug-in Prediction Interval Error (PPIE)* as *NCS* function. As it is adaptive to *Heteroscedasticity*[6] data, it is helpful to the *asset price* data due to the substantial variance over time. Besides the experiments on benchmark regression datasets, they theorized a marginal valid coverage guarantee for the *Conformal Prediction Interval (CPI)* of any *Regression Model (RM)*.

A.2.2.7
Regression Prediction

Dewolf, Baets e Waegeman (2023) analyzed the estimation of *Prediction Interval (PI)* in regression domain over *independent and identically distributed (iid)*[17] benchmark data using four methods such as *Bayesian*, *Ensemble*, *Direct Interval Estimation* and *Conformal Prediction (CP)*.

A.3
Proposal

This appendix aims to provide additional information about this research proposal in case the reader wants it.

A.3.1
Conformal Prediction

From the further details of a toy example, it is possible to verify how easy it is to experiment with the *Conformal Prediction (CP)* method.

Figure A.3 shows the pre-processing code and how it is ordinary and very similar to a pre-processing code for an ordinary prediction model for a regression case.

Figure A.4 shows the code and resulting plot for the *train* and *predict* step, followed by the getting of the *prediction absolute error (PAE)* $\hat{\epsilon}$.

The *train* step is very similar to any ordinary *Regression Model (RM)* when it is fitted on the *train* dataset X_{train}. The *calibration* step begins with a task that is similar to any regular *RM* prediction. It first proceeds the prediction on a different dataset that, in our case, is the *calibration* dataset X_{calib}.

The plot shows the dots as the predicted points $\hat{Y}_{calib}$. The symmetric and inclined line is the ideal position of the predicted points. The line furthest to

[17] The *independent and identically distributed (iid)* is a collection of a *random variables* when for each one of the *random variable*, they are mutually *independent*, and they also have the same and mutual *probability distribution*. Although the *iid* simplifies the statistic, it is not a realistic assumption for some cases, such as *asset price prediction (APP)*.

```python
# Package importing
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import random
```

```python
# Data fetching
X, y = fetch_california_housing(return_X_y=True, as_frame=True)
```

```python
# Set spliting:
#  - Train, Calibration, Test (98.00%, 01.98%, 00.02%)
X_train_and_cal, X_test, y_train_and_cal, y_test = train_test_split(
    X, y, test_size=0.02, random_state=11
)
X_train, X_cal, y_train, y_cal = train_test_split(
  X_train_and_cal, y_train_and_cal,
  test_size=0.01, random_state=11
)
```

Figure A.3: *Traditional pre-processing code (authorship based on (SHAH, 2022) sample).*

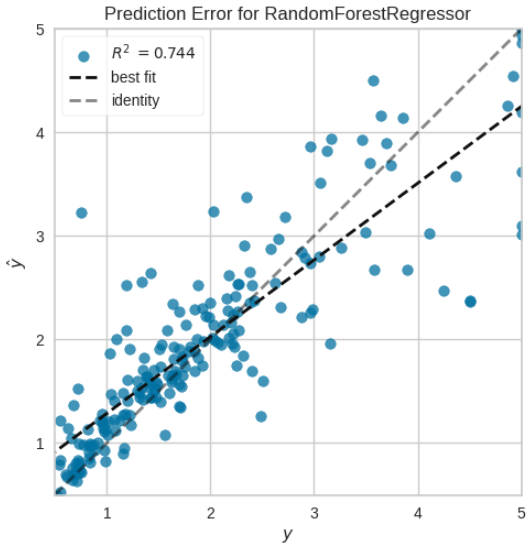
the right is the regression line achieved by the *RM* after the prediction on the *calibration* step.

```python
# Model fitting/training
random.seed(2023)
model = RandomForestRegressor().fit(X_train, y_train)
```

```python
# Predict over the calibration dataset.
y_cal_pred = model.predict(X_cal)
```

```python
# Plot the real values vs predicted values.
from yellowbrick.regressor import PredictionError
visualizer = PredictionError(model)
visualizer.score(X_cal, y_cal)
visualizer.show()
```

(a) Code



(b) Plot

Figure A.4: Prediction result of the calibration step (authorship based on (SHAH, 2022) sample

Figure A.5 shows the code of the *train* and *calibration* steps, and the resulting plot of the *th-quantile* $\mathcal{Q}$ from the distribution of the *prediction absolute error (PAE)* $\hat{\epsilon}$, using the *calibrate* dataset $\hat{Y}_{calib}$.

In the code, we first get the *prediction absolute error (PAE)* $\hat{\epsilon}$ between the *correct label value* y_t and the *predicted label value* $\hat{y}_t$. Second, based on an α that

is the *miscoverage level* defined by the user, and the distribution of the *PAE* $\hat{\epsilon}$, we get the value of the *th-quantile* $1-\alpha$. In this toy experiment, we used an empirical $\alpha = 5$, which consequently generated a *th-quantile* $1-\alpha$ equal to 0.95. It results in a *quantile value* of $1-\alpha$ equal to 1.221569 to be used as a threshold of the *prediction absolute error (PAE)* $\hat{\epsilon}$.

The plot shows the histogram of the distribution of the *PAE* $\hat{\epsilon}$ and a vertical red line as the threshold $1-\alpha$ on *PAE* equal to $\hat{\epsilon} = 1.221569$. In this naive experiment, the practical choice of the α was based on a visual analysis of the histogram plot using the *Elbow Method*. In our case, the right side of the vertical threshold red line has shorter errors, reduced slope variation, and fewer events (mass concentration) in each class with a thin tail.

```
# Calculate the absolute errors.
y_cal_error = np.abs(y_cal - y_cal_pred)
```

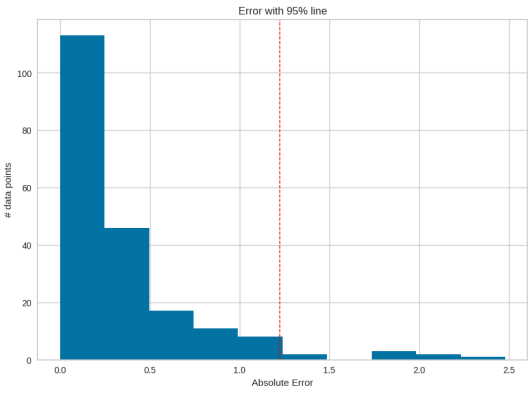
```
# Plot the histogram of the absolute errors.
fig, ax = plt.subplots(figsize =(10, 7))
ax.hist(y_cal_error)
plt.xlabel("Absolute Error")
plt.ylabel("# data points")
plt.title("Error distribution")
plt.show()
```

```
# Calculate the error percentil.
#  - Get the "1-alpha" percentil from
#    the distribution of the absolute errors.
quantile = y_cal_error.quantile(
    q=.95, interpolation='higher'
)
quantile
```

```
1.2215699999999994
```

```
# Plot the histogram with the percentil.
fig, ax = plt.subplots(figsize =(10, 7))
ax.hist(y_cal_error)
plt.axvline(
    quantile, color='red',
    linestyle='dashed', linewidth=1
)
plt.xlabel("Absolute Error")
plt.ylabel("# data points")
plt.title("Error with 95% line")
plt.show()
```

(a) Code



(b) Plot

Figure A.5: Result of the *prediction absolute error (PAE)* and the quantile value threshold (authorship based on (SHAH, 2022) sample).

Figure A.6 shows the code and the resulting table to get the *Conformal Prediction Interval (CPI)* $\mathcal{C}$ based on the *predicted value* $\hat{Y}_{calib}$ on the *calibration* step, and the *quantile value bounds* $\mathcal{Q}$.

In the code, we first get the *predicted value* $\hat{Y}_{calib}$ through a pre-trained *Regression Model (RM)* $\hat{\mathcal{A}}$. Second, we calculate the *CPI*'s bounds $\mathcal{Q}$ based on

the *predicted value* $\hat{Y}_{calib}$ and the *quantile value* of $1-\alpha$ equal to 1.221569. The lower bound is q_{lower}, and the upper bound is q_{upper}.

The table shows the values for the *correct label value (CLV)* Y_{subset}, *predicted label value (PLV)* $\hat{Y}_{subset}$, *lower bound* q_{lower}, and *upper bound* q_{upper}.

```
# Prediction over the test dataset.
y_test_pred = model.predict(X_test)
```

```
# Calculate the interval bounds:
#  Lower Bound = Pred - PredPercentil
#  Upper Bound = Pred + PredPercentil

y_test_interval_pred_left = y_test_pred - quantile
y_test_interval_pred_right = y_test_pred + quantile
```

```
# Add bounds columns to the dataframe.
df = pd.DataFrame(
    list(zip(
        y_test,y_test_pred,
        y_test_interval_pred_left,
        y_test_interval_pred_right
    )),
    columns=[
        'actual','predicted',
        'lower_interval',
        'upper_interval'
    ]
)
df
```

(a) Code

	actual	predicted	lower_interval	upper_interval
0	0.762	0.771250	-0.450320	1.992820
1	1.732	2.365341	1.143771	3.586911
2	1.125	2.227690	1.006120	3.449260
3	1.370	1.818450	0.596880	3.040020
4	1.856	2.229540	1.007970	3.451110
...	...	...	...	...
408	1.073	1.259780	0.038210	2.481350
409	0.517	0.976430	-0.245140	2.198000
410	2.316	1.910720	0.689150	3.132290
411	0.738	0.962040	-0.259530	2.183610
412	2.639	2.805830	1.584260	4.027400

413 rows × 4 columns

(b) Table

Figure A.6: Code and resulting table of the prediction values and the CPI's bounds values (authorship based on (SHAH, 2022) sample).

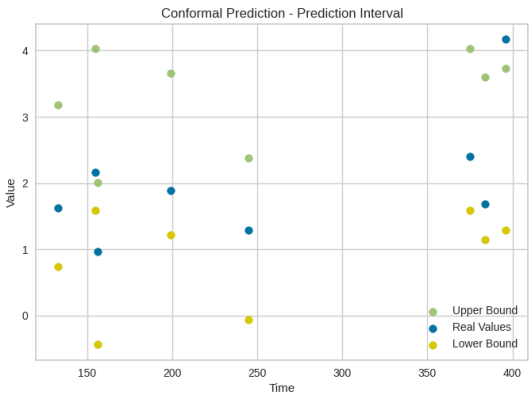
Figure A.7 shows the code and the resulting plot to get the *predicted label values (PLV)* $\hat{Y}_{test}$ and the *Conformal Prediction Interval (CPI)* $\mathcal{C}$, using the *test* dataset X_{test}.

In the code, we used the pre-trained *Regression Model (RM)* $\hat{\mathcal{A}}$ to proceed with the *test* step, known as the prediction, using the *test* dataset X_{test}. Second, we calculate the *CPI*'s bounds $\mathcal{C}$ based on the *predicted label values (PLV)* $\hat{Y}_{test}$ and the *quantile value* of $1-\alpha$ equal to 1.221569. This *quantile value* of $1-\alpha$ equal to 1.221569 is the same obtained on the *calibrate* step. The lower bound is q_{lower}, and the upper bound is q_{upper}.

The scatter plot shows the *predicted values* among the *time-line*. The colors of the values are yellow for the *lower bound*, blue for the *correct*, and green for the *upper bound*.

```
# Plot a small sample of the real values
# and intervals bounds.
plt.scatter(
    x=df1.index,
    y=df1['upper_interval'],
    label="Upper Bound", color='g'
)
plt.scatter(
    x=df1.index,
    y=df1['actual'],
    label="Real Values", color='b'
)
plt.scatter(
    x=df1.index,
    y=df1['lower_interval'],
    label="Lower Bound", color='y'
)
plt.title(
    "Conformal Prediction - Prediction Interval"
)
plt.xlabel('Time')
plt.ylabel('Value')
plt.legend()
plt.show()
```

(a) Code



(b) Table

Figure A.7: Code and resulting scatter plot of the prediction values and the CPI's bounds values (authorship based on (SHAH, 2022) sample).

A.4 Additional Information

This appendix aims to provide some remaining information of this research in case the reader wants it.

A.4.1 Infra-structure

All the experiments were developed in Python Notebook, running in Ubuntu Linux, over Windows, through Windows Subsystem for Linux 2 (WSL2), with 16.0 Gb of RAM, as the following details.

Hardware:

- **Operational System**
  Microsoft Windows 10 Pro, version 10.0.19044 Build 19044.
- **Laptop**
  DELL, model Latitude E6430, processor Intel(R) Core(TM) i5-3340M CPU @ 2.70GHz, 2701 Mhz, 2 Core(s), 4 Logical Processor(s), memory of 15.9 Gb, virtual memory of 23.4 Gb, disk NTFS 291.89 Gb for the operational system, disk NTFS 638.05 Gb for the data.

- **Virtual Machine**
  wsl2.
- **Virtual Operational System**
  Ubuntu 20.04.6 LTS, release 20.04, code-name focal, Memory 12697 Kb, Swap 4096 Kb, Disk 256 Gb, 4 CPUs type Intel(R) Core(TM) i5-3340M CPU @ 2.70GHz.

Software:

- **Integrated Development Environment (IDE)**
  Microsoft VS Code, version 1.80.1 - x64.
- **Programming Language**
  Python, version 3.9.7.
- **Notebook**
  Jupyter, version: IPython=7.29.0, ipykernel=6.4.1, jupyter_client=6.1.12, jupyter_core=4.8.1, jupyter_server=1.4.1, notebook=6.4.5.

Packages:

- **requirement.txt**
  requirements.txt
- **requirement.yaml**
  requirements.yaml