university of engineering and technology peshawar pakistan

EFFICIENT LOW LATENCY HARDWARE ACCELERATOR FOR HIGH FREQUENCY TRADING USING FPGA

By

AZAZ HASSAN KHAN (18JZELE0262)

WAQAS AHMED (18JZEL0254)

Harris Khan

The undersigned certify that they recommend to the UET Peshawar for acceptance of this thesis for the fulfillment of the requirements for the degree stated.

Signature:

Dr. Malik Umer Sharif

Supervisor:

Signature:

Engr. Najvia Khattak

Final Year Project Coordinator:

Signature:

Dr. Amjid Ali

Chairman:

Date:

EFFICIENT HARDWAERE ACCELERATOR FOR HIGH FREQUENCY TRADING USING FPGA

By

AZAZ HASSAN KHAN

WAQAS AHMED

Harris Khan

A Thesis

Submitted to the UET Peshawar

As a Requirement for the Degree of

bachelor OF SCIENCE

DEpartment of ELECTRICAL engineering

UNVIVERSITY OF ENGINEERING AND TECHNOLOGY,

PESHAWAR, JALOZAI CAMPUS.

PAKISTAN

AUGUST 2022

**DEDICATION**

*I dedicate my work to my parent who supports me in throughout my educational journey, without their supports this achievement was not possible. I am extending my dedication to my final year project supervisor Dr. Umar Sharif Assistant Professor, University of Engineering and technology Peshawar, Jalozai campus. I am very thankful to my respected supervisor who motivates and guide me in my final year project and with the help of my respected supervisor I have complete my final year project and achieved the required millstones.*

**ACKNOWLEDGEMENTS**

**AZAZ HASSAN KHAN**

First of I would like to thanks to my parent who support me throughout my journey, without their support this millstone was not possible. I would like to extend my gratitude to all my teachers who help and support me throughout my journey. Special Thanks to my supervisor, Dr. Umar Sharif for providing guidance and feedback throughout this project. Without his support this project was impossible.

**ABSTRACT**

Stock market prediction is one of the most trending problems in today’s world and the only aim of stock market prediction is to make profit and reduce capital risks. According to the efficient market hypothesis theory, it’s almost impossible to predict the stock market prices with 100% accuracy as it takes into account many factors like the political upheaval, interest rates, current events, exchange rate fluctuations, natural calamities and much more.

The invention of machine learning techniques makes it possible to predict the stock market direction with maximum accuracy and increase profit and reduce capital risks by reacting with high speed to those predictions using High-speed hardware like FPGA (Field Programmable Gate Array). High-Speed Hardware System promises an increase in profit of about 0.0005 per share or a twentieth of a penny.

This project aims to design a High-Speed Hardware System to predict stock market prices with maximum accuracy using machine learning techniques. This project will be mainly divided into three parts, in the first stage we will be developing an algorithm in python for stock market prediction using machine learning technique in which first of all we will train our ML model on Stock market data and after analyzing the capability of our developed model in software, in the second stage we will implement our algorithm on FPGA (High-Speed Hardware) using VHDL (Verilog High Description Language ) to achieve high speed to maximize profits and reduce risks and finally in the third stage we will require to capture real-time data for the stock market so we will interfacing FPGA with Ethernet (10 gigabytes) for real-time stock data.

**Figure a. Project Different Stages Overview**

**contents**

**1 Introduction**

**1.1 Motivation**

**1.2 Project Implementation Idea**

**2 Traditional A.I Trading Techniques**

**2.1 RSI indicator Based Strategy**

**2.2 STOCHASTIC Indicator Based Strategy**

**2.3 MACD Indicator Based Strategy**

**2.4 Bollinger Bands Indicator Based Strategy**

**3 Machine Learning Trading Techniques**

**3.1 Methodology**

**3.2 Experimental Results**

**4 High Frequency Trading Techniques**

**4.1 Methodology and implementation of three indicators in hardware (FPGA):**

**5.2 Experimental Results**

**5 References**

**chapter 1**

**introduction**

* 1. **Motivation**

Pakistan is a developing country that came into existence almost 75 years ago and for any developing country financial market system is one of the most important things in its progress but unluckily we still utilize the manual traditional software-based methods for the financial market management system which are too slow as well as manual.

The main motivation of this project is

* To Provide Stock Market prediction with maximum accuracy using machine learning.
* To increase profit of the investors in the stock market using High-Speed Hardware (FPGA).
* To Provide a scalable, cheap, and fast stock market capital management hardware system.
* To Provide a system that is automatic and intelligent with less or no human interface.
* To Provide a system that reduce capital risks of investors during the market crash.
* To Provide a system that works on every market like cryptocurrencies, forex and stock market.
* To Provide a system that can scan multiple markets and exchanges. It enables traders and investors to find more trading opportunities, including arbitraging slight price differences for the same asset as traded on different exchanges.
  1. **Project implementation idea**

The Implementation of this project will be done in following parts:

* Designing and Implementation.
* Testing.
* Evaluation.

**1.2.1. Designing and implementation**

The designing and implementation phase of this project is started in mid of October and we will continue working on the following aspects.

**1.2.2. Designing and implementing stock market data extractor**

We will design and implement a python code to extract free stock data from broker or stock market. The data we extract will include High, Low, Close, Open and Volume data columns.

**1.2.3. Designing and implementing stock market data organizer**

We will design and implement a python code on extracted stock data to make it organize, clean and readable for humans.

**1.2.4. Designing and implementing an algorithm for stock market prediction using machine learning technique**

We will design a python code to develop an algorithm for stock market prediction using machine learning model. This designed algorithm will be the main brain of the system which will be responsible for controlling and generating all the financial decisions in the financial management system and then we will implement this algorithm on stock market data.

**1.2.5. Designing Back tester for developed algorithm**

We will design a back tester in python to analyze and check the intelligence, accuracy and decisions making of developed algorithm on financial stock market systems.

**1.2.6. Designing a hardware-based code for developed algorithm**

After designing an algorithm in software (python) successfully now it will be a perfect time to develop a hardware-based code for an already developed algorithm using VHDL (Very High-Speed Integrated Circuit Hardware Description Language).

**1.2.7. Implementing VHDL algorithm on FPGA board (High-Speed Hardware)**

We will implement already designed VHDL-based algorithm on the FPGA board. We will test our VHDL-based algorithm on dummy data in FPGA board.

**1.2.8. Designing Ethernet interface for FPGA**

After verifying the intelligence, accuracy and decisions making of VHDL based algorithm, we will design an interface to connect Ethernet to the FPGA using VIVADO software for extracting the real-time data.

**1.2.9. Designing an API for Full-duplex communication using Ethernet**

After interfacing Ethernet with FPGA board properly we will design an API for Full-duplex communication between stock market and our FPGA board using C coding, so data can be transmitted as well as received through FPGA board.

**1.2.10. Designing an API for Stock market real-time data parser**

After receiving stock market data in real-time from stock market we will design an API for Stock market data to parse the data and make it processible for FPGA algorithm.

**1.2.11. Designing a full data path from Ethernet interface to FPGA algorithm**

This will be the last step in designing and implementation. In this part, we will design a full two-way data path from the Ethernet interface to FPGA algorithm and vice versa, so that the stock data can be received and predictions on that data can be transmitted simultaneously.

**1.2.12. Testing**

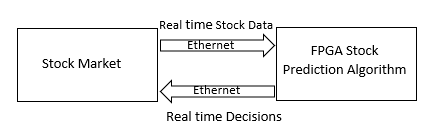
In the testing phase, we will test our FPGA board (High-Speed Hardware System) on real-time stock data by connecting the FPGA board to the computer using Ethernet we will be analyzing the speed and accuracy of the overall process.

**1.2.13. Evaluation**

After we finish the testing of our system, we will be evaluating its performance and we will compare its performance with the software-based system based on accuracy and speed.

**Figure 1.1. Project Implementation Overview**

The Final Product out of this project will be like a real-time stock data will be streaming through Ethernet into the FPGA Hardware system, algorithm implemented inside the FPGA will make a decision based on the stock data like buying/selling of stock and after that those decisions will be sent back to the stock market.



**Figure 1.2: Final Product Overview**

**Chapter 2**

**Traditional AI Trading Techniques**

In this chapter we will discuss traditional AI trading techniques. Traditional AI trading techniques are those trading techniques which need human interference. Our designed strategies are based on the indicators and Candle Sticks. In this chapter we will discuss how we will use those indicators to make buy and sell decisions. We will discuss the following topics:

**Technical Indicators**

* Relative Strength Index
* MACD Indicator
* Stochastic Indicator
* Bollinger Bands

**2.2. Relative Strength Index (RSI) Based Strategy**

The (RSI) is a momentum indicator used in technical analysis that measures the magnitude of recent price changes to evaluate overbought or oversold conditions in the price of a stock or other asset. The Formula for RSI is shown in Equation 1.

**2.2.1. How RSI Indicator Works?**

Now it’s time to reveal our own made RSI indicator-based strategy. As discussed previously that everyone uses the indicators on their own ways. So here I will explain how we use RSI indicators and make an efficient strategy. Here is the setup of our strategy. N for our strategy is 10.

**Buy signal:**

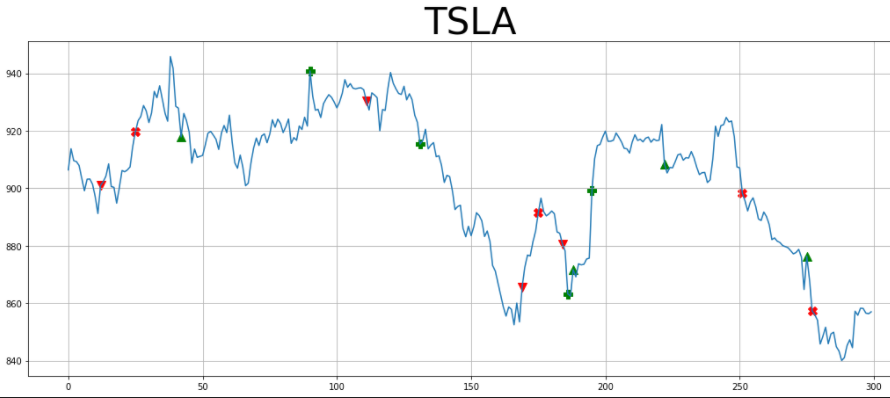
1. Periods RSI is equal to or less than 25 level.
2. Wait for 10 periods RSI to hit 40 level in upward direction

**Stop loss:** When 10 periods RSI hit 35 level in reverse direction, order will close automatically.

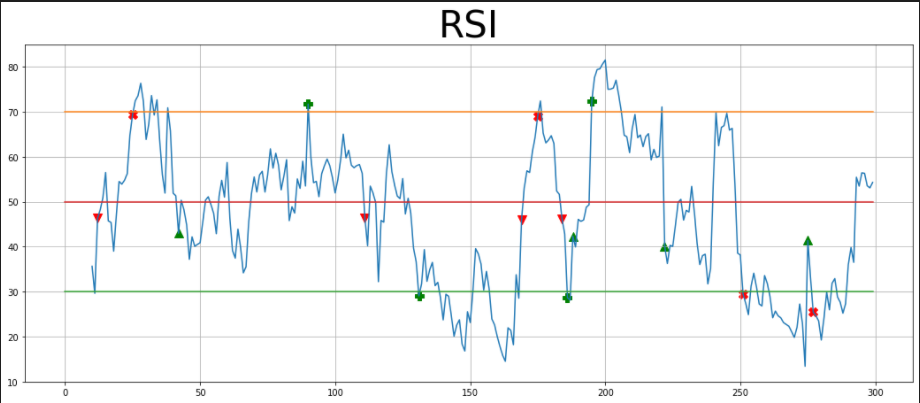
**Take profit:** When 10 periods RSI hit 70 level in upward direction, order will be close automatically

**2.2.2. RSI Results:**

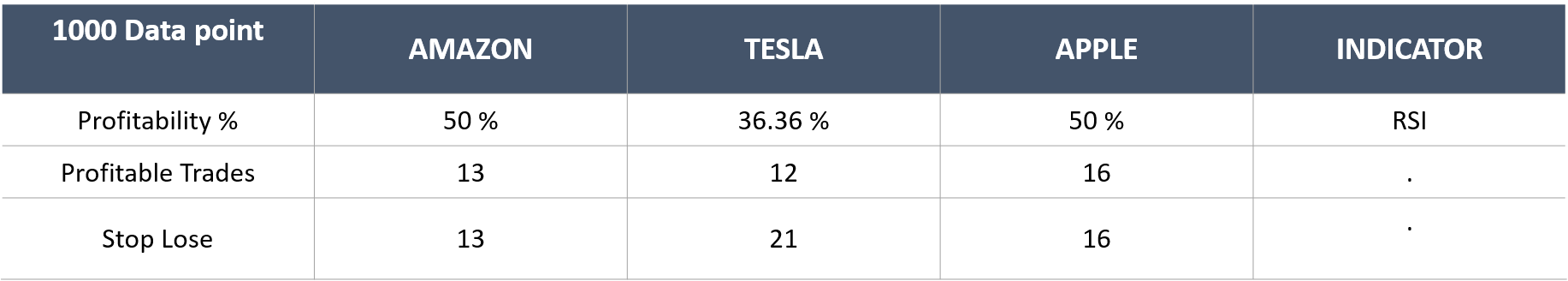
Here we will discuss how we implement RSI strategy in software using built-in tools. The above figure is the result of RSI strategy in software using built-in tools and here we used Tesla data to be analyzed. From figure it is shown that we have Up, down and cross arrows. The Up arrow shows the positions where we have to buy, similarly down arrow is a position where we have to sell and “x” shows where we have to close your orders whether on stop loss or take profit.

****

**Figure 2.1. Prediction of Tesla prices using RSI indicator**

****

**Figure 2.2. Prediction of Tesla prices using RSI indicator**

****

**Table 2.1. Comparison of Different US Stock market companies using RSI**

**2.3. STOCHASTIC Indicator Based Strategy:**

9

The indicator shows how the current price compares to the highest and lowest price levels over a predetermined past period. Stochastic indicator is a popular indicator used for locating the overbought and oversold positions of assets. The Formula is shown in Equation (2).

%K=100×

CP=Most recent closing price

L14=Lowest price of the 14 previous trading sessions

H14=Highest price of the same 14 previous trading sessions​

**2.3.1. How Stochastic indicator Works:**

Now it’s time to reveal our own made STOCHASTIC indicator-based strategy. As discussed previously that everyone uses the indicators on their own ways. So here we will explain how we used RSI indicators and make an efficient strategy. Here is the setup of our strategy

**Buy signal:** When %K line cross above %D line below 30 level.

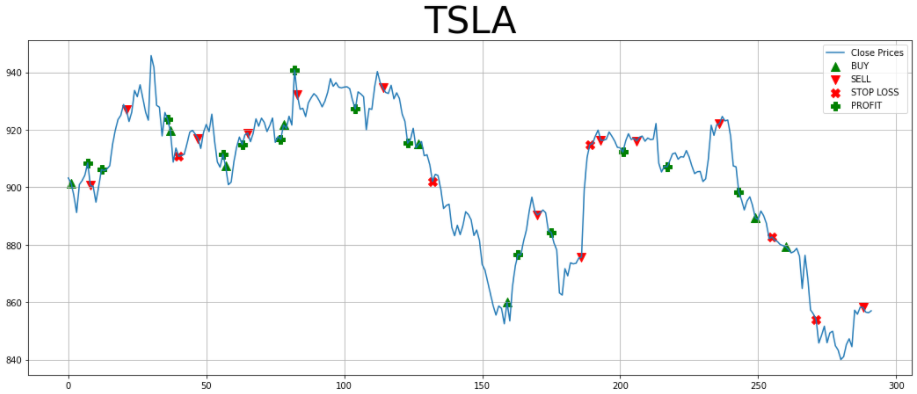
* Wait for %D line to cross 30 level in upward direction.
* When the above two conditions met, then we will buy

**Sell signal:** The conditions for the sell signal are below:

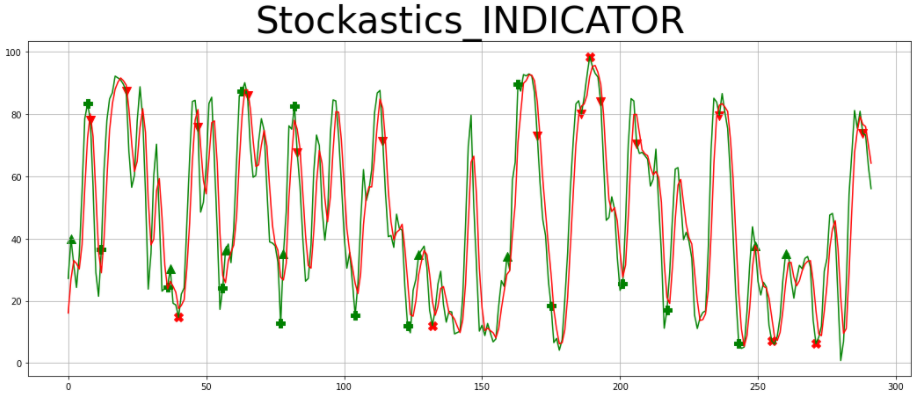
* When %K line cross down %D line above 70 level.
* Wait for %D line to cross 70 level in downward direction.

**2.3.2. Stochastic Results:**

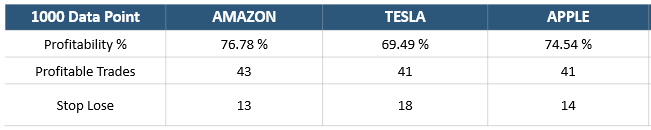
The above figure is the result of STOCHASTICH strategy in software using built-in tools and here we used Tesla data to be analyzed. From figure it is shown that we have up, down and cross arrows. The Up arrow shows the positions where we have to buy, similarly down arrow is a position where we have to sell and “x” shows where we have to close your orders whether on stop loss or take profit.

****

**Figure 2.3. Prediction of Tesla prices using Stochastic indicator**



**Figure 2.4. Prediction of Tesla prices using Stochastic indicator**



**Table 2.2. Comparison of Different US Stock market companies using Stochastic**

**2.4.1. MACD indicator based strategy:**

Moving average convergence divergence (MACD) is a [trend-following](https://www.investopedia.com/terms/t/trendtrading.asp) [momentum](https://www.investopedia.com/terms/m/momentum.asp) indicator that shows the relationship between two [moving averages](https://www.investopedia.com/terms/m/movingaverage.asp) of a security’s price. The MACD is calculated by subtracting the 26-period [exponential moving average](https://www.investopedia.com/terms/e/ema.asp) (EMA) from the 12-period EMA. The formula for MACD is shown in Equation 3-4.

MACD Line=12-Period EMA − 26-Period EMA (3)

Signal Line= 9-Period EMA of MACD Line (4)

MACD is calculated by subtracting the long-term EMA (26 periods) from the short-term EMA (12 periods). An exponential moving average (EMA) is a type of [moving average](https://www.investopedia.com/terms/m/movingaverage.asp) (MA) that places a greater weight and significance on the most recent data points.

**2.4.2. How MACD indicator Works:**

The MACD indicator is famous for cross-over and cross-down strategy that is when MACD line crosses above the signal line so its good time to buy. Similarly, when MACD Line crosses below the signal line so its good time to sell. This strategy is used very widely, so we also use this strategy as it.

**Buy signal:**

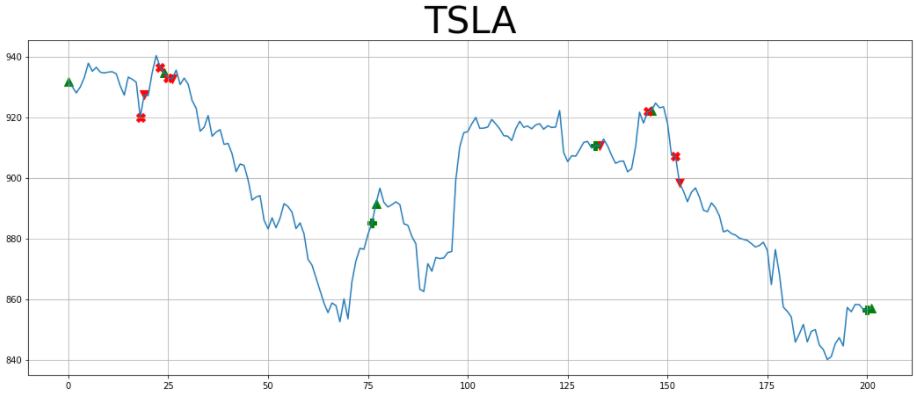
When MACD line cross above signal line, wait for and candle and then enter long(buy). When MACD line cross above signal line, wait for the significant difference between signal line and MACD line then enter long (buy)

**Sell signal:**

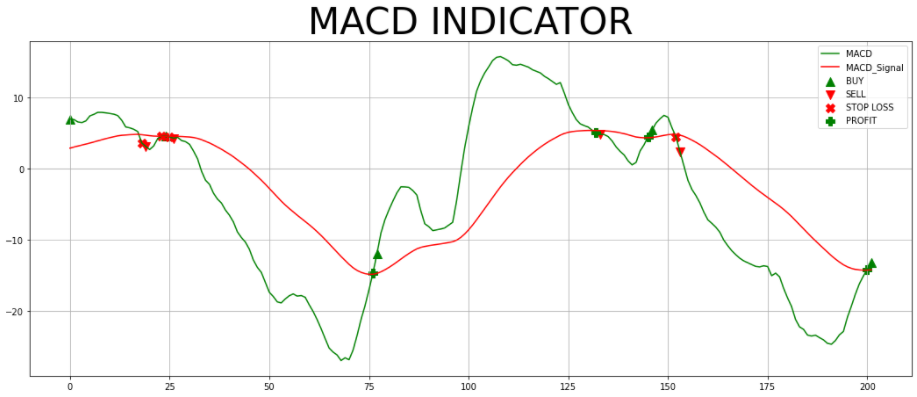
1. When MACD line cross below signal line, wait for and candle and then enter short(sell)
2. When MACD line cross sell signal line, wait for the significant difference/gap between signal line and MACD line then enter short (sell)

When any of the above conditions met, we will sell.

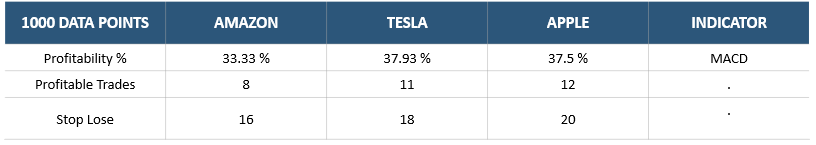
**2.4.3. Results:**



**Figure 2.5. Prediction of Tesla prices using MACD indicator**



**Figure 2.6. Prediction of Tesla prices using MACD indicator**



**Table 2.3. Comparison of Different US Stock market companies using MACD**

**2.5.1. Bollinger Bands indicator based strategy**

Bollinger Bands are envelopes plotted at a standard deviation level above and below a simple moving average of the price. When stock prices continually touch the upper Bollinger Band, the prices are thought to be overbought; conversely, when they continually touch the lower band, prices are thought to be oversold, triggering a buy signal.

**2.5.2. How Bollinger Bands indicator Works:**

The Bollinger Bands indicator working is followed as below:

**Buy signal:**

When Close Prices move above the Upper Bollinger Band then it is indication to buy shares

**Sell signal:**

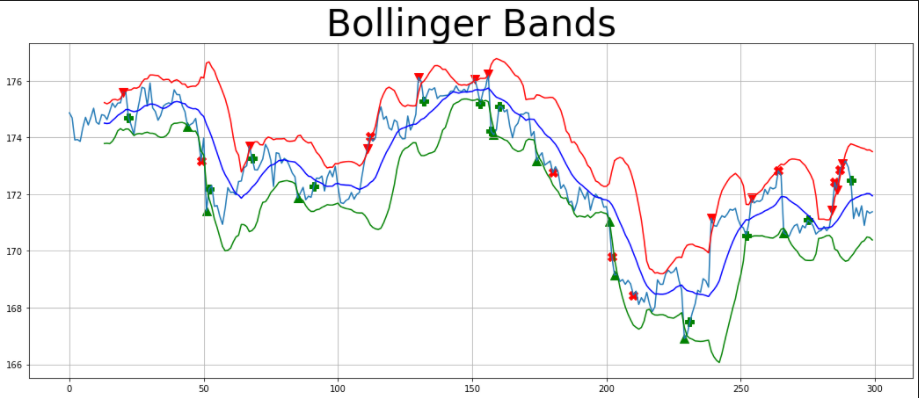
When Close Prices move below the Upper Bollinger Band then it is indication to sell shares.

**2.5.3. Results:**

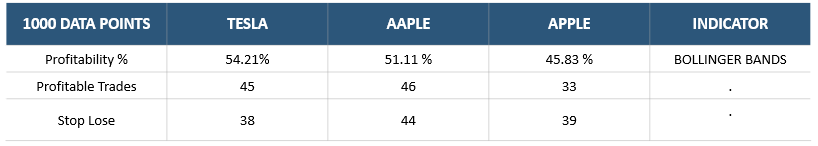
Here we will discuss how we implement RSI strategy in software using built-in tools. The above figure is the result of RSI strategy in software using built-in tools and here we used Tesla data to be analyzed. From figure it is shown that we have Up, down and cross arrows. The Up arrow shows the positions where we have to buy, similarly down arrow is a position where we have to sell and “x” shows where we have to close your orders whether on stop loss or take profit.

****

**Figure 2.7. Prediction of Tesla prices using Bollinger indicator**

****

**Figure 2.8. Prediction of Tesla prices using Bollinger indicator**



**Table 2.4. Comparison of Different US Stock market companies using Bollinger Bands**

**Chapter 3**

**Machine Learning Trading Strategy:**

Stock market prediction is one of the most trending problems currently confronting the world. Stock market prediction is an issue of pervasive interest because the only aim of stock market prediction is to make profit and reduce capital risks. The nature of stock market has always been ambiguous for investors because predicting the performance of a stock market is tough as it considers various factors like the political upheaval, interest rates, current events, exchange rate fluctuations, natural calamities and much more. The challenge is so staggeringly huge that even a small improvement in stock market prediction can lead to huge returns. This paper aims at analyzing this problem in an academic way which provides a different way of prediction on the stock market, instead of predicting the exact price of the stock market which is a risky thing to do we are going to predict the stock market direction. There are only two possible directions for the stock market, i.e., to move up or to move down. In each transaction the number of shares usually traded will have an impact on the value of the shares i.e., greater the number of transactions greater will be the value of shares or vice versa. Generally, there are four ways to analyze the stock market direction. The most basic type of analysis is fundamental analysis which is the way of analyzing the stock market by looking at the company’s economic conditions, reports and future projects. The second and most used technique as describe by Robert Strong is technical analysis which is the way of predicting the stock market by looking at the stock market price charts and analyzing its previous prices. The third and most advance technique is machine learning based analysis which analyzes the market with less human interaction. Machine learning finds the patterns in historical data based on which it tries to predict the stock market prices in the future. The fourth technique is called sentimental based analysis which is the way of analyzing the stock market prices by sentiments of other people like activity on Twitter and financial news on websites.

**3.1 METHODOLOGY**

The methodology we are going to follow in this paper to predict the stock market prices is shown in Figure 1.

**FIGURE 3.1.** **Flow chart of prediction system**

**3.1.1. IMPORTING AND VISUALIZING DATASET**

The dataset we used in our work is imported from the Alpaca broker. We used API communication to import data from alpaca to our python Jupiter notebook and visualize it through Plotly.

**3.1.2. DATA PREPROCESSING**

Data preprocessing is one of the most important step in our algorithm before doing any further processing on stock data, it is the data quality that matters the most, data with noise will reduce the accuracy of an ML algorithm and make the predictions wrong. In this step, we remove unnecessary columns, handle missing values and clean data from unnecessary noise.

**3.1.3.** **INPUT FEATURES AND OUTPUT VARIABLES**

Input feature variables are a set of variables given to the machine learning algorithm with the help of which it can predict stock market direction. Feature variables used in this research is shown in Table 3.

**TABLE 3.1.** **Selected** **Feature variables.**

|  |  |  |
| --- | --- | --- |
| **Feature Variables** | **Parameters** | **Mathematical Equations** |
| RSI indicator | Time period= 14 | UP=Up price change  DW=Down price change |
| SMA Indicator | Time period= 50 |  |
| ADX Indicator | Time period= 20 |  |
| Volume | n/a | n/a |
| Correlation | Time period= 24 | n/a |
| Previous  (Open –Close) | n/a |  |
| Previous (Close – High) | n/a |  |

Output variables are those variables that have been given to machine learning algorithms to predict. We have used two output variables in this work which are move up and move down.

**3.1.4.** **OVERFITTING TESTS**

We have conducted two overfitting tests on feature, one is the correlation test and the other one is a stationary test. Correlation test has been conducted to check the correlation between two or more feature variables and stationary test has been conducted to check the stationary of the feature variables. Stationary variables are those variables that do not change over time both are undesirable phenomena.

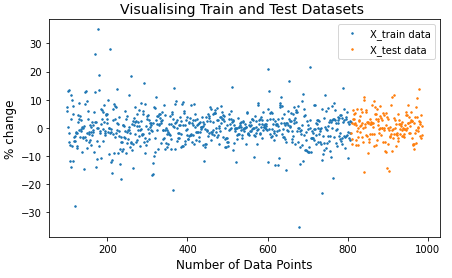
**3.1.5. DATA NORMALIZATION**

Data normalization is the process of fixing the data into a specific range. In our work, we have normalized our feature variables because it is often possible that if feature variables are not normalized, the column with high values should be given more priority in predictions rather than lower ones. The equation for data normalization is in Equation 1.

(1)

**3.1.6. TEST AND TRAIN DATA**

Splitting test and train data is a technique to divide the data into two parts test and train data. Train data has been fed to the machine learning algorithm for learning the patterns inside the data and test data is the unseen data given to the ML algorithm to make predictions. In our work, we split 80% of the data into the training part and 20% of the data into the test part as shown in Figure 2.



**FIGURE 3.2. Splitting test and train data.**

The stock market data ranges are shown in Table 4.

**TABLE 3.2.** **Data ranges of Tesla and Microsoft**

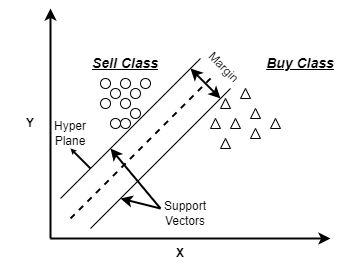
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Time Frame** | **Data Ranges** | **Start Date** | **End Date** | **No of days** |
| **15 min** | Total data | 2021-10-27 | 2021-12-01 | 35 days |
| Training data | 2021-10-27 | 2021-11-23 | 27 days |
| Test data | 2021-11-24 | 2021-12-01 | 7 days |
| **1 day** | Total data | 2016-01-01 | 2022-01-01 | 2192 days |
| Training data | 2016-01-01 | 2020-10-10 | 1744 days |
| Test data | 2020-11-16 | 2022-01-01 | 411 days |

**3.1.7. MACHINE LEARNING ALGORITHM**

In this study, we use 8 machine learning methods (Support Vector Machine, Naïve Bayes, Decision Tree, K-Nearest Neighbors, Logistic Regression, Random Forest, Adaboost and XGboost).

**1) SVM (SUPPORT VECTOR MACHINE)**

The basic idea of SVM is to create a hyperplane that separates the data into classes and the support vectors are data points that are closer to the hyperplanes and influence the position and orientation of the hyperplanes. Figure 3 shows the illustration of SVM.



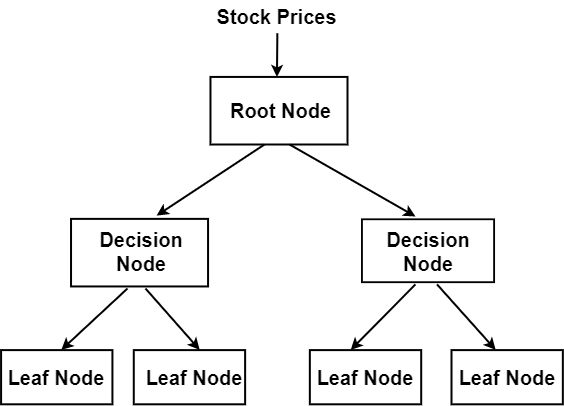
**FIGURE 3.3.** **Schematic illustration of SVM**

We used RBF kernel in our studies which is shown in Equation 2.

(2)

**2) DECISION TREE**

Decision tree is a tree structured model which classifies data by first passing through the root node after passing through the root node the population starts dividing according to various features. Figure 4 shows the schematic illustration of the model.



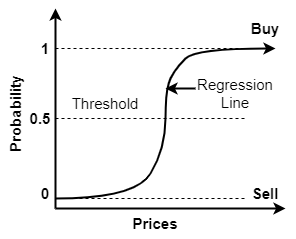
**FIGURE 3.4**. **Schematic illustration of Decision tree**

**3) LOGISTIC REGRESSION**

Logistic regression is similar to linear regression but it solves more complex problems because of sigmoid function. Sigmoid function converts independent variable into the expression of probability. Sigmoid function is shown in Equation 3.

(3)

X is independent variable and e is Euler constant. Figure 5 shows the illustration of logistic regression.



**FIGURE 3.5.** **Schematic illustration of Logistic regression**

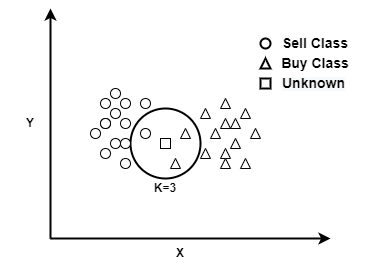
**4) NAÏVE BAYES**

Naive Bayes is a classification algorithm for binary or multiclass problems. It is based on bayes theorem which is used to determine the conditional probability of events. Naive bayes formula is shown in Equation 4.

(4)

**5) KNN (K NEAREST NEIGHBOR)**

It is also called lazy learner because it doesn’t train at all when we supply the training data to it and that why it is one of the fastest and simple ML algorithm. Figure 4 shows the schematic illustration of KNN.



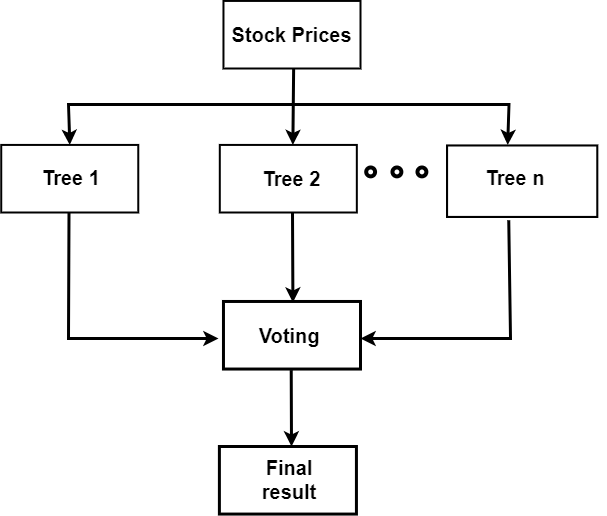
**FIGURE 3.6.** **Schematic illustration of KNN.**

We used Minkowski distance formula in our studies which is shown in Equation 5.

(5)

**6) RANDOM FOREST**

Random Forest uses several decision trees to make a prediction. It uses bagging and features randomness when building each tree, trying to create an uncorrelated forest of trees. Figure 7 shows the schematic illustration of model.



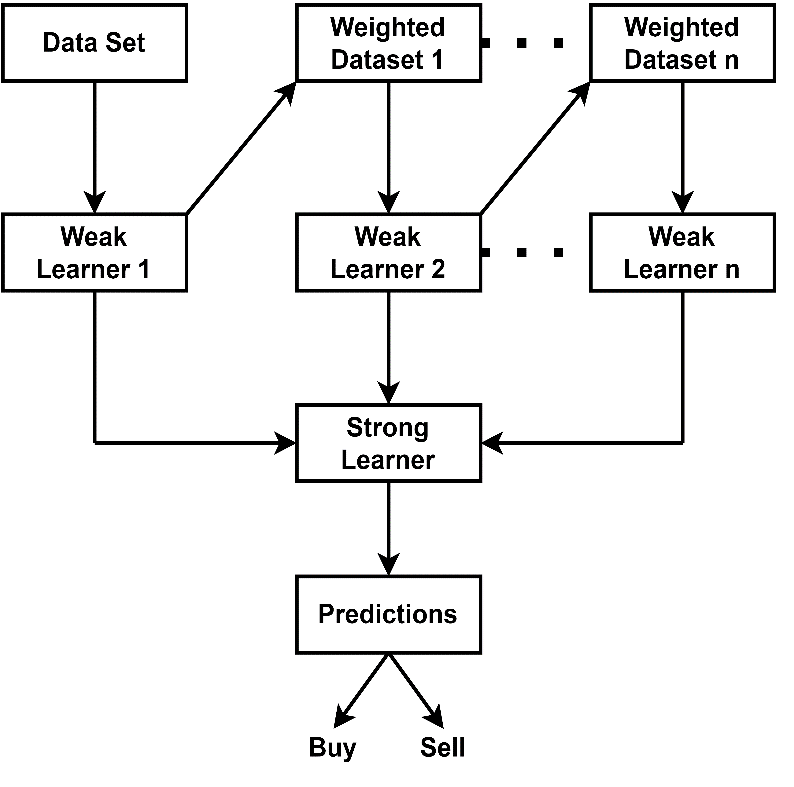
**FIGURE 3.7.** **Schematic illustration of Random Forest.**

We used Gini Criterion in our studies which is shown in Equation 6.

(6)

**7) ADABOOST**

Adaboost combines multiple weak learning models into a single strong learning model. It was created to increase the efficiency of binary classifiers. Adaboost learns from the mistakes of weak classifiers and turn them into strong ones. Figure 8 shows the schematic illustration of model.



**FIGURE 3.8.** **Schematic illustration of Adaboost.**

The formula for strong learner is shown in Equation 7.

(7)

**8) XGBOOST**

XGboost is an advanced version of gradient boosting method. The main aim of this algorithm is to increase the speed and efficiency of model performance. It supports parallelization by creating trees parallelly there is no sequential modeling in this algorithm. Mathematical equation of the model is shown in equation (8).

(8)

**H. CLASSIFICATION METRICES**

After making predictions on stock market data classification metrics are employed to evaluate the performance of the machine learning algorithms. Some strong metrics are selected for classification like Accuracy, ROC-AUC, F1\_score and time of execution. The equations for Accuracy and F1\_Score are shown in Equation 9-10.

(9)

(10)

**I. MODELS PARAMETERS**

Adjustable parametric settings are used to achieve the maximum predictive capacity out of each model. In this paper after repeated experiments optimal parametric settings are selected for each model as shown in Table 5.

**TABLE 3.3.** **Optimal parametric settings for each model**

|  |  |  |
| --- | --- | --- |
| **Models** | **Parameters** | **Values** |
| SVM | C | 1 |
| Kernel | RBF |
| Degree | 3 |
| Decision Tree | Criterion | Gini |
| Random State | 30, 20, 10 |
| Min Sample Split | 2, 3, 4, 5, 6, 7 |
| Logistic Regression | Solver | 1bfgs |
| Algorithm | Euclidean distance |
| Max iteration | 100 |
| Naive Bayes | C | 1 |
| Algorithm | Gaussian |
| KNN | N neighbors | 20, 30, 40, 50 |
| Leaf size | 30 |
| Metric | Minkowski |
| Random Forest | N estimators | 40, 60, 80, 100 |
| Criterion | Gini |
| Random State | 30, 20, 10 |
| Min Sample Leaf | 4 |
| Min Sample Split | 2, 3, 4, 5, 6, 7 |
| Adaboost | N estimator | 40, 60, 80, 100 |
| algorithm | SAMME.R |
| Learning Rate | 1 |

**3.2. EXPERIMENTAL RESULTS**

Based on extensive experimental works by deeming the approaches, the following outcomes are obtained:

In the comparison 1 we have predicted the stock market for 15 minutes time frame as we can see in the Table 6-7 that random forest and Adaboost gives the highest accuracy of (90% and 89.96%) and (91.61% and 91.61%) while KNN and Navies bayes gives the least accuracy with (82.35% and 83.7%) and (83.56% and 82.21) for Microsoft and Tesla stocks. As random forest gives the highest accuracy so its predictions are illustrated in graphical manner in Figure 13-14.

**TABLE 3.4.** **Stock Prediction for Microsoft Stock on (15 min time frame)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameters** | **Prediction Models** | | | |
|  | Decision Tree | Logistic Regression | KNN | Naive Bayes |
| Accuracy | 0.8615 | 0.8823 | 0.8235 | 0.8373 |
| F1\_score | 0.8601 | 0.8852 | 0.8251 | 0.8350 |
| ROC-AUC | 0.8618 | 0.8834 | 0.8245 | 0.8387 |
|  | Random Forest | ADABoost | SVM | XGBoost |
| Accuracy | 0.9000 | 0.8996 | 0.8754 | 0.8961 |
| F1\_score | 0.9000 | 0.8950 | 0.8751 | 0.9010 |
| ROC-AUC | 0.9068 | 0.8998 | 0.8757 | 0.8966 |

**TABLE 3.5.** **Stock Prediction for Tesla Stock on (15 min time frame)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameters** | **Prediction Models** | | | |
|  | Decision Tree | Logistic Regression | KNN | Naive Bayes |
| Accuracy | 0.9093 | 0.9060 | 0.8356 | 0.8221 |
| F1\_score | 0.9102 | 0.9055 | 0.8354 | 0.8253 |
| ROC-AUC | 0.9105 | 0.9052 | 0.8336 | 0.8243 |
|  | Random Forest | ADABoost | SVM | XGBoost |
| Accuracy | 0.9161 | 0.9161 | 0.8791 | 0.9127 |
| F1\_score | 0.9155 | 0.9202 | 0.8800 | 0.9100 |
| ROC-AUC | 0.9165 | 0.9168 | 0.8781 | 0.9128 |

In the comparison 2 we have predicted the stock market for 1-day time frame as we can see in the Table 8-9 that support vector machine and XGboost algorithm gives highest accuracy of (85.86% and 85.51%) on Microsoft stocks while logistic regression and XGboost algorithm gives highest accuracy of (85.51% and 85.30) on Tesla stocks while navies bayes gives the least accuracy of (77.3% and 73.49%) for Microsoft and Tesla stocks. As logistic regression and support vector machine gives the highest accuracies so their predictions illustrated in graphical manner in Figure 15-16.

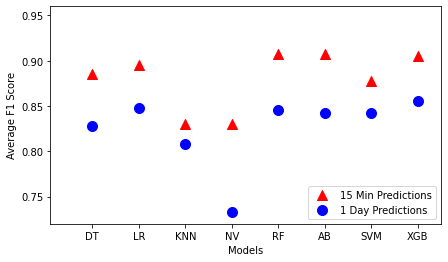
**TABLE 3.6.** **Stock Prediction for Microsoft Stock on (1 day time frame)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameters** | **Prediction Models** | | | |
|  | Decision Tree | Logistic Regression | KNN | Naive Bayes |
| Accuracy | 0.8268 | 0.8445 | 0.8268 | 0.7730 |
| F1\_score | 0.8254 | 0.8411 | 0.8255 | 0.7651 |
| ROC-AUC | 0.8256 | 0.8431 | 0.8253 | 0.7688 |
|  | Random Forest | ADABoost | SVM | XGBoost |
| Accuracy | 0.8410 | 0.8445 | 0.8586 | 0.8551 |
| F1\_score | 0.8410 | 0.8401 | 0.8600 | 0.8550 |
| ROC-AUC | 0.8410 | 0.8455 | 0.8567 | 0.8547 |

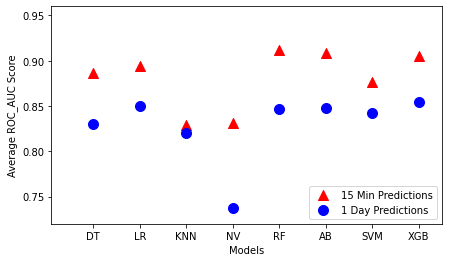
**TABLE 3.7.** **Stock Prediction for Tesla Stock on (1 day time frame)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameters** | **Prediction Models** | | | |
|  | Decision Tree | Logistic Regression | KNN | Naive Bayes |
| Accuracy | 0.8339 | 0.8551 | 0.7915 | 0.7349 |
| F1\_score | 0.8300 | 0.8550 | 0.7912 | 0.7010 |
| ROC-AUC | 0.8358 | 0.8577 | 0.7949 | 0.7050 |
|  | Random Forest | ADABoost | SVM | XGBoost |
| Accuracy | 0.8515 | 0.8468 | 0.8268 | 0.8530 |
| F1\_score | 0.8511 | 0.8453 | 0.8251 | 0.8552 |
| ROC-AUC | 0.8513 | 0.8497 | 0.8282 | 0.8545 |

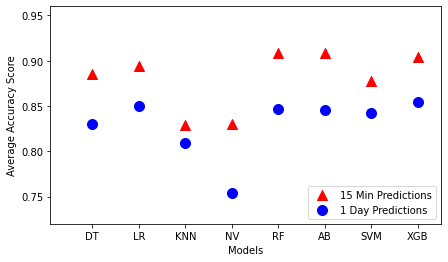
In the comparison 3 we have compared and observe that all evaluated classification metrics are generally greater for 15-min time frame as compare to 1-day as shown in Figure 10-12 that is because of the impacting factors like the political upheaval, interest rates, current events,



**FIGURE 3.9.** **Comparative schematic of average F1\_Score for 15-min and 1-day predictions**



**FIGURE 3.10.** **Comparative schematic of average ROC\_AUC for 15-min and 1-day predictions**



**FIGURE 3.11.** **Comparative schematic of average accuracy for 15-min and 1-day predictions**

exchange rate fluctuations and natural calamities which effect long-time predictions more which causes an uneven class distribution and poor distinguishing between classes which causes decrease in the model accuracy.

In the comparison 4 executional time of algorithms are compared where we can observe from Table 10 that Decision Tree and Logistic Regression has the lowest latency in prediction making while KNN and Naïve Bayes has the lowest latency for training but Naïve Bayes is the only model with overall low latency. Indeed, the running time of all top predictors is more as compare to other algorithms.



**FIGURE 3.12.** **Graphical illustration of Random Forest predictions on Microsoft stocks for (15-min time frame)**



**FIGURE 3.13.** **Graphical illustration of Random Forest predictions on Tesla stocks for (15-min time frame)**



**FIGURE 3.14.** **Graphical illustration of SVM predictions on Microsoft stocks for (1-day time frame)**



**FIGURE 3.15.** **Graphical illustration of Logistic regression predictions on Tesla stocks for (1-day time frame**

**Chapter 4**

**High Frequency Trading Strategy:**

In this chapter we will briefly discuss the implementation of our designed strategies in hardware (FPGA). We will discuss the complete implementation step by step from making pseudo-code to the final result and then we will compare that result with the Software results. One important thing to note that we choose three popular indicators to implement in hardware. We propose a system-on-chip (SoC) design for the implementation of a High Frequency Trading (HFT) system. To combine the flexibility and scalability of software with the performance of hardware, we used Xilinx Zynq-7000 programmable SoC XC7Z020-CLG484-1. The processing system (PS) in our platform is an ARM based microprocessor system whereas the programmable logic (PL) developed for our accelerator is a 28 nmArtix-7 based reconfigurable logic. We implement three different momentum indicators, Relative Strength Index (RSI), Moving Average Convergence/Divergence (MACD) and Aroon to predict stock market prices with desirable accuracy and speed. Two major cryptocurrencies (Bitcoin and Ethereum) are considered for experimental evaluations. Technical indicators are compared for 1-day time frame using ZYNQ SOC platform over 10-Gigabit Ethernet. The evaluated results show that MACD is the most accurate and low latency momentum indicator while Aroon is the least accurate and high latency indicator momentum indicator for Bitcoin and Ethereum datasets.

**4.1. Methodology and implementation of three indicators in hardware (FPGA):**

Here we are going to implement our MACD strategy in Hardware. For MACD calculation as discussed previously, you have found 12-period EMA, 26- period EMA and by subtracting 26-period EMA from 12-period EMA, the MACD line will find. For signal line then you have to take 9-period EMA of MACD. For hardware implementation first you have to code in Vivado software and then through data cable you can upload in FPGA development board. For coding in Vivado you have to follow the following steps:

* **Pseudo-code**
* **Top**
* **Controller (ASM)**
* **Data-path**

Let’s briefly discuss every step.

**4.1.1. Pseudo-code:**

Pseudo-code means a step-by-step procedure for solving a problem. For any problem you are going to solve, first you are thinking about his procedure of solving which is actually his pseudo-code. Pseudo-code is a detailed yet readable description of what a computer program or algorithm must do, expressed in a formally-styled natural language rather than in a programming language.

**Algorithm 4.1.** detailed working of the MACD model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Start** | | | |
| **1:** |  | EMAn1 = 12-day EMA of closing price | | |
| **2:** |  | EMAn2 = 26-day EMA of closing price | | |
| **3:** |  | MACDSIGn3 = 9-day EMA of MACD | | |
| **4:** |  | MACDn1, n2 (t) = EMAn1 (t) -EMAn2 (t) | | |
| **5:** |  | MACD\_Diff (t) =  MACDn1, n2 (t) -MACDSIGn3 (t) | | |
| **6:** |  |  | **if** (MACD\_Diff (t) > 0) | |
| **7:** |  |  |  | Order\_Sig = Buy |
| **8:** |  |  | **else** | |
| **9:** |  |  |  | Order\_Sig = Sell |
| **10:** |  |  | **end if** | |
| **11:** | **END** | | | |

**Algorithm 4.2.** detailed working of the RSI model

|  |  |  |  |
| --- | --- | --- | --- |
| **Input:** | Declare i ∈ Integer  Declare j ∈ Integer  Declare a ∈ Integer (length of file) | | |
| **Output:** | RSIj | | |
|  | **Start** | | |
| **1:** | i = 1 | | |
| **2:** | **REPEAT** | | |
| **3:** |  | **if** (Ci – Ci-1 > 0) then | |
| **4:** |  |  | close\_upi = Ci – Ci-1 |
| **5:** |  | **else** | |
| **6:** |  |  | close\_upi = 0 |
| **7:** |  | **end if** | |
| **8:** |  | **if** (Ci – Ci-1 < 0) then | |
| **9:** |  |  | close\_downi = Ci – Ci-1 |
| **10:** |  | **else** | |
| **11:** |  |  | close\_downi = 0 |
| **12:** |  | **end if** | |
| **13:** | **UNTIL** i = a | | |
| **14:** |  | **j** = 1 | |
| **15:** |  | **REPEAT** | |
| **16:** |  |  | = |
| **17:** |  |  | = |
| **18:** |  |  | j = j+1 |
| **19:** |  | **UNTIL** j=a | |
| **20:** | **END** | | |

**Algorithm 4.3.** detailed working of the Aroon

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Start** | | | |
| **1:** |  | **REPEAT** | | |
| **2:** |  |  | **if** (H\_Pi > H\_Pk-1) then | |
| **3:** |  |  |  | H\_Pi = H\_Pk-1 |
| **4:** |  |  |  | C = 0 |
| **5:** |  |  | **else** | |
| **6:** |  |  |  | C++ |
| **7:** |  |  | **end if** | |
| **8:** |  |  | **if** (K == Current\_Value) then | |
| **9:** |  |  |  | Period (K) = C |
| **10:** |  |  |  |  |
| **11:** |  |  |  | C1= 0 |
| **12:** |  |  | **else** | |
| **13:** |  |  |  | C1++ |
| **14:** |  |  | **end if** | |
| **15:** |  |  | **if** (K == Current\_Value) then | |
| **16:** |  |  |  | Period\_1 (K) = C1 |
| **17:** |  |  |  |  |
| **18:** |  |  | end if | |
| **19:** |  | **UNTIL** K=a | | |
| **20:** | **END** | | | |

**4.1.2. Top level:**

After making pseudo-code and block diagram, the next step is to make top level for coding in VHDL language. Top level shows the all the input and output signals of whatever you are going to design.



**Figure 4.1.** **Top Level Design Interface of MACD Indicator**



**Figure 4.2. Top Level Design Interface of RSI Indicator**



**Figure 4.3. Top Level Design Interface of Aroon Indicator**

**4.1.3. Controller:**

As from name it clear that controller is used for generating the control signals which is used for deciding which operation be performed on which time. Actually, it controller all the operations which is done inside data-path. The following is the controller for our strategy:



**Figure 4.4. ASM of MACD Indicator**

**Figure 4.5. ASM of RSI Indicator**



**Figure 4.6. ASM of Aroon Indicator**

**4.1.4. Data-path:**

A data-path is a collection of functional units such as arithmetic logic units or multipliers that perform data processing operations, registers, and buses. In data-path the actual operations are executing by the application of control signals. Data-path consist up of register, multiplexer, counters, flip flop, rom etc. In data-path a specific operation is executing by the arrival of specific control signals.



**Figure 4.7. Datapath of MACD Indicator**



**Figure 4.8. Datapath of RSI Indicator**

**Figure 4.9. Datapath of Aroon Indicator**

**4.2. Experimental Results:**

Let’s discuss some of the best dependent parameters for each indicator which are selected for optimal results as shown in Table 2. In MACD, we have three dependent parameters which are EMA12, EMA26 and EMA9 and its time periods are 12, 26 and 9 respectively. In RSI, we have two dependent parameters which are Average gain and Average loss and its time periods are 14, 14 respectively. In Aroon, we also have two dependent parameters which are Highest value in period and Lowest value in period and its time periods are 14, 14 respectively.

**TABLE 4.1.** **Parametric settings for each technical indicator**

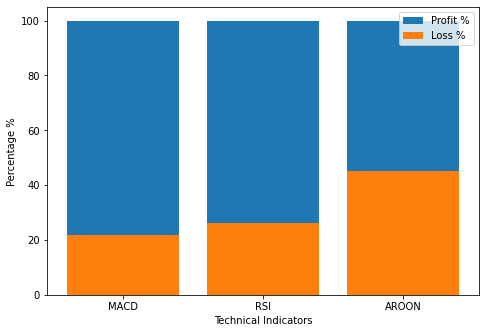
|  |  |  |  |
| --- | --- | --- | --- |
| **Technical Indicator** | **Signals** | **Dependent**  **parameters** | **Time period** |
| MACD | MACD | EMA 12 | 12 |
| EMA 26 | 26 |
| MACD signal | EMA 9 | 09 |
| RSI | RS | Average Gain | 14 |
| Average Loss | 14 |
| Aroon | Aroon up | Highest value in Period | 14 |
| Aroon Down | Lowest value in Period | 14 |

In Table 3, we discuss the profitably rate of different momentum technical indicators using different crypto currencies such as Bitcoin and Ethereum and conclude that how our algorithmic trading strategies react to the different crypto currencies dataset. The profitably rate of different momentum technical indicators are calculated as

**TABLE 4.2. Cryptocurrency prediction using momentum technical indicators**

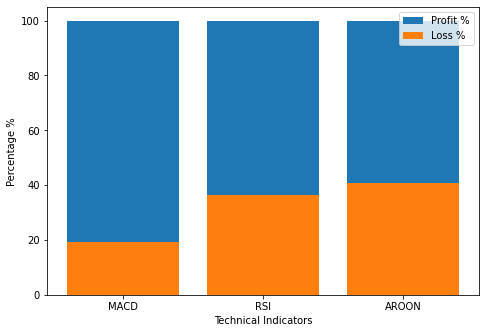
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Crypto**  **currency** | **Technical Indicator** | **Profit in trades** | **Loss in**  **trades** | **Profitably rate %** |
| Bitcoin | MACD | 19 | 5 | 79.1 |
| RSI | 14 | 5 | 73.68 |
| Aroon | 11 | 9 | 55 |
| Ethereum | MACD | 21 | 5 | 80.7 |
| RSI | 14 | 8 | 63.63 |
| Aroon | 13 | 9 | 59.09 |

To consider the performance of different momentum technical indicators and conclude that MACD momentum indicator gives us the best profitability rate on both of the datasets i.e., Bitcoin and Ethereum are 79.1% and 80.7% respectively. FIGURE 1 shows the profitability rate of MACD, RSI and Aroon on the dataset of Bitcoin.



**FIGURE 4.10.** **Profitability % of technical indicators on Bitcoin cryptocurrency**

Similarly, FIGURE 2 shows the profitability rate of MACD, RSI and Aroon on the dataset of Ethereum but if we observed that in this case MACD gives us quite better performance as comparison to RSI and Aroon.



**FIGURE 4.11.** **Profitability % of technical indicators on Ethereum cryptocurrency**

Consider the dataset of Bitcoin crypto currency, MACD takes total of 24 trades out of them 19 are profitable and 5 trades are in loss respectively. FIGURE 3 shows the graphical illustration of MACD technical indicator prediction on Bitcoin (BTC) However, on Ethereum dataset graphically illustrated in FIGURE 6 it works more accurate because it takes total of 26 trades out of them 21 are profitable and 5 trades are in loss. Similarly, RSI profitability rate on the datasets of Bitcoin and Ethereum are 73.68% and 63.63% respectively and it takes total of 19 trades out of them 14 are profitable and 5 trades are in loss respectively. FIGURE 4 shows the graphical illustration of RSI technical indicator prediction on Bitcoin (BTC). However, on Ethereum dataset graphically. illustrated in FIGURE 7 it works less accurate than BTC dataset because it takes total of 22 trades out of them 14 are profitable and 8 trades are in loss.

Similarly, Aroon which gives us the least profitability rate on the datasets of Bitcoin and Ethereum are 55% and 59.09% respectively and it takes total of 20 trades out of them 11 are profitable and 5 trades are in loss on the Bitcoin dataset respectively. FIGURE 5 shows the graphical illustration of Aroon technical indicator prediction on Bitcoin (BTC). However, on Ethereum dataset graphically illustrated in FIGURE 8 it takes total of 21 trades out of them 13 are profitable and 9 trades are in loss.

**5. References**

1. M. Karpe, “An overall view of key problems in algorithmic trading and recent progress” arXiv, 2020, <https://arxiv.org/abs/2006.05515>.
2. [Robert A. Strong](https://www.amazon.com/Robert-A-Strong/e/B001IQUN3E/ref=dp_byline_cont_book_1), “Practical Investment Management” published by South-Western College Pub; 4th edition, March 9, 2006.
3. G. S. Atsalakis and K. P. Valavanis, “Surveying stock market forecasting techniques – Part II: Soft computing methods” Expert Systems with Applications, vol. 36, no. 3, pp. 5932–5941, Apr. 2009.
4. J. Ayala, M. G-Torres, J. L. V. Noguera, F. G-Vela, and F. Divina, “Technical analysis strategy optimization using a machine learning approach in stock market indices, Knowledge-Based Systems,” Volume 225, 2021.
5. T. Segal, “Fundamental Analysis” Available online https://www.investopedia.com/terms/f/fundamentalanalysis.asp. [Accessed: 01-04-2022].
6. R. F. Oğuz, Y. Uygun, M. S. Aktaş and İ. Aykurt, "On the Use of Technical Analysis Indicators for Stock Market Price Movement Direction Prediction," 2019 27th Signal Processing and Communications Applications Conference (SIU), 2019, pp. 1-4, doi: 10.1109/SIU.2019.8806422.
7. G. Jariwala, H. Agarwal and V. Jadhav, "Sentimental Analysis of News Headlines for Stock Market," 2020 IEEE International Conference for Innovation in Technology (INOCON), 2020, pp. 1-5, doi: 10.1109/INOCON50539.2020.9298333.
8. Y. Ding, N. Sun, J. Xu, P. Li, J. Wu, and S. Tang, "[Research on Shanghai Stock Exchange 50 Index Forecast Based on Deep Learning](https://ideas.repec.org/a/hin/jnlmpe/1367920.html)," [Mathematical Problems in Engineering](https://ideas.repec.org/s/hin/jnlmpe.html), Hindawi, vol. 2022, pages 1-9, March.
9. Y. Lin, L. Shancun, Y. Haijun, and W. Harris, “Stock Trend Prediction Using Candlestick Charting and Ensemble Machine Learning Techniques with a Novelty Feature Engineering Scheme.” IEEE Access 9 (2021): 101433-101446.
10. A. Politis, K. Doka and N. Koziris, "Ether Price Prediction Using Advanced Deep Learning Models," 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 2021, pp. 1-3, doi: 10.1109/ICBC51069.2021.9461061.
11. M. Nabipour, P. Nayyeri, H. Jabani, S. S. and A. Mosavi, "Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data; a Comparative Analysis" IEEE Access, vol. 8, pp. 150199-150212, 2020, doi: 10.1109/ACCESS.2020.3015966.
12. M. Mudassir, S. Bennbaia D. Ünal, and M. Hammoudeh, “Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach,” Neural Computing and Applications. 10.1007/s00521-020-05129-6.
13. K. Ullah and M. Qasim, "Google Stock Prices Prediction Using Deep Learning," 2020 IEEE 10th International Conference on System Engineering and Technology (ICSET), 2020, pp. 108-113, doi: 10.1109/ICSET51301.2020.9265146.
14. X. Pang, Y. Zhou, P. Wang, W. Lin, and V. Chang, ‘‘An innovative neural network approach for stock market prediction,’’ J. Supercomputer, vol. 76, no. 3, pp. 2098–2118, Mar. 2020.
15. Y. Li, “Bitcoin price forecasting method based on CNN-LSTM hybrid neural network model,” The Journal of Engineering, vol. 2020, pp. 344–347(3), July 2020.
16. X. Ying, y. Cuijuan, P. Shaoliang, and N. Yusuke, “A hybrid two-stage financial stock forecasting algorithm based on clustering and ensemble learning” in Applied Intelligence, 50(11):3852–3867, 2020.
17. F. Valencia, A. Gómez-Espinosa, and B. Valdés-Aguirre, “Price movement prediction of cryptocurrencies using sentiment analysis and machine learning,” Entropy, vol. 21, no. 6, 2019.
18. D. Shah, W. Campbell and F. H. Zulkernine, "A Comparative Study of LSTM and DNN for Stock Market Forecasting," 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 4148-4155, doi: 10.1109/BigData.2018.8622462.
19. M. U. Gudelek, S. A. Boluk and A. M. Ozbayoglu, "A deep learning based stock trading model with 2-D CNN trend detection," 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017, pp. 1-8, doi: 10.1109/SSCI.2017.8285188.
20. [20] S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon and K. P. Soman, "Stock price prediction using LSTM, RNN and CNN-sliding window model" 2017 International Conference on Advances in Computing, Communic

-ations and Informatics (ICACCI), 2017, pp. 1643-1647, doi: 10.1109/ICACCI.2017.8126078.

1. W. Kristjanpoller, M. C. Minutolo, “Forecasting volatility of oil price using an artificial neural network-garch model” in [Expert Systems with Applications](https://dl.acm.org/toc/exwa/2016/65/C), [Volume 65](https://www.sciencedirect.com/journal/expert-systems-with-applications/vol/65/suppl/C), 15 December 2016, Pages 233-241
2. https://csimarket.com/stocks/growthrates.php. [Accessed: 01-04-2022].
3. Y. Chen and Y. Hao, ‘‘A feature weighted support vector machine and K-nearest neighbor algorithm for stock market indices prediction’’ Expert Syst. Appl., vol. 80, pp. 340–355, Sep. 2017.
4. R Sai VenkatRamana, M Reddi DurgaSree, Ramakrishna Gandi and A. Sankar Reddy, “Comparative Analysis of Stock Market Prediction based on Twitter Stock Data” International Journal of Computer Applications 174(29):22-26, April 2021.

***The End***