

CHAPTER 3: Proposed Methodology

3.1 Introduction

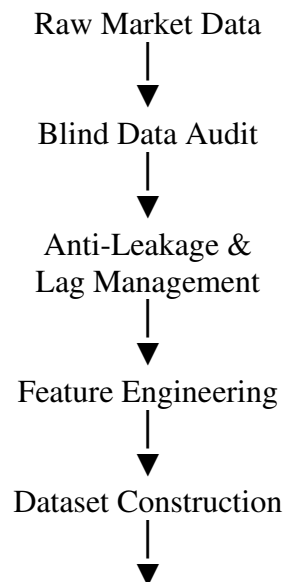
The chapter that follows will present the complete methodology that will be used for the implementation of the artificial intelligence framework that is to be proposed within this dissertation. The framework will include the steps that will be performed to develop the artificial intelligence models for the prediction of financial markets for the Saudi Stock Exchange (Tadawul). This methodology will differ from that which is often used for training of artificial intelligence models in that it will include steps for evaluating the performance of those models with the intention of informing investors of its potential success.

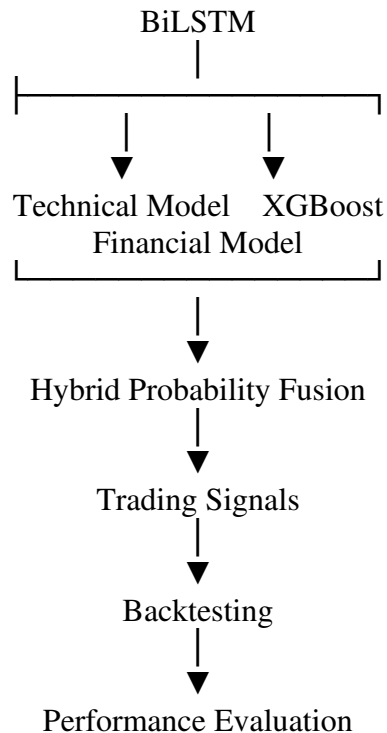
The methodology that will be developed for the artificial intelligence framework for the Saudi Stock Exchange will include nine different phases for the development and evaluation of artificial intelligence models for the market's predictions. The nine phases are as follows:

1. Blind Data Audit
2. Anti-Leakage and Lag Management
3. Feature Engineering
4. Dataset Preparation
5. Model Development
6. Hybrid Probability Fusion
7. Portfolio Backtesting
8. Explainable Artificial Intelligence
9. Experimental Evaluation

Figure 3.1 represents the architecture for the proposed framework for artificial intelligence models for prediction of the Saudi Stock Exchange.

Figure 3.1 Overall Methodology





3.2 Phase 1 - Blind Data Audit

The first phase of the methodology consists of performing a blind data audit of the collected datasets prior to performing any preprocessing steps. Before financial information can be utilized by machine learning algorithms, it is necessary to ensure its integrity. Such a process can reveal issues with the data that may impact its use in modeling, such as missing values, duplicated data, invalid formats, and issues related to the companies represented by the data.

The goal of this audit is to establish a reliable data baseline prior to the application of any preprocessing steps to that data.

Furthermore, it is essential that this audit is performed prior to any data cleaning procedures. Performing the audit on raw data will reveal any issues with the data prior to any attempts to alter that data.

The blind data audit will consist of five steps:

- Dynamic dataset exploration
- Data integrity assessment
- Company identifier validation
- Temporal consistency verification

- Baseline data retention assessment

3.2.1 Dynamic Dataset Exploration

The first validation step examines the structural characteristics of both datasets. Each of the datasets was collected from different sources. Daily market data was collected from the financial website Yahoo Finance. Financial statement data was collected separately, and is integrated into the main dataset according to the chronological order of the financial statements. The complete dataset was then validated prior to preprocessing.

The market data contained 47,110 daily market observations and nine different variables associated with each market observation. The financial data contained 237 financial statement records, each containing 237 different financial variables. Each of these variables was examined prior to performing any additional analysis on the data.

The market data contained mostly numerical variables related to market and trading activity. The financial data contained numerous financial variables related to financial statements of the analyzed companies.

The dynamic exploration of the data helped to establish the characteristics of each of the datasets, and to confirm that they are suitable for the preprocessing steps that will be applied to those datasets.

Table 3.1 Data Sources Used in This Research

Source	Data Collected	Usage in This Research
Yahoo Finance	Daily OHLCV market data	Technical analysis and historical price series
Saudi Exchange (Tadawul)	Company listings and market information	Company selection and market validation
Quarterly Financial Statements	Balance sheet, income statement, cash flow statement	Financial feature engineering
Capital Market Authority	Market regulations	Regulatory background

3.2.2 Data Integrity Assessment

Following the completion of the structural assessment of the two main datasets, an extensive audit was performed to determine the completeness of the data as well as to identify any abnormal values represented by numbers within the data.

The integrity of the data was assessed for the presence of:

- Missing values

- Infinite values within the numerical variables
- Completeness of each variable within the dataset
- Completeness of the entire dataset

As a result of this assessment, no abnormal infinite values were identified within the financial variables. Furthermore, statistics regarding the number of missing values for each feature within the dataset were calculated.

Rather than removing any examples from the dataset that had missing values for these features, the assessment of the integrity of the data allowed for the determination of the quality of the financial information available for each company during the timeframe represented by the dataset.

3.2.3 Ticker Validation

In order to ensure that the data from each of the companies represented within the market and financial datasets were consistent, each of the company identifiers were validated against a whitelist of those companies.

Each of the company identifiers that were discovered within the datasets were compared against the list of the companies that were to be included within the study.

The validation performed ensured that the company datasets included:

- All expected companies
- All identifiers for each company that were missing from the datasets
- No companies that were identified outside of those within the whitelist
- That each company used the same ticker symbols over time within their financial markets

In addition to validating all of the companies against the whitelist, any instances of duplicates within each of the datasets were also examined to ensure that their inclusion would not impact the creation of the features for the machine learning models.

This process of automatic validation of all company identifiers ensures that each company within the market and financial datasets maintain their company identifier throughout the complete engineering process.

3.2.4 Temporal Validation

As the financial information that is used in the construction of the models is both daily market information as well as quarterly financial information for those companies, it was essential to ensure that the temporal data for each dataset was validated.

All dates from both datasets were standardized prior to integration into a unified dataset.

Temporal validation of the two datasets ensured that each of the datasets was validated for:

- Invalid timestamps
- Time period represented by market data
- Time period represented by financial data
- Duplicates within the quarterly financial reports

The market data was standardized and included time periods between January 3rd, 2021 and June 7th, 2026. Furthermore, the financial information covered the time period between September 30th, 2024 and March 31st, 2026.

This validation of the timestamps within each of the datasets created the chronological foundation for the anti-leakage process.

3.2.5 Baseline Data Retention

Prior to any filtering of the datasets, it was essential to determine the initial Data Retention Rate (DRR) for the complete engineering process.

By recording the initial size of the dataset prior to any filtering processes, it becomes possible to determine the impact of each stage within the engineering process upon the availability of data for training the models.

As a result of this baseline data retention rate, it is possible to filter the data at each stage of the engineering process while maintaining visibility into the data retention rates after each phase. This is in contrast to conventional machine learning engineering processes, wherein data is typically removed from the dataset at each phase.

3.3 Phase 2 - Anti-Leakage and Temporal Alignment

One of the most crucial aspects of the creation of the machine learning models is to ensure that the models are trained using only information that would have been available during the formation of the investment decisions. Conventional machine learning datasets typically do not contain time-varying information, and do not incorporate the various information sources that can be combined to create financial datasets with information from different temporal frequencies. Thus, maintaining the chronological integrity of the created datasets is an essential requirement for the development of reliable machine learning models for the determination of investment returns.

The Anti-Leakage and Temporal Alignment stage of the engineering process ensures that each of the daily market observations are synchronized with the financial statements of each company, while eliminating the possibility of incorporating information from future financial reports within the training data for the models. Future financial reports for each company are not published simultaneously with the end of each reporting quarter; each company must take additional time to prepare and publish their financial reports. Thus, any merging of financial information with market data should incorporate a time delay between the end of each financial quarter and the publication of those reports.

This phase of the engineering process ensures that the entire engineering process is constructed in a manner that is chronologically-integrative and realistic regarding the conditions under which investment decisions are to be made.

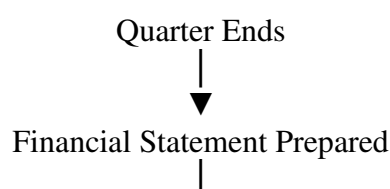
3.3.1 Financial Reporting Delay

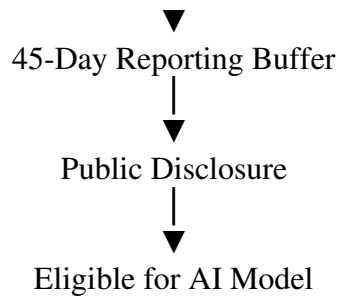
Financial reports for each company are not published immediately after the end of each reporting quarter. Each company must prepare and publish its financial reports during this time to disclose its financial information to its investors.

If the financial reports are merged with market data using the end of each quarter as the time stamp, the machine learning models will have access to financial information that would not have been available during the investment period. This information leakage will artificially enhance the performance of the models.

To eliminate this potential problem, a 45-day buffer period is established between the end of each reporting quarter for companies and the availability of their financial reports. This 45-day time period allows for each company's preparation and publication of its financial reports. This process is illustrated in Figure 3.2.

Figure 3.2 Financial Reporting Timeline





3.3.2 Time-Aware Financial Integration

Following adjustment of the financial reporting dates to incorporate the regulatory reporting buffer, the financial statements were synchronized with daily market observations according to their chronological order.

Whereas financial information is typically published according to calendar quarters, the information for each trading day was associated with the most recent financial statement that had become available to the company for publication.

This process guarantees that the prediction models are provided with information that is accessible to investors on each trading day.

Unlike conventional database joins operations, which typically ignore the time delays for financial report publication, the integration of financial and market information in this approach utilizes a time-aware join strategy that ensures that financial information always precedes market observation of the same company's shares.

3.3.3 Chronological Verification

Following synchronization of the financial and market information, an independent process was performed to verify the chronological integrity of the integrated information.

This verification process automatically inspected each row of the integrated database to determine whether the following conditions were satisfied:

- Financial report availability always preceded trading date;
- No future financial reports were associated with historical market observations;
- Information regarding the company remained consistent throughout the database;
- There were no instances of duplicated financial information for the same company.

This verification process ensured that the resulting database had zero instances of Look-Ahead Bias. This process is one of the main safeguards that is implemented in the methodology for ensuring chronological integrity of the information included in the database.

3.3.4 Data Filtering and Retention Analysis

Following chronological alignment and verification, additional filters were applied to the database to remove any rows of market observations that were not associated with complete financial statements.

Whereas conventional database joins simply remove observations that do not meet the criteria without further analysis, the filtering of the database according to data completeness in this algorithm also includes the monitoring of a statistic known as the Data Retention Rate (DRR).

The goal of implementing a metric to measure Data Retention Rate has two main components:

First, it allows for the measurement of the amount of information that is preserved within the database after each stage of the prediction model development process.

Second, it allows for transparency in the presentation of the impact of data filtering on the dataset that is utilized to train the machine learning models.

The results of this data filtering process indicated that the size of the database was reduced from 13,665 market observations to 11,762 observations after application of the chronological filters and financial information verification. Thus, the Data Retention Rate of the database after this stage was determined to be approximately 86.07%.

By monitoring the DRR throughout the engineering of this database, the repeatability of the engineering process can be ensured, as can the ability of future researchers to evaluate the effectiveness of each preprocessing stage.

Table 3.2 Data Retention During Phase 2

Processing Stage	Remaining Records	DRR (%)
Initial Integrated Dataset	13,665	100.00
After Quality Filtering	11,762	86.07

3.3.5 Value of Anti-Leakage Control

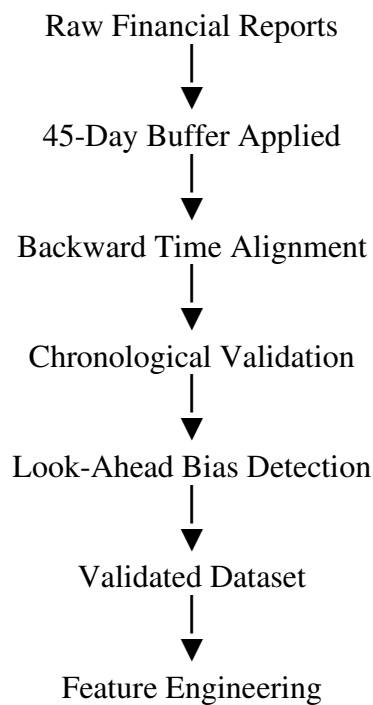
The prevention of temporal information leakage is one of the most valuable methodological concepts in financial artificial intelligence.

An artificial intelligence prediction model trained using future financial information can exhibit extremely high prediction quality during offline validation, but it would perform dismally in real-world investment contexts since the learned behaviours would not be reproducible under realistic trading conditions.

Thus, the proposed Anti-Leakage strategy is better conceptualized not merely as a pre-processing step, but as an essential methodological component that underpins the credibility, reproducibility, and applicability of the proposed prediction methodology.

By maintaining chronological integrity throughout the entire data engineering process, the proposed methodology guarantees that all experiment outcomes reflect realistic investment situations, rather than artificially inflated prediction results.

Figure 3.3 Anti-Leakage Validation Pipeline



3.4 Phase 3 - Feature Engineering

Feature engineering is one of the most influential steps within the proposed hybrid artificial intelligence framework. While prediction algorithms dictate how learning occurs, feature engineering determines what is made available for learning. Thus, the predictive power of artificial intelligence models hinges not only on the algorithms used, but also on the quality, diversity, and representational strength of the features being engineered.

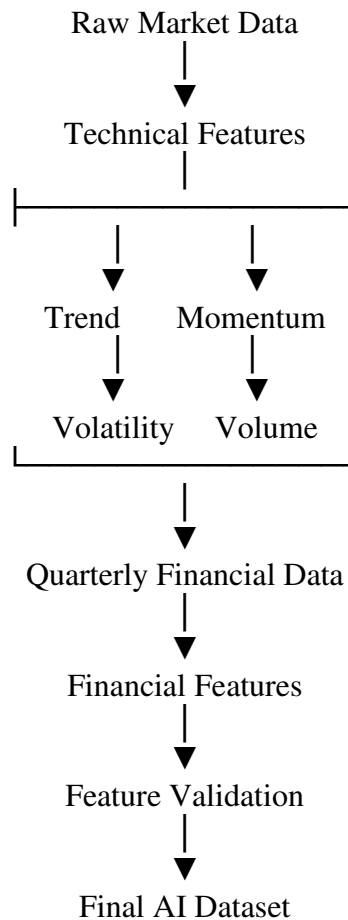
The aim of this phase is to convert financial observations into meaningful numerical features that encapsulate both market behaviour and corporate performance. Instead of using either raw market prices or raw accounting values, the proposed methodology generates a large number of complementary feature groups that describe various aspects of financial behaviour while respecting the chronological order that was established in the previous phase.

The feature engineering pipeline was constructed in accordance with a modular structure, with three main components:

- Technical feature generation.
- Financial feature engineering.
- Feature quality validation.

Each component provides unique prediction information that contributes towards building the input space for the hybrid artificial intelligence framework.

Figure 3.4 shows the overall feature engineering workflow used in this research.



3.4.1 Technical Feature Engineering

The daily financial observations made available in the dataset provide a relatively weak representation of stock behaviour when viewed in isolation. While features such as opening price, closing price, highest price, lowest price, and trading volume contain valuable information about market behaviour, their power for predicting future price movements can be greatly improved through feature engineering.

In line with this thinking, the proposed framework generated an extensive set of technical features that describe various dimensions of market behaviour.

Rather than focusing on individual technical features, the generated technical features were grouped according to their functionality within the prediction framework.

The generated feature groups include:

- Trend indicators.
- Momentum indicators.
- Volatility indicators.
- Volume-based indicators.
- Price relationship indicators.

By structuring the features in this functional way, the prediction algorithms can learn about various complementary aspects of market behaviour while not becoming over-reliant on any specific individual technical indicator.

Unlike the manual analysis performed by a human trader, these indicators should be treated solely as engineered numerical values presented to the artificial intelligence models for analysis.

Table 3.3 Technical Feature Categories

Feature Category	Information Represented
Trend	Direction of market movement
Momentum	Strength and speed of price movement
Volatility	Market uncertainty and price dispersion
Volume	Trading activity and liquidity
Price Relationship	Relative position of current price

3.4.2 Financial Feature Engineering

While technical indicators provided valuable information regarding the behaviour of the market as a whole, financial variables allowed for the description of the internal condition of the listed companies.

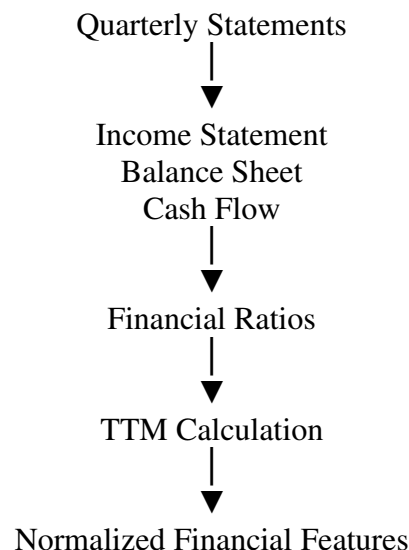
The financial statements were transformed into financial features that described the companies' profitability, liquidity, operational efficiency, leverage, valuation, and growth.

All financial variables were normalized and represented as standardized financial indicators to allow for the comparison of companies of different sizes.

Furthermore, Trailing Twelve Months (TTM) calculations were performed on most financial variables to smooth out the representation of the companies' performance and to allow for the reduction of the influence of seasonal fluctuations on those variables.

The incorporation of TTM calculations allows the machine learning models to use smoothed and more stable information regarding the financial variables of the listed companies.

Figure 3.5 Financial Feature Engineering



3.4.3 Feature Coverage Validation

Creating a large number of financial features ensures that the models will have a variety of potential input variables; however, it does not ensure that each of those features will contain high-quality data.

An analysis of the coverage of each of the created features was performed to ensure that each model would have high data completeness.

An analysis of the percentage of available observations for each of the created features allowed for the identification of any financial variables that were not complete before the creation of the prediction models.

As a result, it was found that most of the technical indicators were complete; the majority of the observations for RSI, MACD, Bollinger Bands, and ADX were above 90%. Furthermore, as a result of the engineering and integration of the TTM calculations, the financial features also had high data completeness and availability.

Thus, each of the created prediction models will use high-quality and complete data.

Table 33 Example Feature Coverage

Feature	Coverage (%)
RSI-14	96.08
MACD	92.46
MACD Signal	90.04
Bollinger Bands	94.27
ADX-14	100.00

3.4.4 Feature Selection Strategy

Rather than creating prediction models with the intention of using as many of the created features as possible, the feature selection was guided by four main principles:

1. Each of the selected features should represent some measure of financial or market behaviour.
2. No feature should include information from the future of the market or company financial variables (chronological validity).
3. The technical and financial features should represent different forms of market and financial information.
4. Financial features with insufficient coverage or those with unstable behaviour were eliminated from the models.

3.4.5 Phase Output

The output of this phase was the creation of a feature matrix that included the technical and financial variables that would be utilized in the creation of the artificial intelligence models.

The creation of the artificial intelligence models was performed after this phase and after the validation of all features.

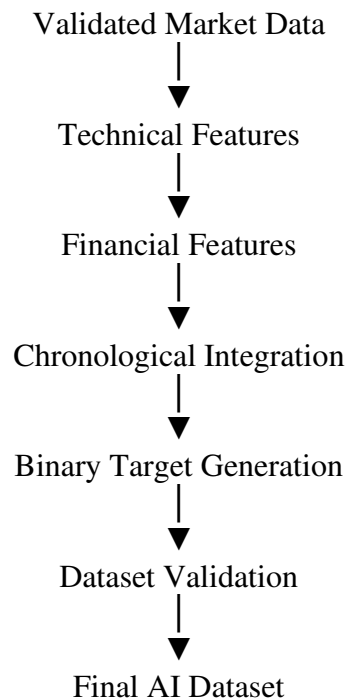
The inclusion of these steps before the creation of the prediction models increases the quality and the credibility of the models, as well as ensures that they can be reproduced in the future in the same area of financial artificial intelligence research.

3.5 Phase 4 - Dataset Construction and Binary Target Engineering

The development of the supervised machine learning models critically depends on the availability of a well-structured and accurately labelled training dataset. While the previous phases of the research described the steps that were necessary to ensure the quality of the available data, the objective of the present phase is to actually construct the machine learning dataset that will be used to train the artificial intelligence models.

As with the available financial datasets, the data that was engineered for this research included features that described the companies' behaviour and financial variables. Furthermore, the goal of creating such a dataset was to allow for the engineering of data that could be used to perform short-term stock market predictions. Thus, the dataset that was engineered for this research is used by the BiLSTM and the XGBoost models to train the models to recognize the features of markets that will lead to specific market movements.

Figure 3.6 Dataset Construction Workflow



3.5.1 Construction of the Master Dataset

After the process of feature engineering, the collected variables were combined into a single master dataset that would be used for supervised machine learning. Each observation in the dataset represented each trading day's market information, but with financial information associated with that market day that had become publicly available through time.

Each observation in the master dataset contained information regarding:

trading date

company

daily market information

engineered technical features

financial variables

binary prediction target

This master dataset enables the artificial intelligence framework to learn how to accurately predict stock market movement while avoiding the need for additional preprocessing of the data.

3.5.2 Binary Target Engineering

The artificial intelligence models that are to be developed will be utilized to perform a task that is essentially a classification problem. Rather than attempting to determine the price of a stock in the future, the goal of the models will be to recognize whether or not there is a positive investment opportunity within a defined period for that stock.

For each observation within the dataset, each data point will be associated with a target value of either a “1” or “0.”

If the target value is “1,” it means that there was a positive investment condition within the prediction period for the stock.

If the target value is “0,” it means that there were no positive investment conditions within the prediction period for the stock.

Thus, each model will essentially learn the probability that a given data observation is associated with a positive investment opportunity. Unlike portfolio management rules, such as stop-loss and take-profit levels, these targets are generated entirely within the supervised machine learning phase.

3.5.3 Prediction Horizon

For each stock market observation within the database, the following period that will be evaluated is a seven-day trading period. Thus, each data point will be associated with the subsequent seven trading days for that stock to determine if there was a positive investment condition during that time period.

By utilizing a period of seven trading days, there are a variety of advantages to this time period. First, it ensures that the stock market prediction becomes a supervised learning problem. Second, it guarantees that each data point is evaluated during the same time period. Finally, it provides a balance between examining the stock market for short-term movements in the stock prices while providing sufficient time for those prices to even change.

This time period will be utilized to define the artificial intelligence models' learning objectives while avoiding the introduction of any portfolio management rules.

3.5.4 Distribution of Binary Targets

The binary target data has been generated and analyzed to determine the distribution of each type of target within the master dataset.

In total, there were 11,598 labeled data points after the application of all of the target generation rules.

Within the dataset:

6,505 stocks had a target value of "1"

5,093 stocks had a target value of "0"

These targets are relatively even compared to other financial data prediction problems. Furthermore, within the stock market, it is typically true that positive investment opportunities exist for only a small fraction of the total labeled data points. Thus, the fact that there are an adequate number of each target class within the data indicates that there is an even distribution of opportunities for positive investments within the defined seven day trading period.

Table 3.5 summarizes the distribution of the binary target values.

Table 3.5 Binary Target Distribution

Target Class	Number of Samples
Target = 1	6,505
Target = 0	5,093
Total	11,598

3.5.5 Interpretation of Target Distribution

The target distribution indicates that there are sufficient observations of both the positive and negative targets. Furthermore, the use of a seven day trading period as the prediction horizon ensures that the artificial intelligence models will be able to learn about short-term investment opportunities while providing enough samples of each target class for the models to recognize the differences between each type of target.

3.5.6 Validation of the Dataset

The final data set created through this process was validated prior to the development of the artificial intelligence models. The validation process ensured that the master dataset contained all of the features and data points that were necessary to create a supervised learning algorithm, including:

chronological ordering of the stocks

binary target data

correctly identified companies

correct integration of technical and financial features

no duplicate data points

absence of look-ahead bias

Following the validation process, the dataset was confirmed as suitable for development of the artificial intelligence models.

3.5.7 Output of Phase 4

The output of this phase of the artificial intelligence framework is a complete and validated machine learning dataset that contains chronological data points of market activity, technical market features, financial features, and labels created through the binary target generation process.

It is important to note that this data was created for the sole use of training and testing the artificial intelligence models. The stop-loss and take-profit levels, such as the +5% take-profit or -2% stop-loss strategy for the stocks, are not generated during this process. These rules will be created during phase five, when the artificial intelligence models are to be tested within the context of a trading portfolio.

Thus, the targets are utilized to define the objectives of the artificial intelligence models, while the stop-loss and take-profit rules will define the strategy for evaluating the performance of those artificial intelligence models.

3.6 Development of the Individual Prediction Models

The machine learning dataset that was created and validated during phase four will be utilized during the development of the individual artificial intelligence models that are to be developed for the portfolio management system. Because there are two different types of data that relate to the stocks, it was decided that each AI model will be trained with the type of data that best relates to that model. Thus, two separate models will be trained and developed: one that learns from the technical market data for those stocks, and another that learns from the financial data for those stocks.

The reason that the technical and financial markets contain different types of information is due to the fact that the technical indicators form a series of observations of the market over time, while the financial data is a series of attributes for each company that is published in the company's financial reports.

Furthermore, it is likely that no single algorithm will be able to learn the features and aspects of both types of data equally well. Thus, the development of these two separate models will allow for each artificial intelligence algorithm to maximize its potential to recognize opportunities for positive investments in the stocks within the portfolio.

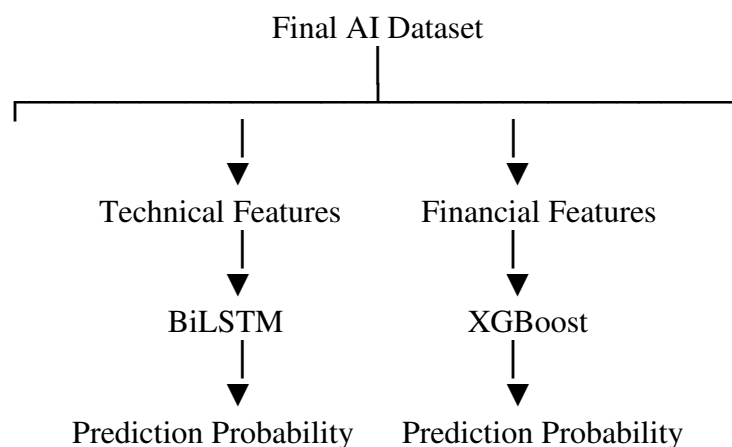
For this reason, two separate models will be developed during this phase:

A Bidirectional Long Short-Term Memory (BiLSTM) model that is trained with the technical market data.

An eXtreme Gradient Boosting (XGBoost) model that is trained with the financial data.

Figure 3.9 depicts the development of these individual AI models.

Figure 3.9 Parallel Model Development



3.6.2 Sequential Data Preparation

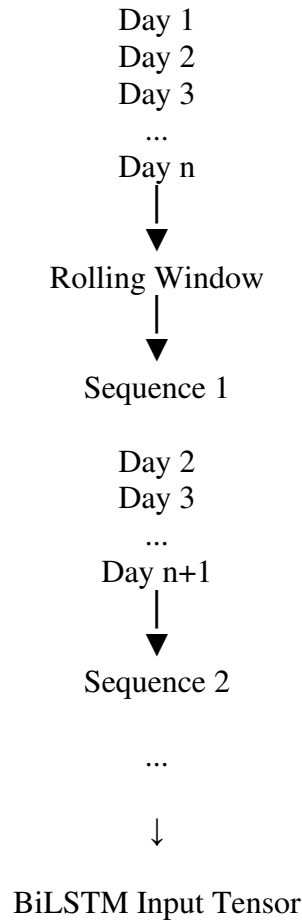
In addition to conventional approaches to data preparation, additional preprocessing is required to prepare the financial data for use with deep learning approaches.

Each group of consecutive daily observations was grouped into a rolling temporal window of a specific size. This enables the BiLSTM model to learn from the relationship among each of the consecutive market observations, rather than analyzing each trading day independently.

The structure of these temporal data groupings ensures that the temporal relationship is maintained throughout the complete dataset, and ensures that no future market observations are incorporated into any of those sequences.

Figure 3.10 illustrates the process of creating temporal sequences from the dataset of daily market observations.

Figure 3.10 Sequential Window Construction



3.6.3 BiLSTM Training Strategy

The chronological training dataset was used to train the BiLSTM model. The data was divided into training, validation, and testing subsets according to their chronological order within the training dataset. As such, no future market observations were used during the training phase of the BiLSTM model. Furthermore, the model was trained according to the validation data to minimize overfitting of the training data, and to maximize the generalization of the BiLSTM model to the observations in the testing data. Consequently, early stopping techniques were utilized during training to ensure that the model was generalized well to the training data, yet still performed well on the validation data.

3.6.4 Output of the BiLSTM Model

Rather than directly converting the output of the BiLSTM model into a trading signal, the probability that the market observation belongs to the positive class of trading signals is calculated. This probability is not directly used to make a trading decision, but is sent to the probability fusion stage of the hybrid model, along with the trading signal probabilities calculated by the XGBoost model. Utilizing the probabilities rather than the binary trading

signals maximizes the potential for finding an appropriate optimization strategy for the BiLSTM and XGBoost model during the training phase.

3.6.5 BiLSTM Model Configuration and Training Strategy

As with other machine learning models, the deep learning approach of BiLSTM models also has various parameters that must be configured prior to training. The BiLSTM model was developed with specific configurations that ensure that it gains the maximum predictive capability while minimizing the potential for overfitting of the training data.

Each of the parameters of the model were configured according to the characteristics of the financial data that was used to train the model.

3.6.5.1 Sequential Observation Window

As with other deep learning models, the BiLSTM model requires the input of various sequential observations. The technical analysis data was converted into rolling sequences using a specific lookback window size. Each example of the training data included multiple consecutive trading sessions. Thus, the model was able to recognize relationships among the various technical indicators that were measured during each of those sessions, and to use those relationships to inform the prediction of future market behavior. Furthermore, a specific size of the lookback window was chosen for the model; various factors informed the choice of that size. The structure of the sequences of daily market observations that were created for training data is displayed in Table 3.6.

Table 3.6 BiLSTM Input Configuration

Parameter	Value
Lookback Window	30
Input Features	X_lstm
Input Shape	(Samples × Timesteps × Features)
Prediction Output	Binary Probability (Sigmoid Output)

3.6.5.2 Network Architecture

The proposed BiLSTM model consists of the following network architecture: multiple Bidirectional LSTM layers and dropout layers, followed by an output layer with a Sigmoid activation function.

The Bidirectional LSTM layer is used to capture the long-term dependencies between market observations from the previously engineered technical indicators. Furthermore, multiple such layers allow the model to sequentially learn and refine the dependencies between these observations prior to making the final prediction.

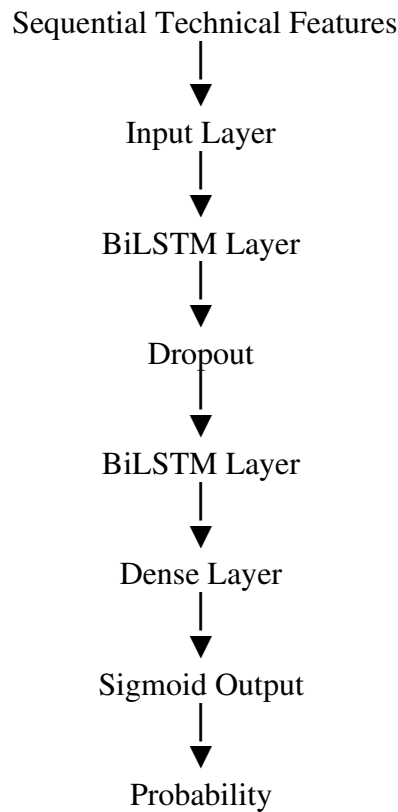
Dropout layers are incorporated into the network between the Bidirectional LSTM layers to regularize the model and reduce overfitting. By randomly deactivating a certain proportion of

neurons within the network during training, the model learns to rely less upon any specific weights within the network to perform well, which improves its generalization ability on market conditions it has not yet seen.

The output layer of the BiLSTM network contains a single neuron with the Sigmoid activation function to determine the probability that a given set of technical indicators belongs to the positive class within the classification problem.

Figure 3.11 illustrates the architecture of the BiLSTM network.

Figure 3.11 BiLSTM Architecture



3.6.5.3 Training Strategy

The training data for the BiLSTM model consisted of the training dataset created during the preprocessing phases of the project.

The training data was divided chronologically into training, validation, and testing data subsets.

By partitioning the training data in this way, the model is trained in a way that more accurately reflects real-world market conditions.

The model's performance on the validation data was used to monitor the training process. Using this validation data, it was possible to automatically adjust the training process to avoid overfitting to the training data alone.

3.6.5.4 Early Stopping Strategy

Deep neural networks such as the BiLSTM network are often vulnerable to overfitting to the training data.

To mitigate this issue, an early stopping strategy was incorporated into the training strategy. The training process is automatically terminated if the model's performance on the validation data does not improve over a pre-defined number of training epochs.

By including this strategy, the model is less likely to overfit the training data while training the network. Furthermore, this strategy allows for the automatic selection of the best model to utilize rather than having it selected manually after training is complete.

3.6.5.5 Model Optimization

In the training of the BiLSTM model, the Adam optimization algorithm was utilized.

The Adam optimization algorithm is a gradient-based optimization algorithm that automatically adapts the learning rate within the network according to the performance of the model. Thus, the algorithm is often used to train deep neural networks, especially on large datasets, such as financial time-series data.

For the BiLSTM model, the objective function to minimize was the binary cross-entropy error function. As the goal of the model is to determine the probability of the observation belonging to each of the two classes, the binary cross-entropy function was the most appropriate loss function to minimize in this case.

Table 3.7 BiLSTM Configuration

Hyperparameter	Configuration
Lookback Window	30
Number of Technical Features	9
Batch Size	64
Maximum Epochs	50
Optimizer	Adam
Learning Rate	0.001
Loss Function	Binary Cross-Entropy
Output Activation	Sigmoid

Early Stopping	Enabled
Validation Strategy	Chronological Validation Set
Gradient Clipping	clipnorm = 1.0
Dropout	0.30
LSTM Units	64
Recurrent Dropout	0.20
L2 Regularization	0.01

3.6.5.6 Training Monitoring

Throughout the training of the BiLSTM model, the loss of both the training and validation data was continuously monitored.

By recording both training and validation loss after each epoch, it is possible to create a visual representation of the training process of the network. Furthermore, such a visual representation can aid in the determination of the best BiLSTM model to utilize.

Figure 3.12 presents the learning process of the BiLSTM network.

Figure 3.12 BiLSTM Learning Curve

3.6.5.7 Output of the BiLSTM Training Phase

The output of the BiLSTM model after the training phase is the probability of the technical indicator observation belonging to the positive class for investments. As these probabilities contain richer information than the classification of these observations as positive or negative classes, these outputs are the input that is fed into the hybrid probability fusion stage.

3.6.6 Development of the XGBoost Model

In contrast to the BiLSTM model, a different machine learning model had to be utilized to analyze the information contained within the financial statements of the listed companies. The financial information within these tables does not have a temporal component to it, unlike the technical indicators. As such, the model that will be utilized within the second component of the framework is eXtreme Gradient Boosting (XGBoost) - a machine learning algorithm that is well-suited to analyzing tabular data.

The XGBoost model will take the financial features created during Phase 3 as its inputs, as these are the only variables that will be fed into this second component of the framework.

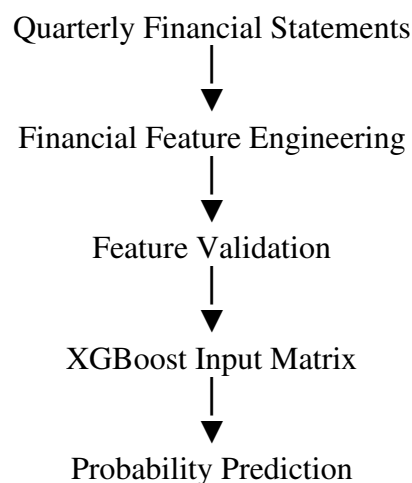
3.6.6.1 Financial Feature Representation

The inputs to the XGBoost model will be the financial variables extracted from the listed companies' financial statements and transformed during the feature engineering stage.

These financial variables represent a variety of different aspects of the financial statements and the companies that published them.

Unlike the technical indicator features, these financial variables have no temporal component to their features. Furthermore, before the development of the XGBoost model, all of the financial observations had already been pre-processed to ensure that they satisfied the conditions required to be used by the BiLSTM model.

Figure 3.13 Financial Learning Pipeline



3.6.6.2 Gradient Boosting Strategy

The XGBoost model is based on the gradient boosting strategy, which trains multiple decision trees to iteratively improve the prediction performance of the previous decision trees.

Instead of relying on a single decision tree to learn and make predictions, the algorithm trains additional decision trees whose predictions are used to compensate for the errors of the previous decision trees.

This strategy allows the model to effectively learn nonlinear relationships between the financial variables and accurately predict the target variable while maintaining good computational efficiency.

The objective of the XGBoost model is to estimate the probability of a financial observation belonging to the positive trading class.

3.6.6.3 Model Configuration

The XGBoost model configuration was chosen to provide a good balance between the model's prediction performance and its generalization ability.

The model adopts a relatively simple configuration to reduce the risk of overfitting while maintaining the model's ability to learn complex nonlinear relationships between the financial variables.

The main parameters that can be configured within XGBoost include the boosting strategy, tree depth, learning rate, sampling strategy, and the regularization controls. The final configuration of the model's hyperparameters is presented in Table 3.8.

Table 3.8 XGBoost Hyperparameter Configuration

Hyperparameter	Value	Purpose
Objective	binary:logistic	Binary probability estimation
Learning Rate (η)	0.03	Controls update step size
Max Depth	4	Controls tree complexity
Number of Trees	300	Number of boosting iterations

3.6.6.4 Training Procedure

The XGBoost model learns the weights associated with each financial feature using the feature matrix obtained from the previous stage of the model's training.

The training, validation, and testing datasets were chronologically separated to ensure that the financial market signals from future trading sessions were not included in the training and testing datasets.

The model was trained on the training and validation datasets using cross-validation, while the testing dataset was used solely to test the model at the end of the training phase.

The chronological training of the XGBoost model ensures that the model's performance is representative of its ability to accurately forecast the stock market's future trading signals.

3.6.6.5 Feature Learning Characteristics

The XGBoost model learns the features in a manner similar to decision trees; it partitions the feature space into different regions according to which region contains the majority of observations with a particular value of the target variable.

This approach allows the model to find complex nonlinear relationships between the financial variables.

Furthermore, as the model is based on decision trees, it is inherently somewhat explainable; the contribution of each financial variable to the learning of the model can be analyzed in the explainable AI stage of this project.

3.6.6.6 Output of the XGBoost Model

The output of the XGBoost model is the probability that a financial observation belongs to the positive trading class.

This probability is not directly used in making trading decisions; instead, it is preserved as an output for use in the fusion of probabilities with the probabilities generated by the BiLSTM model.

By preserving the probabilities from both models, the trading signals can be generated based on the information content of both the technical and financial variables of the company in question.

3.6.7 Comparative Characteristics of the Individual Models

After the development of the two individual models, it is useful to review their different roles within the trading system that is proposed for this project.

Both models solve the same problem of identifying whether a company will experience a positive trading class; however, they use different representations of that financial information.

The BiLSTM model learns the temporal patterns in the sequence of technical indicators for a company, while the XGBoost model learns the nonlinear relationships between financial variables.

These different specializations of the two models will allow them to contribute to the hybrid probability fusion stage of this project.

Table 3.9 presents a comparison between the two individual models within the system.

Characteristic	BiLSTM	XGBoost
Input Data	Sequential Technical Features	Structured Financial Features
Data Representation	Time-Series Sequences	Tabular Financial Variables
Learning Mechanism	Bidirectional Recurrent Neural Network	Gradient Boosted Decision Trees
Output	Probability	Probability
Primary Strength	Temporal Pattern Learning	Structured Financial Modeling
Role in Framework	Technical Prediction	Financial Prediction

3.7 Phase 6 - Hybrid Probability Fusion

Following the separate development of both the BiLSTM and the XGBoost models, the outputs of each model can be fused together to form a more robust trading prediction model. Each of the models was developed to address the same classification problem within trading, yet each examines different aspects of the trading data. Therefore, the individual models can be fused together to take advantage of the strengths of each model while avoiding the limitations of each individual model.

The fusion of the individual prediction models will, therefore, represent the transition from the independent learning of the models to the integrated learning of the framework as a whole.

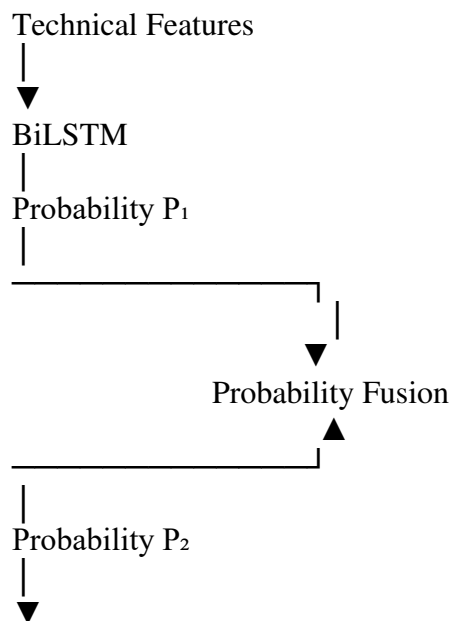
3.7.1 Rationale for Probability Fusion

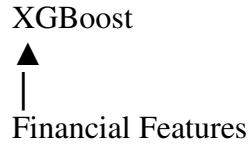
Within the context of artificial intelligence systems, it is often possible to combine multiple prediction models during different stages of the learning process. For instance, the features within the learning data can be combined before each model is trained, the predictions of each model can be combined, or the probabilities of each model can be combined before a decision is made regarding trading activity.

By fusing the probabilities of each individual model rather than the predictions of the models, more information is retained by the framework. By utilizing the probabilities of the models instead of the final trading predictions of the models, additional flexibility is also provided to adjust the contribution of each model.

Finally, by fusing the probabilities of the models, the individual models can be provided with weights that indicate the contribution of each model to the trading prediction, without having to alter the learning algorithms of either of the models.

Figure 3.14 Hybrid Probability Fusion





3.7.2 Probability Integration Strategy

Within the framework, each of the learning models creates probabilities associated with each observation within the data. Each of these probabilities represent the likelihood of the observation to belong to the trading class with a positive outcome.

These probabilities from each of the individual models are combined to create a single, hybrid probability.

The hybrid probability is created as a weighted combination of the probabilities of each of the individual models.

The weighting of each model's probabilities indicates the contribution of each model's prediction within the final trading prediction stage. Thus, the weights allow for the framework to adapt to different trading market conditions.

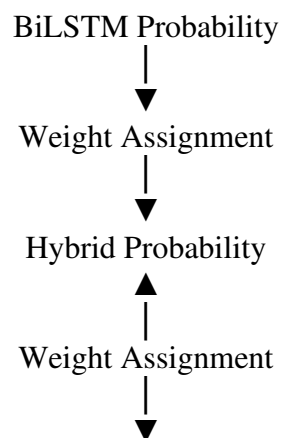
3.7.3 Weight Optimization Strategy

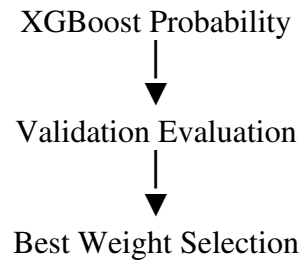
Determining the weights to apply to each of the individual models is one of the important components of the framework's development.

Multiple combinations of weights can be tested using the validation data to determine the weights that create the most effective combination of the two models' prediction capabilities.

These weights will be applied to the independent testing data following their determination with the validation data.

Figure 3.15 Weight Optimization Procedure





3.7.4 Hybrid Probability Output

The output of the fusion stage is a probability value indicating the combined prediction of the two learning models.

Unlike the probabilities generated independently by the BiLSTM and XGBoost models, the probability calculated during the hybrid fusion process incorporates information extracted from both the technical and financial markets.

This probability value indicates the basis upon which investment decisions will be made during the portfolio evaluation stage.

Maintaining the outputs as probabilities rather than making binary decisions allows for more flexibility in adjusting the investment decisions according to the performance of the trading portfolio.

3.7.5 Advantages of the Proposed Hybrid Strategy

The proposed strategy includes several advantages.

First, each of the prediction models includes the type of information that is best for that model's architecture.

Second, the fusion of the probabilities preserves the model's prediction of the target variable.

Third, the hybrid model allows for the optimization of weights to determine the contribution of each model to the final outcome without altering the learning of the prediction models themselves.

Fourth, the model is easily extensible by adding additional prediction models to the learning phase.

Finally, the strategy separates the learning phase from the investment decision phase, allowing the two processes to be optimized independently of one another.

Table 3.10 Advantages of Probability Fusion

Design Component	Methodological Benefit
Independent Learning	Specialized feature learning
Probability Output	Preserves prediction confidence
Weighted Fusion	Flexible contribution of each model
Modular Architecture	Easy extension and maintenance
Separate Decision Stage	Independent portfolio optimization

3.7.6 Output of Phase 6

The output of phase 6 includes a probability for every observation in the test data set.

However, this probability is not the investment decision for those stocks. Rather, it is the prediction of the hybrid model that will determine the investment decisions made during the portfolio evaluation stage.

The separation of these two stages offers flexibility in making investment decisions while also maintaining the independence of the learning phase of the framework.

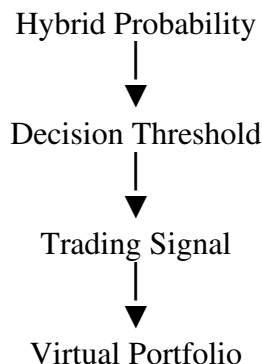
3.8 Phase 7 - Portfolio Evaluation Framework

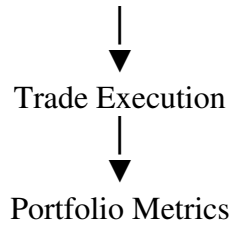
Although the framework was trained as a classification model, its purpose is to allow investors to make investment decisions based upon the model's predictions. Thus, evaluation of the classification models using traditional classification model metrics only provides a limited evaluation of the utility of the model. Accordingly, a portfolio evaluation framework is proposed to enable the model predictions to be evaluated in relation to its potential to enable investors to make investment decisions.

Unlike the other phases of the framework, the portfolio evaluation framework does not influence the training of the prediction models. Instead, the portfolio evaluation stage enables the evaluation of the potential of the model's predictions to enable investors to make investment decisions.

Figure 3.16 illustrates the portfolio evaluation framework.

Figure 3.16 Portfolio Evaluation Framework





3.8.1 From Prediction to Trading Decision

The trading model will generate a trading decision for every observation in the market data.

The trading decision will be made by comparing the value of the prediction of the model's probability to a threshold. If the probability value meets the threshold, then an investment decision will be made with that stock; otherwise, the stock will be ignored by the model.

Thus, the trading model's prediction and investment decisions are independent of one another. Furthermore, the threshold for investment decisions is applied after the trading model makes its predictions, ensuring that the learning phase of the model is independent of the trading decisions of the investor.

3.8.2 Virtual Portfolio

The trading signals generated by the trading model were evaluated using a virtual portfolio.

The virtual portfolio was constructed to evaluate the trading signals generated by the trading model.

The virtual portfolio executed every trade signal generated by the trading model and tracked the performance of each trade.

3.8.3 Trading Rules

Each trade signal was executed according to a set of rules for the virtual portfolio.

The rules were established independently of the trading model and its training.

The trading rules included the following components:

Rules for making trading entries according to the trading model's signals

A holding period for each trade

A take-profit level for each trade

A stop-loss level for each trade

An evaluation of each executed trade

These trading rules are only employed during the portfolio evaluation stage. Thus, they are independent of the training of the trading model.

Table 3.11 Portfolio Evaluation Rules

Component	Description
Signal Source	Hybrid AI Model
Portfolio Type	Virtual Portfolio
Decision Rule	Probability Threshold
Exit Strategy	Fixed Portfolio Rules
Performance Assessment	Portfolio-Level Metrics

3.8.4 Portfolio Performance Recording

Every trade executed by the portfolio was recorded during the evaluation of its performance.

The portfolio engine automatically recorded measures including:

- The entry date for the trade.
- The exit date for the trade.
- The entry price for the trade.
- The exit price for the trade.
- The length of time that the trade was active.
- The return of the individual trade.
- The evolution of the equity of the portfolio.

By recording the complete trading history of the portfolio's trades, it becomes possible to perform a detailed analysis of the performance of the portfolio and to ensure that the evaluation of its performance is completely reproducible.

3.8.5 Performance Measurement Strategy

As with many other forms of machine learning evaluation, the evaluation of the profitability of investments also considers the risk associated with those investments.

A variety of different performance measures were considered for evaluating the profitability of the investments, including measures of profitability, consistency in trading, and risk measures.

Each of these different performance measures allows for a more complete evaluation of the capabilities of the proposed investment framework.

The results of these measures are presented and analyzed in Chapter Four of this thesis.

3.8.6 Separation Between Learning and Evaluation

One of the main characteristics of the proposed framework is the separation that exists between the learning of the models and the evaluation of the trading portfolio.

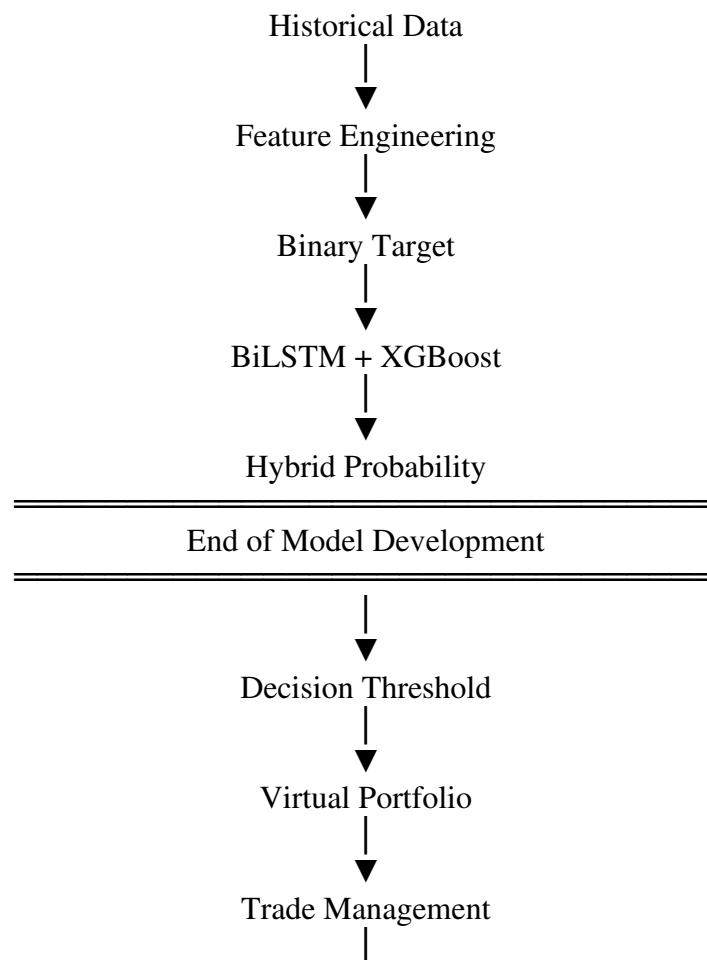
When the learning algorithms are trained to classify the market movements as either positive or negative returns, those algorithms do not consider any factors related to the trading portfolio.

Thus, after the algorithms have learned of the probability that the markets movements will be positive or negative returns, it is only at this point that trading decisions are executed within the portfolio.

This method of separating the learning algorithms from the trading portfolio helps to ensure the objectivity of each of these processes.

Figure 3.17 illustrates this separation between the learning and evaluation of the trading portfolio.

Figure 3.17 Separation Between Learning and Portfolio Evaluation





Performance Evaluation

3.9 Chapter Summary

The engineering methodology to be used for the development of the proposed framework for artificial intelligence trading systems was presented in this chapter.

The methodology included first auditing the data to be used to ensure its integrity, followed by integrating the technical and financial features into a machine learning dataset. The algorithm was used to create a target for each instance of the created dataset. Two different prediction models were created for the system, each with a different set of features. Finally, a portfolio evaluation framework was created to evaluate the profitability of the predictions made by the artificial intelligence trading framework. Furthermore, the method for creating the artificial intelligence trading framework ensured the separation between the learning of the models and the evaluation of the trading portfolio.

The following chapter presents the results of the experiments that were performed for the proposed framework.