PREDICTION AND TREND ANALYSIS OF FINANCIAL TIME SERIES: DEEP LEARNING AND KERNEL ADAPTIVE FILTERING PERSPECTIVE

Thesis submitted in partial fulfillment of the requirements for the Degree of

DOCTOR OF PHILOSOPHY

By

SHAMBHAVI MISHRA (E18SOE803)



School of Computer Science Engineering and Technology

BENNETT UNIVERSITY (Established under UP Act No 24, 2016) Plot Nos 8-11, Tech Zone II, Greater Noida-201310, Uttar Pradesh, India July, 2022

Dedicated to family, friends, and the research community.

@ Copyright Bennett University, Greater Noida, India
 July, 2022
 ALL RIGHTS RESERVED

DECLARATION BY THE SCHOLAR

I hereby declare that the work reported in the Ph.D. thesis entitled "**Prediction and Trend Analysis of Financial Time Series: Deep Learning and Kernel Adaptive Filtering Perspective**" submitted at **Bennett University, Greater Noida, India** is an authentic record of my work carried out under the supervision of **Dr. Tanveer Ahmed** and **Dr. Vipul Kumar Mishra**. I have not submitted this work elsewhere for any other degree or diploma. I am fully responsible for the contents of my Ph.D. thesis.

Shambhavi Mishra

Enrollment No. E18SOE803 School of Computer Science Engineering and Technology Bennett University, Greater Noida, India Date:

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the Ph.D. thesis entitled "**Prediction and Trend Analysis of Financial Time Series: Deep Learning and Kernel Adaptive Filtering Perspective**" submitted by **Shambhavi Mishra** at **Bennett University**, **Greater Noida, India** is a bonafide record of his original work carried out under my supervision. This work has not been submitted elsewhere for any other degree or diploma.

Dr. Tanveer Ahmed

Assistant Professor School of Comp. Sc. Engg. and Tech. Bennett University, Greater Noida, India Date:

Dr. Vipul Kumar Mishra

Associate Professor School of Comp. Sc. Engg. and Tech. Bennett University, Greater Noida, India Date:

ACKNOWLEDGEMENTS

I am very grateful to my supervisors **Dr. Tanveer Ahmed**, Assistant Professor, School of Computer Science Engineering and Technology, Bennett University and **Dr. Vipul Kumar Mishra**, Associate Professor, School of Computer Science Engineering and Technology, Bennett University, for their continuous encouragement, support, guidance, and helpful recommendations during this research work. This research could not have been done without their constant guidance and suggestions. Their constant support and encouragement have inspired me to stay focused on my research. I am grateful to **Prof. Deepak Garg**, RAC Chairman and Head of the School of Computer Science Engineering and Technology, Bennett University, for his valuable support and suggestions during this research.

I would like to express my heartfelt gratitude to my RAC members **Dr. Manoj Sharma** and **Dr. Divya Srivastava**, **Dr. Rishav Singh**, and **Dr. Urvashi Arora** for their valuable and useful suggestions. I am incredibly grateful to **Dr. Prabhu Aggarwal** (Honorable Vice-chancellor, Bennett University) and **Dr. R. K. Shevgaonkar** (Ex Vice-Chancellor, Bennett University) for creating an excellent academic and research environment at Bennett University. I would like to express my gratitude to my family members, who have always stood with me through the highs and lows of the research process. I wish to thank all my colleagues, faculties, and staff members of the CSE department for their suggestions and help. I would like to thank **Dr. Pawan Singh**, who inspired me to pursue a career in research. Special thanks to **Dr. Tanveer Ahmed** without him, this journey would never have been possible for me. Finally, I would like to express my gratitude to everyone who has contributed to and supported me throughout my career as a research scholar.

Contents

A	ABSTRACT i						
LI	LIST OF PUBLICATIONS iii						
LI	LIST OF TABLES v						
LI	ST O	F FIGU	URES	ix			
LI	ST O	F ACR	ONYMS & ABBREVIATIONS	xi			
1	INT	RODU	CTION	1			
	1.1	PREA	MBLE	2			
	1.2	BACK	GROUND	4			
		1.2.1	DEEP LEARNING (DL)	4			
		1.2.2	KERNEL ADAPTIVE FILTERING (KAF)	6			
		1.2.3	FEATURE ENGINEERING (FE)	8			
	1.3	RESE	ARCH PROBLEM	10			
	1.4	RESE	ARCH OBJECTIVES	10			
	1.5	THES	IS OUTLINE	11			
2	REV	IEW C	DF LITERATURE	13			
	2.1	OVER	VIEW OF THE STOCK PRICE PREDICTION	14			
	2.2	TRAD	ITIONAL ML-BASED METHODS AND DATA SOU- RCES FOR STOC	K			
		PRICE	PREDICTION	16			
		2.2.1	DATA SOURCES	20			
	2.3	DL-BA	ASED METHODS FOR STOCK PREDICTION	24			
		2.3.1	DATA SOURCES	29			

	2.4	ONLIN	IE LEARN	NING METHODS FOR STOCK PREDICTION	30
3	SEC	TOR IN	IFLUENC	CE AWARE STOCK TREND PREDICTION USING 3D CON	-
	VOI	OLUTIONAL NEURAL NETWORK			
	3.1	INTRO	DUCTIO	Ν	34
	3.2	PROPO	DSED ME	THODOLOGY	36
		3.2.1	MODEL	ARCHITECTURE	36
			3.2.1.1	3D CONVOLUTIONAL NEURAL NETWORK	37
			3.2.1.2	ENSEMBLED CNN	37
		3.2.2	FEATUR	E SELECTION AND IMAGE GENERATION	38
			3.2.2.1	FEATURE SELECTION USING HIERARCHICAL CLUS-	
				TERING	39
			3.2.2.2	CONVERTING A NON-STATIONARY TIME SERIES INTO	
				IMAGES	40
		3.2.3	LABELI	NG OF STOCK'S CLOSE-PRICE	43
	3.3	RESUI	LTS		46
		3.3.1	DATASE	T AND EVALUATION CRITERION	46
		3.3.2	IMPORT	ANCE OF UNDERSAMPLING	48
		3.3.3	EXPERI	MENTATION WITH DIFFERENT ACTIVATION FUNCTIONS	48
		3.3.4	COMPAI	RISON WITH SIMILAR METHODS	49
		3.3.5	CAPTUR	RING THE INFLUENCE OF ONE COMPANY ON ANOTHER	51
	3.4	DISCU	SSION .		55
	3.5	CONC	LUSIONS	•••••••••••••••••••••••••••••••••••••••	57
4	MU	LTIVAR	RIATE AN	D ONLINE PREDICTION USING KERNEL ADAPTIVE	
	FIL	FERIN	Ĵ		59
	4.1	INTRO	DUCTIO	Ν	60
	4.2	METH	ODOLOG	ΥΥ	61
		4.2.1	DISCUS	SION ON KAF ALGORITHMS	62
			4.2.1.1	LEAST MEAN SQUARE ALGORITHM (LMS)	62
			4.2.1.2	KERNEL LEAST MEAN SQUARE ALGORITHM (KLMS)	63
			4.2.1.3	KERNEL RECURSIVE LEAST SQUARE ALGORITHM	
				(KRLS)	64

		4.2.1.4	KERNEL AFFINE PROJECTION ALGORITHMS (KAPA)	65
		4.2.1.5	QUANTIZED KERNEL LEAST MEAN SQUARE ALGO-	
			RITHM (QKLMS)	66
		4.2.1.6	KERNEL NORMALIZED LEAST MEAN SQUARE AL-	
			GORITHM (KNLMS)	67
		4.2.1.7	PROBABILISTIC LEAST MEAN SQUARE ALGORITHM	
			(PROB-LMS)	69
		4.2.1.8	KERNEL MAXIMUM CROSSENTROPY CRITERION (KMC	C) 69
		4.2.1.9	LEAKY KERNEL AFFINE PROJECTION ALGORITHM	
			(LKAPA)	70
		4.2.1.10	NORMALIZED ONLINE REGULARIZED RISK MINIMIZA	-
			TION ALGORITHM (NORMA)	72
	4.2.2	PROBLE	EM FORMULATION	72
4.3	RESU	LTS		74
	4.3.1	DATASE	TS	75
	4.3.2	EVALUA	ATION CRITERION	76
4.4	EXPE	RIMENTA	L RESULTS ON CLOSE PRICE USING KERNEL ADAP-	
	TIVE	FILTERIN	G	77
	4.4.1	PROCED	OURE: ERROR COMPUTATION	77
	4.4.2	PREDIC	TION, CONVERGENCE, AND RESIDUAL ANALYSIS	78
	4.4.3	COMPR	EHENSIVE EVALUATION OF KAF ALGORITHMS	80
	4.4.4	COMPA	RISON WITH OTHER STATE-OF-THE-ART METHODS	82
	4.4.5	EXPERI	MENTATION WITH DICTIONARY SIZE	83
	4.4.6	IMPORT	ANT NOTE: ERROR MINIMIZATION AND PROFITABIL-	
		ITY		83
4.5	EXPE	RIMENTA	L RESULTS ON MID PRICE USING KERNEL ADAPTIVE	
	FILTE	RING		87
	4.5.1	CALCUI	LATING THE EVALUATION METRICS WITH NIFTY-50 .	87
	4.5.2	PREDIC	TION, CONVERGENCE, AND RESIDUAL ANALYSIS	87
	4.5.3	COMPA	RATIVE EVALUATION OF KAF ALGORITHMS	89
	4.5.4	COMPA	RISON WITH METHODS OF A SIMILAR KIND	90
	4.5.5	EFFECT	OF DIFFERENT KERNELS	91

		4.5.6	EXPERIMENTATION WITH DICTIONARY SIZE	92
		4.5.7	IMPORTANT NOTE: PROFITABILITY AND ERROR MINIMIZA-	
			TION	92
	4.6	CONC	LUSIONS	94
5	STA	TISTIC	CAL FEATURE EXPANSION DRIVEN KERNEL ADAPTIVE FIL-	-
	TER	RING FO	OR MID-PRICE PREDICTION	97
	5.1	INTRO	DDUCTION	98
	5.2	METH	IODOLOGY	99
		5.2.1	KERNEL ADAPTIVE FILTERING TECHNIQUES AND THEIR METH	[-
			ODS	99
		5.2.2	FEATURE ENGINEERING	100
		5.2.3	REGRESSION PROBLEM FORMULATION	103
	5.3	RESU	LTS	105
		5.3.1	DATASET COLLECTION	105
		5.3.2	EVALUATION MERTICS	106
		5.3.3	PREDICTION AND CONVERGENCE ANALYSIS	107
		5.3.4	COMPREHENSIVE EVALUATION OF ALGORITHMS	109
		5.3.5	COMPARISON WITH DIFFERENT STATE-OF-THE-ART METHODS	5110
		5.3.6	EFFECT OF DIMENSIONALITY REDUCTION METHODS	110
		5.3.7	IMPORTANT NOTE: ERROR MINIMIZATION AND PROFITABIL-	
			ITY	112
	5.4	CONC	LUSIONS	113
6	CON	NCLUS	IONS AND FUTURE RESEARCH	115
	6.1	CONC	LUSIONS	116
	6.2	FUTU	RE RESEARCH	117
Rŀ	EFER	ENCES		119

ABSTRACT

Stock price prediction is one of the important applications of time series analysis. Its success stems from its ability to reduce asset management costs, market impacts, and volatility risks. Stock price prediction is a significant challenge, and a plethora of techniques have been investigated. Nevertheless, a true solution is yet to be found. Despite a lot of effort, research has demonstrated that forecasting a stock's price is difficult, especially when considering the nonlinear and non-stationary behavior of financial time series. The prediction of stock prices is traditionally done with regression and classification, thereby requiring a large set of batch-oriented and independent training samples. The work presented in this thesis takes on this challenge and present an automated framework to address two important issues in the field: i) Stock trend analysis and ii) Stock price prediction. For the former issue, we propose a 3D convolutional neural network based approach to classify the directional trends in a stock's price. In contrast to existing literature, where work emphasizes upon predicting the direction in a single stock, we focus on a particular sector. This is done to analyze and capture the influence of one company on another. Further, multiple technical indicators are chosen, and the stock prices are converted into a 3D image. To find the best features, we experiment with hierarchical clustering. Lastly, we also complement the 3D convolutional neural network via the application of 3D ensemble learning. With extensive numerical investigation performed on forty-five different stocks, we found that the proposed work achieved up to 35% returns in some cases, with the average being 9.19%.

To tackle the latter issue (stock price prediction), we focus our efforts on Kernel Adaptive Filtering. During experimentation on the previous problem, we discovered that deep learning based techniques require a lot of computational resources during training. Moreover, relying on offline-trained models and expecting them to perform well in real-world trading like circumstances is a strong assumption. Therefore, to take on the second challenge, we propose the idea of online Kernel Adaptive Filtering based approach to predict a stock's prices. Specifically, we experiment with ten different kernel adaptive filtering algorithms to analyze a stocks' performance and show the efficacy of the proposed work. The experiments are performed with quotes gathered at the window of one minute, five minutes, ten minutes, fifteen minutes, twenty minutes, thirty minutes, one hour, and one day. These time windows represent some of the standard windows used by high-frequency traders. The proposed framework was tested on fifty different stocks of the Indian stock index: Nifty-50. In terms of performance and compared to existing methods, the proposed method achieved a 66% probability of correctly predicting a stock's next up or a down movement. To the best of our knowledge, the work presented in this thesis is the first wherein we analyze the price of a stock at these time windows. Furthermore, this article is the first to test the application of the KAF class of algorithms on all fifty stocks of the Nifty-50. Lastly, we also complement a stock' price with additional statistical features to make the predictions more accurate. The experimental findings show that kernel adaptive filtering is a better option for predicting stock prices and may also be used as an alternative in high-frequency trading.

LIST OF PUBLICATIONS

INTERNATIONAL JOURNAL

- Sinha, S., Mishra, S., Mishra, V., & Ahmed, T. (2022). Sector influence aware stock trend prediction using 3D convolutional neural network. *Journal of King Saud University-Computer and Information Sciences*. [SCI, IF:13.47], doi: https://doi.org/10.1016/j.jksuci. 2022.02.008.
- Mishra, S., Ahmed, T., Mishra, V., Kaur, M., Martinetz, T., Jain, A. K., & Alshazly, H. (2021). Multivariate and Online Prediction of Closing Price Using Kernel Adaptive Filtering. *Computational Intelligence and Neuroscience*. [SCI, IF:3.63], doi: https://doi.org/10.1 155/2021/6400045.
- Mishra, S., Ahmed, T., Mishra, V., Bourouis, S., & Ullah, M. A. (2022). An Online Kernel Adaptive Filtering-Based Approach for Mid-Price Prediction. *Scientific Programming*. [SCI, IF:1.02], doi: https://doi.org/10.1155/2022/3798734.
- 4. Mishra, S., Ahmed, T., Mishra, V.,(2022). Multivariate and Online Prediction of the Mid Price for Indian NIFTY-50 Stocks based on Feature engineering and KAF based techniques. Advances in Data Analysis and Classification Theory, Methods, and Applications in Data Science. [Submitted]

INTERNATIONAL CONFERENCE

- Mishra, S., Ahmed, T., & Mishra, V. K. (2021). Close-price prediction using online kernel adaptive filtering. Thirteenth International Conference on Contemporary Computing (IC3-2021) (pp. 217-222).
- 2. Mishra, S., Ahmed, T., & Mishra, V. (2022). Mid-price Prediction using Online Kernel

Adaptive Filtering. Emerging Technologies for computing, communication and smart cities (ETCCS-2021) (pp. 701-714). Springer, Singapore.

 Mishra, S., Ahmed, T., & Mishra, V. (2021). Mid Price Prediction: Feature Engineering and kernel adaptive filtering (KAF) perspective. Fourteenth International Conference on Contemporary Computing (IC3-2022). [Accepted]

OTHER PUBLICATION

1. Mishra, S., Ahmed, T., & Mishra, V. K. (2022). Reliable Biomedical Applications Using AI Models. *Healthcare Solutions Using Machine Learning and Informatics*.

LIST OF TABLES

2.1	Analysis of different methods on ML. Continued on next page in Table 2.2	21
2.2	Continuation from Table 2.1: Analysis of different methods on ML. Continued	
	on next page in Table 2.3.	22
2.3	Continuation from Table 2.2: Analysis of different methods on ML	23
3.1	Results after undersampling by various amounts.	47
3.2	Results after making changes to model.	49
3.3	Comparison of Annual Return with the other approaches. Continued on next	
	page in Table 3.4	52
3.4	Continuation from Table 3.3: Comparison of Annual Return with the other ap-	
	proaches. Continued on next page in Table 3.5	53
3.5	Continuation from Table 3.4: Comparison of Annual Return with the other ap-	
	proaches.	54
3.6	Results of same sector Vs different sector.	55
4.1	Time window of 1-day (Stock-Reliance). If we choose M=3, then Input=[{0.1685,-	
	1.2431,-2.6372}] Output=[{-0.16194}]	74
4.2	A one-day time frame (Stock-TITAN). If M=3 is selected, then Input=[{-0.7654,	
	0.4300, 2.0796}.] Output=[{-2.4566}]	74
4.3	Parameter description for close price using ten different KAF algorithms. σ =	
	Kernel width , $\sigma_2 n$ = Variance of observation noise , $\sigma_2 d$ = Variance of filter	
	weight diffusion, η = Step-size, ε = Regularization parameter, Λ = Tikhonov reg-	
	ularization , $tcoff$ = Learning rate coefficient, τ = memory size (terms retained	
	in truncation), mu0= Coherence criterion threshold, P= Memory length, nu=	
	Approximate linear dependency (ALD) threshold	78

4.4	Comparison of the proposed work with other state-of-the-art stock prediction	
	method for 60-minute time window (2020 dataset) January 01, 2020 to Decem-	
	ber 31, 2020	82
4.5	Comparison of the proposed work with other state-of-the-art stock prediction	
	method for 60-minute time window (2021 dataset) January 01, 2021 to May 31,	
	2021	83
4.6	Effect of Dictionary size. Algorithm chosen KMCC (60 minutes, 2021 dataset).	83
4.7	Best Embedding Dimension for all time windows according to evaluation met-	
	rics MSE, MAE and DS. (2020)	85
4.8	Best Embedding Dimension for all time windows according to evaluation met-	
	rics MSE, MAE and DS. (2021)	86
4.9	Parameter description of KAF techniques for NSE-50 data set for mid-price	88
4.10	The proposed research is compared to different state-of-the-art stock prediction	
	approaches	91
4.11	Effect of different kernel methods on time window of thirty-minutes (stock-	
	TITAN) using KAPA for (high+low)/2	91
4.12	The Influence of Dictionary Size. KMCC was chosen as the algorithm (30-	
	minutes).	92
4.13	Result in terms of MSE, MAE and DS for mid price (high+low)/2	93
4.14	Result in terms of MSE, MAE and DS for mid price (open+close)/2	94
5.1	KAF algorithms.	101
5.2	If we choose Embedding Dimension <i>M</i> =10, then Input=[{Input Feature 1,	
	,Input Feature 58}]. Output=[{Output}]	102
5.3	If we choose Embedding Dimension <i>M</i> =10, then Input=[{Input Feature 1,	
	,Input Feature 58}]. Output=[{Output}]	103
5.4	Define the evaluation criterion	106
5.5	Parameter description of KAF techniques for NSE-50 data set for mid-price	107
5.6	Comparison of the proposed approach with other methods for one-day time	
	window (High+Low)/2	109
5.7	Comparison of the proposed approach with other methods for one-day time	
	window (Open+Close)/2	110
5.8	Effect of dimensionality reduction using LKAPA for (Open+Close)/2 and (High+I	Low)/2.111

- 5.9 Evaluation results in terms of evaluation criterion for mid price (High+Low)/2. 111
- 5.10 Evaluation results in terms of evaluation criterion for mid price (Open+Close)/2. 112

LIST OF FIGURES

3.1	Proposed framework for Stock Trend Classification.	36
3.2	Architecture of the proposed 3D CNN model.	38
3.3	Hierarchical Clustering of Technical Features.	39
3.4	Clustering using K-means with K=7	40
3.5	Image creation and labeling of stock's price.	41
3.6	One 3D image split into its components. Raw image (Gray Scale, top row),	
	Smoothened Image (Gray Scale, middle row), Smoothened Image (Colored,	
	bottom row)	42
3.7	Comparison of different methods. The numbers here represent on how many	
	stocks a method performed best in terms of annual returns. For instance, the	
	proposed 3D CNN performed best in twelve stocks out of forty-five	50
3.8	Average sector-wise annual returns.	51
4.1	Proposed close price prediction framework.	73
4.2	Proposed mid price prediction framework	73
4.3	Prediction for one stock (Reliance) using KRLS: (a) 2020 dataset and (b) 2021	
	dataset	79
4.4	Error convergence for one stock (Reliance) using KRLS: (a) 2020 dataset and	
	(b) 2021 dataset	80
4.5	Error residuals for one stock (Reliance) using KRLS: (a) 2020 dataset and (b)	
	2021 dataset	81
4.6	Prediction for one stock (TITAN) using KAPA (high+low)/2	88
4.7	Prediction for one stock (TITAN) using KAPA (open+close)/2	88
4.8	Error convergence for one stock (TITAN) using KAPA (high+low)/2	90
4.9	Error convergence for one stock (TITAN) using KAPA (open+close)/2	90

4.10	Error residuals one stock (TITAN) using KAPA (high+low)/2	90
4.11	Error residuals for one stock (TITAN) using KAPA (open+close)/2	90
5.1	Architecture of the proposed work	104
5.2	Prediction Graphs of 2021 dataset with (High+Low)/2	108
5.3	Prediction Graphs of 2021 dataset with (Open+Close)/2	108
5.4	Convergence Graphs of 2021 dataset with (High+Low)/2	108
5.5	Convergence Graphs of 2021 dataset with (Open+Close)/2	108

LIST OF ACRONYMS & ABBREVIATIONS

ML	Machine Learning
NN	Neural network
SVM	Support vector machines
DL	Deep Learning
CNN	Convolutional neural networks
LSTM	Long short-term memory
GRU	Gated recurrent units
KAF	Kernel adaptive filtering
ANN	Artificial neural networks
DBN	Deep belief network
SAE	Sparse Autoencoder
BP	Backpropagation
RNN	Recurrent neural network
HAN	Hybrid attention networks
NLP	Natural language processing
MFNN	Multi-filters neural network
КРСА	Kernel principal component analysis
KFDA	Kernel Fisher discriminant analysis
RKHS	Reproducing Kernel Hilbert Spaces

KLMS	Kernel least mean square
КАРА	Kernel affine projection algorithms
KRLS	kernel recursive least squares
EX-KRLS	Extended kernel recursive least squares
RBF	Radial basis function
ALD	Approximate linear dependency
QKLMS	Quantized kernel least mean square
CV	Cross-validation
AI	Artificial intelligence
EMH	Efficient market hypothesis
AMH	Adaptive markets hypothesis
ARCH	Auto-regressive conditional heteroscedasticity
ARMA	auto-regressive moving average
ARIMA	Autoregressive integrated moving average
RF	Random Forest
KNN	k-nearest neighbour
KOSPI	Korea Composite Stock Price Index
SZSE	Shenzhen Stock Exchange
RBNN	Resilient back-propagation neural networks
MAPE	Mean absolute percent error
PCA	Principal component analysis
GA	Genetic algorithms
SSFS	Supported sequential forward search
TEJ	Taiwan Economic Journal
CART	classification and regression trees
GPUs	Graphics processing units

WDPB	Wavelet denoising-based backpropagation
DRL	Deep reinforcement learning
RRL	Recurrent reinforcement learning
POMDP	Partially Observed Markov Decision Process
NSE	National Stock Exchange
СРІ	Consumer Price Index
GDP	Gross Domestic Product
LMS	Least-mean square
M-QKLMS	Modified quantized kernel least mean square
ReLU	Rectified linear unit
СМО	Chande Momentum Oscillator
RSI	Relative Strength Index (RSI)
CCI	Channel Commodity Index
WILLR	Williams' %R
BAH	Buy and Hold
SGD	Stochastic gradient descent
KNLMS	Kernel Normalized Least Mean Square Algorithm
MKNLMS-CS	Multi kernel normalized least mean square algorithm with coherence based
	sparsification
КМСС	Kernel Maximum Crossentropy Criterion
LKAPA	Leaky Kernel Affine Projection Algorithm
LKAPA NORMA	Leaky Kernel Affine Projection Algorithm Normalized Online Regularized Risk Minimization Algorithm
NORMA	Normalized Online Regularized Risk Minimization Algorithm
NORMA AR	Normalized Online Regularized Risk Minimization Algorithm Autoregressive

MAE	Mean absolute error
DS	Directional symmetry
RS	Rough Set
WNN	Wavelet Neural Network
PFD	Petrosian Fractal Dimension
HFD	Higuchi Fractal Dimension
SVD	Singular Value Decomposition Entropy
FBQKLMS	Fixed budget quantized kernel least-mean-square

CHAPTER 1

INTRODUCTION

1.1 PREAMBLE

A time series is a set of well-defined data items collected at successive points at uniform time intervals. It has applications in a multitude of areas, such as economics, business planning, production, and weather forecasting, to name a few. The impact of time series analysis on scientific applications can be partially demonstrated by compiling a list of various disciplines in which significant time series difficulties can arise. Various well-known time series may be found in economics and were constantly exposed to stock price and trend forecasting. In the context of stock prices, it is understandable that an accurate prediction (of a stock's prices) can yield a significant profit, whereas a bad forecast could be devastating. However, it has been well established that stock price prediction is one of the non-trivial problems of literature. This is owing to the fact that the movement in price is often chaotic (1), complex (2), volatile (3), and a dynamic mix of political, even sometimes environmental factors, e.g., The crash of markets in 2020 owing to the COVID-19 pandemic comes into picture (3). However, accurately predicting the value of a variable is one of the very basic and non-trivial problems of literature. Moreover, the Efficient Market Hypothesis (4) states that stock prices reflect all current information, and any new information leads to unpredictability in stock prices. Naturally, significant work has been done in this area; however, despite a huge amount of work trying to address the issue, research has shown that estimating a stock's price, especially considering the non-linear and the non-stationary behavior of the financial time series, is challenging (5). Predicting a stock's value at any given time in the future is an important problem for academia and industry. The process of analyzing and developing future stock price movements based on previously observed trend data is known as trend analysis and price prediction. Identifying the source, tracing the evolution, and finding any trend is difficult. In context of stock price prediction, several models have been developed, for instance studies focusing on volatility (3), option pricing (6), classifying stock movements (7), predicting prices (8). In recent years, machine learning (ML) has also emerged as a popular method for stock price prediction. Previous Studies have attempted to understand the movement of stocks, analyzing trends, and clustering of financial information (9). Further, a plethora of techniques have been used for predicting stock prices, such as neural networks (NN), support vector machines (SVM), genetic algorithms, fuzzy logic, Bayesian model (10), and so on. These days, deep learning (DL) methods have gained tremendous popularity. This is mainly owing to the advancements in computational power and the increase in the amount of data available to train a model. In the context of analyzing stocks, multiple approaches have been proposed, such as convolutional neural networks (CNN), long short-term memory (LSTM), and gated recurrent units (GRU). Nevertheless, a true solution is yet to be found.

Moreover, during the literature survey, it was found that there is a need for better stock price prediction methods that can accurately predict the prices with less execution time for intraday trading. In this regard, our research focuses on trends and price predictions. We also focus on understanding and analyzing the directional trends in a stock's price. In particular, through this research, first, we make an attempt to classify the trend into three different categories, namely, Buy, Sell, and Hold. The main objective is to increase the annual return on investment while simultaneously capturing the influence of one company on another from the same sector. To that end, we have used 3D CNN. In recent years, CNNs have shown excellent breakthroughs in image classification (11). Hence, following this precedent, we also try to apply the same ideas in the financial markets and analyze the trends using 3D CNN. In simple words, we try to classify the up or down movements in a stock's price.

Stock price prediction is the next important aspect of our research. The main purpose is to determine the future values of stock depending on historical values. Work has emphasized that stock price prediction is difficult due to data nonstationarity in the underlying financial time series. Moreover, the governing dynamics of financial markets change very quickly. In such scenarios, it is unlikely that batch learning would produce good results. We would argue that batch learning is ineffective in financial time series forecasting. The rationale here is backed by the fact that the data of a financial time series is non-stationary. Therefore, relying on a model trained in an offline manner and expecting them to perform well in real market scenarios is a rather strong assumption. To fix this, online learning is proving to be a highly efficient approach (12), (13), (14), (15). In this method, the basis is selected during sample-by-sample training. Moreover, changing circumstances are quickly incorporated, and the algorithm changes its weight vector to make accurate predictions.

With this argument in mind, we would like to emphasize that kernel adaptive filtering (KAF) can be an effective stock price prediction tool. The following observations serve as the foundation for our argument: First, KAF-based algorithms have a faster convergence rate, i.e., the algorithm requires fewer iterations. Second, KAF has demonstrated excellent performance

in non-stationary time-series prediction (16). Third, KAF algorithms exhibit universal function approximation properties that are useful in highly dynamic environments (17). Owing to these reasons, we focus on predicting the financial time series via the kernel adaptive filtering paradigm. Hence, we complement the idea of using KAF with online learning to predict a financial time series.

Lastly, to improve the stock price prediction performance, we also work on the idea of feature engineering. We work with the idea of feature augmentation, wherein, we construct several statistical features to make the prediction more accurate. We provide a comprehensive set of variables for the mid-price prediction challenge representing the underlying fundamentals' statistical qualities. For each stock, the results imply that selecting the proper feature set can help anticipate how long it will take for a stock to rise or fall. We augment the basic financial time series with various statistical embeddings and try to predict the future mid-price. The empirical results validate the superiority of the proposed approach over other existing state-of-the-art methods.

It should be noted here that the practical viability of the methods is validated via experimentation performed on a real-world Nifty 50 datasets. The results show good predictive capability of the work and its ability to be deployed in a high-frequency trading environment.

1.2 BACKGROUND

This section details the necessary background knowledge to better understand the thesis. Specifically, this section includes the detail of deep learning, kernel adaptive filtering, and feature engineering concepts.

1.2.1 DEEP LEARNING (DL)

Deep learning is a subset of machine learning based on artificial neural networks (ANNs). A deep neural network (DNN) is a neural network with multiple hidden layers. DNN can represent complex nonlinear functions and has a high degree of abstraction, which greatly increases

the model's convergence capability. It is a particular class of discriminant model that the backpropagation algorithm can be used to train (18). For example, an intelligent stock and Forex prediction system based on a DNN can be designed to predict stock and Forex trends. This is because a DNN is adept at dealing with prediction problems involving vast volumes of data and sophisticated nonlinear mapping interactions. (19). In recent years, work has noticed a considerable improvement in the performance of DNNs. As a result, deep learning methods have attracted significant attention in various domains. For instance, in stock price prediction, deep learning has been widely applied and has already shown superior performance over traditional statistical and machine learning methods (20), (21). Further, CNN is one of the popular variants of the DNN that has been applied in various applications (22). Because of its automatic feature extraction ability, CNN was widely employed in the field of image recognition. Naturally, its application was also extended to the field of time series prediction (23). CNN differs from a fully connected network's network structure, such as a deep belief network (DBN) (24), Sparse Autoencoder (SAE) (25), or backpropagation (BP) (26), in that the CNN can share the weight across the neurons in each layer of the network. Moreover, 2D CNNs are good for processing spatial data, i.e., 2D images. However, 3D CNN are good for sequential data and videos. As a result, the model adequately avoids local minimization and dimensional disaster while also drastically reducing the weight of the network.

Another popular DNN variant is the recurrent neural network (RNN) (27), it can handle the dependencies in sequential data. RNN can memorize the prior state, which may be employed in the current state computation. The input of the current hidden layer comprises not only the output of the input layer but also the output of the previously hidden layer. As a result, while dealing with sequential data, RNN performs well (28). In contrast to CNNs, which are aimed at dealing with images, RNNs are trained to handle sequence data. However, traditional RNNs have a vanishing gradient problem and are incapable of maintaining long-term dependencies. Other prominent RNN variants that address the constraints of regular RNNs includes long short-term memory (LSTM) (29) and gated recurrent units (GRU). Its main contribution is the introduction of the self-loop design to generate the path of a gradient that may flow continuously for a longer period of time. Each iteration additionally updates the weight of the self-loop, preventing the gradient vanishing problem that occurs when the RNN model updates the weights. An LSTM layer is made up of concurrent units, each of which is divided into four components: an input gate *i*, a cell *c* that serves as a temporary memory for input values, an output gate *o*, and a forget gate f that controls the flow of data into and out of the cell (30). Other deep learning approaches, such as hybrid attention networks (HAN) (31), natural language processing (NLP) (32), multi-filters neural network (MFNN) (33) have been utilized to anticipate stock and Forex movements. Multiple ways of assessing stocks have been presented, and various deep learning methods have been applied to address the challenge.

1.2.2 KERNEL ADAPTIVE FILTERING (KAF)

Kernel-based approaches are widely employed in nonlinear signal processing and machine learning due to their inherent benefits of convex optimization and universality in the space of L2 functions. By mapping the input data into a feature space associated with a Mercer kernel, many successful nonlinear algorithms may be built. Popular kernel approaches include kernel regularisation network (34), support vector machine (SVM) (35), (36), kernel fisher discriminant analysis (KFDA) (37), kernel principal component analysis (KPCA) (38), among others. These nonlinear algorithms outperform their linear counterparts significantly. Online kernel learning (39), (40), (41), (42) has also received substantial attention in statistical signal processing and machine learning literature. It offers effective options for progressively approximating a desired nonlinearity. Because the training data is supplied to the learning system in a sequential manner, online learning requires significantly less memory and computing cost. Kernel-based online adaptive filtering algorithms have recently been created and have emerged as an emerging topic of study (43). Kernel adaptive filters (KAF) are developed in Reproducing Kernel Hilbert Spaces (RKHS) (44), (45) by using well-established linear adaptive filtering algorithms that correspond to nonlinear filters in the original input space utilizing the linear structure and inner product of this space. Kernel affine projection algorithms (KAPA) (46), Kernel least mean square (KLMS) (47), (48), kernel recursive least squares (KRLS) (49), and extended kernel recursive least squares (EX-KRLS) (50) are examples of common KAF algorithms. They develop a growing radial basis function (RBF) network with a radially symmetric Gaussian kernel to learn the network topology and adapt free parameters directly from the training data. Among these KAF algorithms, the KLMS is the simplest and fastest to develop, yet it is also the most effective.

In the KAF algorithms, there are two major open challenges. The first is that its structure

grows with each sample, resulting in higher computing costs and memory needs, particularly in continuous adaption settings. A number of sparsification strategies have been used to limit network growth and achieve a compact representation, with only the most significant input data accepted as new centers. The novelty criterion (51), approximate linear dependency (ALD) criterion (50), surprise criterion (52), and others are currently accessible sparsification criteria. In a recent research (53), the authors presented a unique approach, the quantized kernel least mean square (QKLMS) algorithm, for compressing the input space and thereby constraining network size.

The second remaining issue to overcome when building kernel adaptive filtering methods is selecting a proper Mercer kernel, especially when the training data amount is minimal. In this situation, kernel selection consists of two steps: first, the kernel type is chosen, then its parameters are determined. Because of its universal approximation capabilities, desired smoothness, and mathematical stability, the Gaussian kernel is a common option in kernel adaptive filtering. The normalized Gaussian kernel is defined as:-

$$\kappa < v, v' > = exp \frac{(||v - v'||^2)}{2\sigma^2}.$$
 (1.1)

In the above Equation, where the free parameter σ (σ >0) is referred to as the kernel size (also known as the kernel bandwidth or smoothing parameter). In reality, because the Gaussian kernel is strictly positive definite, it yields a dense RKHS (45), and linear algorithms in this RKHS are universal approximators of smooth L2 functions. In theory, this indicates that the asymptotic features of the mean square approximation are independent of kernel size in the large sample size domain (54). This means that the kernel size in KAF only influences the dynamics of learning because the sample size is always small in the initial steps; hence, the kernel size affects both the accuracy for batch learning and the convergence qualities for online learning. This is in contrast to the influence of kernel size on classification, where the kernel size determines the accuracy and generalization of the best solution (35), (36). Several approaches for choosing a kernel size for the Gaussian kernel have been adopted, from statistics, nonparametric regression, and kernel density estimation. Cross-validation (CV) (55), (56), (57), (58), which can always be utilised since the kernel size is a free parameter, penalising functions

(56), plug-in approaches (56), (59) Silverman'srule (60), and other rules of thumb (61) are the most frequent methods for selecting the kernel size.

According to the arguments presented in the previous paragraph, Kernel adaptive filtering (KAF) can be an effective stock prediction tool. The following observations serve as the foundation for our argument: First, KAF-based algorithms have a faster convergence rate, i.e., the algorithm requires fewer iterations. Second, KAF has demonstrated excellent performance in non-stationary time-series prediction (16). Third, KAF algorithms exhibit universal function approximation properties useful in highly dynamic environments (17). Lastly, KAF has been used extensively in chaotic time-series prediction (52), (53). Hence, it is also worth exploring the idea of financial time-series prediction. Existing literature focuses on multiple kernel learning methods and solves different issues such as kernel size and step size. During our literature survey, we found that the paradigm of KAF is not thoroughly investigated. Although there are a few papers on the topic, e.g., (62), (63), a comprehensive investigation conducted at a large scale eludes literature. We have therefore taken on this shortcoming and applied the paradigm of KAF to stock price prediction for intraday trading. The experimental results show that online learning and KAF is not only good option, but practically speaking, they can also be deployed in high-frequency trading.

1.2.3 FEATURE ENGINEERING (FE)

Feature expansion is a technique for identifying important data characteristics that a model can employ to boost its performance. Optimum feature selection is crucial to improving the outcomes. Recent research has shown that feature expansion has emerged as another alternative to improve the predictive performance of stock price prediction methods. It has been applied in various field such as neurocomputing (64), EEG signals (65), wireless intrusion detection system (66) and time series analysis (67). Further, we anticipate that existing and well-tested prediction models will benefit by extracting new characteristics from data and merging them with different domain variables. Expanding features is crucial when working with online learning models. Regardless of the data or architecture, a poor feature will immediately impact your model. Further, the ability of artificial intelligence (AI) and machine learning methods to extract the subtle patterns in stock prices has made them a preferred choice for stock price prediction

over the years (68). One of the research author (69) proposed a hybrid GA-XGBoost prediction system with improved feature engineering that includes feature set expansion, data preparation, and the best feature set selection utilizing the hybrid GA-XGBoost algorithm. Further, this study also empirically showed that, in order to achieve balance and harmony between the bless-ing of dimensionality and the curse of dimensionality, a combination of feature engineering approaches with a baseline learning model is crucial for good prediction performance.

However, the machine learning methods' performance remains unsatisfactory and needs significant improvement. In this regard, with the advent of deep learning methods, stock price prediction has witnessed tremendous success over traditional ML methods in recent years. The use of deep learning methods such as LSTM, convolutional neural networks (CNN), and recurrent neural networks (RNN) is prevalent in stock price prediction. An LSTM-based method for stock market prediction is proposed in research (70). It is evident that deep learning methods have significantly improved stock price prediction. On the contrary, these methods require an adequate amount of computational resources. Another downside is that they are trained using batch learning, making them less suitable for high-frequency real-time trading. This has given opportunities for online methods. The time-varying and nonlinear nature of the online KAF-based learning methods have made them popular for stock price prediction in recent years (71). KAF is a better choice because of its low computing complexity, fast convergence, and non-parametric behavior. We look at previous research, identify limitations, and present a solution architecture for prediction models with a comprehensive feature expansion approach. With the effectiveness of the feature expansion strategy in combination with online KAF algorithms, various different learning algorithms can now obtain better accuracy scores for predicting shortterm price trends. Instead of analyzing technical indicators, which is the most preferred way of literature trying to predict stock prices so far (72), (68), we argue that feature extension as well as fine-tuning the selected features based on the statistical properties of the financial time series could work well in practice. We look at previous research, identify limitations, and present a potential architecture via feature expansion in our research.

1.3 RESEARCH PROBLEM

Numerous studies have attempted to explain that stock price prediction is challenging to implement because of the inherent non-stationary data. In financial research, predicting the direction of the market has become critical, complex, and contentious. Various models and methods have been developed to predict the trend and the price of a stock. Nevertheless, a true solution is yet to be found. Therefore, we want to address this issue and develop novel techniques for trend analysis that provides higher annual return. At much the same time, we want to predict a stock's price that minimizes the error. Therefore, this research aims to answer the following questions:

- 1. Can we develop new trend analysis techniques that gives higher annual returns? Moreover, can we redefine the existing notion of predicting the trend in a financial time series?
- 2. In contrast to existing literature, can we engineer a method that looks at a sector as a whole and predict the trends in individual stocks?
- 3. Can we develop a novel online method to predict stock prices that minimizes error and at the same time maximizes the probability of success?

1.4 RESEARCH OBJECTIVES

The research objective of this thesis is to propose novel techniques for price prediction and trend analysis. Further, the objective is also to apply the methods in the equities & the futures market. In this regard, the following points highlight the objectives of this thesis:

- 1. To develop a method to analyze and predict the trend in a financial time series. Further, the goal is also to observe the effect of short-term patterns on the long-term movement of stocks.
- 2. To develop a method for predicting a stock's close price and make a comparative analysis of the financial performance of all the Nifty-50 stocks using kernel adaptive filtering algorithms.

- 3. To develop a method for predicting stock's mid-price and make a comparative analysis of the financial performance of all the Nifty-50 stocks using kernel adaptive filtering algorithms.
- To develop a method for predicting the movement of stocks by augmenting the basic time series with additional statistical features. This is done to improve upon the prediction performance.

We must point it out here that these objectives are accomplished and implemented over the different time windows of the Nifty-50 dataset using different deep learning and KAF classes of algorithms. To the best of our knowledge, the work presented in this thesis is the first wherein we comprehensively analyze the price of a stock on multiple time windows and test the applicability of the methods in real trading.

1.5 THESIS OUTLINE

The subject matter of the thesis is presented in the following chapters,

- ✓ Chapter-1 provides an overview on the background and the research objectives of this thesis.
- ✓ Chapter-2 presents a thorough review of the existing methods and techniques for stock price prediction.
- ✓ Chapter-3 includes the details of the method to classify the trend of the market. In this chapter, we propose two different approaches, 3D convolutional neural network and the 3D ensemble method and compare them with different state-of-the-art methods. A detailed description of the proposed methodology, dataset, models used, experiments, and their results are incorporated in this chapter.
- ✓ Chapter-4 includes the detail of the proposed method: online KAF class of algorithms. Here, we try to predict the close and mid prices of fifty stock making up the main index of NSE: Nifty 50. The chapter includes all the details of the proposed methodology, experiments, and results.

- ✓ Chapter-5 includes the details of the feature engineering concept with the help of the KAF class of algorithms. In addition, a detailed description of the proposed methodology, dataset and model used, experiments, and their results are incorporated in this chapter.
- \checkmark Chapter-6 includes the summary and conclusion.

CHAPTER 2

REVIEW OF LITERATURE

Stock price prediction is one of the non-trivial problems of literature. Previous research has shown that stock market prediction is noisy, chaotic, and follows non-linearity (3). In this regard, numerous studies have attempted to explain that stock price prediction is difficult because of inherent non-stationarity in the data (73). In traditional prediction, the techniques were based on technical analysis with standards of resistance, support, and indicators using past prices (74). In this chapter, we summarize the existing methods and algorithms proposed by various authors to forecast stock movements and prices. We also present a performance analysis of the popular machine learning, deep learning, online learning-based, and feature-based methods. The rest of the chapter has been organized into different sections. Section 2.1 provides a general overview of the traditional methods for stock prediction. Section 2.2 and 2.3 discusses the contribution made by various authors using machine & deep learning methods. The contribution of online and feature-based methods is included in Section 2.4.

2.1 OVERVIEW OF THE STOCK PRICE PREDICTION

Stock price prediction is widely regarded as one of the most important yet difficult problems in financial research (75). However, an investor's ability to consistently exceed the market in terms of risk-adjusted return may violate the so-called efficient market hypothesis. According to the efficient market hypothesis (EMH) proposed by Fama in 1970 (4), market prices move in a random manner and that it is impossible to predict how they will change in the future.

The EMH differentiates three types of market efficiency: weak, semi-strong, and strong (4), (76).

The efficient market hypothesis and the assumption that securities are priced rationally have been challenged an increasing number of times throughout time (77), (78). Several market anomalies have been observed (79), including financial market overreaction (80) and underreaction, the presence of long-term reversal, short-term momentum, and excessive asset price volatility, all of which provide evidence against the efficient market theory (especially in its weak-form) (78). Some studies proposed EMH-compatible explanations for such anomalies, such as the fact that over and under-reactions occur at random and are equally common (4), and the idea that institutional investors may offset for the anomalies caused by less skilled investors

(81). However, there was scepticism that a model based on investor rationality could account for the abnormalities (78). This prompted a shift toward models that included human psychology, resulting in the creation of behavioural finance (80), which casts doubt on investors' perfect rationality due to behavioural biases such as overreaction, overreacting, and loss aversion (82).

One attempt to integrate the EMH with behavioural finance is the adaptive markets hypothesis (AMH), which recognises and explains the existence of anomalies in financial markets. In (83), the authors provides a comprehensive detail of the evolution of the efficient market theory. Given the possibility of market anomalies, it is not unexpected that a substantial number of market participants base their expectations for future stock prices on historical market prices, company-specific information such as previous earnings, profits, and other considerations (84). Furthermore, because historical profits may represent investor sentiment, investors generally expect short-term returns to continue. Given such expectations and the prevalence of market anomalies, using historical data to anticipate the stock market becomes reasonable. Fundamental and technical analysis are the two well-known analytical methodologies used in stock market prediction research (85), (86), (87). The focus of fundamental analysis is on the most basic facts. Fundamental information such as a company's revenue and cost, yearly growth rate, market position, and other information included in financial statements or reports (88) is used to anticipate a company's stock price or return.

If a stock index, which represents a collection of many different company stocks, is forecasted, the same type of information as well as market environment information such as exchange rates, trade, national productivity or interest rates can be used. All of these characteristics will have an effect on how the firms in the stock index operate. Technical analysis, on the other hand, is the investigation of past stock price and volume data to forecast stock price changes (89), (86), (90). The majority of previous studies forecasted stock prices and returns using statistical time-series approaches based on historical data (91). The auto-regressive moving average (ARMA), auto-regressive conditional heteroscedasticity (ARCH), moving average, autoregressive integrated moving average (ARIMA), exponential smoothing, and Kalman filtering, are the most commonly used methods. (89), (92), (93). With the rise of AI and soft computing, these techniques have attracted considerable interest in stock market prediction research. These methodologies, unlike standard time series methods, can deal with the stock market's nonlinear, chaotic, noisy, and complicated data, resulting in more accurate predictions (94). As a result, these methodologies provide novel and favourable options for financial market forecasting. Further, data is an important component of stock market forecasting and play a crucial part in the prediction process, in addition to the forecasting technique. All of the abovementioned analytical approaches use selected data (variables) for model construction over a given time period. Time series data such as stock index prices, returns, volatility, and interest rates, are frequently seen in stock price prediction (95).

2.2 TRADITIONAL ML-BASED METHODS AND DATA SOU-RCES FOR STOCK PRICE PREDICTION

ML has emerged as another popular area for time series prediction. Among the available popular techniques, machine learning methods are researched mostly due to their capabilities for recognizing the complex pattern in stock prices (96). The foundation of machine learning is the idea of knowledge extraction from data. (97). In stock market prediction, supervised learning is the most extensively utilised machine learning approach. The procedure begins with the selection of relevant information (such as financial news) and/or time-series data (such as stock price and/or return) from a given time period (98). A target class must either be predetermined or known when the goal is to solve a classification problem. The associated data must be pre-processed, which includes cleaning and removing any data that is missing or blatantly irrelevant such as identifiers. Then, using the underlying time-series data, such as closing price information, technical indicators may be generated (99). After the data has been cleaned and technical indicators have been collected, the data is further pre-processed using scaling and dimensionality reductions (such as feature extraction, and feature creation, and feature selection,) to retrieve useful variables and filter out irrelevant ones (100). Pre-processed data is frequently used to make accurate forecasts. The aim is to choose an existing or innovative machine learning approach to predict the target variable once the input data are available.

In this context, several machine learning algorithms for stock market forecasting have been explored in the literature. The most commonly used approaches include artificial neural networks (ANNs) (101), (102), support vector machines (SVMs) (103), (104), and their derivatives (105), (95), (106). Moreover, based on the time-varying and non-linearity aspect of time

series, there is a massive demand for online prediction algorithms. It follows the idea of sequential calculation and generates faster and more accurate outcomes (14). To date, various methods have been developed, such as neural networks (NN), kernel adaptive filtering (KAF) algorithms, online support vector regression (SVR), and so on (16). However, neural networks suffer from slow convergence and significant computing needs. In addition, SVR and Kernel methods have no problem falling into the local optimum. SVR has a strong generalization ability (107), but it is just suitable for smaller data. In addition, a multi-filters neural network (MFNN) is also used to predict stock price movement. The performance of the MFNN was found to be better than other NN approaches, SVM, and random forests (RF) (33). In (108), the authors combined SVR and kernel principal component analysis (KPCA) to enhance prediction accuracy that may help investors for short-term decisions.

Other machine learning approaches such as random forests (109), (110), decision trees (111), k-nearest neighbour (KNN) classifiers, and Bayesian networks (112) have also been used. With ML's recent success, it has transformed how investors utilise data and now provides ideal analytic options for all sorts of investments. As a result, machine learning is a valuable tool for assisting with financial investments. Table 2.1, Table 2.2 and Table 2.3 shows the benefits and drawbacks of each machine learning technique used in finance. Machine learning algorithms aid in the prediction of future events based on current or historical data. Regression and classification may both be used to make predictions. However, because each of the strategies described have their own strengths and limitations, several studies have attempted to improve their predicting accuracy. As a result, for predicting stock prices or returns, combinations of different algorithms such as KNN+SVM (113), (94), ANN+SVM (114), (115), and others have been examined. In (116), the authors utilized four datasets i.e. NYSE, NIKKEI, NASDAQ, and FTSE, applied seven machine learning algorithms for stock price prediction that includes random forest, bagging, AdaBoost, decision tree, SVM, KNN and ANN. The authors found that the random forest and bagging outperformed other methods. In a different study (117), the authors forecast the daily stock prices index's direction movement using ANN and SVM. The experiments were performed with the Korea Composite Stock Price Index (KOSPI), the Shenzhen Stock Exchange (SZSE), The KSE-100 index of the Pakistan Stock Exchange, and the the Nikkei 225 index of the Tokyo Stock Exchange, for the most recent ten years, i.e. from 2011 to 2020. The daily high, closing, daily low, and opening prices of various technical indicators collected from the daily stock trading were utilized as input layers to develop the architect of a single layer ANN

and SVM model with polynomial kernels, radial basis function (RBF), and linear. The findings show that ANN outperforms SVM model in terms of accuracy and F-score when predicting the direction of the daily closing price movement of the Nikkei 225 index, KOSPI index, SZSE composite index, and KSE-100 index. In addition, (118) also used ANN and SVM models as classifiers to forecast the daily Istanbul Stock Exchange 100-index. The architectures of the two models were created with the help of ten technical indicators as input layers. In (119), to anticipate the price level of five important worldwide stock markets the STSE100, DAX, S&P 500, CAC40, and Nikkei the authors employed resilient back-propagation neural networks (RBNN). In order to estimate the price level, the author used a variety of technical analysis tools, including stochastics, oscillators, indicators, and indexes along with RBNN. Moreover, in (8), the authors used four distinct machine learning algorithms, including random forest, SVM, ANN, and Naive-Bayes, with two different ways for input to these models to forecast the direction movement of the stock and stock price index for the Indian stock market. The experimental findings demonstrate that the performance of the random forest outperforms the other models when the real values of the technical indicators are utilized as input layers. The performance of all four models is enhanced for the second strategy when the technical indicators are applied as trend deterministic layers. Moreover, in (120), to predict the short-term stock prices, the authors proposed the idea of using online data sources with ensemble methods. The system has two main parts; AI platform and a knowledge base. In AI platform, the authors worked with the four widely used ensemble methods; a NN regression ensemble, a random forest regressor, a boosted decision tree, and a support vector regressor ensemble. Further, the "knowledge base" includes the following information: trends in Google searches for a particular stock ticker, the quantity of unique visitors to relevant Wikipedia pages. Historical stock prices, counts and sentiment scores of news articles published about a particular stock, a number of well-known technical indicators. The authors found that the proposed method can accurately forecast the stock price one day in advance with a mean absolute percent error (MAPE) of 1.50 percent.

Moreover, feature selection approaches (121), (122), feature extraction methods such as principal component analysis (PCA) (123), (124), evolutionary algorithms such as genetic algorithms (GA) (125), Wavelet transformations (126), and particle swarm optimizations (127), to name a few, have all improved the forecast accuracy of the methods listed above. However, the high dimension of input variables makes the learning process long, and the final model computational complexity becomes very large (113).

Moreover, feature selection methods have a lot of potentials to improve the model performance. To anticipate market trends, (128) combined SVM with hybrid feature selection method. Supported sequential forward search (SSFS) was used as the wrapper for the feature selection part, and the experiments were performed on multiple datasets. In addition, the authors of the research (111) developed a method based on the mix of multiple feature selection algorithms. Taiwan Economic Journal (TEJ) dataset was used to evaluate the proposed method. To find significant features, the authors applied principal component analysis (PCA). Further, GA and classification and regression trees (CART) algorithms were evaluated, and the comparison of the various feature selection method was also presented. In their investigation, the authors incorporated both fundamental and macroeconomic indexes. In (129), the authors found that feature selection methods improve the performance of the decision tree on multiple datasets. Also, the authors assessed both probabilistic distance-based and multiple inter-class feature selection approaches. In another study (130), the author predicted the hourly movement directions of eight banking stocks listed on the Borsa Istanbul exchange using stock prices and technical indicators as features. The authors then chose various prediction models, including linear-based (SVM), LSTM, ensemble learning (LightGBM). The performance of the model was evaluated in terms of accuracy and F-measure metrics. By utilising a straightforward eight-trigram feature engineering method of the inter-day candlestick patterns, the authors of the research (131) builds an unique ensemble machine learning framework for daily stock pattern prediction.

This framework combines traditional candlestick charting with the most recent artificial intelligence techniques. The authors found that, in most cases, technical indicators, especially momentum indicators enhances the prediction accuracy. A hybrid GA-XGBoost prediction system with better feature engineering was developed in a research (69). This system uses the hybrid GA-XGBoost algorithm to choose the optimal feature set while also expanding the feature set and preparing the data. Additionally, this study empirically showed that, in order to achieve balance and harmony between the curse of dimensionality and the blessing of dimensionality, feature engineering methods should be purposefully combined with a baseline learning model. These machine learning methods had a drawback of large time consumption during the learning process. Much of the current literature on stock market prediction pays particular attention to machine learning (ML) techniques. However, the machine learning methods' performance remains unsatisfactory and needs significant improvement.

2.2.1 DATA SOURCES

Financial variables, macroeconomic variables, and technical indicators have all been identified as the most relevant factors influencing stock price changes in the literature (111). However, because there is no universal agreement on all factors crucial to stock market forecasting, therefore, research used varied sets of variables as input data for their prediction models.

- Technical indicators Since technical indicators are crucial for determining when to buy and sell stocks, they are frequently utilised as input variables in most prediction methods. (87). Basic technical indicators and additional technical indicators are two categories into which the technical indicators can be categorized.
- 2. Macroeconomic indicators Macroeconomic variables have garnered a lot of attention in stock market research (132), (133), (134), (135). Economic performance, exchange rates, commodities, and interest rate & money supply are the four subcategories for this category.
- 3. **Fundamental indicators** Fundamental indicators related variables have also attracted interest in stock market prediction research (136), (121), (137), (85), (138), (134). Two sub-categories of such variables are also discussed: (1) "stock information variables", which are based on or connected to a specific company's stock listed on a public stock market, and (2) "balance sheet & profit and loss statement variables," which are financial reporting variables.
- 4. **Other Variable** The other variables are grouped together as "other variables," with no further subdivisions. Some research, for example, used other indices' (139), (135), (140) price data or factors collected from financial news (94), (141), (142), adhoc announcements (143), and tweets (144) to predict a specific stock market.

Table 2.1: Analysis of different methods on ML. Continued on next page in Table 2.2.

Purpose	Method	Advantage	Disadvantage
Classification	ANNs: Artificial Neu-	Ability to deal with nonlinear patterns that	ANNs are sensitive to parameter selection
and Forecasting	ral network (145),	are complicated; Modeling relationships in	since they simply deliver predicted target
	(146), (147)	data groupings with high precision Both lin-	values for unknown data without any vari-
		ear and nonlinear processes can be supported	ance information to evaluate the prediction;
		by the model.	Overfitting.
	RF: Random Forest	Since its architecture is packed with numer-	Because it generates so many trees, it needs
	(148)	ous decision trees and the feature space is	greater computing power and resources. ;In
		randomly constructed, it is a robust solu-	comparison to decision trees, it takes longer
		tion for predicting and classification issues.	to train.
		;Missing values are dealt with automatically	
		;Both discrete and continuous variables are	
		well-suited to this method.	
	RNN: Recurrent neural	It's particularly handy for displaying the	Training is a challenge.
	networks (149)	temporal interactions that exist between the	
		neural network's inputs and outputs.	
	SVM: Support Vector	Can deliver the best worldwide solution and	Outlier-aware; The choice of parameters has
	Machine (145)	has high forecasting accuracy; Works well	an impact on the sensitivity of the algorithm.
		with a variety of categorization problems, in-	
		cluding ones with a lot of dimensions.	

Purpose	Method	Advantage	Disadvantage
Forecasting	BPNN: Back propa- gation neural network (150), (151) RBF: Radial Basis Function Neural Net- works (152) SVR: Support Vector Regression (153)	 Ability to model nonlinearly in a variety of ways; High adaptability; Learning capabilities and massively parallel computing It's well-known for its ability to anticipate complicated nonlinear systems; Quick reaction; High learning precision Robust in the presence of noisy input; Because there is no back propagation learning, the training is quicker than with a perceptron; Very stable, with the potential to extend; Ability to adapt and learn in a broad sense. Effective for predicting financial time series; Pro- 	Noise-sensitive; Based on the original pa- rameters, the actual performance; Conver- gence is slow; Converging to a local mini- mum with ease. The classification process is more time con- suming than MLP. Sensitive to the free parameters set by the user
Forecasting and Clustering	ARIMA: Autoregres- sive integrated moving average model (154), (155)	For linear time series, it works well; In so- cial science, it is the most effective predict- ing tool;	Doesn't operate well with time series that aren't linear; A model that works for one se- ries may not work for another.

Purpose	Method	Advantage	Disadvantage
Clustering	Hierarchical Clustering (156)	No parameters, such as the number of clus- ters, are required to be specified.	Because of the Euclidean distance calcula- tion criterion, each time series has the same
	k-Means (156), (157)	Searches for spherical clusters well; Small	length. The number of clusters must be determined
		to medium datasets are well-served by this	ahead of time; Noise sensitivity; Clusters
	k-Medoids (PAM)	method. Searches for spherical clusters well: Small	may only be identified as spherical forms. The number of clusters must be determined
		to medium datasets are well-served by this	ahead of time; Clusters may only be identi-
		method.	fied as spherical forms; For huge datasets, it
			does not scale well.
Clustering,	GAs: Genetic Algo-	Suitable for unusually difficult issues with	The selection of parameters has an impact on
Classification and Forecasting	rithms (159), (145)	little or no knowledge of the optimum func- tion and a huge search space.	the sensitivity of the system.
	HMM: Hidden Markov	Statistically strong; Capable of simulating	It is necessary to establish parameters, and
	Model (158) (160)	high-level data (language model, or syntac-	it is reliant on user assumptions that may be
		tical rules).	incorrect, resulting in erroneous clusters; For
			a huge dataset, processing takes a long time.

Table 2.3: Continuation from Table 2.2: Analysis of different methods on ML

2.3 DL-BASED METHODS FOR STOCK PREDICTION

Deep learning is the most recent development, represented by various deep neural network models. Deep learning has seen enormous success in the last couple of years due to the availability of the large datasets, the parallel processing power of graphics processing units (GPUs), and the new network architectures (161). Deep learning models have outperformed both linear and machine learning models on tasks such as stock market prediction, thanks to their excellent capacity to cope with large amounts of data and learn the nonlinear relationship between input characteristics and prediction objective (161). Further, both the fundamental tools for deep learning and novel prediction methods have emerged rapidly. Further, in recent years, there has been a growing body of research on using deep neural networks to estimate financial market prices, such as exchange rates, stock prices, banking, option pricing, and financial crises (162). Another study (163) examines the use of evolutionary computation methods to solve financial issues, including multi-objective evolutionary algorithms, genetic programming, genetic algorithms, co-evolutionary approaches, learning classifier systems, and distribution algorithm estimates.

Besides, in order to increase the performance of standard models, hybrid networks have been created in addition to traditional multilayer feedforward networks with gradient descent backpropagation. Recent evidence reveals that cutting-edge NLP approaches are being employed for financial forecasting, which might be problematic if text such as financial news or tweets is used as input for stock market prediction (164). Another study (165) addresses a broader subject in both machine learning approaches, including deep learning, and quantitative finance, ranging from financial portfolio allocation and optimization to high-frequency trading systems. The authors of (166) focuses on fundamental and technical research, and discovers that the most often utilised machine learning approaches for stock market prediction are support vector machine and artificial neural network. The authors of (167) highlights certain challenges and research prospects based on his assessment of stock analysis including self-defeating, algorithmic trading, sentiment analysis on company filings, long-term projections, live testing,

We believe that the benefits of deep learning techniques will continue to grow with the

accumulation of historical prices and a variety of input data types, such as financial news and twitter, and that it is vital to stay current with this trend for future study. In this regard, numerous studies have attempted to explain that research effort has been made to address the prediction problem. For example, in (168), the authors used a wavelet denoising-based backpropagation (WDBP) neural network to make predictions for stock indices. In (169), the authors analyzed multi-layer perceptron, dynamic, and hybrid neural networks to make short-term predictions. Work in (170) used LSTM network for predictions. The authors of (171) made a comparison between ARIMA and LSTM. The authors found that LSTM outperformed ARIMA by a good margin. (172) proposed a combination of LSTM and GRU for stock market predictions on the S & P 500 stock index. Different versions of ResNet (v50, v101, v152) and Inception (v3, v4) class of neural networks have also been applied in this domain (173).

In (174), the authors argue that further study is needed to determine how certain important events, more especially global and local events, affect various top stock businesses (countrywise). The authors take into account four nations from the lists of established, emerging, and undeveloped economies: the United States, Pakistan, Turkey, and Hong Kong. The authors (174) specifically looked at how several significant events that had place between 2012 and 2016 affected stock markets and used the Twitter data for analysing the sentiment for each of the events. For stock exchange forecasting, the authors utilized linear regression, support vector regression, and deep learning. The authors of the research (175) worked with the Korean and the United States (US) market data from 2012 to 2018 and used top five stocks for experimentation. Further, the authors applied the concept of transfer learning to improve the predictive performance of the mode. Initially, an LSTM model is trained on large data and the second step involves fine-tuning the underlying model to improve performance utilizing a modest quantity of data from a target stock and several input feature types.

The performance of a deep feed forward NN and shallow designs, such as SVM and one-layer NN, is compared in (176) when it comes to forecasting a wide range of stock price indices in developed and emerging markets. A thorough analysis is conducted utilizing tick level, minute, hourly, and daily data pertaining to 34 financial indexes from 32 nations during a six-year period. The authors found that when using DNN to forecast stock price indices, the ReLU activation function outperforms Tanh across all markets and time horizons. In addition, the authors of (177) also worked with various ML and DL methods for stock market group

future predictions. Four groups from the Tehran Stock Exchange were chosen for experimental evaluations: non-metallic minerals, petroleum, diversified financials, and basic metals. The authors claimed that LSTM reveals more accurate results with the best model fitting ability out of all the algorithms employed in this study. Moreover, in another study (178), the authors worked with several ML and DL model on the Indian NSE stock and collected 33 technical indicators from daily stock prices, including the open, high, low, and closing price. Deep learning models outperform machine learning approaches in terms of performance.

Moreover, in another research (179), based on historical data from financial time series, this research proposes a deep learning framework to forecast price movement direction. The framework combines an LSTM network for prediction with a CNN for feature extraction. For data input into the framework, the authors employ a three dimensional CNN, including the information on technical indicators, correlation between stock indices, and time series. The experimental findings show that the framework beats state-of-the-art models in forecasting the direction of stock price movement. In addition, the authors of (180) also proposed LSTM architecture for predicting the next-day closing price of the S&P 500 index. Using the selected input variables, single and multilayer LSTM models were created, and their performances are evaluated using standard evaluation metrics. The experimental findings reveal that single layer LSTM models outperform multilayer LSTM models in terms of fit and prediction accuracy.

In another research (181), the authors proposed a time series to image conversion method utilizing 15 technical indicators and 15 intervals of technical indicators to create an image of dimension 15×15 . However, the strategy neglected the possible effect of correlated stock markets. Moreover, in one of the research (182), the authors proposed a CNN-based framework that can be used to analyze data from many sources, including various markets, to extract characteristics for forecasting the future of such markets. Based on several sets of starting variables, the proposed framework has been used to forecast the direction of movement for the DJI, S&P 500, NYSE, RUSSELL, and NASDAQ indices the next day. To address the issue of (181), the authors of (182) proposed a method to create a 3D input tensor for CNN to extract market features. Furthermore, the experimental findings demonstrated the usefulness of the 3D input tensor in extracting features, which helped to improve the model's performance in forecasting the direction of stock price movement.

Apart from the above discussed methods, there are several studies wherein the authors

comprehensively analyzed existing stock prediction methods. In (183), a large and growing body of literature was investigated that provides a systematic overview of NN applications in business from 1994 to 2015, revealing that the majority of the research has aimed at bankruptcy problems and financial distress, decision support, and stock price forecasting with a special focus on classification tasks. In another research (184), the author provides a review of the most relevant deep learning studies published between 2009 and 2015, which include approaches for projecting future market movements, preprocessing and grouping financial data, and mining financial text data, among others. In (185), the authors analyses about 100 relevant studies focus-ing on neural and neuro-fuzzy approaches developed for forecasting stock markets, including input data classifications, performance evaluation, forecasting methodology and performance measurements employed.

Deep reinforcement learning (DRL) is another popular field of the deep learning. Automated trading is one of the study fields that has benefited from the recent success of deep reinforcement learning (DRL) in handling difficult decision-making problems. Recurrent reinforcement learning (RRL), the groundbreaking work of in (186) proposed sharp ratio (SR), which was introduced as a risk-aversion evaluation as a performance measurement, has been recognised as a pioneering research in automatic trading. Before using a modified RRL technique, the authors of (187) were the first to employ a CNN to extract features from the financial information. A DRL model that takes into account trading numerous assets was presented in (188). Three different NNs (LSTM, CNN, and basic RRN) were utilized in their model, each of which had an equal say in forecasting the trade choice. The authors of the research (189) proposed DRL method with two deep learning layers. To extract the historical price data for each trading asset, the first layer consists of a collection of LSTM units, one for each asset. The interrelationships between trading equities were modeled by the second layer, called CAAN, using an attention mechanism. Furthermore, (190) proposes a unique multi-stock trading model based on free-model synchronous multi-agent DRL that can interact with the trading market and accurately reflect the financial market dynamics. An extensive historical trading data set from the American stock market is used to validate the effectiveness of the proposed model. The financial industry faces major hurdles due to the dynamic nature of stock markets and their growing complexity, which makes it difficult for rigid trading techniques created by seasoned financial professionals to perform effectively in all market situations. Deep reinforcement learning techniques and adaptive stock trading strategies are proposed as solutions to this problem

in (191). In order to effectively circumvent the drawbacks of supervised learning techniques, the work in (192) provides a DRL model to produce profitable stock market trades. The authors modeled the problem as a Partially Observed Markov Decision Process (POMDP) model. The Twin Delayed Deep Deterministic Policy Gradient (TD3) technique is then used to solve the defined POMDP problem. The reader can refer (193), (194) for further details on DRL utilization in the financial market.

Moreover, in (195), the authors applied four deep learning models, namely, multilayer perceptron (MLP), CNN, RNN, and LSTM to forecast stock prediction for the New York Stock Exchange, and the National Stock Exchange (NSE) of India. The authors found that the deep learning models performed superior than ARIMA. In addition, the authors of the research (196) also found that the deep learning models showed better predictive capability than traditional neural networks, naive Bayes, logistic regression, and decision trees. In a research (197), it was found that the trained CNN obtained satisfactory out-of-sample prediction accuracy. The use of 3D CNN is also explored in (198). The authors converted the stock data into 3D images and classified the stocks' directional trends for trend prediction; buy, sell, and hold. The experiments were performed on fifty-five different companies of the NSE-50 dataset. Another study (199) used RNN and LSTM to forecast Bitcoin price, with the feature engineering part optimized using the Boruta algorithm and achieved results similar to the random forest. In (200), authors used a deep learning system that was trained on a universal collection of financial market features. The dataset comprised all buy and sell records and cancellations of orders for the order book. The authors discovered that performing feature selection before training the model would have been better and it is an efficient technique to reduce computational complexity. The multi-filters neural network was proposed by the authors of the research (33) as an end-to-end model that employs deep learning methodologies for classification-based prediction and feature engineering on multivariate financial time series. For classification-based extreme market prediction, feature maps produced by several filters were utilized. Deep learning methodologies were shown to outperform standard machine learning and statistical approaches.

It is evident that deep learning methods have significantly improved stock price prediction. On the contrary, these methods require an adequate amount of computational resources. Another downside is that they are trained using batch learning, making them less suitable for highfrequency real-time trading. This has given opportunities for online methods. The time-varying and nonlinear nature of the online KAF-based learning methods have made them popular for stock price prediction in recent years (71).

2.3.1 DATA SOURCES

- 1. **Market data** All stock market trading actions, such as high, open, low, close prices, trading volume, and so on, are included in market data. It is utilised as an input feature for example, historical prices in a look-back window as well as a forecast target such as the close price of the next day.) (201).
- 2. **Text data** Individuals' text contributions, such as social media, news, and online searches, are referred to as text data. These data are difficult to obtain and analyse as alternative data, but they may give important information not found in market data. On this text data, sentiment analysis may be utilised to generate a sentiment factor such as neural negative or positive that can then be used for prediction (202).
- 3. **Macroeconomics data** Macroeconomic indicators such as the Gross Domestic Product (GDP), Consumer Price Index (CPI), and others, represent the economic situation of a certain nation, region, or sector. These indicators are important in the stock market because they may affirm the quality of a stock market gain or drop by displaying how healthy the overall stock market is (203).
- 4. Knowledge graph data Different firms and markets have some sort of interaction, for example, the movement of stocks in the same sector might be influenced by the same news. Knowledge graph data from free sources like FreeBase and Wikidata may now be leveraged to increase prediction performance (204), thanks to the recently built graph neural networks (205).
- 5. **Image data** Inspired by the success of convolutional neural networks in 2D image processing, such as classification and object identification, candlestick charts are used as input pictures for stock prediction (206). Despite the fact that satellite and CCTV images or videos are used to monitor the status of enterprises and may be beneficial for stock price prediction, they are never included in the surveyed papers owing to the high cost of collection and the risk of privacy leakage (207).

- 6. **Fundamental data** Accounting data, such as assets, liabilities, and so on, is the most prevalent sort of basic data that is provided quarterly. However, it is employed less frequently in research with deep learning models due to the low frequency of reporting and the inaccuracy of the reporting date. A risk of employing future data exists, for instance, since the basic data presented is indexed by the final date included in the report and comes before the release date (166).
- 7. Analytics data Data derived from reports (e.g., a recommendation to sell or purchase a stock) offered by investment banks and research organisations that conduct in-depth analyses of businesses' business models, activities, competitors, and so on is referred to as analytics data (208). While these studies may be pricey, they contain valuation information that is shared among a variety of users who all wish to profit from it.

2.4 ONLINE LEARNING METHODS FOR STOCK PRE-DICTION

To reduce the computational burden, kernel-based online learning algorithms have become gradually popular (33), (47). In this respect, recurrent kernel online learning is applied to predict the transaction price of specific products. It was observed that the model was stable with a low dependency on parameter settings (209). There is sufficient literature that suggests that modeling the movement of a stock price is non-trivial. In this respect, adaptive filtering has proved to be a standard option for the prediction model for streaming data with non-stationary properties (210). KAF can therefore be used for sequential prediction of stock prices by exploiting the market interdependence. KAF are preferred because they are non-parametric, have low computational complexity, and it converges very fast (52). In this domain, multiple algorithms are proposed for non-stationary data. They are preferred due to insensitivity towards design parameters (210). Multi-step predictions for stocks using meta-cognitive recurrent kernel Online Learning are proposed in (211).

Due to its elegance and simplicity, the least-mean square (LMS), which can only be described in terms of inner products, is the workhorse of adaptive signal processing (47). As a result, it has been a natural choice for online kernel-based learning. KAF is not widely applied on stocks data for intraday trading, but its applied on different time series data. Contrary to the widely held belief in kernel techniques, the authors of the research (47) demonstrates that the KLMS algorithm may be properly posed in RKHS in the finite training data scenario without the need for an additional regularization term to penalize solution norms. From the perspective of regularization, the impact of the KLMS step size is also investigated. In order to substantiate the claim that the KLMS method can easily be applied in high-dimensional spaces, notably in RKHS, to create stable, nonlinear algorithms with performance comparable to regularized, batch solutions two experiments were conducted.

A type of kernel approach that has gained significant attention in the study of time series online prediction is kernel recursive least squares (KRLS) (49). KRLS updates in a recursive fashion and has a low computational cost. However, the computational difficulty of constructing the kernel inverse matrix rises as data size does. Additionally, it has several issues with surroundings that change over time. Additionally, it has some difficulties in adapting time-varying environments. To address the challenges of KRLS, in (16) the authors presented an improved KRLS algorithm for multivariate chaotic time series online prediction. The authors validated the effectiveness of the improved KRLS on three time series; yearly sunspots and runoff of the Yellow River chaotic time series, ENSO indexes chaotic time series, and Lorenz chaotic time series.

In another study (212), a modified quantized kernel least mean square (M-QKLMS) algorithm, an enhancement of the quantized kernel least mean square (QKLMS) is proposed wherein the gradient descent approach is used to update the filter coefficient. In a research (213), the authors proposed a KAF-based framework for stock price prediction. By taking into account the inter-dependencies between stock markets, the framework sequentially predicts stock prices. The authors validated the frameworks' effectiveness on 24 stocks from three major economies and achieved comparatively low values for MSE.

CHAPTER 3

SECTOR INFLUENCE AWARE STOCK TREND PREDICTION USING 3D CONVOLUTIONAL NEURAL NETWORK

3.1 INTRODUCTION

This chapter tries to classify the trend in a stock's price and observe the effect of short-term patterns on the long-term movement of stocks. The main objective of the work is to increase the annual return on investment while simultaneously capturing the influence of one company on another from the same sector. Specifically, we attempt to classify the trend into three categories: Buy, Sell, and Hold. The research is motivated from the fact that in recent years CNNs have shown remarkable performance improvement. In this regard, CNNs have shown excellent breakthroughs in image classification (11), hence, following this precedent, we also convert a financial time series into an image (or 3D image). To do that, we complement a stock's closing price via additional features (technical indicators) (169). Furthermore, and in contrast to existing literature, e.g., (172), (181), (214), we do not focus on predicting the trend in a single company, rather, the goal is to propose a framework that could predict the trend while analyzing a sector as a whole. To that end, multiple technical indicators are computed, out of which the best fifteen indicators (or features) are selected via Hierarchical Clustering. After that, data of five similar companies are grouped together and the entire compendium is put into an $15 \times 15 \times 5$ image. Classification of a stock's trend happens on this 3D image. For the purpose of classification, we experiment with 3D CNN and ensembled-CNN. In the context of the proposed ensembled-CNN, studies have found that these methods offer strategies for combining many a single classifier into a global model (215), (216). This combination of methods outperform standalone classifiers in terms of numerical accuracy. Hence, we also follow the same idea in this chapter. To validate the viability of the proposed work in practice, experimentation is performed on real datasets.

We work with forty-five different stocks at the National stock exchange (NSE). Through extensive numerical simulations, we have found that the work presented in this chapter outperforms similar techniques in the literature. Particularly, the work presented here shows good performance in terms of accuracy and annual returns. The model achieved up to 35% returns in some cases, with the average being 9.19%. A more detailed discussion on the numbers is presented in Section 3.3. It should be noted here that for some stocks, the proposed ensemble-CNN outperformed the proposed 3D CNN, whereas the reverse happened in a few cases. Nevertheless, through the work proposed in this article, the trend capturing capacity, however modelled,

is indeed admirable.

We must point out here that the overall objective of this chapter is to present a set of guidelines that could help future work in trend prediction. Moreover, through the work presented here, we also want to emphasize upon the idea of Sector aware stock prediction to achieve optimum results. In other words, we want to engineer an algorithm trading protocol that could generate signals for the medium and the short term. These signals would deliver optimum results with a minimum risk-to-return ratio. In this regard, the following points summarize the essence of the chapter in brief:

- We propose a 3D CNN and a 3D-ensemble CNN approach for stock trend classification. This is accomplished by transforming a non-stationary time series into images. Subsequently, the images are categorized into: Buy, Sell, or Hold.
- The main objective of the chapter is to increase the annual return on investment while simultaneously capturing the influence of one company on another from the same sector.
- We compute several technical features for the financial time series. Out of which, the best ones are selected with the help of Hierarchical Clustering. We have also tried applying K-means, however, it produced inferior results.
- We compare the proposed method with different state-of-the-art methods in the literature. Through extensive numerical testing performed on forty-five stocks, we try to show the supremacy of the work in practice. We further make an attempt to show that though the idea of grouping companies together is new, it could be the way forward in trend classification.
- To the best of our knowledge, first we apply the paradigm of 3D CNN and ensembled-CNN to examine the effect of sector aware trend classification.

This chapter has been organized into various sections. The proposed methodology is discussed in Section 3.2. The experiments performed and the results are described in Section 3.3. We discuss the shortcomings of the work in Section 3.4. Finally, the conclusion and future directions are given in Section 3.5.

3.2 PROPOSED METHODOLOGY

This section expands upon the details of the proposed approach. The overall methodology has been divided into three main parts. They are as follows:

- 1. The first subsection describes the model architecture in detail.
- 2. In the second subsection, we discuss the procedure of feature selection using hierarchical clustering. Herein we also expand upon the process of converting a non-stationary time series into an image.
- 3. Finally, a discussion on labeling of stock's close-price is presented in Section 3.2.3.

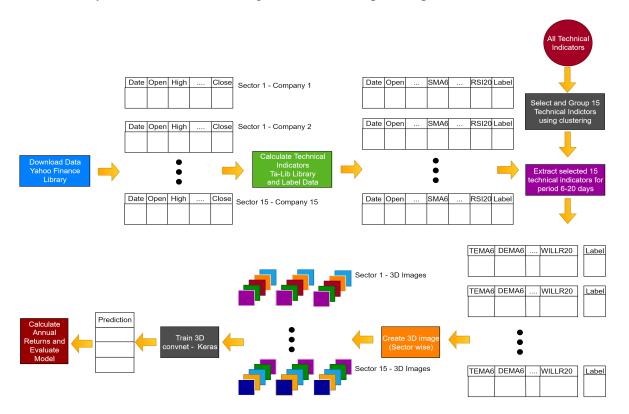


Figure 3.1: Proposed framework for Stock Trend Classification.

3.2.1 MODEL ARCHITECTURE

The overall framework of the proposed approach is presented in Figure 3.1. As visible, the system takes in the Open, High, Low, Close prices. From these variables, various technical

indicators are computed and additional features are constructed. Note, these features are computed for different time periods (6-20). Out of these multiple features, we find the best ones via hierarchical clustering. It was discussed in the previous section that the focus of this chapter is on a certain sector, therefore, five similar companies are grouped together. The entire data is then converted into a 3-Dimensional image of size: $15 \times 15 \times 5$. This image was fed into the proposed system for classification. For classification purpose, we have used two different models. In the rest of the section, we expand upon the procedure in detail.

3.2.1.1 3D CONVOLUTIONAL NEURAL NETWORK

Convolutional neural networks (CNNs) are feed forward neural networks that take 2D or 3D images as its input. The complete architecture of the CNN is shown in Figure 3.2. In this approach, the image of dimensions $15 \times 15 \times 5$ is given as input to the model. The first layer pads the input image with three zeros along all three dimensions. The next two layers are the 3D convolution layers with filters of size $2 \times 2 \times 2$ and a stride of $1 \times 1 \times 1$. The numbers of filters used in the two convolution layers are 32 and 64, respectively. Next, we apply a 3D max pooling layer, also known as sub-sampling layer, with a filter of size $2 \times 2 \times 1$. This sequence is repeated, and we apply two more 3D convolution layers, followed by another 3D max pooling layer. The parameters used in these layers are the same as the previous layers, except, the number of filters in the two convolution layers is increased to 128 and 256, respectively. This resulted in an output of shape $3 \times 3 \times 7 \times 256$. This 3D array constituting of 256 channels is flattened into an array of size 16128 (product of 3, 3, 7, and 256) and fed into a dense layer of 1000 neurons. This layer is also connected to another dense layer having 200 neurons. Both the dense layers use a rectified linear unit (ReLU) activation function. Finally, this layer is connected to a layer with 15 neurons, and the output is reshaped to 5×3 . Further, a softmax activation function is applied to this output to make final predictions for five companies simultaneously.

3.2.1.2 ENSEMBLED CNN

In this section, we discuss the details of the proposed ensembled CNN model. It should be noted here that in this article, we have complemented the idea of a 3D CNN with that of an ensembled-CNN. In the context of building an ensemble model, three factors must be kept in mind. They are as follows:

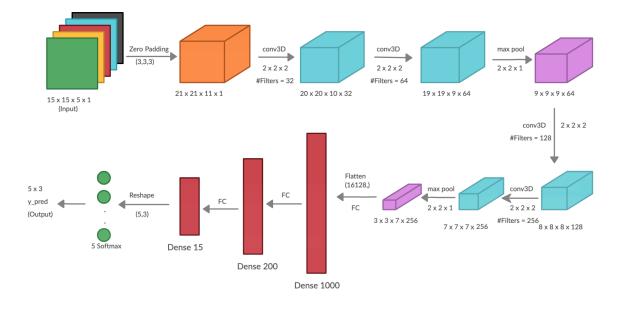


Figure 3.2: Architecture of the proposed 3D CNN model.

- 1. Which classifier should be put together in the ensemble?
- 2. How to combine the result of the chosen classifiers?
- 3. How many classifiers should be ensembled?

To answer the first and the third question, we have combined the proposed 3D CNN and the methods available in (181), (217), and (218). Lastly, to answer the second question, We use a maximum voting ensemble approach to determine the optimum solution. It should be noted here that the method presented in (181) is the predecessor of the work presented in this article. From the discussion in this and the previous subsection, we now have a method that can classify stocks. In the following subsections, we expand upon the details of converting a financial time series into an image.

3.2.2 FEATURE SELECTION AND IMAGE GENERATION

In this section, we highlight the process of feature selection and image creation. We start the discussion with feature selection.

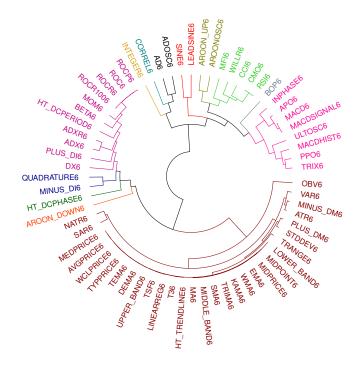


Figure 3.3: Hierarchical Clustering of Technical Features.

3.2.2.1 FEATURE SELECTION USING HIERARCHICAL CLUSTERING

To construct additional features, we use the notion of technical indicators. Technical indicators are used to examine historical information about stocks and estimate their future. They are a part of the technical analysis domain of financial instruments. When selected and used appropriately, technical indicators may reveal incredibly valuable information about a company's stock. We calculated several technical indicators such as Chande Momentum Oscillator (CMO), Relative Strength Index (RSI), Channel Commodity Index (CCI), Williams' %R (WILLR) etc. To be precise, we computed sixty-seven different technical indicators. During analysis, we realized that similar indicators must be grouped together, i.e., they must be placed close to each other in the 3D image. The intuition behind this is that since we are modeling data as a 3D image and passing it through a CNN, our data must have the characteristics of an image. In an image, adjacent pixels are related to each other by default, and they form patterns or features, for example, the shape or color of the object. A CNN, by its design, uses these characteristics for classification. Therefore, while forming images, we placed similar indicators together. However, a fundamental question remained: How to find the best indicators? Moreover, how to find similar indicators? To help answer these questions, we used hierarchical clustering (219).

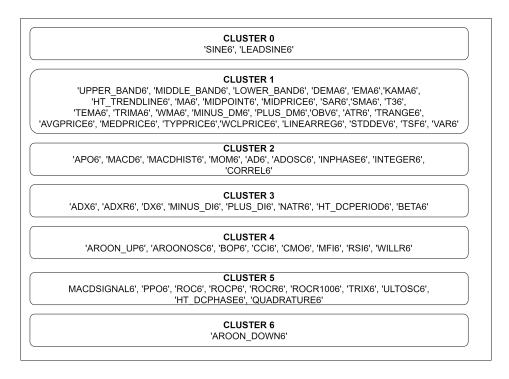
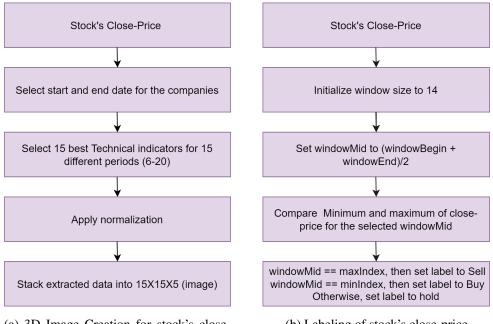


Figure 3.4: Clustering using K-means with K=7.

Hierarchical clustering produces a robust and reliable set of similar features in the form of a dendrogram. Compared to other methods like K-means, it doesn't require any manual intervention. Furthermore, compared to K-means clustering, Hierarchical clustering offers various advantages like fast processing, moreover, the number of clusters does not have to be defined ahead of time. Further, the dendrogram is also used to illustrate features and the relationship between them. To that end, we create a Euclidean distance matrix between all the data points and use linkage-based hierarchical clustering. Figure 3.3 shows the dendrogram obtained after applying the algorithm on the sixty-seven features. Based on this analysis, the top fifteen features were selected and placed in the same order in the image. It should be noted here that we also tried our hand at K-means clustering for feature selection. However, the procedure was too tedious and required a lot of manual intervention. A sample set of similar features, obtained via K-means, is presented in Figure 3.4. We, however, dropped this method owing to performance depreciation and too much manual analysis.

3.2.2.2 CONVERTING A NON-STATIONARY TIME SERIES INTO IMAGES

This subsection discusses the conversion of stock data into 3D images. To do so, we have proposed a novel technique for converting a financial time series into a 3D image. The overall



(a) 3D Image Creation for stock's closeprice.

(b) Labeling of stock's close-price.

Figure 3.5: Image creation and labeling of stock's price.

working of the procedure is summarized in Algorithm 1. To convert the financial time series into 3D images, we first group companies based on the sector they belong to, such as automobiles, metals, etc. After grouping, we kept 15 companies in each sector to maintain consistency. Next, we extracted 15 technical indicators (refer to section 3.2.2.1) for each company and for 15 different periods (6-20). This generated a 2D image of size 15×15 . The next step is to stack 15 x 15 images to form 3D images. However, during this extraction process, we noticed that the available data for the companies (start date and end date) differed heavily. Hence, if we stack the data of all fifteen companies together, selecting common dates would lead to very few 3D images. Therefore, we further divided the group of fifteen companies into three groups of five companies based on nearby start dates. Thus, we create three subgroups in each sector. Next, we created 3D images by stacking fifteen technical indicators for fifteen different time periods (the 15×15 images, as described previously). This creates the training images of size $15 \times 15 \times 5$ each. To generate the label for an image, first, each company's labels (Buy, Hold, Sell) were generated independently. After that, we stacked five companies' labels together, similar to the way we stacked five companies to create a single input image. The labels were then encoded as one-hot vector, so with respect to each $15 \times 15 \times 5$ image, we have a 5×3 label. Thus, the output of the algorithm is a set of images of size $15 \times 15 \times 5$ and a set of labels (Buy, Hold, Sell), with the shape of 5×3 . It should be noted that the above-discussed procedure is for generating images from one sector. The same procedure was repeated for every other sector. The main steps of Algorithm 1 are also summarized in Figure 3.5 (a). Moreover, in Figure 3.6 several images belonging to the Consumer Durable Goods sector are presented. The companies used in this image are Johnson Controls Hitachi Air Conditioning, Havells India, Voltas, IFB India, and Blue Star. Although these five companies are grouped together and trained in the process, Figure 3.6 shows all the five companies separated along with their respective labels. Three different versions, raw grayscale, smoothened grayscale, and smoothened color images, are shown here.

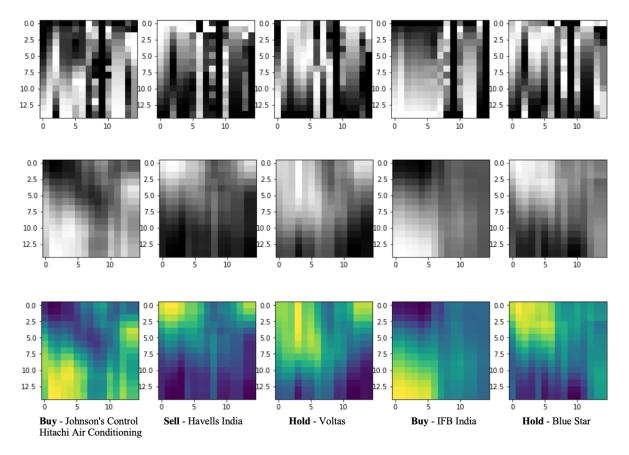


Figure 3.6: One 3D image split into its components. Raw image (Gray Scale, top row), Smoothened Image (Gray Scale, middle row), Smoothened Image (Colored, bottom row).

3.2.3 LABELING OF STOCK'S CLOSE-PRICE

Labeled data is the fuel for supervised learning. This section presents the labeling procedure that has been used to label the stock's close price into buy, sell, and hold labels. In this respect, the procedure for labeling followed in the article is shown in Algorithm 2. It should be noted here that this method is taken from (181). The input to the algorithm is the data frame that contains the close price for each day. The output is the label. The approach used here is a sliding window approach. A window of size 14 is selected for computing labels. In each iteration, we compute the maximum & minimum closing price within that window.

Next, we compared the middle index of the window to the maximum & minimum indexes obtained in the window. If the maximum index corresponds to the middle index, the label is set to "SELL." The reason is that if the price is higher, it will be profitable to sell. Furthermore, if the minimum index corresponds to the middle index, the label is set to "BUY" as it will be cheaper to buy stocks at less rate. If none of these two cases are applicable, then we label it as "HOLD". Once the labeling process is done, and images are generated, they are fed into the model described in Section 3.2.1 for classification. Figure 3.5 (b) shows the procedure followed for labeling the stocks.

Algorithm 1: Converting stock data into 3D images.

```
Input: set of five companies C
   Output: set of images I (15 \times 15 \times 5), set of labels L (5 \times 3)
1 Initialization: initialize start date (S_{date}) and end date (E_{date}) to None
   // Select the start and end date
2 S_{date} \leftarrow Maximum (start date for each company c \in C)
 3 E_{date} ← Minimum (end date for each company c ∈ C) 
   // 3D images conversion
4 foreach date j \in (S_{date}, E_{date}) do
       // Form a 15 by 15 2D array
       foreach company c \in C do
5
           Row_{ii} \leftarrow c_i[j]
 6
           x_{ij} \leftarrow Select 15 technical indicators for 15 periods from Row_{ij}
 7
           Reshape x_{ij} to 15 ×15 Matrix
 8
           Apply MinMaxScalar to x_{ij}
 9
           y_{ij} \leftarrow Row_{ij}[Label]
10
       end
11
       // Stack matrix and labels
       Stack x_{ij} to form a 15 \times 15 \times 5 matrix x_j
12
       Stack y_{ij} to form a 5 × 3 matrix y_j
13
       Add x_i to images I
14
       Add y_i to labels L
15
16 end
```

17 Return 3D images I and respective labels L

Alg	orithm 2: Labeling of stock's close-price
I	nput: Close price of a stock
C	Dutput: Labels for each stock
1 II	nitialization: <i>windowSize</i> = 14
2 W	while rowCounter < number_of_Days do
3	$rowCounter \leftarrow rowCounter + 1$
4	if rowCounter>windowSize then
5	windowBegin \leftarrow counterRow – windowSize
6	$windowEnd \longleftarrow windowBegin + windowSize - 1$
7	$windowMid \leftarrow (windowBegin + windowEnd)/2$
8	for $i \in (windowBegin, windowEnd)$ do
9	$Price \leftarrow ClosePrice[i]$
10	if Price <min td="" then<=""></min>
11	$min \leftarrow Price$
12	$minIndex \leftarrow i$
13	end
14	if Price>max then
15	$max \leftarrow Price$
16	$maxIndex \leftarrow i$
17	end
18	end
	// assign labels
19	If maxIndex == windowMid, Label = 'Sell'
20	If minIndex == windowMid, Label = 'Buy'
21	Otherwise, <i>Label</i> = 'Hold'
22	end
23 e	nd
24 R	Leturn Labels for each stock

3.3 **RESULTS**

3.3.1 DATASET AND EVALUATION CRITERION

In this section, we try to validate the efficacy of the proposed work in practice. To do that, experimentation is performed on the NSE dataset¹. The dataset consisted of the following fields: Open, High, Low, and Close prices for each day. Since the proposed method has a sector-wise focus, hence, we selected fifteen different sectors in our analysis. These sectors are: consumer durable goods, fast-moving consumer goods, financial service, information technology, automobile, banking, energy-oil and gas, pharmaceuticals, metals, media and entertainment, telecommunication, cement, energy-power, construction/realty, and textiles. For each of the sectors mentioned here, we selected three companies each to show the results. Some of these companies are Glenmark Pharmaceuticals, Ambika Cotton Mills, Sun Pharmaceuticals, NDTV, Axis Bank, ICICI Bank and so on. The dataset was normalized between one and minus one. Further, as discussed in section 3.2, we converted the numeric data into images of size $15 \times 15 \times 5$. To evaluate the performance of the proposed method, standard metrics are used (220). They are defined as follows:

$$Precision = \frac{True \ Positives}{True \ Positives + False \ Positives}$$
(3.1)

$$Recall = \frac{True \ Positives}{True \ Positives + False \ Negatives}$$
(3.2)

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$
(3.3)

¹https://in.finance.yahoo.com/quote/%5ENSEI/

It should be noted here that the Buy, Hold, and Sell labels were generated via Algorithm 2. The models were trained and evaluated over a period of 12 epochs with adam optimizer (221) and categorical cross-entropy loss (222). Note that prior to experimentation, we broke the dataset into two parts. The first part consisted of stock prices from January 1, 2000, to December 31, 2016. This part was used to train the model. The second part consists of data from January 1, 2017, to December 31, 2019 (3 years of data). This was used for out-of-sample testing.

Model details	Results			
		Hold	Buy	Sell
	Precision	0.95	0.00	0.00
No undersampling	Recall	1.00	0.00	0.00
	F1-Score	0.95	0.00	0.00
	Overall Accuracy		90%	
	Precision	0.92	0.24	0.22
55% undersampled	Recall	0.88	0.35	0.30
	F1-Score	0.90	0.28	0.26
	Overall Accuracy		83%	
	Precision	0.94	0.21	0.21
71% undersampled	Recall	0.79	0.50	0.52
	F1-Score	0.86	0.29	0.30
	Overall Accuracy		76%	
	Precision	0.94	0.17	0.18
81% undersampled	Recall	0.72	0.54	0.57
	F1-Score	0.81	0.26	0.27
	Overall Accuracy		70%	

Table 3.1: Results after undersampling by various amounts.

3.3.2 IMPORTANCE OF UNDERSAMPLING

During analysis, we observed that the dataset has many Hold labels with very few Buy and Sell Labels. More precisely, there were 5,62,383 Hold labels and, approximately 60,627 Buy and Sell labels each. It is therefore understandable that the dataset is imbalanced. Therefore, to address this, we decided to undersample the data and make the dataset balanced. Therefore, the images with Hold labels were removed. This naturally increases the proportion of Buy and Sell labels in the data set. With this modification, the effect of removing Hold labels by different degrees is shown in Table 3.1. From the results shown in Table 3.1, it is clear that a model, when trained on all the data without any undersampling, does not perform well. It has an F1-Score of 0.95 on Hold classes and F1-Score of 0.01 and 0.00 on Buy and Sells classes, respectively. We, therefore, undersample by various degrees. In this experiment, we observed that too much undersampling also did not result in an improved model either. This is visible from the number presented in Table 3.1. Through several rounds of numerical simulations, we found that with 55% undersampled data, we get the best performance.

3.3.3 EXPERIMENTATION WITH DIFFERENT ACTIVATION FUNC-TIONS

Once the best model was selected and trained after undersampling data by various amounts, we went one step further. We made a few changes to the proposed architecture. The hypothesis was that this would help improve the performance. In other words, we experimented by adding dropout and tried different activation functions. With this setup, the results are summarized in Table 3.2. It is interesting to note that all the variations improved the overall accuracy of the model, however, it was done at the expense of the F1-score. This is unacceptable, as we want the model to make a decent enough prediction on all the classes. Hence, some trade-off is needed. In our proposed architecture, we used softmax to classify the labels. Therefore, we concluded that the modifications did not help much, and the experiment was a failure.

Model	Results				
Details		Hold	Buy	Sell	
	Precision	0.92	0.24	0.22	
Original	Recall	0.88	0.35	0.30	
undersampled	F1-Score	0.90	0.28	0.26	
	Overall Accuracy		83%		
	Precision	0.91	0.29	0.27	
Added	Recall	0.99	0.01	0.03	
Dropout	F1-Score	0.95	0.03	0.05	
	Overall Accuracy		90%		
	Precision	0.91	0.22	0.24	
ReLU	Recall	0.95	0.15	0.12	
Applied	F1-Score	0.93	0.18	0.16	
	Overall Accuracy		87%		
	Precision	0.91	0.27	0.29	
Tanh	Recall	0.97	0.12	0.10	
Applied	F1-Score	0.94	0.17	0.15	
	Overall Accuracy		89%		

Table 3.2: Results after making changes to model.

3.3.4 COMPARISON WITH SIMILAR METHODS

In addition to the failed experiment presented in section 3.3, we also compared the performance of the proposed work with other similar methods such as 2D CNN (181), RSI14 (217), Buy and Hold (BAH) (223), and SMA50 (218). For the proposed ensemble classifier, we combined the results of (181), (217), (218), and proposed 3D CNN. We aggregated the predictions from these models for our ensemble model and selected the class that was predicted the maximum number of times. To do this, we calculated annual returns over the out-of-sample data from 1 January 2017 to 31 December 2019. It should be noted here that equations (3.4)-(3.7) were used to buy or sell a stock and to compute the annual returns. Further, the stocks were held, sold, or bought

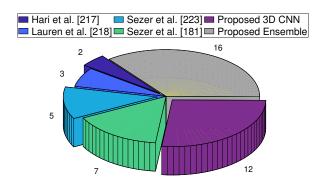


Figure 3.7: Comparison of different methods. The numbers here represent on how many stocks a method performed best in terms of annual returns. For instance, the proposed 3D CNN performed best in twelve stocks out of forty-five.

as per the predicted label.

$$NumStock = \frac{tot_{Money}}{Price} \qquad if \, label = 'Buy' \tag{3.4}$$

$$NumStock = \frac{tot_{Money}}{Price} \quad if \, label = 'Sell' \tag{3.5}$$

NoAction
$$iflabel = 'Hold'$$
 (3.6)

$$AnnRet = \left(\left(\frac{FinalMoney}{startMoney} \right)^{\frac{1}{No \ of \ Years}} - 1 \right) * 100$$
(3.7)

With the setup defined in the previous paragraph, the results are presented in Table 3.3, 3.4 and 3.5. In this table, we have presented annual returns for all the forty-five stocks in the dataset. The numbers presented in these tables clearly outlines the superiority of the work and could

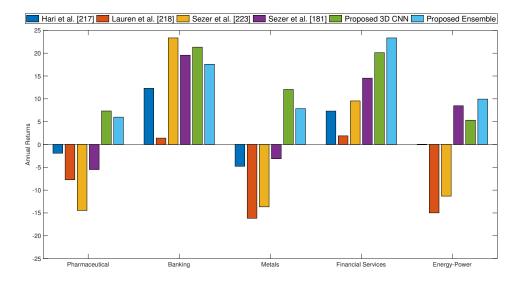


Figure 3.8: Average sector-wise annual returns.

indeed serve as a new benchmark in trend classification. It can be seen from Table 3.3, 3.4 and 3.5 that the proposed 3D model performed best in twelve out of forty-five stocks, whereas the proposed 3D-ensemble method gave the highest return in sixteen out of forty-five stocks. For better visualization, the numbers are also presented in Figure 3.7. From the figure, we can see that the proposed methods performed well on 62.22% of all instruments in the dataset. This number is really astonishing. In addition to this, Figure 3.8 shows the average annual return for a few companies and for five random sectors in the dataset. It is visible from Figure 3.7 that the proposed method is a clear winner in terms of annual returns. From the evidence presented in this section, it can therefore be said that the performance achieved here illustrates the viability of the work in practice.

3.3.5 CAPTURING THE INFLUENCE OF ONE COMPANY ON AN-OTHER

In this section, we emphasize upon the importance of sector wise classification of trends. To do that and to capture the effect of one company on another within the same sector, we trained the base 3D CNN model differently. We grouped four companies from the same sector together, and tested on a company from a different sector. We compared the annual returns on these

Company Details	Details			Annual Ret	l Returns		
Sector	Stock	[Hari et al.]	[Lauren et	[Sezer et	[Sezer et al.]	Proposed	Proposed
		(217)	al.] (218)	al.] (223)	(181)	model	Ensemble
	AMBIKCO	-14.88%	-14.56%	-13.48%	-2.89%	-0.46%	-4.063%
Textile	GARFIBERS	13.90%	-12.20%	10.31%	0.20%	1.49%	8.46%
	RAYMOND	-1.03%	7.55%	1.94%	8.71%	-6.11%	3.75%
	SUNPHARMA	1.35%	-4.63%	-16.16%	-2.83%	2.28%	6.87%
Pharmaceutical	DRREDDY	3.03%	-2.95%	1.69%	9.11%	20.64%	21.26%
	GLENMARK	-10.12%	-15.61%	-28.87%	-22.75%	-0.90%	-10.15%
	NDTV	-5.97%	-11.83%	-28.54%	-24.56%	-7.10%	-2.14%
Entertainment	ZEEL	3.70%	-19.87%	-20.39%	7.46%	7.11%	-4.58%
	PVR	20.26%	-19.98%	0.57%	-8.47%	17.09%	15.37%
	NHPC	0.97%	-9.45%	-8.39%	0.25%	9.61%	6.54%
Energy - Power	SJVN	-2.28%	-21.94%	-19.13%	7.46%	10.63%	17.10%
	IEX	1.26%	-13.56%	-6.40%	17.73%	-4.29%	6.18%
	SAGCEM	-7.22%	-22.31%	-13.32%	-1.11%	5.41%	-1.64%
Cement	RAMCOIND	-4.68%	-13.08%	-5.78%	-15.49%	7.63%	0.29%
	ACC	-2.49%	-9.05%	0.80%	14.33%	-4.79%	-2.52%

Table 3.3: Comparison of Annual Return with the other approaches. Continued on next page in Table 3.4.

age in Table 3.5)
other approaches. Continued on next page in Table	-
	11
al Return with the o	
omparison of Annua	Ŧ
rom Table 3.3: Co	
I: Continuation f	
Table 3.4	

Company Details	Details			Annual	Annual Returns		
Sector	Stock	[Hari et al.]	[Lauren et	[Sezer et	[Sezer et al.]	Proposed	Proposed
		(217)	al.] (218)	al.] (223)	(181)	model	Ensemble
	INFRATEL	-7.22%	-6.44%	-26.45%	-1.09%	5.19%	10.29%
Telecommunication	HFCL	14.45%	-4.19%	10.23%	16.00%	16.63%	25.48%
	ASTRAMICRO	-16.78%	-10.59%	-18.70%	12.24%	-11.47%	0.65%
	AXISBANK	10.72%	-10.33%	14.41%	28.34%	26.97%	22.61%
Banking	ICICIBANK	10.90%	5.61%	30.42%	17.96%	20.13%	15.70%
	CUB	15.29%	8.91%	25.20%	12.37%	16.87%	14.41%
	NATIONALUM	-1.20%	-18.51%	-18.81%	-10.51%	12.60%	5.46%
Metals	VEDL	-11.59%	-16.91%	-17.72%	5.12%	17.10%	13.35%
	MOIL	-1.56%	-13.08%	-4.45%	-3.96%	6.45%	4.73%
	IOC	0.96%	-17.62%	-12.60%	1.03%	6.08%	16.85%
Energy - Oil & Gas	HINDPETRO	-2.68%	-8.53%	-8.98%	0.68%	9.78%	20.65%
	AEGISCHEM	-6.88%	-11.61%	-5.28%	-6.49%	2.79%	7.91%
	TVSMOTOR	-13.09%	-2.31%	1.39%	9.79%	10.06%	10.64%
Automobile	HINDMOTORS	4.56%	-25.01%	-17.24%	10.75%	25.93%	29.01%
	ASHOKLEY	-10.81%	6.46%	-4.15%	12.32%	10.55%	13.21%

Company Details	ails			Annual	Returns		
Sector	Stock	[Hari et al.]	[Lauren et	[Sezer et	[Sezer et al.]	Proposed	Proposed
		(217)	al.] (218)	al.] (223)	(181)	model	Ensemble
	MAHLIFE	4.49%	5.42%	4.19%	8.52%	10.85%	6.35%
Construction/Realty	AHLUCONT	7.46%	6.48%	-3.09%	1.04%	6.14%	10.53%
	OMAXE	-10.32%	3.53%	-3.08%	2.76%	-4.16%	-4.45%
	CHOLAFIN	19.82%	-7.79%	16.96%	33.63%	34.98%	32.38%
Financial Services	SBILIFE	10.74%	13.15%	17.80%	19.36%	23.11%	32.41%
	PFC	-8.61%	0.31%	-6.15%	-9.41%	2.23%	5.24%
	TCS	13.08%	2.63%	23.30%	2.84%	10.00%	12.74%
Information Technology	HCLTECH	11.76%	-8.09%	11.50%	9.50%	11.16%	15.84%
	QUESS	-3.98%	5.23%	-9.99%	-7.37 %	4.86 %	2.09%
	VGUARD	11.23%	-4.43%	7.09%	12.25%	17.05%	15.89%
Consumer Durable Goods	IFBIND	-5.60%	-0.78%	3.11%	1.32%	7.92%	-6.65%
	TTKPRESTIG	7.15%	7.08%	6.06%	18.44%	5.99%	5.95%
	EMAMILTD	-14.92%	-3.31%	-22.25%	-19.92%	-19.76%	-8.93%
Consumer Goods	PG	9.09%	7.99%	20.21%	17.56%	1.58%	3.20%
	BRITANNIA	3.99%	24.55%	27.35%	18.63%	16.80%	19.29%
	Overall Aver-	0.81%	-5.68%	-2.33%	4.46%	8.06%	9.19%
	age						

Table 3.5: Continuation from Table 3.4: Comparison of Annual Return with the other approaches.

companies with the values obtained when they were grouped with the companies from the same sector. For instance, say a company, NHPC, belongs to the energy sector but was placed with four companies from the FMCG sector. Similarly, DRREDDY is a pharmaceutical company, but it was grouped with companies from the textile sector. Following the same pattern, other stocks were also placed with companies from a different sector. With this setup, the results are presented in Table 3.6. In the table, we have also compared the idea when all five companies from the same sector are experimented with. For reasons of brevity, we have shown results for only five companies. Similar figures were obtained for other companies as well. From the evidence presented in this table, it can be seen that the returns are not high when companies from different sectors are clubbed together. On the contrary, when companies from the same sector are placed together, the return are quite good. This evidence, therefore, validates our point of using companies from the same sector together to make a prediction.

	Ann	ual Returns
Stock	Same Sector	Different Sector
NHPC	9.61%	3.85%
DRREDDY	20.64%	7.11%
HINDMOTORS	25.93%	4.88%
TVSMOTOR	10.06%	-5.70%
VEDL	17.10%	0.72%

Table 3.6: Results of same sector Vs different sector.

3.4 DISCUSSION

This paper presents a novel deep learning-based approach for stock trend prediction using 3D CNN. Further, the basic 3D CNN was complemented via the proposed 3D-ensemble CNN. The main objective of the work was to increase the overall annual return, thereby establishing its superiority in practical deployment scenarios. From the discussion presented in Section 3, we showed that the model indeed worked well, however, the work is not without a few shortcomings

of its own. They are as follows:

- It is a commonly held notion that market analysis can help anticipate future behavior. In this regard, we have tried to examine one of the non-trivial problems of literature. The concepts or the ideas presented here should not be regarded as a complete solution. We discussed several methods to enhance the prediction performance and did our best to provide a comprehensive analysis of the same. Furthermore, the work presented here reflects upon a few starting points and presents a potential roadmap that might help in the development of a better and more efficient system. Nevertheless, we must reiterate here that the framework presented here should not be taken as the final solution.
- It can be seen from Table 3.1 that when the models were trained on all the data, without any undersampling, the performance was below par. The F1-Score was 0.95 on the class Hold, and F1-Score was 0.01 & 0.00 for the Buy and the Sell class, respectively. Hence, it is clear that undersampling was necessary to improve the performance of the model. This is the case of the classic imbalanced class classification problem (224). In such a scenario, machine learning and deep learning algorithms do not perform well. In this article, we were also not able to handle this problem. The sole motive of the work was to design a sector influence aware scheme for trend classification.
- According to our findings, the proposed 3D CNN based ensemble model beat all other models in terms of profitability. Our proposed model performed well because we tried several combinations that gave the best results. However, it is also true that the model did not perform well in every case. Moreover, in this article, we are taking a one-day time window for analyzing the stock's trend. For other time windows, such as one week or one month, the proposed approach might perform poorly. This, however, is acceptable as for a different dataset, the method or even the mode of modelling the returns might be different. The statement is backed by the no free lunch theorem in machine learning (225). Hence, one has to experiment and try all possible combinations to get the best answer.
- Connected to the previous point is the issue of testing time period. It is visible from Table 3.3, 3.4 and 3.5 that among all methods, the proposed 3D CNN and proposed 3D-ensemble model gave superior results. This signifies that the proposed method is a good

alternative in terms of returns. In this respect, it is important to note that we considered the testing period from January 1, 2017, to December 31, 2019. If the testing time is changed, the performance of the models may vary.

For the development of accurate forecasting models, hyperparameter optimization has become an increasingly essential issue. We tried several combinations in terms of the number of layers, activation functions, optimizers, number of epochs and batch size. Selecting the right set of hyperparameters has a significant impact on performance. In this article, we found the best parameters via brute force. However, better and automatic techniques for hyperparameter optimization are indeed needed in this regard.

In light of the discussion presented in this section, we must reiterate that stock prediction is a challenging task. We believe that deep learning methods possess tremendous capabilities in this regard. The work presented in the article tried to predict the trend in a stock based on its behavior in a particular sector. To the best of our knowledge, this article is the first wherein we apply these ideas to analyze the influence of a sector in stock prediction using 3D-ensemble CNN. Future work can use the ideas presented here, and thus engineer a model producing even higher returns.

3.5 CONCLUSIONS

This paper presented a 3D CNN and a 3D-ensemble CNN-based framework to classify directional trends in a stock's price. The objective of the article was to present a framework that could forecast the trend in stocks while considering a particular sector as a whole. This idea is in contrast to existing literature where work has tried to predict the trend in a single company. The trend classification problem was formulated as an image classification problem, especially 3D image classification. We further categorized the trend into three classes: Buy, Sell, and Hold. To do that, we constructed multiple technical indicators and augmented them in an image. In total sixty seven indicators were experimented with. Out of which, the best fifteen indicators were selected via hierarchical clustering. To test the feasibility of the work, experimentation was performed with real datasets. Particularly, data of forty-five different stocks was taken from the National Stock Exchange. With extensive numerical investigation, we found that the proposed work achieved up to 35% returns in some cases, with the average being 9.19%. Furthermore, the work presented in the article was also able to outperform other similar methods in literature. The investigation proved that the notion of grouping companies together and doing a sector wise prediction could serve as a potential benchmark in stock trend classification. Moreover, we also tried to show that though the idea of sector wise classification is new, it nevertheless, could be one of the exciting paradigms in stock trend classification.

CHAPTER 4

MULTIVARIATE AND ONLINE PREDICTION USING KERNEL ADAPTIVE FILTERING

4.1 INTRODUCTION

After experimenting in the previous chapter, we observed that DL based models need a lot of computational resources during training. As a result, we depart from the current literature and present an online kernel adaptive filtering-based method to stock price prediction. We experiment with ten different KAF algorithms to analyze stocks' performance and show the efficacy of the work. We focus our attention on financial time series prediction and its application to stock price prediction.

The main contributions of the chapter include:

- We propose a novel online learning-driven KAF approach for stock price prediction.
- We experiment with ten different algorithms belonging to the KAF class. A large-scale investigation of this magnitude so far eludes literature.
- Stock prediction using existing online methods requires a lot of computation time. The main aims to present a general framework wherein the price prediction can be made in a significantly less amount of time.
- The data is collected at multiple time windows, i.e., one day, sixty-minutes, thirty-minutes, twenty-five minutes, twenty-minutes, fifteen-minutes, ten-minutes, five-minutes, and one-minute. The proposed idea is applied to each of these time windows to try and find the best window for stock price prediction.
- The main objective is to predict the close and mid price of a stock. We look at two situations in which the mid-price is measured as (high+low)/2 and (open+close)/2, respectively. To do that, we apply ten different KAF-based algorithms and present a comprehensive discussion detailing every aspect of the analysis. With numerical testing performed on all fifty stocks of the main index (Nifty-50), we show the work's efficacy in this chapter.
- Lastly, we also try to show that although the KAF class of algorithms is new in the arena of stock prediction, they nevertheless are a practically viable candidate.

This chapter has been organized into various sections. In Section 4.2, proposed methodology is explained that will help to achieve the research objectives and Section 4.2.2 elucidates the details of problem statement. Section 4.3 includes the details of the experimental setup and evaluation criterion. Section 4.4 and Section 4.5 discusses the results for close price and mid price, respectively. Finally, the conclusion are summarized in Section 4.6.

4.2 METHODOLOGY

In this section, we discussed different KAF-based techniques. Further, we use online prediction methods. In this regard, KAFs work by self-tuning, where the input-output mapping is formulated according to an optimization criterion usually determined by the error signal. There are two types of adaptive filters: linear and non-linear. In linear filters, the traditional system follows a supervised technique and depends upon error-correction learning. The filter adaptively adjusts weights, $\omega(i-1)$, where *i* denotes the discrete-time interval. Here, the input signal v_i is mapped to an actual response t_i . Correspondingly, an error is denoted by e_i . The error signal adjusts weights by incremental value denoted by $\Delta \omega_i$. At the next iteration, ω_i becomes the current value of the weight to be updated. This process is continuously repeated until the filter reaches convergence; this generally occurs when the weight adjustment is small enough. Linear adaptive filters do not give satisfactory performance for the non-linear system due to the results varied in a non-intuitive manner. In real-world problems, where data patterns are more complex, classes may not be separated easily by hyperplanes. Consequently, we have to look to non-linear methods. In this paradigm, data is projected into high-dimensional linear feature space, and prediction is done in this high dimensional space. Comparing with other existing techniques for regression and classification, KAF has the following advantages:

- KAFs are universal approximators.
- KAFs handle the complexity issues in terms of computation & memory. Moreover, they follow the property of no local minima.
- KAFs follow the idea of online learning and handle non-stationary conditions well.

It was discussed that non-linear adaptive techniques are well suited for real-world prob-

lems. In this regard, kernel methods transform data into a set of points in the RKHS (Reproducing Kernel Hilbert Space). The main idea of KAF can be summarized as the transformation of input data into a high dimensional feature space *G*, via Mercer kernel. For this, the problem can be solved via inner products. There is no need to do expensive computations in high dimensional space, owing to the famous "kernel trick". Considering KAF, suppose we have an input-output mapping as $g: \mathcal{V} \to R$, based on a well known sequence $((v_1, t_1), (v_2, t_2), \dots, (v_i, t_i))$. where, v_i , is the system input with $i = 1, \dots, n$ and t_i is equivalent to desired response. The goal is to estimate *g* from data. In KAFs, generally, the computation involves the use of a kernel. An example of a kernel is given in Equation (4.1):

$$\kappa < v, v' >= exp \frac{(||v - v'||^2)}{\sigma^2}.$$
 (4.1)

Here, κ denotes the kernel and σ denotes the kernel width.

4.2.1 DISCUSSION ON KAF ALGORITHMS

In this subsection, we discuss some of the most popular methods in KAF. For reasons of brevity, we keep the discussion short.

4.2.1.1 LEAST MEAN SQUARE ALGORITHM (LMS)

According to (47), the main aim of LMS algorithm is to minimize the following empirical risk function:

$$\min_{\boldsymbol{\omega}} R_{emp}[\boldsymbol{\omega} \in H_1, R^L] = \sum_{i=1}^N (t_i - \boldsymbol{\omega}(v_i))^2$$
(4.2)

Applying stochastic gradient descent (SGD), Equation (4.2) can be represented as:

 $\omega_0 = 0$

$$e_{i} = t_{i} - \omega_{(i-1)}v_{i}$$

$$\omega_{i} = \omega_{(i-1)} + \eta e_{i}v_{i}$$
(4.3)

where, η is step size and e_i is known as prior error. The weight-update equation results in the following form:

$$\omega_i = \eta \sum_{i=1}^N e_i v_i \tag{4.4}$$

Representing the idea in terms of inner product, we get:

$$t = \omega_i(v) = \eta \sum_{i=1}^n e_i < v_i, v >$$
(4.5)

$$e_i = t_i - \eta \sum_{i=1}^{n-1} e_i < v_i, v >$$
(4.6)

4.2.1.2 KERNEL LEAST MEAN SQUARE ALGORITHM (KLMS)

KLMS (52) is an extension of LMS algorithm, the main difference is input v_i is transformed to $\Psi(v_i)$ in the high dimensional space RKHS. Applying LMS algorithm at new sequences $\{\Psi(i), t_i\}$, we get:

$$\omega_0 = 0$$

$$e_i = t_i - \omega_{(i-1)}^T \Psi(i)$$

$$\omega_i = \omega_{(i-1)} + \eta e_i \Psi(i)$$
(4.7)

where, e_i is the prediction error, ω_i is the weight vector in G, η is the step size.

Using the kernel trick, KLMS now be written as:-

$$g_{0} = 0$$

$$e_{i} = t_{i} - g_{i-1}(v_{i})$$

$$g_{i} = g_{i-1} + \eta e_{i}\kappa < v_{i}, . > .$$
(4.8)

KLMS assigns new unit for every input v_i as the center with ηe_i as its coefficient. Following the radial basis function (RBF), the algorithms represented as follows:

$$g_i = \sum_{j=1}^{i} b_j(i) \kappa < v_j, .>$$
(4.9)

4.2.1.3 KERNEL RECURSIVE LEAST SQUARE ALGORITHM (KRLS)

According to (52), in KRLS, the objective function is complemented via a regularization parameter. This can be represented as follows:

$$\min_{\boldsymbol{\omega}} \sum_{j=1}^{i} |t(j) - \boldsymbol{\omega}^{T} \Psi(j)|^{2} + \Lambda ||\boldsymbol{\omega}||^{2}$$
(4.10)

where, Λ stand for regularization vector.

It is shown that $\omega_i = \psi(i)b(i)$, where $b(i) = [\Lambda I + L(i)]^{-1}t_i$, here $t_i = [t_1, t_2, t_3, ..., t_i]^T$, $L(i) = \psi(i)^T \psi(i)$, and $\psi(i) = [\Psi(1), \Psi(2)\Psi(3)...\Psi(i)]$.

Complementing the above equation with RBF, we get

$$g_i = \sum_{j=1}^{i} b_j(i) \kappa < v_j, .>$$
(4.11)

The whole idea here can now be summarized as :-

$$R(i-1) = (\Lambda I + L(i-1))^{-1}$$

$$O(i) = \Psi(i-1)^{T} \Psi(i)$$

$$E(i) = R(i-1)O(i)$$

$$U(i) = \Lambda + \kappa < v_{i}, v_{i} > -E(i)^{T}O(i)$$
(4.12)

Following the sequential property of KRLS we have:

$$g_0 = 0$$

 $e_i = t_i - g_{i-1}(v_i)$
(4.13)

$$g_{i} = g_{i-1} + U(i)^{-1} e_{i} \kappa < v_{i}, .> -\sum_{j=1}^{i-1} U(i)^{-1} e_{i} E_{j}(i) \kappa < v_{j}, .>$$
(4.14)

KRLS update all previous coefficient through $-U(i)^{-1} e_i$ E(i), whereas KLMS never updates previous coefficients. Here, $E_j(i)$ is the j^{th} component of E(i). The computational complexity of KRLS is $O(i^2)$.

4.2.1.4 KERNEL AFFINE PROJECTION ALGORITHMS (KAPA)

KAPA (226) derives the idea of KLMS while reducing boosting performance and gradient noise. In KAPA, we formulate with sequences $\{t_1, t_2\}$ and $\{\Psi(1), \Psi(2)\}$ to minimize the cost function and estimate with weight vector ω .

$$\min_{\omega} _{emp} |t - \omega^T \Psi(v)|^2$$
(4.15)

Using stochastic gradient descent, we replace covariance matrix and cross covariance vector by local approximation directly from the data. Hence, we get the following equations:

$$\boldsymbol{\omega}_{i} = \boldsymbol{\omega}_{(i-1)} + \boldsymbol{\eta} \, \boldsymbol{\psi}(i) [t(i) - \boldsymbol{\psi}(i)^{T} \, \boldsymbol{\omega}_{(i-1)}]$$
(4.16)

where, $\psi(i) = [\Psi(i - K + 1), ..., \Psi(i)]$ and, K is the observation & regressor.

4.2.1.5 QUANTIZED KERNEL LEAST MEAN SQUARE ALGORITHM (QKLMS)

QKLMS is a famous algorithm proposed in (53). It is an extension of KLMS algorithm to deal with the issue of data redundancy. Using quantization operator the core idea can be written as:

$$\omega_{0} = 0$$

$$e_{i} = t_{i} - \omega_{(i-1)}^{T} \Psi(i)$$

$$\omega_{i} = \omega_{(i-1)} + \eta e_{i} \mathscr{Q}[\Psi(i)] \qquad (4.17)$$

where, in feature space G, $\mathcal{Q}[.]$ denotes the quantization. The learning rule for QKLMS is:

$$g_{0} = 0$$

$$e_{i} = t_{i} - g_{i-1}(v_{i})$$

$$g_{i} = g_{i-1} + \eta e_{i} \kappa(Q[v_{i}], .).$$
(4.18)

QKLMS and KLMS have almost the same computational complexity. The only difference between the two algorithms is that QKLMS deal with the issue of data redundancy to locally update the coefficients of closest center.

In short, the central idea of QKLMS is given below:-

Initialization: Determine quantization size $\varepsilon_{\mathscr{V}} \ge 0$, step size $\eta > 0$, kernel parameter $\sigma > 0$. **Output:** The center set and coefficient vector: $D(1) = \{v_1\}, b_1 = [\eta t_1]$ **Conditions:** while $\{v_i, t_i\}(i > 1)$ is available

- Calculate the result of the adaptive filters as $y_i = \sum_{j=1}^{size(D(i-1))} b_j(i-1)\kappa(D_j(i-1), v_i)$
- Calculate the error between actual and desired output $e_i = t_i y_i$
- Calculate the distance between v_i and D(i-1) $distance(v_i, D(i-1)) = \min_{1 \le j \le size(D(i-1))} ||v_i - D_j(i-1)||$
- If the distance(v_i, D(i − 1)) ≤ ε_V then codebook does not require any changes: D(i) = D(i − 1). Quantize v_i to the closest code vector: b_{j*}(i) = b_{j*}(i − 1) + ηe_i

where, $j^* = argmin_{1 \le j \le size(D(i-1))} ||v_i - D_j(i-1)||$,

Otherwise update the codebook with new center and update center D(i) = {D(i-1), v_i} and b_i = [b(i-1), ηe_i].
end while

4.2.1.6 KERNEL NORMALIZED LEAST MEAN SQUARE ALGORITHM (KNLMS)

According to (210), KNLMS algorithm is used for dictionary designing with coherence criterion. Here we discuss KNLMS from the point of view of MKNLMS-CS (multi kernel normalized least mean square algorithm with coherence based sparsification).

Assume $\kappa_m \colon \mathscr{V} \times \mathscr{V} \to R$ where $m \in \mathscr{M} \colon \{1, 2, \dots, M\}$ is a set of M distinct kernel. Consider $\mathscr{J}_n^{cs} := j_1^{(n)}, j_2^{(n)}, \dots, j_{r_n}^{(n)} \subset \{0, 1, \dots, n-1\}$ to be the dictionary $\{\kappa_m(., v_j)\}_{m \in \mathscr{M}, i \in \mathscr{J}_n^{cs}}$.

Here, $r_n := |\mathscr{J}_n^{cs}|$ is the size of dictionary. The filter works as per the following set of rules:

$$\Psi_n^{cs}(v) = \sum_{m \in \mathscr{M}} \sum_{j \in \mathscr{J}_n^{cs}} h_{j,n}^{(m)} \kappa_m(v, v_j), v \in \mathscr{V}$$
(4.19)

where, $h_{j,n}^{(m)} \in R, m \in \mathcal{M}, j \in \mathcal{J}_n^{cs}$. The estimated error $\hat{t}_n := \Psi_n^{cs}(v_n)$ of t_n can be written as:-

$$\Psi_n^{cs}(v_n) = \sum_{j \in \mathscr{J}_n^{cs}} h_{j,n}^T \kappa_{j,n}$$
(4.20)

where,

$$\kappa_{j,n} := [\kappa_1(v_n, v_j), \kappa_2(v_n, v_j), \dots, \kappa_M(v_n, v_j)]^T \in \mathbb{R}^M$$
$$h_{j,n} := [h_{j,n}^{(1)}, h_{j,n}^{(2)}, h_{j,n}^{(3)}, \dots, h_{j,n}^{(M)}]^T \in \mathbb{R}^M$$

Let the initial dictionary be indicated as $\mathscr{J}_0^{cs} := \emptyset$. This makes H_0 to be an empty size matrix. Following algorithm, we only add a new point *n* into \mathscr{J}_n^{cs} if the following condition holds:

$$||\kappa||_{\max} := \max_{m \in \mathscr{M}} \max_{j \in \mathscr{J}_n^{cs}} |\kappa_m(v_n, v_j)|, \leq \Delta, n \in N$$
(4.21)

where, $\Delta > 0$ is the threshold. Let $\eta \in [0,2]$ denotes the step size and $\Lambda > 0$ the regularization parameter, The update rule is given below:

• If Equation (4.21) is satisfied $\mathscr{J}_{n+1}^{cs} := \mathscr{J}_n^{cs} \cup \{n\}$

$$H_{n+1} := \bar{H}_n + \eta \frac{t_n - \langle \bar{K}_n, \bar{H}_n \rangle}{||\bar{K}_n||^2 + \Lambda} \bar{K}_n$$
(4.22)

• If Equation (4.21) is not satisfied $j_{n+1}^{cs} := j_n^{cs}$

$$H_{n+1} := H_n + \eta \frac{t_n - \langle H_n, K_n \rangle}{||K_n||^2 + \Lambda} K_n$$
(4.23)

where, $\bar{H}_n := [H_n 0]$ and $\bar{K}_n := [K_n \bar{k}_n]$ with $\bar{k}_n := [\kappa_1(v_n, v_n), \kappa_2(v_n, v_n), \kappa_3(v_n, v_n), \dots, \kappa_M(v_n, v_n)]^T$ where $0 \in \mathbb{R}^M$, is the zero vector. For KNLMS, the value of M is 1.

4.2.1.7 PROBABILISTIC LEAST MEAN SQUARE ALGORITHM (PROB-LMS)

The probabilistic approach to the LMS filter is an efficient approximation method. It provides an adaptable step-size LMS algorithm together with a measure of uncertainty about the estimation. In addition it also preserves the linear complexity of the standard LMS. Some of the advantages of Probabilistic models is that 1) They force the designer to specify all the assumptions of the model, 2) They provide a clear separation between the model and the algorithm used to solve it, and 3) They usually provide some measure of uncertainty about the estimation. It is assumed observation models to be Gaussian with this distribution:

$$pr(t_k|\boldsymbol{\omega}_k) = \mathcal{N}(t_k; \boldsymbol{v}_k^T \boldsymbol{\omega}_k, \boldsymbol{\sigma}_n^2)$$
(4.24)

where ω_k = parameter vector, σ_n^2 = variance for observation noise. v_k = Regression vector

4.2.1.8 KERNEL MAXIMUM CROSSENTROPY CRITERION (KMCC)

The algorithm's main aim is to maximize crossentropy between desired t_i and actual output y_i (227). Using MCC criterion and SGD, the algorithm can be written as:

$$\omega_{0} = 0$$

$$\omega_{(i+1)} = \omega_{i} + \eta \frac{\partial \kappa_{\sigma}(t_{i}, \omega_{i}^{T} \Psi(v_{i}))}{\partial \omega_{i}}$$

$$= \omega_{i} + \eta [(exp \frac{(-e_{i}^{2})}{2\sigma^{2}})e_{i}\Psi(i)]$$
....
$$= \eta \sum_{i=1}^{n} [(exp \frac{(-e_{i}^{2})}{2\sigma^{2}})e_{i}\Psi(i)]$$
(4.25)
$$(4.26)$$

where, σ is the kernel width and η is the step size.

The complete prediction and error calculation can be summarized as:

$$y_i = \eta \sum_{i=1}^{n} [(exp \frac{(-e^2)}{2\sigma^2})e_i \kappa < v_i, v_n >]$$
(4.27)

$$e_i = t_i - y_i; \tag{4.28}$$

4.2.1.9 LEAKY KERNEL AFFINE PROJECTION ALGORITHM (LKAPA)

The LKAPA (226) is the extension of KAPA discussed in Section 4.2.1.4. According to Equation (4.16), weight updation is a difficult task in high dimensional space, Here Equation (4.16) is modified. This can be done as follows:

 $\omega_0 = 0$

$$\Psi(i)^{T} \omega_{(i-1)} = \left[\sum_{j=1}^{i-1} b_{j}(i-1)\kappa_{i-K+1,j,\dots,i}\right]^{T}$$
$$\sum_{j=1}^{i-1} b_{j}(i-1)\kappa_{i-1,j}, \sum_{j=1}^{i-1} b_{j}(i-1)\kappa_{i,j}\right]^{T},$$

$$e_i = t_i - \boldsymbol{\psi}(i)^T \boldsymbol{\omega}_{(i-1)}$$

$$\omega_i = \omega_{(i-1)} + \eta \, \psi(i) e_i \tag{4.29}$$

$$\sum_{j=1}^{i=1} b_j(i-1)\Psi(j) + \sum_{j=1}^K \eta e_j(i)\Psi(i-j+K)$$
(4.30)

where,

 $\kappa_{i,j} = \kappa(v(i), v(j))$

=

The weight vector is computed using the following criterion:

$$\boldsymbol{\omega}_i = \sum_{j=1}^i b_j(i) \Psi(i) \qquad \quad \forall_i \ge 0, \tag{4.31}$$

From the perspective of empirical risk minimization, we minimize the following objective function:

$$\min_{\boldsymbol{\omega}} e_{mp} |t - \boldsymbol{\omega}^T \Psi(v)|^2 + \Lambda ||\boldsymbol{\omega}||^2$$
(4.32)

Then we get:

$$\boldsymbol{\omega}_{i} = (1 - \Lambda \boldsymbol{\eta}) \; \boldsymbol{\omega}_{(i-1)} + \boldsymbol{\eta} \, \boldsymbol{\psi}(i) [t(i) - \boldsymbol{\psi}(i)^{T} \; \boldsymbol{\omega}_{(i-1)}] \tag{4.33}$$

where, $\Psi(i) = [\Psi(i - K + 1),, \Psi(i)]$

Finally coefficient $b_{\kappa}(i)$ is updated as:

$$b_{\kappa}(i) = \begin{cases} k = i, \quad \eta(t_i - \sum_{j=1}^{i-1} b_j(i-1)k_{i,j}) \\\\ for\{i - K + 1 \le k \le i - 1\} \\\\ (1 - \Lambda \eta)b_k(i-1) + \\\\ \eta\left(t(k) - \sum_{j=1}^{i-1} b_j(i-1)\kappa_{k,j}\right) \\\\ 1 \le k < i - K + 1 \quad (1 - \Lambda \eta)b_k(i-1) \end{cases}$$

4.2.1.10 NORMALIZED ONLINE REGULARIZED RISK MINIMIZATION ALGO-RITHM (NORMA)

NORMA (226), is one of the kernel-based version of LKAPA described in Section 4.2.1.9. It is also correlated with the KLMS algorithm summarized in Section 4.2.1.2. NORMA includes the regularization and non-linear functional approach. It allows to reject old values ones in a sliding window manner.

4.2.2 PROBLEM FORMULATION

In this subsection, we discuss the results of stock prediction using all the ten discussed algorithms. The purpose of stock prediction is to determine the future values of a stock depending upon the historical values. As discussed in the Introduction section, our main aim is to predict the close and mid price. To this end, we calculated the percentage change for close and mid price. Subsequently, we apply the idea of auto-regression of the order *m* to predict the future change in the stock price. An autoregressive (AR) model forecasts future behavior using data from the past. When there is a correlation between the values in a time series and the values that precede and succeed them. In such situations, AR models have shown tremendous potential. In context of the work presented here, the problem is formulated as:

$$(V_i) = \sum_{i=1}^m \omega_{(i-1)}(V_i)$$
(4.34)

 (V_i) is the actual price in the high dimensional space, ω is the weight vector. Since, we follow AR model, it is imperative to estimate the weight vector. To estimate the weight vector, KAF techniques discussed in the previous subsection are used. A sample of the formulation is shown in Table 4.1 and Table 4.2. In this table, we have shown problem formulation by considering the day-wise close and mid price. This type of procedure is followed commonly in multivariate time series prediction, e.g., (228) (229). It should be noted here that the procedure was followed for all the window sizes. Subsequently, the problem became: Autoregression based next percentage prediction. The actual price can then be computed from the percentage change easily. The overall framework followed in the chapter is shown in Figure 4.1 and Figure 4.2 and the

experiments were performed on the Nifty-50 dataset ¹.

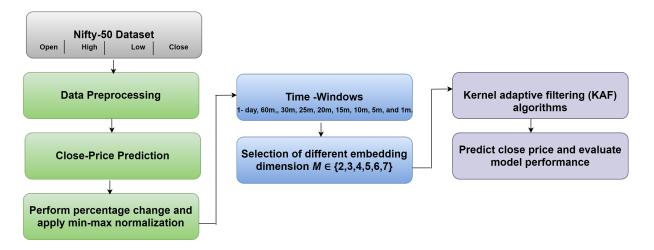


Figure 4.1: Proposed close price prediction framework.

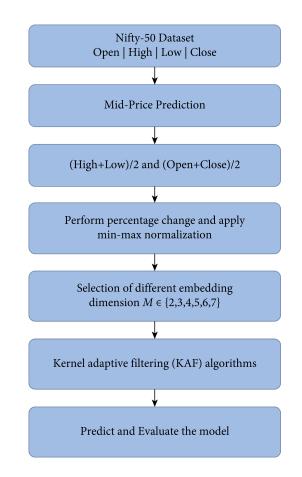


Figure 4.2: Proposed mid price prediction framework.

¹https://shorturl.at/lnvF2

Table 4.1: Time window of 1-day (Stock-Reliance). If we choose M=3, then Input=[{0.1685,-1.2431,-2.6372}] Output=[{-0.16194}]

Day	Actual Price	Change in Price
1 Day	1987.5	0.1685
2 Day	1990.85	-1.2431
3 Day	1966.1	-2.6372
4 Day	1914.25	-0.16194
5 Day	1911.15	1.17991
6 Day	1933.7	-1.8849
7 Day	1897.25	3.1519
8 Day	1957.05	-0.9325
9 Day	1938.8	1.1244
10 Day	1960.6	NA

Table 4.2: A one-day time frame (Stock-TITAN). If M=3 is selected, then Input=[{-0.7654, 0.4300, 2.0796}.] Output=[{-2.4566}]

Day	High Price	Low Price	(high+low)/2	Change in Price
1 Day	1573	1555.95	1564.475	-0.7654
2 Day	1567	1538	1552.5	0.4300
3 Day	1576.85	1541.5	1559.175	2.0796
4 Day	1621.35	1561.85	1591.6	-2.4566
5 Day	1570	1535	1552.5	-0.9082
6 Day	155.3	1521.5	1538.4	NA

4.3 **RESULTS**

In this section, we elucidate the details of the various experiments performed to evaluate the proposed method's effectiveness. The proposed method is evaluated on a different class of KAF algorithms and datasets. In this section, we have described the experimental details of the Nifty-50 dataset (230). Nifty-50 is the largest stock exchange in India according to the rate of

total and average daily turnover for equity shares. We collected the data of all stocks from 9:15 to 3:30. The original data was available for one-minute open, high, low, and close (OHLC) prices. From this granular data, we clubbed the OHLC quotes to get the data for other time windows. In particular, we created and pre-processed the dataset according to nine prediction windows (one day, sixty-minutes, thirty-minutes, twenty-five minutes, twenty-minutes, fifteen-minutes, ten-minutes, five-minutes and one-minute). Recall that we focused on predicting the percentage changes in close and mid price. To that end, we also normalized the data between the 0 to 1 range. Then ten distinct KAF algorithms were applied to the final pre-processed data for every stock. Finally, it's worth noting that the experimental findings obtained with the KAF algorithm on the Nifty-50 dataset demonstrate the work's superiority and could serve as a new benchmark for the field's future state-of-the-art.

4.3.1 DATASETS

During experiments, we selected different dataset, according to close and mid prices prediction.

- For close price dataset, we experiment with two different years. First, we try to predict stock prices for the year 2020 from January 01, 2020, to December 31, 2020. Second, we apply the same set of parameters on the most recent data (2021) January 01, 2021, and May 31, 2021 and try to show the efficacy of the work. Through experiments performed on these two different years, we have found the method proposed in the chapter outperforms similar methods in literature.
- For Mid-Price dataset, we look at two situations in which the mid-price is measured as (high+low)/2 and (open+close)/2, respectively. The main motivation for looking into mid-price was that mid-price time series is less noisy than close- price time series. We collected data between January 01, 2021, and May 31, 2021, to show the efficacy of the work.

4.3.2 EVALUATION CRITERION

To evaluate and compare the performance of various KAF algorithms, we use standard error evaluation metrics such as mean squared error (MSE), directional symmetry (DS), and mean absolute error (MAE). The metrics are elaborated in the following text-

Minimum Square Error (MSE)

MSE also known as mean squared deviation (MSD) which calculates the average squared difference between the actual and predicted observation.

$$\mathbf{MSE} = \sum_{i=1}^{n} \left(a_i - p_i \right)^2$$

$$(4.35)$$

Mean Absolute Error (MAE)

MAE calculates the average magnitude between actual and predicted observations in a set of predictions, without observing their directions, i.e. the average prediction error.

$$\mathbf{MAE} = \frac{1}{n} \sum_{i=1}^{n} |a_i - p_i|$$

$$(4.36)$$

Directional Symmetry (DS)

Directional symmetry in terms of time series analysis measures the model's performance to predict positive and negative trends from one time period to the next.

$$\mathbf{DS} = \frac{1}{n} \sum_{i=1}^{n} d_i \tag{4.37}$$

where,

$$d_i = \begin{cases} 0 & \text{otherwise} \\ 1 & (a_i - a_{i-1})(p_i - p_{i-1}) \ge 0 \end{cases}$$

where, n is the time-step, a_i represents the actual values, and p_i represents the predicted output. In the following procedure, we discuss upon the details to compute error values.

4.4 EXPERIMENTAL RESULTS ON CLOSE PRICE US-ING KERNEL ADAPTIVE FILTERING

The main objective is to predict the closing price of a stock. We apply ten different KAFbased algorithms and present a comprehensive discussion detailing every aspect of the analysis. Numerical testing was performed on all fifty stocks of the main index (Nifty-50).

4.4.1 PROCEDURE: ERROR COMPUTATION

- We worked with Nifty-50 firms with 2020 and 2021 datasets, as mentioned in Section 4.3.1. Moreover, it was also pointed out that we work with ten different algorithms. The parameter listed in Table 4.3 were tuned manually. In order to find the optimal values of the parameters, multiple experiments were performed.
- 2. To compute the error values for each stock and every algorithm, we formulated the problem as an autoregressive problem (see Section 4.2.2) and computed the error values for all 50 stocks. In total, we get 50X3 error values. One for MSE, MAE, and DS. Moreover, we pointed out that we have nine different prediction windows. Hence, error estimation was done for all stocks, all windows, and all ten algorithms.
- 3. Subsequently, for a particular algorithm, and for a single time window, we take the average of all 50-error metrics (one for every stock) to come up with the final number. The

number is presented in this section. This number shows the overall predictive capability of the model on all fifty stocks.

Table 4.3: Parameter description for close price using ten different KAF algorithms. σ = Kernel width , $\sigma_2 n$ = Variance of observation noise , $\sigma_2 d$ = Variance of filter weight diffusion, η = Step-size, ε = Regularization parameter, Λ = Tikhonov regularization , *tcoff*= Learning rate coefficient, τ = memory size (terms retained in truncation), mu0= Coherence criterion threshold, P= Memory length, nu= Approximate linear dependency (ALD) threshold.

Parameter	КАРА	KLMS	КМСС	KNLMS	KRLS	LKAPA	LMS	NORMA	PROB-LMS	QKLMS
(σ)	4.0	4.0	4.0	4.0	3.0	5.0		7.0		3.0
$(\boldsymbol{\eta})$	1.7	1.1	1.5	1.7		0.09		1.1		1.2
$(\boldsymbol{\varepsilon})$	1E-4			1E-2						0.3
(Λ)						1E-4		1E-2	0.4	
$(\sigma_2 n)$									2	
$(\sigma_2 d)$									6	
mu0	0.2			2			0.2			
(P)	20					20				
пи					1E-2					
τ								500		
tcoff								0.9		

4.4.2 PREDICTION, CONVERGENCE, AND RESIDUAL ANALYSIS

In this subsection, we analyze the performance of KAF algorithms for close price prediction. In this regard, the prediction graph for one stock (Reliance) with KRLS. Figure 4.3 shows the results for 2020 and 2021 datasets. It is visible from the figure that we are getting good results. It should be noted here that we have presented the result for one stock (Reliance) and one prediction window (Sixty-Minutes). Similar results were obtained for other companies in the dataset. It is also visible from the figure that although the prediction is not cent percent accurate, it is close. It, therefore, implies the superior performance of KAF algorithms in prediction. It should be noted here that although we are getting good results, there are always chances of overfitting. In this chapter, since we are using online learning, therefore, the architecture itself naturally minimize the chances of overfitting, but it is possible that the superior results might be

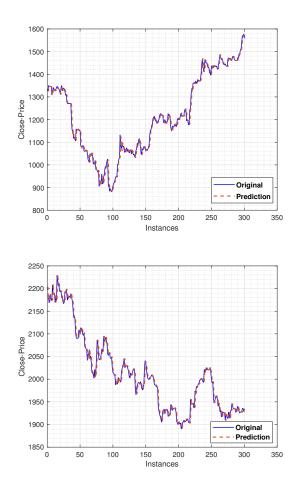


Figure 4.3: Prediction for one stock (Reliance) using KRLS: (a) 2020 dataset and (b) 2021 dataset.

due to overfitting. It is expected from any machine learning algorithm that it should converge as we train the model with more instances, in other words, the error as we progress through the training should decrease to an acceptable range. In this regard and in addition to presenting the results for prediction in Figure 4.3, we have also presented the result of convergence in Figure 4.4 for 2020 and 2021 dataset, respectively. Similar to the previous case, we have only plotted the result considering single stock (Reliance) and one prediction time window. The convergence graphs of the algorithm were plotted taking MSE as the error metrics. Figure 4.4 shows the error convergence graph for both the datasets and KRLS algorithm for the Reliance stock. In Figure 4.4, x-axis shows the number of instances and y-axis shows the MSE. It can be seen from Figure 4.4 that the algorithm reached convergence very quickly. In fact, the algorithm reached convergence at 1000th data point. Convergence is very important in KAF as it shows the ability of the algorithm to adapt itself and learn from the data quickly. Though, there are minor fluctuation in the end, it nevertheless is acceptable as there will always be minor changes

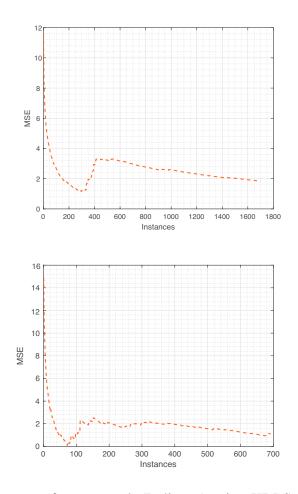


Figure 4.4: Error convergence for one stock (Reliance) using KRLS: (a) 2020 dataset and (b) 2021 dataset.

in the new data. To complement the prediction results, we have also presented the distribution of error residuals in Figure 4.5 for 2020 and 2021 dataset respectively. As visible from the figure, the residuals are following a normal distribution. This type of behavior is excellent as there are very few outliers. Moreover, the overall variance of the residuals is also less, showing the excellent prediction potential of the algorithm.

4.4.3 COMPREHENSIVE EVALUATION OF KAF ALGORITHMS

In contrast to batch learning techniques, which generates the best predictor by learning on the full training dataset at once, we employ an online learning concept in which data becomes available in a sequential order (sample by sample training) and is used to update the best predictor for future data at each step. As we have used ten different algorithms, it is logical to compare

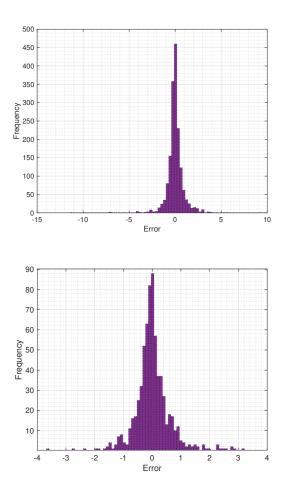


Figure 4.5: Error residuals for one stock (Reliance) using KRLS: (a) 2020 dataset and (b) 2021 dataset.

the performance of all algorithms. In this regard, we have shown the result in two different datasets. First, we attempt to forecast stock prices for 2020. Second, we use the same set of parameters on the most recent data (2021) to demonstrate the work's efficacy. To evaluate the performance of KAF-based methods, we try to experiment with different values of M (the embedding dimension). We vary the underlying dimensions from 2 to 7 with a step size of 1 i.e. $M \in \{2, 3, 4, 5, 6, 7\}$. With this setup, the results are presented in Table 4.7 and Table 4.8. It is visible from the table that once again, KRLS performed well in terms of error minimization. The best number for the embedding dimension is two when we consider MSE and MAE. However, when it came to DS, the numbers and the algorithms are different because a market trend is a term used to describe how a market moves over time. A trend can generally move upward or downward. For instance, considering daily data (1 Day in the table), the best performing algorithm is LKAPA with embedding dimension (M)=5. In fact, for this metric (DS), we see much conflict in terms of the best algorithm. Nevertheless, the experimentation revealed the

superiority of KRLS, PROB-LMS, LKAPA.

Method	MSE	RMSE	Execution Time (s)
Gao et al. (231)	0.51917	0.7205	400.39
Moghar et al. (232)	0.51800	0.7197	1265.11
Nikou et al. (21)	0.51838	0.7199	5006.19
Proposed Method	0.0034	0.0583	5.234

Table 4.4: Comparison of the proposed work with other state-of-the-art stock prediction method for 60-minute time window (2020 dataset) January 01, 2020 to December 31, 2020

4.4.4 COMPARISON WITH OTHER STATE-OF-THE-ART METHODS

We compared our result with other learning methods such as (231), (232), and (21), among other learning approaches. The deep learning (DL) algorithms were taught and assessed over a period of 25 epochs utilizing an 80:20 split. The amount of time taken to train and make prediction was recorded. Based on the architecture details and hyper-parameters settings provided in the relevant articles, the DL-based methods (231), (232), and (21) stock were re-implemented. All of the techniques were trained on the Nifty-50 dataset. We chose fifty equities for the sixty-minute time periods to maintain uniformity across different ways for experimentation. In terms of MSE, RMSE, and execution time, all of the approaches were then compared to the suggested KAF method (KRLS). For the 2020 and 2021 datasets, Table 4.4 and Table 4.5 show that the proposed approach outperforms previous stock prediction methods in the literature.

We must point it out here that since all the models belong to the same category of kernel adaptive filtering, the complexity of all the models is almost similar. For neural networks used in the article, we collect the architecture from their respective papers (231), (232), and (21). It should be noted here that KAF is also analogous to the neural network architecture with a single layer. Further, even though it has a single layer, but it is giving good results.

Method	MSE	RMSE	Execution Time (s)
Gao et al. (231)	0.70202	0.8378	362.67
Moghar et al. (232)	0.6975	0.8351	1082.90
Nikou et al. (21)	0.70232	0.8380	2250.87
Proposed Method	0.0081	0.09	4.256

Table 4.5: Comparison of the proposed work with other state-of-the-art stock prediction method for 60-minute time window (2021 dataset) January 01, 2021 to May 31, 2021

4.4.5 EXPERIMENTATION WITH DICTIONARY SIZE

In addition to the experiment conducted in the previous section, we have also experimented with the dictionary size of KAF algorithms. The result for this experiment is presented in Table 4.6. As visible, increasing the dictionary size decreases the performance of the system. Moreover, increasing the dictionary size also increased the execution time. It should be noted here that the execution time for predicting the next closing price for a single stock with dictionary size 500 is 0.675 seconds. This figure (0.675 seconds) clearly shows the applicability of the KAF class of algorithms in high-frequency trading, where latency is a key factor.

Table 4.6: Effect of Dictionary size. Algorithm chosen KMCC (60 minutes, 2021 dataset).

Dictionary size	MSE	MAE	DS	Execution time (seconds)
500 dictionary size	0.0202	0.687	0.511	0.675
1000 dictionary size	0.0193	0.674	0.497	0.696
5000 dictionary size	0.0193	0.674	0.4971	0.702

4.4.6 IMPORTANT NOTE: ERROR MINIMIZATION AND PROFITABIL-ITY

From Tables 4.7 and 4.8, we can see that KRLS performed well in minimizing error. Moreover, the lowest error (MSE) that we get is in order of 10^{-4} . It should be noted here that we got this

error for the time window of 1 minute. In this regard, it is common sense that if we minimize the error, we can get close to the actual values. Indeed true. However, considering the time window of one minute, there is an issue. In this interval, the fluctuation in the price is low. This means that minimizing error won't result in too much profit. In other words, the volatility in one minute is less. Hence, predictions are very close. However, the chances of taking a position and getting profit in a low volatile environment is also very less. Therefore, one has to maintain a balance between error minimization and profitability.

Table 4.7: Best Embedding Dimension for all time windows according to evaluation metrics MSE, MAE and DS. (2020)

Time Window	MSE	Best Embedding Dimensions	Algorithms
1-Day	0.0143	2	KRLS
60 Minutes	0.0034	2	KRLS
30 Minutes	0.0020	2	KRLS
25 Minutes	0.0018	2	KRLS
20 Minutes	0.0015	2	KRLS
15 Minutes	0.0012	2	KRLS
10 Minutes	0.0008	2	KRLS
5 Minutes	0.0004	2	KRLS
1 Minute	0.00010	2	KRLS
Time Window	MAE	Best Embedding Dimensions	Algorithms
1-Day	2.1324	2	KRLS
60 Minutes	0.6602	2	KRLS
30 Minutes	0.4666	2	KRLS
25 Minutes	0.4317	2	KRLS
20 Minutes	0.3812	2	KRLS
15 Minutes	0.3295	2	KRLS
10 Minutes	0.2667	2	KRLS
5 Minutes	0.1886	2	KRLS
1 Minute	0.0852	2	KRLS
Time Window	DS	Best Embedding Dimensions	Algorithms
1-Day	0.5013	5	LKAPA
60 Minutes	0.4919	5	KRLS
30 Minutes	0.4913	5	KRLS
25 Minutes	0.4925	3	KRLS
20 Minutes	0.4881	2	PROB-LMS
15 Minutes	0.4893	7	QKLMS
10 Minutes	0.4902	2	KRLS
5 Minutes	0.4910	2	PROB-LMS
1 Minute	0.4715	2	KRLS

Best Embedding Dimensions Algorithms Time Window MSE 0.0342 2 KRLS 1-Day 60 Minutes 2 0.0081 KRLS 2 30 Minutes 0.0047KRLS 25 Minutes 0.0042 2 KRLS 2 20 Minutes 0.0033 KRLS 2 15 Minutes 0.0028 KRLS 2 10 Minutes 0.0020 KRLS 2 KRLS 5 Minutes 0.0011 1 Minute 0.0003 2 KRLS Time Window MAE Best Embedding Dimensions Algorithms]

Table 4.8: Best Embedding Dimension for all time windows according to evaluation metrics MSE, MAE and DS. (2021)

1-Day	1.6901	2	KRLS		
60 Minutes	0.5480	2	KRLS		
30 Minutes	0.3961	2	KRLS		
25 Minutes	0.3671	2	KRLS		
20 Minutes	0.3238	2	KRLS		
15 Minutes	0.2803	2	KRLS		
10 Minutes	0.2259	2	KRLS		
5 Minutes	0.1601	2	KRLS		
1 Minute	0.0729	2	KRLS		
Time Window	DS	Best Embedding Dimensions	Algorithms		
1-Day	0.4870	4	NORMA		
60 Minutes	0.4930	4	KNLMS		
30 Minutes	0.4849	4	KRLS		
25 Minutes	0.4878	6	LKAPA		
20 Minutes	0.4891	7	LKAPA		
15 Minutes	0.4881	2	PROB-LMS		
10 Minutes	0.4891	2	PROB-LMS		
5 Minutes	0.4846	2	PROB-LMS		
1 Minute	0.4751	2	KRLS		
86					

4.5 EXPERIMENTAL RESULTS ON MID PRICE USING KERNEL ADAPTIVE FILTERING

The primary motivation for looking into mid price was that mid price time series are less noisy than close price time series. A novel KAF-based online method for forecasting a stock's mid price is introduced. We look at two situations where the mid price is measured as (high+low)/2 and (open+close)/2, respectively.

4.5.1 CALCULATING THE EVALUATION METRICS WITH NIFTY-50

- The parameter listed in Table 4.9 were tuned manually. The parameter description for ten different algorithms are presented in Table 4.9. These values were found after multiple rounds of experimentation.
- For the error values, we applied the methods to all stocks and tried to quantify the predictive performance via the metrics discussed in this section. In total, we get 50×3 (one for each stock) error values for MSE, MAE, and DS, respectively.
- Then, for each of the 50 stocks, error estimation was performed using nine different prediction windows for ten different KAF algorithms.
- Finally, we used the average of all fifty-error metrics for a single time window and a single stock to reach at the final value, which is presented in Table 4.13 and Table 4.14. On all 50 stocks, the provided number represents the models' overall predictive capacity.

4.5.2 PREDICTION, CONVERGENCE, AND RESIDUAL ANALYSIS

In this subsection, we examined prediction, converge and residual analysis with the help of KAF algorithms. Regarding this, we have shown the prediction graphs with the KAPA algo-

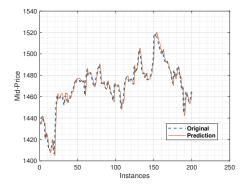


Figure 4.6: Prediction for one stock (TI-TAN) using KAPA (high+low)/2

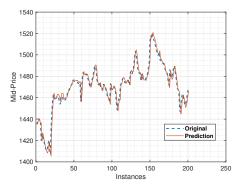


Figure 4.7: Prediction for one stock (TI-TAN) using KAPA (open+close)/2

Parameter	KAPA	KLMS	KMCC	KNLMS	FBQKLMS	LKAPA	LMS	NORMA	PROB-	QKLMS
									LMS	
(σ)	5.0	7.0	4.0	7.0	5	6	-	3	-	4
$(\boldsymbol{\eta})$	1.5	1.7	1.7	1.7	0.2	0.03	-	-	-	0.2
$(\boldsymbol{\varepsilon})$	1E-4	-	-	1E-4	0.4	-	-	1.5	-	0.5
(β)	-	-	-	-	0.85	-	-	-	-	-
(Λ)	-	-	-	-	-	1E-2	-	1E-2	0.4	-
$(\sigma_2 n)$	-	-	-	-	-	-	-	-	2	-
$(\sigma_2 d)$	-	-	-	-	-	-	-	-	3	-
mu0	0.2	-	-	2	-	-	0.2	-	-	-
(P)	20	-	-	-	-	20	-	-	-	-
τ	-	-	-	-	-	-	-	5000	-	-
tcoff	-	-	-	-	-	-	-	4	-	-

Table 4.9: Parameter description of KAF techniques for NSE-50 data set for mid-price.

rithm (discussed in Section 4.2.1.4) for one stock (TITAN). Figure 4.6 shows the results for (high+low)/2, while Figure 4.7 shows the results for (open+close)/2. The predictive curve suits well against the original curve, as can be seen from the prediction graphs. It's worth noting that we've only given results for one prediction window (thirty-minutes) with one stock (TI-TAN). However, we must note out that other stocks in the dataset produced similar result. The prediction graphs clearly show that the predictions are not exact, although they are close. To be precise, the numbers for MSE and MAE are presented in Table 4.13 and 4.14. We must point out here that getting accurate value in financial time series forecasting is tough. The goal has always been to get close enough values. The result that we achieved, therefore, shows the

good predictive capability of the work. Figure 4.8 and Figure 4.9 shows the convergence graph for mid-price for (high+low)/2, (open+close)/2 respectively. We've provided the results using the KAPA algorithm with only one prediction window (thirty minutes) and one stock (TITAN), similar to the prior scenario. The algorithm converges quickly, as evidenced by the graphs, at the 1000th data point. We can see in KAF algorithms, capacity to adapt and converge quickly. One more important point to note from the convergence graphs is that although there is some fluctuation in the graphs, it is nevertheless acceptable. This is because there will be noise in the new data and minor changes are inevitable. In addition to the results discussed so far, we have complemented the analysis by presenting the distribution of error residuals in Figure 4.10 and Figure 4.11. It can be seen from figures that residuals follow a normal distribution. Moreover, the outliers are also less. Furthermore, the residual's variance is low, demonstrating the KAF algorithm's superior prediction capability and potential in predicting the next immediate, mid-priced occurrence. Directional symmetry is used to determine the continuity of actual and expected prices in terms of stock movement. It is a measure for determining a model's ability to predict a stock's direction. We examined the ten different algorithms mentioned in Section 3 to better understand the actions of a stock's movement. The experiment revealed that using KNLMS, we have a 66% percent chance of accurately predicting the next up or down movement. This is shown in Table 4.14. The best result is obtained at the window of ten-minutes, and the worst result is obtained at the one-minute window. From the table, it is also visible that there is a big difference in the number obtained for the one-minute window and that for the rest of the windows. This is expected as there is a lot of noise in a minute, which indeed affects prediction. It should be noted here that literature often ignores looking at these different time windows. Work mostly focuses on predicting daily prices, e.g., (233), (234). We discovered the perfect balance by playing with various time windows. Furthermore, when trading, it is recommended to strike a balance between error minimization and directional symmetry.

4.5.3 COMPARATIVE EVALUATION OF KAF ALGORITHMS

Since we have used ten algorithms in our experimentation; therefore, it becomes essential to compare their performance. In this context, we present the topic in two separate situations. First, we analyze the results considering mid-price as: (high+low)/2 to find the best algorithm.

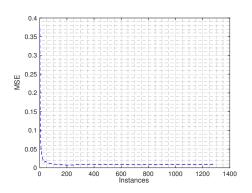


Figure 4.8: Error convergence for one stock (TITAN) using KAPA (high+low)/2

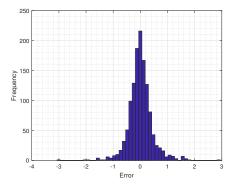


Figure 4.10: Error residuals one stock (TI-TAN) using KAPA (high+low)/2

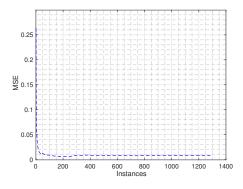


Figure 4.9: Error convergence for one stock (TITAN) using KAPA (open+close)/2

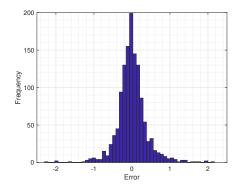


Figure 4.11: Error residuals for one stock (TITAN) using KAPA (open+close)/2

In the next scenario, we tried mid-price using (open+close)/2. Table 4.13 and Table 4.14 show the outcome of this experiment. In terms of MSE and MAE, the Tables shows that KAPA outperforms other algorithms. When it came to directional symmetry, we can see that there is a contradiction. In directional symmetry, we see a conflict. Here, NORMA and KNLMS gives the best performance.

4.5.4 COMPARISON WITH METHODS OF A SIMILAR KIND

We have also compared the result other existing techniques such as (231), (232), and (21). These are some of the most recent deep learning (DL) based algorithms for predicting stock prices. It should be noted here that these methods were trained and tested using 80:20 splits for 25

Table 4.10: The proposed research is compared to different state-of-the-art stock prediction approaches.

Methods	MSE	Execution Time (s)
Gao et al. (231)	0.6831	256.16
Moghar et al. (232)	0.6817	945.09
Nikou et al. (21)	0.6824	1770.40
Proposed method (KAPA)	0.0091	2.132

epochs. The time it took to train and make a prediction was maintained record of. Specifically, these methods (231), (232), and (21) were reimplemented based on the architecture details and hyper-parameters setting found in the respective papers. The Nifty-50 dataset was used to train all of the methods. To ensure consistency across different methods for experimentation, we use sixty-minute time periods, for fifty stocks. All of the methods' results were then compared to the proposed method.Table 4.10 contains the results. The table's data clearly demonstrate the proposed method's superiority over a number of other ways.

Table 4.11: Effect of different kernel methods on time window of thirty-minutes (stock-TITAN) using KAPA for (high+low)/2

	(High-	+Low)/2	(Open+Close)/2			
Kernel Function	MSE	MAE	DS	MSE	MAE	DS
RBF kernel[gauss]	0.008120	0.29857	0.53421	0.00827	0.26047	0.53188
Anisotropic RBF kernel [gauss-	0.01919	0.40617	0.51555	0.02535	0.34868	0.57931
anis]						
Laplace kernel [Laplace]	0.008128	0.29872	0.53343	0.00828	0.26061	0.53110

4.5.5 EFFECT OF DIFFERENT KERNELS

In Table 4.11, we can see the effect of different kernel methods. For this test, we used a thirtyminute time window (Stock-TITAN) with the algorithm KAPA and analyzed the best performance of RBF kernel in terms of MSE that' why we are chosen RBF kernel (gauss) for each algorithm.

4.5.6 EXPERIMENTATION WITH DICTIONARY SIZE

We also conducted experiments with various dictionary sizes. The result for this test is shown in Table 4.12. For this test, we used a thirty-minute time window with the algorithm KMCC. It is visible from Table 4.12 that increasing the dictionary size leads to an improvement in the system's performance. It should be noted here that when the size is 1000, the performance has fallen. The reason for this behaviour could be the erratic behaviour of the stock, the presence of noise, or too much irrelevant data. The exact reason is unknown. However, it's worth noting that with a dictionary size of 500, for forecasting a single stock, execution time is 0.82 seconds. This low number clearly shows the advantage one can achieve in high-frequency trading.

Dictionary Size	MSE	MAE	DS	Execution Time (s)
500	0.0148	0.368	0.5458	0.8202
1000	0.0323	0.5302	0.6143	0.8533
5000	0.01137	0.3080	0.6446	1.314

Table 4.12: The Influence of Dictionary Size. KMCC was chosen as the algorithm (30-minutes).

4.5.7 IMPORTANT NOTE: PROFITABILITY AND ERROR MINIMIZA-TION

We obtain an MSE of 10^{-4} as the lowest error. We can observe from Tables 4.13 and 4.14 that KAPA gives best results in terms of MSE and MAE. It's important to note that in the one-minute time window, we reached a minimum error value. From the Tables, we can also see that going down the column (for MSE and MAE only), the results are improving with the one-minute time window giving the best figures. However, because the time window is one minute, the volatility is low enough that decreasing error won't result in too much benefit. Moreover, there is too much noise while trading at one minute window. To look at it another way, one-minute

Time Window	MSE	Best algo- rithms out of ten dis- cussed (Ac- cording to MSE)	MAE	Best algo- rithms out of ten dis- cussed (Ac- cording to MAE)	DS	Best algo- rithms out of ten dis- cussed (Ac- cording to DS)
1-Day	0.0306	KAPA	1.4129	КАРА	0.5378	NORMA
60 Minutes	0.0091	KAPA	0.5096	КАРА	0.5592	NORMA
30 Minutes	0.0053	КАРА	0.3595	КАРА	0.5558	PROB-
						LMS
25 Minutes	0.0047	KAPA	0.3314	KAPA	0.5578	NORMA
20 Minutes	0.0038	KAPA	0.2909	KAPA	0.5550	NORMA
15 Minutes	0.0030	KAPA	0.2534	КАРА	0.5556	NORMA
10 Minutes	0.0021	KAPA	0.2019	КАРА	0.5472	NORMA
5 Minutes	0.0012	КАРА	0.1447	КАРА	0.5342	PROB-
						LMS
1 Minute	0.00027	KAPA	0.0590	KAPA	0.5496	NORMA

Table 4.13: Result in terms of MSE, MAE and DS for mid price (high+low)/2.

volatility is lower, resulting in very close predictions. However, in a low-volatility environment, the chances of taking a position and making a highly profitable trade are also low.

Time Win- dow	MSE	Best algo- rithms out of ten dis- cussed (Ac- cording to MSE)	MAE	Best algo- rithms out of ten dis- cussed (Ac- cording to MAE)	DS	Best algo- rithms out of ten dis- cussed (Ac- cording to DS)
1-Day	0.0324	KAPA	1.2989	KAPA	0.5970	NORMA
60 Minutes	0.0090	КАРА	0.4541	КАРА	0.6367	NORMA
30 Minutes	0.0048	КАРА	0.3168	КАРА	0.6547	NORMA
25 Minutes	0.0041	КАРА	0.2903	КАРА	0.6592	NORMA
20 Minutes	0.0033	КАРА	0.2543	КАРА	0.6580	NORMA
15 Minutes	0.0026	КАРА	0.2198	KAPA	0.6652	KNLMS
10 Minutes	0.0017	КАРА	0.1747	KAPA	0.6666	KNLMS
5 Minutes	0.0009	КАРА	0.1224	KAPA	0.6628	KNLMS
1 Minute	0.00025	КАРА	0.0552	КАРА	0.6018	KNLMS

Table 4.14: Result in terms of MSE, MAE and DS for mid price (open+close)/2.

4.6 CONCLUSIONS

This chapter focuses on predicting a stock's mid-price. Predicting a financial non-stationary time series is an open fundamental and a non-trivial problem of literature. To address this, we proposed a framework based on online learning driven KAF algorithms. In the proposed work, ten different KAF algorithms were evaluated and analyzed on Indian National Stock Exchange (Nifty-50). In contrast to the existing methods, experiments are performed on nine different time windows. This was done keeping in mind the applicability of the method in Intraday and swing trading. Previous studies in literature often underestimated the importance of intraday time windows, we therefore, tried to bridge this gap through the work presented here. The experimental results show the superiority and predictive capabilities of the work. The KAF class of algorithms was also discovered to be not only efficiently work in execution

time, but also provide best results of error minimization, demonstrating their importance in high-frequency trading. The goal of the research was to propose a KAF-based method for the prediction of stock's close and mid-price. The empirical results on Nifty-50 dataset show that the proposed method achieved superior performance over existing stock prediction methods. It's worth noting that every KAF-based algorithm is hyperparameter-sensitive. As a result, in the future, we will experiment with various hyper-parameter optimization approaches in order to enhance the framework's predictive capabilities.

CHAPTER 5

STATISTICAL FEATURE EXPANSION DRIVEN KERNEL ADAPTIVE FILTERING FOR MID-PRICE PREDICTION

5.1 INTRODUCTION

Previous research has shown that feature expansion can help improve the predictive performance of stock price prediction methods (235). However, as expected, optimum feature selection is crucial to improving the outcomes. Feature expansion is a technique for identifying important data characteristics that a model can employ to boost its performance. It is understandable that a terrible feature will immediately impact the model, regardless of data or any underlying architecture. Nevertheless, we anticipate that existing and well-tested prediction models will benefit by extracting new characteristics from data and merging them with different domain variables. To that end, we propose using different statistical, frequency, and time-based features. Instead of analyzing technical indicators, which is the most preferred way of literature trying to predict stock prices so far (72), (68), we argue that feature extension as well as fine-tuning the selected features based on the statistical properties of the financial time series could work well in practise. We look at previous research, identify limitations, and present a potential architecture via feature expansion. We would also argue that by expanding new characteristics from data and combining them with current domain variables will benefit existing and well-tested prediction models in the financial domain.

In this chapter, motivated by the predictive capabilities of the online learning methods and statistical feature expansion, we propose an online KAF-based method for stock price prediction. We use and construct multiple additional features from the stock data to make predictions more accurate. To our knowledge, this is the first study to look into the suitability of online KAF algorithms leveraging feature expansion techniques. Even though most studies focuses on technical analysis to expand feature sets, we take on the issue of analyzing statistical, temporal, and frequency-based features for online learning. We apply various statistical, frequency, and time-based features to achieve the objective. To do this, we construct an additional fifty-eight features. Further, we use ten different online KAF algorithms for eight different time windows. The time windows are constructed via the quotes gathered for one day down to five minutes. All the experiments are performed on the main index of NSE: Nifty 50. Precisely, we work with all the fifty stocks of the index. The results show the superior performance of the proposed method in actual deployment scenarios. The following points summarize the main contributions of the chapter:

- 1. We present a novel online learning KAF-based approach complemented with feature expansion for stock price prediction.
- Rather than focusing on predicting stock prices for a single stock, the study proposes the usage of ten KAF algorithms that can predict the prices of fifty different stocks simultaneously.
- 3. To the best of our knowledge, this article is the first work on feature expansion and evaluating numerous KAF algorithms, as well as offering a detailed analysis of Nifty 50.
- 4. This study's main goal is to forecast stocks' mid-price. The empirical results indicate the superiority of the work presented.

The rest of the chapter has been structured in various sections. Section 5.2 elucidate the details of the proposed KAF-based stock prediction approach and Section 5.2.3 details the problem statement. The experiments performed with various KAF algorithms, their results, and their comparison with other stock price prediction methods are summarized in Section 5.3. Finally, conclusion is discussed Section in 5.4.

5.2 METHODOLOGY

This section describe the details of the proposed approach. The entire approach for stock price prediction has been divided into three subsections. The first section describes the online KAF-based approach. The second subsection discusses the feature selection procedure and its advan-tages. Finally, a discussion on the problem statement is given in the next subsection.

5.2.1 KERNEL ADAPTIVE FILTERING TECHNIQUES AND THEIR METHODS

In this article, the objective is to learn the input and output mapping $f: S \to R$, based on a well known sequence $((s_1, t_1), (s_2, t_2), \dots, (s_i, t_i))$, where $S \subseteq R^L$ is the input space s_i , where i = 1, ..., n is the system input at sample time and t_i is the desired response. The input data

can be converted into a high-dimensional feature space F to make non linear predictions. The famous "kernel trick" eliminates the need for costly computations in high-dimensional space. In addition to this, KAF has several advantages. They are as follows:

- 1. KAF follows the approach of universal approximators.
- 2. KAF also uses gradient descent learning. Therefore, it does not suffer from local minima and it can deal with challenges like computing and memory complexity.
- 3. KAFs follow the idea of online learning, here, it can handle a non-stationary environment more effectively.

KAF-based computations generally involve the use of a Kernel. One such kernel is given in equation 5.1:

$$\kappa < s, s' >= exp \frac{(||s - s'||^2)}{\sigma^2}$$
(5.1)

Before proceeding further, we must point out that the work presented in this article builds upon our previous work presented in (236). In this article, we have used the methods discussed in our previous work and have tried to augment the financial time series with additional statistical features. The list of algorithms used in the article are described in Table 5.1. For reasons of brevity, we do not expand upon each and every algorithm in this article. The interested reader is referred to (236).

5.2.2 FEATURE ENGINEERING

Feature engineering have been used in different fields such as neurocomputing and time-series. Our major objective is to estimate the mid-price using feature expansion and apply online KAF algorithms for prediction. To that end, we calculate a vast number of statistical, temporal, and frequency-based features that have all been applied to the eight different time windows. In feature engineering, we examined many aspects of the best possible approach for predicting

Table 5.1: KAF algorithms.

Algorithm	Description
KRLS	Kernel Recursive Least Mean Square
LMS	Least Mean Square
LKAPA	Leaky Kernel Affine Projection
KLMS	Kernel Least Mean Square
NORMA	Normalized Online Regularized Risk Minimization
КМСС	Kernel Adaptive Filtering with Maximum Cross-entropy
	Criterion
PROB-LMS	Probabilistic Least-Mean Square Filter
KNLMS	Kernel Normalized Least Mean Square
QKLMS	Quantized Kernel Least Mean Square
FBQKLMS	Fixed Budget Quantized Kernel Least Mean Square Algo-
	rithm

short-term prices: feature engineering, financial domain knowledge, and the prediction algorithm. Then, for each aspect, we try to address two important questions:

- 1. What are the advantages of feature engineering to improve model prediction?
- 2. What impact do financial domain results have on building prediction models?

Solutions

- 1. The first question concerns feature engineering. We want to learn how the feature selection strategy improve model prediction performance. From the large body of prior research, it can be inferred that stock price data contains a high amount of noise and typical correlations, making price prediction challenging. That is also why most earlier studies treated feature engineering as an optimization module. In our case, we used the concept with the help of novel online KAF class algorithms to improve the performance.
- 2. The second study topic is the effectiveness of the findings. We expand features from the statistical, frequency, and temporal domains. Unlike past studies, our evaluation focuses

Table 5.2: If we choose Embedding Dimension *M*=10, then Input=[{Input Feature 1,, ,Input Feature 58}]. Output=[{Output}]

Instance num-	actual price	Instance num-	actual price	
ber		ber		
Input Feature 1	Actual price	Input Feature 30	Longest strike below mean	
Input Feature 2	Actual price	Input Feature 31	Longest strike above mean	
Input Feature 3	Actual price	Input Feature 32	Count above mean	
Input Feature 4	Actual price	Input Feature 33	Count below mean	
Input Feature 5	Actual price	Input Feature 34	Last location of maximum	
Input Feature 6	Actual price	Input Feature 35	First location of maximum	
Input Feature 7	Actual price	Input Feature 36	Number cwt peaks	
Input Feature 8	Actual price	Input Feature 37	Time reversal asymmetry	
			statistic	
Input Feature 9	Actual price	Input Feature 38	C3 statistics to measure non	
			linearity	
Input Feature 10	Actual price	Input Feature 39	Binned entropy	
Input Feature 11	Petrosian Fractal Dimen-	Input Feature 40	Approximate entropy	
	sion(PFD)			
Input Feature 12	Higuchi Fractal Dimen-	Input Feature 41	Fourier entropy	
	sion(HFD)			
Input Feature 13	Katz Fractal Dimension	Input Feature 42	Lempel ziv complexity	
Input Feature 14	Fisher Information	Input Feature 43	Quantile	
Input Feature 15	Singular Value Decomposi-	Input Feature 44	Autocorrelation	
	tion Entropy (SVD)			

on the effectiveness of newly added characteristics retrieved from a different realm rather than the standard evaluation of data models, such as training costs and scores. While we only received a few particular conclusions from earlier studies, the raw data associated with them must be transformed into usable features. We combine the features with a fixed embedding dimension after expanding related features from the different domains to vote out the features with the highest impact. The list of features used in the article is shown in Table 5.2 and Table 5.3.

Table 5.3: If we choose Embedding Dimension M=10, then Input=[{Input Feature 1,, ,Input Feature 58}]. Output=[{Output}]

Instance num-	actual price	Instance num-	actual price	
ber		ber		
Input Feature 16	Mean	Input Feature 45	Calc centroid	
Input Feature 17	Mean change	Input Feature 46	Mean absolute diff	
Input Feature 18	Ratio beyond r sigma	Input Feature 47	Mean diff	
Input Feature 19	Sum values	Input Feature 48	Median absolute diff	
Input Feature 20	Cid ce	Input Feature 49	Median diff	
Input Feature 21	Mean absolute change	Input Feature 50	Signal, distance	
Input Feature 22	Mean second derivative cen-	Input Feature 51	Sum absolute diff	
	tral			
Input Feature 23	Median	Input Feature 52	Total energy	
Input Feature 24	Standard deviation	Input Feature 53	Slope	
Input Feature 25	Variance	Input Feature 54	Area under the curve	
Input Feature 26	Skewness	Input Feature 55	Peak to peak distance	
Input Feature 27	Kurtosis	Input Feature 56	interquartile range	
Input Feature 28	Root mean square	Input Feature 57	spectral distance	
Input Feature 29	Absolute sum of changes	Input Feature 58	App entropy	

5.2.3 REGRESSION PROBLEM FORMULATION

As mentioned in Section 5.1, the primary goal of this research is to forecast the stock's midprice. The aim of stock price prediction is to determine future stock prices based on previous values. Therefore, in this research, the stock price prediction is formulated as regression problem and we employ the notion of auto-regression of the order *m* to predict future stock price. In addition to this, we try to predict the percentage change in the mid-price. Using historical data, an autoregressive (AR) model can effectively predict future behaviour. When there is a relationship between a time series' values and the values that precede or succeed them, AR models have demonstrated enormous potential in those circumstances. Hence, we have chosen this type of modeling in our work. The problem of stock price prediction is formulated as:

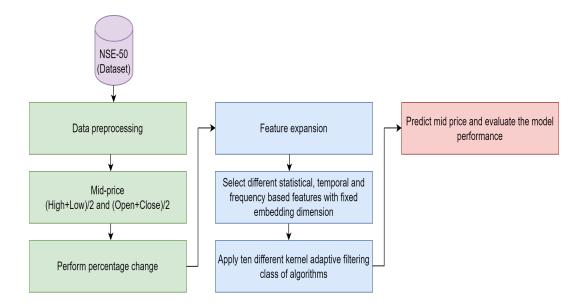


Figure 5.1: Architecture of the proposed work.

$$(S_i) = \sum_{i=1}^{m} \omega_{(i-1)}(S_i)$$
(5.2)

In equation (2), S_i represents the change in mid-price, and ω represents the weight vector. Further, the proposed methodology includes feature expansion. To that end, we developed several features, tested them, and collected the best fifty-eight. Lastly, we also experimented with a fixed embedding dimension. Finally, for experimentation, we used algorithms from the KAF family. It is worth mentioning that the method outlined above here was followed for every stock. Further, we use the similar concept given in Equation 5.2 and reformulate with feature expansion as shown in Equation 5.3.

$$(S_{i+1}) = \hat{f}(S(1....n - \tau) \oplus \psi(1:58))$$
(5.3)

For input, we utilize the Equation 5.3 and concatenate (\oplus) fifty-eight additional features. They are based on the statistical, temporal, and frequency-based for different time windows. Table 2 and 3 shows the list of features selected in our work. In the above equation, τ represents the embedding dimension. For experimentation, in our case, we selected $\tau = 50$. Fig. 5.1 show the overall framework of the proposed methodology. As shown in Fig. 5.1, the analysis was carried out using the Nifty-50 dataset. We have utilized two scenarios: (High+Low)/2 and (Open+Close)/2 to calculate the mid-price. Moreover, the proposed feature-based approach presented here differs significantly from those previously proposed in the literature. We tested the performance of the KAF family of algorithms by calculating a large variety of statistical, temporal, and frequency-based characteristics that were all applied to the eight-time windows.

5.3 **RESULTS**

5.3.1 DATASET COLLECTION

This section describes the detail of the dataset used to evaluate the proposed method. The data of stock was taken from the the National Stock Exchange of India. We collected data from January 01, 2021 to December 31, 2021, where we dealt with the official trading hours of all stocks from 9:15 a.m. to 3:30 p.m. In addition, we applied the concept of feature engineering for each stock. As discussed in the previous section, ten KAF algorithms were utilized to compare performance. For experimentation we are using NSE-50 dataset ¹. The dataset is available for one minute in the granular form. The data that we considered for our research includes open, high, low, and close (OHLC) prices and represent open values on the day, the highest value on the day, the lowest value on the day, and the closing value on the day. The dataset was constructed and preprocessed according to eight prediction windows: one day, sixty minutes, thirty minutes, twenty-five minutes, twenty-minutes, fifteen minutes, ten minutes, and five minutes. It should be noted here that in this research we are attempting to estimate the mid-price using two distinct scenarios: (high + low)/2 and (open + close)/2. Furthermore, we try to predict the next percentage change in the mid price. It should be noted here that all the values in the feature set were normalized between 0 and 1.

¹https://shorturl.at/lnvF2

5.3.2 EVALUATION MERTICS

To validate the effectiveness and efficacy of the proposed method, standard evaluation metrics are used. The different evaluation metrics used are: mean squared error (MSE) and directional symmetry (DS). They are presented in Table 5.4. In Table 5.4 D_i represents the following cases:-

$$D_i = \begin{cases} 1 & (a_i - a_{i-1})(p_i - p_{i-1}) \ge 0\\ 0 & \text{otherwise} \end{cases}$$

Where, *n* is the time-step, a_i represents the actual values, and p_i represents the predicted output.

Criteria	Formulation
MSE	$\sum_{i=1}^{n} \left(y_i - d_i \right)^2$
DS	$\frac{1}{n}\sum_{i=1}^{n}D_{i}$

Table 5.4: Define the evaluation criterion

Overall Strategy Followed

- As mentioned in Section 5.1, we work with NSE-50 companies and used 2021 datasets (from January 01, 2021 to December 31, 2021). Further, the hyperparameter values for each of the ten algorithms is given in Table 5.5. The ideal values for these hyperparameters were determined through multiple rounds of experimentation.
- 2. We analyze the performance of mid-price prediction with average share prices and apply different feature expansion methods. We set up the problem as an autoregressive problem (see Section 5.2.3) and calculated the error values for each of the 50 stocks. One for MSE and one for DS. Hence, we obtain 50×2 error estimates. Please note that there are eight distinct prediction windows. Consequently, for all stocks and for all windows, ten algorithms were subjected to price estimation tests.
- 3. We obtained the final value by averaging all fifty-error measures for a single time window

and a single stock. The obtained values show the model's predictive capacity for all 50 stocks.

Algorithms	σ	$\sigma_2 n$	$\sigma_2 d$	η	ε	Λ	tcoff	τ	тиО	P	пи
LMS	-	-	-	-	-	-	-	_	0.04	-	-
KRLS	7.0	-	-	-	-	-	-		-	-	1E-
											5
KLMS	7.0	-	-	1.7	-	-	-		-	-	-
LKAPA	6.0	-	-	0.05	-	1E-	-	-	-	20	-
						2					
NORMA	7.0	-	-	-	1.5	1E-	0.9	5000	-	-	-
						2					
FBQKLMS	2.0	-	-	0.9	0.2	-	-	-	0.95	20	-
QKLMS	4.0	-	-	1.7	1.3	-	-	-	-	-	-
КМСС	7.0	-	-	1.4	-	-	-	-	-	-	-
KNLMS	2.0	-	-	1.9	1E-	-	-	_	2	-	-
					4						
PROB-LMS	-	2	3	-	-	0.9	-	-	-	-	-

Table 5.5: Parameter description of KAF techniques for NSE-50 data set for mid-price.

5.3.3 PREDICTION AND CONVERGENCE ANALYSIS

In this subsection, we investigate the performance of several KAF techniques for mid-price prediction. In this regard, we present the prediction graphs for one stock (CIPLA) using the FBQKLMS algorithm. Fig. 5.2 and Fig. 5.3 demonstrate the outcomes for (high + low)/2 and (open + close)/2, respectively. It can be seen from the prediction graphs that the predictive curve fits the original curve well. Further, it is important to note that we have only provided results for one forecast window (one day) with one stock (CIPLA). Other stocks in the dataset had similar results. We cannot say that they are 100 percent correct, but they are close. In addition to the prediction graphs, the numbers for MSE are summarized in Table 5.9 and Table 5.10. We must emphasize how difficult it is to accurately estimate financial time series values. Therefore,

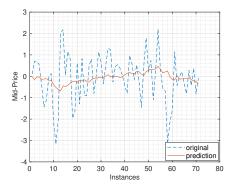


Figure 5.2: Prediction Graphs of 2021 dataset with (High+Low)/2.

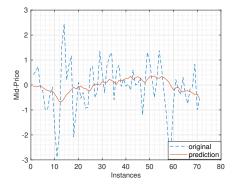
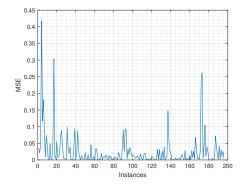


Figure 5.3: Prediction Graphs of 2021 dataset with (Open+Close)/2.

obtaining values that are close enough has always been the objective. Thus, the outcome that we were able to accomplish demonstrates the work's strong capability to forecast.



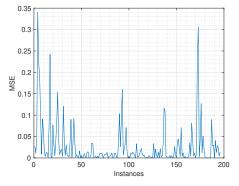


Figure 5.4: Convergence Graphs of 2021 dataset with (High+Low)/2.

Figure 5.5: Convergence Graphs of 2021 dataset with (Open+Close)/2.

Fig. 5.4 and Fig. 5.5 illustrate the convergence results for (High+Low)/2 and (Open+Close)/2 respectively. Similar to the earlier scenario, we have shown the results utilizing the FBQKLMS algorithm with just one forecast window (one day) and one stock (CIPLA)). As visible, the method quickly converges around the 100th data point. KAF algorithms demonstrate the ability to adapt and converge fast. There is one more significant finding from the convergence graphs: while there is some variation in the graphs, it is still acceptable. This is due to the fact that there will be noise in the new data and that small changes are inevitable.

Regarding stock movement, the correlation of actual and desired prices is assessed using directional symmetry. It is a metric for assessing a model's capacity to forecast the movement of a stock. To comprehend how a stock moves, we looked at ten different methods. According

to the experiment, we have a 57% chance of correctly forecasting the next upward or downward movement when employing KLMS, as shown in Table 5.9. The five-minute time window yields the highest results. It is also clear from the table that there is a significant difference between the results obtained for the one-day window and those for the other windows.

5.3.4 COMPREHENSIVE EVALUATION OF ALGORITHMS

Unlike batch learning methods that are aimed at generating the best predictor through learning the entire training set at once. In our proposed approach, we used an online learning strategy in which data becomes available in sequential order (sample by sample training) and is used to update the best predictor for future data at each step. As stated earlier, there are ten different types of online KAF algorithms. We compare the performance of all algorithms for each particular stock. We presented the results in two ways: (High+Low)/2 and (Open+Close)/2, and expand the features based on the statistical domain. We attempted to experiment with the NSE-50, 2021 dataset to demonstrate the work's efficacy. The results are shown in Table 5.9 and Table 5.10 using this configuration. It is evident from the tables that LKAPA outperformed the competitiveness in error minimization once again. The figure and the algorithm for DS, on the other hand, is different. It is KLMS and KRLS that performs best for DS. Indeed, there is a divergence. Nevertheless, KLMS, KRLS, NORMA, and KMCC were found to be superior in the experiment.

Table 5.6: Comparison of the proposed approach with other methods for one-day time window (High+Low)/2.

Method	MSE
Agrawal et al. (237)	0.4957
Henrique et al. (238)	0.0561
Kamble et al. (239)	0.0365
Khaidem et al. (109)	0.02425
Proposed method (FBQKLMS)	0.01991

5.3.5 COMPARISON WITH DIFFERENT STATE-OF-THE-ART METH-ODS

We compared the results of the proposed method with other stock prediction methods i.e. (237), (238), (239), and (109). The NSE-50 dataset was used to train all of the approaches. We picked fifty stocks for one-day intervals to preserve uniformity across multiple testing procedures. An 80:20 split was used to train the machine learning algorithms. Subsequently, the result of prediction was compared to the best KAF method. To perform all the experiments, we set an embedding dimension (τ)=50. Table 5.6 and Table 5.7 summarises the empirical results of (High+Low)/2 and (Open+Close)/2. It can be seen from the table that the proposed method outperforms existing stock prediction algorithms in the literature. This validates the superiority of KAF and feature extension based stock price prediction.

Table 5.7: Comparison of the proposed approach with other methods for one-day time window (Open+Close)/2.

Method	MSE
Agrawal et al. (237)	0.3925
Henrique et al. (238)	0.0419
Kamble et al. (239)	0.0348
Khaidem et al. (109)	0.02221
Proposed method (FBQKLMS)	0.01862

5.3.6 EFFECT OF DIMENSIONALITY REDUCTION METHODS

Optimization techniques like principal component analysis (PCA) were also used to anticipate short-term stock prices. To minimize the number of dimensions, PCA and Kernel PCA (KPCA) was used. In the proposed method, we selected the most commonly used statistical characteristics and utilize the feature extension approach to expand the feature set. From the extended feature set, we have selected the *i* features that are the most effective. The data with *i* selected

features is then fed into the system, which reduces the dimension to j features. Finally, we transformed the data into a new feature set (j). This reduced feature set was then used for prediction. Table 5.8 shows the figures for a one-day time window. The numbers compare the effects of PCA and KPACA to the original time windows. As visible from the table, dimensionality reduction techniques had no effect on the results. The reason why we witness such a behavior, however, is unknown. We expected either PCA or KPCA to improve upon the performance, but the experiment reveal something different.

Table 5.8: Effect of dimensionality reduction using LKAPA for (Open+Close)/2 and (High+Low)/2.

	(Open+Close)/2			
Method	MSE	DS	MSE	DS
PCA	0.01935	0.4294	0.01935	0.429431
КРСА	0.02220	0.39587	0.02012	0.43634
Proposed [1-Day]	0.01991	0.4857	0.01862	0.5193

Table 5.9: Evaluation results in terms of evaluation criterion for mid price (High+Low)/2.

Time Window	MSE	Best algorithms	DS	Best algorithms
		(According to		(According to
		MSE)		DS)
Five-Minutes	0.00083	LKAPA	0.5731	KLMS
Ten-Minutes	0.0013	LKAPA	0.5216	KLMS
Fifteen-Minutes	0.0019	LKAPA	0.4787	KRLS
Twenty-Minutes	0.0022	LKAPA	0.4814	KRLS
Twenty-five-Minutes	0.0027	LKAPA	0.4837	KRLS
Thirty-Minutes	0.00317	LKAPA	0.4861	KRLS
Sixty-Minutes	0.00549	LKAPA	0.4902	KRLS
One-Day	0.01991	FBQKLMS	0.4857	KRLS

Time Window	MSE	Best algorithms	DS	Best algorithms
		(According to		(According to
		MSE)		DS)
Five-Minutes	0.00070	LKAPA	0.5555	КМСС
Ten-Minutes	0.001110	LKAPA	0.5401	KLMS
Fifteen-Minutes	0.00167	LKAPA	0.5367	NORMA
Twenty-Minutes	0.00197	LKAPA	0.5302	NORMA
Twenty-five-Minutes	0.00239	LKAPA	0.5304	NORMA
Thirty-Minutes	0.00289	LKAPA	0.5261	NORMA
Sixty-Minutes	0.00516	LKAPA	0.5152	KNLMS
One-Day	0.01862	FBQKLMS	0.51939	KNLMS

Table 5.10: Evaluation results in terms of evaluation criterion for mid price (Open+Close)/2.

5.3.7 IMPORTANT NOTE: ERROR MINIMIZATION AND PROFITABIL-ITY

Minimization of error explains the concept that predicted data is close to the actual data. From Table 5.9 and Table 5.10 we can see LKAPA gives best performance in terms of error minimization. Furthermore, we obtain the lowest error (MSE) on the order of 10^-4 . However, there is a concern with the five-minute time window. The price variation is low over this time period. As a result, sometimes minimizing error will not yield a large profit. To put it in another way, the volatility in five minutes is less. As a result, forecasts are extremely close. In a low-volatility environment, however, the chances of initiating a position and profiting are likewise low. As a result, a balance must be achieved between error minimization and profitability.

5.4 CONCLUSIONS

Stock price prediction is a difficult task owing to its nonlinear, complex, and volatile nature. Yet, it is one of the essential problems of literature. Successful prediction provides several intriguing advantages that could influence a trader's decision to buy or sell a financial asset. The Stock Exchange's index is one of the most important factors that investors evaluate while making an investment. Therefore, the objective of this study was to predict the mid-price of the main index of NSE. For this purpose, we proposed a feature expansion method complemented via the online KAF class of algorithms. First, we constructed several features and finalized the best fifty-eight statistical, temporal, and frequency-based features. We then used ten different KAF classes of algorithms for eight-time windows and analyzed the performance of each stock. Finally, we compared existing methods with our proposed online KAF approach. After extensive numerical simulations, combining the feature expansion approach with online KAF prediction algorithms provided the best predictive performance. Lastly, feature reduction techniques were tried but had no effect on the performance of the method. To the best of our knowledge, this paper is the first wherein numerous KAF algorithms with feature augmentation have been applied to the stock window at such a granular level. Furthermore, the proposed method outperformed previous methods in terms of error minimization and execution time, indicating its utility in actual deployment.

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH

6.1 CONCLUSIONS

Stock price prediction is a challenging and tedious task. Although various methods have been developed, an investigation of accurate and low latency methods is not given much attention. Among the available popular techniques, machine learning methods are researched most due to their capabilities for recognizing complex patterns in the financial time series. In our research, two problems were addressed: trend analysis and stock price prediction. First, we propose a 3D CNN and 3D ensemble methods for trend analysis. This was done to analyze the directional trends in a stock's movement. Furthermore, and in contrast to existing literature, e.g., (172), (181), (214), we do not focus on predicting the trend in a single company, the goal is to propose a framework that could predict the trend while analyzing a sector as a whole. To that end, multiple technical indicators were computed. Out of which, the best indicators (or features) are selected via Hierarchical Clustering. Subsequently, data of five similar companies are grouped together, and the entire compendium was put into a $15 \times 15 \times 5$ image. Classification of a stock's trend happened on this 3D image. For the purpose of classification, we experimented with 3D CNN and ensembled CNN. Particularly, the work shows good performance in terms of accuracy and annual returns. The model achieved up to 35% returns in some cases, with the average being 9.19%.

Second, we propose an online learning-based kernel adaptive filtering approach for stock price prediction. In other words, price prediction is the subsequent problem addressed in this thesis. We worked with ten different algorithms and proposed a method to predict the next close and the mid-price. The idea was tested on fifty stocks of the NSE index with nine different time windows such as one-minute, five-minutes,ten-minutes, fifteen-minutes, twenty-minutes, thirty-minutes, one hour, and one day. It should be noted here that this work is the first wherein stock is analyzed by looking at nine different time windows. In addition to this, we augment financial time by computing several additional statistical features to improve predictive performance. The empirical results suggest that kernel adaptive filtering could be an efficient tool for high-frequency trading as well. The work presented here demonstrates the superiority of the kernel adaptive filtering family of algorithms over traditional classification and regression methods in terms of their predictive power.

The following text highlights the summary of the research in short:

- We focus on trend analysis and propose 3D CNN and 3D ensembled CNN to predict the long-term movement in stocks. We experiment with the notion of section aware stock trend prediction. In this work, we do not focus on predicting the trend in a single company, rather, the purpose of the work is to present a framework for predicting the trend while examining a sector as a whole. To the best of our knowledge, this work is the first wherein we apply the paradigm of 3D CNN and ensembled CNN to examine the effect of sector aware trend classification. The results we got also show promise and can lay the groundwork for future work in stock trend prediction.
- In addition to trend prediction, we also focus on the stock price prediction. To solve the problem of price prediction, we use online KAF methods. To the best of our knowledge, we are the first to comprehensively test the predictive performance of the KAF class of algorithms in stocks on multiple time-windows. The proposed method has very little execution time, thereby allowing traders to quickly sell and buy stocks with a high probability of profits. In the same problem, we also explored the idea of feature engineering with KAF. We complemented the basic financial time with additional statistical features to make the prediction more accurate. To validate the effectiveness and feasibility of the proposed work, the empirical evaluation was performed on the stock consisting of the main index of NSE: Nifty 50.

6.2 FUTURE RESEARCH

The potential future research will be aimed at the following:

• To improve the performance of base 3D CNN, transfer learning can play an important role. In this context, future work will be directed at exploring the feasibility of transfer learning to further enhance the model performance. Furthermore, by incorporating the most modern sentiment analysis methods with feature engineering, a more comprehensive prediction system that is trained on a number of data sources, including news, tweets, and other text-based data, is possible.

- Finding the appropriate value of the hyperparameters has a significant impact on the performance of the prediction model. In our work, we found the best parameters via brute force. However, better optimization techniques such as Bayesian optimization, reinforcement learning, and other population-based techniques can reduce the efforts to find the best value of the parameters.
- Another important research direction could be extending the features by using alternate data sources such as Wikipedia trends and Google analytics. This could be useful via utilizing Google's real-time search terms. By examining the most relevant trend, interest over time, and trending searches, traders can gather information into other people's search interests and learn more about their interests. This is the subsequent objective of our future work.

REFERENCES

- A. Abraham, N. S. Philip, and P. Saratchandran, "Modeling chaotic behavior of stock indices using intelligent paradigms," *arXiv preprint cs/0405018*, 2004.
- [2] D. Berger, K. Pukthuanthong, and J. J. Yang, "International diversification with frontier markets," *Journal of Financial Economics*, vol. 101, no. 1, pp. 227–242, 2011.
- [3] R. Singh and S. Srivastava, "Stock prediction using deep learning," *Multimedia Tools and Applications*, vol. 76, no. 18, pp. 18569–18584, 2017.
- [4] E. F. Fama, "Efficient capital markets: A review of theory and empirical work," *The journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.
- [5] D. Hartman and J. Hlinka, "Nonlinearity in stock networks," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 8, p. 083127, 2018.
- [6] X. Liang, H. Zhang, J. Xiao, and Y. Chen, "Improving option price forecasts with neural networks and support vector regressions," *Neurocomputing*, vol. 72, no. 13-15, pp. 3055– 3065, 2009.
- [7] L. K. Chan, J. Lakonishok, and B. Swaminathan, "Industry classifications and return comovement," *Financial Analysts Journal*, vol. 63, no. 6, pp. 56–70, 2007.
- [8] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques," *Expert systems with applications*, vol. 42, no. 1, pp. 259–268, 2015.
- [9] B. Krollner, B. J. Vanstone, and G. R. Finnie, "Financial time series forecasting with machine learning techniques: a survey." in *ESANN*, 2010.
- [10] M. Vijh, D. Chandola, V. A. Tikkiwal, and A. Kumar, "Stock closing price prediction using machine learning techniques," *Procedia Computer Science*, vol. 167, pp. 599–606, 2020.

- [11] W. Chen, M. Jiang, W.-G. Zhang, and Z. Chen, "A novel graph convolutional feature based convolutional neural network for stock trend prediction," *Information Sciences*, vol. 556, pp. 67–94, 2021.
- [12] J. Q. Candela and O. Winther, "Incremental gaussian processes," in Advances in Neural Information Processing Systems, 2003, pp. 1025–1032.
- [13] L. Yu, S. Wang, and K. K. Lai, "An online learning algorithm with adaptive forgetting factors for feedforward neural networks in financial time series forecasting," *Nonlinear dynamics and systems theory*, vol. 7, no. 1, pp. 51–66, 2007.
- [14] J. Pardo, F. Zamora-Martínez, and P. Botella-Rocamora, "Online learning algorithm for time series forecasting suitable for low cost wireless sensor networks nodes," *Sensors*, vol. 15, no. 4, pp. 9277–9304, 2015.
- [15] T. Guo, Z. Xu, X. Yao, H. Chen, K. Aberer, and K. Funaya, "Robust online time series prediction with recurrent neural networks," in 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA). IEEE, 2016, pp. 816–825.
- [16] M. Han, S. Zhang, M. Xu, T. Qiu, and N. Wang, "Multivariate chaotic time series online prediction based on improved kernel recursive least squares algorithm," *IEEE transactions on cybernetics*, vol. 49, no. 4, pp. 1160–1172, 2018.
- [17] W. Liu, J. C. Principe, and S. Haykin, *Kernel adaptive filtering: a comprehensive introduction*. John Wiley & Sons, 2011, vol. 57.
- [18] P. Yu and X. Yan, "Stock price prediction based on deep neural networks," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1609–1628, 2020.
- [19] Z. Hu, Y. Zhao, and M. Khushi, "A survey of forex and stock price prediction using deep learning," *Applied System Innovation*, vol. 4, no. 1, p. 9, 2021.
- [20] A. Sharma, D. Bhuriya, and U. Singh, "Survey of stock market prediction using machine learning approach," in 2017 International conference of electronics, communication and aerospace technology (ICECA), vol. 2. IEEE, 2017, pp. 506–509.

- [21] M. Nikou, G. Mansourfar, and J. Bagherzadeh, "Stock price prediction using deep learning algorithm and its comparison with machine learning algorithms," *Intelligent Systems in Accounting, Finance and Management*, vol. 26, no. 4, pp. 164–174, 2019.
- [22] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, vol. 9, no. 2, pp. 85–112, 2020.
- [23] J.-F. Chen, W.-L. Chen, C.-P. Huang, S.-H. Huang, and A.-P. Chen, "Financial timeseries data analysis using deep convolutional neural networks," in 2016 7th International conference on cloud computing and big data (CCBD). IEEE, 2016, pp. 87–92.
- [24] Y. Hua, J. Guo, and H. Zhao, "Deep belief networks and deep learning," in *Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things*. IEEE, 2015, pp. 1–4.
- [25] A. Ng et al., "Sparse autoencoder," CS294A Lecture notes, vol. 72, no. 2011, pp. 1–19, 2011.
- [26] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*. Elsevier, 1992, pp. 65–93.
- [27] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.
- [28] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific reports*, vol. 8, no. 1, pp. 1–12, 2018.
- [29] A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.
- [30] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.
- [31] Q. Chen, W. Zhang, and Y. Lou, "Forecasting stock prices using a hybrid deep learning model integrating attention mechanism, multi-layer perceptron, and bidirectional longshort term memory neural network," *IEEE Access*, vol. 8, pp. 117 365–117 376, 2020.

- [32] S. S. Abdullah, M. S. Rahaman, and M. S. Rahman, "Analysis of stock market using text mining and natural language processing," in 2013 International Conference on Informatics, Electronics and Vision (ICIEV). IEEE, 2013, pp. 1–6.
- [33] W. Long, Z. Lu, and L. Cui, "Deep learning-based feature engineering for stock price movement prediction," *Knowledge-Based Systems*, vol. 164, pp. 163–173, 2019.
- [34] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural computation*, vol. 7, no. 2, pp. 219–269, 1995.
- [35] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 1999.
- [36] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2018.
- [37] M.-H. Yang, "Face recognition using kernel methods," Advances in neural information processing systems, vol. 14, 2001.
- [38] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.
- [39] F. Orabona, J. Keshet, and B. Caputo, "Bounded kernel-based online learning." *Journal of Machine Learning Research*, vol. 10, no. 11, 2009.
- [40] P. Zhao, S. C. Hoi, and R. Jin, "Double updating online learning," *Journal of Machine Learning Research*, vol. 12, p. 1587, 2011.
- [41] X. Xu, H. Qu, J. Zhao, X. Yang, and B. Chen, "Quantised kernel least mean square with desired signal smoothing," *Electronics Letters*, vol. 51, no. 18, pp. 1457–1459, 2015.
- [42] C. Saide, R. Lengelle, P. Honeine, C. Richard, and R. Achkar, "Nonlinear adaptive filtering using kernel-based algorithms with dictionary adaptation," *International Journal of Adaptive Control and Signal Processing*, vol. 29, no. 11, pp. 1391–1410, 2015.
- [43] J. C. Príncipe, W. Liu, and S. Haykin, Kernel adaptive filtering: a comprehensive introduction. John Wiley & Sons, 2011.

- [44] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.
- [45] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.
- [46] W. Liu and J. C. Príncipe, "Kernel affine projection algorithms," EURASIP Journal on Advances in Signal Processing, vol. 2008, pp. 1–12, 2008.
- [47] W. Liu, P. P. Pokharel, and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 543–554, 2008.
- [48] P. Bouboulis and S. Theodoridis, "Extension of wirtinger's calculus to reproducing kernel hilbert spaces and the complex kernel lms," *IEEE Transactions on Signal Processing*, vol. 59, no. 3, pp. 964–978, 2010.
- [49] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on signal processing*, vol. 52, no. 8, pp. 2275–2285, 2004.
- [50] W. Liu, I. Park, Y. Wang, and J. C. Principe, "Extended kernel recursive least squares algorithm," *IEEE Transactions on Signal Processing*, vol. 57, no. 10, pp. 3801–3814, 2009.
- [51] J. Platt, "A resource-allocating network for function interpolation," *Neural computation*, vol. 3, no. 2, pp. 213–225, 1991.
- [52] W. Liu, I. Park, and J. C. Principe, "An information theoretic approach of designing sparse kernel adaptive filters," *IEEE transactions on neural networks*, vol. 20, no. 12, pp. 1950–1961, 2009.
- [53] B. Chen, S. Zhao, P. Zhu, and J. C. Príncipe, "Quantized kernel least mean square algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 22–32, 2011.
- [54] —, "Mean square convergence analysis for kernel least mean square algorithm," *Signal Processing*, vol. 92, no. 11, pp. 2624–2632, 2012.
- [55] G. Wahba, Spline models for observational data. SIAM, 1990.

- [56] W. Härdle, Applied nonparametric regression. Cambridge university press, 1990, no. 19.
- [57] J. Racine, "An efficient cross-validation algorithm for window width selection for nonparametric kernel regression," *Communications in Statistics-Simulation and Computation*, vol. 22, no. 4, pp. 1107–1114, 1993.
- [58] S. An, W. Liu, and S. Venkatesh, "Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression," *Pattern Recognition*, vol. 40, no. 8, pp. 2154–2162, 2007.
- [59] E. Herrmann, "Local bandwidth choice in kernel regression estimation," *Journal of Computational and Graphical Statistics*, vol. 6, no. 1, pp. 35–54, 1997.
- [60] B. W. Silverman, Density estimation for statistics and data analysis. Routledge, 2018.
- [61] M. C. Jones, J. S. Marron, and S. J. Sheather, "A brief survey of bandwidth selection for density estimation," *Journal of the American statistical association*, vol. 91, no. 433, pp. 401–407, 1996.
- [62] S.-C. Huang, C.-C. Chiou, J.-T. Chiang, and C.-F. Wu, "A novel intelligent option price forecasting and trading system by multiple kernel adaptive filters," *Journal of Computational and Applied Mathematics*, vol. 369, p. 112560, 2020.
- [63] T.-C. Huang, R. N. Zaeem, and K. S. Barber, "It is an equal failing to trust everybody and to trust nobody: Stock price prediction using trust filters and enhanced user sentiment on twitter," *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 4, pp. 1–20, 2019.
- [64] Z. Cui, X. Shi, and Y. Chen, "Sentiment analysis via integrating distributed representations of variable-length word sequence," *Neurocomputing*, vol. 187, pp. 126–132, 2016.
- [65] K. Li, W. Pan, Y. Li, Q. Jiang, and G. Liu, "A method to detect sleep apnea based on deep neural network and hidden markov model using single-lead ecg signal," *Neurocomputing*, vol. 294, pp. 94–101, 2018.
- [66] S. M. Kasongo and Y. Sun, "A deep learning method with filter based feature engineering for wireless intrusion detection system," *IEEE access*, vol. 7, pp. 38 597–38 607, 2019.

- [67] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018.
- [68] D. Kumar, P. K. Sarangi, and R. Verma, "A systematic review of stock market prediction using machine learning and statistical techniques," *Materials Today: Proceedings*, 2021.
- [69] K. K. Yun, S. W. Yoon, and D. Won, "Prediction of stock price direction using a hybrid ga-xgboost algorithm with a three-stage feature engineering process," *Expert Systems with Applications*, vol. 186, p. 115716, 2021.
- [70] Q. Gao, "Stock market forecasting using recurrent neural network," Ph.D. dissertation, University of Missouri–Columbia, 2016.
- [71] K. George and P. Mutalik, "A multiple model approach to time-series prediction using an online sequential learning algorithm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 5, pp. 976–990, 2017.
- [72] W. Wu, J. Chen, L. Xu, Q. He, and M. L. Tindall, "A statistical learning approach for stock selection in the chinese stock market," *Financial Innovation*, vol. 5, no. 1, pp. 1–18, 2019.
- [73] N. Zhang, A. Lin, and P. Shang, "Multidimensional k-nearest neighbor model based on eemd for financial time series forecasting," *Physica A: Statistical Mechanics and its Applications*, vol. 477, pp. 161–173, 2017.
- [74] Y.-S. Chen, C.-H. Cheng, and W.-L. Tsai, "Modeling fitting-function-based fuzzy time series patterns for evolving stock index forecasting," *Applied intelligence*, vol. 41, no. 2, pp. 327–347, 2014.
- [75] Y.-J. Chen, Y.-M. Chen, and C. L. Lu, "Enhancement of stock market forecasting using an improved fundamental analysis-based approach," *Soft Computing*, vol. 21, no. 13, pp. 3735–3757, 2017.
- [76] G. S. Atsalakis and K. P. Valavanis, "Forecasting stock market short-term trends using a neuro-fuzzy based methodology," *Expert systems with Applications*, vol. 36, no. 7, pp. 10696–10707, 2009.

- [77] S. Borovkova and I. Tsiamas, "An ensemble of lstm neural networks for high-frequency stock market classification," *Journal of Forecasting*, vol. 38, no. 6, pp. 600–619, 2019.
- [78] K. Daniel, D. Hirshleifer, and A. Subrahmanyam, "Investor psychology and security market under-and overreactions," *the Journal of Finance*, vol. 53, no. 6, pp. 1839–1885, 1998.
- [79] B. Malkiel, S. Mullainathan, and B. Stangle, "Market efficiency versus behavioral finance," *Journal of Applied Corporate Finance*, vol. 17, no. 3, pp. 124–136, 2005.
- [80] W. F. De Bondt and R. Thaler, "Does the stock market overreact?" *The Journal of finance*, vol. 40, no. 3, pp. 793–805, 1985.
- [81] R. J. Shiller, "From efficient markets theory to behavioral finance," *Journal of economic perspectives*, vol. 17, no. 1, pp. 83–104, 2003.
- [82] A. W. Lo, "The adaptive markets hypothesis," *The Journal of Portfolio Management*, vol. 30, no. 5, pp. 15–29, 2004.
- [83] K.-P. Lim and R. Brooks, "The evolution of stock market efficiency over time: A survey of the empirical literature," *Journal of Economic Surveys*, vol. 25, no. 1, pp. 69–108, 2011.
- [84] P. B. Patel and T. Marwala, "Forecasting closing price indices using neural networks," in 2006 IEEE international conference on systems, man and cybernetics, vol. 3. IEEE, 2006, pp. 2351–2356.
- [85] M. Lam, "Neural network techniques for financial performance prediction: integrating fundamental and technical analysis," *Decision support systems*, vol. 37, no. 4, pp. 567– 581, 2004.
- [86] C. Lohrmann, P. Luukka, M. Jablonska-Sabuka, and T. Kauranne, "A combination of fuzzy similarity measures and fuzzy entropy measures for supervised feature selection," *Expert Systems with Applications*, vol. 110, pp. 216–236, 2018.
- [87] M. Sedighi, H. Jahangirnia, M. Gharakhani, and S. Farahani Fard, "A novel hybrid model for stock price forecasting based on metaheuristics and support vector machine," *Data*, vol. 4, no. 2, p. 75, 2019.

- [88] J. Murphy, "A comprehensive guide to trading methods and applications," 1999.
- [89] L.-Y. Wei, T.-L. Chen, and T.-H. Ho, "A hybrid model based on adaptive-network-based fuzzy inference system to forecast taiwan stock market," *Expert Systems with Applications*, vol. 38, no. 11, pp. 13 625–13 631, 2011.
- [90] T. Turner, A Beginner's Guide To Day Trading Online 2nd Edition. Simon and Schuster, 2007.
- [91] R. Efendi, N. Arbaiy, and M. M. Deris, "A new procedure in stock market forecasting based on fuzzy random auto-regression time series model," *Information Sciences*, vol. 441, pp. 113–132, 2018.
- [92] M.-Y. Chen and B.-T. Chen, "A hybrid fuzzy time series model based on granular computing for stock price forecasting," *Information Sciences*, vol. 294, pp. 227–241, 2015.
- [93] C.-Y. Yeh, C.-W. Huang, and S.-J. Lee, "A multiple-kernel support vector regression approach for stock market price forecasting," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2177–2186, 2011.
- [94] Y. Chen and Y. Hao, "A feature weighted support vector machine and k-nearest neighbor algorithm for stock market indices prediction," *Expert Systems with Applications*, vol. 80, pp. 340–355, 2017.
- [95] D. Enke and S. Thawornwong, "The use of data mining and neural networks for forecasting stock market returns," *Expert Systems with applications*, vol. 29, no. 4, pp. 927–940, 2005.
- [96] T. Singh, N. Saxena, M. Khurana, D. Singh, M. Abdalla, and H. Alshazly, "Data clustering using moth-flame optimization algorithm," *Sensors*, vol. 21, no. 12, p. 4086, 2021.
- [97] M. Kubat, "The genetic algorithm," in *An Introduction to Machine Learning*. Springer, 2017, pp. 309–329.
- [98] T.-c. Fu, F.-l. Chung, V. Ng, and R. Luk, "Pattern discovery from stock time series using self-organizing maps," in *Workshop notes of KDD2001 workshop on temporal data mining*, vol. 1. Citeseer, 2001.

- [99] T. Chavarnakul and D. Enke, "A hybrid stock trading system for intelligent technical analysis-based equivolume charting," *Neurocomputing*, vol. 72, no. 16-18, pp. 3517– 3528, 2009.
- [100] Y. Chen, W. Lin, and J. Z. Wang, "A dual-attention-based stock price trend prediction model with dual features," *IEEE Access*, vol. 7, pp. 148 047–148 058, 2019.
- [101] Y. Rajihy, K. Nermend, and A. Alsakaa, "Back-propagation artificial neural networks in stock market forecasting. an application to the warsaw stock exchange wig20," *Aestimatio: The IEB International Journal of Finance*, no. 15, pp. 88–99, 2017.
- [102] N. O'Connor and M. G. Madden, "A neural network approach to predicting stock exchange movements using external factors," in *International Conference on Innovative Techniques and Applications of Artificial Intelligence*. Springer, 2005, pp. 64–77.
- [103] L. Cao and F. E. Tay, "Financial forecasting using support vector machines," *Neural Computing & Applications*, vol. 10, no. 2, pp. 184–192, 2001.
- [104] W. Huang, Y. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine," *Computers & operations research*, vol. 32, no. 10, pp. 2513–2522, 2005.
- [105] R. Ebrahimpour, H. Nikoo, S. Masoudnia, M. R. Yousefi, and M. S. Ghaemi, "Mixture of mlp-experts for trend forecasting of time series: A case study of the tehran stock exchange," *International Journal of Forecasting*, vol. 27, no. 3, pp. 804–816, 2011.
- [106] Y. Pan, Z. Xiao, X. Wang, and D. Yang, "A multiple support vector machine approach to stock index forecasting with mixed frequency sampling," *Knowledge-Based Systems*, vol. 122, pp. 90–102, 2017.
- [107] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 871–882, 2013.
- [108] A. Nahil and A. Lyhyaoui, "Short-term stock price forecasting using kernel principal component analysis and support vector machines: the case of casablanca stock exchange," *Procedia Computer Science*, vol. 127, pp. 161–169, 2018.

- [109] L. Khaidem, S. Saha, and S. R. Dey, "Predicting the direction of stock market prices using random forest," *arXiv preprint arXiv:1605.00003*, 2016.
- [110] C. Lohrmann and P. Luukka, "Classification of intraday s&p500 returns with a random forest," *International Journal of Forecasting*, vol. 35, no. 1, pp. 390–407, 2019.
- [111] C.-F. Tsai and Y.-C. Hsiao, "Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches," *Decision support systems*, vol. 50, no. 1, pp. 258–269, 2010.
- [112] L. S. Malagrino, N. T. Roman, and A. M. Monteiro, "Forecasting stock market index daily direction: A bayesian network approach," *Expert Systems with Applications*, vol. 105, pp. 11–22, 2018.
- [113] J. Cao and J. Wang, "Stock price forecasting model based on modified convolution neural network and financial time series analysis," *International Journal of Communication Systems*, vol. 32, no. 12, p. e3987, 2019.
- [114] C.-J. Lu and J.-Y. Wu, "An efficient cmac neural network for stock index forecasting," *Expert Systems with Applications*, vol. 38, no. 12, pp. 15194–15201, 2011.
- [115] B. Weng, M. A. Ahmed, and F. M. Megahed, "Stock market one-day ahead movement prediction using disparate data sources," *Expert Systems with Applications*, vol. 79, pp. 153–163, 2017.
- [116] A. Subasi, F. Amir, K. Bagedo, A. Shams, and A. Sarirete, "Stock market prediction using machine learning," *Procedia Computer Science*, vol. 194, pp. 173–179, 2021.
- [117] M. Ali, D. M. Khan, M. Aamir, A. Ali, and Z. Ahmad, "Predicting the direction movement of financial time series using artificial neural network and support vector machine," *Complexity*, 2021.
- [118] Y. Kara, M. A. Boyacioglu, and Ö. K. Baykan, "Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange," *Expert systems with Applications*, vol. 38, no. 5, pp. 5311– 5319, 2011.

- [119] S. Lahmiri, "Resilient back-propagation algorithm, technical analysis and the predictability of time series in the financial industry," *Decision Science Letters*, vol. 1, no. 2, pp. 47–52, 2012.
- [120] B. Weng, L. Lu, X. Wang, F. M. Megahed, and W. Martinez, "Predicting short-term stock prices using ensemble methods and online data sources," *Expert Systems with Applications*, vol. 112, pp. 258–273, 2018.
- [121] S. Barak and M. Modarres, "Developing an approach to evaluate stocks by forecasting effective features with data mining methods," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1325–1339, 2015.
- [122] X. Zhang, Y. Hu, K. Xie, S. Wang, E. Ngai, and M. Liu, "A causal feature selection algorithm for stock prediction modeling," *Neurocomputing*, vol. 142, pp. 48–59, 2014.
- [123] Y. Chen and Y. Hao, "Integrating principle component analysis and weighted support vector machine for stock trading signals prediction," *Neurocomputing*, vol. 321, pp. 381– 402, 2018.
- [124] J. Wang and J. Wang, "Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks," *Neurocomputing*, vol. 156, pp. 68–78, 2015.
- [125] F. Ye, L. Zhang, D. Zhang, H. Fujita, and Z. Gong, "A novel forecasting method based on multi-order fuzzy time series and technical analysis," *Information Sciences*, vol. 367, pp. 41–57, 2016.
- [126] W.-C. Chiang, D. Enke, T. Wu, and R. Wang, "An adaptive stock index trading decision support system," *Expert Systems with Applications*, vol. 59, pp. 195–207, 2016.
- [127] J. Chai, J. Du, K. K. Lai, and Y. P. Lee, "A hybrid least square support vector machine model with parameters optimization for stock forecasting," *Mathematical Problems in Engineering*, vol. 2015, 2015.
- [128] M.-C. Lee, "Using support vector machine with a hybrid feature selection method to the stock trend prediction," *Expert Systems with Applications*, vol. 36, no. 8, pp. 10896– 10904, 2009.

- [129] C. Sprengel, K.-H. Baumann, and S. Neuer, "Seasonal and interannual variation of coccolithophore fluxes and species composition in sediment traps north of gran canaria (29 n 15 w)," *Marine Micropaleontology*, vol. 39, no. 1-4, pp. 157–178, 2000.
- [130] H. Gunduz, "An efficient stock market prediction model using hybrid feature reduction method based on variational autoencoders and recursive feature elimination," *Financial Innovation*, vol. 7, no. 1, pp. 1–24, 2021.
- [131] Y. Lin, S. Liu, H. Yang, and H. Wu, "Stock trend prediction using candlestick charting and ensemble machine learning techniques with a novelty feature engineering scheme," *IEEE Access*, vol. 9, pp. 101 433–101 446, 2021.
- [132] D. Enke, M. Grauer, and N. Mehdiyev, "Stock market prediction with multiple regression, fuzzy type-2 clustering and neural networks," *Procedia Computer Science*, vol. 6, pp. 201–206, 2011.
- [133] M. T. Leung, H. Daouk, and A.-S. Chen, "Forecasting stock indices: a comparison of classification and level estimation models," *International Journal of forecasting*, vol. 16, no. 2, pp. 173–190, 2000.
- [134] C.-F. Tsai, Y.-C. Lin, D. C. Yen, and Y.-M. Chen, "Predicting stock returns by classifier ensembles," *Applied Soft Computing*, vol. 11, no. 2, pp. 2452–2459, 2011.
- [135] X. Zhong and D. Enke, "Forecasting daily stock market return using dimensionality reduction," *Expert Systems with Applications*, vol. 67, pp. 126–139, 2017.
- [136] S. Barak, A. Arjmand, and S. Ortobelli, "Fusion of multiple diverse predictors in stock market," *Information Fusion*, vol. 36, pp. 90–102, 2017.
- [137] R. K. Lai, C.-Y. Fan, W.-H. Huang, and P.-C. Chang, "Evolving and clustering fuzzy decision tree for financial time series data forecasting," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3761–3773, 2009.
- [138] D. Olson and C. Mossman, "Neural network forecasts of canadian stock returns using accounting ratios," *International Journal of Forecasting*, vol. 19, no. 3, pp. 453–465, 2003.

- [139] S. T. A. Niaki and S. Hoseinzade, "Forecasting s&p 500 index using artificial neural networks and design of experiments," *Journal of Industrial Engineering International*, vol. 9, no. 1, pp. 1–9, 2013.
- [140] X. Zhong and D. Enke, "Predicting the daily return direction of the stock market using hybrid machine learning algorithms," *Financial Innovation*, vol. 5, no. 1, pp. 1–20, 2019.
- [141] D. L. Minh, A. Sadeghi-Niaraki, H. D. Huy, K. Min, and H. Moon, "Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network," *Ieee Access*, vol. 6, pp. 55 392–55 404, 2018.
- [142] Y. Shynkevich, T. M. McGinnity, S. A. Coleman, and A. Belatreche, "Forecasting movements of health-care stock prices based on different categories of news articles using multiple kernel learning," *Decision Support Systems*, vol. 85, pp. 74–83, 2016.
- [143] S. Feuerriegel and J. Gordon, "Long-term stock index forecasting based on text mining of regulatory disclosures," *Decision Support Systems*, vol. 112, pp. 88–97, 2018.
- [144] L. Shi, Z. Teng, L. Wang, Y. Zhang, and A. Binder, "Deepclue: visual interpretation of text-based deep stock prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 6, pp. 1094–1108, 2018.
- [145] S. Wang, L. Yu, L. Tang, and S. Wang, "A novel seasonal decomposition based least squares support vector regression ensemble learning approach for hydropower consumption forecasting in china," *Energy*, vol. 36, no. 11, pp. 6542–6554, 2011.
- [146] M. Göçken, M. Özçalıcı, A. Boru, and A. T. Dosdoğru, "Integrating metaheuristics and artificial neural networks for improved stock price prediction," *Expert Systems with Applications*, vol. 44, pp. 320–331, 2016.
- [147] J. Zhou and P. Fan, "Modulation format/bit rate recognition based on principal component analysis (pca) and artificial neural networks (anns)," OSA Continuum, vol. 2, no. 3, pp. 923–937, 2019.
- [148] D. Pradeepkumar and V. Ravi, "Forecasting financial time series volatility using particle swarm optimization trained quantile regression neural network," *Applied Soft Computing*, vol. 58, pp. 35–52, 2017.

- [149] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.
- [150] L. Wang, Y. Zeng, and T. Chen, "Back propagation neural network with adaptive differential evolution algorithm for time series forecasting," *Expert Systems with Applications*, vol. 42, no. 2, pp. 855–863, 2015.
- [151] J. Singh and P. Tripathi, "Time series forecasting using back propagation neural network with ade algorithm," *International Journal of Engineering and Technical Research*, vol. 7, no. 5, p. 265026, 2017.
- [152] A. P. Markopoulos, S. Georgiopoulos, and D. E. Manolakos, "On the use of back propagation and radial basis function neural networks in surface roughness prediction," *Journal* of Industrial Engineering International, vol. 12, no. 3, pp. 389–400, 2016.
- [153] N. Nava, T. Di Matteo, and T. Aste, "Financial time series forecasting using empirical mode decomposition and support vector regression," *Risks*, vol. 6, no. 1, p. 7, 2018.
- [154] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.
- [155] S. Selvin, R. Vinayakumar, E. Gopalakrishnan, V. K. Menon, and K. Soman, "Stock price prediction using lstm, rnn and cnn-sliding window model," in 2017 international conference on advances in computing, communications and informatics (icacci). IEEE, 2017, pp. 1643–1647.
- [156] X. Wang, K. Smith, and R. Hyndman, "Characteristic-based clustering for time series data," *Data mining and knowledge Discovery*, vol. 13, no. 3, pp. 335–364, 2006.
- [157] M. Boomija and M. Phil, "Comparison of partition based clustering algorithms," *Journal of Computer Applications*, vol. 1, no. 4, pp. 18–21, 2008.
- [158] S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah, "Time-series clustering-a decade review," *Information systems*, vol. 53, pp. 16–38, 2015.
- [159] R. Alfred *et al.*, "A genetic-based backpropagation neural network for forecasting in time-series data," in 2015 International Conference on Science in Information Technology (ICSITech). IEEE, 2015, pp. 158–163.

- [160] S. Belgacem, C. Chatelain, and T. Paquet, "Gesture sequence recognition with one shot learned crf/hmm hybrid model," *Image and Vision Computing*, vol. 61, pp. 12–21, 2017.
- [161] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [162] Y. Li and W. Ma, "Applications of artificial neural networks in financial economics: a survey," in 2010 International symposium on computational intelligence and design, vol. 1. IEEE, 2010, pp. 211–214.
- [163] R. Aguilar-Rivera, M. Valenzuela-Rendón, and J. Rodríguez-Ortiz, "Genetic algorithms and darwinian approaches in financial applications: A survey," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7684–7697, 2015.
- [164] F. Z. Xing, E. Cambria, and R. E. Welsch, "Natural language based financial forecasting: a survey," *Artificial Intelligence Review*, vol. 50, no. 1, pp. 49–73, 2018.
- [165] F. Rundo, F. Trenta, A. L. di Stallo, and S. Battiato, "Machine learning for quantitative finance applications: A survey," *Applied Sciences*, vol. 9, no. 24, p. 5574, 2019.
- [166] I. K. Nti, A. F. Adekoya, and B. A. Weyori, "A systematic review of fundamental and technical analysis of stock market predictions," *Artificial Intelligence Review*, vol. 53, no. 4, pp. 3007–3057, 2020.
- [167] D. Shah, H. Isah, and F. Zulkernine, "Stock market analysis: A review and taxonomy of prediction techniques," *International Journal of Financial Studies*, vol. 7, no. 2, p. 26, 2019.
- [168] J.-Z. Wang, J.-J. Wang, Z.-G. Zhang, and S.-P. Guo, "Forecasting stock indices with back propagation neural network," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14346–14355, 2011.
- [169] E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10389–10397, 2011.

- [170] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.
- [171] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "A comparison of arima and lstm in forecasting time series," in 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2018, pp. 1394–1401.
- [172] M. Asiful, R. K. Hossain, R. THulasiram, N. D. Bruce, and Y. Wang, "Hybrid deep learning model for stock price prediction," in *IEEE Symposium Series on Computational Intelligence, SSCI, Bangalore, India*, 2018.
- [173] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," *arXiv preprint arXiv:1605.07678*, 2016.
- [174] H. Maqsood, I. Mehmood, M. Maqsood, M. Yasir, S. Afzal, F. Aadil, M. M. Selim, and K. Muhammad, "A local and global event sentiment based efficient stock exchange forecasting using deep learning," *International Journal of Information Management*, vol. 50, pp. 432–451, 2020.
- [175] T.-T. Nguyen and S. Yoon, "A novel approach to short-term stock price movement prediction using transfer learning," *Applied Sciences*, vol. 9, no. 22, p. 4745, 2019.
- [176] L. O. Orimoloye, M.-C. Sung, T. Ma, and J. E. Johnson, "Comparing the effectiveness of deep feedforward neural networks and shallow architectures for predicting stock price indices," *Expert Systems with Applications*, vol. 139, p. 112828, 2020.
- [177] M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, and E. Salwana, "Deep learning for stock market prediction," *Entropy*, vol. 22, no. 8, p. 840, 2020.
- [178] N. Naik and B. R. Mohan, "Stock price movements classification using machine and deep learning techniques-the case study of indian stock market," in *International Conference* on Engineering Applications of Neural Networks. Springer, 2019, pp. 445–452.
- [179] C. Yang, J. Zhai, and G. Tao, "Deep learning for price movement prediction using convolutional neural network and long short-term memory," *Mathematical Problems in Engineering*, 2020.

- [180] H. N. Bhandari, B. Rimal, N. R. Pokhrel, R. Rimal, K. R. Dahal, and R. K. Khatri, "Predicting stock market index using lstm," *Machine Learning with Applications*, p. 100320, 2022.
- [181] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Applied Soft Computing*, vol. 70, pp. 525–538, 2018.
- [182] E. Hoseinzade and S. Haratizadeh, "Cnnpred: Cnn-based stock market prediction using a diverse set of variables," *Expert Systems with Applications*, vol. 129, pp. 273–285, 2019.
- [183] M. Tkáč and R. Verner, "Artificial neural networks in business: Two decades of research," *Applied Soft Computing*, vol. 38, pp. 788–804, 2016.
- [184] R. C. Cavalcante, R. C. Brasileiro, V. L. Souza, J. P. Nobrega, and A. L. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Expert Systems with Applications*, vol. 55, pp. 194–211, 2016.
- [185] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniquespart ii: Soft computing methods," *Expert Systems with applications*, vol. 36, no. 3, pp. 5932–5941, 2009.
- [186] J. Moody and M. Saffell, "Learning to trade via direct reinforcement," *IEEE transactions on neural Networks*, vol. 12, no. 4, pp. 875–889, 2001.
- [187] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 3, pp. 653–664, 2016.
- [188] Z. Jiang, D. Xu, and J. Liang, "A deep reinforcement learning framework for the financial portfolio management problem," *arXiv preprint arXiv:1706.10059*, 2017.
- [189] J. Wang, Y. Zhang, K. Tang, J. Wu, and Z. Xiong, "Alphastock: A buying-winnersand-selling-losers investment strategy using interpretable deep reinforcement attention networks," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 1900–1908.

- [190] R. AbdelKawy, W. M. Abdelmoez, and A. Shoukry, "A synchronous deep reinforcement learning model for automated multi-stock trading," *Progress in Artificial Intelligence*, vol. 10, no. 1, pp. 83–97, 2021.
- [191] X. Wu, H. Chen, J. Wang, L. Troiano, V. Loia, and H. Fujita, "Adaptive stock trading strategies with deep reinforcement learning methods," *Information Sciences*, vol. 538, pp. 142–158, 2020.
- [192] T. Kabbani and E. Duman, "Deep reinforcement learning approach for trading automation in the stock market," 2022.
- [193] A. Mosavi, Y. Faghan, P. Ghamisi, P. Duan, S. F. Ardabili, E. Salwana, and S. S. Band, "Comprehensive review of deep reinforcement learning methods and applications in economics," *Mathematics*, vol. 8, no. 10, p. 1640, 2020.
- [194] A. Charpentier, R. Elie, and C. Remlinger, "Reinforcement learning in economics and finance," *Computational Economics*, pp. 1–38, 2021.
- [195] M. Hiransha, E. A. Gopalakrishnan, V. K. Menon, and K. Soman, "Nse stock market prediction using deep-learning models," *Proceedia computer science*, vol. 132, pp. 1351– 1362, 2018.
- [196] T. Sun and M. A. Vasarhalyi, "Predicting credit card delinquencies: An application of deep neural networks," in *Handbook of Financial Econometrics, Mathematics, Statistics, and Machine Learning.* World Scientific, 2021, pp. 4349–4381.
- [197] S. Galeshchuk and S. Mukherjee, "Deep networks for predicting direction of change in foreign exchange rates," *Intelligent Systems in Accounting, Finance and Management*, vol. 24, no. 4, pp. 100–110, 2017.
- [198] S. Sinha, S. Mishra, V. Mishra, and T. Ahmed, "Sector influence aware stock trend prediction using 3d convolutional neural network," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 4, pp. 1511–1522, 2022.
- [199] S. McNally, J. Roche, and S. Caton, "Predicting the price of bitcoin using machine learning," in 2018 26th euromicro international conference on parallel, distributed and network-based processing (PDP). IEEE, 2018, pp. 339–343.

- [200] J. Sirignano and R. Cont, "Universal features of price formation in financial markets: perspectives from deep learning," *Quantitative Finance*, vol. 19, no. 9, pp. 1449–1459, 2019.
- [201] A. Nayak, M. M. Pai, and R. M. Pai, "Prediction models for indian stock market," *Procedia Computer Science*, vol. 89, pp. 441–449, 2016.
- [202] A. Sun, M. Lachanski, and F. J. Fabozzi, "Trade the tweet: Social media text mining and sparse matrix factorization for stock market prediction," *International Review of Financial Analysis*, vol. 48, pp. 272–281, 2016.
- [203] S. Fischer and R. C. Merton, "Macroeconomics and finance: The role of the stock market," in *Carnegie-Rochester conference series on public policy*, vol. 21. Elsevier, 1984, pp. 57–108.
- [204] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008* ACM SIGMOD international conference on Management of data, 2008, pp. 1247–1250.
- [205] J. Long, Z. Chen, W. He, T. Wu, and J. Ren, "An integrated framework of deep learning and knowledge graph for prediction of stock price trend: An application in chinese stock exchange market," *Applied Soft Computing*, vol. 91, p. 106205, 2020.
- [206] C.-F. Tsai and Z.-Y. Quan, "Stock prediction by searching for similarities in candlestick charts," ACM Transactions on Management Information Systems (TMIS), vol. 5, no. 2, pp. 1–21, 2014.
- [207] R. M. I. Kusuma, T.-T. Ho, W.-C. Kao, Y.-Y. Ou, and K.-L. Hua, "Using deep learning neural networks and candlestick chart representation to predict stock market," *arXiv* preprint arXiv:1903.12258, 2019.
- [208] G. V. Attigeri, M. P. MM, R. M. Pai, and A. Nayak, "Stock market prediction: A big data approach," in *TENCON 2015-2015 IEEE Region 10 Conference*. IEEE, 2015, pp. 1–5.
- [209] Z. Liu, C. K. Loo, and K. Pasupa, "Recurrent kernel online sequential extreme learning machine with kernel adaptive filter for time series prediction," in 2017 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2017, pp. 1–7.

- [210] M. Yukawa, "Multikernel adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 60, no. 9, pp. 4672–4682, 2012.
- [211] Z. Liu, C. K. Loo, K. Pasupa, and M. Seera, "Meta-cognitive recurrent kernel online sequential extreme learning machine with kernel adaptive filter for concept drift handling," *Engineering Applications of Artificial Intelligence*, vol. 88, p. 103327, 2020.
- [212] Y. Zheng, S. Wang, J. Feng, and K. T. Chi, "A modified quantized kernel least mean square algorithm for prediction of chaotic time series," *Digital Signal Processing*, vol. 48, pp. 130–136, 2016.
- [213] S. Garcia-Vega, X.-J. Zeng, and J. Keane, "Stock price prediction using kernel adaptive filtering within a stock market interdependence approach," *Available at SSRN*, 2018.
- [214] B. Oancea and Ş. C. Ciucu, "Time series forecasting using neural networks," arXiv preprint arXiv:1401.1333, 2014.
- [215] I. K. Nti, A. F. Adekoya, and B. A. Weyori, "A comprehensive evaluation of ensemble learning for stock-market prediction," *Journal of Big Data*, vol. 7, no. 1, pp. 1–40, 2020.
- [216] Q. Wang, W. Xu, X. Huang, and K. Yang, "Enhancing intraday stock price manipulation detection by leveraging recurrent neural networks with ensemble learning," *Neurocomputing*, vol. 347, pp. 46–58, 2019.
- [217] Y. Hari and L. P. Dewi, "Forecasting system approach for stock trading with relative strength index and moving average indicator," Ph.D. dissertation, Petra Christian University, 2018.
- [218] S. Lauren and S. D. Harlili, "Stock trend prediction using simple moving average supported by news classification," in 2014 International Conference of Advanced Informatics: Concept, Theory and Application (ICAICTA). IEEE, 2014, pp. 135–139.
- [219] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 2, no. 1, pp. 86– 97, 2012.

- [220] E. K. Ampomah, Z. Qin, and G. Nyame, "Evaluation of tree-based ensemble machine learning models in predicting stock price direction of movement," *Information*, vol. 11, no. 6, p. 332, 2020.
- [221] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [222] S. Mannor, D. Peleg, and R. Rubinstein, "The cross entropy method for classification," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 561–568.
- [223] O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu, "An artificial neural network-based stock trading system using technical analysis and big data framework," in *proceedings of the southeast conference*, 2017, pp. 223–226.
- [224] R. Singh, T. Ahmed, A. Kumar, A. K. Singh, A. K. Pandey, and S. K. Singh, "Imbalanced breast cancer classification using transfer learning," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 18, no. 1, pp. 83–93, 2020.
- [225] H. Xu, C. Caramanis, and S. Mannor, "Sparse algorithms are not stable: A no-free-lunch theorem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 1, pp. 187–193, 2011.
- [226] W. Liu and J. C. Príncipe, "Kernel affine projection algorithms," EURASIP Journal on Advances in Signal Processing, vol. 2008, pp. 1–12, 2008.
- [227] S. Zhao, B. Chen, and J. C. Principe, "Kernel adaptive filtering with maximum correntropy criterion," in *The 2011 International Joint Conference on Neural Networks*. IEEE, 2011, pp. 2012–2017.
- [228] S. K. Ahn and G. C. Reinsel, "Estimation for partially nonstationary multivariate autoregressive models," *Journal of the American statistical association*, vol. 85, no. 411, pp. 813–823, 1990.
- [229] T. Ouyang, H. Huang, Y. He, and Z. Tang, "Chaotic wind power time series prediction via switching data-driven modes," *Renewable Energy*, vol. 145, pp. 270–281, 2020.

- [230] N. S. Exchange. (2021) Nifty-50 Database. [Online]. Available: https://www.kaggle. com/rohanrao/nifty50-stock-market-data
- [231] P. Gao, R. Zhang, and X. Yang, "The application of stock index price prediction with neural network," *Mathematical and Computational Applications*, vol. 25, no. 3, p. 53, 2020.
- [232] A. Moghar and M. Hamiche, "Stock market prediction using lstm recurrent neural network," *Procedia Computer Science*, vol. 170, pp. 1168–1173, 2020.
- [233] A. H. Moghaddam, M. H. Moghaddam, and M. Esfandyari, "Stock market index prediction using artificial neural network," *Journal of Economics, Finance and Administrative Science*, vol. 21, no. 41, pp. 89–93, 2016.
- [234] M. R. Hassan, "A combination of hidden markov model and fuzzy model for stock market forecasting," *Neurocomputing*, vol. 72, no. 16-18, pp. 3439–3446, 2009.
- [235] H. Gündüz, Z. Çataltepe, and Y. Yaslan, "Stock daily return prediction using expanded features and feature selection," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 25, no. 6, pp. 4829–4840, 2017.
- [236] S. Mishra, T. Ahmed, V. Mishra, S. Bourouis, and M. A. Ullah, "An online kernel adaptive filtering-based approach for mid-price prediction," *Scientific Programming*, 2022.
- [237] J. Agrawal, V. Chourasia, and A. Mittra, "State-of-the-art in stock prediction techniques," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 2, no. 4, pp. 1360–1366, 2013.
- [238] B. M. Henrique, V. A. Sobreiro, and H. Kimura, "Stock price prediction using support vector regression on daily and up to the minute prices," *The Journal of finance and data science*, vol. 4, no. 3, pp. 183–201, 2018.
- [239] R. A. Kamble, "Short and long term stock trend prediction using decision tree," in 2017 International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2017, pp. 1371–1375.