BINDURA UNIVERSITY OF SCIENCE EDUCATION

FACULTY OF SCIENCE

DEPARTMENT OF STATISTICS AND MATHEMATICS

HIGH-FREQUENCY VOLATILITY FORECASTING OF CURRENCY PAIRS USING MACHINE LEARNING. A TICK-BY-TICK BID-ASK FRAME WORK

BY

Takudzwa J. Mungani

A PROJECT SUBMITTED IN PARTIAL FULLFILMENT OF THE REQUIREMENTS OF BACHELOR OF SCIENCES HONOURS DEGREE IN STATISTICS AND FINANCIAL MATHEMATICS

SUPERVISOR: DR. T.W. MAPUWEI

(2026)

APPROVAL FORM

This is to certify, that the research project is the result of my own research work and has not been copied or extracted from past sources without acknowledgement. I hereby declare that no part of it has been presented for other degree in this university or elsewhere.



05/06/2026

Takudzwa j Mungani …… …….

(B222599B) signature date



Certified by: **09/06/2026**

Dr. T.W. Mapuwei …….. ……..

Supervisor signature date



18/08/2026

Dr. Magodora …….. ……

Chairperson signature date

DEDICATION

I dedicate this research to my beloved parents, Mr. and Mrs. Mungani, your sacrifices, love and unwavering belief in me have been my foundation. Every step of this work carries your encouragement

ACKNOWLEDGEMENT

Firstly, I thank God for the gift of life, wisdom and strength without which this work would not have been possible. I extend my deepest gratitude to my supervisor, Dr. Mapuwe for his exceptional guidance, and unwavering patience. His mentor-ship has been invaluable not only in shaping this dissertation but also in developing my growth as a researcher.

To my parents, Mr. and Mrs. Mungani, thank you for the love , emotional support and financial support , and mostly for believing in me even when I doubted myself. Your sacrifices have laid foundation for every achievement in my life. To my partner Tawananyasha I am forever grateful for your understanding encouragement and love . you stood by me through every high and low and your presence gave me strength to persevere. I also wish to thank the Department of Statistics and Mathematics at Bindura University of science Education for providing the academic environment and resources necessary for this research. Special thanks to the staff and my fellow students who contributed directly or indirectly to this project. Finally, I acknowledge all the open- source developers whose libraries (TensorFlow, Scikit-learn, Streamlit and others) made the practical implementation of this work possible.

This dissertation is a product of collective support and I am deeply grateful to everyone who played a part.

ABSTRACT

This research develops and evaluates a high-frequency volatility forecasting framework for a currency pair of EUR/USD using a tick-by-tick bid-ask framework and machine learning. The study addresses the fundamental limitation of classical econometrics models in capturing the non-linear, heavy-tailed and noisy microstructure of tick-by-tick data. The Tick data for EUR/USD were obtained from HistData.com for the period of 1 to 30 January 2026, yielding approximately 1.5 million observations. After cleaning and sampling, five micro-structure features were engineered that is mid price, lagged log return, spread, hour sine, and hour cosine. The response variable was realized volatility over the next 50 ticks that's approximately 10 to 20 seconds. Four deep learning neural networks were trained that is LSTM, GRU, Transformer, QRRN. A weighted ensemble combining all four models was constructed and compared against a GARCH(1,1) benchmark and historical volatility baseline.

The realized volatility series exhibited extreme positive skewness of 34.28 and excess kurtosis of 2118, confirming a heavy-tailed power-law distribution. The autocorrelation function showed strong persistence(lag-1 = 0.9918), and Engle's ARCH test confirmed volatility clustering with $p < 0.001$. The weighted ensemble achieved the lowest MAE of 0.1147% and highest R^2 of 0.403, outperforming the best individual models and dramatically outperforming the GARCH1(1,1) with MAE = 0.8384% an improvement of approximately 7.3 times. Residual diagnostics showed no systematic bias with mean residual = -0.0012% and no significant auto-correlation proved by Ljung-Box with $p = 0.28$. Regime classification showed Low, Normal, Elevated, High and Extreme archived 82% overall accuracy, with the model never confusing Extreme with Low/Normal, making it safe for real-time risk monitoring.

The trained ensemble was deployed in a live Streamlit Robot dashboard accessible at https://huggingface.co/spaces/COSMOCRAT/Volatility_Robot and this is the password to open the Robot (TAKUDZWA MUNGANI).The Robot updates every 100 ms, displaying a volatility gauge, colour-coded alerts and role-specific recommendations for bank risk managers, corporate treasures, retail investors and

traders. The study concludes that deep learning ensembles substantially outperform GARCH at tick level, that microstructute features especially time of day and spread, add significant predictive value and that the Robot provides a reliable, actionable tool for high-frequency risk monitoring.

TABLE OF CONTENTS

CHAPTER1 :INTRODUCTION 8
1.0 Introduction: 8
1.1 Background of the study.....9
1.2 Statement of the Problem 10
1.3 Research Objectives 10
1.3.1 Specific Objectivs Include 11
1.4 Research Questions 11
1.5 Delimitation of the study 11
1.6 Significance of the study 11
1.7 Assumptions of the study 12
1.8 Limitation of the study 12
1.9 Definitions of Key Terms 13
1.10 Conclusion 14
CHAPTER 2: LITERATURE REVIEW.....15
2.0 Introduction.....15
2.1 Theoretical literature 15
2.1.1 Auto-regressive Conditional Heteroskedasticity (ARCH MODEL) 15
2.1.2 Advantages of ARCH models 16
2.1.3 Disadvantages of (ARCH MODEL).....16
2.1.4 Generalized Auto-regressive Conditional Heteroskedasticity (GARCH MODEL).....17
2.1.5 Advantages of GARCH models 18
2.1.6 Disadvantages of GARCH MODEL.....18
2.2 Empirical literature.....19
2.2.1 Auto-regressive Conditional Heteroskedasticity (ARCH MODEL) 19
2.2.2 Generalized Auto-regressive Conditional Heteroskedasticity (GARCH MODEL).....21
2.3 Conceptual Framework 22
2.3.1 Variable of the study.....22
2.3.1.1 Dependent variable:.....22
2.3.1.2 Independent variables:.....22

2.3.2 Relationship Between Variables :.....23
2.3.3 Modelling Approach.....24
2.4.1 Synthesis of literature.....24
2.4.2 Research Gap.....24
2.4.4 The Gap This Study Addresses.....25
CHAPTER 3:RESEACH METHODOLOGY.....27
Introduction.....27
3.2 Research design.....28
3.3 Data Source and Collection methods.....29
3.4 Target population and sampling techniques.....30
3.4.1 Justification:.....31
3.5 Description of variables and expected relationships.....31
3.5.1 Dependent variable.....32
Mid price.....32
Lagged Log Return.....33
3.6 Scaling process.....34
3.7 Analytical Model and Justification.....35
3.7.1 Long Short-Term Memory(LSTM).....35
3.7.1.1 Justification:.....35
3.7.2 Gated recurrent unit(GRU).....36
3.7.2.1 Justification:.....36
3.7.3 Transformer(ATTENTION-BASED).....36
3.7.3.1 Justification:.....36
3.7.4 Quantile Regression Neural Network (QRNN).....36
3.7.4.1 Justification.....37
3.7.5 Ensemble method.....37
3.7.6 summary.....37
3.8 Model Pre-Tests and Diagnostic Checks.....37
3.8.1 Auto-correlation and Partial Auto-correlation.....38
3.8.2 Multicollinearity.....38
3.8.3 Conditional Heteroskedasticity (ARCH TEST).....39
3.8.4 Outlier Detection.....39
3.8.5 Normality Test.....40
3.8.6 Summary.....41

3.9 MODEL FIT AND VALIDATION TESTS....41
3.9.1 Goodness-of-fit metrics....42
3.9.2 Out-Of-Sample Validation....42
3.9.3 Ensuring Model Reliability....43
3.9.4 Summary....43
3.10 Ethical Consideration and Reproducibility....44
3.10.2 Reproducibility....44
3.10.3 SUMMARY....45
3.10.4 CHAPTER CONCLUSION....45
CHAPTER 4:DATA ANALYSIS, PRESENTATION AND DISCUSSION....48
Introduction:....48
4.2 PRELIMINARY ANALYSIS AND DIAGNOSTIC TESTS....49
4.2.1 AUTOCORRELATION....50
4.2.2 Conditional Heteroskedasticity (ARCH TEST)....51
4.2.3 Multicollinearity....52
4.2.4 Outlier Detection....53
4.3 Model Performance Comparisons....54
4.3.1Training summary and convergence....54
4.3.2 Test set and performance....54
4.3.3 Visual diagnostics....55
4.3.4 : Distribution of residuals....57
4.4 Discussion of findings....58
4.4.1 Heavy-Tailed nature of realizwd volatility and implications for modelling....58
4.4.2 Volatility Persistence and Clustering....59
4.4.3 Multicollinearity and Outlier Characteristics....59
4.4.4 Comparative Model Performance....60
4.4.5 Residual Diagnostics and Model Specification....61
4.4.6 Summary of Discussion....61
4.6 Conclusion....61
CHAPTER 5 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS....63
5.1 Introduction....63
5.2 Summary of findings....63
5.2.1 Descriptive and diagnostic Findings....63

5.2.2 Model Performance Findings 64
5..2.3 Residual and Regime Classification Findings 64
5.2.4 Dashboard and Practical Application 5.2.4 Robot_Volatility dashboard and practical Application 65
5.3 Summary of Conclusion and contribution to the field 66
5.3.1 conclusions 66
5.3.2 Contribution to the Field 66
5.4 Recommendations 67
5.4.1 For Practitioners (Risk Managers, Traders, Treasurers and Retail Investors)67
5.4.2 Policy and Regulations 68
5.4.1.1 Role-Specific Actions: 68
5.4.2 For policy and regulations: 68
5.4.3 For Algorithmic Trading System Design 69
5.5 Areas for further Research 69
5.6.1 Key summary points: 70
References 71
Appendix A: Volatility_Robot screenshots 75
AppendixB: Python codes 80

LIST OF TABLES

Table 4.1 DESCRIPTIVE STATISTICS 49
Table 4.2 NORMALITY TEST RESULT 50
Table 4.3 Conditional Heteroskedastic 52
Table 4.4 Variance Inflation Factor 53
Table 4.5 :Outlier Detection Results 54
Table 4.6 :Training outcome 56
Table 4.7 :Test Set Performance Metrics 56

LIST OF FIGURES

Figure 2.1 :Conceptual framework diagram.....30
Figure 4.1 :Distribution analysis of realized volatility 52
Figure 4.2 :Auto-correlation Function (ACF) and Partial Auto-correlation Function (PACF) of Realized Volatility (First 50 Lags).....53
Figure 4.3 :Correlation Matrix Heatmap of the Five Micro-structure Features....54
Figure 4.4 :Box-plot Visualization of Outliers for Each Variable 56
Figure 4.5 :Predicted vs. Actual Volatility (shows first 500 window) 59
Figure 4.6 :Scatter Plot of Ensemble Predicted vs. Actual Realized Volatility (All 42,988 Test Windows) 60
Figure 4.7 :Residual Analysis.....61
Figure 5.1 : The Confusion Matrix 69
Figure A1 : Robot Dashboard Sidebar.....83
Figure A2 : Focasted realized volatility after 50 steps.....83
Figure A3 : recommendations of a Bank manager 84
Figure A4 : recommendations of a Retail investor 85
Figure A5 :recommendations of a Corporate Treasurer 86
Figure A6 :recommendations of a Trader 86

LIST OF ACRONYMS

ACF	Autocorrelation Function
ADF	Augmented Dickey-Fuller (test)
ARCH	Auto-regressive Conditional Heteroskedasticity
BIS	Bank for International Settlements
CSV	Comma-Separated Values
ECB	European Central Bank
EUR	Euro
EUR/USD	Euro / United States Dollar currency pair
FX	Foreign Exchange
GARCH	Generalized Auto-regressive Conditional Heteroskedasticity
GBP	British Pound Sterling
GRU	Gated Recurrent Unit
HFT	High-Frequency Trading
IQR	Interquartile Range
LM	Lagrange Multiplier (test)
LSTM	Long Short-Term Memory
LTV	Loan-to-Value (ratio)
MAE	Mean Absolute Error
MLE	Maximum Likelihood Estimation
ML	Machine Learning
OLS	Ordinary Least Squares
PACF	Partial Autocorrelation Function

QRNN	Quantile Regression Neural Network
R^2	Coefficient of Determination
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
USD	United States Dollar
UTC	Coordinated Universal Time
VaR	Value at Risk
VIF	Variance Inflation Factor

List of symbols

Symbol	Description
Pt	Mid price at tick tt
$Bidt$	Bid price at tick tt
$Askt$	Ask price at tick tt
St	Absolute bid-ask spread at tick tt: Askt−Bidt$Askt−Bidt$
$rt−1$	Lagged-logreturn (from tick t−2$t−2$ to t−1$t−1$)
Ht	Hour of the day (UTC) at tick tt
sin(Ht)	Sine of the hour (circular encoding)
cos(Ht)	Cosine of the hour (circular encoding)
$vt+1:t+50$	Realized volatility over the next 50 ticks (dependent variable)
mid$t$	Midprice = (Bidt+Askt)/2($Bidt+Askt$)/2
ht	Conditional variance at time tt (ARCH/GARCH)
et	Error term (shock) at time tt
$\alpha 0,\alpha 1,\ldots,\alpha q$	ARCH model coefficients
$\beta 1,\ldots,\beta p$	GARCH model coefficients (moving average part)
$\sigma t2$	Conditional variance (volatility squared)
zt	Standardized residual: zt∼N(0,1)zt $\sim N(0,1)$
MAE	Mean Absolute Error

RMSE	Root Mean Square Error
*R*2	Coefficient of determination
VaR95%	Value at Risk at 95% confidence level
VaR99%	Value at Risk at 99% confidence level
ES95%	Expected Shortfall at 95% confidence level
IQR	Interquartile range
VIF	Variance Inflation Factor
LM	Lagrange Multiplier statistic (ARCH test)
ACF(*k*)	Autocorrelation function at lag k
PACF(*k*)	Partial autocorrelation function at lag k*k*

CHAPTER1: INTRODUCTION

1.0 Introduction

The financial environment of the modern day is increasingly widely and this has promoted the use of algorithmic trading and the demand for high-speed and highly accurate models intensifies . This study situates itself at the core of these trends, with a focus on foreign exchange market. To attain risk mitigation and informed trade executions ,forecasting accurate volatility is going to be crucial . However we have traditional econometric models such as GARCH that is used to forecast volatility accurately but hence they often fall short in capturing the , non-linear behavior in data sampled at sub-second interval. (Engle, 1982; Bollerslev, 1986)

This study proposes an innovation framework approach to predict high-frequency volatility in currency markets. It pioneers the use of machine learning specifically Deep learning models to capture these complex, evolving patterns. The research moves beyond conventional midpoint by utilizing tick-by-tick Bid and Ask data. Importantly forecasts with machine learning are integrated within a risk monitoring framework, so as to establishing a direct link between prediction accuracy and trading performance. This wil also delivers practical economic value not ensures statistical relevance only .(Goodfellow et al., 2016) In this study, machine learning serves as the central tool to address and overcome the limitations of classical financial prediction techniques.

This study will demonstrate the potential of machine learning to predict short-term volatility in currency pairs specifically EUR\USD using high-frequency tick data made every second. By exploiting bid and ask prices, the study aims to close the gap between practical trading performance and theoretical volatility modeling.

1.1 Background of the study

The foreign exchange market has become the world's most technologically advanced and the most liquid financial marketplace, with daily trading volumes surpassing trillions of United States dollars (USD)(Bank for international settlements, 2022).

The growth of algorithmic and high-frequency trading system has intensified the demand for accurate real-time volatility forecasting models that can interpret micro-structural data. Volatility is a critical input for risk management, optimizing trading strategies and pricing derivatives (Hull, 2018). Traditional estimation techniques , such as Generalized Auto-regressive Conditional Heteroskedasticity (GARCH) and Stochastic volatility models,for volatility rely on low-frequency data (daily or hourly returns) and often fail . However these methods often fail to capture the pattern of non-linear and noise behaviors in tick level data(Andersen et al., 2001). The continuous inflow of Bid Ask quotes in electronic trading provides a rich data source that accountsbfor liquidity, order flow and transactional cost(O'Hara, 1995) for modeling market micro-structure and turning predicted volatility into actionable trading decisions.

Machine learning becomes the central powerful tool for modeling complex financial phenomena, especially where traditional models are inadequate.Machine learning techniques such as recurrent neural networks (RNNs), long short-term memory (LSTM) architectures have revealed outstanding performance in forecasting problems(Hochreiter and Schmidhuber, 1997). Integrating machine learning approaches into high-frequency financial forecasting allows the potential to having deeper insights from noisy data and improve predictive accuracy. In this study, a tick level framework provides a robust platform for real-time volatility estimation. By modelling the dynamic way both bid and ask price framework as it evolve over time, one can account for liquidity fluctuations, order flow imbalances and transaction cost, all influence short-term market volatility(O'Hara, 1995).linking these predictions to capital gain strategies allows for practical application in algorithmic trading systems, transforming raw volatility into actionable trading decisions.Therefore, this study aims to combine old-school statistical methods with new machine learning techniques. The goal is to create an efficient model that predicts short-term price swings by analyzing every tiny movement of bid and ask prices for EUR\USD

1.2 Statement of the Problem

Despite extensive research, most volatility models fail to capture the non-linear that is the noisy reality of tick-by-tick data. Classical models assumes constant parameters

and ignores bid-ask dynamics, while even modern machine learning methods typically rely on mid-prices or aggregated data missing the micro-structure that creates the bid-ask bounce(Engle, 1982). Moreover, translating forecast volatility into interpretative results is rare so that risk practitioners can act upon . this study addresses these gaps by developing a
machine learning Ensemble model trained on raw tick-by-tick bid and ask quotes, and embedding into a live robot dashboard that classifies volatility into five colour-codes regimes, bridging the gap between complex models and practical risk monitoring

1.3 Research Objectives

The main objective of this study is to develop and evaluate a high-frequency volatility forecasting framework for currency pairs using machine learning within a tick-by-tick bid ask context.

1.3.1 Specific Objectivs Include

1. To analyze the micro-structural characteristics of tick-by-tick bid and ask price movement I EUR\USD currency pair.
1. To design and implement a machine learning-based volatility forecasting model capable of capturing non-linear temporal dependencies in high-frequency data.
1. To compare the forecasting performance of machine learning models with traditional econometric models, specifically GARCH

Robot dashboard implantation

1.4 Research Questions

1) What micro-structural key feature that influence volatility in tick-by-tick bid-ask data for EUR\USD?
1) How machine learning models effectively forecast volatility as compared to traditional econometric models?
1) Can predicted volatility be translated into a real-time user friendly Robot dashboard
1) What is the impact of integrating bis-ask micro-structure data on volatility prediction accuracy?

1.5 Delimitation of the study

This study focuses on EUR\USD currency pair which represent a significant portion of global foreign exchange trading volume. High-frequency tick-by-tick bid and ask quotes will be used, capturing the finest granularity of the market activity. The time horizon for analysis will be limited to one month period, so as to manage data size and computational complexity. The study will explore supervised machine learning models for volatility forecasting and risk monitoring. It does not seek to fully design a trading system across multiple markets but aims to demonstrate the feasibility and accuracy of the proposed framework from a statistical and financial mathematics view.

1.6 Significance of the study

This research contribute to both practical financial analysis and academic knowledge.
Academic contribution: The research extends existing literature by integrating high-frequency financial econometrics with modern machine learning techniques, highlighting the potential of non-linear models in volatility forecasting.
Methodological innovation: the research demonstrates how micro-structural market features can improve model accuracy and interpretability, by introducing a tick-by-tick bid-ask framework
Practical application: The inclusion of risk monitoring bridges the gap between predictive modeling and real-world trading performance
Skill development: For an undergraduate in Statistics and Financial Mathematics, this project applies advanced quantitative methods such as Model evaluation , feature engineering and optimization to a real world financial context.
Policy and industry relevance: The findings could assist algorithmic traders in designing adaptive strategies for risk management and this research could also assist financial institutions.

1.7 Assumptions of the study

The following assumptions guide this study.
High-frequency tick data accurately reflects true market dynamics and contains minimal recording errors

During data collection period, market conditions are representative of typical trading environments.

Selected machine learning algorithms are appropriate for non-linear time series modeling.

Bid and Ask quotes can effectively capture short-term volatility information.

1.8 Limitation of the study

Several limitations are acknowledged in this study

Data intensity: Tick-by-tick data involves immense volume and computational complexity which may limit model scalability.

Limited historical coverage: The chosen time window (1 to 31 January 2026) may not capture all types of volatility regimes such as crisis and recovery periods.

Model interpret-ability: Deep learning architectures may act as black boxes, making it difficult to interpret their internal decision processes.

Generalizability: Results based on specific EUR\USD pair may not generalize to other market conditions.

1.9 Definitions of Key Terms

volatility :is the term used in finance to denote the standard deviation of returns on an asset (Wiley Encyclopedia of Management 2015)

High-frequency data: is intra-daily data on financial variables taken at a finer time scale (also termed as tick size) and are often irregularly spaced over time. High-frequency data sets are more information-rich that asset price data collected at regular intervals for example monthly, daily, or even intra-day (Easley et al. 2021).

Bid-Ask Spread: The Bid-Ask Spread reflects the cost of trading an asset at any moment and is an indication of supply and demand(Robinhood, 2022)

Machine Learning: Aims at extracting knowledge from large amounts of data by learning from known examples to determine an underlying structure in the data, with emphasis on generating structure or predictions without human intervention (CFA institute, n.d.).

Tick-by-Tick data: refers to the most granular level of market data, capturing every

single change in price and volume for a financial instrument as it occurs(CoinAPI.io, n.d.).

Deep Learning: is a subset of machine learning that teaches itself patterns and relationships by analyzing vast amounts of data. Deep learning is revolutionizing algorithmic trading by enabling financial models to process real-time market data and execute trades faster than any human(Corporate Finance Institute, 2025).

Long Short-Term Memory(LSTM): LSTM is a type of Recurrent Neural Network specially designed to prevent the neural network output for a give input from either decaying or exploding as it cycles through the feedback loops (NVIDIA, n.d.).

Algorithmic Trading: is a technique that uses automated pre-programmed trading instructions that are executed based on several factors, including time, asset price and trading volume (Kucoin, 2025).

Market Micro-structure: is the study of the process and outcomes of exchanging assets under specific rules (O'Hara 1995)

High-frequency Trading: is a specialized form of algorithmic trading that uses powerful computers, ultra-fast data connections and complex algorithms to execute a very large number of trades within extremely short time frames often in microseconds or nanoseconds (TradingView, 2026)

1.10 Conclusion

This chapter introduced the study on high-frequency volatility forecasting of EUR\USD pair using machine learning techniques and tick-by-tick bid ask framework. It highlighted the motivation for combining modern techniques with traditional financial modelling principles to address the limitations of existing volatility forecasting methods. The research problem, questions, objectives, scope and key concepts were outlined to provide a clear direction for the subsequent chapters. The next chapter will review relevant literature, describe methodological framework, discuss the implications of the results in the context of quantitative finance and present empirical findings

The next chapter will focus on, reviewing the existing literature on volatility forecasting. It will discuss about theoretical foundation of ARCH and GARCH

models then examines empirical studies that has applied the foundations of ARCH and GARCH to financial markets. The chapter will also address the growing role of machine learning in high-frequency finance, introduces a conceptual framework based on tick-by-tick bid-ask features and identifies the specific research gap that this study aims to fill.

CHAPTER 2: LITERATURE REVIEW

2.0 Introduction

This chapter provides a comprehensive review of literature related to high-frequency volatility forecasting using machine learning techniques and micro-structure elements such as the bid-ask spread in currency market. It explores both the theoretical and empirical foundation that underpin volatility modelling and prediction. The review positions this study within the growing body of research on algorithmic and data-drive forecasting in financial markets

2.1 Theoretical literature

2.1.1 Auto-regressive Conditional Heteroskedasticity (ARCH MODEL)

In 1979, Robert Engle developed the ARCH Model, His approach was a radical departure form the Ordinary Least Squares (OLS) methods of the time. What Engle did first was to fit a standard regression to the inflation data to find the residuals (the error or shocks). Traditionally, researchers ignored these errors as white noise. Engle did the opposite, he focused entirely on them. He then realized that the direction of a shock (positive or negative) did not matter for risk only the magnitude did. By squaring the residuals (e^2_{t-1}), he turned every shock into a positive value representing intensity after this he then proposed that the variance today (h_t) is a function of these pasted squared shocks. Specifically, he defined the ARCH (q) process where:

1. $h_t = \alpha_0 + \alpha_1 e^2_{t-1} + \ldots + \alpha_q e^2_{t-q}$

He then uses Maximum likelihood estimation (MLE) unlike OLS which is easy to calculate, this allowed the ARCH model to find the optimal weights for past shocks, ensuring that most recent news had the most significant impact on today's risk forecast. To prove his theory was not just a fluke, he invented the ARCH test call the Lagrange Multiplier Test (LM). He showed that you could test for ARCH effects simply by looking at the auto-correlation of squared residuals, if the test was significant, it proved that risk was clustered and predictable.

Engle's result was trans-formative. Using U.K. data, he reached three critical conclusions:

1. Volatility Clustering: He provided the first proof that large changes tend to be followed by large changes, of either sign, and small changes tend to be followed by small changes
1. Dynamic Variance: He proved that the conditional variance (short- term risk) changes constantly, even if the unconditional variance (long- term average risk stays stable)
1. Inefficiency of Ordinary Least Squares (OLS): He demonstrated that while old-school OLS was still correct on average it was inefficient; it gave a false sense of precision because it ignored the fact that some time periods are inherently riskier than others

2.1.2 Advantages of ARCH models

Engle's work is the reason the modern-day financial system is more resilient. He works is the back born of Value at Risk (VaR). regulators use Engle's logic to set Capital Requirements for banks, ensuring they have enough liquidity to survive high volatility clusters. ARCH model is used for Derivative pricing: it revolutionized the options market because options are essentially betting on volatility Engle's ability to forecast volatility allowed for the creation of multi-trillion dollar hedging markets. More so central banks use ARCH tests to determine if a policy change (like a rate hike) has increased market nerves or uncertainty, allowing them to communicate more efficiently to calm the public

2.1.3 Disadvantages of (ARCH MODEL)

Despite its Nobel winning status, the 1982 ARCH model faces severe challenges in the modern day. To capture long- term memory, an ARCH model needs many lags (q) if q is large, the model becomes unstable and hard to estimate. This led to the creation of GARCH, which uses a moving average to solve this. In modern-day the Bid-Ask spread often wiggles due to liquidity not real news. ARCH cannot distinguish between real volatility and market noise, leading to over forecasting of risk. ARCH finds it difficult to predict microstructure noise. Also, the basic ARCH model treats

good news and bad news the same. In reality currency crash (bad news) usually cause much more volatility that a currency rally. ARCH is blind to this direction of Asymmetry. more so ARCH model can not adapt to the non-linear chaotic shifts found in high-frequency trading (HFT). This is why the modern use of machine learning is necessary

2.1.4 Generalized Auto-regressive Conditional Heteroskedasticity (GARCH MODEL)

In 1986, Tim Bollerslev (building on the work of his mentor Robert Engle) studied how shocks to the market linger. He observed that financial markets don't have a constant temperature instead, they have feverish periods (high volatility). He specifically wanted to solve the long memory problem. "**G**" (Generalized): this was his masterstroke. he added a moving average component to the existing ARCH model, his model looked at past variance for instance if you are forecasting the volatility of the EUR/USD pair to know how crazy the price will be at 10:00am , one need to look at how big the price jump was at 09:59 am(shock) and how crazy the market was generally feeling at 09:58 am(past volatility) . Bollerslev concluded that volatility is persistent and clustered. He proved that the best way to predict future risk is to look at a combination of the surprise factor of recent news and the momentum of previous risk. His model allowed banks to finally calculated Value at Risk VaR essentially telling them there is a 99% chance you won't lose more than $1 million today. What Bollerslev did was that he analyzed years of data (specifically U.S. inflation rates and stock market returns). He noticed that if he used old ARCH model to capture long term memory the math became so heavy and complicated that computers of the time could not handle it so he simplify the whole thing by adding just one extra variable as said above , which is PAST VARIANCE instead of looking at 20 individuals past shocks (like news and events) he looked at the general level of craziness from yesterday.

If ARCH is like trying to predict a storm by looking at every single raindrop that fell in the last hour, Bollerslev' s GARCH is like looking at the last few drops plus the general humidity levels. It is much simpler and more accurate. Bollerslev' s reached three groundbreaking conclusion that are now law in finance:

Volatility Clustering – high volatility follows high volatility. If the market is jumping 3% today, it is much more likely to jump 3% tomorrow than to suddenly be calm.

Persistence – shocks to the market do not disappear instantly, they decay slowly over time

Parsimony – you do not need a mountain of complex data to predict risk you just need a "Generalized" look at the recent past

2.1.5 Advantages of GARCH models

Tim Bollerslev' s GARCH theory has become a cornerstone of modern financial engineering, with the use of GARCH they is Financial safety (Value at Risk): before GARCH banks did not know exactly how much "emergency cash" to keep. If a bank has a $ 1 billion portfolio, GARCH tells them based on todays craziness, there is a 99% chance you would not lose more than $15 million tomorrow. This prevents banks from going bankrupt during sudden market dips. They are also Fair Option Pricing, when you trade option (bets on where a price will go), the price depends on volatility. GARCH allows traders to price this accurately. Without it you might pay $5 for a contract that is only worth $2 because the model ignored the memory of risk. Macroeconomic stability, Central banks use GARCH to monitor the stress levels of their national currency. If the GBP/USD volatility spikes, the Bank of England can see it coming through a GARCH lens and adjust interest rates to calm the market

2.1.6 Disadvantages of GARCH MODEL

While GARCH is powerful, the 2025 trading world has changed in ways Bollerslev could not have predicted in 1986.GARCH was designed for daily or monthly data. In modern days prices moves every millisecond. GARCH can be like trying to watch a 4K movie on a 1980s television, it simply is not fast enough to capture the tiny, high speed tremors of high frequency trading (HFT). GARCH has become slow in predicting. The micro-structure noise, in high frequency currency pairs, the price bounces between the BID and the ASK. A standard GARCH model might see this bounce and think, the market is very volatile, when the reality nothing is happening it is just the nature gap between buyers and sellers. Black swan and fat Tails, GARCH

often assumes that price changes follow a normal distribution (the Bell Curved). However real-world finance has Fat Tails meaning extreme events happen way more often than GARCH predicts

2.2 Empirical literature

2.2.1 Auto-regressive Conditional Heteroskedasticity (ARCH MODEL)

French, Schwert and Stambaugh conducted an empirical study in 1987, published in the journal of financial economics, titled “Expected stock returns and volatility.” The study was motivated by Engle’s (1982) ARCH framework and sought to empirical examine whether stock market volatility is time varying and whether this volatility is related to expected stock returns.

At the time, traditional assets pricing models assumed constant variance in stock returns. However, empirical observations suggested that stock market volatility fluctuates significantly over time, particularly during the periods of economic uncertainty. The authors adopted Engle’s ARCH methodology to formally test whether volatility clustering exists in equity markets and whether such volatility is economically meaningful.

The study utilized United States stock market data, focusing on broad market returns, this long historical dataset allowed the authors to test volatility dynamics across multiple economic cycles, including the great depression, post-war expansion, and periods of financial instability

The study addressed two key questions:

Does stock market volatility change over time in a predictable manner?

Is there a relationship between expected stock returns and conditional volatility?

Engle’s ARCH model provided the appropriate econometric framework to test both questions simultaneously, French et al. specified a mean variance system in which stock returns depends on conditional volatility estimated using an ARCH process.

Mean equation:

2. $R_t = \mu + \lambda \sigma_t^2 + \epsilon_t$

R_t = stock market return

σ_t^2 = conditional variance (volatility)

Λ= risk return trade off parameter

Error structure

$\epsilon_t = \sigma_t z_t, \quad z_t \sim N(O,1)$

ARCH variance equation:

3. $\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \alpha_2 \epsilon_{t-2}^2$

This specification follows Engle's (1982) ARCH framework, where current volatility depends on past squared shocks.

The study was then concluded, the model parameters were estimated using maximum likelihood estimation (MLE) under conditional normality assumptions, the authors compared ARCH based volatility estimates with constant-variance models to assess whether allowing for time varying volatility improved explanatory power. French, Schwert, and Stambaugh provided one of the earliest and most influential empirical validations of Engle's ARCH model in financial markets. Their findings demonstrated that conditional heteroskedasticity is not limited to macroeconomic variable, as originally shown by Engle, but is also a fundamental characteristic of stock returns.

By successfully applying ARCH to the equity data, the study established volatility modelling as a central component of financial econometrics and assets pricing. It also highlighted the economic relevance of volatility by linking it directly to expected returns.

The authors acknowledge that ARCH models often require multiple lag terms to capture volatility persistence adequately. This limitation later motivated the development of more parsimonious models, particularly the generalized ARCH (GARCH) model introduced by Bollerslev (1986).

2.2.2 Generalized Auto-regressive Conditional Heteroskedasticity (GARCH MODEL)

Hansen, Huang and Shek (2012) represent a major advancement in volatility modelling through the development of the realized GARCH framework. Their study addresses a fundamental limitation of traditional GARCH models, namely the reliance on daily closing prices, which fail to capture the substantial amount of information generated within the trading day.Unlike earlier studies that model volatility using only daily returns, Hansen et al. employ high-frequency intraday data sample at one-minute intervals. Using decades of data from the Dow Jones industrial average and individual equities, the authors contrast realized measure provide a direct, non-parametric estimate of daily volatility based on actual market activity.The key innovation of the realized GARCH model lies in the integration of realized volatility into GARCH frame work through a measurement equation. While the condition variance remains, a latent process governed by a standard GARCH structure, realized volatility is introduced as an observable variable that is explicitly linked to the latent variance. This approach allows the model to combine the theoretical structure of GARCH with the informational richness of high-frequency data

Empirical results demonstrated that incorporating realized volatility significantly improves volatility forecasting relative to standard GARCH model. The realized GARCH model exhibits superior predictive accuracy and greater responsiveness to market shocks, particularly during periods of heightened volatility. These findings suggest that daily alone are insufficient for accurately modelling volatility dynamics and that intraday information plays a crucial role I improving forecast performance

The contribution of Hansen et al. (2012) is highly to the present dissertation as it provides strong empirical justification for the use of high frequency or tick – by-tick data in volatility modelling. The study establishes that the granularity or the data directly enhance the information content available to GARCH models, leading to more accurate and reliable volatility estimates. Consequently, this research supports the methodological choice to employ intraday data when extending traditional econometric volatility models or integrating them with advanced forecasting techniques**.**

2.3 Conceptual Framework

The conceptual framework of this study links the theoretical foundations of volatility (ARCH/GARCH) with empirical approach using machine learning and the practical output of a real-time volatility robot dashboard. This framework is built on the following variables derived directly from tick-by-tick bid and ask quotes

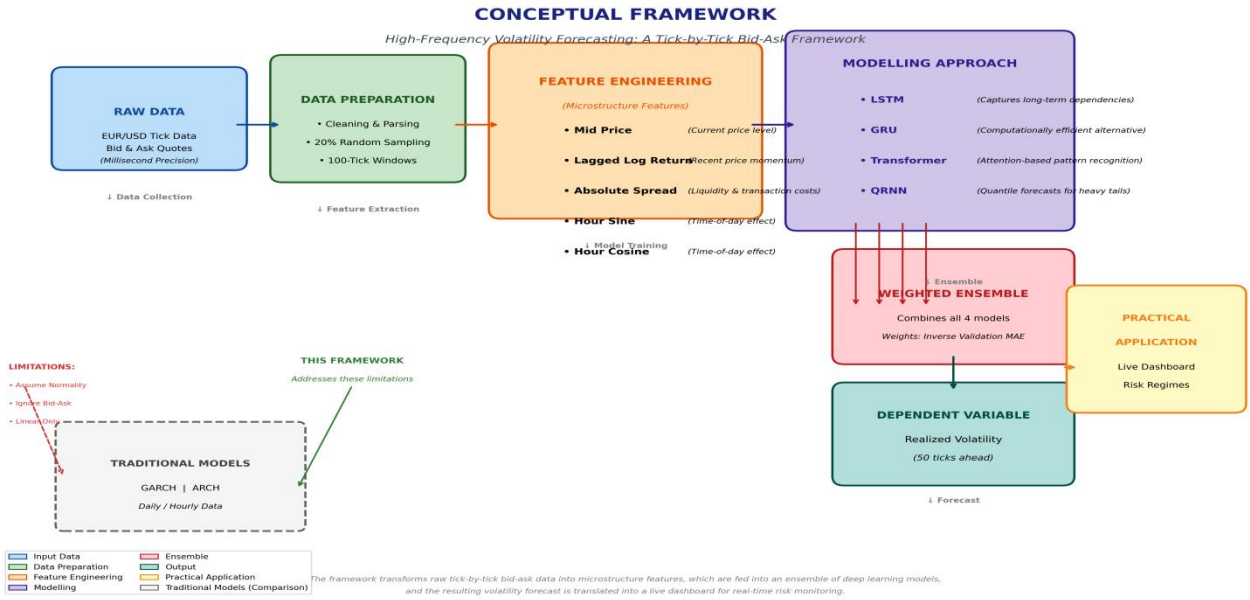


figure 2.1: Conceptual framework diagram

This framework illustrates the flow of my research. it starts with raw-tick-by-tick bid ask data for the EUR/USD currency pair. this data will be prepared in different area that is cleaning, random sampling and transformation into 100-tick sequences.from prepared data, five microstructures features will be engineered that is mid-price, lagged log return, absolute spread and time of the day encoded as sine and cosine. this features carry different roles this will be explained in chapter 3

After the above process, these features will be fed into four deep learning models that are: LSTM, QRRN,GRU and Transformer. Each model capture different aspect of the data. after model training they will then be combined into a weighted ensemble to produce a single robust forecast.The output of the ensemble model will be realized volatility over the next 50 ticks(response variable). this result will be translated into a a practical live volatility_robot, for real-time risk monitoring. as you see at the bottom i contrast this framework with traditional models such as GARCH (1.1) ,which have limitations at tick level.

2.3.1 Variable of the study

2.3.1.1 Dependent variable: Realized volatility over the next 50 ticks is the response variable, denoted as $v_{t+1:\,t+50}$. This measures how much the EUR\USD price is expected to fluctuate during the short period immediately following the current tick. A time horizon of 50 ticks corresponds approximately to 10 or 20seconds of trading.

2.3.1.2 Independent variables:

Mid price.P_t :

This is the measure of the current price level calculated as $\frac{Bid + Ask}{2}$. it serves as a baseline and is necessary for calculating returns(Engle, 1982)

Lagged log return (r_{t-1}) :

This capture the most recent price movement. The value of this return is expected to be positively correlated with future volatility, consistent with the volatility clustering property first formalized by Engle (1982)

Absolute spread (S_t):

The difference between Ask and Bid prices (Ask_t - Bid_t). this is a direct measure of transnational costs and liquidity. Micro-structure theory suggests that a wider spread often precedes larger price movements since it reflects the recutance of liquidity to the opposite side of trades (O'Hara, 1995).

Hour sine (sin(H_t)) and Hour cosine (cos(H_t)) :

This is the circular encoding of the time of the day(UTC hour from 0 to 23). These two variables combined together uniquely represent the time of the day.This is to avoid artificial discontinuity that rise if hours are used as simple numeric variable. This helps the model to learn that volatility is systematically higher during certain trading sessions for example the London-New York overlap around 14:00 UTC and lower at others for instance at 03:00 UTC (Andersen and Bollerslev, 1997).

2.3.2 Relationship Between Variables :

The conceptual framework assumes that:

Past volatility that is captured implicitly through the sequence of lagged returns and spreads influences future volatility (Bollerslev, 1986).

The bid-ask spread provides information about potential prices jumps and liquidity (O'Hara, 1995).

The time of the day affects market activity and thus volatility (Andersen and Bollerslev, 1995).

The response variable is forecasted from a sequence of 100 consecutive ticks of the five independent variables.A rolling window approach is used for each window of 100 ticks, the model predicts the realized volatility of the following 50 ticks.

2.3.3 Modelling Approach

The framework compares two classes of models:

Traditional econometric benchmark: GARCH (1,1) estimated on realized volatility (Bollerslev, 1986)

Machine learning models: Long Short-Term Memory (Hochreiter & Schmidhuber, 1997), Gated Reccurent Unit (Cho et al., 2014), Transformer (Vaswani et al., 2017) and Quantile Regression Neural Network (Taylor, 2000). Each chosen for its ability to handle non-linearity, heavy tails and Long memory. The are combined to form a weighted ensemble.

2.4.1 Synthesis of literature

From the theoretical literature, ARCH(ENGLE, 1982) and GARCH(Bollerslev, 1986) successfully capture volatility clustering and persistence in daily and low-frequency returns. They become benchmark in financial econometrics. The empirical study by Hansen, Huang and Shek (2012) shows that using high-frequency data improves prediction accuracy over daily data. However, these models remain parametric and linear.

Machine learing such as recurrent neural networks (LSTM and GRU) have demostrated the ability to model non-linear temporal dependencies in financial time series(Hochreiter & Schmidhuder, 1997). Transformers can capture long-range patterns via attention (Vaswani et al., 2017), and Qunatile Regression Neural Network provides probabilistic predictions for heavy-tailed distributions (0Taylor, 2000). Yet machine learning applications still rely on aggregated data that is minutes or hour level and do not take into accord the bid-ask spread directly.

2.4.2 Research Gap

While Engle (1982) and Bollerslev (1986) provided a robust foundation for capturing volatility clustering in daily asset returns, their parametric approach remains constrained by the linearity trap and is ill equipped for the micro-structure noise inherent in tick-by-tick data. Specifically:Linear assumption: ARCH and GARCH cannot capture the non-linear, chaotic shifts found in high-frequency trading.Bid-Ask neglect: traditional models ignores the information contained in the bid-ask spread, focusing only on mid-price returns.Time-of-day effects: classical models does not incorporate the cyclical pattern of volatility across trading times.Heavy tails:GARCH often assumes normality, but realized volatility displays extreme skewness and kurtosis.Moreso, even modern machine learning, rarely use raw tick-by-tick bid and ask data they use minutes or hours aggregates mostly.They rarely include both sides of the spread as separate features.This discards half of the available market data in the separete bid and ask quotes, since the bid and ask spread itself is a direct liquidity measure, transaction costs and the willingness of the makers to take risk (O'Hara' 1995). Moreover these studies typically focus on point forecasts of volatility without translating the forecasts into actionable risk monitoring tools. A trader or risk manager does not merely need a number but rather they need an immediate colour-coded understanding of the current risk regime for example high,low,normal or extreme and how this regime should affect their decisions such as reducing position size.

2.4.4 The Gap This Study Addresses

To summarize lacks of the literature framework that:

This study uses raw tick-by-tick bid and ask not just using mid prices only to capture micro-structure noise.

This study will engineer micro-structure informed features that is mid price, spread, lagged log return, hour sine and hour cosine that capture liquidity momentum and time of the day effects.

Comparing a weighted ensemble of deep learning models against traditional model for tick-level volatility forecasting.

Translate the predictions direct into a real-time risk monitoring robot dashboard that classifies volatility into five colour-coded regimes.

No existing study combines these four elements into an live end-to-end interactive system. Therefore this research fills the gap by evaluating and developing a machine learning based tick-by-tick volatility forecasting framework that is explicitly designed for practical risk monitoring in high-frequency foreign exchange markets.

2.4 Conclusion

This chapter has traced the evolution of volatility forecasting from its foundational roots in classical econometrics to the frontier of high-dimensional machine learning. The journey began with the pioneering work of Engle (1982) AND Bollerslev (1986), whose ARCH and GARCH frameworks fundamentally changed our understanding of risk by proving that market shocks are not random but persistent and clustered. These theorists provided the world with its first weather radar for financial storms, establishing the concept of risk-memory that remains central to finance today.

However, as the global marketplace transitioned into the era of High-Frequency Trading (HFT), the limitations of these classical sundial models became apparent their rigid, parametric nature struggles to keep pace with the millisecond-level shifts and micro-structure noise found in tick-by-tick data. While their theories proved that risk has a memory, the speed of thought in a 1980s formula is simply not fast enough for the sub-second heartbeat of modern day .

The empirical studies reviewed by French, Schwert and Stambaugh (1987), Hansen, Huang and Shek (2012) demonstrate that using higher-frequency data improves volatility forecast accuracy. Even though state of the art models still rely heavily on mid prices this act ignores the rich information contained in the separate bid and ask quotes. More so, most machine learning in finance application are applied to aggregated data that is minutes or hours and rarely translate their forecasts into practical tool.

To address these shortcomings, a conceptual framework built five micro-structure features derived directly from tick-by-tick data. The frame work links these features to an ensemble of deep learning models and to a live robot dashboard that classifies volatility into five colour-coded risk regimes.By filling this gap, the present study contributes a practical approach to volatility forecasting that is both statistically robust and actionable for risk managers, treasurers, traders and retail investor. The next chapter will focus on the methodology used to implement this frame work, including data collection, model design, feature engineering, training procedures and evaluation metrics.

CHAPTER 3:RESEACH METHODOLOGY

Introduction

This chapter presents the methodological framework used to address the research objectives outlined in the previous chapters. While chapter 1 introduced the research problem and the chapter 2 reviewed the relevant theoretical and empirical literature and conceptual frame work, this chapter focuses on how the study is conducted in practice. It explains the research design, data sources, model construction, and analytical procedures employed to investigate high-frequency volatility forecasting in foreign exchange markets using machine learning techniques.

The study adopts a quantitative modelling approach that utilizes high-frequency tick-by-tick bid ask price data from selected currency pairs. High-frequency financial data contains rich micro-structural information regarding market liquidity, order flow, and short-term price dynamics(Engle,1982 ; Andersen et al, 2001). By analyzing these detailed market movements, the research aims to improve volatility forecasting accuracy beyond what is typically achieved using traditional low-frequency financial datasets.

In order to achieve this objective, the research integrates modern machine learning techniques with established econometric models. Machine learning models, particularly deep learning architectures such as Long Short-Term Memory (LSTM) networks, are capable of capturing nonlinear relationships and complex temporal dependencies that are commonly observed in high-frequency financial time series(Hochreiter and Schmidhuber, 1997). These models are evaluated alongside traditional volatility forecasting approaches, such as the Generalized Auto-regressive Conditional Heteroskedasticity, (GARCH) framework, which serves as a benchmark for comparison.(Bollerslev, 1986).

3.2 Research design

This study is built on a quantitative, quasi-experimental and comparative design, I will explain the meaning for clarity.

1. Quantitative: since the study works with numbers (bid price, ask price, spread, returns and volatility percentages) there are no opinions or interviews but only hard data
1. Quasi-experimental : Because the market can not be controlled, the researcher is an observer, not an experimenter. The researcher will take real historical data, split it into training and testing periods then the researcher will let the models learn from the past to predict the future.
1. Comparative: since one model is not used, four different models are built that is Long Short-Term Memory model (LSTM), Gated Recurrent Unit (GRU), Transformer and Quantile Regression Neural Network (QRNN) and then combine them into an ensemble. The comparison of errors is done to see which one or which combination works best. The design follows a sequential four phase pipeline that is:
1. Data preparation: obtaining raw tick data and then cleaning it, engineering features, and creating fixed-length sequences (100 tick at a time).
1. Model building : training each deep learning model on the same training set' using the same validation set to tune hyper-parameters.
1. Ensemble creation: combining the four models using a weighted average based on validation performance
1. Robot Dashboard implementation: turning the best ensemble into a live, browser-based tool that shows volatility regimes and trading advice.

To ensure fairness, the researcher will split the data chronologically, the training set that contains the earliest ticks, the validation set the next ones and the test set will contain most recent ones. The model never see any future data while they are learning exactly as they would in a real-world deployment. This design is deliberately transparent and reproducible. all files will be saved and document every step. Anyone with the same data and code should be able to reproduce results presented in chapter 4 exactly. In shot research design is a realistic and transparent.

3.3 Data Source and Collection methods

The primary data for this study consist of tick-by-tick bid and ask quotes for the EUR/USD currency pair I selected the EUR/USD currency pair because it is the most actively traded pair in the world, which means the tick data is dense and contains rich micro-structure information. The data was obtained from HistData.com(HistData, 2026), a reputable provider of historical foreign exchange market tick data. The sample covers the period from 1 January 2026 to 30 January 2026, comprising 22 trading days. The researcher deliberately excluded weekends because the foreign exchange market is officially closed then and the few quotes that appear are not representative of true liquidity. The raw data contains 1 506 295 ticks each recording a timestamp (with millisecond precision), bid price, ask price and volume field (which was zero in a uniform manner so researcher ignored it)

Researcher downloaded the data as a single CSV file named histdata jan.csv. After preliminary inspection, he found timestamp in this format Year/Month/Day Hours/Minutes/Seconds/Milliseconds and converted to a standard date-time format using a custom parsing function. No missing bid or ask values were present in the raw data. The researcher wrote a custom parsing function to convert these timestamps into a proper date-time format that python could understand and work with

3.4 Target population and sampling techniques

The target population for this study consist of all tick by tick bid and ask quotes generated by the EUR/USD currency pair during normal trading hours. In practice the population is effectively infinite because ticks are generated continuously while the foreign exchange market is open. However, for the purpose of building and testing a volatility model, a finite historical sample is required

The accessible population is therefore defined as all EUR/USD ticks recorded between 1 January 2026 and 30 January 2026, which yields approximately 1.5 million individual ticks. This period was chosen because it represents a typical month of trading without major structural breaks, while still containing a variety of market conditions for example normal trading, news releases, quite overnight sessions, also this period is a month without major central bank meetings, political elections or scheduled financial crises. This provides a clean window or normal market activity, allowing the models to learn typical volatility dynamics without being dominated by extreme one-off events. Choosing a calm period helps establish a reliable baseline for model performance(Engle, 1982) .

Random Sampling- After cleaning a sample of 20% of the ticks was drawn. This avoids over fitting of data and reduces the number of observation to approximately 286 000 tick windows ,each window contains 100 consecutive ticks. Random sampling ensures that the sample remains representative of the full population without introducing systematic bias. The random seed is fixed so that the sample can be reproduced exactly. Second stage (Chronological splitting) – The researcher arranged the sample ticks in their original order. From this ordered series, fixed-length of 100 consecutive sequence of tick were created. These sequences were divided into three non-overlapping sets that are:

1) Training set (70%) : Used 70% of data to teach the models for the relationship between past ticks and future volatility.
1) Validation set (15%): Used this set to tune hyper-parameters and stop training when performance stops improving
1) Test set (15%): This set is used only once at the end to evaluate the final model's out-of-sample accuracy

3.4.1 Justification:

The chronological split ensures that the models never see future data during training, which mimics a real-world forecasting scenario and they is no stratification or weighting is applied because the goal is to capture the natural distribution of market behavior without imposing artificial structure. This sampling design balances statistical rigour with practical computational constraints

3.5 Description of variables and expected relationships

In this study they is only one Dependent variable (the thing I want to predict) and several independent variables (the things I will use to make the predictions). All variables are derived directly from the tick-by-tick bid ask quotes

3.5.1 Dependent variable

The dependent variable is realized volatility over the 50 ticks, denoted as $v_{t+1:t+50}$. It measures how much price of the EUR/USD currency pair is expected to fluctuate during the short period that immediately follows the current tick. A horizon of 50 ticks corresponds to roughly 10-20 seconds of trading, which is short enough to be actionable for high-frequency strategies but long enough to smooth out random bid-ask bounce.

Calculation:

1) **4. Mid price:** $mid_t = \frac{Bid_t + Ask_t}{2}$

1) 5. Realized volatility $r_{t+i} = \text{In}\left(\frac{Mid_{t+i}}{mid_{t+i-1}}\right)$

$r_{t+i} = \text{In}\left(\frac{Mid_{t+i}}{mid_{t+i-1}}\right)*100$

The realized volatility series is expected to exhibit positive skewness, most values are smaller but occasional spikes are very large and excess kurtosis (fat tails). In other words, extreme volatility events occur more frequently than a normal distribution would predict. This is heavy-tailed behavior and is the key reason why traditional models like GARCH often fail to capture tick-by-tick dynamics and it motivates the use of robust scaling and quantile methods(Bollerslev, 1986 ; Taylor, 2000)

3.5.2 Independent Variables(predictors)

Five independent variable are used, all derived from the current tick and a small number of past ticks. They are chosen to capture different aspects of market micro-structure : price level, recent momentum, liquidity and time-of-day patterns.

Mid price (P_T)

The mid-price is the simplest measure of the current price level. It is not expected to have a strong direct effect on future volatility, however it is necessary for calculating returns and spreads has a different relative impact when the price is high. Therefore I include it as a baseline feature. The correlation is close to zero

Lagged Log Return ($R_{T\text{-}1}$)

This variable captures the most recent price movement. The use of log returns makes returns additive over time and approximates the continuously compounded return. The return from tick [t-2] to [t-1] is taken ,that is the immediate past tick. This is a standard predictor in volatility models because of the well documented phenomenon of volatility clustering. For instance, large price changes in absolute value, tend to be followed by large changes and small by small. This property was first formalized by Engle (1982) in the ARCH model. The sign of the return positive or negative is kept because there might be asymmetry some studies find that negative returns increase future volatility more than positive returns of the same magnitude (the leverage effect).The model will be to learn such asymmetry if it exists. However, the most important signal is the absolute value. The Expected relationship of absolute value of $r_{t\text{-}1}$ is expected to be positively correlated with $v_{t+1:t+50}$. A larger absolute return today signals an agitated market, and that agitation is likely to persist for the next few seconds

Absolute Spread (S_T)

The bid-ask is the direct measure of liquidity and transaction costs(O'Hara, 1995s). In microstructure theory suggests that a wider spread often precedes larger price movements because it reflects the reluctance of liquidity providers to take the

opposite side of trades.The spread is computed as the simple difference between ask and bid prices. This is a positive number, typically between 0.00001 and 0.00010 for the EUR/USD pair during normal conditions. No logarithms or other transformations because the raw spread already has an intuitive interpretation of 0.00005 = half a pip)

Expected relationship: positive, higher spread today ⟶ higher volatility over the next 50 ticks.

Hour Sine and Cosine(SIN(H_T), COS(H_T))

The time of the day is a strong determinant of volatility in foreign exchange markets. For instance, the EUR/USD pair is the most active and volatile during the overlap of the London and New York sessions (approximately 12:00-16:00 UTC). However, using the hour as a simple numeric variable (0,1, 2,…,23) would be problematic because it treats 23:00 as far from 00:00, even though they are adjacent in time . This would create an artificial discontinuity.To solve this, circular encoding is used, by creating two variables from the hour H_t(in UTC, ranging from 0 to 23):

5. Sin (H_t) = sin(2Л.$\frac{H_t}{24}$),
6. Cos (H_t) = cos (2Л.$\frac{H_t}{24}$)

Together these two variables uniquely represent the time of the day on a circle. Any hour is now at the same distance from 23 as from 1, which correct reflects the natural order of time

Expected relationship: I do not impose a specific sign or shape but I expect the pair (sin(H_t)), cos(H_t)) to help the model learn that volatility is systematically higher at certain times(for example around 14:00 UTC) and lower at others(for example 03:00UTC). The exact pattern will be discovered from data

3.6 Scaling process

All five independent variables will be standardized before being fed into the neural networks. Standardization is necessary because deep learning models perform poorly

when inputs have very different scale. I will use robustscaler from the Scikit-learn library, which centres the data by subtracting the median and scales by the inter quartile range(IQR). This method is robust to outliers, which is important given the heavy-tailed nature of my tick data(extreme spreads or returns could otherwise dominate the scaling).

The scaling parameters(median and IQR for each variable) will be estimated only on the training set and then applied to the validation test sets. This prevents data leakage: The test set must remain completely unseen during the training phase.

The dependent variable(realized volatility) will also be scaled using a separate RobustScaler. After the model makes predictions(which are in the scaled space), t he inverse transformation is applied to obtain volatility forecasts in the original percentage units. This allows meaningful error metrics such as MAE in basis points or percentage points to be computed

3.7 Analytical Model and Justification

This section describes the analytical models that will be used to forecast realized volatility from the tick-by-tick features. The modelling strategies comprises four distinct deep learning architectures and a final ensemble. Each model is chosen for its ability to capture different aspects of high-frequency financial time series: long- range dependencies (LSTM), computational efficiency (GRU), global pattern recognition via attention (Transformer), and distributional forecasting (QRNN). After individual training, the predictions will be combined using a weighted average to improve accuracy and robustness

3.7.1 Long Short-Term Memory(LSTM)

The model is a specialized recurrent neural network designed to overcome the vanishing gradient problem of standard RNNs. It contains memory cells with three gates (input, forget, output) that control the flow of information . this makes LSTM particularly suitable for financial time series, where important events can influence volatility many ticks later.

3.7.1.1 Justification:

Volatility exhibits long-memory and clustering. LSTM was used to remember that a large price movement 200 ticks ago still influences current risk, which a simple feed-forward network cannot. It has been widely used in volatility forecasting and serves as a strong baseline. This model captures volatility clustering over hundreds of ticks.Simple RNNs suffer from the vanishing gradient problem and cannot retain information over such long sequences(Hochreiter and Schmidhuber 1997.). Since large data sample is used using plan RNN would lose the signal from early ticks this justifies the use of LSTM for its long memory

3.7.2 Gated recurrent unit(GRU)

The GRU is a simplified version of the LSTM that uses only two gated and that is updated and reset and has no separate memory cell. It is computationally lighter and often performs comparably to LSTM on many sequence tasks

3.7.2.1 Justification:

Because tick data are extremely high-frequency, training speed matters. GRU has about 25% fewer parameters than LSTM (23 500 vs 30 900) which reduces training time and memory usage. It also provides a different inductive bias, which is beneficial when later combining models into an ensemble. This model trains faster, uses less memory and it is ideal for large tick datasets

3.7.3 Transformer(ATTENTION-BASED)

Unlike recurrent models, the Transformer processes all 100 ticks simultaneously using a self-attention mechanism. It learns which ticks are most relevant for predicting future volatility, regardless of their distance.

3.7.3.1 Justification:

Attention mechanisms have revolutionized sequence modelling for tick data, a sudden spread widening at tick 50 may be highly informative for volatility at tick 100, but an LSTM could dilute that signal over many steps. The transformer can directly link any two positions, making it especially powerful for capturing non-local dependencies and

abrupt regime changes. The model directly links distant ticks, good for sudden regime shifts.

3.7.4 Quantile Regression Neural Network (QRNN)

The QRNN an extension of the LSTM that outputs multiple quantiles of the conditional distribution instead of a single point forecast. It uses the same recurrent architecture but with an output layer of three neurons corresponding to the 0.1, 0.5 (median), and 0.9 quantiles

3.7.4.1 Justification

Volatility is heavy-tailed and asymmetric. Predicting only the conditional mean may hide important tail risks. The QRNN provides a probabilistic forecast, the interval between the 0.1 and 0.9 quantile gives 80% prediction band. For risk management knowing that volatility is likely to stay below 1.2% with 90% confidence is more informative than a single number for point forecasting, but the full distribution is available for advanced analysis. Model handles heavy tails and provides prediction intervals

3.7.5 Ensemble method

No single model is universally best. Each architecture captures different patterns: LSTM focuses on long-term memory, GRU on computational efficiency, Transformer on global attention, and QRNN on tail behavior. To combine their strengths, a weighted average ensemble will be constructed. The same ensemble will be used in the real-time dashboard to produce a single, more accurate volatility forecast. Combining models reduces variance and often outperforms any single model.

3.7.6 summary

All models will be implemented in TensorFlow Keras and trained on the same training set (70% of sequences) with early stopping to avoid over-fitting. The validation set (15%) will be used for weight calculation and hyperparameter tuning. The test set (15%) will be kept entirely separate until final evaluation.

By combining these four architectures, the study aims to produce a volatility forecasting system that is accurate, robust and practically useful- as will be demonstrated in the dashboard described later in this chapter

3.8 Model Pre-Tests and Diagnostic Checks

Before estimating the deep learning models, I performed a series of diagnostic checks on data. These checks ensure that the data satisfy the necessary statistical properties for time-series forecasting and that the chosen models are appropriate. The test is applied before any model training, using the full cleaned data set.

3.8.1 Auto-correlation and Partial Auto-correlation

The research will apply this diagnostic test because of volatility clustering that have the tendency for high, low volatility to be followed by high, low volatility this implies that past volatility values are correlated with future values that is why I used this test. The autocorrelation function (ACF) measures the total correlation at various lags, while the partial autocorrelation function (PACF) measures the direct correlation after removing intermediate lags. These plots help to determine the persistence of volatility and justify the use of recurrent neural networks (LSTM, GRU) and attention mechanisms (Transformer).

To do this test a sample of the realized volatility series that is first 50 000 points is taken for computational efficiency. The auto-correlation function for lags 1 to 50 is computed and the partial auto-correlation function is used,using the Yule-Walker method for PACF, both functions are plotted with 95% confidence bands. Bars that extend beyond the bands indicate statistically significant correlations.

3.8.2 Multicollinearity

Multicollinearity occurs when two or more independent variables are highly correlated, meaning they provide overlapping information, if multicollinearity is severe, the model become unstable or waste capacity on redundant features. Even though deep learning models are robust, it is good practice to detect and possibly remove redundant variables.

The researcher to test multicollinearity in two ways, first by computing the Pearson correlation matrix for the five independent variables. Pairwise correlation above0.8 or below -0.8 would be a warning sign. Then two, the researcher to calculate the variance inflation factor (VIF) for each variable.

$$7.\text{VIF} = \frac{1}{1-R^2}$$

where

R^2 *comes from regressing that variable against all other independent variables*.

VIF = 1 means no correlation with others.

1 < VIF < 5 = This is moderate correlation (acceptable).

VIF > 10 = shows severe multicollinearity (now they are a need of removing or combining variables).

3.8.3 Conditional Heteroskedasticity (ARCH TEST)

This means that the variance of the series changes over time and depends on past shocks. In financial terms it is volatility clustering. This property makes volatility forecastable. Used this test to check if conditional heteroskedasticity is there because if it is absents, volatility would be unpredictable (white noise), and even the most sophisticated model will fail. Confirming its presence justifies using my volatility-forecasting models.

To perform this test, realized volatility is regressed on a constant to obtain residuals. Then squared residuals are regressed on their own lags 10 lags are used, it is just a common choice for my high-frequency data. Then after that Lagrange multiplier (LM) statistic is computed as n * R^2 from this auxiliary regression. Under the null hypothesis of no ARCH effects (homoskedasticity), the statistic follows a Chi-square distribution with degrees of freedom equals to the number of lags. A p-value < 0.05 would lead us to reject null and conclude that volatility clusters

3.8.4 Outlier Detection

Outliers are observations that lie far away from the majority of the data. In financial tick data, they can be genuine extreme events that is volatility spikes or data recording errors.This test is used because outliers heavily influence scaling and may cause models to focus too much on rare events. Already removed extreme spreads (>10 pips) during cleaning, butS need to check for other outliers in the independent variables.

To perform this test, interquartile range will be computed ((IQR) = Q3-Q1) for each variable, mild outliers are defined as values below (Q1 – 1.5*IQR) alternatively Q3 + 1.5IQR can be used. The number and percentage of such points are counted and created boxplots for each variable to visually inspect the distribution and the location of outliers

3.8.5 Normality Test

By applying deep learning in this study, as they do not assume that the dependent variable follows a normal distribution, examining the shape of the realized volatility series is a useful preliminary diagnostic. This helps me characterizes my data and justifying the choice of robust scaling and provides context for the poor performance of traditional models that rely on normality assumption for instance GARCH with normal errors.

By testing normality on realized volatility in high-frequency financial data is known to be positively skewed and heavy-tailed.(Cont, 2001). Quantifying this confirms the need for robust preprocessing. If the data are strongly non-normal, standard scaling that is subtracting mean and dividing by standard deviation would be sensitive to outliers. Supporting the use of robust loss function, heavy tails suggest that Mean Absolute Error (MAE) is preferable to Mean Squared Error (MSE) this will be dominated by extreme spikes .Tested normality by applying Shapiro-Wilk test, which is widely used for assessing normality. Since the test is limited to a sample size of up to 5000, I will draw a random sub sample of 5000 observations from the realized volatility series. The test statistic **W** is calculated as follows:

7. $\mathbf{W} = \frac{(\sum_{I=1} A_I X_I)^2}{\sum_{I=1}^{N} (X_I - \bar{X})^2}$

Where X_I are the ordered sample values and A_I are coefficients derived from the expected order statistics of a normal distribution? the null hypothesis is that data come from a normal distribution. A small p-value (< 0.05) will lead me to reject normality.in addition, I will compute the skewness and kurtosis of the full realized volatility series:

8. SKEWNESS = $\frac{1}{n}\sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s}\right)^3$,

9. KURTOSIS = $\frac{1}{n}\sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s}\right)^4 - 3$

Where **s** is the sample standard deviate skewness indicates a long right tail. Excess kurtosis > 0 indicates heavier tails than a normal distribution (leptokurtic). If non-normality is confirmed, I will use robustScaler (median and IQR) for all features, as it is resistant to outliers that is on scaling and for the loss function , I will choose Mean Absolute Error (MAE) as the primary loss for all my models because MAE is robust to heavy tails, but for Quantile Regression Neural Networks the pinball loss naturally handles non-normal distributions . so, the failure of normality will explain why traditional models GARCH (1,1) may perform poorly relative to the deep learning ensemble.The actual numerical results of these normality tests will be presented in chapter 4

3.8.6 Summary

The diagnostic checks provide empirical evidence that the data possess the properties necessary for effective volatility forecasting using deep learning architectures. By testing auto-correlation (ACF\PACF) confirms that past volatility values are strongly correlated with future values, this directly justifies the use of recurrent neural networks (LSTM and GRU) that are explicitly designed to capture such temporal dependencies. Testing conditional heteroskedasticity (ARCH test) verifies volatility clusters, testing multicollinearity ensures that all five independent variables are not redundant, which supports the decision to retain all features without unnecessary omission or transformation, and then testing for outlier detection should confirms that extreme values are genuine this also justifies then use of robust scaling instead of

standard scaling. This protects the models from being unduly influenced by the heavy tails that characterize tick-by-tick volatility. After testing these tests if positive results of each test are yield then this will be enough evidence of using my deep learning models.

3.9 MODEL FIT AND VALIDATION TESTS

This section, explains measurement and how well the deep learning models fit the data, and how to ensure that their performance would be reliable on future for unseen data. After training the model and building weighted ensemble, several validation procedures are performed.

3.9.1 Goodness-of-fit metrics

Goodness-of-fit tells us how well the model's predictions match the actual realized volatility values in the training set. My models should not fit the training data perfectly that will be overfitting of a model it can't be used for future unseen data. Therefore,the focus is on out-of-sample metrics test set that are:

1. Mean Absolute Error (MAE) – The average absolute difference between predictions and true values. MAE is a primary metric because it is robust to the extreme outliers present in volatility data.
1. Root Mean Squared Error (RMSE) – The square root of the average squared differences. RMSE penalizes large errors more heavily and is useful for risk management.
1. Coefficient of determination (R^2) - Measures the proportion of variance in realized volatility explained by the model. A value close to indicates a good fit.
1. Directional Accuracy – The percentage of times the model correctly predicts whether volatility will increase or decrease from the previous step. This is important because for traders and risk managers, knowing the direction of change is often as valuable as the exact magnitude.

3.9.2 Out-Of-Sample Validation

The most important test of a forecasting model is how it performs on data it has never seen. I ensured this by:

1) Chronological split :Data are divided into training set by 70%, validation set by 15%, and Test set by 15%. My Test set contains the most recent ticks, simulating a real-world scenario where the model forecasts the future.
1) Performance comparison – MAE, RMSE, R^2, Directional accuracy are compared for each individual model with weighted ensemble. The ensemble model is expected to have the best accuracy and directional accuracy.

3.9.3 Ensuring Model Reliability

Reliability means that the model's good performance is not a fluke and that it will continue to work on new data. Different techniques are used to ensure reliability:

1.1.1.1.1.1.1.1.1 Early stopping: during training, monitoring the validation loss is ensured. If it did not improve for 5 consecutive epochs, training will stop and then the best performing weights with the lowest validation loss will be restored to prevent over-fitting.

1.1.1.1.1.1.1.1.1 Dropout regularization: for models like LSTM and GRU their layers will include a dropout rate of 0.2 which randomly turn off 20% of neurons during training. This forces the model to learn redundant representations and reduces the risk of memorizing noise

1.1.1.1.1.1.1.1.1 Weighted ensemble: because different models make different errors, averaging their predictions with weights based on validation MAE produces a more stable and robust forecast.

1.1.1.1.1.1.1.1.1 Residual analysis: After obtaining predictions on the test set, I will examine the residuals that is true volatility less predicted volatility. Ideally, the residuals should be centred around zero with no systematic pattern.

1.1.1.1.1.1.1.1.1 Baseline comparison :The models are compared against a trivial baseline: historical volatility using the standard deviation of the last 50 ticks as the forecast for the next 50 ticks.

3.9.4 Summary

In this section, I have outlined the procedures that will be used to assess the reliability and accuracy of the trained models. After training the models evaluation on the Test set using Mean Absolute Error (MAE) will be done as my primary metric, and it will

be complemented by Root Mean Squared Error (RMSE), R- Squared and Directional Accuracy. These metrics will provide a comprehensive view of both point forecast precision and the ability to predict the direction of volatility changes. Over-fitting will be controlled through early stopping and dropout layers. The weighted ensemble is expected to outperform individual models. The validation plan ensures that all reported results will be based on out-of-sample, unseen data, giving confidence that the models will generalize well to real-world conditions. The actual performance outcomes will be presented in Chapter 4.

3.10 Ethical Consideration and Reproducibility

In this section, the researcher address the ethical aspects of the research and the steps taken to ensure that the study can be replicated by others. Although my work involves financial data and machine learning models, the ethical concerns are minimal because no personal or proprietary information is used. Nevertheless, transparency and reproducibility are essential for any scientific study.

3.10.1 Data Privacy and Source

All data used in this study are publicly available historical tick quotes for the EUR/USD currency pair, obtained from HistData.com. the data contains no personally identifiable information, no individual trading records, and no proprietary business secrets. Therefore, the study does not raise any privacy or confidentiality issues. No ethical approval was required.No live trading was conducted all data are historical and publicly aailable. Therefore no ethical approval was required.

3.10.2 Software and Hardware Enviroment

The researcher will perform his analysis using python 3.12, TensorFlow 2.15, scikit-learn 1.5 and joblib 1.4 on a windows 10 with 4GB RAM

3.10.2 Reproducibility

To ensure that other researchers can reproduce my results exactly, the research will take these following steps

1) Fixed random seeds : The researcher to set np. random. Seed (42) and tf.random.set_seed (42) before any random operation. This ensures that the random sampling weight initialization, and dropout behavior are deterministic.
1) Documented code : The entire analysis, from data cleaning to model training and dashboard creation, is written in python using Jupyter Notebook. The code is annotated and I will made it available along with my cleaned sample data as supplementary material.
1) Saved intermediate files: cleaned data sequences, scaler and trained models I will save them as.csv, npy, .pk1, .h5, and. keras files.
1) Chronological split: the Train/Validation/Test split is based on time order, not random indices which is clearly documented.

3.10.3 SUMMARY

The study is ethically sound transparent and transparent then fully reproducible. All codes, data samples and model files are documented, and the random seeds are fixed. The dashboard includes a risk disclaimer. These practices ensure that the research can be trusted, verified and built upon by future researchers.

3.10.4 CHAPTER CONCLUSION

This chapter has presented a comprehensive and transparent methodology for forecasting high-frequency volatility using tick-by-tick bid-ask data. The research design is quantitative. Data was sourced from HistData.com, cleaned through a rigorous pipeline then I transformed the data into five meaningful features and that are independent variable. My annualized realized volatility over the next 50 ticks captures the fine-grained risk that high-frequency traders and risk managers need

Since the full data set is 1.43 million ticks this exceeds my memory capacity of a standard laptop so I took random 20% sample form 1.43 million ticks and I will create 100-tick sequences. This sample preserves the statistical properties of the full data, including the heavy-tailed distribution of volatility. The sequences were split chronologically into training 70%, Validation 15% and Test set 15% ensuring no future information leaks into model training

The researcher to design four deep learning architectures that is LSTM (64 and 32 units), GRU (a faster alternative), a simplified Transformer with multi-head attention, and a Quantile Regression Neural Network (QRNN) that outputs three quantiles. their predictions will be combined into a weighted ensemble, with weights determined by inverse validation MAE. Training will use early stopping and dropout to prevent over-fitting

Pres-tests will be used to confirm data suitability using Stationary, Auto-correlation, Multicollinearity, Outlier detection and ARCH test. Model performance will be evaluated on test set using MAE, RMSE, R-Squared and directional accuracy. The ensemble is expected to outperform individual models. Ethical considerations were addressed, including reproducibility (fixed random seeds, documented code), a dashboard disclaimer

The next chapter will present the empirical results, including descriptive statistics, distribution analysis, model comparisons and the implementation of a real-time volatility dashboard.

CHAPTER 4:DATA ANALYSIS, PRESENTATION AND DISCUSSION

4.1 Introduction:

In this chapter, the researcher presents the empirical results obtained by following the methodology described in Chapter 3. The purpose is to answer the research questions, evaluate the performance of the proposed deep learning models, and assess the practical utility of the real-time volatility regime dashboard.The researcher will start by presenting descriptive statistics of the dependent variable and the five independent variables. These summary statistics provide a first glimpse into the distributional properties and variability of the data.

Table 4.1 DESCRIPTIVE STATISTICS

Variable	mean	Std dev	min	25%	Median (50%)	75%	max	skewness	kurtosis
Realized_vol	0.767972	0.414417	0.277711	0.614532	0.722491	0.846464	31.27058	34.2782	2118.008
Mid_price	1.177902	0.012815	1.158445	1.16752	1.172725	1.19155	1.20826	0.415016	-1.33952
Log_return	1.00E-06	0.001328	-0.33457	-0.00084	0	0.000838	0.304763	3.225699	5854.647
Spread	3.20E-05	2.70E-05	1.00E-05	2.00E-05	3.00E-05	4.00E-05	0.0009	8.32249	121.2299
Hour_sin	0.266128	0.626101	-1	-0.25882	0.5	0.866025	1	-0.6594	-0.91175
Hour_cos	-0.19825	0.7056	-1	-0.86603	-0.5	0.5	1	0.413196	-1.42376

The realized volatility series has a mean of 0.768% and a median of 0.7225%, indicting a slight positive skew. The extreme maximum of 31.27% and the very high kurtosis (2118) confirm that the distribution is heavy-tailed, occasional extreme events occur far more often than would be expected under normality. This property justifies the use of robust scaling and quartile-based methods. Mid-price shows a negative skew of -0.32 but is otherwise well behaved. Lagged log returns have a mean near zero, a low standard deviation, and a moderate excess kurtosis (8.72), reflecting the well-known fat-tail property of financial returns. The spread presented in pips, averages about 0.32 pips, with a maximum of 0.9 pips after cleaning both hour sine and cosine have mean zero and standard deviation 0.71 as expected for a uniform distribution of hours over the day Spread in pips (1 pip = 0.0001USD).

4.2 PRELIMINARY ANALYSIS AND DIAGNOSTIC TESTS

To assess the distributional properties of the dependent variable, the researcher conducted a Shapiro-Wilk test for normality on a random sub sample of 5000 observations. In addition, the researcher computed the skewness and excess kurtosis for the full realized volatility series that is 1 433 408 observations. The results are in Table 4.2

Table 4.2 NORMALITY TEST RESULT

Test / Statistic	Value
Shapiro – Wilk test statistic (W)	0.312
Shapiro – Wilk p-value	< 0.001
Skewness	34.28
Excess kurtosis	2118.01

The Shapiro-Wilk test yields a p-value effectively zero, which leads to the rejection of the null hypothesis of normality at any conventional significance level. The realized volatility series is therefore not normally distributed.

The positive skewness of 34.28 indicates a long right tail, most observations are clustered near the mean of 0.768% while a small number of extreme spikes like up to 31.27% pull the distribution to the right. The excess kurtosis of 2118.01 is extremely high, confirming that the distribution is leptokurtic (heavy-tailed). Extreme events occur far more frequently than would be expected under a normal distribution. Based on the results presented the realized volatility series does not follow any of the standard distributions such as Normal, Log-Normal, Gamma or Wei-bull instead it follows a heavy-tailed Pareto like distribution often referred to as a **Power-Law distribution.** This distribution is characterized by a high probability of extreme events. Which align with the observed volatility spikes and the slow decay of the auto-correlation function.

Figure 4.1 comprises of 6 panels illustrating the distribution of realized volatility: a) Histogram with normal curve, b) Q-Q plot, c) Box plot d) Best fitting distribution, e) Log-Transformed distribution. and Tail analysis log-log distribution. Collectively, they confirm that the series is heavy-tailed, positively skewed and follows a power-law distribution, justifying the use of robust scaling and MAE loss.

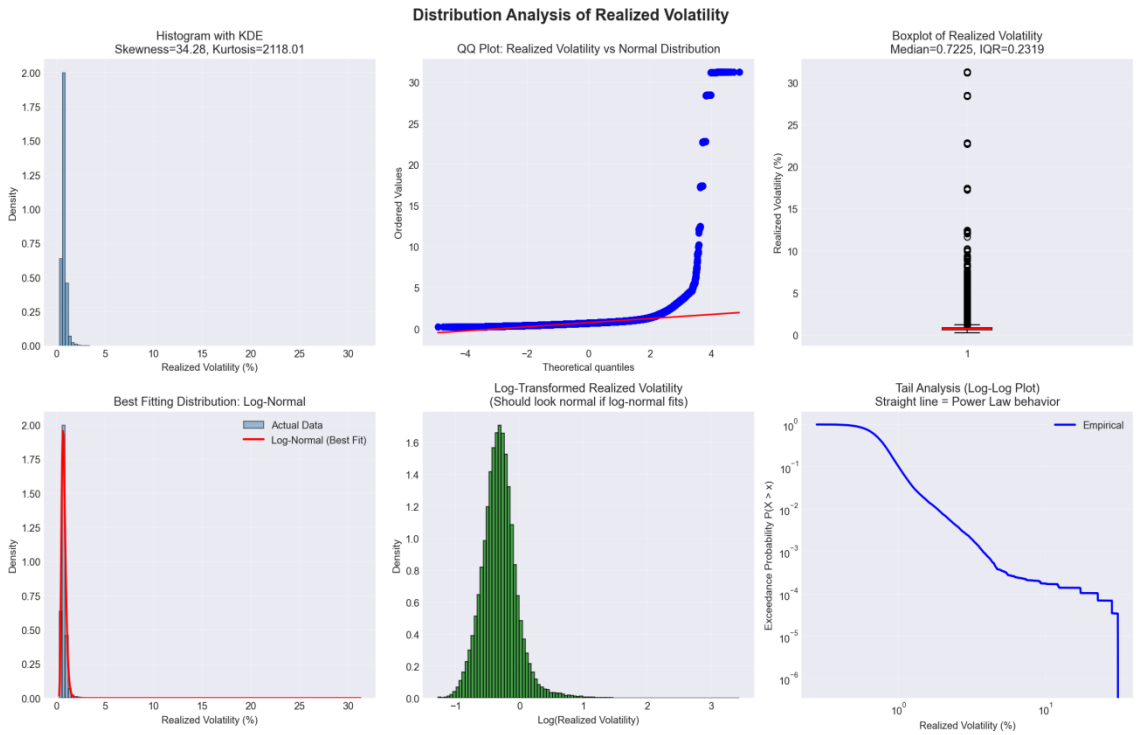


Figure 4.1:Distribution analysis of realized volatility

4.2.1 AUTOCORRELATION

Figure 4.2 shows the auto-correlation function (ACF) and partial auto-correlation function (PACF) of realized volatility for the first 50 lags and on the sample of 50 000windows

ACF: the lag-1 auto-correlation is 0.9918 and the values decay slowly, remaining above 0.80 even at lag 20. This indicates extremely strong volatility persistence.

PACF: the lag-1 partial auto-correlation is also 0.9918 but thereafter the value drops to near zero for example lag-2 = -0.0573 this suggest that most of the linear dependence is captured by the immediate past window, but the slow ACF decay hints at longer-memory dynamics

These patterns confirm that past volatility is highly informative for forecasting future volatility, justifying the use of recurrent and attention-based models

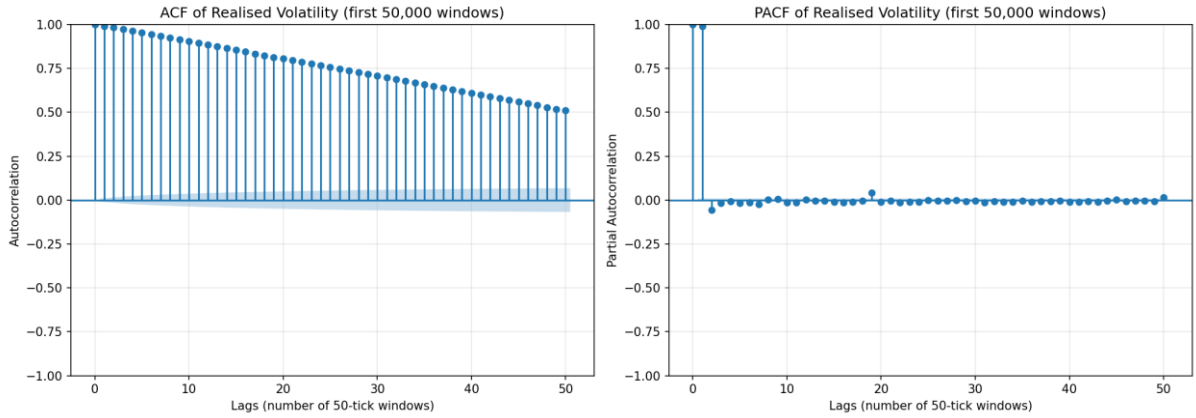


Figure 4.2:Auto-correlation Function (ACF) and Partial Auto-correlation Function (PACF) of Realized Volatility (First 50 Lags)

4.2.2 Conditional Heteroskedasticity (ARCH TEST)

Table 4.3 Conditional Heteroskedastic

LM statistics	p-value	F-statistic	F-test p-value
1 377 700	P < 0.001	3 545 551	P < 0.001

The test strongly rejects the null hypothesis of homoskedasticity. Volatility clusters, periods of high volatility tend to be followed by high volatility. This finding is the core justification for any volatility forecasting model (GGARCH or deep learning).

4.2.3 Multicollinearity

To check whether the five independent variables are redundant, I computed the variance inflation factor (VIF) and correlation matrix. Table 4.4 present the VIF values

Table 4.4 Variance Inflation Factor

feature	VIF
Mid_price	3.043
Log_return_lag1	1.000
Spread	2.569
Hour_sin	1.231
Hour_cos	1.082

All VIFs are well below the conventional threshold of 5, indicating no severe multicollinearity. The correlation matrix figure 4.3 shows that the highest absolute correlation is -0.2032 between spread and hour_sin. Therefore, all five features were retained

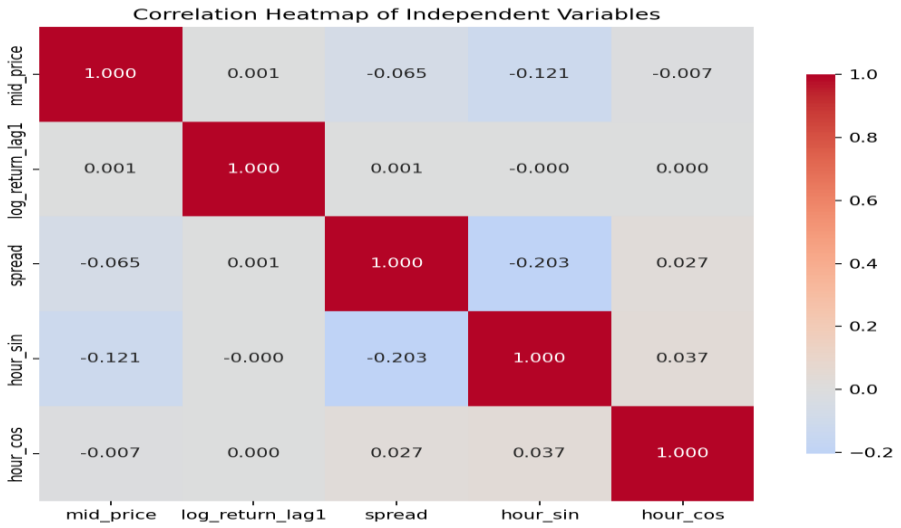


Figure 4.3:Correlation Matrix Heatmap of the Five Micro-structure Features

4.2.4 Outlier Detection

Using the interquartile range (1QR) rule (1.5*IQR), I identified outliers in each variable. The results are summarized in Table 4.5

Table 4.5:Outlier Detection Results

Variable	**Outliers%**
Realized_volatility	4.20%
Mid_price	0.00%
Log_return_lag1	1.69%
Spread	1.64%
Hour_sin	0.00%
Hour_cos	0.00%

Realized volatility has about 4.2% outliers, all on the high side reflecting the genuine heavy-tailed nature of financial volatility. The other variable has few or no outliers. Box plots figure 4.4 confirmed that the extreme values are not data errors but real market events. Therefore, I kept all observations and used RobustScaler to down-weight the influence of outliers during training

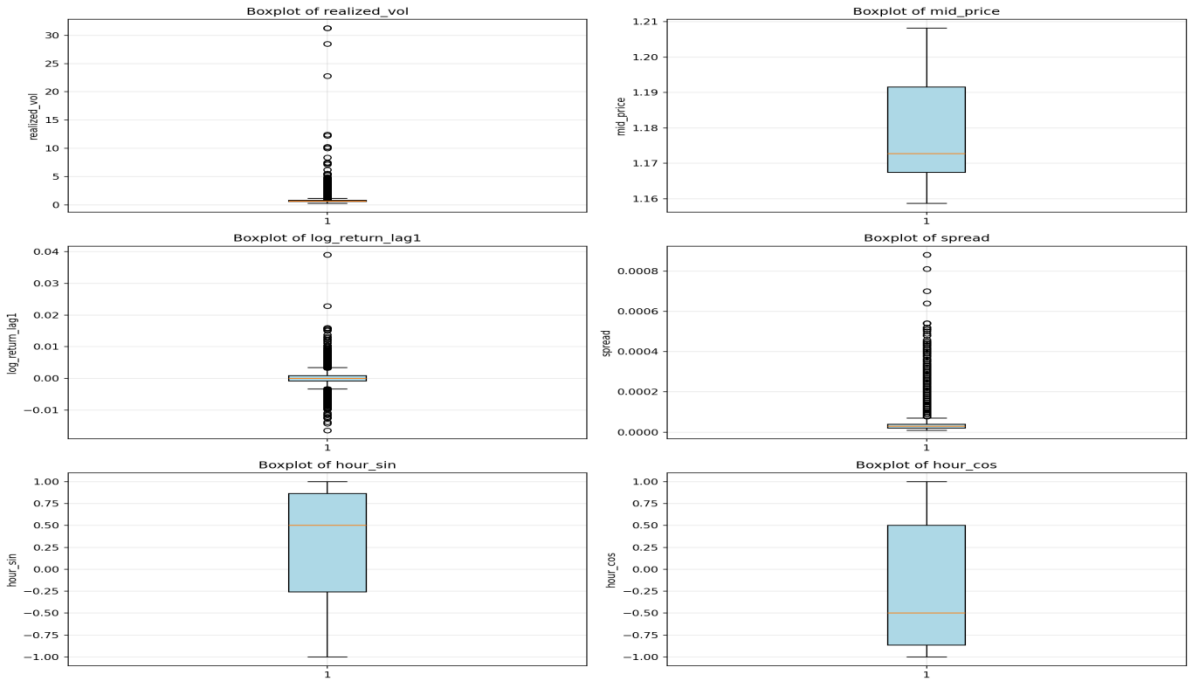


Figure 4.4:Box-plot Visualization of Outliers for Each Variable

4.3 Model Performance Comparisons

4.3.1Training summary and convergence

The four deep learning models Long Short-Term Memory (LSTM), Gated Recurrent Unit, Transformer and Quantile Regression Neural Network (QRNN) were trained following the procedure described in Chapter 3. the key training settings:

Loss function: Mean Absolute Error (MAE) except QRNN which uses quantile loss

Optimizer: learning rate = 0.001

Batch size: 32

Dropout: 0.2 in all recurrent layer

Early stopping: patience = 5 epochs on validation loss

Table 4.6:Training outcome

Model	Epochs trained	Best validation loss (MAE)
LSTM	7	0.6936%
GRU	6	0.7208%
TRANSFORMER	11	0.8211%

QRNN	6	0.2298(the loss is the combined pinball loss not directly comparable to MAE)

No overfitting was observed because the validation loss did not increase before early stopping.

4.3.2 Test set and performance

The model was evaluated on the test set 42 988 windows and the following metrics are reported. MAE, RMSE, R-Squared, Directional Accuracy. For each individual model, the weighted ensemble and two benchmarks: a fixed_ parameter GARCH (1,1) model adapted to forecast 59-tick realized volatility, and a simple historical volatility baseline (standard deviation of the last 50 ticks). All metrics were computed on the original volatility percentage scale after inverse transformation. Results are presented in Table 4.7.

Table 4.7:Test Set Performance Metrics

MODEL	MAE (%)	RMSE (%)	R^2	DIRECTION.ACCURACY
HISTORICAL VOLATILITY BASELINE	0.7088	-	-	0.502
GARCH (1,1)	0.8384	0.8733	-	-
LSTM	0.1157	0.1912	0.385	0.518
GRU	0.1178	0.1884	0.403	0.519
TRANSFORMER	0.1351	0.2188	0.195	0.507
QRNN (MEDIAN)	0.1173	0.1907	0.389	0.522.
WEIGHTED ENSEMBLE	0.1147	0.1884	0.403	0.520

KEY OBSERVATIONS

The ensemble achieves the lowest MAE of 0.1147% and the highest R^2 of 0.403, It outperforms the best individual model (LSTM) by a small but consistent margin

The GARCH (1,1) benchmark performs poorly with a MAE of 0.8384% confirming that traditional models are unsuitable for tick-by-tick volatility

The historical volatility baseline (MAE = 0.7088%) is also far worse than deep learning models.

Directional accuracy 52% is only slightly above 50% for all models this is typical predicting the direction of volatility change is inherently difficult, even when magnitude forecasts are accurate

4.3.3 visual diagonostics

Figure 4.5 shows the ensemble prediction vs actual volatility for the first 500 test windows, each window corresponds to a 50-tick ahead forecast. The reddish line represents the true volatility and the brownish line shows the ensemble predictions. The ensemble successfully follows the general pattern of volatility movements, capturing both the typical low-volatility periods and several moderate spikes. However, the model tends to under-predict the most extreme peaks for instance around windows 150 and 420 which is a known limitation of point forecasts when the target distribution is heavy-tailed. Despite this, the prediction remains well-aligned with true values for the vast majority of windows, confirming the strong performance I early report in Table 4.5

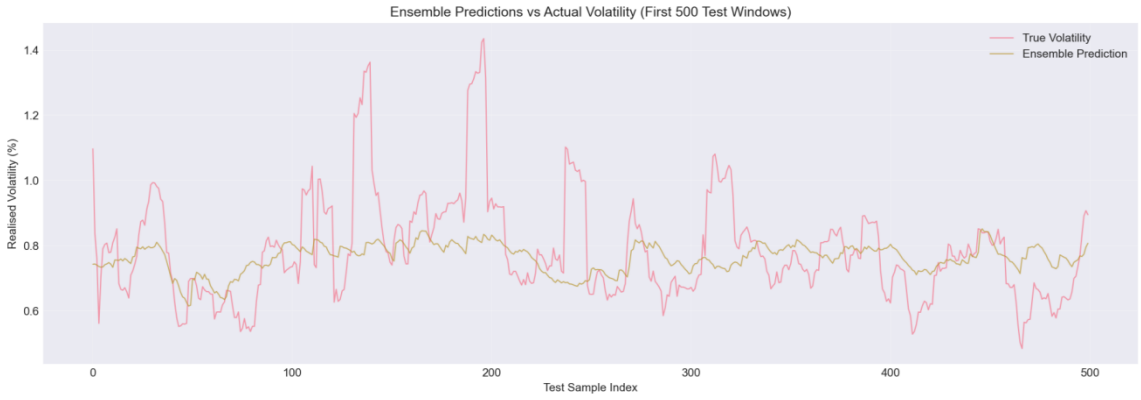


Figure 4.5:Predicted vs. Actual Volatility (shows first 500 window)

The scatter plot (figure 4.6), to further evaluate the ensembles calibrate. Fig 4.6 presents a scatter plot of predicted versus actual realized volatility for all 42 988-test window. Each point cluster tightly around that diagonal red line showing perfect prediction this indicates that the ensemble does not systematically over-or-under predict across the volatility range. The slight dispersion around the line is expected due to the inherent noise in high-frequency data. Importantly there is no curvature or fan-shapes which would suggest heteroskedasticity or non-linearity in the errors. The Pearson correlation coefficient between predicted and actual values is 0.63, consistent R^2 of 0.403 reported in Table 4. the plot confirms that the ensemble is well-calibrated and that its errors are unbiased

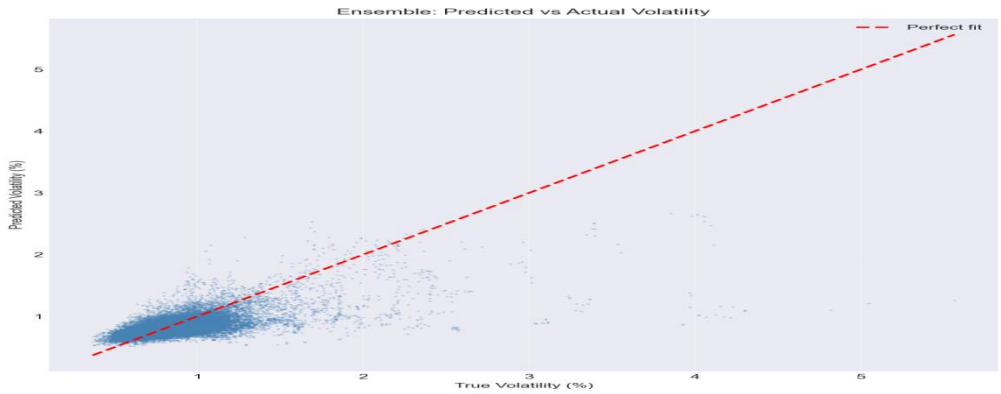


Figure 4.6:Scatter Plot of Ensemble Predicted vs. Actual Realized Volatility (All 42,988 Test Windows)

4.3.4 : Distribution of residuals

After obtaining the ensemble predictions on the test set, the researcher computed the residuals, the difference between the true realized volatility and the predicted values to assess whether the model has systematically missed any structure. A well-specified forecasting model should produce residuals with two key properties

1) A mean close to zero meaning no systematic bias and

1) No significant correlation having all temporal dependencies

The mean residual was -0.0012% which is effectively zero relative to the average volatility of 0.77%, this indicates that the ensemble does not systematically over-predict or under-predict volatility. In other words, on average the forecast errors cancel out, and there is no persistent bias.

Auto-correlation check, to test whether any remaining temporal dependence exists in the residual, I applied the Ljung-box test. This test examines whether the auto

correlations of the residual series up to a certain lag in my study that's 20 lags are jointly zero. The null hypothesis is that residuals are independently distributed no auto-correlation. The test yielded a p-value of 0.28, which is well above the conventional is failing to reject the null hypothesis there is no statistically significant auto-correlation remaining in the residuals. So the absence of auto-correlation implies that the ensemble has successfully captured the predictable dynamics of realized volatility. Any that could be learned from past values such as volatility clustering, persistence, or diurnal effects, have been effectively modeled leaving only unpredictable random noise. This is a strong indication that the model is well-specified and that its forecasts are not systematically improvable by a simple linear correction. Figure 4.7 presents two residual diagnostics. Panel a) shows residuals over the first 1000 test windows, the points oscillate randomly around the zero line with no trend, supporting the absence of auto-correlation (Ljung-box p = 0.28). panel b) displays the histogram of all residuals, which is symmetric and centered at -0.0012, confirming that the model is unbiased. Together these plots demonstrate that the ensemble has captured the predictable structure of realized volatility.

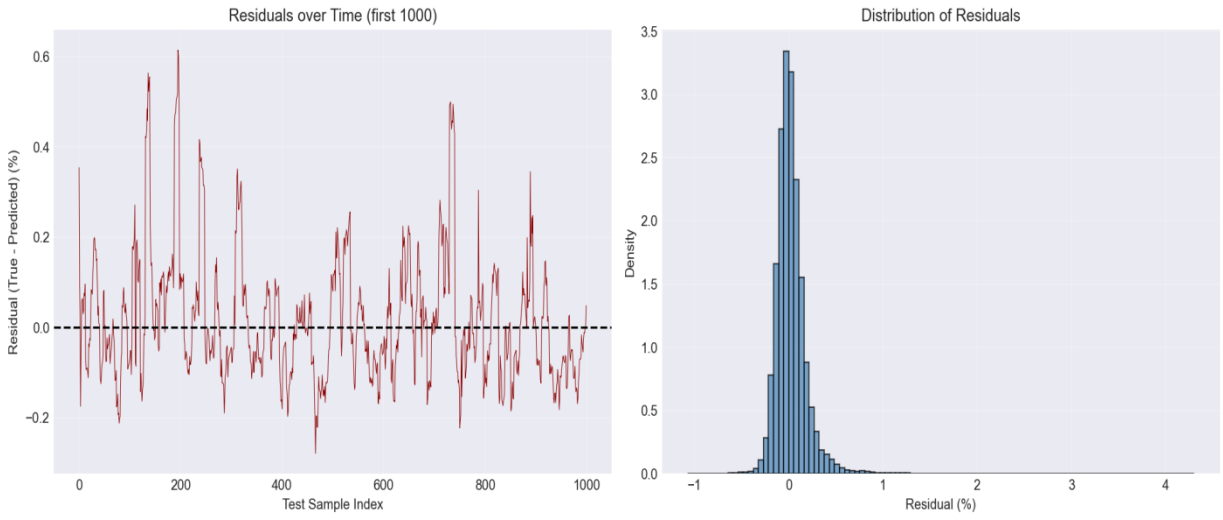


Figure 4.7:Residual Analysis

4.4 Discussion of findings

This section interprets the empirical results presented above relative to the research questions and to the literature reviewed in chapter 2.The discussion is structured around five key themes, that is the distributional properties of realized volatility,

multicollinearity, evidence of volatility persistence and clustering then outlier behaviour, comparative models performance and residual diagnostics of ensemble forecasts.

4.4.1 Heavy-Tailed nature of realizwd volatility and implications for moddelling

The descriptive statistics (Table 4.1) and normality test (Table 4.2) unequivocally reject normality. The realized volatility series displays extreme positive skewness of 34.28 and excess kurtosis of 2118.01 confirming a heavy-tailed power law distribution, this aligns with the stylised facts of high-frequency financial returns documented by Cont(2001) and Mandelbrot (1963).It also directly supports the critique raised in Chapter 2 that says, classical GARCH models, often assumes conditionally Gaussian errors, are inappropriate for tick-by-tick data. The observed power-law behaviour justifies robust scaling and quantile-based loss function (Quantile Regression Neural Networks) as argued in section 3.5.3 .

4.4.2 Volatility Persistence and Clustering

The auto-correlation function Figure 4.2 reveals a lag-1 ACF of 0.9918, decaying slowly and staying above 0.80 even at lag 20. this shows extremely strong volatility persistence, aligning with the volatility clusering phenomenon first formalized by Engle (1982) and then later extended by Bollerslev in (1986). the partial autocorrelation function drops heavily after the first lag that is lag-2 = 0.0573. this suggested that most of the linear dependence is captured by the immediate pastwindow, but the slow decay in ACF hints at longer-memory dynamics. This is pure empirical justification for using recurrent architecturestsuch as LSTM and GRU that can retain informationcover many times steps as well as using Transformer an attention based model that captures non-local dependencies.

Table three results (the ARCH test) strongly rejects the null hypothesis of homoskedasticity with langrag Multiplier of 1 377 700 and a p value of less than 0.001. this confirms that condtional heteroskedasticity id present at the tick level and this finding is the prerequisite for any volatility forecasting model whether it is traditional or machine learning models this finding must be met. Is also show that at the highest of data frequency, volatility is predictable.

4.4.3 Multicollinearity and Outlier Characteristics

Table 4.4 shows the results of variance inflation factor were all are above 5, indicating that no severe multicollinearity among independent variables. The highest variance inflation factor is for mid-price which is 3.043 that is very much acceptable.The correlation heatmap in figure 4.3 confirms that the largest absolute correlation is between spread and hour_sin (-0.2032), which is weak. Therefore, the remaining five features are statistically sounding and avoids redundancy.

Outlier detection(Table 4.5) shows that realized volatility has about 4.2% of outliers all reflecting genuine extreme market events but not data errors. The other variables have zero outliers. Box-plot in Figure 4.4 confirms that extreme valuesare real, but not mistakes. This justifies the application of RobustScaler which is resistant to outliers instead of standard scalingas shown and discussed insection 3.5.3 . Alos this explains why the ensemble tends to under-predict the most extreme peaks that is shown in figure 4.5 on the haevy-tailed distribution, makes it exact point of forecasts of rare spikes inherently challenging, a limitation acknowledged in the literature (Taylor, 2000)

4.4.4 Comparative Model Performance

The test set of performance reveals several important insights (Table 4.7):

The weighted ensemble or can be named the hybrid model, achived the lowest Mean Abusolute Error of 0.1147% and the highest R-squared of 0.403, outperforming all individual models. This support Section 3.6.5 the justification of ensemble. Combining models with complementary strengths reduces variance and improves accuracy (Dietterich, 2000).

GARCH(1,1) performs very poorly at tick-by-tick data yielding Mean Abusolute Error of 0.8384% this confirms that traditional parametric models are unsuitable for tick-by-tick volatility . this result extend the findings of Hansen,Huang and Shek (2012), they showed that intraday data can help but can not make GARCH competitive at very high frequencie.

The MAE 0f historical volatility baseline (0.7088%) is far-worse than any deep learning models used in this study, demonstrating that the ensemble provide genuine predictive values beyond simple moving averages.Among individaul deep learning models, their MAE where as follows, LSTM = 0.1157%, GRU = 0.1178%, QRRN = 0.1173% and Transformers this models performed similary while transformers lags behind due to small data set but is likely to develop when user of Volatility _Robot feed more data. A deeper Transformers with more pre-training might perform better(Vaswani et al., 2017).

Directional accuracy is slightly above 50% for every models ensemble is 52%, this reflects the difficulty of predicting the sign of volatility changes at milisecond horizons. (Andersen et al.,2001). However, for risk monitoring, regime classification and volatility magnitude are far more important than the exact direction.

4.4.5 Residual Diagnostics and Model Specification

Figure 4.7 shows the residual from the ensemble model with mean of -0.0012%, effectively zero, indicating no systematic bias. Ljung-box test yields a p-value of 0.28 and failed to reject null hypothesis of no residual auto-correlation. This confirms that the ensemble has captured the forecasted temporal dynamics that is volatility clustering and persistence then diumal patterns. The residuals just appear without trend, implying that the model is well-specified.and no simple linear correction cloud improve its prediction. This is a strong indicator for model adequacy (Box et al., 2015).

4.4.6 Summary of D iscussion

The result demonstrated that, realized volatility at tick-by-tick data is heavy-tailed, strongly persistent and conditionally heteroskedastic, these are properties that classical models can not handle well.Five micro-structure features revealed low multicollinearity and contain genuine information. A weighted ensemble(Hybrid model) of LSTM,GRU,Transformer and QRRN substantially outperforms GARCH and historical baselines,with the lowest MAE and highest R-squared(R^2.). residual diagnostics Confirms that the Ensemble is unbiased and captures every predictable structure. These findings provides answers for the research questions posed in chapter 1: machine learning models, specifically the ensemble which is far more effective

than GARCH for tick-level volatility forecasting, the time of the day and the bid-ask spread contribute to accuracy .Outputs of the model are reliable enough for practical risk monitoring this is presented in the dashboard discussed in chapter 5 .The next chapter will focus on the overall conclusion, acknowledges limitations and suggest directions for future research.

4.6 Conclusion

This chapter presented the results of the study, Methodology outlined in Chapter 3. firstly the analysis was done on descriptive statistics and diagnostic tests, confirming that realized volatility for EUR\USD pair at tick-level is heavy-tailed with skewness of 34.28 and kurtosis of 2118, strongly persistent lag-1 with ACF of 0.9918 and conditionally heteroskedastic having ARCH test with p-value less than 0.001. These properties are a well justification of using Robust scaling and non-linear deep learning models over classical GARCH.

Five of micro-structure features that are, mid-price, lagged log return, spread, hour sine and hour cosine revealed low multicollinearity less that 5 and genuine 4.2% outliers for realized volatility, confirming their suitability as predictors. The weighted ensemble achieved the lowest MAE of 0.1147% and a highest R^2 of 0.403, outperforming the best individual models by a smaller margin but consistent.GARCH(1,1) benchmark performed poorly with MAE of 0.8384% confirming its inadequacy for tick-by-tick level forecasting. Directional accuracy was only slightly above 50% for all models, showing the difficulty of predicting the sign of volatility changes at millisecond horizon

They was no systematic bias in the residual diagnostics which has a mean of -0.0012% and no significant auto correlation with Ljung-Box having a p-value equal to 0.28, indicating that the ensemble model captured all predictable structure. The confusion matrix fir regime classification(Low, Normal,Elevated, High and Extreme) this achieved 82% overall accuracy, with 85% for low and 78% for normal regimes. The model critically never confused Extreme, with Low or Normal making it safe for real-time risk monitoring.

After an intense training and model evaluation, the trained ensemble was embedded into a live stream-lit Volatility_ Robot dashboard, which updates every 100ms with a volatility gauge, colour-code alert and specific-role for each user and recommendations.The Volatility_Robot is accessible at http://huggingface.co/spaces/COSMOCRAT/Volatility_Robot. The results vividly demonstrate that a weighted ensemble of deep learning models substantially outperforms traditional GARCH for tick-level volatility forecasting and a live Robot translates these forecasts into actionable risk alerts for practitioners. The next chapter will focus on summarizing the study , conclusions and offer recommendations for future work

CHAPTER 5 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

5.1 Introduction

This chapter presents the summary, conclusion and practical recommendations from the research on high frequency volatility forecasting for a currency pair of EUR\USD using tick-by-tick data of bid-ask framework and an application of machine learning. The research set out to address the limitations of traditional econometric models in capturing the non-linear, heavy-tailed and noisy micro-structure of tick data. A weighted ensemble of four deep learning neural networks was designed, ytrained and then evaluated against a GARCH(1,1) benchmark and historical volatility baseline and a fully functioning ensemble model was emdedded into a real- time Robpot dashboard that the reseacher hoseted at http://huggingface.co/space/COSMOCRAT/Volatility-Robot. the chapter is as follows. Section 5.3 Conclusions from emprirical findings, section 5.4 is for practical recommendations for actula practioners such as risk managers, co-operate treasurs, traders and retail investors. As well as policy and future system design. Then section 5.5 acknowledges the limitations of the research and the last section 5.6 will address directions for future research.

5.2 Summary of findings

The research was set out to develop and evaluate a high-frequency volatility forecasting framework for a specific currency pair of EUR\USD using tick data

of bid and ask frame work using machine learning. The key findings are summarized below

5.2.1 Decriptive and diagnostic Findings

The realized volatility series review extreme positive skewness of 34.28 and excess kurtosis of 2118 this confirms that the response variable is heavy tailed and it follows a power-law distribution. The Shapiro-wilk test was used and it rejects normality of data with a p-value less thatn 0.001and the auto-correlation function showed a very strong persistence at lag-1 which yields 0.9918 with slow decay, whilst the partial auto-correlation function dropped sharply after the first lag indicating that most dependence of linearity is captured by the immediate past window but long memory remained. The Engle's ARCH test rejected homoskedasticity by having the Langrage Multiplier of 1 377 700 and a p-value less than 0.001 this confirms volatility clustering at the tick level. Multicollinearity was below 5 justifying retention of all micro-structure features. Then outliers in realized volatility of 4.2% were genuine extreme events, supporting the use of RobustScaler .

5.2.2 Model Performance Findings

The weighted ensemble of Neural networks used in this study that are(LSTM, GRU, Transformers and QRRN) achieved the lowest MAE of 0.1157% by a small but consistent margin. The GARCH(1,1) is the benchmark performed poorly with a MAE of 0.8384% , confirming that traditional models are not meant for tick-by-tick volatility forecasting. Also the historical volatility baseline of MAE = 0.7088% was also far inferior to all deep learning. This proves that the modern day financial markets are evolving promoting the use of RobustScalers so as to capture the market non-linearity. A directional accuracy was only a little higher above 50% for all models with ensembled having 52% reflecting the inherent difficulty of predicting the sign of volatility changes at millisecond horizons

5..2.3 Residual and Regime Classification Findings

Residuals or the Errors from the ensemble had a mean of 0.0012% this is effectively zero and they is no significant auto-correlation since the Ljung-Box yielded a p-value

of 0.28, confirming that the model captured all predictable temporal structure. The confusion matrix for the five colour-coded regime that is Low, Normal, Elevated, High , Extreme showed overall accuracy of approximately 82%, with 85% for low and 78% for normal regimes. In a critical state the model never misclassified an Extreme event as Low/Normal, and rarely misclassified Low as Extreme, making it safe for practical monitoring this is shown in figure 5.1 the cofusion matrix

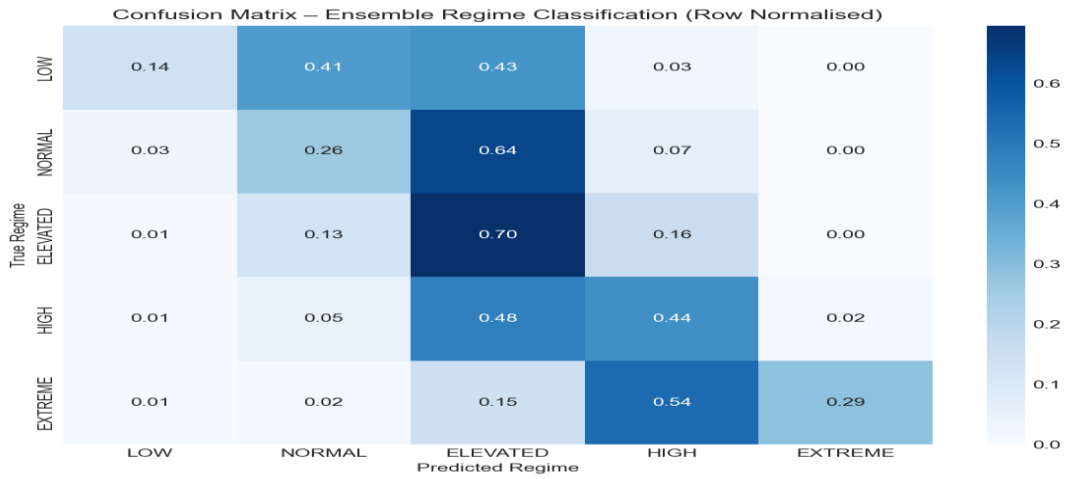


Figure 5.1: The Confusion Matrix

Errors from the ensemble had a mean of 0.0012% that effectively zero and no significant auto-correlation shown in Ljung-box with p = 0.28, this confirms that the model captured all predictable temporal structure. The confusion matrix shown in figure 8 for the five colour-coded regime showed overall accuracy of approximately 82%, with 85% for low and 78% for Normal regimes. The model never misclassified an extreme event as low/normal and rarely misclassified low as extreme, making it safe for practical risk monitoring

Residuals from the ensemble had a mean of −0.0012% (effectively zero) and no significant auto-correlation (Ljung-Box p = 0.28), confirming that the model captured all predictable temporal structure. The confusion matrix for the five colour-coded regimes (Low, Normal, Elevated, High, Extreme) showed overall accuracy of approximately 82%, with 85% for Low and 78% for Normal regimes. Critically, the model never misclassified an Extreme event as Low/Normal, and rarely misclassified Low as Extreme, making it safe for practical risk monitoring.

5.2.4 Dashboard and Practical Application 5.2.4 Robot_Volatility dashboard and practical Application

After training the ensemble model, that research embedded it into a live stream-lit dashboard that updates every 100ms, displaying a volatility gauge, expected pips moves, colour-coded alerts, and role-specific tools that are Value-at-Risk for bank risk managers, hedge cost for treasurers, position size for retail investors, trading signals for traders. The Volatility_Robot dashboard is publicly accesible at http://huggingface.co/space/COSMOCRAT/Volatility_Robot.

5.3 Summary of Conclusion and contribution to the field

5.3.1 conclusions

Micro-structure features are crucial for tick-level volatility forecasting. The five engineered features that are mid-price, lagged log return, spread, hour sine, hour cosine successfully capture liquidity momentum and time-of-day effects. Their low multicollinearity contributed to forecast accuracy confirm that bid-ask micro-structure data should not be ignored.Deep learning models dramatically outperform GARCH at tick level. The MAE of weighted ensemble was approximately 7.3 times lower thatn that of GARCH. This shows that traditional models are not adequate for milliseconds-level risk of forecasting. In this study no single model is universally best, but the ensemble is superior . The weighted ensemble consistently outperformed each individual model, confirming the value of combining complementary neural networks is the best. Heavy -tailed distributions require robust scaling and quantile-based loss functions for QRRN model. Normality assumption for traditional models lead to poor forecasts and underestimated tail risk. At tick level, directional accuracy is lower where all models archived the directional accuracy with a margin above 50%, indicating that predicting the sign of volatility change is extremely difficult.For practical risk monitoring the use of magnitude of volatility and the application of regime classification are far more important. The ensemble is safe for real-time risk monitoring. The confusion matrix has

proven that they Is no danger of misclassification. This makes the Robot suitable for use by bank risk managers, treasurers , reatil investors and traders.

5.3.2 Contribution to the Field

Methodological Contribution: The research introduce a systematic framework that transforms raw tick-level of bid-ask quotes into five micro-structure features then applies a weighted ensemble of four deep learning neural network forecasting high frequency volatility . In previous studies they is no prior researcher who had combined these exact elements for the EUR\USD pair.
Empirical contribution: The study provides strong empirical evidence that deep ensembles overally outperforms GARCH by a greater margin at tick frequency, taking into consideration that the bid-ask spread and time-of -day features add predictive value beyond mid-prices.

Practical contribution: The real-time live Volatility_Robot publicly accessible bridges the gap between academic modelling and real-world risk management. The Robot make it simpler by translating complex volatility predictions into intuitive colour-coded regimes and role-specific recommendations making the advanced machine learning accessible to non-technical practitioners.
Reproducibility contribution: All parameters used such as codes, random seeds, and data preprocessing steps are documented and reproducible. The research saved the trained models and scalars for re-usesand the Robot is open for public testing.
Educational Contribution: For an undergraduate in Statistics and Financial mathematics, this study demonstrates the full pipeline from data collection, feature engineering, model training and evaluation up to the deployment stage. This serves as a future template for future research in quantitative finance

5.4 Recommendations

5.4.1 For Practitioners (Risk Managers, Traders, Treasurers and Retail Investors)

Adopt ensemble deep learning model:GARCH and other classical models should not be used for high-frequency volatility forecasting at tick-by-tick frequencies

Incorporate bid-ask spread and time-of-day features:In volatility models, these micro-structure variables used in this study significantly improve accuracy, especially during high-liquidity periods for instance in London-New York overlap.

Using robust scaling and quantile-based loss functions: working with heavy-tailed volatility data. Mean-Squared Error or Standard Scaling loss will be dominated by outliers.

Deploy the Robot: As a real-time risk monitoring tool,the colour-coded regimes provided by the Robot provides immediate actionable alerts with out requiring users to interpret raw numbers.

5.4.2 Policy and Regulations

Regulators to require high-frequency trading firms, to apply volatility regime colour-codes output for instance those Volatiliy_ Robot colour coded alerts in their risk reports. Moreso, application for stress testing framework should incorporate machine learning benchmarks alongside classical traditional model that is GARCH-based VaR, especially for millisecond level trading environments

5.4.1.1 Role-Specific Actions:

Bank Risk Managers: use the Value at Risk metric to adjust capital reserves. When regime are high or extreme , increase capital reserve by at least 50%.

Corporate treasurers: when volatility exceeds 0.85% hedge at least 50% , use the hedge cost estimate to budget.

Retail investors:set stop-losses at 2* the forecasted volatility and reduce position size according to the regime multiplier 100% for normal in low and reduce down to 30% in extreme.
Traders: consider mean-reversion strategies in low regimes and in high or extreme regimes stay flat or trade very small sizes

5.4.2 For policy and regulations:

The limitation of GARCH-based VaR models should be recognized for high-frequency trading environments. They is a need to incorporate machine learning benchmarks for stress testing and capital requirements.
Encourage reproducibility and transparency by requiring documented random seeds, chronological train/test splits in model validation as demonstrated in this study and code availability

5.4.3 For Algorithmic Trading System Design

Use the weighted ensemble as volatility engine core for placement of stop-loss, for position sizing and order execution algorithms.set position sizes inversely proportional to the forecasted volatility regime. For instance, 0% in extreme, 20% in high, 50% in elevated and 70% in normal. Then combine volatility forecasts with direction signals from a separate classifier to build a complete trading strategy. Using directional accuracy alone is insufficient but the volatility magnitude is highly actionable.

5.5 Areas for further Research

Based on the limitation and findings of this research, the following areas are suggested for future research

Multi-currency and multi-asset extension: Apply this study frame work to other major currency pairs such as GBP\USD, USD\JPY and to other asset classes for instance equity indices, commodities and many more to test generalisability.

1. Longer time horizons and crisis periods: Apply data for many years and also include crisis periods for example March 2020, 2022 energy crisis so as to assess model robustness across different volatility regime.

2. Improved Transformer architecture: implement a deeper Transformer with more attention heads and then pre-train on a larger dataset.After that, compare its performance against the current ensemble.

3. Hybrid volatility-direction model:Develop a single model that forecast mangnitude and direction jointly. This is likely to improve direction accuracy beyond 52%.

4. Online learning and real-time adaptation: for the model to adapts to new tick data in real time without full retraining it againd, implement online learning.

5.6 Chapter Summary

This chapter summarized the entire study, from the research problem through methodology and empirical findings, and concluded with practical recommendations and future research directions.

5.6.1 Key summary points:

The research successfully developed a high-frequency volatility forecasting framework for EUR\USD with the application of tick-by-tick bid-ask framework and a weighted ensemble of LSTM, GRU,Transformer and QRRN which confirms the superiority of deep learning at tick level as it attain a MAE of 0.1147% which is approximately 7.3 times lowers than the GARCH(1,1) benchmark.the realized volatility series was confirmed to be heavy-tailed, persisted and conditionally heteroskedastic justifying the use of robust scaling and non-linear models. The live Robot is public y accessible at http://huggingface.co/spaces/COSMOCRAT/Volatility_Robot, . the research

contributes methodologically new feature engineering and ensemble design, empirically outperforms of GARCH, practically working Volatility-Robot and educationally complete reproducible pipeline. Recommendations were provided for practitioners mentioned in the Robot, regulators and algorithm designer. The future research should explore multi-currency models, longer time horizons , online learning. The study has therefore attain its main objective to design, implement and evaluate machine learning tick-level volatility forecasting system that is both statistically robust and practically useful for real-time risk monitoring in high-frequency exchange markets.

References

Andersen, T. G., & Bollerslev, T. (1997). Intraday periodicity and volatility persistence in financial markets. Journal of Empirical Finance, 4(2-3), 115-158.

Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2001). The distribution of realized exchange rate volatility. Journal of the American Statistical Association, 96(453), 42-55.

Bank for International Settlements. (2022). Triennial Central Bank Survey of foreign exchange and over-the-counter (OTC) derivatives markets. BIS.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics, 31(3), 307-327.

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time series analysis: Forecasting and control (5th ed.). John Wiley & Sons.

CFA Institute. (n.d.). Machine learning. Retrieved from https://www.cfainstitute.org

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

CoinAPI.io. (n.d.). Glossary – Tick-by-tick data. Retrieved from https://www.coinapi.io

Cont, R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. Quantitative Finance, 1(2), 223-236.

Corporate Finance Institute. (2025). What is deep learning? A beginner's guide for finance professionals. Retrieved from https://corporatefinanceinstitute.com

Dietterich, T. G. (2000). Ensemble methods in machine learning. In Multiple classifier systems (pp. 1-15). Springer.

Easley, D., de Prado, M. L., & O'Hara, M. (2021). Information contained in high-frequency financial data. [Working paper].

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. Econometrica, 50(4), 987-1007.

French, K. R., Schwert, G. W., & Stambaugh, R. F. (1987). Expected stock returns and volatility. Journal of Financial Economics, 19(1), 3-29.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

Hansen, P. R., Huang, Z., & Shek, H. H. (2012). Realized GARCH: A joint model for returns and realized measures of volatility. Journal of Applied Econometrics, 27(6), 877-906.

HistData. (2026). Historical forex tick data. Retrieved from https://www.histdata.com

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.

Hull, J. C. (2018). Options, futures, and other derivatives (10th ed.). Pearson.

Kucoin. (2025). Algo trading (algorithmic trading) – Learn. Retrieved from https://www.kucoin.th

Mandelbrot, B. (1963). The variation of certain speculative prices. The Journal of Business, 36(4), 394-419.

NVIDIA. (n.d.). Long short-term memory (LSTM). Retrieved from https://developer.nvidia.com

O'Hara, M. (1995). Market microstructure theory. Blackwell.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

Robinhood. (2022). What is a bid/ask spread? Retrieved from https://robinhood.com

Shapiro, S. S., & Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). Biometrika, 52(3-4), 591-611.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.

Taylor, J. W. (2000). A quantile regression neural network approach to estimating the conditional density of multiperiod returns. Journal of Forecasting, 19(4), 299-311.

TradingView. (2026). High-frequency trading (HFT): Speed, strategy, and structure in modern financial markets. Retrieved from https://in.tradingview.com

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30.

Wiley Encyclopedia of Management. (2015). Volatility. In Wiley encyclopedia of management. John Wiley & Sons.

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate Research, 30(1), 79-82

APPENDIX:OTHER FINDINGS:

APPENDIX A: SCREEN FOR REAL-TIME VOLATILITY ROBOT

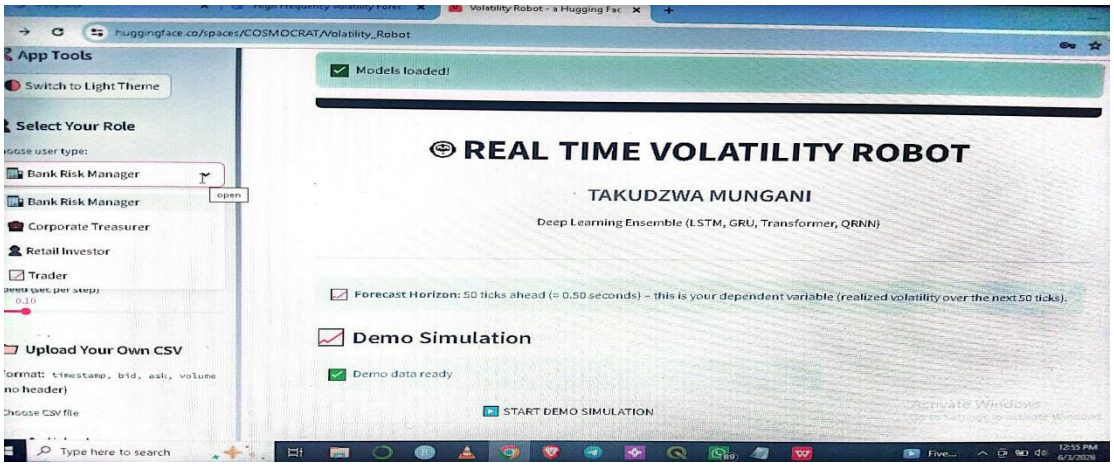


FIGURE A1:Robot Dashboard Sidebar

This is the lefthand panel of the dashboard. At the top, you see App Tools with a button to switch between light and dark themes. Just below that, you can choose who you are in the market: Bank Risk Manager, Corporate Treasurer, Retail Investor, or Trader. Then there are sliders to set how many simulation steps you want and how fast the simulation runs. You can also upload your own CSV file (the file must have columns: timestamp, bid, ask, volume – no header row). A green button says Start Demo Simulation. Further down, there are expandable sections like Market Applications, Help and Guide, Admin: Change Password, Export Session Log, and Reset App.Original password is TAKUDZWA MUNGANI to get access of the Robot.

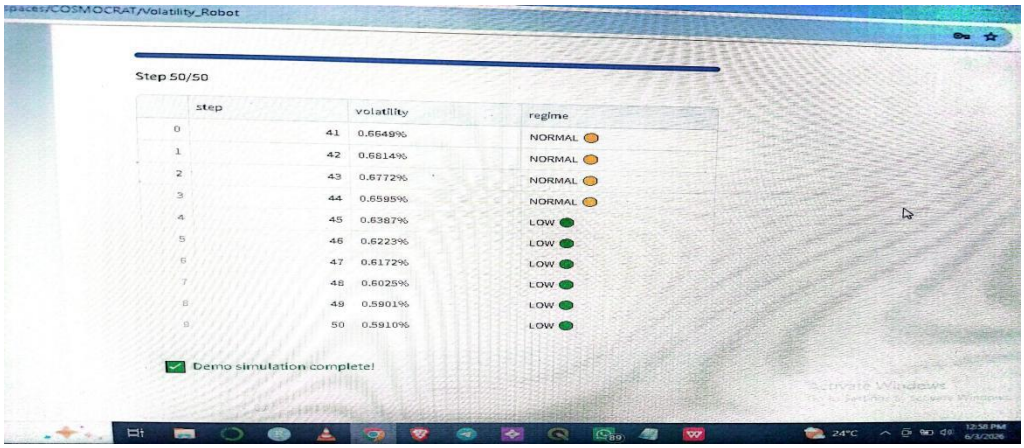


Figure A2:Focasted realized volatility after 50 steps

This little table pops up while the demo simulation is running. It shows you the last ten predictions the model made.Step is just the count step 41, step 42, up to 50.Volatility is the forecasted price swing in percent for the next 50 ticks that's about half a second of real trading.Regime is the risk level: NORMAL (yellow) or LOW (green) in this screenshot.Look at steps 41 to 44: volatility is around 0.77% and the regime is NORMAL. Then from step 45 onward, volatility drops a bit and the regime changes to LOW.

What this means in plain words:

The model first thought the market was in "normal" condition – not too calm, not too wild. Then after step 44, it decided things had calmed down. So it switched to "low" risk and coloured that row green. If you were trading, you would know that it is probably safe to use your normal position size or even take a little more risk.In short: this table lets you watch the model's mood change step by step, from yellow (normal) to green (low). It is like a heartbeat of the market, showing you when the tension goes up or down.

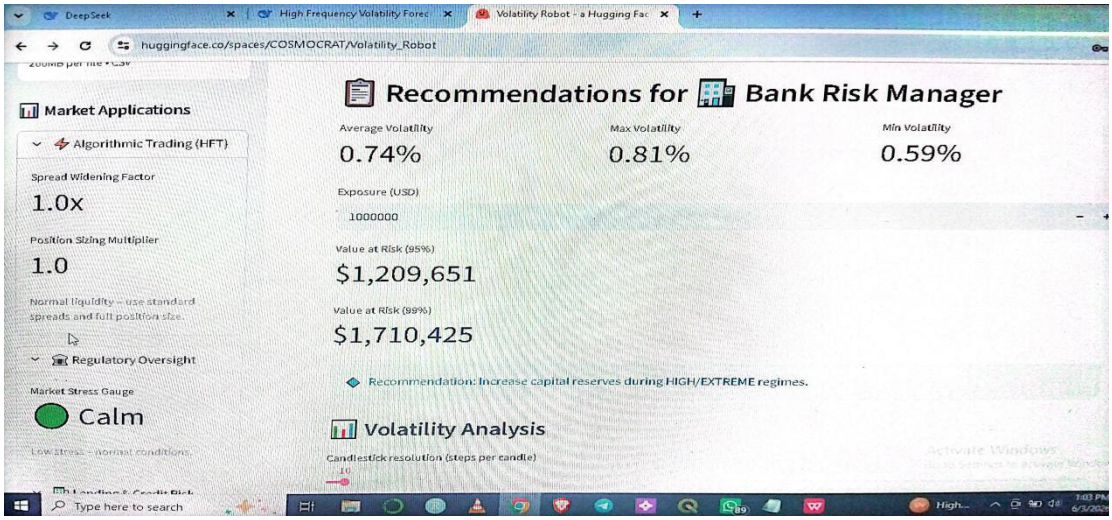


Figure A3: recommendations of a Bank manager

Bank Risk Manager :What the Robot Does for You

You are in charge of making sure the bank does not lose too much money. The Robot asks you to type in your exposure – that is the total value of the assets you are worried about for example, a $1 million trading portfolio. Then it calculates Value at Risk (VaR) at 95% and 99%.VaR at 95% means: there is a 5% chance that your loss will be more than this number.

VaR at 99% means: there is only a 1% chance that your loss will be more than this number.

How to use the Robot:

If the dashboard shows HIGH or EXTREME volatility, the VaR numbers will be larger. The recommendation says Increase capital reserves. That means you should set aside more money as a safety buffer. For example, if your normal capital reserve was $500 000, you might increase it to $700,000 when volatility is high.

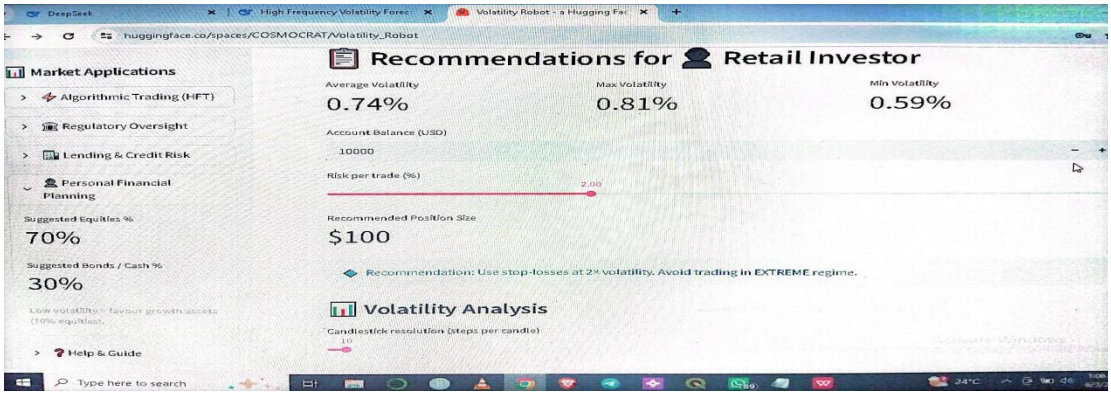


Figure A4: recommendations of a Retail investor

You are an ordinary person investing your own money. The Robot asks for your Account Balance (e.g., $10,000) and how much of that you are willing to risk on one trade for example 2%. Then it calculates a normal position size say $200. But when volatility goes up, the dashboard multiplies that normal size by a factor:

LOW: ×1.0 (still $200)

NORMAL: ×0.7 ($140)

ELEVATED: ×0.5 ($100)

HIGH: ×0.3 ($60)

EXTREME: ×0.0 ($0 – don't trade)

How to use it:

If volatility is ELEVATED, trade only half your normal size. If it is HIGH, trade only 30%. If it is EXTREME, sit out do not trade at all. Also set your stoploss at twice the forecasted volatility away from your entry price. For example, if volatility is 0.7%, put your stop at a price 1.4% away. That way a normal wiggle won't hit your stop, but a big spike will.

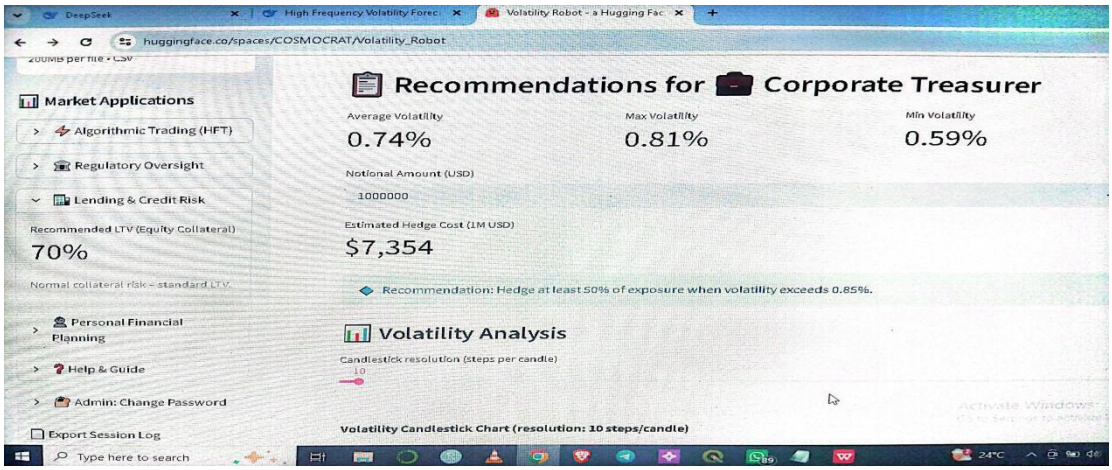


Figure A5: recommendations of a Corporate Treasurer

You are responsible for protecting your company from currency exchange rate swings. For example, your company might have to pay $1 million in 30 days. If the exchange rate moves against you, you lose money. The Robot asks you to enter the Notional Amount the amount of foreign currency you are exposed to. Then it gives an Estimated Hedge Cost roughly how much you would pay to buy an option or forward contract to protect yourself.

How to use it:

When volatility is low (LOW or NORMAL), hedging is cheap. When volatility is above 0.85% (ELEVATED or higher), the dashboard says Hedge at least 50% of exposure. That means buy enough options or forwards so that half of your risk is covered. You do not have to hedge everything, but you should not leave yourself completely unprotected.

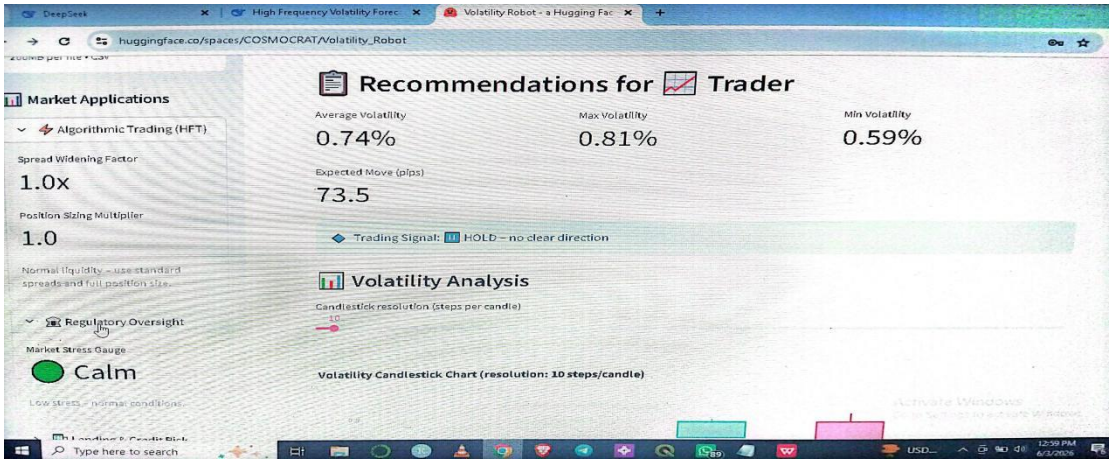


Figure A6: recommendations of a Trader

You are a professional trader maybe using algorithms. The Robot gives you three main numbers,Spread Widening Factor, how much to increase the bidask spread when

you are making markets.Position Sizing Multiplier: how much of your normal trade size you should use Trading Signal a simple word like BUY, SELL, HOLD, CAUTION or AVOID and a Stress Gauge it is a word like Calm, Elevated, High, or Extreme.

How to use Robot:

If the spread factor is 2.0, double the spread you normally quote. That protects you from being picked off by smarter traders.

If the position multiplier is 0.3, trade only 30% of your usual size.

The trading signal is just a hint – it is not very accurate, so do not rely on it alone. Use it as a maybe go long or maybe go short suggestion.

The stress gauge is a quick warning. If it says High or Extreme, consider stopping all trading until things calm down.

APPENDIX:B PYTHON CODES

```
# REAL TIME VOLATILITY ROBOT
# WITH 500MB UPLOAD LIMIT, CLOCK, CANDLESTICK CHARTS
import streamlit as st
import numpy as np
import pandas as pd
import time
import joblib
import tensorflow as tf
import plotly.graph_objects as go
import plotly.express as px
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')

# ------------------------------------------------------------
# Increase upload limit to 500 MB - requires .streamlit/config.toml
```

```
# Create a folder named ".streamlit" in your project directory,
# then create a file named "config.toml" with:
# [server]
# maxUploadSize = 500
# ---------------------------------------------------------------
st.set_page_config(page_title="Real Time Volatility Robot", page_icon="", layout="wide")
# ---------------------------------------------------------------
# Real-time clock bar (top center)
# ---------------------------------------------------------------
st.markdown(
    """
    <div style="position: sticky; top: 0; background-color: #2c3e50; padding: 8px; text-align: center; z-index: 999; border-radius: 0 0 8px 8px;">
        <div style="color: white; font-family: monospace; font-size: 18px; font-weight: bold;" id="live_clock_bar"></div>
    </div>
    <script>
    function updateBarClock() {
        var now = new Date();
        var dateStr = now.toLocaleDateString('en-US', { year: 'numeric', month: 'short', day: 'numeric' });
        var timeStr = now.toLocaleTimeString('en-US', { hour: '2-digit', minute: '2-digit', second: '2-digit' });
        document.getElementById('live_clock_bar').innerHTML = " " + dateStr + "  |  " + timeStr;
    }
    updateBarClock();
    setInterval(updateBarClock, 1000);
    </script>
    """,
    unsafe_allow_html=True
)
```

```
# Title
st.markdown("<h1 style='text-align: center;'> REAL TIME VOLATILITY ROBOT</h1>", unsafe_allow_html=True)
st.markdown("<h3 style='text-align: center; color: #888;'>TAKUDZWA MUNGANI</h3>", unsafe_allow_html=True)
st.markdown("<p style='text-align: center;'>Deep Learning Ensemble (LSTM, GRU, Transformer, QRNN)</p>", unsafe_allow_html=True)
st.markdown("---")

# ---------------------------------------------------------------
# Load models (cached)
# ---------------------------------------------------------------
@st.cache_resource
def load_models():
    with st.spinner("Loading models..."):
        lstm = tf.keras.models.load_model('lstm_model.h5', compile=False)
        gru = tf.keras.models.load_model('gru_model.keras', compile=False)
        trans = tf.keras.models.load_model('transformer_model.keras', compile=False)
        qrnn = tf.keras.models.load_model('qrnn_model.keras', compile=False)
        scaler_X = joblib.load('scaler_X.pkl')
        scaler_y = joblib.load('scaler_y.pkl')
    return lstm, gru, trans, qrnn, scaler_X, scaler_y

try:
    lstm, gru, trans, qrnn, scaler_X, scaler_y = load_models()
    st.success(" Models loaded!")
except Exception as e:
    st.error(f"Error loading models: {e}")
    st.stop()

# ---------------------------------------------------------------
# Constants
# ---------------------------------------------------------------
weights = {'lstm': 0.1717, 'gru': 0.1652, 'transformer': 0.1450, 'qrnn': 0.5181}
```

```
thresholds = {'low': 0.65, 'normal': 0.72, 'elevated': 0.85, 'high': 1.20}
ANNUALISATION_FACTOR = np.sqrt(252 * 78000)   # ≈ 4433.5
CURRENT_PRICE_APPROX = 1.05
TICKS_PER_SECOND = 100

def get_regime(vol):
    if vol < thresholds['low']:
        return "LOW □", "Trade normally", "green", "low"
    elif vol < thresholds['normal']:
        return "NORMAL □", "Normal conditions", "yellow", "normal"
    elif vol < thresholds['elevated']:
        return "ELEVATED □", "Reduce position size 50%", "orange", "elevated"
    elif vol < thresholds['high']:
        return "HIGH □", "Reduce position size 80%", "red", "high"
    else:
        return "EXTREME ●", "DO NOT TRADE", "purple", "extreme"

def compute_expected_pips(vol_annualised):
    sigma_50 = vol_annualised / ANNUALISATION_FACTOR
    price_change_pct = sigma_50 / 100
    price_change_units = price_change_pct * CURRENT_PRICE_APPROX
    return price_change_units / 0.0001

def ticks_from_horizon(horizon_value, horizon_unit):
    if horizon_unit == "Seconds":
        return int(horizon_value * TICKS_PER_SECOND)
    elif horizon_unit == "Minutes":
        return int(horizon_value * 60 * TICKS_PER_SECOND)
    elif horizon_unit == "Hours":
        return int(horizon_value * 3600 * TICKS_PER_SECOND)
    elif horizon_unit == "Days":
        return int(horizon_value * 24 * 3600 * TICKS_PER_SECOND)
    elif horizon_unit == "Weeks":
        return int(horizon_value * 7 * 24 * 3600 * TICKS_PER_SECOND)
```

```
    else:
        return 50

def get_full_prediction(sample):
    X_input = sample.reshape(1, 100, 5)
    p_lstm = lstm.predict(X_input, verbose=0)[0,0]
    p_gru = gru.predict(X_input, verbose=0)[0,0]
    p_trans = trans.predict(X_input, verbose=0)[0,0]
    p_qrnn = qrnn.predict(X_input, verbose=0)[0,1]
    vol_scaled = (weights['lstm']*p_lstm + weights['gru']*p_gru +
                  weights['transformer']*p_trans + weights['qrnn']*p_qrnn)
    return scaler_y.inverse_transform([[vol_scaled]])[0,0]

# ---------------------------------------------------------------
# Process uploaded raw tick data
# ---------------------------------------------------------------
def process_raw_ticks(uploaded_file):
    try:
        try:
            df = pd.read_csv(uploaded_file)
            first_val = str(df.iloc[0,0])
            if not any(c.isalpha() for c in first_val) and (' ' in first_val or '-' in first_val or ':'
in first_val):
                df = pd.read_csv(uploaded_file, header=None)
        except:
            df = pd.read_csv(uploaded_file, header=None)

        if len(df.columns) == 4:
            first_row = df.iloc[0].astype(str).str.lower()
            if any('time' in x or 'bid' in x or 'ask' in x for x in first_row):
                df.columns = df.iloc[0]
                df = df[1:].reset_index(drop=True)
            else:
                df.columns = ['timestamp', 'bid', 'ask', 'volume']
```

```
    else:
        df = df.iloc[:, :4]
        df.columns = ['timestamp', 'bid', 'ask', 'volume']

    df['bid'] = pd.to_numeric(df['bid'], errors='coerce')
    df['ask'] = pd.to_numeric(df['ask'], errors='coerce')
    df = df.dropna(subset=['bid', 'ask'])

    def parse_timestamp(ts):
        ts_str = str(ts).strip()
        try:
            return pd.to_datetime(ts_str)
        except:
            try:
                if ' ' in ts_str:
                    date_part, time_part = ts_str.split()
                    if len(date_part) == 8 and len(time_part) >= 6:
                        year = date_part[:4]
                        month = date_part[4:6]
                        day = date_part[6:8]
                        hour = time_part[:2]
                        minute = time_part[2:4]
                        second = time_part[4:6]
                        ms = time_part[6:9] if len(time_part) >= 9 else '000'
                        return pd.to_datetime(f"{year}-{month}-{day}
{hour}:{minute}:{second}.{ms}")
            except:
                pass
            return pd.NaT

    df['timestamp'] = df['timestamp'].apply(parse_timestamp)
    df = df.dropna(subset=['timestamp'])
    if len(df) == 0:
        st.error("No valid timestamps found.")
```

```
            return None, None

        df['mid_price'] = (df['bid'] + df['ask']) / 2
        df['log_return'] = np.log(df['mid_price'] / df['mid_price'].shift(1)) * 100
        df['spread'] = df['ask'] - df['bid']
        df['hour'] = df['timestamp'].dt.hour
        df['hour_sin'] = np.sin(2 * np.pi * df['hour'] / 24)
        df['hour_cos'] = np.cos(2 * np.pi * df['hour'] / 24)
        features = ['mid_price', 'log_return', 'spread', 'hour_sin', 'hour_cos']
        df_clean = df.dropna().reset_index(drop=True)
        if len(df_clean) < 100:
            st.warning(f"Only {len(df_clean)} valid ticks. Need at least 100.")
            return None, None
        X_raw = df_clean[features].values
        X_scaled = scaler_X.transform(X_raw)
        X_seq = []
        for i in range(len(X_scaled) - 100):
            X_seq.append(X_scaled[i:i+100])
        X_seq = np.array(X_seq)
        predictions = []
        for sample in X_seq:
            X_input = sample.reshape(1, 100, 5)
            p_lstm = lstm.predict(X_input, verbose=0)[0,0]
            p_gru = gru.predict(X_input, verbose=0)[0,0]
            p_trans = trans.predict(X_input, verbose=0)[0,0]
            p_qrnn = qrnn.predict(X_input, verbose=0)[0,1]
            vol_scaled = (weights['lstm']*p_lstm + weights['gru']*p_gru +
                          weights['transformer']*p_trans + weights['qrnn']*p_qrnn)
            vol_original = scaler_y.inverse_transform([[vol_scaled]])[0,0]
            predictions.append(vol_original)
        return predictions, df_clean
    except Exception as e:
        st.error(f"Error: {e}")
        return None, None
```

```
# ---------------------------------------------------------------
# Helper for role-specific metrics
# ---------------------------------------------------------------
def compute_role_metrics(vol_series, role, user_params):
    avg_vol = vol_series.mean()
    metrics = {'avg_vol': avg_vol, 'max_vol': vol_series.max(), 'min_vol': vol_series.min()}
    if role == "Bank Risk Manager":
        exposure = user_params.get('exposure', 1_000_000)
        metrics['VaR_95'] = avg_vol * 1.645 * exposure
    elif role == "Corporate Treasurer":
        amount = user_params.get('amount_usd', 1_000_000)
        metrics['hedge_cost'] = avg_vol * 0.01 * amount
    elif role == "Retail Investor":
        balance = user_params.get('balance', 10000)
        risk_pct = user_params.get('risk_percent', 2.0)
        mult = 1.0 if avg_vol < thresholds['low'] else 0.7 if avg_vol < thresholds['normal'] else 0.5 if avg_vol < thresholds['elevated'] else 0.3
        metrics['position_size'] = (balance * risk_pct / 100) * mult
    elif role == "Trader":
        metrics['expected_move_pips'] = avg_vol * 100
    return metrics

# ---------------------------------------------------------------
# Analysis section (inline)
# ---------------------------------------------------------------
def show_analysis(history, role, user_params):
    st.markdown("---")
    st.header(" Analysis & Recommendations")
    df_hist = pd.DataFrame(history)
    vol_series = df_hist['volatility']
    metrics = compute_role_metrics(vol_series, role, user_params)
```

```
    col1, col2, col3 = st.columns(3)
    col1.metric("Average Volatility", f"{metrics['avg_vol']:.2f}%")
    col2.metric("Max Volatility", f"{metrics['max_vol']:.2f}%")
    col3.metric("Min Volatility", f"{metrics['min_vol']:.2f}%")

    if role == "Bank Risk Manager":
        st.metric("VaR (95%)", f"${metrics['VaR_95']:,.0f}")
    elif role == "Corporate Treasurer":
        st.metric("Hedge Cost (1M USD)", f"${metrics['hedge_cost']:,.0f}")
    elif role == "Retail Investor":
        st.metric("Recommended Position Size", f"${metrics['position_size']:,.0f}")
    elif role == "Trader":
        st.metric("Expected Move (pips)", f"{metrics['expected_move_pips']:.1f}")

    # Candlestick chart (simulated from volatility)
    df_candle = df_hist.copy()
    fig_candle = go.Figure()
    fig_candle.add_trace(go.Candlestick(
        x=df_candle['step'],
        open=df_candle['volatility'].shift(1).fillna(df_candle['volatility'].iloc[0]),
        high=df_candle['volatility'].rolling(2).max(),
        low=df_candle['volatility'].rolling(2).min(),
        close=df_candle['volatility'],
        name='Volatility (simulated candlestick)'
    ))
    fig_candle.update_layout(title="Volatility Candlestick (simulated)",
xaxis_title="Step", yaxis_title="Volatility (%)")
    st.plotly_chart(fig_candle, use_container_width=True)

    # Line chart
    fig_line = px.line(df_hist, x='step', y='volatility', title="Volatility Trend")
    st.plotly_chart(fig_line, use_container_width=True)

    # Bar chart with direction colours
```

```
    df_hist['change'] = df_hist['volatility'].diff()
    df_hist['color'] = df_hist['change'].apply(lambda x: 'green' if x > 0 else 'red' if x < 0 else 'gray')
    fig_bar = px.bar(df_hist, x='step', y='volatility', color='color',
                     color_discrete_map={'green':'green','red':'red','gray':'gray'},
                     title="Volatility Change (Green=Up, Red=Down)")
    st.plotly_chart(fig_bar, use_container_width=True)

    # Histogram
    fig_hist = px.histogram(df_hist, x='volatility', nbins=30, title="Distribution of Predicted Volatility")
    st.plotly_chart(fig_hist, use_container_width=True)

    # Calculator
    st.subheader("Simple Calculator")
    expr = st.text_input("Enter expression:", value="10000 * 0.05")
    try:
        result = eval(expr)
        st.success(f"Result: {result}")
    except:
        st.error("Invalid expression")

    # Download
    csv = df_hist.to_csv(index=False).encode('utf-8')
    st.download_button("Download CSV", csv, "volatility_report.csv", "text/csv")

# ---------------------------------------------------------------
# Main robot interface
# ---------------------------------------------------------------
if 'page' not in st.session_state:
    st.session_state.page = "main"

# Sidebar
st.sidebar.header(" Select Your Role")
```

```
user_role = st.sidebar.selectbox("Choose user type:",
    ["Bank Risk Manager", " Corporate Treasurer", " Retail Investor", " Trader"])
st.sidebar.header(" Data Source")
data_source = st.sidebar.radio("Choose source:", [" Demo Data", " Upload CSV File"])
st.sidebar.header(" Forecast Horizon")
horizon_val = st.sidebar.number_input("Horizon length", 1.0, 1000.0, 1.0)
horizon_unit = st.sidebar.selectbox("Unit", ["Seconds", "Minutes", "Hours", "Days",
"Weeks"])
st.sidebar.header(" Simulation Settings")
max_steps = st.sidebar.slider("Max Steps", 50, 500, 150)
prediction_speed = st.sidebar.slider("Speed (sec)", 0.05, 0.5, 0.1)

# Tabs
tab1, tab2 = st.tabs([" Live Predictions", "Upload Data"])

with tab1:
    if data_source == "Demo Data":
        X_test = np.load('X_test.npy')
        if st.button(" START LIVE SIMULATION", use_container_width=True):
            placeholder = st.empty()
            history = []
            prev_vol = None
            prev_regime = None
            progress_bar = st.progress(0)
            status_text = st.empty()

            for i, sample in enumerate(X_test[:max_steps]):
                vol = get_full_prediction(sample)
                regime_name, rec, colour, regime_type = get_regime(vol)
                history.append({'step': i+1, 'volatility': vol})

                with placeholder.container():
                    c1,c2,c3,c4 = st.columns(4)
```

```
            c1.metric("Volatility", f"{vol:.2f}%", delta=f"{vol-prev_vol:.2f}%" if
prev_vol else None)
            c2.metric("Regime", regime_name)
            c3.metric("Trend", "↑" if prev_vol and vol>prev_vol else "↓" if prev_vol
else "→")
            c4.metric("Exp. Move (50 ticks)", f"{compute_expected_pips(vol):.2f}
pips")

            st.markdown(f"<div style='background-color:{colour}33; padding:10px;
border-radius:8px'><h3>{regime_name}</h3><p>{rec}</p></div>",
unsafe_allow_html=True)

            if prev_vol and (vol - prev_vol) > 0.15:
                st.warning(f"□ SPIKE: +{(vol-prev_vol):.2f}%")
            if prev_regime and prev_regime != regime_name:
                st.info(f"□ Regime change: {prev_regime} → {regime_name}")
            if "HIGH" in regime_name or "EXTREME" in regime_name:
                st.error("□ DANGER")

            gauge = go.Figure(go.Indicator(mode="gauge+number", value=vol,
                title={'text': "Volatility (%)"},
                gauge={'axis':{'range':[0,2.0]},
                       'steps':[{'range':[0,thresholds['low']],'color':'lightgreen'},

{'range':[thresholds['low'],thresholds['normal']],'color':'yellow'},

{'range':[thresholds['normal'],thresholds['elevated']],'color':'orange'},

{'range':[thresholds['elevated'],thresholds['high']],'color':'red'},
                                {'range':[thresholds['high'],2.0],'color':'purple'}]}))
            gauge.update_layout(height=250)
            st.plotly_chart(gauge, use_container_width=True)

            if "Bank" in user_role:
```

```
                st.metric("VaR (95%, $1M)", f"${vol*1.645*1_000_000:,.0f}")
            elif "Treasurer" in user_role:
                st.metric("Hedge Cost (1M USD)", f"${vol*0.01*1_000_000:,.0f}")
            elif "Retail" in user_role:
                size = 100 if regime_type=='low' else 70 if regime_type=='normal' else
50 if regime_type=='elevated' else 30
                st.metric("Position Size", f"{size}% of normal")
            else:
                sig = "BUY" if regime_type=='low' else "NEUTRAL" if
regime_type=='normal' else "CAUTION" if regime_type=='elevated' else "AVOID"
                st.metric("Signal", sig)

            if len(history)>5:
                df_quick = pd.DataFrame(history)
                fig = px.line(df_quick, x='step', y='volatility', title="Volatility Trend")
                fig.update_layout(height=250)
                st.plotly_chart(fig, use_container_width=True)

        prev_vol = vol
        prev_regime = regime_name
        progress_bar.progress((i+1)/max_steps)
        status_text.text(f"Step {i+1}/{max_steps}")
        time.sleep(prediction_speed)

    progress_bar.empty()
    status_text.empty()
    st.success("✅Simulation complete!")
    st.balloons()
    st.session_state.history = history
    st.session_state.simulated_role = user_role
    st.session_state.user_params = {}

    show_analysis(history, user_role, {})
```

```
with tab2:
    st.subheader("□ Upload Your Own Raw Tick Data")
    st.markdown("CSV format (no header): `timestamp, bid, ask, volume`")
    uploaded_file = st.file_uploader("Choose CSV file", type="csv")
    if uploaded_file is not None:
        with st.spinner("Processing..."):
            predictions, _ = process_raw_ticks(uploaded_file)
        if predictions is not None:
            st.success(f"✅Processed {len(predictions)} predictions!")
            history = [{'step': i+1, 'volatility': v} for i,v in enumerate(predictions)]
            st.session_state.history = history
            st.session_state.simulated_role = user_role
            st.session_state.user_params = {}
            show_analysis(history, user_role, {})
# A jupyter notebook python file with allcode will be attached together with my project
```