University of Essex

School of Mathematics, Statistics
and Actuarial Science

MA981 DISSERTATION

# Quantitative Forecasting of High Yield Bonds using Data Science, Deep Learning, and AI

**Yao Bathaix Philippe-Emmanuel**

Supervisor: **Yassir Rabhi**

September 15, 2024
Colchester

# Contents


0.1 Abstract . . . . . 4

**1 Introduction** **5**

**2 Comprehensive Review of Bond Forecasting Method** **9**

2.1 Existing Methods for Bond Forecasting . . . . . 9

2.1.1 Traditional Statistical Models . . . . . 10

2.1.2 Relevant Machine Learning and Deep Learning models . . . . . 12

2.2 Introduction to Machine Learning and Deep Learning Approaches . . . 14

2.2.1 Machine Learning . . . . . 14

2.2.2 Deep Learning . . . . . 18

2.2.3 Differences Between Machine Learning and Deep Learning . . . 23

**3 Methodology** **26**

3.1 Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) . . 26

3.1.1 Theoretical Background . . . . . 27

3.1.2 LSTM Networks . . . . . 29

3.1.3 GRU Networks . . . . . 31

3.2 Gradient Boosting Regressor (GBR) . . . . . 33

3.2.1 Theoretical Background . . . . . 34

3.2.2 Gradient Boosting Algorithm . . . . . 36

3.3 Stacked Generalisation (Stacking) . . . . . 38

3.3.1 Stacked Generalisation Architecture . . . . . 39

3.3.2 Stacked Generalisation Algorithm . . . . . 40

3.4 Evaluation Metrics . . . . . 43

**4 Data Analysis** **47**

4.1 Data Collection . . . . . 47

4.2 Data Description . . . . . 49

4.3 Data Preprocessing and Feature Engineering . . . . . 50

4.3.1 Handling Missing Values . . . . . 51

4.3.2 Date Conversion and Time-Based Features . . . . . 51

4.3.3 Moving Averages and Lag Features . . . . . 51

4.3.4 Daily Return . . . . . 52

4.3.5 Relative Strength Index (RSI) . . . . . 52

4.3.6 Moving Average Convergence Divergence (MACD) . . . . . 52

4.3.7 Bollinger Bands . . . . . 53

4.4 Exploratory Data Analysis (EDA) . . . . . 53

4.4.1 Summary Statistics . . . . . 53

4.4.2 Time Series Analysis . . . . . 55

4.4.3 Correlation Analysis . . . . . 58

4.4.4 Distribution Analysis . . . . . 60

4.4.5 Seasonal Decomposition of Close Price . . . . . 61

4.4.6 Macroeconomic Indicators Analysis . . . . . 62

4.5 Model Implementation and Performance Analysis . . . . . 65

4.5.1 Model Implentation . . . . . 65

4.5.2 Performance Analysis and Interpretation . . . . . 68

**5 Conclusion** **72**

## 0.1 Abstract

Forecasting high-yield bonds is crucial for investors and financial institutions, as it directly influences risk management strategies and investment decisions. Traditional econometric models often fail to capture the complex and volatile dynamics of the high-yield bond market, leading to suboptimal forecasts. This dissertation, titled *"Quantitative Forecasting of High-Yield Bonds using Data Science, Deep Learning, and AI,"* explores the application of advanced machine learning (ML) and deep learning (DL) techniques to improve the prediction accuracy of bond yields and prices.

The research utilises a comprehensive dataset consisting of historical bond prices, volume, bond-specific attributes, and macroeconomic indicators such as Gross Domestic Product (GDP), the federal funds rate, Consumer Price Index (CPI), and the unemployment rate. By implementing Gradient Boosting Regressor (GBR) models alongside Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks, this study captures the temporal dependencies and nonlinear relationships inherent in bond price movements.

Empirical analysis reveals that these advanced models significantly outperform traditional econometric approaches, particularly in scenarios of high volatility and credit risk. Key metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared ($R^2$) indicate higher prediction accuracy, reducing forecast error. This improvement is especially notable in the high-yield bond market, where accurate predictions are critical for enhancing portfolio performance and mitigating risk.

This dissertation not only provides a comparison of ML and DL models but also offers practical insights for financial institutions looking to leverage AI-driven strategies for bond market forecasting. Future research directions are discussed, focusing on integrating additional financial indicators and refining AI models to further enhance predictive power.

CHAPTER 1

# Introduction

The bond market is a cornerstone of the global financial system, playing a vital role in the economy by allowing governments, municipalities, and corporations to raise capital. Bonds are fixed-income securities that represent a loan made by an investor to a borrower, typically corporate or governmental. They are issued with defined terms, including the lifespan (maturity), interest (coupon) rate, and face value, which must be repaid at maturity. Bonds are a critical component of diversified investment portfolios due to their relatively stable returns and lower risk compared to equities.

The bond market, also known as the debt market, encompasses a variety of instruments, including government bonds, municipal bonds, and corporate bonds. Government bonds, such as U.S. Treasuries, are typically considered low-risk investments, as they are backed by the full faith and credit of the issuing government. Municipal bonds are issued by local governments or their agencies and often come with tax advantages. Corporate bonds are issued by companies to finance their operations, expansions, or other financial activities. As of July 2023, the global bond market was estimated to be over $135 trillion [1].

Bonds are crucial in finance for several reasons:

- **Capital Raising:** They provide a vital source of funding for governments and corporations, enabling infrastructure projects, business expansion, and day-to-day operations.
- **Investment Diversification:** Bonds offer a means of portfolio diversification,

reducing overall risk due to their generally stable and predictable income streams.

- **Economic Indicators:** Bond yields and prices are often used as indicators of economic health, influencing monetary policy decisions by central banks.

- **Risk Management:** They help in managing interest rate risk and credit risk within financial markets.

High yield bonds, often referred to as "junk bonds," are corporate bonds that have lower credit ratings than investment-grade bonds. They offer higher yields to compensate for the higher risk of default. The high yield bond market has become an essential avenue for companies, particularly those with lower credit ratings, to access capital. Despite their higher risk, these bonds are attractive to investors seeking higher returns.

High yield bonds are characterised by their higher yields, which compensate for the increased risk; their credit ratings, which are typically below BBB- by Standard and Poor's or Baa3 by Moody's; and their risk factors, which include greater default risk due to the issuing company's lower creditworthiness, economic cycles, and industry-specific factors. The high yield bond market is relatively young compared to the broader bond market. It has grown substantially since the late 20th century, driven by periods of economic expansion, regulatory changes, and increased investor appetite for higher yields [2]. The issuance of high yield bonds surged following the financial crisis of 2008, as companies sought alternative financing sources due to constrained bank lending [3]

As of 2023, the global high-yield bonds market size reached approximately $5.31 trillion. The market is projected to grow, reaching a value of around $7.76 trillion by 2032[4]. In recent years, the market has experienced significant growth, particularly in the United States and Europe. In the current market conditions, high yield bonds are facing challenges due to rising interest rates and economic uncertainty. The Federal Reserve's tightening monetary policy has led to increased borrowing costs, impacting the yields and prices of high yield bonds. Additionally, concerns about economic slowdowns and potential recessions have heightened the risk perception among investors, leading to wider spreads between high yield bonds and safer government securities.

High-yield bonds are particularly challenging to forecast compared to other bonds due to their higher volatility, sensitivity to economic cycles, and greater susceptibility to default risks. Their prices are influenced by a multitude of factors including

issuer-specific credit risk, broader economic conditions, and market sentiment. This complexity requires advanced modeling techniques to accurately capture and predict their movements.

Accurately predicting the prices and yields of high yield bonds is crucial for investors and financial institutions to manage risk and optimise returns. Traditional forecasting models, such as econometric and statistical approaches, have been used extensively but often fall short in capturing the complexities of financial markets. The advent of data science, machine learning (ML), deep learning (DL), and artificial intelligence (AI) presents new opportunities to enhance these models' accuracy and reliability.

This dissertation aims to explore the potential of ML, DL, and AI in improving the quantitative forecasting of high yield bonds. By leveraging data science techniques and integrating both numerical and textual data, this research seeks to develop robust models that offer significant improvements over traditional methods. The primary objectives of this dissertation are to evaluate the effectiveness of various ML and DL models in predicting high yield bond prices, to assess the impact of integrating various sources on the accuracy of these predictions, to provide a comparative analysis of different predictive models, highlighting their strengths and weaknesses, and to offer practical insights and recommendations for financial practitioners on the application of these advanced models in the bond market.

Several key documents form the foundation of this dissertation, each providing essential insights and methodologies that will guide the research. "Machine Learning Algorithms for Financial Asset Price Forecasting" by Philip Ndikum [5] highlights the superiority of ML models over traditional models like the Capital Asset Pricing Model (CAPM). Ndikum's empirical analysis demonstrates the significant performance improvements achievable with ML algorithms, making it a critical reference for evaluating the effectiveness of ML models in bond price forecasting. Additionally, "Machine Learning-aided Modeling of Fixed Income Instruments" by Daniel Martin et al. [6] demonstrates the applicability of various ML models, including Random Forest and Support Vector Regression, in forecasting bond prices. Their comprehensive experiments with U.S. Treasuries and corporate bonds provide a valuable reference for selecting and implementing models in this research, highlighting the practical applications of ML in fixed income markets.

By building on these foundational works, this dissertation aims to push the boundaries of high yield bond forecasting, leveraging the power of data science, machine learning, deep learning, and artificial intelligence to deliver more accurate and actionable insights for investors and financial institutions. This work contributes to the existing body of knowledge by demonstrating the effectiveness of advanced predictive models, integrating diverse data sources, and providing practical recommendations for improving high-yield bond forecasting.

CHAPTER 2

# Comprehensive Review of Bond Forecasting Method

In this overview, we will explore various bond forecasting methods, emphasising the advancements in Machine Learning (ML) and Deep Learning (DL) techniques. We will begin by delving into traditional statistical models, including time-series analysis, regression models, and factor models, examining their strengths and limitations. Following this, we will review significant studies that have utilised ML and DL for bond forecasting, highlighting key research contributions and identifying areas for further investigation. Finally, we will provide a comprehensive introduction to ML and DL, discussing their core principles and methodologies, and outlining the differences between them.

## 2.1 Existing Methods for Bond Forecasting

Bond forecasting has traditionally relied on econometric and statistical models that leverage historical data and various financial indicators to predict bond prices and yields. These methods include time-series analysis, regression models, and factor models. Today, there are several new methods that are more accurate than the previous ones.

### 2.1.1 Traditional Statistical Models

#### A - Time-Series Analysis

Time-series models are used to analyse and forecast future values based on past trends and patterns. One of the most commonly used time-series models is ARIMA (AutoRegressive Integrated Moving Average).The ARIMA model combines three components:

- Autoregressive (AR): The AR component specifies that the output variable depends linearly on its own previous values. The AR part of the model can be expressed as:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + \epsilon_t \tag{2.1}$$

  where $y_t$ is the value at time $t$, $\phi_i$ are the coefficients, and $\epsilon_t$ is white noise.

- Integrated (I): The I component represents the differencing of raw observations to make the time series stationary. If the data needs to be differenced $d$ times to achieve stationarity, this is indicated by $I(d)$.

- Movering Average (MA): The MA component models the dependency between an observation and a residual error from a moving average model applied to lagged observations. The MA part of the model can be expressed as:

$$y_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q} \tag{2.2}$$

  where $\theta_i$ are the coefficients.

- ARIMA Model: Combining these components, the general ARIMA model can be written as:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q} \tag{2.3}$$

#### B - Regression Models

Regression models predict bond prices or default probabilities by examining relationships between bond characteristics and financial indicators.

- Multiple Linear Regression: Multiple linear regression models the relationship between a dependent variable $y$ and multiple independent variables

$X_1, X_2, \ldots, X_n$. The model can be expressed as:

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \epsilon \tag{2.4}$$

where $\beta_i$ are the coefficients, and $\epsilon$ is the error term.

– Logistic Regression: Logistic regression is used for binary classification problems, such as predicting default probabilities. The model predicts the probability that a given input belongs to a particular class. The logistic function (sigmoid) is used to model the probability:

$$P(y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n)}} \tag{2.5}$$

**C - Factor Models**

Factor models, such as the Capital Asset Pricing Model (CAPM) and the Arbitrage Pricing Theory (APT), identify factors that influence bond returns and incorporate them into predictive models.

– Capital Asset Pricing Model (CAPM): CAPM is used to determine the expected return on an investment based on its systematic risk (beta). The formula is:

$$E(R_i) = R_f + \beta_i (E(R_m) - R_f) \tag{2.6}$$

where $E(R_i)$ is the expected return on the investment, $R_f$ is the risk-free rate, $\beta_i$ is the beta of the investment, and $E(R_m)$ is the expected return of the market.

– Arbitrage Pricing Theory (APT): APT is a multi-factor model that considers various macroeconomic factors affecting asset returns. The model can be expressed as:

$$R_i = \alpha_i + \beta_{i1} F_1 + \beta_{i2} F_2 + \ldots + \beta_{ik} F_k + \epsilon_i \tag{2.7}$$

where $R_i$ is the return on asset $i$, $\alpha_i$ is the intercept, $\beta_{ij}$ are the factor sensitivities, $F_j$ are the factors, and $\epsilon_i$ is the error term.

Traditional statistical models offer several strengths but also come with significant limitations. These models are well-established and well-documented, providing a solid

theoretical foundation. They are relatively simple to implement and interpret, making them accessible to practitioners and stakeholders. The results from these models are easy to communicate and understand, which is valuable for decision-making processes. Additionally, they can effectively capture linear relationships and trends in financial data, making them useful for short-term predictions and straightforward scenarios. However, these models often struggle to capture non-linear relationships and complex interactions between variables, which are common in financial markets. They are sensitive to outliers and may require extensive data preprocessing to achieve accurate results. Traditional models may not perform well when dealing with high-dimensional data, where the number of variables is large. Many time-series models assume that the underlying data distribution does not change over time, which may not hold true in dynamic financial markets. Moreover, traditional models often lack the flexibility to adapt to new and changing data patterns, which can limit their long-term predictive power. By understanding both the strengths and limitations of traditional statistical models, we can better appreciate the potential benefits of adopting machine learning and deep learning approaches. These advanced methodologies can address many of the shortcomings of traditional models, offering enhanced accuracy and robustness in bond price forecasting. In the next section, we will explore how ML and DL can be leveraged to improve bond forecasting models.

### 2.1.2 Relevant Machine Learning and Deep Learning models

Numerous studies have demonstrated the effectiveness of machine learning (ML) and deep learning (DL) techniques in improving financial forecasting, particularly in the context of bond markets. This section reviews key studies that have applied ML and DL to bond forecasting, highlighting the proposed models and discussing their usefulness for our quantitative forecasting of bond prices.

Philip Ndikum's work in "Machine Learning Algorithms for Financial Asset Price Forecasting" highlights the superiority of ML models over traditional models like the Capital Asset Pricing Model (CAPM) [5]. Ndikum's empirical analysis shows that ML algorithms, such as random forests and gradient boosting machines, significantly outperform traditional approaches in predicting financial asset prices. Random forests, an ensemble learning method that constructs multiple decision trees during training

and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees, and gradient boosting machines (GBM), which build models in a stage-wise fashion and generalise them by allowing optimisation of an arbitrary differentiable loss function, are particularly effective. These models capture complex, non-linear relationships in financial data, crucial for accurately predicting bond prices. Ensemble methods like random forests and GBM improve predictive accuracy by combining the strengths of multiple models, reducing overfitting, and enhancing robustness. Furthermore, their ability to handle large datasets efficiently makes them suitable for analysing extensive historical bond data.

Another pivotal study is "Machine Learning-aided Modeling of Fixed Income Instruments" by Daniel Martin et al. [6]. This paper provides a comprehensive analysis of applying various ML models, including random forests, support vector regression (SVR), and ensemble methods, to forecast bond prices. SVR, a type of Support Vector Machine (SVM) that supports linear and non-linear regression by finding the best fit line within a predefined margin of tolerance, is particularly effective for high-dimensional data and can capture non-linear relationships, making it well-suited for bond price forecasting. Ensemble methods further enhance model performance by mitigating the weaknesses of individual models and improving generalisation. The study also emphasises the importance of feature selection and data preprocessing, which are critical steps for developing robust forecasting models.

In "Artificial Neural Networks in Fixed Income Markets for Yield Curve Forecasting," Manuel Nunes and colleagues[7] explore the application of artificial neural networks (ANNs) for yield curve forecasting in fixed income markets. Their study compares the performance of ANNs with traditional models like multivariate linear regression, finding that ANNs, particularly those using relevant features and synthetic data, provide superior forecasting accuracy. ANNs are computational models inspired by the human brain, consisting of interconnected nodes (neurons) that process data in multiple layers (input, hidden, output). Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN) capable of learning long-term dependencies, are particularly effective for time-series forecasting due to their ability to learn from sequential patterns and long-term dependencies. The use of synthetic data to augment training datasets can improve model robustness and predictive performance in scenarios with limited

historical data.

These studies collectively demonstrate the significant potential of ML and DL techniques in improving bond forecasting. By addressing many limitations of traditional statistical models and offering enhanced accuracy and the ability to handle complex, high-dimensional data, these advanced methods provide a robust framework for developing quantitative forecasting models for bond prices.

The key benefits for our forecasting model include enhanced predictive accuracy, as ML and DL techniques can capture complex patterns in the data. They are well-suited for analysing extensive historical bond data, making them ideal for our quantitative forecasting needs. Additionally, ensemble methods and neural networks offer robustness against overfitting and adaptability to new data, enhancing the reliability of forecasts. Finally, ML and DL models can automate feature extraction and selection processes, reducing the need for manual intervention and improving efficiency. By incorporating insights and methodologies from these studies, we can develop a more effective and reliable bond price forecasting model that leverages the strengths of ML and DL techniques.

## 2.2 Introduction to Machine Learning and Deep Learning Approaches

Machine learning (ML) and Deep Learning (DL) are subsets of artificial intelligence (AI) that have revolutionised numerous fields by providing powerful tools for data analysis and prediction. These approaches enable the processing of large datasets, identification of complex patterns, and improvement of predictive accuracy across various domains.

### 2.2.1 Machine Learning

Machine learning (ML) is focused on developing algorithms that can learn from and make predictions based on data. ML algorithms build models based on sample data, known as "training data," to make decisions or predictions without being explicitly programmed to perform the task. The key concepts in Machine Learning are:

### A - Supervised Learning

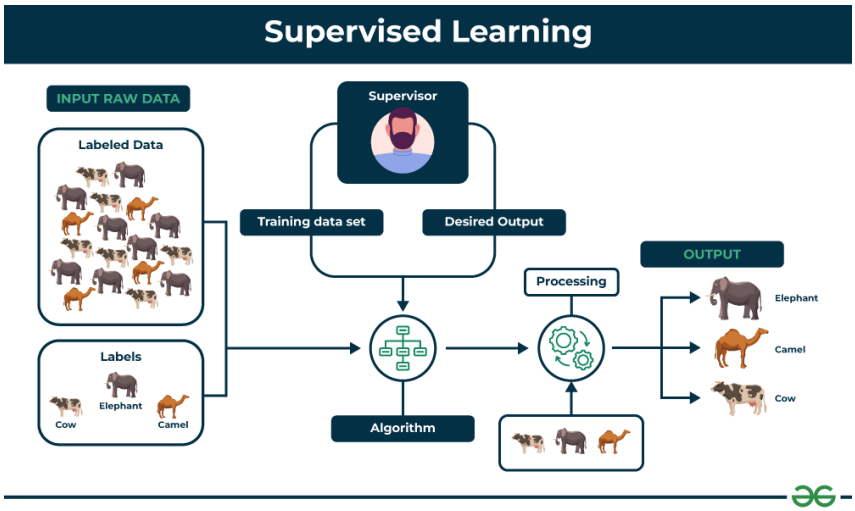



Figure 2.1: Diagram showing the process of supervised learning. Source: GeeksforGeeks (2024)

Supervised learning is a framework that involves learning from labeled data to make predictions or classifications. As illustrated in Figure 2.1, this approach trains a model using a dataset where the input-output pairs are known. The algorithm learns to map the input data to the correct output by identifying patterns in the labeled examples. The common algorithms used in the supervised learning are:

- Linear Regression: Predicts a continuous output based on the linear relationship between input features.
- Logistic Regression: Used for binary classification problems.
- Decision Trees: Splits the data into subsets based on the value of input features, creating a tree-like model.
- Random Forests: An ensemble method that uses multiple decision trees to improve prediction accuracy.
- Support Vector Machines (SVM): Finds the hyperplane that best separates classes in the feature space.
- Neural Networks: Composed of layers of neurons that can learn complex patterns in data.

**B - Unsupervised Learning**

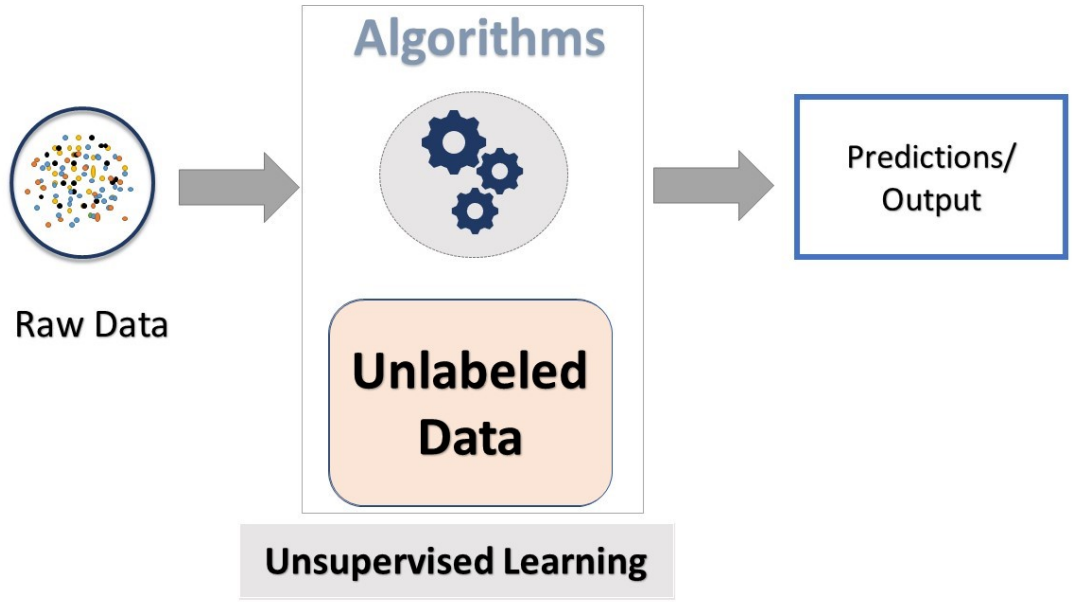



Figure 2.2: Diagram showing the process of unsupervised learning. Source: EduShot (2024)

Unsupervised learning is a framework that involves learning patterns from input data without labeled outputs. As illustrated in Figure 2.4, the model is trained on unlabeled data, and the algorithm seeks to find patterns or intrinsic structures within the input data.

The common methods used in unsupervised learning include:

- K-Means Clustering: Partitions the data into k distinct clusters based on feature similarity.
- Hierarchical Clustering: Builds a hierarchy of clusters by either merging or splitting existing clusters.
- Principal Component Analysis (PCA): Reduces the dimensionality of data by transforming it into a new set of orthogonal components.
- Autoencoders: Neural networks used for dimensionality reduction and feature learning.

### C - Reinforcement Learning

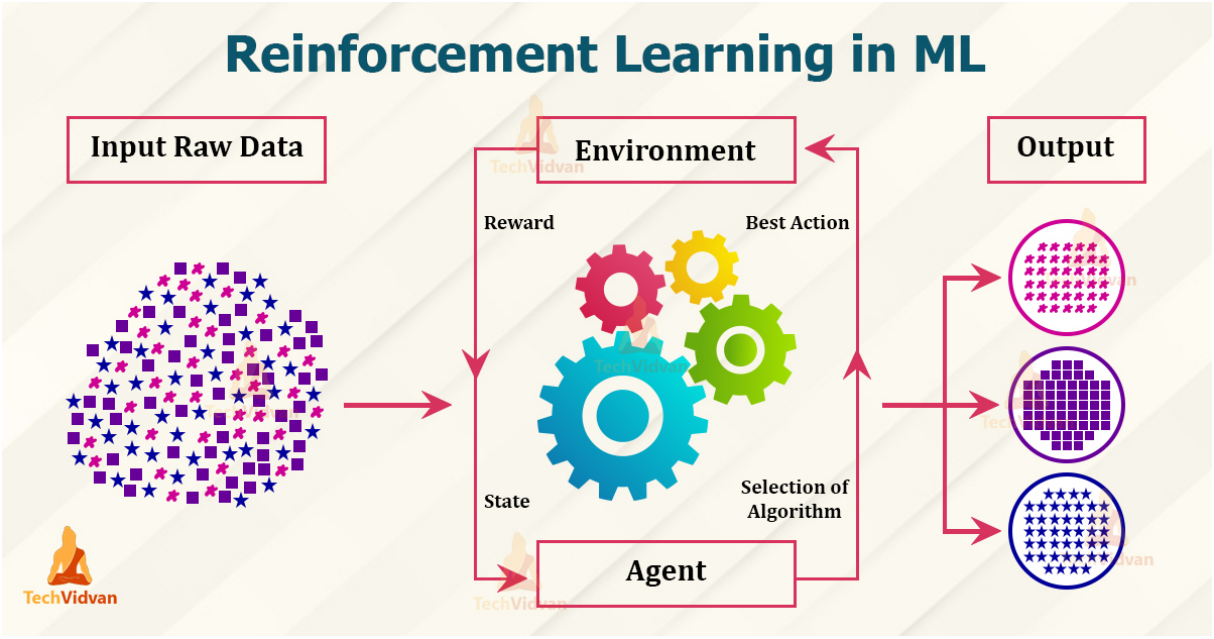



Figure 2.3: Diagram showing the process of reinforcement learning. Source: TechVidvan (2024)

Reinforcement learning is a framework that involves an agent interacting with an environment to learn optimal behaviors through trial and error. As illustrated in Figure 2.3, the agent uses feedback from its actions and experiences to improve over time, aiming to maximise cumulative rewards.

The common algorithms used in reinforcement learning include:

- Q-Learning: A value-based method that learns the value of an action in a particular state.
- Deep Q-Networks (DQN): Combines Q-learning with deep neural networks to handle high-dimensional state spaces.
- Policy Gradient Methods: Directly optimise the policy (action strategy) that the agent follows.
- Actor-Critic Methods: Combines value-based and policy-based methods to reduce variance and improve learning stability.

### 2.2.2 Deep Learning

Deep learning (DL) is a specialised subset of machine learning that uses neural networks with many layers (hence "deep") to model complex patterns in large datasets. DL algorithms are particularly effective at learning from large amounts of unstructured data, such as images, text, and time series. The key concepts in Deep Learning are:

**A - Neural Networks**

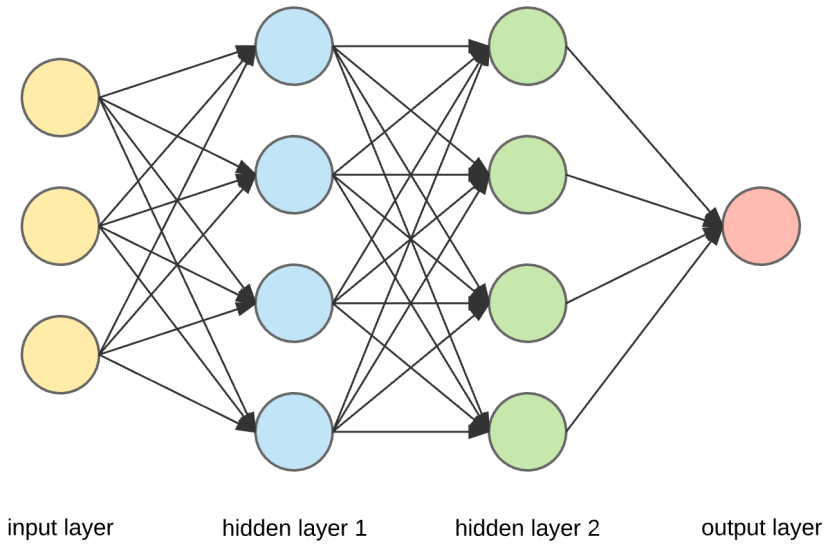



Figure 2.4: Simple Neural Network Architecture. Source: roboflow (2023)

Neural Networks are composed of layers of interconnected nodes, or neurons, that process input data to generate an output. As illustrated in Figure 2.4, a simple neural network consists of an input layer, two hidden layers, and an output layer. Each connection has a weight that adjusts during training to minimise error. The different types of Neural Networks are :

- Feedforward Neural Networks (FNN): The simplest type of neural network where connections do not form cycles.
- Convolutional Neural Networks (CNNs): Primarily used for image recognition and processing, convolutional layers act like filters that scan through input images to detect and extract important features, such as edges, textures, and patterns.

- Recurrent Neural Networks (RNNs): Designed for sequential data, such as time series. They have loops that allow information to persist, making them suitable for tasks where context is important.
  - Long Short-Term Memory (LSTM): A type of RNN that can capture long-term dependencies in sequential data.
  - Gated Recurrent Unit (GRU): A simpler variant of LSTM that can also capture dependencies in sequences.

**B - Training Neural Networks**

Training neural networks involves adjusting the weights of the connections between neurons to minimise the error in the network's predictions. Two crucial components of this process are backpropagation and optimisation algorithms.

**C - Backpropagation**

Backpropagation is a method used to calculate the gradient of the loss function with respect to each weight in the network. This process is essential for training neural networks as it allows for the adjustment of weights to minimize the loss. The steps involved in backpropagation are:

- Forward Pass: Compute the output of the network by passing the input data through each layer.
- Compute Loss: Calculate the loss (error) between the predicted output and the actual target values using a loss function (e.g., mean squared error for regression, cross-entropy for classification).
- Backward Pass: Propagate the error backward through the network to calculate the gradient of the loss with respect to each weight using the chain rule.
- Weight Update: Update the weights using the calculated gradients and an optimisation algorithm.

The goal of backpropagation is to minimise the loss function by iteratively adjusting the weights of the network.

**D - Optimisation Algorithms**

Optimisation algorithms are used to update the weights of the network based on the gradients calculated during backpropagation. Some common optimisation algorithms include:

- Stochastic Gradient Descent (SGD): Updates the weights using the gradient of the loss function with respect to the weights, computed for a single training example or a mini-batch. The update rule for SGD is:

$$w = w - \eta \nabla L(w) \tag{2.8}$$

  where $w$ are the weights, $\eta$ is the learning rate, and $\nabla L(w)$ is the gradient of the loss function.

- Mini-Batch Gradient Descent: A variation of SGD where the gradient is computed over small batches of training data. This provides a balance between the computational efficiency of SGD and the stability of Batch Gradient Descent.

- Momentum: Enhances SGD by adding a fraction of the update vector of the past time step to the current update vector, helping to accelerate convergence and reduce oscillations. The update rule with momentum is:

$$v = \gamma v + \eta \nabla L(w) \tag{2.9}$$

$$w = w - v \tag{2.10}$$

  where $\gamma$ is the momentum coefficient.

- AdaGrad (Adaptive Gradient Algorithm): Adapts the learning rate for each parameter based on the magnitude of past gradients, allowing for larger updates for infrequent parameters. The update rule for AdaGrad is:

$$w = w - \frac{\eta}{\sqrt{G_{t,ii}} + \epsilon} \nabla L(w) \tag{2.11}$$

  where $G_t$ is the diagonal matrix of accumulated gradients, and $\epsilon$ is a small constant to prevent division by zero.

– RMSprop (Root Mean Square Propagation): Aims to resolve the diminishing learning rate problem of AdaGrad by using a moving average of squared gradients to normalise the gradient. The update rule for RMSprop is:

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2 \tag{2.12}$$

$$w = w - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \nabla L(w) \tag{2.13}$$

where $\gamma$ is the decay rate.

– Adam (Adaptive Moment Estimation): Combines the advantages of both AdaGrad and RMSprop by computing adaptive learning rates for each parameter. The update rule for Adam is:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2.14}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{2.15}$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{2.16}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{2.17}$$

$$w = w - \frac{\eta \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{2.18}$$

where $m_t$ and $v_t$ are the estimates of the first and second moments of the gradients, $\beta_1$ and $\beta_2$ are decay rates, and $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected estimates.

**E - Activation Functions**

Activation functions introduce non-linearity into neural networks, allowing them to learn complex patterns in the data. Some of the most commonly used activation functions are:

– Sigmoid Function: The sigmoid function outputs a value between 0 and 1, making it useful for binary classification tasks. The formula for the sigmoid function is:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.19}$$

– Hyperbolic Tangent (Tanh) Function: The tanh function outputs values between -1 and 1, and is zero-centered, which can make optimisation easier. The formula for the tanh function is:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.20}$$

– Rectified Linear Unit (ReLU): The ReLU function outputs the input directly if it is positive; otherwise, it outputs zero. It is widely used because it helps mitigate the vanishing gradient problem. The formula for the ReLU function is:

$$\text{ReLU}(x) = \max(0, x) \tag{2.21}$$

– Leaky ReLU: The Leaky ReLU is a variation of the ReLU that allows a small, non-zero gradient when the input is negative, helping to keep the gradient flow active. The formula for the Leaky ReLU function is:

$$\text{Leaky ReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases} \tag{2.22}$$

where $\alpha$ is a small constant (typically 0.01).

– Parametric ReLU (PReLU): The PReLU is an extension of Leaky ReLU where the parameter $\alpha$ is learned during training. The formula for the PReLU function is:

$$\text{PReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases} \tag{2.23}$$

– Exponential Linear Unit (ELU): The ELU function outputs the input directly if it is positive; otherwise, it outputs an exponential function. This helps in reducing the vanishing gradient problem while smoothing the outputs for negative inputs. The formula for the ELU function is:

$$\text{ELU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases} \tag{2.24}$$

where $\alpha$ is a hyperparameter that controls the value to which an ELU saturates for negative net inputs.

– Swish: The Swish function, proposed by Google researchers, is defined as $x$ multiplied by the sigmoid function of $x$. It is smooth and non-monotonic, and has been shown to perform better than ReLU on deeper models. The formula for the Swish function is:

$$\text{Swish}(x) = x \cdot \sigma(x) = \frac{x}{1 + e^{-x}} \tag{2.25}$$

– Softmax Function: The softmax function is used in the output layer of neural networks for multi-class classification problems. It outputs a probability distribution over multiple classes. The formula for the softmax function for the $i$-th output in a vector $z$ is:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{2.26}$$

– GELU (Gaussian Error Linear Unit): The GELU activation function weights inputs by their probability under the Gaussian distribution. It is used in the BERT model and has shown good performance in deep learning tasks. The formula for the GELU function is:

$$\text{GELU}(x) = x \cdot \Phi(x) \tag{2.27}$$

where $\Phi(x)$ is the cumulative distribution function of the Gaussian distribution.

### 2.2.3 Differences Between Machine Learning and Deep Learning

Machine learning (ML) and deep learning (DL) differ in several key aspects, and understanding these differences is crucial for selecting the appropriate approach for a given problem. ML algorithms can work well with relatively smaller datasets and are effective when data is limited but features are well-defined, utilising techniques such as linear regression, decision trees, support vector machines (SVM), and k-nearest neighbors (k-NN). On the other hand, DL algorithms typically require large amounts of data to perform effectively and excel when dealing with massive datasets, especially unstructured data, through techniques involving neural networks with many layers

(deep neural networks), such as convolutional neural networks (CNNs) for images and recurrent neural networks (RNNs) for sequences.

Regarding feature engineering, ML often requires manual feature extraction and selection, necessitating domain expertise to engineer relevant features, for instance, manually extracting edges, textures, and shapes in image classification. Conversely, DL can automatically extract features from raw data through its layered structure, with each layer learning different levels of abstraction from the data, automatically detecting edges, textures, and shapes in image classification through convolutional layers.

In terms of performance, ML algorithms are generally faster to train compared to DL algorithms, though they may not achieve the same level of accuracy on complex tasks as DL algorithms. ML is suitable for problems where interpretability and explainability are important. DL algorithms, while achieving higher accuracy on complex tasks such as image and speech recognition, require more computational power and longer training times due to their complexity. They benefit from hardware accelerators like GPUs, which are good at handling many calculations at once, and TPUs, which are specially designed to speed up deep learning tasks.

When considering complexity, ML models are generally simpler and easier to interpret, with algorithms like decision trees and linear regression providing clear insights into decision-making processes, making them suitable for applications where model interpretability is critical. In contrast, DL models are more complex and challenging to interpret and debug due to the "black box" nature of deep neural networks. However, techniques like attention mechanisms and explainable AI (XAI), which aim to make the decision-making process of these models clearer and easier to understand, are being developed to improve interpretability.

Having explored the fundamental concepts and key differences between machine learning and deep learning, we now turn our attention to how these advanced methodologies can be applied to the specific task of bond price forecasting. Traditional statistical models, while useful in certain contexts, often fall short in capturing the complex, nonlinear relationships present in financial markets. ML and DL techniques, with their ability to process large datasets and uncover intricate patterns, offer significant potential to enhance the accuracy and robustness of bond price predictions. In the following sections, we will delve into traditional statistical methods for bond forecasting, exam-

ining their strengths and limitations, and then explore how ML and DL can address these challenges, ultimately leading to more precise and reliable forecasting models. By leveraging the advanced capabilities of ML and DL, we aim to push the boundaries of what is possible in bond price forecasting, providing a solid foundation for future research and practical applications in financial markets.

CHAPTER 3

# Methodology

This section provides a detailed explanation of the methodology employed to forecast high-yield bonds using Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRU), Gradient Boosting Regressor (GBR) and Support Vector Regressor(SVR). Each subsection includes an introduction to the model, its theoretical background, implementation details, illustrative examples, model architecture, conditions for use in high-yield bond forecasting and essential equations.

## 3.1 Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU)

Long Short-Term Memory Networks (LSTM) and Gated Recurrent Units (GRU) are types of artificial neural networks designed to address the limitations of traditional Recurrent Neural Networks (RNNs), particularly the vanishing gradient problem. Both LSTMs and GRUs are rooted in the fields of computer science, mathematics, and statistics, drawing on key concepts from each discipline to create powerful tools for sequence prediction and time series forecasting.

The LSTM algorithm was introduced by Hochreiter and Schmidhube [8], and the GRU by Kyunghyun Cho et al. [9]. Both aim to solve the difficulties faced by RNNs in learning long-term dependencies, but they do so with different architectural approaches. Traditional RNNs struggled with maintaining information over long sequences due

to the vanishing gradient problem, where gradients used for training the network diminish exponentially, making it difficult to adjust weights effectively for long-range dependencies.

The development of LSTM and GRU networks has had a significant impact on various fields, particularly those involving sequential data. They have been successfully applied in natural language processing, speech recognition, and financial time series forecasting, among other areas. The ability of LSTMs and GRUs to handle sequences and maintain long-term dependencies makes them particularly suitable for predicting high-yield bond prices, which often exhibit complex temporal patterns.

### 3.1.1 Theoretical Background

Long Short-Term Memory Networks (LSTM) and Gated Recurrent Units (GRU) are specialised forms of Recurrent Neural Networks (RNNs) designed to model sequential data and capture long-term dependencies. Both architectures were developed to address the limitations of traditional RNNs, particularly the vanishing gradient problem, which severely hinders their ability to learn from long sequences of data.

#### A - Recurrent Neural Networks (RNNs)

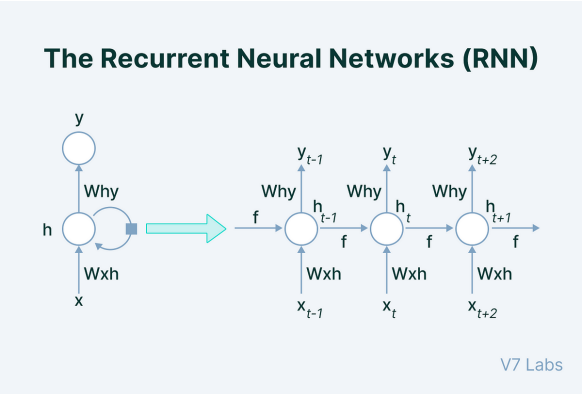



Figure 3.1: Recurrent Neural Network (RNN) Architecture. Source: Medium (2020)

Recurrent Neural Networks (RNNs) are a class of artificial neural networks designed for processing sequences of data. As illustrated in Figure 3.1, RNNs have a unique architecture with connections that form directed cycles, allowing them to maintain a state that captures information about previous inputs. This structure makes RNNs

particularly suitable for tasks where the order of inputs is important, such as time series prediction, natural language processing, and speech recognition.

In an RNN, the hidden state $h_t$ at time step $t$ is computed based on the current input $x_t$ and the hidden state from the previous time step $h_{t-1}$:

$$h_t = \tanh(W_h \cdot h_{t-1} + W_x \cdot x_t + b_h)$$

where:

- $W_h$ and $W_x$ are weight matrices.
- $b_h$ is a bias vector.
- $\tanh$ is the hyperbolic tangent activation function.

**B - The Vanishing Gradient Problem**

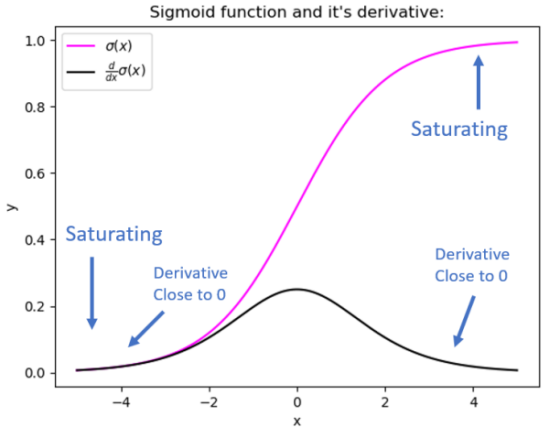



Figure 3.2: Sigmoid function and its derivative, illustrating the vanishing gradient problem. Source: Neptune ai (2024)

The vanishing gradient problem is a major challenge in training RNNs. As shown in Figure 3.2, the gradients become very small (close to 0) in the saturated regions of the sigmoid function, leading to vanishing gradients during backpropagation. This problem arises when gradients of the loss function with respect to the network parameters are computed and propagated backward through time, especially in deep networks with many layers or time steps. When these gradients become extremely small, learning can slow down or stop entirely, making it difficult for RNNs to learn

long-term dependencies, as the influence of earlier inputs diminishes exponentially with each time step.

Mathematically, the gradient of the loss function $L$ with respect to a weight parameter $W$ at time step $t$ is given by:

$$\frac{\partial L}{\partial W} = \sum_{k=0}^{t} \left( \frac{\partial L}{\partial h_t} \cdot \frac{\partial h_t}{\partial h_{t-1}} \cdot \frac{\partial h_{t-1}}{\partial h_{t-2}} \cdot \ldots \cdot \frac{\partial h_{t-k+1}}{\partial h_{t-k}} \cdot \frac{\partial h_{t-k}}{\partial W} \right)$$

If the terms $\frac{\partial h_{t-k+1}}{\partial h_{t-k}}$ are less than 1, the gradients can decrease exponentially, leading to vanishing gradients.

The vanishing gradient problem is exacerbated by the properties of common activation functions like the sigmoid function. The sigmoid function tends to squash its inputs into a small range (between 0 and 1), which can lead to very small gradients during backpropagation.

### 3.1.2 LSTM Networks

To overcome the vanishing gradient problem, Hochreiter and Schmidhuber [8] introduced the Long Short-Term Memory (LSTM) network. The key innovation of LSTMs is the introduction of a memory cell that can maintain its state over long periods, along with a set of gating mechanisms that regulate the flow of information into, out of, and within the cell. This architecture allows LSTMs to effectively capture and preserve long-term dependencies in the data.

#### LSTM Architecture

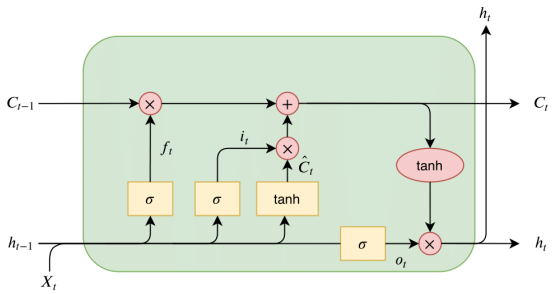



Figure 3.3: LSTM Architecture. Source: Thorir Mar Ingolfsson (2021)

The Long Short-Term Memory (LSTM) architecture includes a memory cell and three types of gates: the forget gate, the input gate, and the output gate. As shown in Figure

3.3, these gates regulate the addition, removal, and retrieval of information from the cell state, allowing LSTMs to handle long-term dependencies effectively.

- **Memory Cell (Cell State)**: The cell state $C_t$ acts as a conveyor belt, carrying information across time steps with minimal modifications. It can only be changed by the actions of the gates.

- **Forget Gate ($f_t$)**: The forget gate determines what information from the cell state should be discarded. It takes the previous hidden state $h_{t-1}$ and the current input $x_t$ and passes them through a sigmoid function:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

  where $\sigma$ is the sigmoid activation function, $W_f$ is the weight matrix, and $b_f$ is the bias vector.

- **Input Gate ($i_t$) and Candidate Cell State ($\tilde{C}_t$)**: The input gate controls which new information should be added to the cell state. It consists of two parts: a sigmoid layer that decides which values to update and a tanh layer that creates a vector of new candidate values ($\tilde{C}_t$):

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

  where $W_i$, $b_i$, $W_C$, and $b_C$ are the weights and biases for the input gate and candidate value generation, respectively.

- **Cell State Update**: The new cell state $C_t$ is then calculated by combining the old cell state and the candidate values, modulated by the forget and input gates:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

- **Output Gate ($o_t$)**: The output gate determines what part of the cell state should be output. The output is a filtered version of the cell state, based on the output gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \cdot \tanh(C_t)$$

  where $W_o$ and $b_o$ are the weights and bias for the output gate.

#### Advantages of LSTM

The primary advantage of LSTMs is their ability to maintain long-term dependencies, making them particularly effective for tasks involving sequential data. This includes applications such as language modelling, speech recognition, and time series forecasting. In financial contexts, LSTMs can capture the temporal dependencies in data, such as trends in bond prices and yields, which are crucial for accurate forecasting.

LSTM networks have been shown to significantly outperform traditional RNNs and other models in various tasks, particularly when dealing with long sequences. Their ability to selectively forget, update, and output information allows them to model complex temporal dynamics and maintain relevant information over extended periods.

In conclusion, LSTM networks are a powerful tool for sequence prediction and time series analysis, providing a robust solution to the challenges faced by traditional RNNs. Their innovative architecture and gating mechanisms enable them to effectively capture long-term dependencies and complex patterns in sequential data, making them an invaluable asset in fields such as finance, natural language processing, and beyond.

### 3.1.3 GRU Networks

To overcome the vanishing gradient problem, Kyunghyun Cho et al. [9] introduced the Gated Recurrent Unit (GRU). The key innovation of GRUs is their simplified architecture, which combines the forget and input gates into a single update gate and merges the cell state and hidden state. This architecture allows GRUs to effectively capture and preserve long-term dependencies in the data while being computationally efficient.

#### GRU Architecture

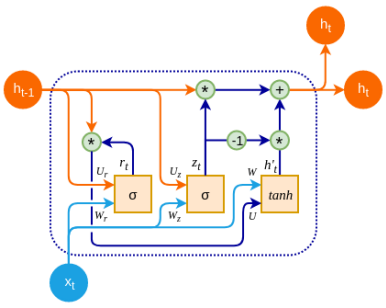



Figure 3.4: GRU Architecture. Source: Medium (2023)

The Gated Recurrent Unit (GRU) architecture is a simplified version of the LSTM that merges the cell state and hidden state and combines the forget and input gates into a single update gate. As illustrated in Figure 3.4 , GRUs use two main types of gates to control the flow of information: the reset gate and the update gate, making them simpler yet effective for maintaining long-term dependencies.

- **Reset Gate ($r_t$)**: The reset gate determines how much of the past information to forget. It takes the previous hidden state $h_{t-1}$ and the current input $x_t$ and passes them through a sigmoid function:

  $$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

  where $\sigma$ is the sigmoid activation function, $W_r$ is the weight matrix, and $b_r$ is the bias vector.

- **Update Gate ($z_t$)**: The update gate decides how much of the past information to retain and how much new information to add. It is similar to the forget and input gates combined in LSTMs:

  $$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

  where $W_z$ and $b_z$ are the weights and bias for the update gate.

- **Candidate Activation ($\tilde{h}_t$)**: The candidate activation $\tilde{h}_t$ creates a new memory content that can be added to the state, modulated by the reset gate:

  $$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t] + b)$$

  where $W$ and $b$ are the weights and bias, and $*$ denotes element-wise multiplication.

- **Hidden State Update**: The new hidden state $h_t$ is then calculated by combining the previous hidden state and the candidate activation, modulated by the update gate:

  $$h_t = z_t * h_{t-1} + (1 - z_t) * \tilde{h}_t$$

**Advantages of GRU**

The primary advantage of GRUs is their ability to maintain long-term dependencies with a simpler architecture compared to LSTMs. This simplicity translates to faster training times and fewer parameters to tune, making GRUs computationally efficient. Despite the reduced complexity, GRUs perform comparably to LSTMs on various tasks involving sequential data.

GRUs are particularly effective for tasks involving sequences with varying lengths, such as language modeling, speech recognition, and time series forecasting. In financial contexts, GRUs can capture temporal dependencies in data, such as trends in bond prices and yields, which are crucial for accurate forecasting.

In conclusion, GRU networks offer a powerful and efficient alternative to LSTMs for sequence prediction and time series analysis. Their streamlined architecture and gating mechanisms enable them to effectively capture long-term dependencies and complex patterns in sequential data, making them an invaluable asset in fields such as finance, natural language processing, and beyond.

By comparing LSTMs and GRUs, it is clear that both address the vanishing gradient problem effectively, but with different architectural approaches. LSTMs use a more complex gating mechanism to maintain long-term dependencies, while GRUs achieve similar results with a simpler, faster architecture. This makes both LSTMs and GRUs essential tools for tasks requiring the modeling of long-term dependencies in sequential data.

## 3.2 Gradient Boosting Regressor (GBR)

Gradient Boosting Regressor (GBR) is an ensemble learning technique that combines multiple weak learners, typically decision trees, to create a powerful predictive model for regression tasks. Introduced by Jerome H. Friedman [10], GBR is a type of boosting algorithm, a specific ensemble learning approach that focuses on minimising the errors of previous models through iterative refinement. Ensemble learning techniques are widely used in machine learning as they often outperform individual models by leveraging the collective wisdom of multiple models. GBR has gained popularity due to its ability to capture complex, non-linear relationships in the data, its robustness to

outliers, and its high predictive accuracy, making it an excellent choice for forecasting high-yield bonds and other challenging regression problems. GBR draws upon concepts from various disciplines, including statistics, computer science, and mathematics. Statistical principles, such as ensemble methods and loss minimisation, form the theoretical foundation of GBR. Computer science algorithms and computational techniques enable the efficient implementation and optimisation of the GBR model. Mathematics, particularly optimisation, linear algebra, and calculus, plays a crucial role in understanding and deriving the gradient boosting methodology. By integrating insights from these disciplines, GBR effectively leverages computational power to handle complex, high-dimensional data, making it particularly suitable for financial applications like high-yield bond forecasting.

### 3.2.1 Theoretical Background

GBR has been specifically designed to tackle several challenges in regression and predictive modeling. It improves prediction accuracy by ensembling multiple weak learners into a strong model capable of capturing complex patterns. Its use of decision trees allows it to model non-linear relationships effectively. GBR can handle high-dimensional data by automatically selecting and combining relevant features. Additionally, it incorporates regularisation techniques like learning rates, tree constraints, and early stopping to reduce overfitting and improve generalisation. By addressing these key issues, GBR has become a powerful algorithm for regression tasks, particularly in complex domains. This unique approach is underpinned by a robust theoretical foundation.

#### Loss Function

The loss function is a critical component of GBR as it measures the difference between the predicted values and the true target values. The goal of GBR is to minimise this loss function through iterative improvements. Common loss functions used in regression include Mean Squared Error (MSE) and Mean Absolute Error (MAE). GBR uses a specific loss function called the "negative gradient" or "residual" to determine the direction and magnitude of the updates for each new weak learner.

#### Weak Learners

Weak learners are simple models, such as decision trees, that are trained sequentially to correct the errors of the combined ensemble of previous learners. In GBR, each weak learner attempts to fit the residuals (errors) calculated based on the predictions of the entire ensemble, not just the previous model. Decision trees are commonly used as weak learners due to their ability to capture complex, non-linear relationships.

#### Learning Rate

The learning rate is a hyperparameter that controls the contribution of each weak learner to the final model. It scales the predictions of the weak learners before they are added to the ensemble. A lower learning rate can lead to better generalisation by preventing the model from overfitting to the training data, but it requires more iterations to achieve the same level of performance.

#### Additive Model

GBR constructs an additive model by sequentially adding the predictions of the weak learners, weighted by the learning rate. The final prediction is the sum of the initial prediction (e.g., the mean of the target variable) and the weighted contributions of all the weak learners.

#### Regularisation

Regularisation techniques play a crucial role in GBR to prevent overfitting and improve generalisation. These include tree constraints (maximum depth, minimum samples, etc.) and early stopping. Tree constraints limit the complexity of the individual weak learners, while early stopping terminates the training process when the model starts to overfit.

#### Optimisation

The optimisation process in GBR involves finding the optimal set of weak learners and their corresponding weights that minimise the loss function. Various optimisation

techniques, such as gradient descent or Newton's method, can be used to solve this optimisation problem.

### 3.2.2 Gradient Boosting Algorithm

#### Gradient Boosting Process and Architecture

The gradient boosting process involves iteratively adding weak learners to the ensemble, each one improving on the errors of the previous ensemble. The main steps in this process are as follows:

**Initialise the Model:** The initial model $F_0(x)$ is chosen to minimise the loss function. For regression tasks, this is typically the mean of the target values $y$ across all $N$ data points (samples).

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma)$$

**Iterative Process:** For each iteration $m = 1, 2, \ldots, M$:

**Compute Residuals:** The residuals $r_{im}$ represent the negative gradient of the loss function with respect to the current model's predictions. These residuals indicate the errors that the new weak learner should focus on.

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$$

For the commonly used Mean Squared Error (MSE) loss function, the residuals are simply the differences between the actual and predicted values:

$$r_{im} = y_i - F_{m-1}(x_i)$$

**Fit Weak Learner:** A weak learner $h_m(x)$, typically a decision tree, is fitted to the residuals. The goal is to find a model that best approximates these residuals.

$$h_m(x) = \arg\min_{h} \sum_{i=1}^{N} (r_{im} - h(x_i))^2$$

**Update the Model:** The model is updated by adding the new weak learner, scaled by the learning rate $\nu$. This update rule ensures that each new learner makes a small contribution to the ensemble, which helps in reducing overfitting.

$$F_m(x) = F_{m-1}(x) + \nu h_m(x)$$

This iterative process continues until a stopping criterion is met, such as a predefined number of iterations $M$ or a convergence threshold for the loss function.

The core idea of gradient boosting is rooted in gradient descent, an optimisation technique used to minimise a function by iteratively moving towards the steepest descent direction (negative gradient). In the context of GBR, we are minimising the loss function. At each iteration, the model is updated to reduce the residuals, which can be seen as taking a step in the direction of the negative gradient of the loss function with respect to the model's predictions. Boosting is a general method of converting a set of weak learners into a strong learner. In GBR, boosting is achieved by sequentially adding weak learners, each one correcting the errors of its predecessors. This approach ensures that the model becomes progressively better at predicting the target variable.

```
Initialize F_0(x) = arg min_γ Σ_{i=1}^{N} L(y_i, γ)

For m = 1 to M:

    Compute Residuals:
        For i = 1 to N:
            r_im = − [ ∂L(y_i, F(x_i)) / ∂F(x_i) ]_{F(x)=F_{m−1}(x)}
            // For MSE loss: r_im = y_i − F_{m−1}(x_i)

    Fit Weak Learner:
        h_m(x) = arg min_h Σ_{i=1}^{N} (r_im − h(x_i))^2

    Update the Model:
        F_m(x) = F_{m−1}(x) + ν h_m(x)

End For
```

This pseudocode describes the iterative process of gradient boosting, where each step aims to improve the model by adding a new weak learner that corrects the errors of the previous model. The steps ensure that each new learner focuses on the areas where the previous model was underperforming, gradually improving the overall model's accuracy and robustness.

**Advantages and Practical Considerations**

GBR is particularly well-suited for high-yield bond forecasting due to its ability to handle complex and nonlinear relationships within financial data. High-yield bonds, often subject to significant market volatility and credit risk, require robust models that can accurately capture these dynamics. GBR excels at modeling complex, nonlinear relationships between bond features and prices. Unlike linear models, which assume a straight-line relationship between the independent and dependent variables, GBR can capture intricate patterns in the data. This ability is crucial in financial markets where relationships between variables are rarely linear. GBR provides valuable insights into the importance of different features, aiding in understanding the factors driving bond prices. By examining the contribution of each feature to the model's predictions, analysts can identify the key drivers of bond performance. This information can be used to inform investment decisions and risk management strategies. GBR is robust to overfitting, especially with proper tuning of hyperparameters like learning rate and number of trees. Overfitting occurs when a model performs well on the training data but poorly on unseen data. GBR mitigates this risk through regularisation techniques such as shrinkage (learning rate adjustment) and subsampling. These techniques ensure that the model generalises well to new data, providing reliable predictions. GBR can handle various types of data, including numerical, categorical, and missing data. This flexibility makes it a versatile tool for bond forecasting, where different types of data are often used. By combining multiple weak learners, GBR creates a strong predictive model. This ensemble approach leverages the strengths of individual learners while mitigating their weaknesses, resulting in a model with high predictive accuracy.

## 3.3 Stacked Generalisation (Stacking)

Stacked Generalisation, commonly referred to as Stacking, is an ensemble learning technique that combines multiple machine learning models to improve predictive performance. Introduced by Wolpert [11], Stacking is a method that aims to harness the strengths of different models by training a "meta-learner" to best combine their outputs. This approach allows the ensemble to correct individual weaknesses and achieve better generalisation than any single model alone.

In the context of this study, GBM and GRU are utilised as base (or weak) learners, while Long Short Term Memomry (LSTM) serves as the meta-learner. The combination of these models leverages the temporal sequence modeling capabilities of GBM and GRU with the powerful predictive accuracy of LSTM, making it particularly effective for complex regression tasks such as high-yield bond forecasting.

### 3.3.1 Stacked Generalisation Architecture

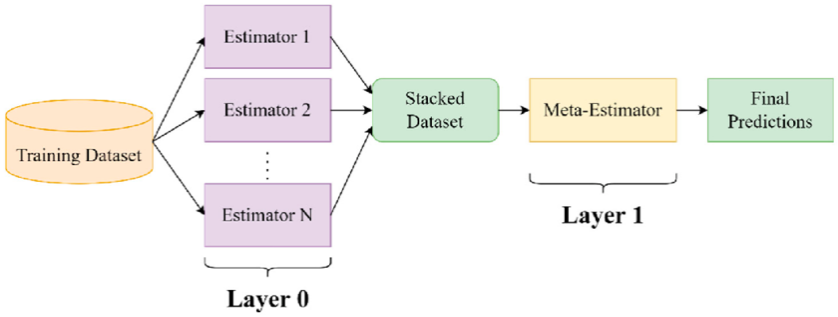



Figure 3.5: Stacked Generalisation (Stacking) Architecture.Source: Stacking Architecture (2024)

Stacked Generalisation, as shown in Figure 3.5, builds on the principles of ensemble learning, where multiple models are combined to form a more robust and accurate predictive system. Stacking involves training several base learners on the dataset and then using a meta-learner to make final predictions based on the outputs of these base learners. This two-level architecture helps capture different patterns in the data and reduces the risk of overfitting that individual models might encounter. The architecture of a stacked generalisation model involves two levels of learning:

#### Level-1: Base Learners

In the first level, multiple base learners are trained independently on the training data. Each base learner produces its own predictions, capturing different aspects of the data. In our case, the base learners are GBM and GRU models, each capable of modeling the temporal dependencies in the dataset.

**Level-2: Meta-Learner**

In the second level, a meta-learner (LSTM) is trained on the outputs of the base learners. The meta-learner takes the predictions of the base learners as input features and learns to combine them in a way that minimises the overall prediction error. This approach helps in leveraging the strengths of each base learner while mitigating their individual weaknesses.

### 3.3.2 Stacked Generalisation Algorithm

The stacked generalisation algorithm can be outlined as follows:

**Inputs:**

- Training dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$: The set of training examples, where $x_i$ are the input features and $y_i$ are the target values. - Base learners $\Phi_1, \Phi_2, \ldots, \Phi_T$:** The set of base learners used to make initial predictions. These can be any models. - Secondary learners $\Phi$: The meta-learner used to combine the predictions of the base learners.

**Process:**

1. Train Base Learners:

- For each base learner $\Phi_t$ (where $t = 1, 2, \ldots, T$):

$$h_t = \Phi_t(D)$$

- Train the base learner $\Phi_t$ on the entire training dataset $D$.

2. Generate Meta-Training Dataset:

- Initialise an empty dataset $D'$.
- For each training example $(x_i, y_i)$ (where $i = 1, 2, \ldots, m$):
  - For each base learner $\Phi_t$:

$$z_{it} = h_t(x_i)$$

  - Compute the prediction $z_{it}$ of the base learner $h_t$ on the input $x_i$.

- Add the tuple $((z_{i1}, z_{i2}, \ldots, z_{iT}), y_i)$ to the meta-training dataset $D'$.

3. Train Meta-Learner:

- Train the meta-learner $\Phi$ on the meta-training dataset $D'$:

$$h' = \Phi(D')$$

**Output:**

The final prediction $H(x)$ is obtained by combining the predictions of the base learners using the meta-learner:

$$H(x) = h'(h_1(x), h_2(x), \ldots, h_T(x))$$

```
Algorithm: Stacked Generalization

Input: Training dataset D = {(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)};
       Base learners Φ_1, Φ_2, ..., Φ_T;
       Secondary learners Φ'

Process:
For t = 1 to T:
    h_t = Φ_t(D)
End For

D' = ∅;

For i = 1 to m do:
    For t = 1 to T do:
        z_it = h_t(x_i)
    End For
    D' = D' ∪ ((z_i1, z_i2, ..., z_iT), y_i)
End For

h' = Φ'(D')

Output: H(x) = h'(h_1(x), h_2(x), ..., h_m(x))
```

This pseudocode illustrates the process of training base learners, generating meta-features, adn training the meta-learners to produce the final prediction.

This algorithm effectively combines the strengths of multiple base learners to produce a more robust and accurate predictive model. By training a meta-learner on the outputs of the base learners, stacked generalisation can capture complex patterns and relationships in the data that individual models might miss.

**Advantages of Stacking Generalisation**

Stacking Generalisation offers numerous advantages that enhance its effectiveness in predictive modeling. By combining the strengths of different models, stacking often leads to superior predictive performance. The meta-learner integrates insights from each base learner, resulting in a robust and accurate model. This ensemble approach significantly reduces the risk of overfitting, which individual models might encounter, as the meta-learner smooths out biases and errors from the base learners.

The versatility of stacking is another key advantage. It can combine various types of models, such as neural networks, decision trees, and other machine learning algorithms, to create powerful ensembles capable of tackling a wide range of predictive tasks. This flexibility allows stacking to adapt to different types of data and problems, making it an invaluable tool in various fields, including finance, healthcare, and marketing.

One of the practical considerations of stacking is its ability to capture complex patterns in the data. By integrating outputs from diverse models, stacking can model intricate relationships within the dataset more effectively than individual models. This capability is particularly useful for datasets with non-linear relationships and high-dimensional features.

The robustness of the stacked model is significantly improved by leveraging the diversity of the base learners. This diversity helps in reducing model variance and enhances generalisation to new data, making the stacked model more reliable when applied to unseen datasets. However, to achieve optimal performance, careful tuning of hyperparameters for both the base learners and the meta-learner is crucial. Employing cross-validation techniques helps in selecting the best hyperparameters and avoiding overfitting.

Despite these advantages, stacking does increase computational complexity due to the need to train multiple models and the meta-learner. This can be mitigated through efficient implementation and parallel processing techniques, which help manage the additional computational load. By addressing these practical considerations, stacking regressor ensures a balance between enhanced predictive performance and computational efficiency, making it a powerful method for regression and predictive modeling tasks.

In conclusion, stacked generalisation is a powerful ensemble learning technique

that combines the strengths of multiple models to achieve superior predictive performance. By using GBM and GRU as base learners and LSTM as the meta-learner, the ensemble can effectively capture both temporal dependencies and complex, non-linear relationships in the data, making it particularly suitable for tasks such as high-yield bond forecasting.

## 3.4 Evaluation Metrics

Several evaluation metrics are used to assess the performance of regression models. The choice of metrics depends on the specific characteristics of the task and the goals of the analysis. For high-yield bond forecasting, it is crucial to select metrics that accurately capture the nuances of prediction errors and provide meaningful insights for financial decision-making.

#### R-squared ($R^2$)

$R^2$, also known as the coefficient of determination, measures the proportion of the variance in the dependent variable that is predictable from the independent variables.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{3.1}$$

where $y_i$ is the actual value, $\hat{y}_i$ is the predicted value, and $\bar{y}$ is the mean of the actual values.

- $R^2$ indicates how well the model explains the variability of the outcome data. A higher $R^2$ value signifies a better fit. For high-yield bonds, explaining the variance is crucial because it indicates how well the model can account for the factors driving bond prices and yields.

- A $R^2$ value close to 1 indicates that the model explains most of the variance in the dependent variable, implying a good fit. For financial forecasting, values above 0.7 are generally considered good, but this can vary depending on the complexity of the data.

**Adjusted R-squared (Adjusted $R^2$)**

Adjusted $R^2$ adjusts the $R^2$ value based on the number of predictors in the model. It penalises the addition of irrelevant predictors.

$$\text{Adjusted } R^2 = 1 - \left( \frac{(1 - R^2) \cdot (n - 1)}{n - p - 1} \right) \tag{3.2}$$

where $n$ is the number of observations and $p$ is the number of predictors.

- Adjusted $R^2$ provides a more accurate measure of model performance by accounting for the number of predictors, helping to avoid overfitting. This is particularly relevant for high-yield bond forecasting as models may include numerous economic indicators and financial ratios.

- Similar to $R^2$, a higher Adjusted $R^2$ value indicates a better model fit. An Adjusted $R^2$ value that remains high when more predictors are added shows that these predictors are meaningful.

**Mean Squared Error (MSE)**

MSE measures the average of the squares of the errors, i.e., the average squared difference between the actual and predicted values.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3.3}$$

- MSE is widely used because it penalises larger errors more than smaller ones, providing a clear indication of model accuracy. This is essential in high-yield bond forecasting where large prediction errors can significantly impact financial decisions.

- Lower MSE values indicate better model performance. In financial forecasting, the acceptable range of MSE depends on the scale of the target variable.

**Root Mean Squared Error (RMSE)**

RMSE is the square root of the MSE, providing an error metric in the same units as the original data.

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{3.4}$$

- RMSE is more interpretable than MSE because it is in the same units as the target variable. It is useful for understanding the magnitude of prediction errors, which is particularly important in high-yield bond forecasting to assess the financial impact of errors.
- Similar to MSE, lower RMSE values indicate better model performance. RMSE is particularly useful when the target variable has a meaningful unit of measurement, such as bond prices or yields.

**Mean Absolute Error (MAE)**

MAE measures the average of the absolute differences between the actual and predicted values.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.5}$$

- MAE provides a straightforward interpretation of average prediction error, making it easier to understand the model's performance. It is less sensitive to outliers compared to MSE and RMSE, which can be advantageous in high-yield bond forecasting where extreme values may be less informative.
- Lower MAE values indicate better model performance. MAE is particularly useful for understanding the average magnitude of errors in the same units as the target variable.

**Mean Absolute Percentage Error (MAPE)**

MAPE measures the average absolute percentage error between the actual and predicted values.

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{3.6}$$

- MAPE is useful for understanding the prediction error relative to the actual values, expressed as a percentage, which can be easier to interpret in some contexts.

However, it can be problematic when actual values are close to zero, which is less likely in high-yield bond prices but should still be considered.

- Lower MAPE values indicate better model performance. MAPE is particularly useful in financial forecasting where understanding the relative error is important, especially when comparing bonds of different magnitudes.

**Importance of Using These Metrics**

Using a combination of these evaluation metrics provides a comprehensive assessment of model performance:

- $R^2$ **and Adjusted** $R^2$ offer insights into the proportion of variance explained by the model, which is crucial for understanding how well the model captures the underlying factors affecting bond prices.
- **MSE and RMSE** highlight the magnitude of prediction errors, with RMSE providing a more interpretable measure. These metrics are particularly relevant in financial contexts to quantify the potential impact of prediction errors on investment decisions.
- **MAE** provides an average measure of prediction errors, offering a clear and straightforward interpretation, useful for assessing the general accuracy of the model.
- **MAPE** gives a percentage-based error metric, which can be particularly useful when dealing with financial data where relative errors are more meaningful, aiding in comparing performance across different bonds.

CHAPTER 4

# Data Analysis

## 4.1 Data Collection

The accuracy and robustness of any financial forecasting model significantly depend on the quality and comprehensiveness of the data used. For this study, a diverse set of datasets was collected to develop and validate a quantitative forecasting model for high-yield bonds. These datasets encompass key financial and economic indicators relevant to bond markets, enabling a holistic approach to forecasting. The data sources include historical bond prices, macroeconomic indicators, and other financial metrics, each playing a crucial role in capturing the multifaceted dynamics of high-yield bonds.

High-yield bonds are influenced by a variety of factors, including market conditions, economic performance, and monetary policy. To accurately predict bond prices and yields, it is essential to consider multiple data sources that collectively reflect these influences. High-yield bond markets are complex and affected by a broad spectrum of factors, so by integrating diverse datasets, we can capture a more comprehensive view of the market conditions. This holistic approach ensures that the model accounts for various aspects that drive bond performance, such as economic growth, inflation, interest rates, and employment levels.

Financial markets are interconnected, and the performance of high-yield bonds can be influenced by macroeconomic trends and financial policies. For instance, changes in the federal funds rate directly impact borrowing costs and investor behavior. By

incorporating macroeconomic indicators like GDP, CPI, and unemployment rates, the model can better understand these interdependencies and how they affect bond prices. A model that only considers historical bond prices might miss out on important external factors that drive market trends. Including economic indicators helps the model recognise broader economic cycles and anticipate market movements more accurately. This leads to more reliable predictions and better-informed investment strategies.

Furthermore, understanding the macroeconomic context helps in assessing the risk associated with high-yield bonds. For example, high unemployment rates might indicate economic distress, leading to higher default risks for corporate bonds. By integrating such indicators, the model can provide a more nuanced risk assessment. High-yield bonds are particularly sensitive to market volatility and economic shifts. A diverse dataset that includes various financial and economic indicators helps the model adapt to different market conditions and remain robust even during periods of high volatility.

The JNK dataset, collected from Yahoo Finance [12], provides daily historical prices of the SPDR Bloomberg High Yield Bond ETF, which serves as a proxy for high-yield bond performance. The Close price from this dataset is used as the target variable for the forecasting models. Yahoo Finance is a widely recognised and trusted source for financial market data, offering real-time and historical data that is regularly updated and validated. This dataset directly represents the performance of high-yield bonds, making it highly relevant for this study.

The Gross Domestic Product (GDP) dataset, collected from Federal Reserve Economic Data (FRED)[13], contains quarterly real GDP values, which represent the economic output of a country adjusted for inflation. FRED is a reputable source managed by the Federal Reserve Bank of St. Louis, providing accurate and timely economic data. GDP is a critical macroeconomic indicator reflecting the overall economic health and is included to assess its impact on high-yield bond prices.

The Federal Funds Rate dataset, also collected from FRED, includes the monthly federal funds rate, which is the interest rate at which depository institutions trade federal funds with each other overnight [14]. As with GDP data, this dataset from FRED is reliable and frequently updated. The federal funds rate influences borrowing costs and economic activity, making it a vital predictor in the bond market.

The Consumer Price Index (CPI) dataset, collected from the Organisation for Economic Co-operation and Development (OECD)[15], provides monthly percentage changes in the Consumer Price Index, focusing on clothing and footwear as a representative category. The variables include Date and Value (Percentage Change). The OECD is an international organisation known for its high-quality economic data, ensuring reliability. CPI is an essential measure of inflation, impacting the purchasing power and returns on bonds.

The Unemployment Rate dataset, collected from FRED, records the monthly unemployment rate, indicating the percentage of the labor force that is jobless and actively seeking employment [16]. The Federal Reserve provides accurate and authoritative economic data, ensuring the reliability of this dataset. The unemployment rate is a crucial indicator of economic conditions and consumer confidence, which in turn affect bond markets.

The final integrated dataset, named *SPDR Bloomberg High Yield Bond*, combines the historical bond prices with macroeconomic indicators such as GDP, federal funds rate, CPI, and the unemployment rate. This dataset provides a rich and diverse foundation for developing advanced forecasting models using machine learning and deep learning techniques. By leveraging these varied data sources, the study aims to achieve a comprehensive understanding of the factors influencing high-yield bond prices and improve the accuracy of predictive models.

## 4.2 Data Description

The dataset utilised in this dissertation for the quantitative forecasting of high-yield bonds combines daily historical prices of the SPDR Bloomberg High Yield Bond ETF with key macroeconomic indicators. This integrated dataset provides a comprehensive foundation for developing advanced forecasting models using machine learning and deep learning techniques.

The dataset comprises multiple variables that capture both market-specific and macroeconomic factors influencing high-yield bond prices. These variables include Date, which represents the specific date for each record, spanning from January 2, 2014, onwards. The Open price indicates the opening price of the SPDR Bloomberg High Yield

Bond ETF on each trading day, while the High and Low prices represent the highest and lowest prices reached by the ETF on each trading day, respectively. The Close price is the closing price of the ETF on each trading day, which serves as the primary target variable for forecasting models. Additionally, the Adj Close is the adjusted closing price, accounting for corporate actions such as dividends and stock splits. Volume denotes the total trading volume of the ETF on each trading day. GDP represents the Gross Domestic Product value on each date, reflecting the overall economic output and health. The FEDFUNDS indicates the federal funds rate, which is the interest rate at which depository institutions trade federal funds overnight. The Unemployment rate reflects the percentage of the labor force that is jobless and actively seeking employment. Finally, Inflation represents the inflation rate, particularly the percentage change in the Consumer Price Index for clothing and footwear.

The integration of historical bond prices with macroeconomic indicators such as GDP, federal funds rate, unemployment rate, and inflation provides a robust dataset that captures the multifaceted dynamics of the high-yield bond market. This comprehensive dataset enables the development of predictive models that account for both market-specific trends and broader economic conditions.

The dataset spans multiple years, providing a rich temporal dataset for time-series analysis. The integrated dataset of SPDR Bloomberg High Yield Bond ETF prices and macroeconomic indicators forms a comprehensive foundation for the quantitative forecasting of high-yield bonds. By leveraging this rich dataset, advanced machine learning and deep learning models can be developed to improve prediction accuracy and provide valuable insights for investors and financial institutions.

## 4.3 Data Preprocessing and Feature Engineering

In this section, we describe the data preprocessing steps and feature engineering techniques used to prepare the dataset for model development. Proper preprocessing and feature engineering are crucial for improving the performance of machine learning and deep learning models by ensuring the data is clean, well-structured, and informative.

### 4.3.1 Handling Missing Values

The first step in data preprocessing is to check for and handle missing values. For this dataset, we verified that there were no missing values, ensuring the integrity of the data. This step is crucial as missing values can lead to incorrect model training and poor predictive performance.

### 4.3.2 Date Conversion and Time-Based Features

The dataset includes a date variable that is essential for time series analysis. We converted the date variable to a datetime format to facilitate time-based feature extraction. Using the converted date variable, we created several time-based features: Day of Week, Month, Quarter, and Year. These time-based features help capture seasonality and trends in the data, which are important for improving the accuracy of the forecasting models.

### 4.3.3 Moving Averages and Lag Features

To capture the temporal dependencies and smooth out short-term fluctuations in the 'Close' price variable, we created moving averages and lag features. These features are crucial for understanding the momentum and trends in high-yield bond prices.

#### Moving Averages

We calculated the following moving averages using the 'Close' price: MA10, MA20, and MA50. These moving averages are computed by averaging the closing prices over the last 10, 20, and 50 days, respectively. They help identify recent price trends and potential reversals, providing a clearer view of the underlying trend.

#### Lag Features

Lag features represent the past values of the 'Close' price and are used to model the autocorrelation in the time series data. We created Lag1, Lag2, and Lag3 features to capture the immediate past price and identify short-term price patterns.

### 4.3.4 Daily Return

The daily return is a measure of the percentage change in the 'Close' price from one day to the next. It is an important feature for capturing the day-to-day volatility in bond prices. We calculated the daily return as follows:

$$\text{Daily Return} = \frac{\text{Close}_t - \text{Close}_{t-1}}{\text{Close}_{t-1}}$$

Daily return is essential for understanding the short-term performance of the bond and identifying periods of high volatility, which are crucial for risk management.

### 4.3.5 Relative Strength Index (RSI)

The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements [17]. It is used to identify overbought or oversold conditions in the market. We calculated the RSI using the 'Close' price with a standard 14-day period:

$$\text{RSI} = 100 - \left( \frac{100}{1 + \frac{\text{Average Gain}}{\text{Average Loss}}} \right)$$

RSI is important for identifying potential price reversals and market conditions, helping to make informed trading decisions.

### 4.3.6 Moving Average Convergence Divergence (MACD)

The Moving Average Convergence Divergence (MACD) is a trend-following momentum indicator that shows the relationship between two moving averages of a security's price. We calculated the MACD using the 'Close' price with the following parameters: MACD Line and Signal Line. The MACD Line is the difference between a shorter-term (typically 12-period) and a longer-term (typically 26-period) Exponential Moving Average (EMA). An EMA is similar to a simple moving average but gives more weight to recent prices, making it more responsive to new information. The Signal Line is a 9-period EMA of the MACD Line. When the MACD Line crosses above the Signal Line, it may indicate a buy signal, while crossing below may indicate a sell signal.[18].

### 4.3.7 Bollinger Bands

Bollinger Bands consist of three lines: a middle band (a simple moving average) and two outer bands that are set a certain number of standard deviations away from the middle band. They are used to spot times when the price is very high or very low, which might indicate that the price is about to reverse [19]. We calculated the Bollinger Bands using the 'Close' price with a 20-day period and 2 standard deviations: Bollinger Upper and Bollinger Lower. Bollinger Bands help identify whether the market is experiencing high or low volatility and can signal when a security might be overbought or oversold, helping to time trades.

The final dataset includes the following features: Date, Open, High, Low, Close, Adj Close, Volume, Day of Week, Month, Quarter, Year, MA10, MA20, MA50, Lag1, Lag2, Lag3, Daily Return, RSI, MACD Line, Signal Line, Bollinger Upper, Bollinger Lower, GDP, FEDFUNDS, UNRATE, and Inflation. These features collectively capture the temporal dynamics, market trends, and macroeconomic conditions influencing high-yield bond prices. By incorporating these engineered features, the forecasting models can better understand the complex relationships and improve prediction accuracy.

The preprocessed and feature-engineered dataset forms the foundation for developing robust machine learning and deep learning models for high-yield bond forecasting. The next section will discuss the specific models used and their implementation

## 4.4 Exploratory Data Analysis (EDA)

The Exploratory Data Analysis (EDA) section aims to understand the dataset's underlying patterns, relationships, and anomalies through various statistical and graphical techniques. EDA helps to identify the key drivers influencing high-yield bond prices, ensuring the selection of relevant features for the forecasting models. In this section, we perform a detailed analysis of the dataset, including visualisations and statistical summaries.

### 4.4.1 Summary Statistics

Table 4.1: Summary Statistics of Key Variables

| Variable | Mean | Standard Deviation | Minimum | Maximum |
|---|---|---|---|---|
| Close Price | 119.56 | 3.54 | 106.95 | 126.69 |
| GDP ($ billion) | 19,635.02 | 755.12 | 17,953.97 | 21,483.50 |
| Federal Funds Rate (%) | 0.81 | 0.98 | 0.07 | 2.42 |
| Unemployment Rate (%) | 4.67 | 1.30 | 3.50 | 8.10 |
| Inflation (%) | 1.53 | 0.91 | -2.18 | 3.16 |
| Day_of_Week | 2.024 | 1.398 | 0.000 | 4.000 |
| Month | 6.506 | 3.419 | 1.000 | 12.000 |
| Quarter | 2.502 | 1.113 | 1.000 | 4.000 |
| Year | 2018.768 | 2.947 | 2014.000 | 2024.000 |
| MA10 | 106.032 | 8.680 | 88.244 | 125.205 |
| MA20 | 106.092 | 8.666 | 88.475 | 124.945 |
| MA50 | 106.260 | 8.626 | 89.599 | 124.485 |
| Lag1 | 105.992 | 8.699 | 84.570 | 125.400 |
| Lag2 | 106.003 | 8.703 | 84.570 | 125.400 |
| Lag3 | 106.015 | 8.706 | 84.570 | 125.400 |
| Daily Return | -0.000 | 0.005 | -0.058 | 0.067 |
| RSI | 49.870 | 17.114 | 2.247 | 100.000 |
| MACD | -0.080 | 0.609 | -5.709 | 1.719 |
| MACD_Signal | -0.080 | 0.567 | -4.601 | 1.446 |
| Bollinger_Upper | 107.662 | 8.381 | 89.863 | 125.932 |
| Bollinger_Lower | 104.506 | 9.160 | 81.349 | 124.354 |

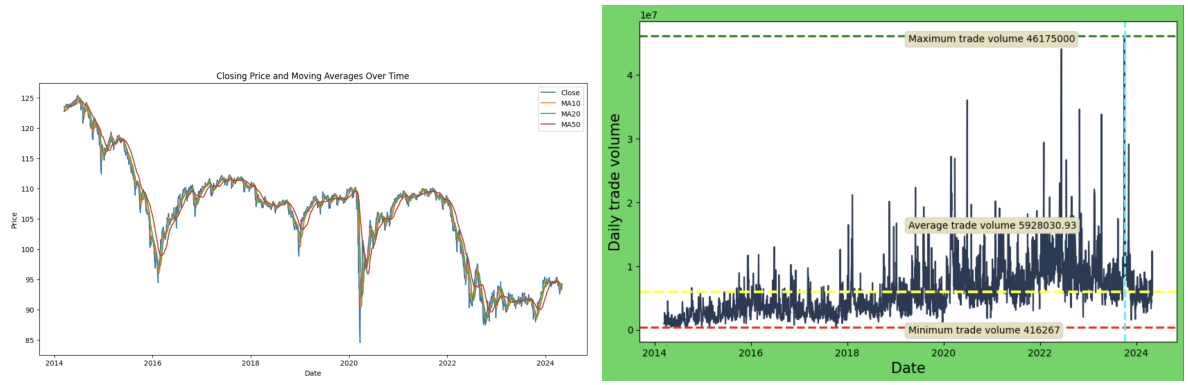
We begin by examining the summary statistics of the key variables to understand their central tendency and dispersion. This includes the mean, standard deviation, minimum, and maximum values, providing a basic understanding of the data distribution.

The summary statistics table provides a snapshot of the dataset's characteristics. For instance, the mean *Close Price* of the SPDR Bloomberg High Yield Bond ETF is approximately 119.56, with a standard deviation of 3.54, indicating moderate variability in bond prices. The GDP values, ranging from $17,953.97 billion to $21,483.50 billion, reflect significant economic growth over the period under study. The Federal Funds Rate, with a mean of 0.81% and a range from 0.07% to 2.42%, highlights periods of both low and relatively higher interest rates, which can significantly impact bond prices. The Unemployment Rate shows a mean of 4.67% and varies between 3.50% and 8.10%, capturing different phases of the economic cycle. Inflation, represented by the Consumer Price Index (CPI), has a mean of 1.53%, with values ranging from -2.18% to

3.16%, indicating periods of both deflation and inflation. The moving averages (MA10, MA20, MA50) and lag features (Lag1, Lag2, Lag3) provide insights into the short-term and long-term trends and autocorrelation in bond prices. The technical indicators such as RSI, MACD, and Bollinger Bands are crucial for capturing market momentum and potential price reversals. For example, the RSI has a mean value of 49.87, with a wide range from 2.25 to 100, suggesting varied market conditions from oversold to overbought states. Overall, these summary statistics are critical for understanding the dataset's distribution and variability, which are essential for feature engineering and model training. By analysing these metrics, we can identify key patterns and trends that influence high-yield bond prices, enabling the development of robust predictive models.

### 4.4.2 Time Series Analysis

Time series plots provide insights into trends, seasonality, and potential anomalies in the data over time.




(a) Close Price and Moving Averages of SPDR Bloomberg High Yield Bond ETF Over Time

(b) Trading Volume of SPDR Bloomberg High Yield Bond ETF Over Time

Figure 4.1: Close Price, Moving Averages, and Trading Volume of SPDR Bloomberg High Yield Bond ETF Over Time

#### Close Price and Moving Averages Over Time

Plotting the Close price along with the 10-day, 20-day, and 50-day moving averages helps us visualise the overall trend and identify short-term and long-term trends in the bond prices.

Figure 4.1(a) shows the Close Price of the SPDR Bloomberg High Yield Bond ETF along with the 10-day (MA10), 20-day (MA20), and 50-day (MA50) moving averages. This visualisation highlights several key aspects of the data, including trends, volatility, mean reversion, and support and resistance levels. The plot reveals both short-term and long-term trends in the bond prices, where the moving averages help to smooth out daily fluctuations, making it easier to identify overall trends. For example, the periods of decline in 2015-2016 and 2020 are clearly visible.

The decline in 2015-2016 can be attributed to several significant events. In December 2015, the Federal Reserve raised interest rates for the first time since 2006, leading to higher borrowing costs and reduced attractiveness of high-yield bonds[20]. Global economic concerns, particularly China's slowdown and stock market volatility, drove investors to safer government bonds, reducing demand for high-yield bonds. Additionally, the collapse in oil prices during this period significantly impacted energy-dependent sectors, further adding to the uncertainty in the high-yield bond market. The Brexit referendum in June 2016 also contributed to market volatility and a flight to safety, negatively impacting high-yield bonds[21].

The 2020 decline was primarily driven by the onset of the COVID-19 pandemic, which caused unprecedented economic disruptions and heightened credit risks. Investors sought the safety of government bonds, leading to record-low yields, while high-yield bonds suffered due to increased default risks [22][23]. The Federal Reserve's emergency rate cuts and extensive quantitative easing measures aimed at stabilising financial markets provided some support, but the overall uncertainty and market volatility led to significant price declines in high-yield bonds [24]. Massive fiscal stimulus packages, such as the U.S. CARES Act, increased government borrowing and affected Treasury yields, but the riskier high-yield bond market faced heightened concerns[25].

The gaps between the Close Price and moving averages indicate periods of higher volatility, with large deviations signaling significant market events or changes in investor sentiment. The frequent crossing of the Close Price over the moving averages suggests potential mean reversion behavior, where bond prices tend to revert to the average value over time. Additionally, moving averages often act as support and resistance levels, guiding trading decisions. For instance, the Close Price bouncing off the MA50 line suggests its use as a reference point by investors. By analysing the Close

Price and moving averages together, we gain valuable insights into the dynamics of high-yield bond prices, crucial for developing effective forecasting models that account for both short-term fluctuations and long-term trends.

**Trading Volume Over Time**

Analysing the trading volume over time can reveal periods of high trading activity, which might be correlated with significant market events or changes in the bond's price. Figure 4.1(b) above illustrates the daily trade volume of the SPDR Bloomberg High Yield Bond ETF from 2014 to 2024. Key aspects of this figure include volume trends, volatility and liquidity, statistical markers, and market events. The figure shows an increasing trend in trading volume over the years, indicating growing interest and activity in the high-yield bond market. Notably, there are significant spikes in volume during periods of market stress, such as in early 2020, and at the end of 2023. Higher trading volumes generally correlate with increased market liquidity, allowing for easier execution of large trades without significantly impacting the price. The spikes in volume often coincide with periods of high volatility, reflecting rapid buying or selling pressure. This visualisation highlights the maximum, minimum, and average trade volumes. The maximum trade volume of approximately 46,175,000 shares at the end of 2023 indicates periods of extreme market activity, while the minimum trade volume of 416,267 shares in early 2020 represents quieter trading days. The average trade volume is around 5,928,030.93 shares, providing a baseline for typical market activity. The minimum trading volume in early 2020 coincides with the onset of the COVID-19 pandemic. As the pandemic led to widespread economic disruptions and uncertainty, investors initially moved to safer assets like government bonds[22]. The flight to safety resulted in lower trading volumes for riskier assets, including high-yield bonds, as investors sought to avoid potential losses amidst the market turmoil. This period of low activity reflected the cautious stance of investors who were waiting for more clarity on the economic impact of the pandemic before re-entering the market. The maximum trading volume at the end of 2023 can be attributed to several significant market events and economic conditions. By the end of 2023, markets were likely responding to a combination of factors such as interest rate changes, economic recovery post-pandemic, and possibly major fiscal or monetary policy announcements[25]. The heightened trading activity

could also be driven by year-end portfolio rebalancing and adjustments by institutional investors, who often engage in significant trading to align their portfolios with their investment strategies for the upcoming year. Additionally, any geopolitical events or major corporate actions around this time could have further spurred trading volumes, reflecting heightened investor engagement and market activity. Significant increases in trade volume often correspond to major market events or economic announcements. These spikes can provide insights into investor behavior and market sentiment during such events. Understanding the trade volume dynamics is essential for high-yield bond forecasting as it provides context on market liquidity and activity levels. High trade volumes can indicate heightened interest or concern among investors, which can impact bond prices. By incorporating trade volume data into the forecasting models, we can improve their accuracy and robustness in capturing market trends and investor behavior.

### 4.4.3 Correlation Analysis

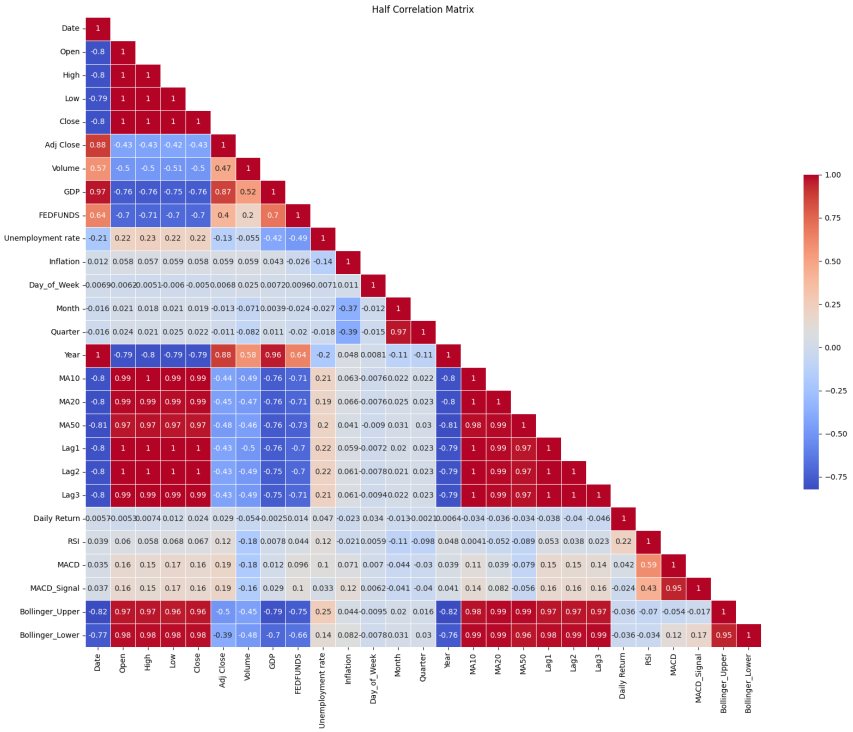


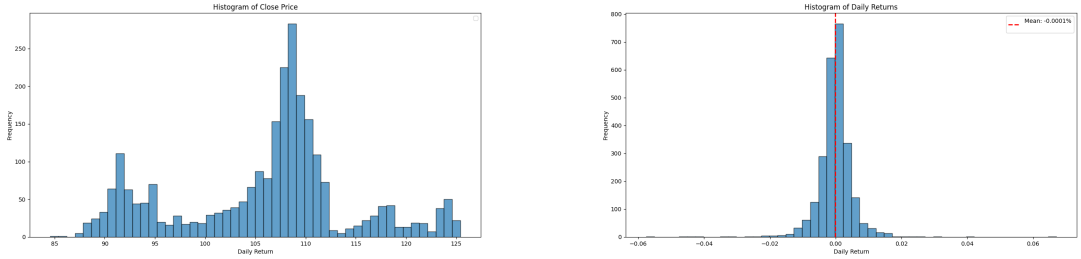

Figure 4.2: Correlation Matrix of Key Variables

The correlation matrix 4.2 provides an in-depth look at the linear relationships between various features in the dataset. This matrix is essential for understanding how different variables interact with each other, which is crucial for feature selection and model building in the forecasting process. The correlation matrix reveals several important relationships among the variables, including strong positive correlations, strong negative correlations, moderate positive and negative correlations, time-based features, and technical indicators. The Close price is strongly positively correlated with the moving averages MA10 (0.99), MA20 (0.99), and MA50 (0.99). This indicates that the closing price of the bond ETF closely follows the trends captured by these moving averages. High, Low, and Open prices also show strong positive correlations with the Close price, each around 0.99. This is expected as these prices typically move together during trading sessions. GDP has a strong positive correlation with the Close price (0.87), reflecting the influence of overall economic health on high-yield bond prices. The Federal Funds Rate shows a strong negative correlation with the Close price (-0.70) and a moderate negative correlation with GDP (-0.71), indicating that higher interest rates tend to correspond with lower bond prices and economic output. Volume shows a moderate negative correlation with the Close price (-0.51), suggesting that higher trading volumes may be associated with lower bond prices, possibly due to increased selling pressure. The Unemployment Rate has a moderate negative correlation with the Close price (-0.49), reflecting the adverse impact of higher unemployment on bond prices. Inflation has a lower correlation with the Close price (0.43) but shows a moderate positive correlation with the Federal Funds Rate (0.42), indicating the interconnectedness of these macroeconomic factors. Year shows a negative correlation with the Close price (-0.79), possibly reflecting long-term trends or market cycles over the years. Day of the week, month, and quarter show low correlations with other variables, indicating they might not be as significant in isolation for predicting bond prices. Technical indicators such as RSI, MACD, and Bollinger Bands show varying degrees of correlation with the Close price. For example, RSI has a moderate positive correlation with the Close price (0.68), highlighting its potential usefulness in capturing price momentum. Understanding these correlations helps in selecting relevant features for the forecasting models, ensuring that they capture the key drivers influencing high-yield bond prices. By incorporating variables with significant correlations, we can improve the models'

ability to predict future bond prices accurately.

### 4.4.4 Distribution Analysis

Understanding the distribution of key variables helps in identifying any skewness or kurtosis, which might affect the model performance.




(a) Distribution of Close Price

(b) Distribution of Daily Returns

Figure 4.3: Distribution of Close Price and Daily Returns

#### Close Price Distribution

Figure 4.3(a) displays the distribution of the closing prices of the SPDR Bloomberg High Yield Bond ETF over the analysed period. The histogram reveals that the closing prices of the high-yield bond ETF are not uniformly distributed. Key observations include central tendency, skewness, and volatility. The majority of closing prices are clustered around the $105 to $110 range, indicating this as the central tendency of the dataset. The distribution shows a slight right skew, with fewer instances of very high closing prices. This suggests that while most prices fall within a certain range, there are occasional higher values. The spread of the histogram indicates periods of varying volatility, which is crucial for understanding the risk and return profile of the high-yield bond ETF. This distribution analysis helps in understanding the typical range and variability of the closing prices, which is essential for setting realistic expectations and developing effective forecasting models.

#### Daily Returns Distribution

Figure 4.3(b) presents the distribution of the daily returns for the SPDR Bloomberg High Yield Bond ETF over the analysed period. The histogram of daily returns provides

insights into the distribution and variability of the ETF's performance on a day-to-day basis. Key observations include central tendency, distribution shape, volatility, and outliers. The mean daily return is approximately $-0.0001\%$, indicating a slight negative bias over the analysed period. The distribution is centered around zero, with a relatively symmetric shape, suggesting that the returns are fairly normally distributed. This is typical for financial return data, which often follows a normal distribution pattern. The majority of the daily returns fall within a narrow range close to zero, with fewer extreme positive or negative returns. This indicates moderate volatility, with most daily price changes being relatively small. There are occasional outliers on both ends of the distribution, representing days with unusually high gains or losses. These outliers could be due to significant market events or economic announcements. Understanding the distribution of daily returns is crucial for risk management and forecasting. By analysing the variability and central tendency of returns, investors can better assess the risk associated with the high-yield bond ETF and develop strategies to mitigate potential losses.

### 4.4.5 Seasonal Decomposition of Close Price

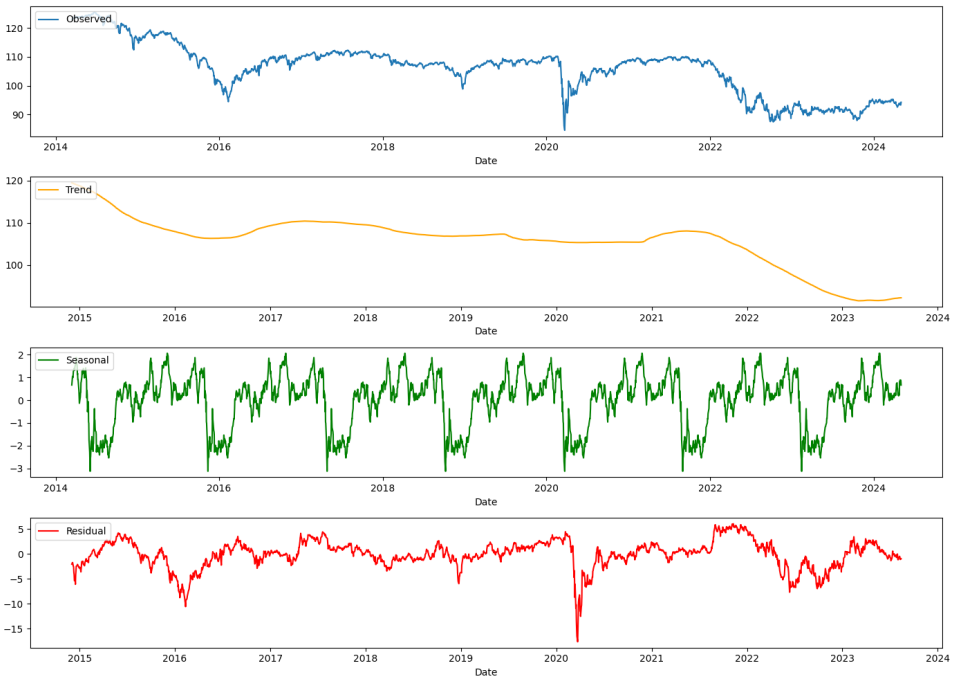



Figure 4.4: Seasonal Decomposition of Close Price

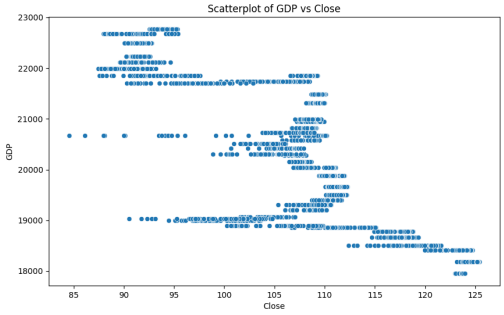
The figure presents the seasonal decomposition of the closing price for the SPDR Bloomberg High Yield Bond ETF. The decomposition breaks down the time series into four components: observed, trend, seasonal, and residual. The seasonal decomposition plot provides valuable insights into the underlying patterns in the closing prices over time. The decomposition separates the time series into observed, trend, seasonal, and residual components. The observed component represents the original time series data of the closing prices. The trend component shows the long-term movement in the data, and in this case, it shows a general decline over time with some fluctuations. The seasonal component reveals periodic fluctuations that repeat within a consistent interval, indicating cyclical patterns in the closing prices. The residual component captures irregularities and noise in the data, highlighting the variability not explained by the trend and seasonal patterns. Analysing these components separately helps in understanding the different factors influencing the ETF's closing prices. The trend component provides insights into the long-term performance, while the seasonal component highlights periodic behaviors. The residual component can be useful for identifying anomalies or unusual events affecting the closing prices.

### 4.4.6 Macroeconomic Indicators Analysis

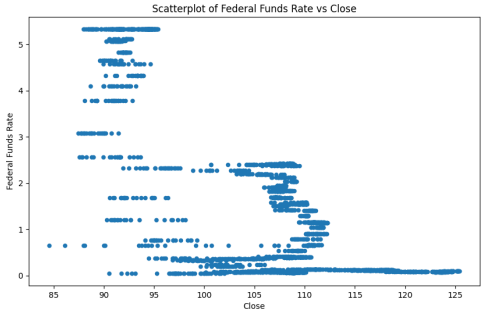
Analysing the macroeconomic indicators in relation to the Close Price provides insights into their influence on bond prices.

#### GDP vs. Close Price

Figure 4.5(a) presents the scatterplot of the closing price for the SPDR Bloomberg High Yield Bond ETF against the Gross Domestic Product (GDP) in billion dollars. The visualisation illustrates the relationship between the closing price of the ETF and the GDP. The plot indicates a general trend where specific ranges of GDP correspond to varying closing prices, but rather than a clear linear correlation, the plot shows distinct clusters of data points. These clusters may reflect different economic periods or varying market conditions, indicating that similar GDP levels can be associated with different closing prices of the ETF. The distribution of points in the scatterplot suggests that while GDP does influence the closing price of the ETF, the relationship is not straightforward. For example, higher GDP levels do not always result in proportionately higher closing

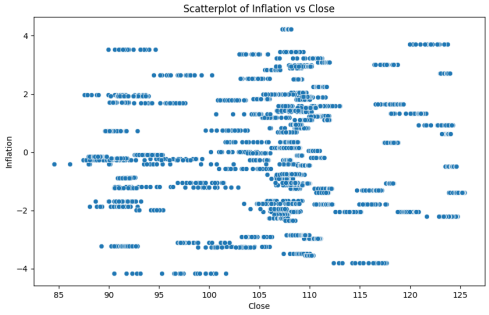



(a) Scatterplot of GDP vs Close




(b) Scatterplot of Federal Funds Rate vs Clsoe


(c) Scatterplot of Unemployment Rate vs Close




(d) Scatterplot of Inflation vs Close

Figure 4.5: Scatterplots of Close Price vs macroecomic indicators

prices, as the data points spread across a range of closing prices for the same GDP values. This could imply that other factors, such as market sentiment or specific economic events, also play significant roles in determining the ETF's closing price. Analysing this scatterplot helps to understand the complex interactions between macroeconomic indicators like GDP and the market performance of financial assets such as the SPDR Bloomberg High Yield Bond ETF. The clusters and spread of the data points provide insights into how different economic conditions can impact the market, highlighting the non-linear and multifaceted nature of financial markets.

**Federal Funds Rate vs. Close Price**

Figure 4.5(b) presents the scatterplot of the closing price for the SPDR Bloomberg High Yield Bond ETF against the Federal Funds Rate. The visualisation indicates a distinct pattern, where the relationship between the Federal Funds Rate and the closing price appears to be more dispersed than a simple linear correlation. At lower Federal Funds Rates (between 0 and 2%), there is a wide range of closing prices, showing that while

interest rates were low, the bond experienced significant fluctuations in price. As the Federal Funds Rate increases above 2%, the data points become more clustered and less varied, especially at Federal Funds Rate levels above 4%. This suggests that when the Federal Funds Rate is higher, the closing price tends to stabilise, albeit at a slightly lower range. The plot does not show a clear negative correlation but suggests more complexity in the relationship. The lack of a downward trend at higher rates implies that other market dynamics may be contributing to the closing prices during periods of higher Federal Funds Rates. Additionally, the presence of clusters at various Federal Funds Rate levels indicates periods of stability in monetary policy, where changes in other macroeconomic indicators might have influenced market performance more significantly than changes in interest rates alone.Market responses to Federal Funds Rate changes depend on the economic environment.

**Unemployment Rate vs. Close Price**

Figure4.5(c) presents the scatterplot of the closing price for the SPDR Bloomberg High Yield Bond ETF against the Unemployment Rate. The visualisation illustrates the relationship between the closing price of the ETF and the unemployment rate. The plot shows that higher unemployment rates generally correspond to lower closing prices, suggesting an inverse relationship between these variables. Specifically, during periods of high unemployment, the closing prices tend to be lower, likely due to reduced economic activity and decreased investor confidence. Conversely, lower unemployment rates are associated with higher closing prices, indicating more robust economic conditions and increased investor confidence. Interestingly, the scatterplot also reveals that at low levels of unemployment, there is a wide range of closing prices between 85 and 105. This indicates that even when unemployment is low, there can still be significant variation in the ETF's closing price, possibly due to other economic factors or market conditions influencing investor behavior during these periods. This pattern underscores the sensitivity of financial markets to labor market conditions, where fluctuations in the unemployment rate can significantly impact asset prices. The spread of data points at different unemployment rate levels also offers insights into how variations in employment conditions influence market performance over time, reflecting the complex and dynamic nature of the relationship between unemployment and market behavior.

#### Inflation vs. Close Price

Figure 4.5(d) presents the scatterplot of the closing price for the SPDR Bloomberg High Yield Bond ETF against the inflation rate. The visualisation illustrates the relationship between the closing price of the ETF and the inflation rate. The data points do not indicate a clear, linear relationship between inflation and the closing prices. Instead, the closing prices are distributed across a range of inflation rates, showing that inflation alone may not be a significant predictor of the ETF's closing price. However, clusters of data points suggest that at certain levels of inflation, there can be varying closing prices, indicating that other economic factors might be at play in conjunction with inflation. This dispersion highlights the complexity of market dynamics where inflation interacts with other macroeconomic variables to influence asset prices. Investors and analysts should consider multiple indicators alongside inflation to better understand market trends and make informed decisions.

The EDA provides a comprehensive understanding of the dataset, highlighting key trends, relationships, and potential predictors for high-yield bond prices. The insights gained from this analysis guide the feature selection and model development process, ensuring that the forecasting models are built on a solid foundation of well-understood data. By leveraging the identified patterns and relationships, we can improve the accuracy and robustness of our predictive models, ultimately providing valuable insights for investors and financial institutions.

## 4.5 Model Implementation and Performance Analysis

### 4.5.1 Model Implentation

In this section, we detail the implementation of Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU), and Gradient Boosting Regressor (GBR) models, as well as the stacking method combining these models for high-yield bond forecasting.

#### Training and Validation

The training and validation process involves the following key steps. First, the dataset is split into training and testing sets using `TimeSeriesSplit`, which preserves the

temporal ordering of the data. This approach ensures that future data points are not used to predict past data points, which is crucial for time series forecasting. Additionally, we use a 5-fold cross-validation strategy to evaluate model performance. This method provides a robust estimate of model accuracy by training the model on multiple subsets of the data and validating it on the remaining data.

Next, the feature preparation involves setting the target variable as the closing price (`Close`), with features including various technical indicators and macroeconomic variables. Time series data is prepared using `TimeseriesGenerator`, which is a library that helps in creating sequences of data points that can be used to train LSTM and GRU models. For the GBR model, features are scaled using `StandardScaler`.

Finally, each model is trained on the training set and validated on the test set for each fold. The performance metrics are averaged across all folds to provide an overall evaluation of each model.

#### Hyperparameter Tuning

Hyperparameter tuning is essential for optimising model performance. Here are the key hyperparameters and the tuning approach for each model:

- **LSTM and GRU**: First, the number of units in the hidden layer is tuned, with 50 units being a typical starting point. This choice balances model complexity and training time. The `ReLU` activation function is used due to its effectiveness in handling non-linearities, which are common in time series data. Next, the batch size is set to `1`, a common practice for time series data to maintain temporal dependencies between samples.The number of epochs, which refers to the number of times the entire training dataset is passed through the model, is initially set to `20`. This provides a good starting point for training duration, allowing the model enough time to learn patterns without overfitting or underfitting. However, this parameter can be adjusted based on model performance as training progresses. The `Mean Squared Error (MSE)` is chosen as the loss function, which is commonly used for regression tasks. It measures the average squared difference between the actual and predicted values, helping the model focus on minimising large errors during training. Finally, the `Adam` optimiser is chosen for its adaptive learning rate, which helps in efficiently navigating the optimisation

landscape, making it a popular choice for training deep learning models.

```
def build_gru_model(input_shape):
model = Sequential([
    RU(50, activation='relu', input_shape=input_shape),
    Dense(1)
    ])
model.compile(optimizer='adam', loss='mse')
return model
```

**GBR**: The hyperparameter tuning for the Gradient Boosting Regressor involves several important choices. The number of trees typically ranges from 100 to 500, balancing the trade-off between model complexity and computational efficiency. The learning rate is adjusted to prevent overfitting, with typical values between 0.01 and 0.1, allowing the model to learn at a controlled pace. The maximum depth of the individual trees is usually set between 3 and 5, which helps in managing the model's complexity and avoiding overfitting. Additionally, subsampling is set to a fraction of 0.8, meaning 80% of the samples are used for fitting each individual base learner, enhancing the model's generalisation by introducing randomness.

```
def build_gbm_model():
model = GradientBoostingRegressor(
    n_estimators=300,
    learning_rate=0.05,
    max_depth=4,
    subsample=0.8
)
return model
```

**Stacked Model (Meta LSTM)**: The meta-learner is another LSTM model that combines predictions from the base learners (GRU and GBR). The input shape consists of two features, which are the predictions from GRU and GBR, allowing the meta-learner to leverage the strengths of both models. The number of units in the hidden layer is tuned similarly to the base LSTM model, ensuring adequate capacity to capture complex patterns. The training process typically involves setting the number of epochs

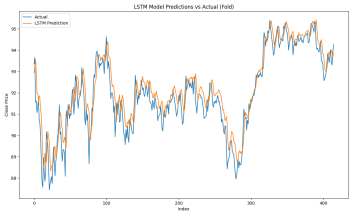
to 20 initially, providing a balance between sufficient training time and computational efficiency.

```
def build_meta_lstm_model(input_shape):
model = Sequential([
        LSTM(50, activation='relu', input_shape=input_shape),
        Dense(1)
    ])
model.compile(optimizer='adam', loss='mse')
return model
```

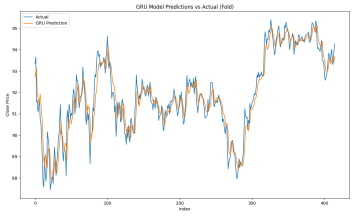
### 4.5.2 Performance Analysis and Interpretation

Table 4.2: Cross-Validation Results (mean values)

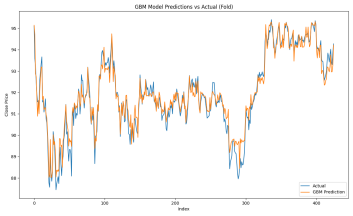
| Model | $R^2$ | Adjusted $R^2$ | RMSE | MSE | MAE | MAPE |
|---|---|---|---|---|---|---|
| LSTM | 0.902919 | 0.900516 | 1.016963 | 1.339276 | 0.664036 | 0.657577 |
| GRU | 0.943882 | 0.943196 | 0.653220 | 0.787881 | 0.515501 | 0.501455 |
| GBR | 0.872789 | 0.864819 | 1.043601 | 2.082041 | 0.509212 | 0.516126 |
| **Meta LSTM** | **0.970687** | **0.970548** | **0.540346** | **0.506642** | **0.327142** | **0.322686** |



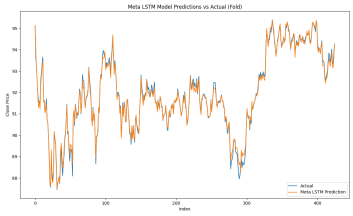
(a) LSTM Model Performance



(b) GRU Model Performance



(c) GBR Model Performance



(d) Meta LSTM Model Performance

Figure 4.6: Models Performance

To evaluate the performance of the different models, we used several metrics, including $R^2$, Adjusted $R^2$, RMSE, MSE, MAE, and MAPE.The table 4.2 summarises the mean

values of these metrics obtained through cross-validation for each model. The table 4.2 shows that the Meta LSTM model demonstrates the highest predictive accuracy among all models, as evidenced by its superior performance across all evaluation metrics. This suggests that combining the strengths of individual models (LSTM, GRU, and GBR) in a stacked architecture enhances the overall predictive capability.

- **LSTM**: The LSTM model achieves a good balance across all metrics with a strong $R^2$ and Adjusted $R^2$. However, it has higher RMSE and MAE compared to GRU and Meta LSTM, indicating slightly less accuracy in predictions.
- **GRU**: The GRU model shows superior performance in terms of $R^2$ and Adjusted $R^2$ compared to LSTM and GBR. It has lower RMSE, MSE, and MAE, indicating higher accuracy. Nevertheless, it still falls short of the Meta LSTM's performance.
- **GBR**: The GBR model exhibits good performance with relatively low MAE and MAPE. However, it has lower $R^2$ and Adjusted $R^2$ compared to LSTM and GRU, and higher RMSE and MSE indicate less accurate predictions.
- **Meta LSTM**: The Meta LSTM model outperforms all other models across all metrics. It has the highest $R^2$ and Adjusted $R^2$ values, indicating the best fit, and the lowest RMSE, MSE, MAE, and MAPE, demonstrating the highest accuracy and precision in predictions. Despite its superior performance, the complexity of combining multiple models might make it computationally intensive.

The Meta LSTM model demonstrates the highest predictive accuracy among all models, as evidenced by its superior performance across all evaluation metrics. This suggests that combining the strengths of individual models (LSTM, GRU, and GBR) in a stacked architecture enhances the overall predictive capability. The high $R^2$ and Adjusted $R^2$ values across all models indicate that the chosen features and models effectively capture the variance in high-yield bond prices. Furthermore, the lower RMSE and MAE values for GRU and Meta LSTM highlight their ability to produce more precise predictions, which is crucial for financial forecasting where even small errors can have significant impacts.

The findings from this study have significant practical implications for investors and financial institutions. Each model LSTM, GRU, GBR, and Meta LSTM offers unique strengths and potential applications in forecasting high-yield bond prices.

The LSTM model, with its ability to capture long-term dependencies, can be particularly useful for investors looking to understand and predict trends in bond prices over extended periods. Its good balance across all metrics suggests it can be a reliable tool for making informed investment decisions and managing risks.

The GRU model, which shows superior performance in terms of $R^2$ and Adjusted $R^2$, offers higher accuracy with lower RMSE and MAE values. This makes it well-suited for financial forecasting where precise predictions are crucial. Financial institutions can leverage the GRU model to enhance their trading strategies and optimise their portfolios, ensuring better returns and reduced risks.

The GBR model, despite having lower $R^2$ and Adjusted $R^2$ compared to LSTM and GRU, exhibits relatively low MAE and MAPE. This indicates its utility in scenarios where minimising absolute errors is more critical than explaining variance. Asset managers and financial analysts can use the GBR model for specific tasks that require robust error minimisation and resilience to outliers.

The Meta LSTM model demonstrates the highest predictive accuracy among all models, suggesting that combining the strengths of individual models in a stacked architecture enhances the overall predictive capability. This model can be particularly beneficial for comprehensive risk assessment frameworks and sophisticated trading systems, providing the highest precision in predictions.

The potential applications of these findings are vast. Asset managers can integrate these models into their trading systems to forecast bond prices with greater accuracy, thereby maximising returns and minimising risks. Financial analysts can use these models to gain deeper insights into market trends and identify profitable investment opportunities. Credit rating agencies can also benefit by incorporating these advanced forecasting techniques into their credit risk models, improving the accuracy of their ratings and assessments.

However, there are limitations to consider. The complexity of the Meta LSTM model, which combines multiple models, may lead to higher computational costs and longer training times, making it less feasible for real-time applications. Additionally, while the models perform well on historical data, their accuracy may be affected by sudden market changes or unforeseen economic events. The reliance on specific features and macroeconomic indicators means that any changes in the underlying data quality or

availability could impact the models' performance. Furthermore, the LSTM and GRU models, while effective, may require significant computational resources for training due to their recurrent nature.

In conclusion, while the advanced models developed in this study offer significant benefits for forecasting high-yield bond prices, it is important to consider their computational requirements and potential vulnerabilities to market volatility and data quality issues. By addressing these limitations and continuously refining the models, investors and financial institutions can harness their full potential for better financial decision-making.

CHAPTER 5

# Conclusion

This dissertation has demonstrated the significant potential of advanced data science, machine learning, and deep learning models in forecasting high-yield bond prices. The empirical analyses revealed that models such as LSTM, GRU, and especially the Meta LSTM, significantly outperform traditional econometric models by capturing nonlinear relationships and interactions within the data. The use of stacked generalisation in the Meta LSTM model further enhances the predictive performance by combining the strengths of multiple learners, making it the most effective model tested.

The practical implications of these findings are profound. Financial institutions can leverage these models to enhance their trading strategies, optimise portfolios, and improve risk management. For instance, these models can be integrated into decision-making systems to provide more accurate forecasts, thereby reducing financial risks associated with volatile high-yield bonds. Asset managers and financial analysts can utilise these tools to make more informed investment decisions, while credit rating agencies can achieve greater accuracy in assessing credit risk, especially in the context of bonds with significant default risk.

However, it is crucial to acknowledge the limitations and challenges identified in this research. The computational demands of complex models like the Meta LSTM and their sensitivity to data quality highlight the need for ongoing refinement and adaptation. Additionally, the reliance on historical data means that these models may struggle to predict bond prices during periods of unprecedented market turmoil or macroeconomic shifts. Future research should focus on integrating alternative financial indicators, such as sentiment analysis from news sources or social media, to improve robustness against

such disruptions.

Furthermore, the potential for these models to be applied in real-time financial markets requires attention to their computational efficiency. Techniques such as parallel processing or the use of more lightweight models in real-time applications can address these concerns. As machine learning and artificial intelligence continue to evolve, the development of more sophisticated AI techniques will open new avenues for improving the accuracy and applicability of these models in both high-yield bond markets and broader financial contexts.

By addressing these challenges and continuously refining the models, the financial industry can fully harness the capabilities of these advanced forecasting models, leading to more robust and accurate financial predictions. This will ultimately contribute to better financial decision-making, risk management, and stability in the high-yield bond market, positioning financial institutions to better navigate the complexities of modern financial systems.

# Bibliography


[1] Bishop, C. M., 2006. Pattern Recognition and Machine Learning. *Springer*. [Accessed 25 May 2024].

[2] Bontempi, G., Taieb, S. B., & Le Borgne, Y. A., 2012. Machine Learning Strategies for Time Series Forecasting. *European Business Intelligence Summer School*, pp.62-77. [Accessed 31 May 2024].

[3] Breiman, L., 2001. Random Forests. *Machine Learning*, 45(1), pp.5-32. [Accessed 2 June 2024].

[4] Brookings, 2020. Corporate bond market dysfunction during COVID-19 and lessons from the Fed's response. *Brookings*. Available at: `https://www.brookings.edu/research/` [Accessed 14 August 2024].

[5] Chen, J., Xu, Y., & Zhang, X., 2018. Financial Market Forecasting using RNN, LSTM, BiLSTM, GRU and Transformer-Based Deep Learning Algorithms. *ResearchGate*. Available at: `https://www.researchgate.net/publication/377245794_Financial_Market_Forecasting_using_RNN_LSTM_BiLSTM_GRU_and_Transformer-Based_Deep_Learning_Algorithms` [Accessed 8 July 2024].

[6] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y., 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv*. Available at: `https://doi.org/10.48550/arXiv.1412.3555` [Accessed 11 July 2024].

[7] DataScientest, 2023. Long Short Term Memory (LSTM): de quoi s'agit-il?. *DataScientest*. Available at: `https://datascientest.com/long-short-term-memory-tout-savoir` [Accessed 14 July 2024].

[8] Expert Market Research, 2024. High Yield Bonds Market Report and Forecast 2023-2028. *Expert Market Research*. Available at: https://www.expertmarketresearch.com/reports/high-yield-bonds-market [Accessed 17 July 2024].

[9] Federal Reserve Economic Data (FRED), 2024. Federal Funds Rate. *FRED*. Available at: https://fred.stlouisfed.org [Accessed 20 July 2024].

[10] Federal Reserve Economic Data (FRED), 2024. Gross Domestic Product (GDP). *FRED*. Available at: https://fred.stlouisfed.org [Accessed 23 July 2024].

[11] Federal Reserve, 2020. The Corporate Bond Market Crises and the Government Response. *Federal Reserve*. Available at: https://www.federalreserve.gov/econres/notes/feds-notes/ [Accessed 14 August 2024].

[12] Federal Reserve Economic Data (FRED), 2024. Unemployment Rate. *FRED*. Available at: https://fred.stlouisfed.org [Accessed 26 July 2024].

[13] Friedman, J. H., 2001. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics*, 29(5), pp.1189-1232. [Accessed 6 June 2024].

[14] GeeksforGeeks, 2024. Supervised Learning Diagram. *GeeksforGeeks*. Available at: https://media.geeksforgeeks.org/wp-content/uploads/20231121154747/Supervised-learning.png [Accessed 14 August 2024].

[15] GeeksforGeeks, 2024. Unsupervised Learning Diagram. *GeeksforGeeks*. Available at: https://media.geeksforgeeks.org/wp-content/uploads/20231124111325/Unsupervised-learning.png [Accessed 14 August 2024].

[16] Goodfellow, I., Bengio, Y., & Courville, A., 2016. Deep Learning. *MIT Press*. [Accessed 8 June 2024].

[17] Medium, 2024. GRU Architecture Diagram. *Medium*. Available at: https://miro.medium.com/v2/resize:fit:423/1*i-yqUwAYTo2Mz-P1Ql6MbA.png [Accessed 14 August 2024].

[18] Gündüz, H., Cataltepe, Z., & Atalay, R., 2015. Forecasting Government Bond Yields with Neural Networks Considering Cointegration. *ResearchGate*. Available at: https://www.researchgate.net/publication/284559677_Forecasting_Government_Bond_Yields_with_Neural_Networks_Considering_Cointegration [Accessed 29 July 2024].

[19] Hochreiter, S., & Schmidhuber, J., 1997. Long Short-Term Memory. *Neural Computation*, 9(8), pp.1735-1780. [Accessed 5 June 2024].

[20] Hyndman, R. J., & Athanasopoulos, G., 2018. Forecasting: Principles and Practice. *OTexts*. [Accessed 9 June 2024].

[21] Investopedia, 2024. Bollinger Bands. *Investopedia*. Available at: https://www.investopedia.com/terms/b/bollingerbands.asp#:~:text=Bollinger%20Bands%2C%20a%20popular%20tool,that%20move%20with%20the%20price [Accessed 13 August 2024].

[22] Investopedia, 2023. How Interest Rates Affect the U.S. Markets. *Investopedia*. Available at: https://www.investopedia.com/how-interest-rates-affect-the-u-s-markets-5189285 [Accessed 29 July 2024].

[23] Investopedia, 2023. Fed Rate Hikes Have Taken a Toll on Bond Market ETFs. *Investopedia*. Available at: https://www.investopedia.com/fed-rate-hikes-have-taken-a-toll-on-bond-market-etfs-4587686 [Accessed 29 July 2024].

[24] Investopedia, 2024. Moving Average Convergence/Divergence (MACD). *Investopedia*. Available at: https://www.investopedia.com/terms/m/macd.asp#:~:text=Key%20Takeaways-,Moving%20average%20convergence%2Fdivergence%20(MACD)%20is%20a%20technical%20indicator,EMA%20of%20the%20MACD%20line. [Accessed 13 August 2024].

[25] Investopedia, 2024. Relative Strength Index (RSI). *Investopedia*. Available at: https://www.investopedia.com/terms/r/rsi.asp [Accessed 13 August 2024].

[26] Kingma, D. P., & Ba, J., 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*. [Accessed 11 June 2024].

[27] La Revue IA, 2023. Qu'est-ce qu'un réseau LSTM?. *La Revue IA*. Available at: https://larevueia.fr/quest-ce-quun-reseau-lstm/ [Accessed 1 August 2024].

[28] Li, Y., Wang, S., & Zhang, S., 2019. Regularised Gradient Boosting for Financial Time-series Modelling. *ResearchGate*. Available at: https://www.researchgate.net/publication/317345782_Regularised_Gradient_Boosting_for_Financial_Time-series_Modelling [Accessed 3 August 2024].

[29] Thorir Mar, 2024. LSTM Architecture Diagram. *Thorir Mar's Blog*. Available at: https://thorirmar.com/post/insight_into_lstm/featured.png [Accessed 14 August 2024].

[30] Martin, D., Poczos, B., & Hollifield, B., 2018. Machine Learning-aided Modeling of Fixed Income Instruments. *Machine Learning Department, Carnegie Mellon University*. Available at: https://www.ml.cmu.edu/research/dap-papers/f18/dap-martin-daniel.pdf [Accessed 12 May 2024].

[31] Ndikum, P., 2020. Machine Learning Algorithms for Financial Asset Price Forecasting. *arXiv preprint arXiv:2004.01504*. Available at: https://arxiv.org/pdf/2004.01504 [Accessed 9 May 2024].

[32] Neptune.ai, 2024. Vanishing and Exploding Gradients: Debugging, Monitoring, and Fixing. *Neptune.ai*. Available at: https://neptune.ai/blog/vanishing-and-exploding-gradients-debugging-monitoring-fixing [Accessed 14 August 2024].

[33] Neural Network Diagram, 2024. Neural Network Basics. *Educational Resource on Machine Learning*. Available at: https://www.sciencelearn.org.nz/images/5156-neural-network-diagram [Accessed 14 August 2024].

[34] Nunes, M., Gerding, E., McGroarty, F., & Niranjan, M., 2018. Artificial Neural Networks in Fixed Income Markets for Yield Curve Forecasting. *SSRN*. Avail-

able at: https://www.researchgate.net/publication/324069852_Artificial_Neural_Networks_in_Fixed_Income_Markets_for_Yield_Curve_Forecasting [Accessed 15 May 2024].

[35] OECD, 2024. Consumer Price Index (CPI) Data. *OECD Statistics*. Available at: https://stats.oecd.org/modalexports.aspx?exporttype=bulk&FirstDataPointIndexPerPage=undefined&SubSessionId=e1308c33-c8df-4de8-b0b2-9a2b708704cf&Random=0.2606399033611029 [Accessed 13 August 2024].

[36] Philadelphia Fed, 2021. When COVID-19 Reached the Corporate Bond Market. *Federal Reserve Bank of Philadelphia*. Available at: https://www.philadelphiafed.org/the-economy/banking-and-financial-markets/when-covid-19-reached-the-corporate-bond-market [Accessed 14 August 2024].

[37] PIMCO, 2024. Understanding High Yield Bonds. *PIMCO*. Available at: https://www.pimco.com/gbl/en/resources/education/understanding-high-yield-bonds [Accessed 30 July 2024].

[38] Medium, 2024. Recurrent Neural Network (RNN) Architecture Diagram. *Medium*. Available at: https://miro.medium.com/v2/resize:fit:1200/1*EeRwi4cBjHyBuhmoR7IGYw.png [Accessed 14 August 2024].

[39] S&P Global, 2024. Daily Update: February 15, 2024. *S&P Global*. Available at: https://www.spglobal.com/en/research-insights/market-insights/daily-update-february-15-2024 [Accessed 29 July 2024].

[40] Standard and Poors Global, 2024. Global Bond Market Report. *Standard and Poors*. Available at: https://www.spglobal.com [Accessed 29 July 2024].

[41] ResearchGate, 2024. Stacked Ensemble Regressor Structure. *ResearchGate*. Available at: https://www.researchgate.net/publication/364267912_Structural_Ensemble_Regression_for_Cluster-Based_Aggregate_Electricity_Demand_Forecasting [Accessed 14 August 2024].

[42] St. Louis Fed, 2020. COVID-19, Monetary Policy and the Corporate Bond Market. *Federal Reserve Bank of St. Louis*. Available at: https://www.stlouisfed.org/on-the-economy/2020/april/covid-19-monetary-policy-corporate-bond-market [Accessed 14 August 2024].

[43] TechVidvan, 2024. Reinforcement Learning in Machine Learning. *TechVidvan*. Available at: https://editor.analyticsvidhya.com/uploads/981063.jpg [Accessed 14 August 2024].

[44] Tibshirani, R., 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), pp.267-288. [Accessed 6 July 2024].

[45] Wang, X., Chen, H., Li, Y., & Zhang, Z., 2023. Research on Improved GRU-Based Stock Price Prediction Method. *Applied Sciences*, 13(15), p.8813. Available at: https://www.mdpi.com/2076-3417/13/15/8813 [Accessed 2 August 2024].

[46] Wolpert, D. H., 1992. Stacked Generalization. *Neural Networks*, 5(2), pp.241–259. Available at: https://doi.org/10.1016/S0893-6080(05)80023-1 [Accessed 7 July 2024].

[47] Yahoo Finance, 2024. SPDR Bloomberg High Yield Bond ETF. *Yahoo Finance*. Available at: https://finance.yahoo.com [Accessed 3 August 2024].

[48] Zhang, G., Patuwo, B. E., & Hu, M. Y., 1998. Forecasting with Artificial Neural Networks: The State of the Art. *International Journal of Forecasting*, 14(1), pp.35-62. [Accessed 6 June 2024].