APPLICATION OF MACHINE LEARNING AND DEEP LEARNING IN US HOUSING MARKETS

by

Satyanshu Kumar

A Dissertation Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

in Economics

at

The University of Wisconsin-Milwaukee

May 2024

ABSTRACT

APPLICATION OF MACHINE LEARNING AND DEEP LEARNING IN US HOUSING MARKETS

by

Satyanshu Kumar

The University of Wisconsin-Milwaukee, 2024
Under the Supervision of Professor Kundan Kishor

This dissertation explores the utilization of machine learning (ML), deep learning (DL), and social media data analysis to improve the accuracy of predicting stock returns in the housing market. The dissertation aims to investigate different models and data sources in order to enhance the accuracy of forecasting, surpassing the effectiveness of previous methods.

The first chapter of dissertation investigates the efficacy of machine learning models in predicting housing values at different time intervals. The system utilizes various datasets containing home prices and optimizes hyperparameters using cross-validation. The findings indicate that machine learning models surpass traditional time series models, especially when it comes to longer forecast periods. When it comes to shorter forecasts, the performance varies. Simple models like ARMA demonstrate similar or slightly inferior outcomes compared to more complicated machine learning models.

In the second hcapter the research is expanded to include Real Estate Investment Trusts (REITs) by utilizing both Machine Learning (ML) and Deep Learning (DL) models to forecast changes in prices. The daily data from many categories of Real Estate Investment Trusts (REITs), such as office, retail, and data centers, is used for the time period spanning from October 30, 2015, to October 13, 2023. The study utilizes many models such as Random Forest, XGBoost, LightGBM, Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTMs), Gated Recurrent Units (GRUs), and Bidirectional LSTMs (Bi-LSTMs). The results indicate that machine

learning (ML) models outperform deep learning (DL) models in terms of forecasting accuracy, as evidenced by their lower Root Mean Squared Errors. Significantly, both machine learning (ML) and deep learning (DL) models surpass a simplistic random walk baseline in terms of performance.

While the third chapter of dissertation explores the potential of utilizing social media data, particularly semantic information collected from Twitter, to enhance the accuracy of predictions for homebuilder stock returns, building upon the achievements of machine learning models. The study evaluates the forecasting accuracy of features extracted from Twitter in comparison to the widely employed macroeconomic indicator, the daily mortgage rate. There are nine prominent machine learning models used for prediction. The results indicate that all models based on features perform better than the random walk baseline, emphasizing the importance of including extra data. Significantly, the utilization of Twitter sentiment elements results in increased prediction accuracy for four out of six homebuilders when compared to solely relying on the mortgage rate. These findings indicate that analyzing social media sentiment can be a helpful method for capturing market sentiment and enhancing predictions of stock returns in the homebuilding industry.

Overall, this dissertation adds value to the study of housing market prediction by demonstrating the efficacy of machine learning models and the potential of social media data to improve forecasting accuracy. The findings offer useful insights for investors, analysts, and researchers who aim to enhance stock return projections in the housing market.

To

my parents.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Forecasting House Prices Using Machine Learning Models

1.1 Introduction

When discussing financial asset prices, the widely accepted Efficient Market Theory (Fama, 1970) comes to mind, asserting that these prices reflect all publicly available information at all times. The author argues that investors react to sudden price changes by selling or purchasing until asset overvaluation or undervaluation corrects itself. Recent research has scrutinized this theory, and it has garnered criticism.

According to the theory, investors can sell assets short to rectify overpricing. However, housing, being a real and illiquid asset, lacks such a mechanism for shorting residential homes during housing price surges. Therefore, if the housing market adheres to the Efficient Market Theory, the increase in housing prices since 2012 should be justified by new information, even if it remains undisclosed. Consequently, a substantial body of literature in the housing market has concentrated on identifying these housing market indicators and selecting appropriate models to enhance predictive modeling for financial time series data. One way to address this problem is by the introduction of semantic features.

Leamer (2007) argued that housing market predictors are more equipped indicators

to inform us about the state of the economy. He showed that eight out of ten successful predictions of recessions post-World War II were made using these indicators. Vargas-Silva (2008) showed that housing price adjustments played an essential role in determining the phase of the business cycle, stating, "Housing is a Business Cycle." According to Vargas, during economic expansion, house prices rise due to increased housing construction and labor demands. However, during recessions, homeowners do not reduce prices to match decreased demand caused by reduced personal income. Thus, limiting sales volume becomes the only option to curtail demand and lower housing prices. Therefore, housing constructions also play a significant role in GDP and explain much of economic well-being.

Despite historically low borrowing rates in the US, which have driven housing prices up since the 2008 crisis and saw their most significant growth of 18.8% last year (2021), this cannot be the sole reason for the growth. This housing market growth has led industry giants like Zillow to project a bullish housing market in 2023, claiming a 16.6% increase. However, according to a report by Staff (2022), foreclosure filings have been continuously increasing for the past eight months and are expected to rise in 2023. Foreclosure rates have surged by 129% since February of the previous year, a trend likely to continue as pandemic-related moratoriums and foreclosure restrictions were lifted on July 31, 2021. Housing price trends undoubtedly influence individuals' investment decisions and choices to live and work in specific areas, making accurate house price trend forecasts crucial for both market participants and policymakers. To achieve accurate and robust forecasting of economic and financial time series data, models such as AR, VAR, VECM, and their variations have gained wider acceptance in various use cases.

Time series models serve multiple purposes, including predicting future outcomes, understanding past outcomes, making policy recommendations, and more. Rapid advancements in computing have facilitated the analysis of large, complex non-linear models for forecasting using machine learning and deep learning techniques. This paper aims to construct a forecast comparison between hyper-parameter-tuned machine learning forecast models and the base random walk model to determine if ongoing

financial series, such as housing prices, exhibit any improvements.

While a substantial body of literature exists on forecast model comparison for housing data, this paper aims to fill the gap by comparing ML models and classical models with the naive model as the base. Given the volatility during the period under analysis for forecasting financial time series, such as the housing price index, this comparison provides valuable insights into whether we should explore ML/Classical models or continue using the naive model. Additionally, we conduct a forecast comparison using four distinct HPI datasets of the United States, including both public and private data. Lastly, we attempt to perform a comparison of tuned ML models using cross-validated train-test data to represent the most optimized models. This extensive comparison includes 30 models, consisting of one 'naive model,' 12 classical time series forecasting models, and 17 machine learning models.

In this paper, we conduct a comprehensive analysis comparing the performance of machine learning (ML) and classical models in multi-step ahead forecasts using various datasets. The study reveals interesting findings: ML models excel in 3-step forecasts, while classical models outperform in 6-step forecasts, only for ML models to regain superiority in 12-step forecasts. Despite not being the most popular choice, ML models consistently rank among the top 5 performers. Furthermore, the study highlights that ML models are particularly effective on longer time series datasets, indicating their ability to detect complex patterns. Most models provide rapid results, with exceptions such as TBATS, BATS, and Auto-Arima, which are slower and less accurate, leading the author to recommend avoiding these classical models in forecasting tasks.

The research presents three primary findings, indicating that the utilization of cross-validation leads to a substantial enhancement in forecast performance compared to the conventional approach of employing a simple train-test split at an arbitrary position. The utilization of the hedonic price method for calculating house price indices results in reduced volatility and decreased forecast error distortion. In the context of forecasting financial time series data, the utilization of naive models for non-hedonic price indexes is found to be more advantageous compared to machine learning models

during periods of volatility.

The following sections of the paper discuss a summary of the literature review on forecasting methods used for house price forecasting, different housing market data indexes, the data used in this analysis, the methodology adopted for this experiment, results, and conclude.

1.2 Literature Review

The literature on the forecasting performance of the housing market can be summarized below based on different predictive models and their comparisons with each other.

There have been studies focused on single-model performance.Rapach and Strauss (2007) Conducted real housing price growth of the Federal Reserve's Eighth District using ARDL (Autoregressive distributed lag) model framework and showed improvement in performance with selected features like state housing price-to-income ratio, state unemployment rate, and national inflation rate. Later, Rapach and Strauss (2009) extended this study to 20 other U.S. states and regions and saw the ARIMA model outperformed the ARDL model for the coastal and interior states. Gupta and Das (2010) used BVAR (Bayesian Vector Autoregression) on 20 US-states to forecast downturn over 2007:1 to 2008:1. The model is well equipped with forecasting future directions but underestimated the decline in house prices due to a lack of information on fundamentals in the estimation process. Gupta et al. (2011) later expanded forecasting HPI index to structural and non-structural models with and without economic fundamentals having BVAR and DFM (Dynamic Factor Modelling). Further, they also examined the explanatory power of these economic fundamentals towards housing using theoretical models like VAR, BVAR, FAVAR, and BFAVAR. The small-scale Bayesian-Shrinkage model with ten variables outperformed large-scale BVARs, but DSGE models were the only model able to forecast downturns with any accuracy.

Some studies focused on comparing several ML model performances are mentioned

below. Park and Bae (2015) carried house price forecast for Fair Fax County, Virginia, using four machine learning models (C4.5, Ripper, Naive-Bayesian, AdaBoost) and compared their prediction capabilities where RIPPER outperformed all other ML models. Ho et al. (2021) made a forecast comparison between SVM, RF, and GBM (gradient boosting machine) and found RF and GBM to have better prediction results than SVM. Finally, Milunovich (2020) ran a comparative forecast comparison of 47 machine and deep learning models on the Australian real house price index. SVR was able to outshine six out of the eight best forecasts, while the other two best performers came from forecast combinations. Linear AR and VAR models produced accurate one-period ahead forecasts, while ML and DL methods were suitable in medium and long-run forecast horizons.

There also have been studies focused on comparing several ML model performances with traditional and linear models namely. Plakandaras et al. (2015) created a novel hybrid forecast methodology by combining EEMD (Ensemble Empirical Mode Decomposition) with SVM and used ElNet for feature selection and compared forecast results with RW (random walk) and BVAR. The proposed model outperformed traditional models with half the error of the random walk model, making it a robust model as an early warning system to detect a sudden drop in house prices. Li et al. (2017) forecasted stocks and REITs using the group method of data handling (GMDH) Neural Network model, which outperformed linear and traditional models like single/double exponential smoothing, ARIMA, and back-propagation NN.Lim et al. (2016) further showed superior performance of ANN (Artificial Neural Network) over ARIMA on Singapore condominium price index.

A subsection of literature on improving housing forecasts using model combinations also exists. Taffese (2007) used a combination of CBR (Case-Based Reasoning) and Neural Networks AI models to get better forecast results. Whereas Drought and McDonald (2011) used the idea of combining forecast results with weights based on the out-of-sample error performance of all forecasting models used.Wei and Cao (2017) used a dynamic Model Averaging (DMA) to forecast 30 Chinese cities and outperformed

other model averaging methods like Bayesian Model Averaging (BMA), Information-theoretic Model averaging (ITMA), and equal-weighted averaging in both recursive and rolling forecasting modes.

There also have been studies focused on Neural network model performances for the housing market. Wilson et al. (2002) One of the initial approaches toward modelling using Neural Networks, the Residential property price index, was forecasted using ANN and had mixed results. Piao et al. (2019) modelled a novel CNN model to forecast house prices and showed a much better approach to the feature selection process using this method.Yasnitsky et al. (2021) model is based on a neural network trained on examples considering both construction and operational characteristics, as well as geographical and environmental characteristics, along with time-changing macroeconomic parameters that describe the economic state of a specific region, country, and the world.

There also exists a vast literature on SVM-based models for housing forecasts. Gu et al. (2011) created a combination of genetic algorithm and SVM(G-SVM) on the Chinese housing market and found this model to give superior results even compared to Grid Algorithm (GA)-SVM as the former showed better performance and needed less computing power. Wang et al. (2014) exploited this idea and went beyond genetic and grid algorithms and introduced particle swarm optimization (PSO) to determine parameters of SVM, and this proposed model showed better forecast performance. Phan (2018) also conducted the same experiment for the Australian house price market using step-up and PCA for feature selection followed by SVM. Li et al. (2009) also conducted a house price forecast using SVR and BPNN (Back-Propagation Neural Network) and found SVR performance to be better than later.

Some papers which worked on ensemble learners are mentioned below. Shahhosseini et al. (2019) proposed an optimization model of ensembles that tries to minimize bias and variance on the Boston and Ames dataset. The model result was very competitive in predicting future trends of both data sets. Li et al. (2020) further build on the idea of Plakandaras et al. (2015) and proposed a 4-step forecasting model. Firstly, they decomposed the data into three components: short-term fluctuation (originating from

normal market disequilibrium of supply and demand), the effect of extreme events, and long-term trends. Then it passes through a fine-to-coarse reconstruction algorithm followed by ARIMA and SVM and has shown superior prediction capabilities for housing time series.

Finally, some literature which works on hybrid models to study housing forecast performances also exists. Wu et al. (2009) proposed a novel prediction system based on hybrid genetic-based support-vector regression (HGA-SVR) with feng shui theories and outperformed models like backpropagation NN and Fuzzy Neural Networks (FNN). Sarip

Sarip et al. (2016) proposed and applied FLSR (Fuzzy Least Square Regression-based) to predict real estate prices for the first time using this model in property price prediction literature. The model projected better prediction capacity compared to ANN (Artificial Neural Network) and FIS (Fuzzy Inference systems)

Later Ge et al. (2019) used an LSTM dense NN (Multi-Task Hierarchical Graph Representation Learning (MugRep)) framework for accurate real estate appraisal. MugRep framework approach for housing price forecast outperformed statistical and Machine Learning models (LR, SVR, GBRT) and improved the state-of-the-art PDVM model based on ANN. Moreover, it tells us about the advantage of using ANN models for financial data having non-linear properties. Finally, Liu and Wu (2020) obtained better prediction results when modified Holt exponential smoothing (MHES) was applied to historical house price data with the Whale Optimization Algorithm (WOA) to obtain optimal parameters for the model. The WOA-MHES can be applied effectively to house price forecast literature as the model showed minor prediction errors and shorter computation time.

Rafiei and Adeli (2016) introduced a method to correctly predict the price of the house at the time of construction using DRBM (Deep Belief Restricted Boltzmann machine) and did a dimensionality reduction of the large dataset using NGA (Non-mating Genetic algorithm). This NGA-induced DRBM model helped reduce error rates from 10.3% - 64.8% to 3.7%. They used feature selection methods of standard genetic search, best

first, linear forwards selection, and correlated-based feature subsets using SVM, Bagging, and Naive Bayes models as a classifier. Hence the model is better suited for learning high-dimensional complicated pattern recognition problems.

1.3 Data

House price Indexes not only help us in investigating economic outlook (analyses of the housing bubble and indicators for housing affordability) or better understand housing markets (factors influencing housing prices and studying the housing business cycle). Some recent research has also concluded the possibility of hedging housing risk(Hinkelmann and Swidler (2008), Bertus et al. (2008), Bao et al. (2020)) which requires accurate price indexes for housing. Hence understanding these house price indexes either from academic research or investment point of view is crucial, and there is a need to deep dive further into these indexes

Since there are several methods for constructing these housing price indexes, three methods are commonly used (the hedonic method, repeated sales method, and hybrid method). Weinberg (2014) summarizes all such data available and sourced by the government and later published a second paper Weinberg (2015) summarizing all such housing data sourced by private institutions.

The commonly used government-sourced U.S. housing data to analyze the housing market in empirical research are:

- FHFA (Federal Housing Finance Administration) - only homes with mortgages reported to Freddie Mac and Fannie Mae are included (excluding jumbo loans)
- Case-Shiller index - based on the repeat sale index of FHFA, this series also includes data on houses purchased via subprime and other conventional loans. But we don't have data from 13 states and incomplete coverage on 29 states.
- NAR (National Association of Realtors) provides the median purchase price for homes directly using a fixed subset of its 2000 associations.

- FMHPI (Freddie Mac price index) - based on conventional conforming mortgages for single-unit residential houses that

  were purchased or securitized by Freddie Mac or Fannie Mae.

- CCPI (Commercial Property Price Index) - index with information on several property types (Office, Residential, Industrial, Retail, Commercial)

Data sources from private sectors:

- NAHB (National Association of Home Builders) - On a subscription basis provide more than 40 data series for housing state in the region including (Building material prices like lumber cost, multifamily production and vacancy indexes)

- NAR and NAHB also provide housing affordability indexes

- CoreLogic, Zillow, Black Knight Financial Services provides property level data

- RealtyTrac - Data set includes foreclosure information covering 100 million homes in 2,200 counties

As we are familiar with, there exist empirical challenges in collecting these financial data at any aggregation level, making housing price predictive analysis challenging and their accuracy problematic. The standard method of creating these housing data indexes is the repeated sales method (Case-Shiller HPI) which uses average sales prices of sold housing and reports a three-month moving average having a lag of 2 months in reporting. It also does not capture the value of new homes in its index until it gets resold and shows up in the repeated sale index. Another approach is to use the median sale price index to construct these series (FHFA index). However, these indexes are biased as they do not account for jumbo loans acquired to make these housing purchases, leading to a sample selection problem since low-cost houses get sold more frequently. In order to get around these issues, the hedonic regression approach is used to adjust these sales prices based on the market pricing of these attributes using transactional data (Zillow home value index). Given the issues with data on house prices, there is a need for better robust house price indexes.

The literature on creating a new house price index is not recent, and people in the past have created new indexes to overcome these issues Barr et al. (2017)-Dorsey et al. (2010)-Baroni et al. (2007). The American Enterprise Institute's HPA (National House Price Appreciation Index) data is based on a 'quasi' repeat sale index with the hedonic element. The construction of this index includes artificial sales pairs of actual sale 'artificial' sales measured by the property's automated valuation model (AVM), making the best use of both repeated sales index (Case-Shiller, FHFA) and hedonic (Zillow, realtor). We will use the AEI HPA index against Case-Shiller, FHFA and Zillow housing indexes to compare for a better modelling performance in housing forecast literature.

For our analysis, we will choose four price indexes from both sources to compare model results and see if any changes are due to differences in data collecting seeds. For the analysis, we chose the Zillow Home Value Index (ZHVI) and AEI's Home Price Appreciation (HPA) Indexes from privately sourced data as the housing price index. And choose Case-Shiller (CS) and FHFA (Federal House Price Index) for the government-sourced data series. The CS index contains foreclosure data on houses. However, it has only actual house transactions and does not include refinance-appraisals (included in the FHFA index) which leads to "appraisal smoothing bias"Edelstein and Quan (2006). Therefore, it will be interesting to see how these two data perform in predictive modelling. Chicago Mercantile Exchange (CME) also uses the CS index for housing futures and options; hence we consider this index as the base index.

We take log differences for these data (growth rate) for our evaluation except for the Zillow index where we take log seasonal differences for all other indexes. One significant advantage of taking log-difference is to make the series stationary and get a linear trend, as most macroeconomic time series are tied to population growth and may have an exponential trend which can lead to spurious regression results. We apply random hyperparameter optimization to ML models in order to optimize them, along with 5-fold CV for training the data and report the percentage improvement in root mean squared error (rmse) in comparison to the naive model.

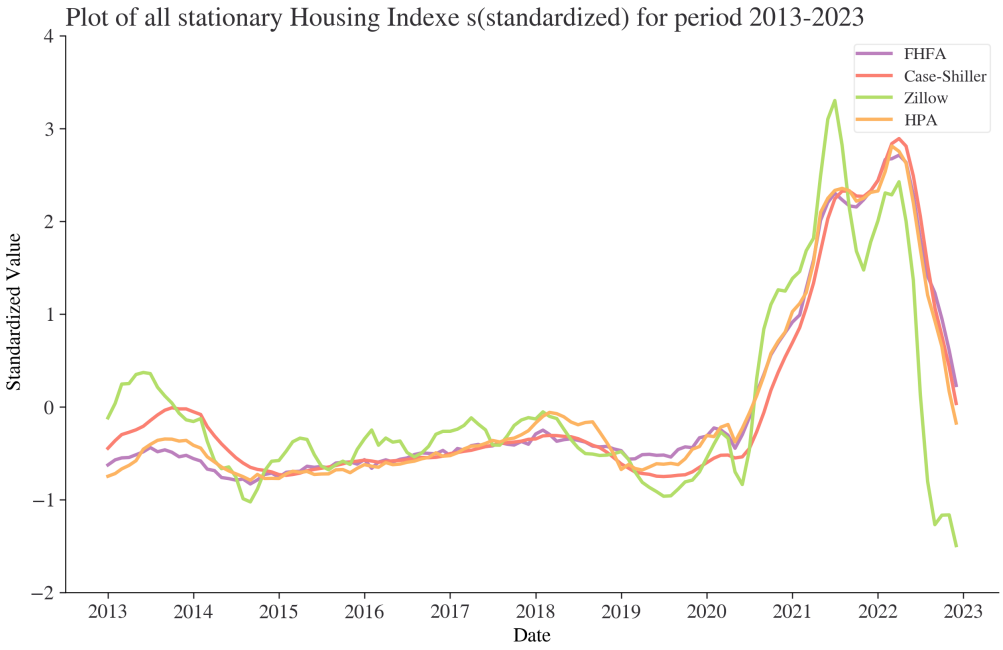


Figure 1.1: Plot of all 4 housing Indexes

Table 1.1: Data Description

Indexes	Start	Obs.	Mean	Median	SD	ADF(p-value)	KPSS(p-value)	Data Transformation
Case-Shiller	Feb-87	431	0.0035	0.0036	0.0054	0.0124	0.033	log(diff (1))
FHFA	Feb-92	383	0.0035	0.004	0.005	0.0083	0.01	log(diff (1))
Zillow	Feb-00	275	0.0037	0.0047	0.0056	0.0332	0.01	log(diff (1))
AEI's HPA	Jan-13	120	12.1071	8.836	8.0268	0.0365	0.01	diff (12)

1.4 Methodology

1.4.1 Model Specification

As Zhang et al. (2013) paper examines whether the housing index is stationary data and whether the intertemporal evolution of log house price growth can be considered a stationary process. The article questioned the validity of analyses of house prices based on cointegration and error correction models. The housing literature considers the data to be trend stationary or weakly stationary and applies the classical and ML models to the housing dataset. The widely used methods to check for data stationery can be summarized below in the table, and the results of the unit root test for our dataset are reported below.

Table 1.2: Various Stationarity Test

Methods	Tests	Reference
Visualization	ACF and PACF plots	
Parametric tests	The Dickey-Fuller Test	Said and Dickey (1984)
	Phillips-Perron Testing	Phillips and Perron (1988)
	KPSS Testing	Kwiatkowski et al. (1992)
	Zivot-Andrew's test	Zivot and Andrews (2002)
Non-Parametric tests	Variance Ratio tests	Breitung (2002)

Financial forecasting can be considered an example of a challenging signal processing problem due to small sample sizes, collection methods, high noise, non-stationarity, and non-linearity issues concerns. I propose three approaches to address this problem and improve predictive modelling for the Housing market. i) Checking the performance of the widely used and respected Case-Shiller Index with more robust looking AEI's HPA Index in modelling performance and comparing it with the other two indexes. ii) Make a model forecast comparison of classical and machine learning models with base models by allowing for hyper-parameter tuning for ML models using optimization techniques and conditional seasonal decomposition of data wherever needed.

The non-stationary time series predictive modelling becomes more challenging as the knowledge for transforming data and forecast models is required. Furthermore, since data preprocessing is a crucial step for time series forecasting, it needs data to be stationary

(i.e., statistical properties such as mean and standard deviation do not change over time) and have uniform volatility. However, since there is no silver bullet for which data transformation method is best, we take either log differenced or differenced series in the range where they are stationary to apply classical statistical models for this experiment.

1.4.2 Forecast horizon and multi-step ahead forecast strategy

Rolling multi-step forecasting is used to evaluate forecasts that are three, six, and twelve steps ahead. It's vital to remember that the outcomes vary depending on the multi-step forecasting approach you employ. My prediction horizon differs in each of the three scenarios since I am using a distinct multi-step forecast with 5-fold cross-validation steps to select the train-teat split. My out-of-sample period for a three-step recursive forecast is from September to November 2022, and the validation period is divided into three chunks of three months, from June to August 2022, Mar to May 2022, and Dec 21 to Feb 22. My out-of-sample time frame for a six-step forecast is from June to November 2022. The cross-validation periods are December 21–June 22, June–November 21, and December 20–June 21. Finally, my out-of-sample period for the 12-step forecast is from December 21 to November 22. And my CV's dates are December 20, November 21, December 19, and December 19.

The Recursive method is the most straightforward method for multi-step forecasting; thus I'm sticking with it for my analysis. It operates by training a single model for forecasting one step forward. That is, assuming what would happen next. The model is then iterated using its prior forecasts to obtain predictions for a number of steps. The appeal of the rolling strategy is that you only need one model to cover the entire forecasting horizon. Also, you are not required to fix the forecasting horizon in advance. However, there are significant disadvantages. Errors spread when the same model is iterated with its own forecasts as input. As a result, long-term forecasts perform worse when forecasting.

Other multi-step forecasting techniques include:

I) Direct

For each horizon, the Direct method creates a single model.

This method prevents error propagation because no model iterations are required. But there are also some negatives. The additional models call for increased computing power. Furthermore, it presumes the independence of each horizon. This presumption frequently yields subpar outcomes.

II) Direct Recursive (Chaining)

Direct Recursive, as its name suggests, aims to combine the concepts of Direct and Recursive. Each horizon has a model created for it (following Direct). Yet, the input data grows with the preceding model's prediction at each stage (following Recursive).

III) DaD (Data as Demonstrator)

DaD is a multi-step forecasting meta-learning system. It makes an effort to reduce the Recursive error propagation issue. The goal is to correct errors made during multi-step prediction using the training set. These corrections are iteratively added to the training set. After that, utilizing the enriched training data, a recursive technique is used.

IV) DFML (Dynamic factor Machine Learning)

The multivariate time series is the focus of the DFML approach. Yet univariate ones can also benefit from its concepts.

The plan is to use a dimensionality reduction technique to preprocess the time series (e.g. PCA). Hence, you would only need to forecast a small number of hidden variables rather than H values. The forecasts can then be returned to their original dimension by performing a transformation reversal.

V) Multi Output

The approaches that have been discussed thus far are single-output strategies; they model one horizon at a time. They fail to take into account the dependence between

several horizons, which may be a limitation. Better multi-step forecasts may depend on capturing this relationship.

Multi-output models offer a solution to this problem. They correspond to a single model that jointly learns all predicting horizons. Typically, the output of learning algorithms only requires one variable. The target variable is the name given to this variable. Yet, some algorithms can logically accept a number of output variables. There is a variation of the multi-output approach that fuses Direct's concepts with them. In various subsets of the forecasting horizon, this variant used the Direct technique.

1.4.3 Models

In this exercise, we ran a total of 30 models, including 12 classical models, 17 machine learning models, and one base model. The classical models include seasonal naive, polytrend, grand means, Auto ARIMA, SARIMAX, exponential smoothing, croston, ETS, theta forecasts, BATS, and TBATS models from the sktime python library. Linear models such as linear, ridge, LASSO, elastic net, least angular (LAR), Bayesian ridge, LASSO LAR, huber, passive aggressive regression, KNN regressor, and orthogonal matching pursuit are among the 17 machine learning models. In addition to tree-based machine learning models including random forest, decision tree, and extra tree, boosting models including gradient boosting, Adaboost, and light gradient boosting. All ML models have been extracted from the scikit-learn Python module. This experiment also utilized the Python packages pandas, NumPy, and matplotlib. In the appendix, more information about the models and their origins are provided.

1.4.4 Model evaluation metrics

I evaluate the model's effectiveness using 5 different methods, including (i) RMSLE (Root Mean Squared Logarithmic Error), (ii) R2, (iii) RMSE (Root Mean Squared Error), (iv) R2 (Mean Absolute Percentage error), and (v) sMAPE (symmetric Mean Absolute Percentage Error). The appendix contains the formula for each of the aforementioned

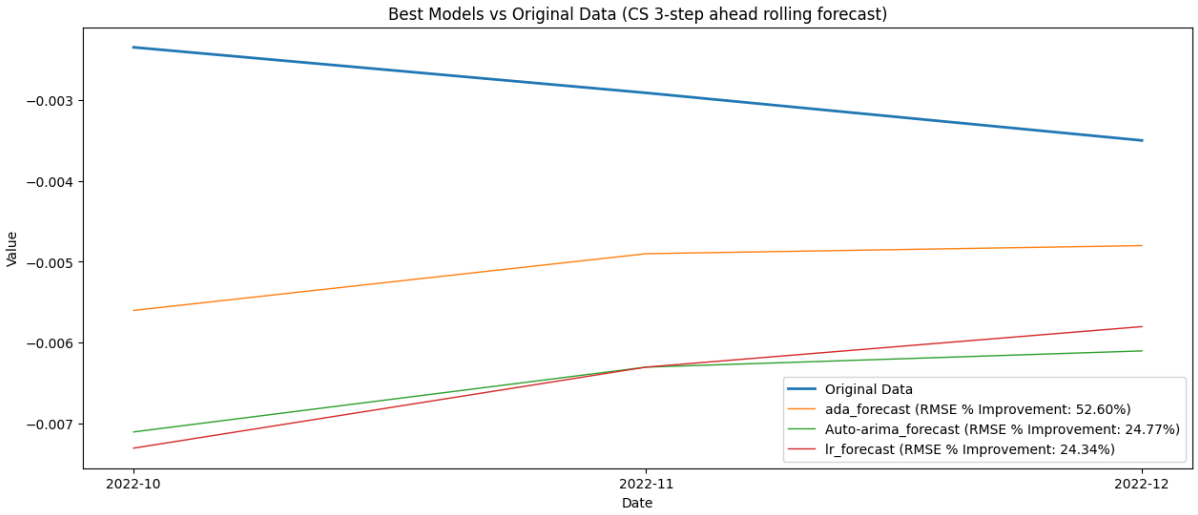


Figure 1.2: 3-step ahead Actual vs predicted [Case-shiller]

parameters. We run every model for each of our four time series indexes, and in each case, we select the best five models to be described in the results section. Further, we also extend the Case-Shiller Index, one of the housing indexes with the longest time series we have in our example, to compare the performances of these models in case they are unable to capture the long business cycle of the housing market due to the smaller time series data we used in our comparison case above. In addition, we utilize a 5-fold CV rather than a 3-fold CV for our larger time series index since we have more datapoints (473 as opposed to 119) to examine the gain in model performance brought on by the use of a k-fold CV rather than a straightforward train-test split.

1.5 Results

The findings of the best-performing models differ from those of the 3-step ahead, 6-step ahead, and 12-step ahead forecasts due to the fact that, depending on the horizon, the data for various steps forecasts has been highly volatile. Since a 5-fold CV strategy was used to train the model, the comparison of multi-ahead-of-time forecast performance yields distinct results for each forecast horizon and data set. Five times each, the ML and classical models outperformed our base model in comparison to the 12 results, whereas the naïve model outperformed our base model on two occasions. In general, among ML

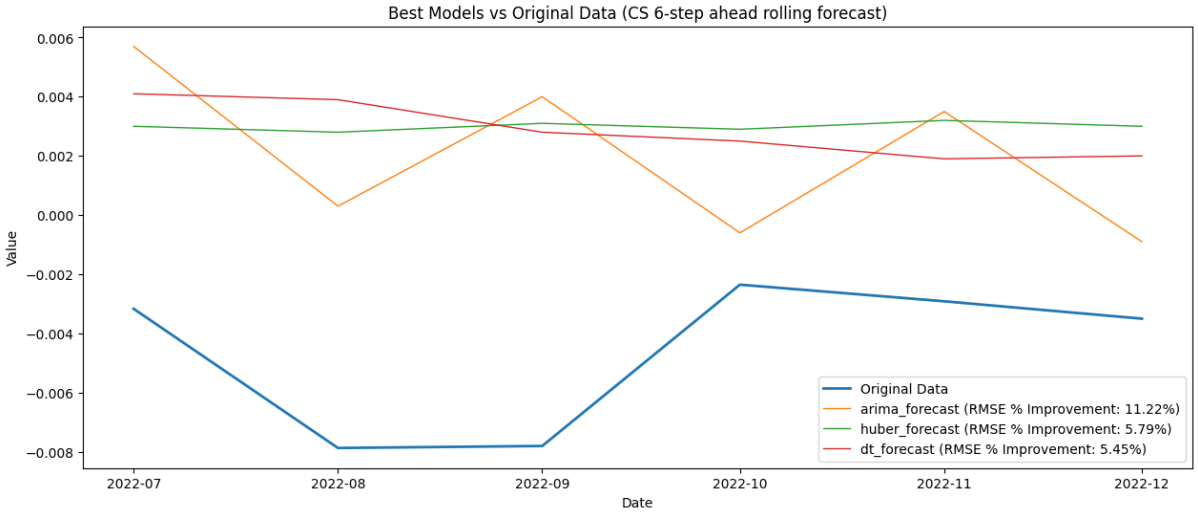


Figure 1.3: 6-step ahead Actual vs predicted [Case-shiller]

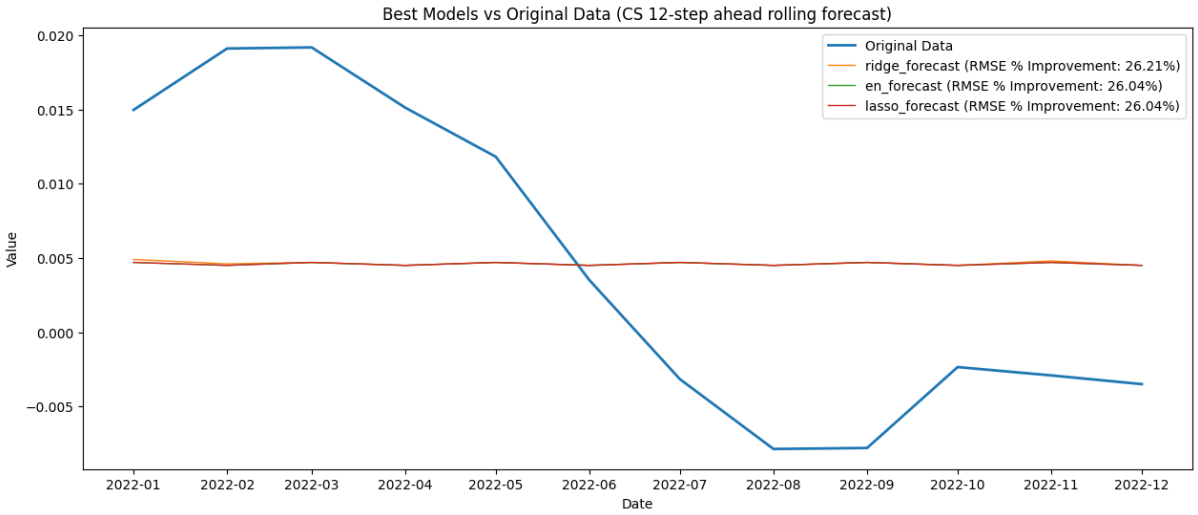


Figure 1.4: 12-step ahead Actual vs predicted [Case-shiller]

Case-Shiller 3-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
ada_forecast	52.60	40.34	53.38
Auto-arima_forecast	24.77	16.93	24.85
lr_forecast	24.34	17.90	24.87
lars_forecast	24.34	17.90	24.87
Bridge_forecast	24.34	17.90	24.87
gbr_forecast	19.39	30.79	28.65
omp_forecast	19.30	13.28	19.35
tbats_forecast	17.95	12.40	17.98
bats_forecast	17.32	11.94	17.33
huber_forecast	11.68	7.91	11.54
ETS_forecast	2.52	1.56	2.40
expsmooth_forecast	1.34	0.73	1.36
theta_forecast	0	0	0
snaive_forecast	-1.35	-0.72	-1.36
knn_forecast	-8.00	-4.26	-7.25
lightgbr_forecast	-13.37	-6.97	-12.98
rf_forecast	-23.61	-11.64	-23.25
et_forecast	-26.22	-12.87	-25.70
gm_forecast	-33.45	-118.48	-30.13
ridge_forecast	-54.68	-118.48	-50.55
arima_forecast	-56.17	-23.79	-54.10
en_forecast	-58.63	-118.48	-54.85
lasso_forecast	-58.63	-118.48	-54.85
llars_forecast	-58.63	-118.48	-54.85
par_forecast	-58.63	-118.48	-54.85
poly_forecast	-59.92	-118.48	-56.37
dt_forecast	-74.89	-29.84	-68.56
croston_forecast	-182.13	-118.48	-180.93

models, boosting models outperformed tree-based and linear ML models. In contrast, the victors among classical models were the Arima and exponential smoothing models. When comparing the improvement in rmse forecast, a general trend is observed. In comparison to 6- and 12-step forecasts, the improvement in model performance for 3-step forecasts can reach up to 90%. Since we are employing a recursive rolling multi-step strategy, the forecast improvement degrades due to the propagation of errors caused by this strategy. The HPA dataset demonstrates the greatest increase in forecast accuracy, followed by the Zillow index, when compared to other datasets. We can confidently

assert that these privately sourced indexes should be used with ML models as opposed to a naive model, even under volatile conditions, because it improves the models in the vast majority of instances. In contrast to other indexes, publicly available housing indexes, such as Case-Shiller and FHFA, do not demonstrate a significant advance over the base model. We can contend that the noise in data from the two sources is distinct, and that privately sourced data contains more accurate information, thereby assisting our machine learning models in producing better results for these indexes.

When comparing the performance of various multi-step ahead forecasts, machine learning models perform better for 3-step forecast horizons. Classical models are able to outperform machine learning (ML) models as we progress to a 6-step ahead forecast. Lastly, when comparing forecast models for a 12-step advance forecast, ML models once again outperform classical models. It is also surprising to see that among the top 5 models, ML models are always present, despite not being the most popular models. Also, when we examine the datasets for which ML models perform better, we find that for lengthier time series indexes such as CS and FHFA, ML models perform better than Zillow or HPA. The results demonstrate that since ML models require more data to perform better, it is therefore able to detect more patterns in lengthier datasets.

Finally, we discuss the speed of these models, despite the small size of the dataset in our analysis, which ranges from 120 to 430 datapoints. Except for three models, nearly all ML and classical models were able to provide results in a few seconds. The model with the lowest performance was TBATS, followed by BATS and Auto-Arima. Compared to BATS and TBATS models, Auto-Arima models required an average of 80 to 100 percent more time, while BATS and TBATS models required 500 to one thousand percent more time. Given that none of the three models mentioned were able to place in the top three models in our twelve results, it is best to avoid these classical models as they are neither accurate nor fast.

1.6 Conclusion

I've drawn three findings or consequences from the studies mentioned above. Firstly, if forecasting is done during a volatile period, using k-fold cross-validation not only helped decrease errors but also provided a better forecast performance across ML models. So, using the CV approach rather than the basic train-test split as an arbitrary point is preferable when forecasting in a dynamic context. Secondly, The results also make it clear why hedonic price-based indexes in the housing market are superior to repeated sales methods or hybrid methods for modeling housing forecasts. And lastly, I came to the conclusion that it is preferable to use ML models for privately sourced housing index compared to naive or classical models when forecasting in an unstable environment.

It is possible to conduct more research to confirm the accuracy of my conclusions. To cross-check the results, you may expand the experiment to use the various multi-fold strategies discussed above. Deep learning models like RNN and LSTM can provide deeper insights into this since the outcomes of ML models have not been favorable after the data has been stationarized and scaled to standard scaling. Using wavelet transform and fast Fourier transform algorithms can aid in capturing the missing misses in the model if we are forecasting during a volatile period. Finally, we can enhance the performance of financial time series data in a turbulent period by utilizing semantic features and transfer learning.

In comparison to running ML models on time series data that is experiencing volatility for the first time in the market, ML models perform better when we have long time series data that includes multiple volatility periods from the past. We saw this in the results for the full index forecast comparison of the Case-Shiller index. The forecast improvement for our data set was further improved by expanding the validation set in k-fold cross-validation, further demonstrating why a k-fold CV is preferable to a straightforward train-test split for supervised learning models. This comparison revealed, among other things, that when forecasting for shorter time horizons (3,6 months ahead) as opposed to longer time horizons, machine learning models beats conventional statistical models compared to longer time horizon (12-months ahead)

forecasts where even seasonal naive model beats ML models.

Case-Shiller 6-step ahead rolling forecast results (% improvement)

Model	**RMSE**	**sMAPE**	**MAPE**
arima_forecast	11.22	13.59	22.65
huber_forecast	5.79	7.43	6.83
dt_forecast	5.44	6.88	10.71
Auto-arima_forecast	3.82	3.12	5.06
lr_forecast	1.19	1.15	0.75
Bridge_forecast	1.19	1.15	0.75
expsmooth_forecast	1.02	0.12	1.54
ets_forecast	1.02	0.13	1.57
theta_forecast	-1.14	0	-1.39
gm_forecast	-2.28	-3.12	-2.78
lars_forecast	-3.33	-3.78	-4.28
omp_forecast	-3.33	-3.78	-4.28
rf_forecast	-3.65	-4.44	-3.80
tbats_forecast	-4.18	-8.88	-7.32
ada_forecast	-6.94	-12.12	-9.37
bats_forecast	-7.06	-14.34	-10.06
lightgbm_forecast	-7.63	-14.34	-10.30
en_forecast	-19.13	-24.50	-23.11
ridge_forecast	-19.13	-24.50	-23.11
lasso_forecast	-19.13	-24.50	-23.11
llars_forecast	-19.13	-24.50	-23.11
par_forecast	-19.28	-24.50	-23.48
polytrend_forecast	-19.52	-24.50	-23.63
et_forecast	-20.72	-32.10	-24.94
gbr_forecast	-26.09	-28.78	-22.62
snaive_forecast	-56.06	-95.67	-55.88
knn_forecast	-74.26	-112.54	-97.84
croston_forecast	-108.34	-240.56	-129.29

Case-Shiller 12-step ahead rolling forecast results (% improvement)

Model	**RMSE**	**sMAPE**	**MAPE**
ridge_forecast	26.20	24.68	46.71
en_forecast	26.04	24.99	46.75
lasso_forecast	26.04	24.99	46.75
llar_forecast	26.04	24.99	46.75
par_forecast	25.87	24.99	46.52
polytrend_forecast	25.78	25.27	46.26
gm_forecast	25.50	29.63	51.99
Bridge_forecast	16.23	0.15	18.33
omp_forecast	16.14	0.15	18.51
lr_forecast	16.08	0.15	18.14
lar_forecast	14.35	0.15	16.46
bats_forecast	10.57	0.52	12.76
dt_forecast	7.85	1.72	8.63
huber_forecast	7.32	0.26	8.13
croston_forecast	6.56	1.69	7.58
tbats_forecast	6.53	0.85	7.40
knn_forecast	6.23	0.56	5.92
snaive_forecast	5.53	1.12	8.03
arima_forecast	4.48	0.01	6.66
ada_forecast	1.69	0.46	1.89
expsmooth_forecast	1.67	0.20	1.90
rf_forecast	0.60	0.94	0.23
Auto-arima_forecast	0.41	0.36	0.77
theta_forecast	0	0	0
et_forecast	-0.20	2.26	0.80
gbr_forecast	-4.17	0.12	-4.37
ets_forecast	-4.27	0.09	-4.34
lightgbm_forecast	-7.28	0.24	-8.18

Chapter 2

Application of Machine Learning and Deep Learning in Forecasting REIT's

2.1 Introduction

A real estate investment trust (REIT) is an entity that has, manages, or provides financial backing for real estate properties that generate revenue, encompassing a range of property industries. The establishment of U.S. Real Estate Investment Trusts (REITs) can be traced back to 1960, when the United States Congress introduced this financial vehicle with the aim of providing small-scale investors with opportunities to participate in income-generating real estate assets. Since its inception, this asset class has had significant growth, expanding its presence to encompass 40 countries, including all G-7 nations and two-thirds of OECD countries. As a result, it has become an important component of investment portfolios in these regions. Once initiated, investors have the opportunity to participate in many categories of Real Estate Investment Trusts (REITs), including but not limited to Community Shopping Centers, Railroad Real Estate, Lodging, and Apartments. The emergence of the current period of Real Estate Investment Trusts (REITs) may be attributed to the implementation of the Tax Reform Act in 1986. This legislative development paved the way for the blooming of REITs in subsequent years, providing investors with opportunities to participate in various real estate properties, including but not limited to self-storage facilities, automobile dealerships, data centers, and medical offices. The availability of Real Estate Investment Trusts (REITs) has provided investors with an alternative avenue for investment, enabling them to generate returns from various assets such as post offices, refrigerated storages, and even casinos, without the need for direct ownership.

According to the National Association of Real Estate Investment Trusts (NAREIT), the market capitalization of Real Estate Investment Trusts (REITs) has exhibited substantial growth throughout time. Specifically, it increased from $1.4 billion in 1961 to $124.2 billion in 1999, and further surged to $1.27 trillion in 2022. The number of Real Estate Investment Trusts (REITs) has experienced significant growth throughout the years, increasing from 34 in 1971 to 206 in 2022. The overall market value of Real Estate Investment Trusts (REITs) in the United States is estimated to be approximately $4 trillion. Out of this amount, approximately $2.5 trillion is contributed

by public REITs, which possess a portfolio of approximately 575,000 properties and 15 million acres of timberland. Real Estate Investment Trusts (REITs) are obligated to distribute a minimum of 90% of their investment earnings to shareholders in the form of dividends. With the exception of mortgage real estate investment trusts (REITs), which provide financing for real estate investments and generate income from interest payments, who are the financial entities that fund these real estate ventures? Considering the magnitude and potential of alternative investment opportunities in segmented asset classes, it becomes essential to thoroughly explore this tradable investment inside the stock market.

Time series forecasting is of great importance in a wide range of fields, including banking and healthcare, due to its ability to provide accurate predictions of future values. In recent times, scholars have progressively acknowledged the significance of utilizing advanced machine learning (ML) and deep learning (DL) models for the purpose of time series forecasting. The utilization of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models has demonstrated significant value in the prediction of intricate patterns by effectively capturing long-term dependencies within sequential data. Recurrent Neural Networks (RNNs) possess recurrent connections that enable them to effectively capture and simulate sequential dependencies. This inherent characteristic renders RNNs well-suited for problems involving time series prediction. In contrast, ensemble approaches such as XGBoost and Random Forest leverage the collective strength of several weak learners, hence providing enhanced accuracy and resilience. Multiple studies have emphasized the superiority of these models compared to conventional methods, showcasing their ability to effectively handle complex temporal patterns and surpass standard time series forecasting methodologies [1][2][3].

This study focuses on the analysis of time series forecasting using a univariate dataset. Various models, such as LSTM, GRU, RNN, XGBoost, and Random Forest, are employed for this purpose. The next paragraphs provide a thorough examination of the comparative outcomes, emphasizing the strengths and weaknesses of each approach. Additionally, we analyze the percentage of improvement attained by machine learning

(ML) and deep learning (DL) models in comparison to a benchmark model known as the Random Walk. Interestingly, our research reveals that the LightGBM model demonstrates superior performance compared to other machine learning models, although deep learning approaches exhibit comparatively worse performance. The aforementioned insight serves as a catalyst for conducting additional research on the effects of hyperparameter tuning. This is due to the fact that our current evaluation methodology only considers basic implementations of each model for a rolling 200-step ahead forecast.

In the following sections, we will provide a comprehensive analysis of our dataset, examine pertinent literature on time series forecasting, clarify our approach, and elaborate on the findings and conclusions we have derived. The last portion provides a concise overview of the forthcoming content, presenting a comprehensive guide for readers to traverse the complexities of our study. The appendix contains supplementary information, while the bibliography presents a wide array of sources that have contributed to our research.

2.2 Data

2.2.1 Data Description

This study utilized a dataset consisting of 11 REITs, sourced from Yahoo Finance. The selected REITs represent a single category among the 11 available in the United States. Real Estate Investment Trusts (REITs) are classified into various industries, including but not limited to Office, Industrial, Data, Retail, Lodging, Residential, Health, Infrastructure, Diversified, Specialty, and Mortgage. The table presented below displays the results of the exploratory data analysis conducted on the differenced series. Additionally, graphs depicting the datasets spanning from November 2015 to October 2023 have been included. To facilitate computation and enable performance comparison between models and individual REITs datasets, we have opted to utilize data from the beginning of November 2015 to October 2023. The 11 REIT's dataset used for this

exercise is mentioned below.

Exploratory Data Analysis

REITs category	*Ticker*	*Horizon*	*Min*	*Max*	*Mean*	*Std. dev.*
Office	VNO	2015-10-30 to 2023-10-13	12.38	89.62	55.21	21.35
Industrial	EGP		50.11	228.56	121.22	42.55
Data	EQIX		265.05	882.83	558.54	166.12
Retail	GTY		16.23	36.43	28.09	4.31
Lodging	IHT		0.77	11.55	2.15	1.06
Residential	BRT		5.41	25.35	14.02	5.12
Health	NHI		33.56	90.79	67.59	10.39
Infrastructure	AMT		83.66	303.62	188.01	57.78
Diversified	ALX		156.91	447.34	320.09	77.09
Specialty	PCH		23.42	63.44	44.53	7.71
Mortgage	BXMT		13.02	40.51	29.44	4.85

Table 2.1: Data Description

Short description of REIT's data

ALX (Alexander's Inc.): ALX is a real estate investment trust that owns and manages a diverse portfolio of properties, including office and retail spaces.

AMT (American Tower Corporation): AMT is a global REIT focused on owning and operating wireless and broadcast communications infrastructure, including cell towers.

BRT (BRT Apartments Corp): BRT is a REIT primarily engaged in the ownership, operation, and development of multi-family properties.

BXMT (Blackstone Mortgage Trust Inc.): BXMT is a REIT specializing in real estate debt investment, primarily in senior loans collateralized by commercial real estate.

EGP (East Group Properties Inc.): EGP is a REIT that focuses on the development, acquisition, and management of industrial properties in the United States.

EQIX (Equinix Inc.): EQIX is a global REIT known for its data center and interconnection services, providing infrastructure solutions for businesses.

GTY (Getty Realty Corp): GTY is a REIT specializing in the ownership, leasing, and financing of convenience store and gasoline station properties.

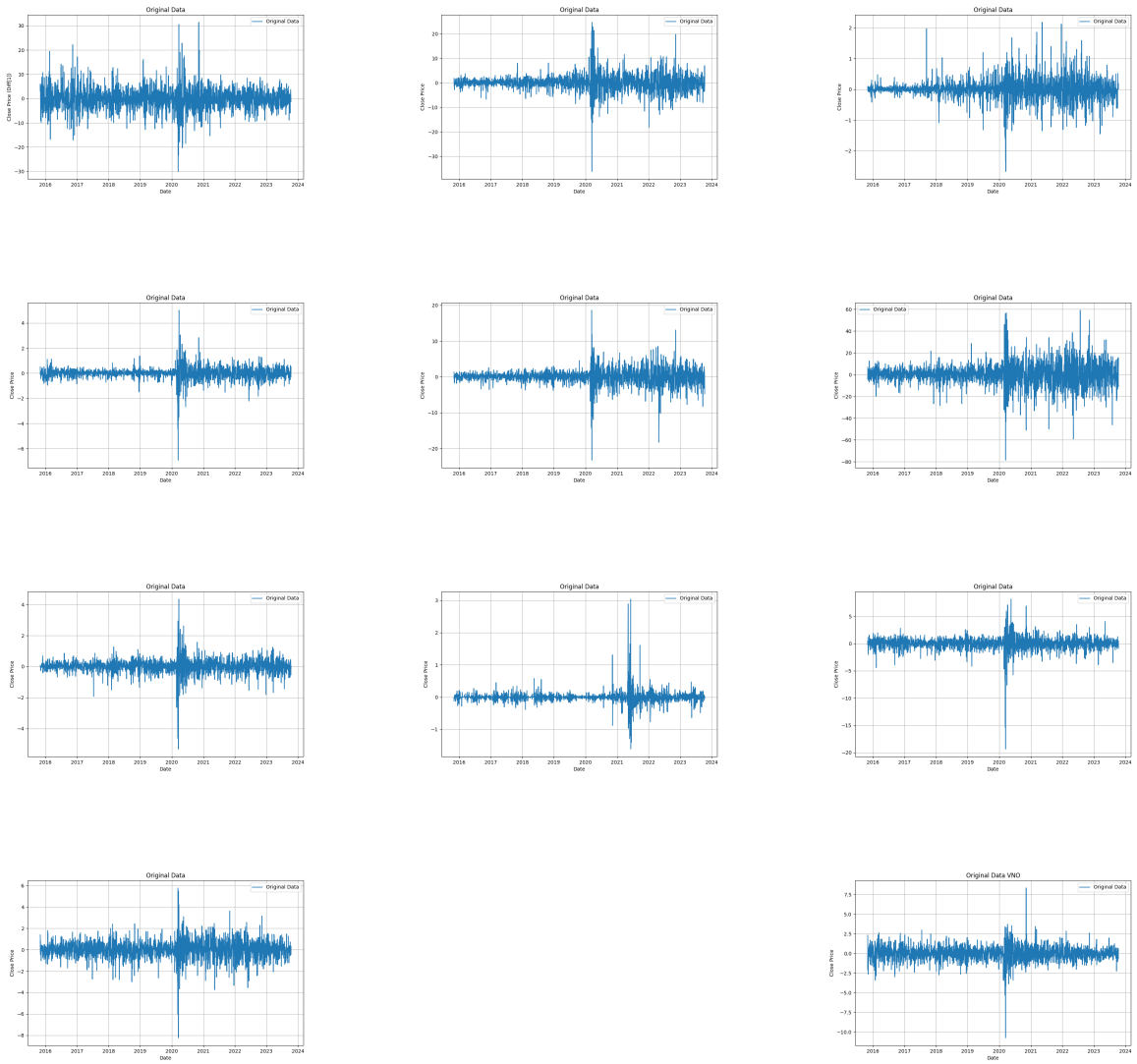
IHT (InnSuites Hospitality Trust): IHT is a REIT engaged in the ownership and operation of hotel properties, primarily under the InnSuites Hotels brand.

NHI (National Health Investors Inc.): NHI is a REIT that invests in healthcare properties, including senior housing, skilled nursing facilities, and medical office buildings.

PCH (PotlatchDeltic Corporation): PCH is a timberland REIT involved in the ownership and management of forestland and the sale of wood products.

VNO (Vornado Realty Trust): VNO is a REIT with a diverse portfolio that includes office, retail, and residential properties, primarily concentrated in New York City and Washington, D.C.

Figure 2.1: REIT's (Diff=1) Graphs



Unit Root Test

The ADF unit root test was conducted on the differenced data and results are presented in table below. All the data series has a *p-value* almost zero indicating all the series to be stationary. Further *in-sample* AR(1) is conducted to these datasets in below section.

Unit Root Test

		Unit Root Test (ADF)	
REIT's ticker	*Horizon*	*p-value*	*Lags*
ALX	2015-10-30 to 2023-10-13	0.00	0
AMT		0.00	19
BRT		0.00	0
BXMT		0.00	21
EGP		0.00	26
EQIX		0.00	26
GTY		0.00	8
IHT		0.00	26
NHI		0.00	18
PCH		0.01	6
VNO		0.01	11

Table 2.2: Unit Root Test Results

In-Sample AR(1) Regression

Examining the AR(1) findings of the REIT's data inside the sample reveals that the R-squared values are very low, indicating that conventional statistical models have challenges in forecasting these financial series. The preliminary analysis of the dataset supports the utilization of Machine Learning (ML) and Deep Learning (DL) models over traditional methods, since they have demonstrated higher performance in recent times.

Correlation Plot

Despite the fact that Deep Learning models, such as LSTM, RNN, or GRU, are adept at handling non-linear data and effectively capturing non-linear trends, the utilization of solely stationarized datasets persists. The term "stationary" refers to the property of a model's statistics remaining constant over time, specifically in a "locally" stationary manner. ARIMA models are a type of regression model that utilizes historical data

In-sample AR(1) Results

		AR(1) results	
REIT's ticker	*Horizon*	*Coef*	*R-square*
ALX	2015-10-30 to 2023-10-13	-0.071	0.005
AMT		-0.070	0.005
BRT		0.037	0.001
BXMT		-0.103	0.010
EGP		-0.057	0.003
EQIX		-0.109	0.011
GTY		-0.132	0.017
IHT		0.046	0.002
NHI		-0.094	0.009
PCH		0.002	0.00
VNO		0.002	0.00

Table 2.3: *in-sample* AR(1) Results

points as input for linear regression analysis to forecast the subsequent data point. The augmented reality component serves this purpose, at the very least. The acquisition of the model involves the acquisition of the regression coefficients. When dealing with a time series, the process of learning the correlation between the preceding N data points and the subsequent point, and subsequently utilizing this knowledge to forecast the next value based on a distinct set of N data points, entails an implicit assumption that the identical correlation persists between the N predictor points and the ensuing N+1st point that is being predicted. This pertains to the concept of stationarity.

The process involves acquiring a pattern from a specific segment of a time series, which is subsequently utilized to generate predictions for another segment of the same time series. The learning process is still ongoing. The model acquires a reduced portrayal of the temporal sequence. In the event that this portrayal is valid for the training set but not for the test set, the model's performance will be suboptimal. In contrast to ARIMA, Recurrent Neural Networks (RNNs) possess the ability to acquire nonlinearities, with LSTM nodes being particularly adept at this task. LSTM and GRU models exhibit exceptional proficiency in acquiring knowledge of long-term dependencies. For instance, this blog entry may serve as an illustration. The utilization of RNNs results in a less fragile interpretation of 'stationarity', thereby reducing its level of significance. In order to acquire the capability of learning long-term dependencies, a substantial amount of

data is required for training purposes. The correlation heatmap of all the 11 REIT's data with its last five lags are shown below.

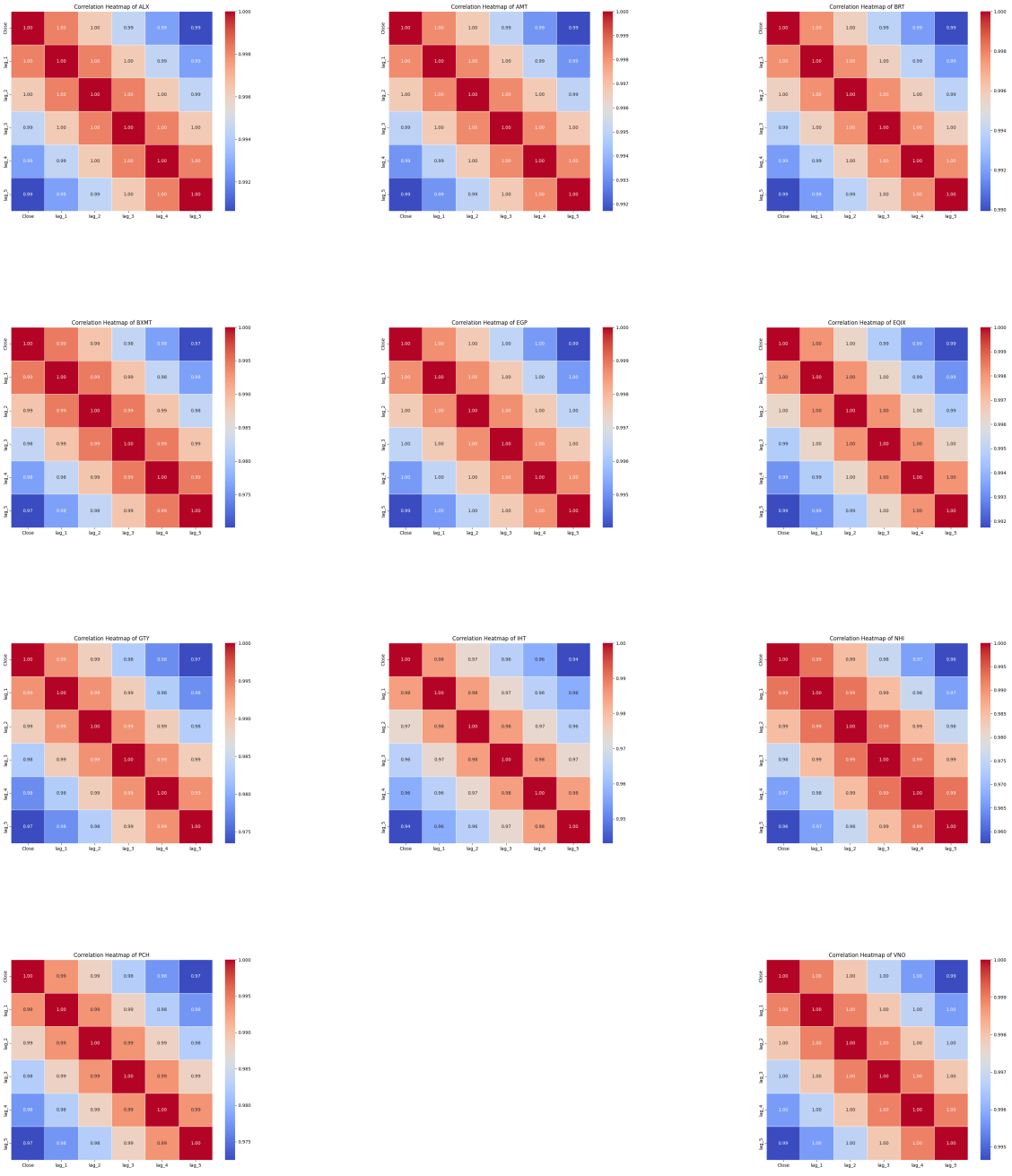


Figure 2.2: Correlation graph of all REIT's with their respective 5 lags

2.3 Literature Review

The term "Efficient Markets Hypothesis" was introduced by Roberts (1967). He also established the categorization of weak and strong form tests, which has since become

a well-known taxonomy in Fama (1970). The Efficient Market Hypothesis (EMH), also referred to as the Efficient Market Theory, posits that stock prices incorporate all available information and thus generating consistent alpha is unattainable. The Efficient Market Hypothesis (EMH) posits that stocks are consistently traded at their intrinsic worth on exchanges, hence precluding investors from acquiring undervalued stocks or selling equities at inflated prices. Hence, it is seen implausible to surpass the performance of the broader market by means of adept stock picking or market timing, leaving the acquisition of riskier investments as the sole avenue for investors to attain elevated returns.

Numerous attempts have been undertaken subsequent to the introduction of the Efficient Market Hypothesis (EMH) to ascertain if the stock market adheres to a random walk pattern or may be forecasted utilizing statistical methodologies and available data. LeRoy and Porter (1981) conducted a study employing a variance bound test, which provided evidence for the existence of a random walk in stock price data. Consequently, this finding implies that stock prices cannot be accurately projected. According to Kendal (1953), it is suggested that there is a lack of regularity or periodicity that can be quantified from this data, making it difficult to forecast such series. Fama (1995) confirmed the random walk hypothesis and demonstrated that forecasted series exhibit behavior consistent with that of a random walk. However, there has been a significant amount of new research that challenges these findings and demonstrates greater predicting ability as a result of the utilization of more advanced models, particularly those based on machine learning and deep learning techniques.

The majority of scholarly investigations pertaining to the modeling of Real Estate Investment Trusts (REITs) have focused on either the prediction of volatility or the forecasting of returns. The present study conducted by Lee and Pai (2010) aims to examine the impact of return distribution specification for Real Estate Investment Trusts (REITs) on the efficacy of volatility forecasting, utilizing three distinct GARCH models, namely GARCH-N, GARCH-ST, and GARCH-SGED. The present research endeavors to examine the impact of return distribution specification for Real Estate Investment Trust (REIT) on the efficacy of volatility forecasting through the application

of three Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models. The empirical findings suggest that the GARCH-SGED model outperforms other models in predicting the volatility of REITs in the United States. Meanwhile, Zhou and Kang (2011) conducted a comparative analysis of various established models to forecast volatility in REITs. The study exhibits that Long-memory models, namely ARFIMA and FIGARCH, exhibit superior forecast accuracy compared to other models. Conversely, asymmetric models, such as EGARCH and FIEGARCH, demonstrate inferior performance in comparison to the aforementioned models.

The study conducted by Wang et al. (2011) aimed to utilize a structural time series model in predicting the returns of Real Estate Investment Trusts (REITs) in the United States. The researchers employed monthly data spanning from January 1994 to November 2009 for their analysis. The findings of the empirical analysis suggest that the relationship between cycle and the returns of industrial REIT and manufactured home REIT can be more readily explicated due to the comparatively lower and more consistent returns exhibited by these types of REITs in comparison to other REITs. Pavlova et al. (2014) have suggested a revised version of the fractionally integrated (FIGARCH) model to predict outcomes at daily and weekly intervals. In their study, Chen et al. (2014) utilized the Grey Relational Analysis (GRA) and Artificial Neural Network (ANN) methodologies to evaluate the influence of crucial factors on the predictive accuracy of real estate investment trust (REIT) returns.

The study conducted by Akinsomi et al. (2016) investigates the predictability of real estate returns through the utilization of US real estate investment trusts (REITs) and a range of potential predictors spanning from January 1991 to December 2014. In addition to the conventional variables analyzed in pertinent literature, we incorporate a range of sentiment and uncertainty measures that could serve as noteworthy determinants of REITs' performance. The findings derived from empirical analysis suggest that the reliable indicators of Real Estate Investment Trusts (REITs) returns exhibit temporal and forecast horizon-dependent fluctuations. The findings indicate that REITs returns can be effectively predicted by a range of factors, including

economy-wide indicators, monetary policy instruments, and sentiment indicators. From an economic perspective, a prognostication-driven investment approach exhibits superior performance compared to a buy-and-hold strategy.

The study conducted by Loo et al. (2016) examines the volatility patterns exhibited by the real estate investment trust (REIT) markets in Asia. The utilization of intraday data for volatility forecasting in the real estate literature is first attempted by Zhou (2017). They employ these methodologies in analyzing various prominent REIT markets worldwide. The study conducted by Loo (2020) aims to ascertain the comparative efficacy of artificial neural network (ANN) and linear regression models in forecasting the excess return of real estate investment trusts (REITs) in Hong Kong. Bonato et al. (2022) utilized a global dataset consisting of intraday data at 5-minute intervals across nine prominent markets and regions. The authors employed this dataset to create metrics for realized volatility, realized jumps, realized skewness, and realized kurtosis of returns for international Real Estate Investment Trusts (REITs) over the period spanning from September 2008 to August 2020.

The study conducted by Bianchi and Guidolin (2014) aims to investigate the feasibility and methodology of expanding basic linear predictability models of the vector autoregressive (VAR) category to encompass the characteristic bull and bear trends observed in various asset classes, such as REITs. Assaf (2015) conducted an empirical study examining the presence of long memory in the returns and volatility of real estate investment trusts (REITs) markets across several countries, including the United States, United Kingdom, Hong Kong, Australia, and Japan. The study conducted by Ismail et. al. (2016) utilized the daily returns of stock market indices from four African countries. The data was collected for a period spanning from January 2, 2000, to December 31, 2014.

The study conducted by Leippold et al. (2022) examines an extensive range of return prediction factors through the application of diverse machine learning algorithms. The study conducted by Yuan et al. (2020) involved the utilization of diverse feature selection algorithms to select features, and the establishment of machine

learning-based models for predicting stock price trends. The parameters of these models were determined through time-sliding window cross-validation, utilizing an 8-year dataset obtained from the Chinese A-share market.

2.4 Methodology

The methodology section utilizes a variety of Machine and Deep learning methods to assess their efficacy in contrast to the Random Walk model. In the present study, a 1-step rolling forecast was performed utilizing the latest year of data, encompassing the time period from December 27th, 2022, to October 13, 2023. The selected time period was utilized as our test set and represents out-of-sample data. The training dataset covers the time period from October 30, 2015, to December 26, 2022. The ADFuller approach was employed to attain stationarity of the data, with the exception of two data sets, IHT() and GTY(), which were already stationary for the specified time period. In order to do a comparative forecast comparison, we also utilize 1-level differenced data for the IHT and GTY datasets as well. It is noteworthy that the ML and DL models employed in the study exhibit robust performance and are capable of effectively operating with non-linear datasets, hence obviating the necessity for a stable dataset. The exploratory data table presents the p-value for the 1-level differenced dataset of each Real Estate Investment Trust (REIT). The dataset for the entire experiment consists of 11 sets of daily price indices for REITs, each representing a different field. Each set contains 2000 data points, with 1800 data points used for training models and 200 data points used for evaluating the forecast performance of these models using a 1-step rolling forecast approach.

The genesis of deep learning models, alternatively referred to as neural network models, may be traced back to the human brain. Cognitive scientists and neuroscientists have displayed early interest in and undertaken research on neural network models Posner (1989), which have since undergone further development to reach their present state. The persistent efforts to understand the functioning of the brain have led to the emergence of neural network models, which have been subject to

much debate on their potential inferiority compared to the human brain. Computer processing units, like to the neurons found in the human brain, are capable of receiving diverse external inputs through their dendrites. Similarly, computer memory operates in a manner reminiscent to synapses, with a parallel connection count of around 10-4. The sole distinction lies in the fact that computer memory operates in a passive manner, whereas processor units remain in a permanent state of activity. The human brain consists of synapses, responsible for memory, and neurons, which function as processors. These components are dispersed across the brain's network and operate concurrently. Although there is a lack of comprehensive knowledge on the processing capacity of the neurons that constitute the human brain, these neural processors are generally considered to be rather rudimentary and operate at a slower pace when compared to computers.

To get comprehension of the neural network model, it is necessary to commence with the first and most rudimentary kind of neural network model known as the perceptron. A perceptron may be seen as a fundamental processing unit that is capable of receiving inputs from the environment or outputs from other perceptron. Each input $x_j \in \Re, j = 1, ..., d$, is accompanied by a connection weight, or synaptic weight $w_j \in \Re$, . In the simplest example, the output y is obtained by taking the weighted sum of the inputs, and w_0 represents the intercept value that enhances the generality of the model.

$$y = \sum_{j=1}^{d} w_j x_j + w_0 \tag{2.1}$$

It is intriguing to note that when the value of $d = 1$ and x is provided as input from the environment through an input unit, it transforms into an equation of a line and exhibits characteristics of a linear fit model. Furthermore, when there are many inputs, the line evolves into a hyperplane, enabling the perceptron to function as a multivariate linear fit model. In addition to the establishment of a hyperplane within each perceptron, a further procedure occurs whereby the input space is partitioned into two regions utilizing a linear discriminant function or activation function. $z = Act(y)$.

$$y = Act(\sum_{j=1}^{d} w_i x_i) \tag{2.2}$$

A basic neural network design may be imagines as having three layers namely input, output and hidden layers constituted of this perceptron. Inside each perceptron a weighted y feeded is converted into z using the activation function provided and then additional weight is added before it flows to output layer. This flow of data forward via input to hidden to output layer with weights being added throughout each layer is termed forward propagation and these weights are rectified and updated using back propagation to minimize the losses according to below rule.

$$w_{new} = w_{old} - \eta(\partial L/\partial w_{old}) \tag{2.3}$$

Where η is defined as learning rate and L is the loss function.

To conduct a comparative analysis of the forecasting efficacy of machine learning and deep learning models, I have selectively chosen models that are widely recognized in the field of univariate time series forecasting, specifically for the stock prices dataset. The foundational model utilized is the Random Walk model, wherein the current price is predicted to be equivalent to the previous day's price. In the field of machine learning, I have selected four models, namely Random Forest, XGBoost, LightGBM, and Elastic Net. These models represent one linear, one tree-based, and two boosting methods. In the realm of Deep Learning model analysis, three widely employed models for time series forecasting are the Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and Gated Recurrent Unit (GRU).

In the context of model evaluation, a comparison is made between models utilizing random walk methodology. The average root mean square error (RMSE) improvement is then analyzed over a one-year period of rolling forecast performances. It is worth noting that deep learning models have demonstrated their utility in enhancing long-term forecasting. Therefore, employing a rolling forecast approach, as opposed to a point forecast approach, may provide valuable insights into the performance of these models.

The findings are analyzed and elaborated upon in the subsequent section, accompanied by a supplementary table and graphs included in the appendix.

Once the value of yhat has been computed, the loss function is formulated, and an optimizer is employed to minimize the loss function by optimizing the weights through the process of backpropagation, as determined by the aforementioned rule. The gradient descent algorithm is frequently employed as an optimization technique. The speed at which a model reaches a global minimum might vary depending on the selection of the learning rate and loss function. Consequently, hyperparameter optimization is conducted to determine the optimal values for these parameters during the construction of neural network models. The process of forward and backward propagation continues to iterate until the global minimum of the loss function is obtained. The Kloss function is minimized by the process of backpropagation, which involves updating the weights in each hidden layer. This allows for flexibility in achieving the desired outcome by incorporating several hidden layers.

The vanishing gradient problem has been a persistent issue in neural network models for an extended period, particularly when sigmoid functions were employed as activation functions. However, this problem was effectively addressed with the introduction of the Rectified Linear Unit (ReLU) as an alternative activation function. The vanishing gradient problem arises when the number of layers grows, resulting in diminishing derivatives. To address this issue, the Rectified Linear Unit (ReLU) was introduced and shown to be beneficial.

The forward and backward propagation techniques facilitate the flow of information across the layers in neural network models. A further concern that may arise is the occurrence of the gradient explosion problem. As a consequence of the chain rule, it is possible for the optimization process to diverge and fail to converge to the global minimum, rather than converging and reaching the minimum. This phenomenon may occur in models such as artificial neural network (ANN) models, where an excessive number of weights (w_j) and biases can lead to overfitting of the data. It is acknowledged that when utilizing a single layer, the primary concern is underfitting.

However, when employing two or more layers, the issue of overfitting emerges. This problem can be mitigated by the implementation of regularization techniques or feature selection methods, which restrict the inclusion of specific input features.

The final issue that must be taken into account is the strategic selection of the weight initialization approach. One should adhere to the fundamental principles of selecting weights that are relatively smaller, same, and possess a favorable variance. The uniform distribution is frequently employed for weight initialization in several machine learning models. However, other activation functions such as the Gorat distribution or He initialization are utilized in conjunction with the sigmoid and ReLU activation functions, respectively. When it comes to the selection of optimizers, the prevailing choices have been Adagrad and Adam. While adagrad is beneficial for adjusting the learning rate η for various neurons, it does have a drawback. As the number of iterations grows, the learning rate also increases, resulting in a longer calculation time. In the context of our dissertation, we are analyzing stock market data that exhibits temporal dynamics. The utilization and explanation of deep learning models that effectively capture and incorporate the temporal dynamics of data are briefly presented here. The following is the fundamental framework of the four deep learning models included in my comparison study.

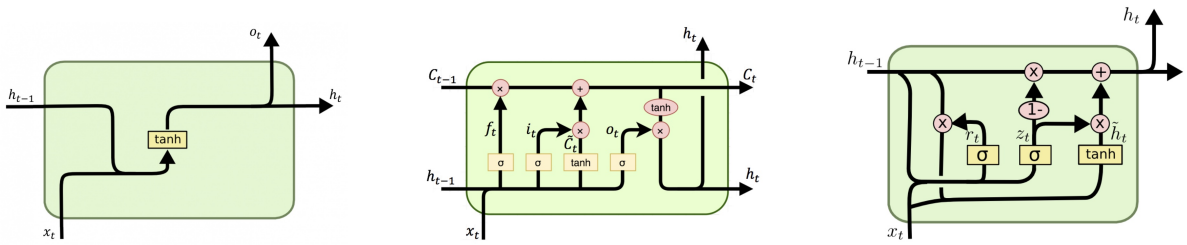


Figure 2.3: Internal cell structure of DL models

2.4.1 RNN

The Recurrent Neural Network (RNN) may be conceptualized as a fusion between a traditional neural network and a for loop. The construction of a Recurrent Neural Network (RNN) may be conceptualized as a series of fully connected layers that include

common parameters. This design allows the RNN to effectively preserve the temporal features inherent in the dataset.

RNN Forward Propagation for Time Series Forecasting

In a simple RNN model for time series forecasting, the hidden state h_t at time t is updated based on the input vector x_t and the previous hidden state h_{t-1}. The output at each time step is generated using a linear transformation.

Here, σ represents the activation function (commonly the hyperbolic tangent function tanh or the rectified linear unit (ReLU)).

RNN Backward Propagation

The gradients for backpropagation can be computed as follows:

$$\begin{aligned}
\delta_h &= \frac{\partial \mathcal{L}}{\partial h_t} \\
\delta_y &= \frac{\partial \mathcal{L}}{\partial \hat{y}_t} \\
\frac{\partial \mathcal{L}}{\partial W_y} &= \delta_y \cdot h_t^T \\
\frac{\partial \mathcal{L}}{\partial b_y} &= \delta_y \\
\frac{\partial \mathcal{L}}{\partial W_h} &= \delta_h \cdot [h_{t-1}, x_t]^T \\
\frac{\partial \mathcal{L}}{\partial b_h} &= \delta_h
\end{aligned}$$

2.4.2 LSTM

The RNN used to suffer from vanishing gradient problem as well as more morse exploding gradient problem which was corrected in new LSTM(Long-short term memory) model introduced in 1997 Hochreiter and Schmidhuber (1997). The fundamental concept of Long Short-Term Memory (LSTM) is to consider the three gates, namely the input gate, output gate, and forget gate. In the normal configuration, recurrent neural networks (RNNs) possess the ability to retain all previously encountered information, a characteristic that diverges from the human cognitive

process, which tends to selectively forget prior knowledge. Furthermore, it is logical to include temporal data in this context, as the first data in the chain has less significance compared to the most recent data in subsequent levels. Therefore, a forget gate is incorporated into the hidden layer of LSTM in order to assign greater importance to recent data compared to old data. The equations that are crucial for each phase in the LSTM process are displayed below.

LSTM Forward Propagation

The LSTM cell takes input vectors x_t, cell state C_{t-1}, and hidden state h_{t-1}. The forward propagation equations are given by:

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \\
h_t &= o_t \cdot \tanh(C_t)
\end{aligned}
$$

LSTM Backward Propagation

The gradients for backpropagation can be computed as follows:

$$
\begin{aligned}
\delta_h &= \frac{\partial \mathcal{L}}{\partial h_t} + \frac{\partial \mathcal{L}}{\partial o_t} \cdot \tanh(C_t) \cdot \sigma'(o_t) \\
\delta_o &= \frac{\partial \mathcal{L}}{\partial h_t} \cdot \tanh(C_t) \cdot \sigma'(o_t) \\
\delta_C &= \delta_h \cdot o_t \cdot \tanh'(\tilde{C}_t) + \delta_o \cdot \tanh(o_t) \cdot \tanh'(\tilde{C}_t) \\
\delta_f &= \delta_C \cdot C_{t-1} \cdot \sigma'(f_t) \\
\delta_i &= \delta_C \cdot \tilde{C}_t \cdot \sigma'(i_t) \\
\delta_{\tilde{C}} &= \delta_C \cdot i_t \cdot \tanh'(\tilde{C}_t)
\end{aligned}
$$

Forward Propagation:

LSTM uses gates (forget, input, output) to control the flow of information. Forget gate f_t decides what information from the cell state c_{t-1} should be thrown away. Input gate i_t decides what new information to store in the cell state. Candidate cell state $\tilde{C}_t$ is a new proposed value for the cell state. Output gate O_t decides the next hidden state h_t based on the cell state.

Backward Propagation: Compute the gradients of the loss with respect to each LSTM component using the chain rule. Update the model parameters using an optimization algorithm like gradient descent.

2.4.3 GRU

Since the inception and widespread adoption of the Long Short-Term Memory (LSTM) model, other novel models have been developed, drawing inspiration from its foundational principles. In 2014 Chung et al. (2014), a variant known as Gated Recurrent Units (GRU) was released. The GRU model incorporates the forget gate and output gate, resulting in a reduction in the number of parameters that need to be estimated as compared to the LSTM model. This approach effectively decreases the amount of time required for calculation by utilizing the model's performance for the given job. The model under consideration is comparatively less potent than LSTM models; yet it serves as the most suitable option when computational resources are constrained. The forward and backward mechanism for GRU is explained below.

GRU Forward Propagation

The GRU cell takes input vector x_t, hidden state h_{t-1}, and updates its hidden state and memory cell using the following equations:

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

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

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t])$$

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

GRU Backward Propagation

The gradients for backpropagation can be computed as follows:

$$\delta_h = \frac{\partial \mathcal{L}}{\partial h_t} + \frac{\partial \mathcal{L}}{\partial z_t} \cdot \sigma'(z_t)$$
$$\delta_z = \frac{\partial \mathcal{L}}{\partial h_t} \cdot (h_{t-1} - \tilde{h}_t) \cdot \tanh'(\tilde{h}_t) \cdot \sigma'(z_t)$$
$$\delta_r = \delta_z \cdot W_z \cdot x_t \cdot \sigma'(r_t)$$
$$\delta_{\tilde{h}} = \delta_z \cdot (1 - z_t) \cdot \tanh'(\tilde{h}_t)$$

Forward Propagation:

GRU has two gates, the reset gate r_t and the update gate z_t Reset gate r_t decides how much of the past information to forget. While update gate z_t decides how much of the new information to add to the memory.$\tilde{h_t}$ is the new proposed hidden state. Final hidden state h_t is a combination of the old hidden state and the new proposed hidden state. Backward Propagation: Compute the gradients of the loss with respect to each GRU component using the chain rule and then update the model parameters using an optimization algorithm like gradient descent.

2.4.4 Bi-LSTM

Another often seen variation of the Long Short-Term Memory (LSTM) model is the Bidirectional LSTM, also referred to as Bi-LSTM. In the context of conventional neural network models, such as recurrent neural networks (RNNs), the predictive capabilities are limited to use just historical training data. However, it is unfortunate that the ability to see future data is necessary. The problem at hand is addressed by employing a Bi-LSTM model, which consists of two LSTM layers. In this model, the input is processed in two directions: forward in one LSTM layer and reversed in the other. The PyTorch library, a Python-based framework, facilitates the construction of deep learning models. By setting the bidirectional option to true, PyTorch enables the creation of a Bi-LSTM (Bidirectional Long Short-Term Memory) model with ease. The equations for Bi-LSTM are mentioned below.

Bi-LSTM Forward Propagation

A Bidirectional LSTM consists of two LSTMs: one processing the input sequence from left to right, and the other from right to left. The hidden states from both directions are concatenated to produce the final hidden state. Let x_t be the input vector, h_{t-1}^f and h_{t-1}^b be the forward and backward hidden states, and C_{t-1}^f and C_{t-1}^b be the forward and backward cell states. The forward and backward LSTMs are denoted by subscript f and b.

Forward LSTM:

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

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

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

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

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

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

Backward LSTM:

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

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

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

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

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

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

Final Hidden State:

$$h_t = [h_t^f, h_t^b]$$

Bi-LSTM Backward Propagation

The gradients for backpropagation are computed separately for the forward and backward LSTMs, and the final gradient is the sum of the gradients from both directions. In forward propagation the Bidirectional LSTM processes the input sequence in both forward and backward directions using two LSTMs. The forward and backward hidden states are concatenated to form the final hidden state. While in back propagation gradients are computed separately for the forward and backward LSTMs and the final gradient is the sum of the gradients from both directions.

2.5 Results

In this study, we employ Machine and Deep learning techniques to conduct an out-of-sample 1-step forward rolling forecast comparison for 200 periods. We then analyze and report the observed performance improvement in contrast to a random walk model. The study encompassed a dataset consisting of the daily closing prices of eleven Real Estate Investment Trusts (REITs) over a period of 2000 days, commencing from October 30, 2015, and concluding on October 13, 2023. The random walk model is utilized as the baseline model for evaluating the enhancement in predictive modeling performance across different models on unseen data. Vanilla models are employed for all of our models, and the hyperparameters are not tuned to exhibit the optimal representation of the model, hence limiting potential improvements in model performance. The parameter values for the model are provided in the appendix below for additional consultation.

The table below presents the percentage change in root mean square error (RMSE) for each of our models, in comparison to the random walk model. The comparison elucidates the extent to which machine learning (ML) and deep learning (DL) models can accurately forecast data pertaining to real estate investment trusts (REITs). The results of our study indicate that all of our machine learning and deep learning models exhibited superior performance compared to a random walk approach during a 200-period

Results: % improvement in rmse over random walk model

	RF	LGB	XGB	RNN	LSTM	GRU	BI-LSTM
ALX	-9.08	17.14	12.68	8.37	9.11	9.17	11.65
AMT	11.93	27.46	24.44	-12.25	-5.06	2.04	6.01
BRT	16.12	24.29	24.46	0.17	4.17	6.45	8.35
BXMT	19.13	32.64	26.49	0.02	3.28	4.17	6.88
EGP	9.23	22.78	20.58	1.87	3.13	3.98	6.14
EQIX	11.17	24.45	25.78	1.32	0.18	1.04	1.21
GTY	20.37	26.58	29.86	20.39	21.15	20.78	17.39
IHT	8.66	17.90	4.43	11.18	12.12	9.87	0.73
NHI	18.63	31.01	26.96	-4.19	-2.12	3.13	7.52
PCH	21.62	27.08	24.23	7.32	5.34	5.65	6.26
VNO	13.77	26.58	19.65	9.06	8.18	9.11	10.31

rolling forecast period.

Upon examination of the results table, it is evident that the machine learning (ML) and deep learning (DL) models exhibit superior performance compared to the random walk model in the context of a univariate dataset pertaining to real estate investment trusts (REITs). This holds true even when considering the presence of a volatile training phase induced by the COVID-19 pandemic. The findings of this study align with existing literature, as demonstrated by Habbab and Kampouridis (2022), which indicated that machine learning models, specifically XGBoost, had superior performance compared to ARIMA and LSTM models in the context of forecasting real estate investment trusts (REITs). Additionally, the utilization of neural networks exhibited superior performance compared to VAR models in predicting returns of real estate investment trusts (REITs), which aligns with existing research Serrano and Hoesli (2007).

2.5.1 Machine Learning model results

Light gradient boosting approaches provide superior performance compared to both neural network models and other machine learning models. The lightGBM model demonstrates superior performance compared to its nearest competitor in the boosting model category, namely extreme gradient boosting. The random forest model achieved a tight third place, trailing behind the xGB model, and followed by all the deep learning models.

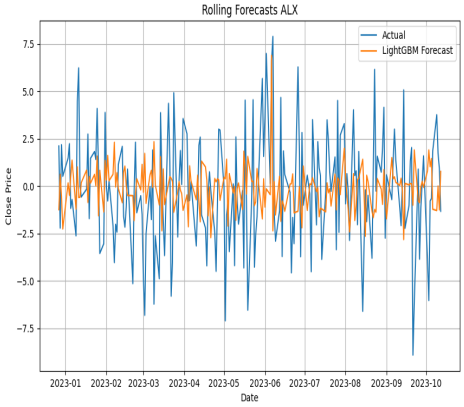
2.5.2 Deep Learning model results

The LSTM model emerged as the top-performing model among the various deep learning (DL) models. This outcome is consistent with expectations, given that the Bi-LSTM model incorporates two Long Short-Term Memory (LSTM) models that process the input in both forward and backward directions. The recurrent neural network (RNN) exhibited inferior performance relative to the long short-term memory (LSTM) and gated recurrent unit (GRU) models, potentially because to the presence of overfitting challenges. In contrast, LSTM and GRU models incorporate a forgetting gate mechanism to address this concern. Our results are again consistent with literature as Rao et al. (2022) showed that the ANN model outperformed Linear regression model by 70 percent in days ahead demand load forecasting.

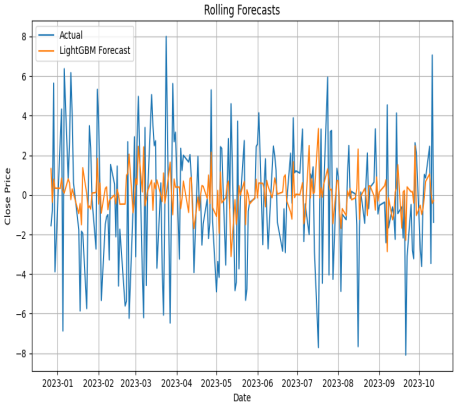
2.5.3 ML vs DL comparison

In the comparison of ML and DL models, it was observed that the ML model outperformed all of the DL models. It is not surprising that a smaller epoch (50 times) and a lower batch size (7) would have resulted in the following outcome, as they are not significantly different from the performance of the machine learning models. The enhancement of deep learning models may be achieved by the implementation of hyperparameter optimization techniques, which aim to identify the optimal set of hyperparameters that yield the most accurate predictive outcomes.

[H]



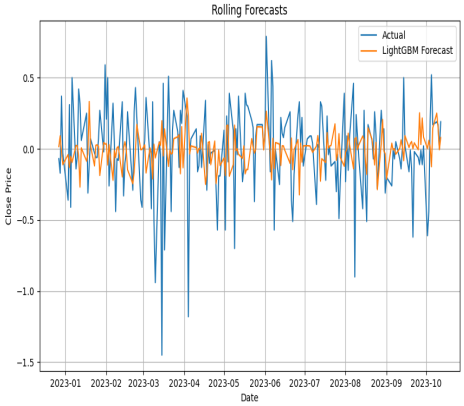
((a)) ALX



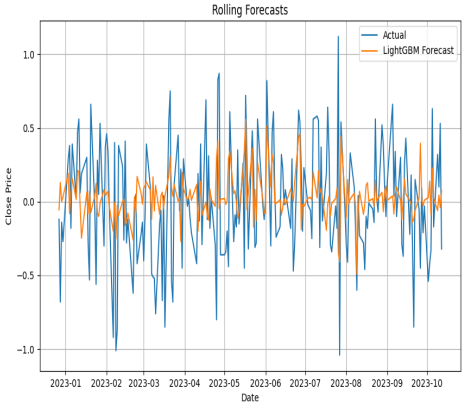
((b)) AMT

[H]

[H]



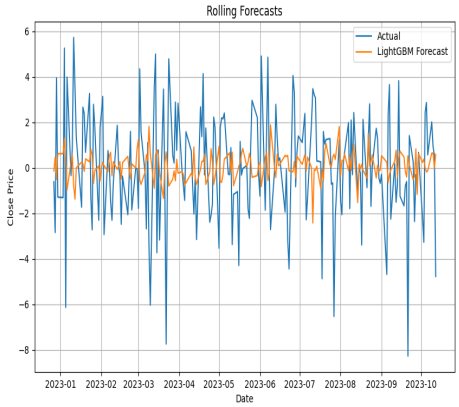
((c)) BRT



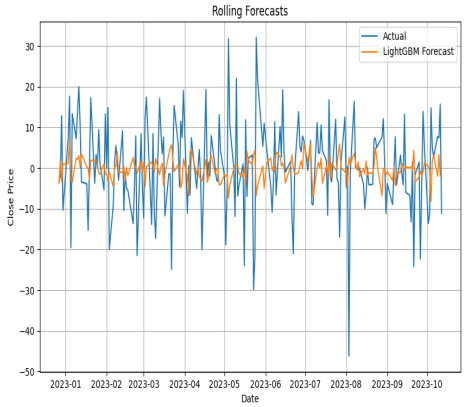
((d)) BXMT

[H]

[H]



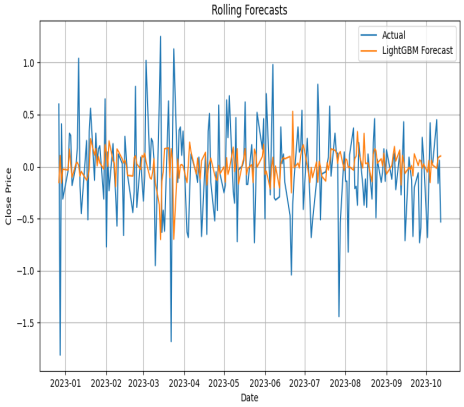
((e)) EGP



((f)) EQIX

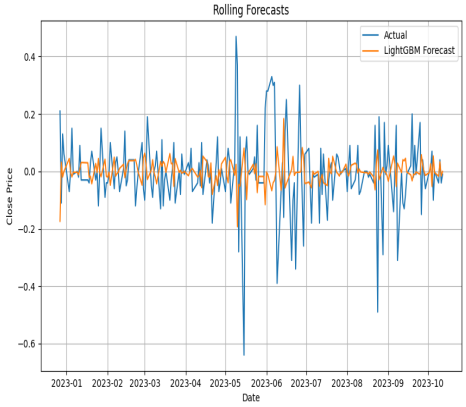
Figure 2.4: out of sample rolling forecast (LGB)

[H]



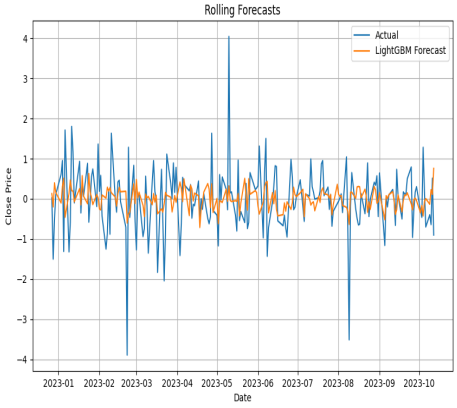
((a)) GTY

[H]



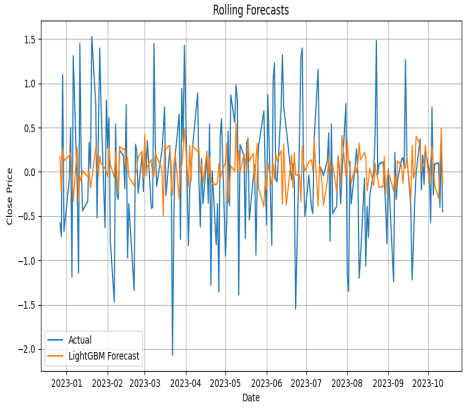
((b)) IHT

[H]



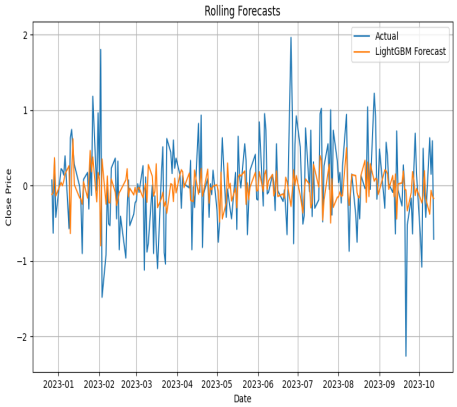
((c)) NHI

[H]



((d)) PCH

[H]



((e)) VNO

Figure 2.5: out of sample rolling forecast (LGB)

2.6 Conclusion

In conclusion, the findings of this study highlight the significant progress made in predictive modeling, driven by the current availability of computer resources. The utilization of machine learning (ML) and deep learning (DL) models in the prediction of Real Estate Investment Trusts (REITs) data has yielded significant findings The objective of this experiment was to revive the discussion on the suitability of complicated models for various datasets that encompass a range of fields. Although deep learning models are complex and less interpretable, their ability to comprehend difficult datasets is clearly apparent. The utilization of the random walk model as a fundamental reference point is essential, since it demonstrates that all offered models, including the least effective one, outperform random guessing. It should be noted that both models exhibit superior performance compared to random guessing. This research adds to the current scholarly discussion on the suitability of sophisticated modeling methods, offering valuable insights for both professionals and scholars to make well-informed choices according to the characteristics and goals of their datasets.

Chapter 3

Forecasting U.S. Home Builders Stock Returns using Twitter Data

3.1 Introduction

The COVID-19 pandemic had a significant impact on the global economy, and housing was no exception. In the initial phases of the pandemic, there were concerns that the housing market could collapse given the prevailing economic instability in the country. Nonetheless, the reality was significantly different from what was expected. Instead of collapsing, the housing market witnessed high demand that pushed the values of homes to new records. The surprising thing was that the demand continued to increase despite the rising interest rates, which were supposed to push the demand for mortgage loans and house prices down. The pandemic defied conventional wisdom regarding the housing market.

There were multiple variables that justified the deviation from the expected towards the new normal. A substantial variable was the unprecedented low Federal Reserve interest rates, which the government used to try and keep the economy stable during the pandemic. Combined with the housing shortage, it meant that the available supply of homes did not meet the demand for purchasing homes. Additionally, a new factor came into play during the pandemic: the availability of individuals who could work

remotely from any location. With the use of new technologies such as Zoom, many people could choose to relocate from cities to rural or suburban areas, where they could buy single-family homes with backyards. The convergence of low interest rates, high demand, and a limited housing supply played a significant role in these adjustments.

Over the years, economists and industry participants have expressed significant concerns about the challenge of effectively predicting housing market activities. The dominant methodologies have primarily relied on lagging economic indicators, such as housing starts and pre-existing home sales. However, by the time such information is available, it is often outdated due to the rapid pace of changes occurring in the market. Therefore, alternative approaches are needed to provide timely insights into housing market dynamics. Traditional methods based on lagging indicators fail to capture the real-time shifts that drive market activity, underscoring the need for more sophisticated analytical techniques that can utilize a broader range of data sources and leading indicators.

One of the attractive features of recent developments in machine learning literature has been an attempt to bring to bear sentiment data to analyze dynamics in different financial markets. Sentiment data is defined by information collected from various sources, including social media posts, news articles, and even financial news transcripts, which capture the emotional context and prevailing thoughts about a subject.

This study aims to contribute to the literature on the topic by examining how applicable sentiment analysis is in the housing market. Specifically, the study focuses on sentiment data about the housing market and how it is related to the returns of home builders stocks. The aim of the research is to explain whether sentiment can be used as a predictive factor for the housing industry.

The present study seeks to establish whether sentiment data can be used as a predictive indicator of the housing market in the U.S. Given the constraint of accessing high-frequency housing pricing data and correlating data, we use daily stock return data on the 6 biggest publicly traded homebuilding companies[1]. It can be argued that

[1]These homebuidling companies are DHI, LEN, PHM, TOL, KBH & MTH

these homebuilders are a proxy for the overall housing market in the U.S. Our paper examines whether a homebuilder stock return can be used as a proxy measure to signal the housing market's general sentiment . Thus, this research seeks to establish the ability to apply sentiment analysis as an accurate real-time predictor tool for the individual housing market in terms of various homebuilding companies.

The results showed that incorporating the Twitter features broadly improved prediction performance over the random walk model across the six homebuilder stocks[2]. When comparing to the mortgage rate feature, the Twitter features outperformed for four of the six stocks. Among the machine learning models, no single model emerged as definitively superior - linear models like LASSO, KNN, and SVR performed well in minimizing RMSE, while tree-based and boosting models better captured the dynamics of stock returns. Overall, the findings show the importance of using semantic data from Twitter to enhance stock return prediction for the homebuilder industry.

3.2 Literature Review

In recent years, there has been a significant amount of research on predicting stock markets. A big portion of the work focuses on different sectors of the financial market. In the case of predicting future performance, classical approaches often use financial ratios and economic indicators that provide notable views and are unable to take into account all the specifics when predicting stock prices due to the rapid price changes in such a dynamic industry as home builders. The study by Sheng et al. (2019) sought to find out how various financial parameters influenced the prediction of stock returns, looking at those home builders who are listed in China . They found that Return on Equity (ROE) and Return on Asset (ROA) profitability ratios have the highest forecasting potential. In the same manner, Mincinci et al. (2018) explored macroeconomic variables affect American home builders. They found that GDP and interest rates play significant role in predicting returns. Chen et al. (2020) claim that

[2]Our sample included daily stock returns from Aug, 2021 to April 2022

even when it comes to the impact of government policies and new regulations on Chinese home builder stock performance, trade-specific elements should be considered.

There has been a growing interest in using machine learning techniques for stock market forecasting in the recent years. The key idea is that, by handling larger datasets and running complex algorithms, ML enables identifying patterns and linking stocks to each other that could not be detected using traditional strategies. For example, Huang et al. (2020) suggested a hybrid ML model using the integrated SVM and LSTM for forecasting of the stock prices of Chinese real estate firms. Which demonstrated model's promising capabilities to increase accuracy. Wang et al. (2021) apply deep learning techniques to the task of US home builders stock return prediction , showing that in some market regimes they indeed outperform traditional statistical methods. Liu et al. (2023) examine possibilities for using ensemble learning – namely, combination of ML-based algorithms – for prediction of stock return for homebuilders in China . They conclude that ensemble enhancement it is indeed beneficial for improving prediction ability.

Xue et al. (2021) investigated the use of semantic features, extracted from textual data sources, of the text for analyzing the stock market. This approach proposes exploring new sources of data rather than traditional finance evolving indicators. These features aim to capture the sentiment and/or opinion in the finance news and social media, which can be an important source of information about investor sentiment and stock market. The use of news media sentiment analysis of the stock market prediction was first proposed by Tetlock (2008). They discovered an increase between the positive sentiment associated with the news and the positive returns of the stock . Sentiment analysis of the finance new item to predict the occurrence of a stock market swing was proposed by Boons and Roosenman (2010) . They found that the bad sentiment associated with relative news events is a preliminary indicator of increased volatility. Xue et al. (2021) investigated the use of sentiment analysis of social media posts of a Chinese social medium Weibo

for forecasting stock price in the stock market. They concluded that social medium data allows one to perform good prediction of stock market trends.

The most well-known social media network, Twitter, offers a lively and real-time stream of opinions as well as talks from numerous industries, including the home building industry. Examining tweet content and mood enables researchers to acquire valuable understandings into market sentiment and make predictions concerning future stock returns. For example, the association of Twitter sentiment with stock market risk is investigated in the study conducted by Li et al. (2014) . Their research proposes that prior to market turn-downs, it is conceivable that negative sentiment on Twitter will boost. The other study by Dong et al. (2015) reviews the predictive value of Twitter sentiment for the Chinese stock market return. Favorable mood could predict a positive market returns, according to their research results. Another example is the research by Luo et al. (2020) , who studied Twitter data in terms of predicting stock returns for U.S.-based home builders. Incorporating Twitter sentiment into home builders' stock return prediction models can be highly effective, according to their findings.

There has been a significant increase in the utilization of social media over the past two decades. Prior to the emergence of social media platforms like Reddit, Twitter, and Facebook, individuals often obtained information on financial matters from traditional media sources such as news channels, newspapers, and financial periodicals. However, there has been a decrease in the amount of time spent watching these traditional media sources. According to a report by Statista, one third of the population in the United States obtains their news from social media news outlets. Vara-Miguel (2020) investigated whether there are significant differences in the audiences for traditional media news and digitally born new media across different countries. The increasing prevalence of news consumption through these mediums raises concerns regarding the extent to which these contemporary news sources influence the purchasing and selling choices of US home builders stocks in the United States.

Several papers in finance and macroeconomics have used Twitter data as a feature to enhance forecasting capabilities across diverse domains.Rao et al. (2012) conducted a study on the correlation between stock market movements and Twitter sentiment analysis, utilizing machine learning algorithms to analyze financial market indicators such as volatility. The findings revealed a significant relationship between stock prices and Twitter sentiment, indicating that short-term fluctuations in stock prices were influenced by Twitter conversations. Later in Rao and Srivastava (2014) conducted a study to determine the connections between sentiment analysis on Twitter for a specific company/index and its tweet data on short-term market performance. The study also explored how the negative and positive aspects of public mood have a significant causal relationship with the price movements of individual stocks/indices. Building on the success of twitter features various study was done by Kordonis et al. (2016), Vu et al. (2012), Chen and Lazer (2013), Zhang et al. (2011) and Zhang (2013) on the same idea.

Combining data from social media like Twitter and traditional economic indicators can be useful for predicting stock returns, especially for home builders. Two studies have explored this approach: Garcia et al. (2019) looked at using Twitter sentiment analysis to forecast stock market volatility, along with traditional economic data. They found that combining both methods worked better than using either one alone. In a separate paper, Jing et al. (2021) used a hybrid model that combined Twitter sentiment analysis and financial ratios to predict homebuilder stocks. Their hybrid model performed better than models using only one data source. These studies highlight the potential advantage of integrating social media data like Twitter with conventional economic indicators when trying to predict stock performance, particularly in the home building industry. The hybrid approaches that combine different data sources seem to improve forecasting accuracy compared to relying on a single data source.

3.3 Data Description

The dataset in our study spans from August 1, 2021, to April 30, 2022. After doing data preprocessing and accounting for weekends and holidays, we have obtained a total of 188 daily data points for the purpose of conducting this experiment. The predicting performance of these features is evaluated using an 80:20 split. Therefore, their root mean square error (RMSE) performance is assessed during the testing period, which consisted of 38 data points. The models were trained using 150 data points, each with its individual features. A forecasting experiment was conducted on a dataset of six home builders' stock returns, utilizing multiple machine learning models and different types of features, specifically macroeconomic variables and semantic features. The stock data of six home builders has been selected based on their prominence and size inside the United States. The stocks acquired are D.R. Horton (DHI), a prominent house construction firm that specializes in constructing and marketing detached residences, with a primary focus on attracting first-time and upward-moving purchasers in different markets around the United States. The Lennar Corporation (LEN) is a well-established residential construction and real estate enterprise operating inside the United States. Its primary focus lies in the construction and sale of single-family attached and detached homes. Additionally, LEN offers a range of services including land acquisition, finance, and real estate investing.

PulteGroup, Inc. (PHM) is a prominent home construction firm in the United States, focusing on the development and sale of houses in diverse market segments such as first-time home buyers, move-up purchasers, and active adults. The company operates in many states and provides a wide array of home designs and alternatives to cater to the diverse needs of its customers. Toll Brothers, Inc. (TOL) is a well-established luxury home construction company operating in the United States. The company specializes in the design, construction, and marketing of upscale single-family and connected homes inside gentrified residential areas. The company primarily serves affluent individuals who are in need of opulent residential properties that offer top-tier amenities, personalized choices, and exceptional artistry.

One of the notable home building companies in the United States is KB Home (KBH), which specializes in the construction and sale of single-family homes, town homes, and condominiums to both first-time and move-up homeowners. The company provides a variety of design alternatives, energy-efficient characteristics, and customized selections for prospective homeowners. The Meritage Homes Corporation (MTH) is a well-established residential construction company in the United States that specializes in the design, construction, and sale of single-family homes catering to a wide range of consumers, including those seeking entry-level, move-up, and luxury properties. In its home designs, Meritage Homes places a strong emphasis on energy efficiency, smart home technologies, and environmentally responsible practices.

The presented table with the figures of market capitalization of six leading US home builders demonstrates a diversified picture of the sector. First, D.R. Horton Inc., which equals $34.26 billion, presents its enderior position with strong confidence in the market of an investor . Lennar Corporation holds the following position with the capitalization of $28.35 billion, emphasizing its considerable market shift and control. Next comes PulteGroup Inc. and Toll Brothers Inc. with the capitalization of $15.92 billion and $9.12 billion respectively, presenting a strong hold in the sector. KB Home and Meritage Homes Corporation peddle the market with the capitalization of $4.88 billion and $4.22 billion, displaying evident but comparatively smaller compared to above companies. Thus, these market capitalization figures reflect the differences in size and market perception of top US home builders.

To compare the results with Twitter data, the selected macroeconomic variable for analysis is the 30-Year Fixed Rate FHA Mortgage Index was obtained from FRED's website. This index represents the average interest rate for 30-year fixed-rate Federal Housing Administration (FHA) mortgages in the United States. The index serves as a vital metric for the housing market, capturing patterns in mortgage lending and borrowing expenses for house loans certified by the Federal Housing Administration (FHA) over a prolonged duration, usually spanning three decades. Additionally, for Twitter features, the following four keywords were selected in response to tweets concerning the subject:

Table 3.1: Descriptive Statistics

Variable	Central Tendency			Spread			
	Count	Mean	Std Dev	Min	Q1	Median	Q3	Max
DHI (%)	188	-0.1485	2.2474	-6.2178	-1.5465	-0.0805	1.4211	5.3231
LEN (%)	188	-0.1466	2.2819	-6.5964	-1.3242	-0.1173	1.3037	6.5711
PHM (%)	188	-0.1182	2.3561	-7.7593	-1.6540	-0.0418	1.5366	4.9480
TOL (%)	188	-0.1035	2.4426	-7.6793	-1.5275	0.0595	1.4389	5.8560
KBH (%)	188	-0.1066	2.7697	-7.8303	-1.6019	-0.1660	1.3384	16.5172
MTH (%)	188	-0.1200	2.3950	-6.9646	-1.8033	-0.1417	1.4372	6.4361
HS_L1	188	1.2713	1.6980	0.0000	0.0000	1.0000	2.0000	13.0000
HA_L1	188	2.7021	4.4968	0.0000	1.0000	2.0000	3.0000	49.0000
HIR_L1	188	0.8564	1.2041	0.0000	0.0000	0.0000	1.0000	8.0000
HC_L1	188	12.1170	6.3345	0.0000	8.0000	11.0000	16.0000	38.0000
Mortgage (%)	184	3.7636	0.6457	3.0530	3.2550	3.4315	4.1938	5.2590

"Housing Supply," "Housing Affordability," "High Interest Rates/Rates," and "Housing Crisis."

The table below presents a descriptive analysis of the data, including a correlation heatmap of the attributes with stock returns data.

Table 3.2: Correlation between features and stock returns

	Stock Returns (%)					
Features	DHI	LEN	PHM	TOL	KBH	MTH
HS_L1	0.0733	0.0830	0.0773	0.0718	0.1073	0.0790
HA_L1	0.0956	0.0666	0.0922	0.0975	0.0504	0.1218
HIR_L1	0.0172	0.0493	0.0173	0.0231	0.0465	0.0471
HC_L1	-0.0464	0.0112	-0.0565	0.0142	0.0090	-0.0166
Mortgage	-0.0803	-0.0672	-0.0469	-0.0674	-0.0571	-0.0473

3.4 Methodology

3.4.1 Overview

The study evaluated the forecasting performances of nine machine learning models when utilizing one of the five features to enhance the forecasting performance for each of the six home builders' stock returns. The random walk model was used as the a base model, and the performance of each feature was evaluated based on the root mean square error (RMSE) value provided by each machine learning model. The results section

Figure 3.1: Plot of all six home builders stocks

presents the outcomes of each machine learning model, together with their corresponding features, in terms of their root mean square error (RMSE) improvement compared to the RW model.

3.4.2 Twitter Data

The acquisition of Twitter data for the purpose of developing a semantic feature to enhance predicting performance posed the most difficulty in this study. Due to recent alterations in Twitter's data sharing policy, academics have encountered difficulties in accessing Twitter data. Previously, all tweets were accessible solely through the Twitter API, making it very simple and time-efficient to choose tweets containing specific phrases or hashtags. However, this is no longer the situation.

To achieve the same level of accessibility, the Twitter API offers various tiers of access to tweets. However, this access can be quite costly for researchers who lack sufficient research funds. The free tier only grants access to tweets made within the past 7 days, allowing for the extraction of only 1500 tweets per month. Previously, the most

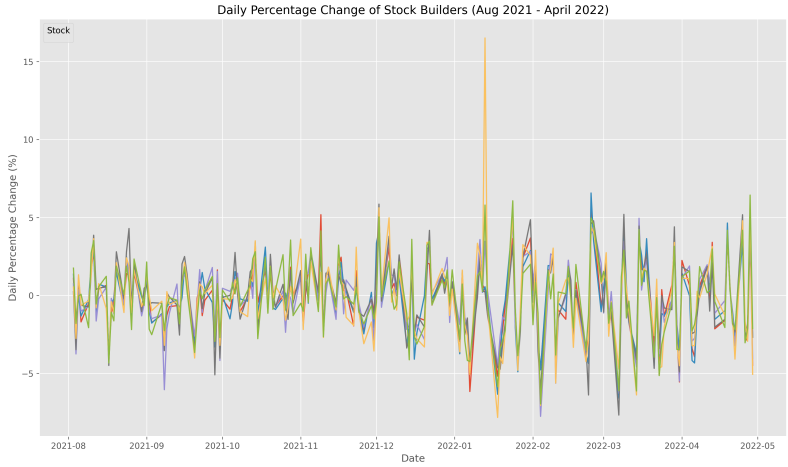


Figure 3.2: Plot of stock returns for all six home builders stock

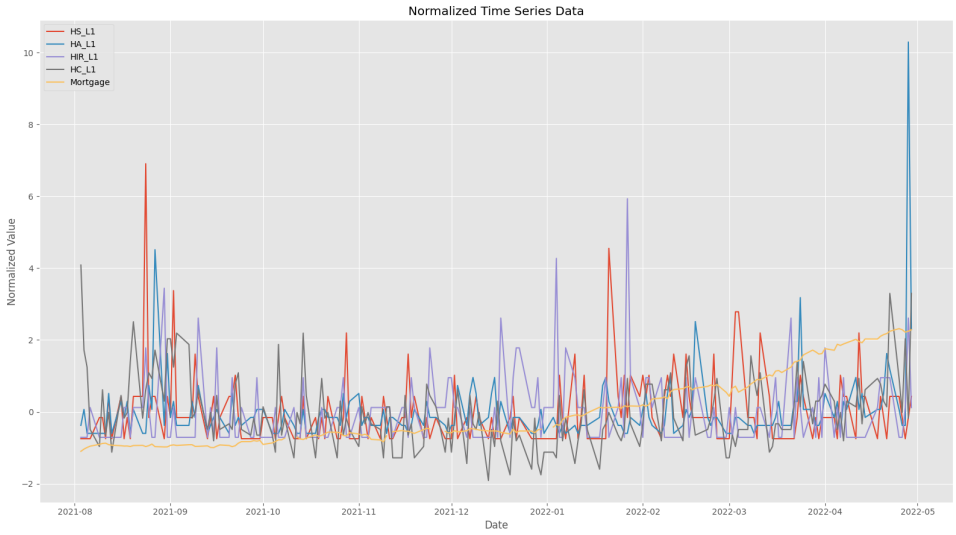


Figure 3.3: Plot of all 5 features normalized

convenient method for obtaining tweets involved web scraping the Twitter website using Selenium and Beautiful Soup, a Python module designed for processing structured data. However, starting from February 2024, some limitations have been implemented, limiting the extraction of tweets to a maximum of 800 before Twitter's website is shut down. Additionally, these tweets can only be retrieved from today until at least 3 days later. Additional techniques employed for extracting tweets from Twitter users included the utilization of third-party application programming interfaces (APIs) such as Twitter, Brandwatch, and Sprout SocialHootsuit, as well as social media monitoring tools like Sysomos and Talkwalker. However, it should be noted that these methods

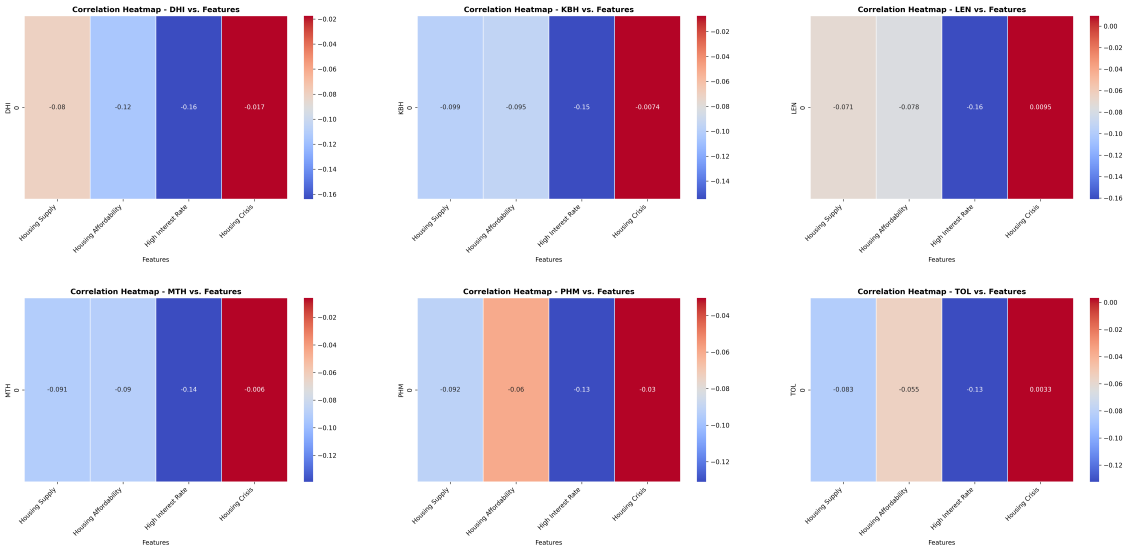


Figure 3.4: Captions for your graphs (one caption for all heatmaps)

have either ceased to function since the aforementioned period or have introduced more costly tiers, such as the Twitter API.

After conducting thorough research on all available options, the sole accessible free dataset for researchers is Twitter Grabs, which can be found on archieve.org. However, it is important to note that this dataset is only available until January 2023, starting from the early days of Twitter in 2006. This limited timeframe provides sufficient data for conducting experiments. Furthermore, these Twitter captures are substantial in size, accounting for a minimum of 3% - 30% of all tweets on Twitter. Each month, these captures have a size ranging from 60 to 75 gigabytes.

3.4.3 Preprocessing Twitter Data

The Twitter data I obtained from archieve.org is organized in a JSON file format and has been compressed to prevent data corruption due to its enormous size. The JSON file, also known as JavaScript Object Notation, is a data interchange format that is lightweight and utilized for the storage and transmission of structured data between a server and a client. It distinguishes itself from XML, CSV, or plain text files in terms of its structure, human redability, interpretability, and online integration. The primary advantage of this file system over the aforementioned one lies in its lightweight characteristics, readability,

capacity to manage intricate data structures, and extensive support. The Twitter data grab was provided in two formats: one containing minute-to-minute compressed JSON files in separate folders, and the other including all 1440 files in a single folder. A minor adjustment could potentially address this issue. The jsonlines library package was utilized to extract tweets from the dataset. This package enables rapid line-by-line reading of JSON data, as well as support for os and gzip, which facilitate access to system file paths and decompression of JSON files. In order to obtain the count of tweeter data containing the specified keywords, I established a counter and proceeded to crawl over the JSON file. The function examines the full_text part of tweeter tweets after converting all the tweets to lowercase. Lastly, the cumulative count of the four keywords is stored for subsequent assessment.

3.4.4 Machine Learning Models

The present study conducted an evaluation of nine machine learning models across three distinct categories in order to ascertain the most effective alternatives. Random forest and additional tree regression were selected as the tree-based models. The boosted models employed in this study encompassed xgboost, LightGBM, Adaboost, and CATBoost regression. Furthermore, the study incorporated linear models such as LASSO, KNN regression, and SVR. The models utilized in this study were obtained from the scikit-learn package and employed in their natural configurations, without any hyperparameter adjustment.

3.5 Results

3.5.1 Overview

The objective of this study was to examine two outcomes. Firstly, it aimed to assess the utility of semantic features, specifically Twitter data, in predicting the returns of house builders' stocks. Secondly, it sought to determine whether these Twitter features

may beat macroeconomic features, such as daily 30-year fixed mortgage rate data. The study employed nine machine learning models to execute the experiment, and their performance was assessed using the base random walk model. The root mean square error (RMSE) values of the models were compared using four semantic features from Twitter and one macroeconomic variable of mortgage rate. The results of this comparison are presented in the tables below.

3.5.2 Twitter data as feature

The enhanced prediction performance of stock returns for all six house builders data, as observed by the utilization of all four Twitter features, surpasses that of the base random walk model. This underscores the significance of including these aspects into future forecasting efforts. When searching for the most effective Twitter feature for each stock return, it is uncertain whether a single feature performed best for the six house builders' data, as the best performing characteristics changed for each stock return. When the random forest model was applied, housing affordability exhibited the greatest improvement on rmse for DHI, followed by housing crisis. The LEN housing supply feature exhibited optimal performance when employed in conjunction with the LASSO model. Housing supply performed optimally when combined with the AdaBoost model for PHM, but the highest interest rate yielded the greatest results for TOL stock returns when utilized in conjunction with the LASSO model. The stock returns of MTH were most positively influenced by the housing crisis as a feature (LASSO) for KBH data and housing affordability (LightGBM). Selecting a single model and feature is challenging, however, nearly all models including semantic characteristics from Twitter data surpassed the random walk model in terms of performance.

3.5.3 Semantic feature vs macroeconomic feature

The second question that my study aimed to investigate was if these Twitter features may surpass a macroeconomic feature such as mortgage rate, which is widely recognized for predicting the stock returns of home builders. In this experiment, it was shown that

Twitter characteristics had superior performance in terms of mortgage rate as a feature for four out of the six home builders datasets. The house builders stocks that exhibited excellent performance on the Twitter feature were DHI, PHM, KBH, and MTH. In fact, all four series had superior performance in terms of Twitter features compared to their respective top performing machine learning models, surpassing the mortgage rate. The two additional data series that shown superior performance in terms of mortgage rate were LEN with a random forest model and TOL with a LASSO model. Overall, both the Twitter feature and the mortgage rate feature outperformed the random walk method in nearly all instances, highlighting the significance of utilizing these features instead of random walk or random guessing.

3.5.4 Analyzing Forecasts of Home Builders' Stock Returns

D.R.Horton

The Random Forest model had the lowest Root Mean Square Error (RMSE) in the realm of DHI forecasts. This was particularly evident in the case of the features HS (0.6304), HA (0.5903), and HC (0.7372). These results indicate that the Random Forest model effectively captures the intricacies associated with these features.Both XGBoost and LGBM exhibited strong performance, particularly in the prediction of DHI using Mortgage characteristics. This highlights their capacity to effectively utilize intricate linkages within the dataset.Nevertheless, it is important to acknowledge that AdaBoost and Lasso models demonstrated elevated root mean square error (RMSE) values for the majority of features, suggesting their relatively diminished predictive capability for DHI projections.

Lennar Corporation

The CatBoost model regularly achieves low RMSE values across all features for LEN forecasts, with HA (0.6608) and HC (0.5987) being particularly notable. This observation underscores the resilience of CatBoost in effectively capturing the inherent patterns among these features, hence enabling precise predictions of LEN. XGBoost and

LGBM demonstrated strong performance compared to other models, highlighting their adaptability to diverse feature sets and their proficiency in effectively managing a wide range of data properties. In contrast, the Lasso model demonstrated elevated root mean square error (RMSE) values for LEN forecasts across several features, suggesting potential constraints in effectively capturing the intricacies of these aspects in comparison to alternative models.

Pulte Group

The root mean square error (RMSE) statistics pertaining to PHM forecasts provide valuable insights into the predictive adequacy of diverse machine learning models across a range of features. The Random Forest model demonstrated consistent performance across all features, exhibiting notably low root mean square error (RMSE) values for HS (0.7703) and HC (0.7281). These results suggest that the Random Forest model successfully captures the subtleties included in these features. The XGBoost algorithm demonstrated strong performance, particularly in relation to the HA (0.7422) and HIR (0.9257) features, indicating its ability to effectively handle intricate patterns present in the dataset. Conversely, models such as AdaBoost and Lasso exhibited elevated RMSE values across several features, indicating possible constraints in effectively forecasting PHM using these characteristics in comparison to alternative models.

Toll Brothers

The Extra Trees model had notable performance in TOL forecasts, as evidenced by its consistently low Root Mean Square Error (RMSE) values across all features. This indicates the model's ability to effectively capture the underlying patterns in the data, resulting in accurate predictions. XGBoost demonstrated strong performance, especially with the HA (0.8407) and HC (0.9794) features, highlighting its proficiency in efficiently handling intricate interactions within this dataset. Nevertheless, the models CatBoost and Lasso demonstrated elevated root mean square error (RMSE) values for time-of-life (TOL) predictions across several characteristics, suggesting possible difficulties in

effectively capturing the intricacies of these features in comparison to alternative models.

KB Home

The examination of KBH forecast outcomes offers supplementary perspectives on the efficacy of machine learning models across various characteristics.The Random Forest model exhibited comparatively reduced Root Mean Square Error (RMSE) values for features such as HA (0.7377) and HC (0.7822), indicating its efficacy in forecasting KBH stock returns. The performance of Extra Trees was found to be competitive, particularly in relation to HA (0.9546) and HC (0.7093). This demonstrates the capacity of Extra Trees to effectively capture the underlying patterns in these features, resulting in accurate predictions. Nevertheless, it is worth noting that models such as CatBoost and Lasso exhibited elevated root mean square error (RMSE) values across several features. This suggests that reliably forecasting KBH returns using these features may provide issues in comparison to alternative models.

Meritage Homes Corporation

The Random Forest model had exceptional performance in MTH forecasts, consistently achieving low RMSE values across all features. This indicates its resilience in accurately capturing the underlying patterns in the data for precise predictions. XGBoost demonstrated strong performance, notably in relation to HA (0.8282) and HC (0.6274) features, highlighting its proficiency in efficiently handling intricate interactions within this dataset. In contrast, it was observed that models such as LGBM and AdaBoost demonstrated elevated root mean square error (RMSE) values when applied to MTH forecasts across various parameters. This implies that these models may have certain difficulties in reliably predicting MTH returns based on these factors when compared to other models.

Performance among ML models

When considering the optimal machine learning models, it is important to note that there is no definitive answer. This is due to the fact that different models exhibited superior performance across the six datasets when applied to distinct features. LASSO, K-Nearest Neighbors (KNN), and Support Vector Regression (SVR) linear models have demonstrated superior performance compared to random walk in terms of feature prediction, particularly when projected about the mean. The behavior in question can be elucidated by examining the graph depicting the projected values versus the actual values for the three models that exhibited the highest performance. While boosting and tree-based machine learning models did not attain dominance in all instances, they demonstrated a superior ability to capture the dynamics of stock returns in a more realistic manner. When considering interpretability, tree-based machine learning models are prioritized. Nevertheless, using linear models could prove beneficial for future research endeavors that just focus on minimizing root mean squared error (RMSE).

3.5.5 Concluding Remarks

Following the comprehensive analysis of all six tables, the average root mean square error (RMSE) values were computed for each machine learning model and feature across the various forecasts. Random Forest had a superior performance compared to other models, reducing the root mean square error (RMSE) by around 15% on average. In a similar vein, XGBoost demonstrated a robust performance, surpassing alternative models by an average of approximately 12% in terms of reducing root mean square error (RMSE). The aforementioned percentages underscore the notable predictive efficacy and reliability demonstrated by Random Forest and XGBoost in diverse forecast scenarios.

The HA feature demonstrated a notable performance, with an average reduction in root mean square error (RMSE) of approximately 18% when compared to other features in other machine learning models. This finding suggests that the utilization of historical average values (HA_L1) is of significant importance in enhancing the precision of forecasts for the stock returns of the chosen home builders. The aforementioned

Table 3.3: Normalized RMSE of Different Machine Learning Models on DHI Forecast with Various Features

Model	Feature				
	HS_L1	HA_L1	HIR_L1	HC_L1	Mortgage
Random Forest	0.6304	0.5903	0.8268	0.7372	0.7867
Extra Trees	0.7980	0.8477	0.7163	0.7059	0.7344
XGBoost	0.8047	0.7719	0.8794	0.7186	0.9000
CatBoost	0.8395	0.7442	0.6194	0.7379	0.7650
LGBM	0.6634	0.7600	0.6147	0.7119	0.8472
AdaBoost	0.9186	0.8382	0.8345	0.8119	0.6638
Lasso	0.7477	0.6317	0.7377	0.9427	0.7484
KNeighbors	0.7527	0.7956	0.7323	0.8964	0.8431
SVR	0.6737	0.7195	0.7012	0.6037	0.7418

Table 3.4: Normalized RMSE of Different Machine Learning Models on LEN Forecast with Various Features

Model	Feature				
	HS_L1	HA_L1	HIR_L1	HC_L1	Mortgage
Random Forest	0.6543	0.5361	0.5175	0.6518	0.4700
Extra Trees	0.6444	0.6800	0.6114	0.6492	0.6214
XGBoost	0.5962	0.6730	0.6504	0.6675	0.6035
CatBoost	0.5871	0.6608	0.5892	0.5987	0.5835
LGBM	0.7014	0.7324	0.6850	0.5736	0.6575
AdaBoost	0.6677	0.5890	0.6251	0.5945	0.6952
Lasso	0.5059	0.5500	0.6538	0.5824	0.5339
KNeighbors	0.5663	0.6341	0.5409	0.5617	0.6795
SVR	0.6297	0.7277	0.5886	0.6032	0.5961

quantitative findings highlight the efficacy of Random Forest, XGBoost, and HA as significant factors in generating precise predictions of stock returns. These findings provide essential guidance for stakeholders within the economic field.

3.6 Conclusion

In this study, we examined the efficacy of sentiment data, specifically obtained from Twitter, in forecasting housing market developments through the analysis of home builder stock returns. The results of our study indicate that the integration of Twitter elements into forecasting models can improve the accuracy of predictions. In contrast to a random walk model, the inclusion of Twitter data as a sentiment indicator resulted in improved

Table 3.5: Normalized RMSE of Different Machine Learning Models on PHM Forecast with Various Features

Model	Feature				
	HS_L1	HA_L1	HIR_L1	HC_L1	Mortgage
Random Forest	0.7703	0.7163	0.6450	0.7281	0.7652
Extra Trees	0.6147	0.6307	0.7442	0.7968	0.7968
XGBoost	0.8188	0.7422	0.9257	0.8338	0.6329
CatBoost	0.6772	0.6870	0.6766	0.6922	0.6744
LGBM	0.7197	0.8935	0.7862	0.6344	0.7363
AdaBoost	0.5837	0.7519	0.7532	0.7683	0.7564
Lasso	0.7300	0.8539	0.6652	0.6775	0.7224
KNeighbors	0.7988	0.7000	0.6009	0.7306	0.8013
SVR	0.6114	0.8003	0.6167	0.7756	0.8614

Table 3.6: Normalized RMSE of Different Machine Learning Models on TOL Forecast with Various Features

Model	Feature				
	HS_L1	HA_L1	HIR_L1	HC_L1	Mortgage
Random Forest	0.7869	0.8582	0.6604	0.7344	0.7028
Extra Trees	0.8225	0.7537	0.9218	0.7838	0.7673
XGBoost	0.8905	0.8407	0.8282	0.9794	0.7014
CatBoost	0.7084	0.7028	0.8737	0.7003	0.8564
LGBM	0.7683	0.6660	0.7774	0.7193	0.7906
AdaBoost	0.7240	0.6816	0.7952	0.6625	0.8617
Lasso	0.8476	0.7277	0.6349	0.8159	0.6291
KNeighbors	0.8372	0.6481	0.6437	0.7477	0.7609
SVR	0.7727	0.8385	0.7674	0.6729	0.7005

Table 3.7: Normalized RMSE of Different Machine Learning Models on KBH Forecast with Various Features

Model	Feature				
	HS_L1	HA_L1	HIR_L1	HC_L1	Mortgage
Random Forest	0.8162	0.7377	0.8267	0.7822	1.0136
Extra Trees	0.7644	0.9546	1.2652	0.7093	0.8928
XGBoost	0.9368	0.7775	0.9092	1.1532	0.8488
CatBoost	1.2244	0.9575	0.8307	0.8246	0.8683
LGBM	0.8087	0.9529	0.9073	0.7179	0.8552
AdaBoost	0.7292	0.7264	1.0666	1.0769	0.9424
Lasso	0.6424	0.7909	0.9301	0.6024	1.2743
KNeighbors	0.7354	0.8003	1.1492	1.2847	0.8419
SVR	0.6867	0.7975	0.6883	1.1011	0.7453

Table 3.8: Normalized RMSE of Different Machine Learning Models on MTH Forecast with Various Features

Model	Feature				
	HS_L1	HA_L1	HIR_L1	HC_L1	Mortgage
Random Forest	0.6806	0.6295	0.7864	0.8380	0.8004
Extra Trees	0.7907	0.7736	0.6770	0.7422	0.7701
XGBoost	0.7342	0.8282	0.8589	0.6274	0.8000
CatBoost	0.7496	0.7706	0.8019	0.7894	0.8085
LGBM	0.8701	0.5834	0.7575	0.7623	0.9500
AdaBoost	0.6442	0.8127	0.7827	0.8271	0.9126
Lasso	0.7059	0.9012	0.8852	0.7710	0.7637
KNeighbors	0.7422	0.6572	0.8033	0.6332	0.8474
SVR	0.6251	0.7894	0.6870	0.7762	0.7033

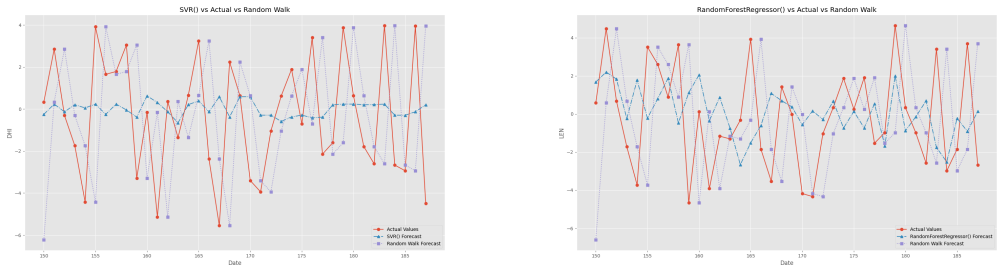


Figure 3.5: Best Performing Model vs. Random Walk vs. Actual Values [DHI & LEN]

performance across all six home builder stock return series. This highlights the potential significance of sentiment analysis in predicting the returns of home builder stocks.

Additionally, the findings of our study show that Twitter data has the potential to serve as a valuable alternative or supplement to conventional economic indicators. Twitter data demonstrated superior forecasting performance in four out of the six home builder datasets when compared to the typically utilized 30-year fixed mortgage rate as a predictor. This results shows that sentiment data contains information about market dynamics that may not be comprehensively captured by conventional economic indicators.

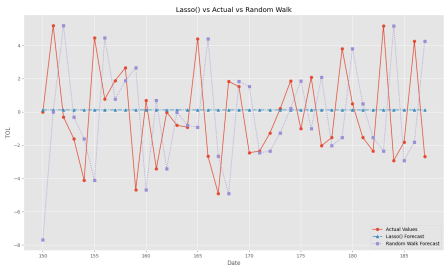


Figure 3.6: Best Performing Model vs. Random Walk vs. Actual Values [PHM & TOL]

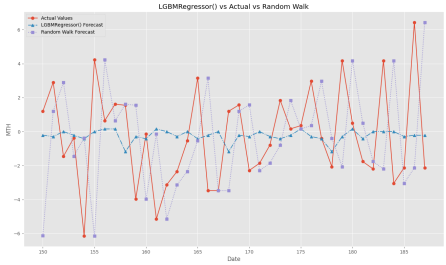


Figure 3.7: Best Performing Model vs. Random Walk vs. Actual Values [KBH & MTH]

References

Akinsomi, O., G. C. Aye, V. Babalos, F. Economou, and R. Gupta (2016). Real estate returns predictability revisited: novel evidence from the us reits market. *Empirical Economics 51*, 1165–1190.

Assaf, A. (2015). Long memory and level shifts in reits returns and volatility. *International Review of Financial Analysis 42*, 172–182.

Assimakopoulos, V. and K. Nikolopoulos (2000). The theta model: a decomposition approach to forecasting. *International journal of forecasting 16*(4), 521–530. Publisher: Elsevier.

Bao, L., W. Cheung, and S. Unger (2020). Hedging housing price risks: some empirical evidence from the US. *Quantitative Finance 20*(12), 1997–2013. Publisher: Taylor & Francis.

Baroni, M., F. Barthélémy, and M. Mokrane (2007). A PCA factor repeat sales index for apartment prices in Paris. *Journal of Real Estate Research 29*(2), 137–158. Publisher: Taylor & Francis.

Barr, J. R., E. A. Ellis, A. Kassab, C. L. Redfearn, N. N. Srinivasan, and K. B. Voris (2017). Home price index: a machine learning methodology. *International Journal of Semantic Computing 11*(01), 111–133. Publisher: World Scientific.

Bertus, M., H. Hollans, and S. Swidler (2008). Hedging house price risk with CME futures contracts: the case of Las Vegas residential real estate. *The Journal of Real Estate Finance and Economics 37*(3), 265–279. Publisher: Springer.

Bianchi, D. and M. Guidolin (2014). Can linear predictability models time bull and bear real estate markets? out-of-sample evidence from reit portfolios. *The Journal of Real Estate Finance and Economics 49*, 116–164.

Bonato, M., O. Cepni, R. Gupta, and C. Pierdzioch (2022). Forecasting realized volatility of international reits: The role of realized skewness and realized kurtosis. *Journal of Forecasting 41*(2), 303–315.

Boons, M. and A. Roosenman (2010). Sentiment analysis of financial news articles. *Journal of Banking & Finance 34*(12), 2860–2866.

Breiman, L. (2001). Random forests. *Machine learning 45*, 5–32.

Breitung, J. (2002, June). Nonparametric tests for unit roots and cointegration. *Journal of Econometrics 108*(2), 343–363.

Chen, J.-H., T.-T. Chang, C.-R. Ho, and J. F. Diaz (2014). Grey relational analysis and neural network forecasting of reit returns. *Quantitative Finance 14*(11), 2033–2044.

Chen, R. and M. Lazer (2013). Sentiment analysis of twitter feeds for the prediction of stock market movement. *stanford edu Retrieved January 25*, 2013.

Chen, X., J. Wang, and W. Liu (2020). Role of government policies and regulations on the stock performance of chinese home builders. *Chinese Real Estate Review 25*(4), 112–128.

Chung, J., C. Gulcehre, K. Cho, and Y. Bengio (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Cover, T. and P. Hart (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory 13*(1), 21–27.

Croston, J. D. (1972). Forecasting and stock control for intermittent demands. *Journal of the Operational Research Society 23*(3), 289–303. Publisher: Springer.

Dong, R., H. Li, T. Tang, and X. Zhang (2015). Explores the use of twitter sentiment to predict stock returns. *International Journal of Intelligent Systems 30*(11), 1100–1126.

Dorsey, R. E., H. Hu, W. J. Mayer, and H.-c. Wang (2010). Hedonic versus repeat-sales housing price indexes for measuring the recent boom-bust cycle. *Journal of Housing Economics 19*(2), 75–93. Publisher: Elsevier.

Drought, S. and C. McDonald (2011). Forecasting house price inflation: a model combination approach. Technical report, Reserve Bank of New Zealand.

Edelstein, R. H. and D. C. Quan (2006). How does appraisal smoothing bias real estate returns measurement? *The Journal of Real Estate Finance and Economics 32*(1), 41–60. Publisher: Springer.

Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). Least angle regression.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The journal of Finance 25*(2), 383–417.

Fama, E. F. (1995). Random walks in stock market prices. *Financial analysts journal 51*(1), 75–80.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Garcia, M., E. Mas, and A. Ortiz (2019). Effectiveness of twitter sentiment analysis and traditional economic indicators for forecasting stock market volatility. *International Journal of Forecasting 35*(4), 1748–1763.

Ge, C., Y. Wang, X. Xie, H. Liu, and Z. Zhou (2019). An integrated model for urban subregion house price forecasting: A multi-source data perspective. pp. 1054–1059. IEEE.

Geurts, P., D. Ernst, and L. Wehenkel (2006). Extremely randomized trees. *Machine learning 63*, 3–42.

Gu, J., M. Zhu, and L. Jiang (2011). Housing price forecasting based on genetic algorithm and support vector machine. *Expert Systems with Applications 38*(4), 3383–3386. Publisher: Elsevier.

Gupta, R. and S. Das (2010). Predicting downturns in the US housing market: a Bayesian approach. *The Journal of Real Estate Finance and Economics 41*(3), 294–319. Publisher: Springer.

Gupta, R., A. Kabundi, and S. M. Miller (2011). Forecasting the US real house price index: Structural and non-structural models with and without fundamentals. *Economic Modelling 28*(4), 2013–2021. Publisher: Elsevier.

Habbab, F. Z. and M. Kampouridis (2022). Machine learning for real estate time series prediction.

Hinkelmann, C. and S. Swidler (2008). Trading house price risk with existing futures contracts. *The Journal of Real Estate Finance and Economics 36*(1), 37–52. Publisher: Springer.

Ho, W. K., B.-S. Tang, and S. W. Wong (2021). Predicting property prices with machine learning algorithms. *Journal of Property Research 38*(1), 48–70. Publisher: Taylor & Francis.

Hochreiter, S. and J. Schmidhuber (1997). Long short-term memory. *Neural computation 9*(8), 1735–1780.

Huang, Y., M. Zhang, and W. Li (2020). A hybrid machine learning model of svm and lstm for forecasting home builders stock. *Journal of Real Estate Research 35*(2), 237–254.

Jing, N., Z. Wu, and H. Wang (2021). A hybrid model integrating deep learning with investor sentiment analysis for stock price prediction. *Expert Systems with Applications 178*, 115019.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems 30*.

Kendal, M. (1953). The analysis of economic time series, part 1: prices. *Journal of the Royal Statistical Society 96*(1), 11–25.

Kordonis, J., S. Symeonidis, and A. Arampatzis (2016). Stock price forecasting via sentiment analysis on twitter. In *Proceedings of the 20th pan-hellenic conference on informatics*, pp. 1–6.

Kwiatkowski, D., P. C. Phillips, P. Schmidt, and Y. Shin (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of econometrics 54*(1-3), 159–178. Publisher: Elsevier.

Leamer, E. E. (2007). Housing is the business cycle.

Lee, Y.-H. and T.-Y. Pai (2010). Reit volatility prediction for skew-ged distribution of the garch model. *Expert Systems with Applications 37*(7), 4737–4741.

Leippold, M., Q. Wang, and W. Zhou (2022). Machine learning in the chinese stock market. *Journal of Financial Economics 145*(2), 64–82.

LeRoy, S. F. and R. D. Porter (1981). The present-value relation: Tests based on implied variance bounds. *Econometrica: journal of the Econometric Society*, 555–574.

Li, B., J. Friedman, R. Olshen, and C. Stone (1984). Classification and regression trees (cart). *Biometrics 40*(3), 358–361.

Li, D.-Y., W. Xu, H. Zhao, and R.-Q. Chen (2009). A SVR based forecasting approach for real estate price prediction. Volume 2, pp. 970–974. IEEE.

Li, R. Y. M., S. Fong, and K. W. S. Chong (2017). Forecasting the REITs and stock indices: group method of data handling neural network approach. *Pacific Rim Property Research Journal 23*(2), 123–160. Publisher: Taylor & Francis.

Li, X., X. Zhang, and J. Li (2014). The relationship between twitter sentiment and stock market volatility. *Journal of Information & Computational Science 11*(13), 4773–4781.

Li, Y., Z. Xiang, and T. Xiong (2020). The behavioral mechanism and forecasting of Beijing housing prices from a multiscale perspective. *Discrete Dynamics in Nature and Society 2020*. Publisher: Hindawi.

Lim, W. T., L. Wang, Y. Wang, and Q. Chang (2016). Housing price prediction using neural networks. In *2016 12th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD)*, pp. 518–522. IEEE.

Liu, L. and L. Wu (2020). Predicting housing prices in China based on modified Holt's exponential smoothing incorporating whale optimization algorithm. *Socio-Economic Planning Sciences 72*, 100916. Publisher: Elsevier.

Liu, X., X. Zhang, and Q. Yang (2023). Ensemble learning techniques combining ml models for forecasting home builders stock. *Journal of Housing Research 30*(1), 125–142.

Loo, W. K. (2020). Predictability of hk-reits returns using artificial neural network. *Journal of Property Investment & Finance 38*(4), 291–307.

Loo, W. K., M. Ahmad Anuar, and S. Ramakrishnan (2016). Modeling the volatility of asian reit markets. *Pacific Rim Property Research Journal 22*(3), 231–243.

Luo, C., Z. Wang, L. Cao, X. Liu, and C. Chen (2020). Twitter data to forecast us stock home builders. *Journal of Forecasting 39*(7), 1010–1026.

Milunovich, G. (2020). Forecasting Australia's real house price index: A comparison of time series and machine learning methods. *Journal of Forecasting 39*(7), 1098–1118. Publisher: Wiley Online Library.

Mincinci, E., D. Smith, and M. Johnson (2018). Influences of macroeconomic factors on us homebuilders. *Journal of Real Estate Economics 30*(2), 75–92.

Park, B. and J. K. Bae (2015). Using machine learning algorithms for housing price prediction: The case of Fairfax County, Virginia housing data. *Expert systems with applications 42*(6), 2928–2934. Publisher: Elsevier.

Pati, Y. C., R. Rezaiifar, and P. S. Krishnaprasad (1993). Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar conference on signals, systems and computers*, pp. 40–44. IEEE.

Pavlova, I., J. H. Cho, A. Parhizgari, and W. G. Hardin III (2014). Long memory in reit volatility and changes in the unconditional mean: a modified figarch approach. *Journal of Property Research 31*(4), 315–332.

Phan, T. D. (2018, December). Housing Price Prediction Using Machine Learning Algorithms: The Case of Melbourne City, Australia. In *2018 International Conference on Machine Learning and Data Engineering (iCMLDE)*, pp. 35–42.

Phillips, P. C. and P. Perron (1988). Testing for a unit root in time series regression. *Biometrika 75*(2), 335–346. Publisher: Oxford University Press.

Piao, Y., A. Chen, and Z. Shang (2019). Housing price prediction based on cnn. In *2019 9th international conference on information science and technology (ICIST)*, pp. 491–495. IEEE.

Plakandaras, V., R. Gupta, P. Gogas, and T. Papadimitriou (2015). Forecasting the US real house price index. *Economic Modelling 45*, 259–267. Publisher: Elsevier.

Posner, M. I. (1989). *Foundations of cognitive science*. MIT press Cambridge, MA.

Rafiei, M. H. and H. Adeli (2016). A novel machine learning model for estimation of sale prices of real estate units. *Journal of Construction Engineering and Management 142*(2), 04015066. Publisher: American Society of Civil Engineers.

Rao, S. N. V. B., V. Yellapragada, K. Padma, D. J. Pradeep, C. P. Reddy, M. Amir, and

S. S. Refaat (2022). Day-ahead load demand forecasting in urban community cluster microgrids using machine learning methods. *Energies*.

Rao, T. and S. Srivastava (2014). Twitter sentiment analysis: How to hedge your bets in the stock markets. In *State of the art applications of social network analysis*, pp. 227–247. Springer.

Rao, T., S. Srivastava, et al. (2012). Analyzing stock market movements using twitter sentiment analysis.

Rapach, D. E. and J. K. Strauss (2007). Forecasting real housing price growth in the eighth district states. *Federal Reserve Bank of St. Louis. Regional Economic Development 3*(2), 33–42.

Rapach, D. E. and J. K. Strauss (2009). Differences in housing price forecastability across US states. *International Journal of Forecasting 25*(2), 351–372. Publisher: Elsevier.

Roberts, H. (1967). Statistical versus clinical prediction of the stock market. *Unpublished manuscript 252*.

Said, S. E. and D. A. Dickey (1984). Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika 71*(3), 599–607. Publisher: Oxford University Press.

Sarip, A. G., M. B. Hafez, and M. N. Daud (2016). Application of fuzzy regression model for real estate price prediction. *Malaysian Journal of Computer Science 29*(1), 15–27.

Serrano, C. and M. Hoesli (2007). Forecasting ereit returns. *European Economics: Macroeconomics & Monetary Economics eJournal*.

Shahhosseini, M., G. Hu, and H. Pham (2019). Optimizing ensemble weights for machine learning models: A case study for housing price prediction. pp. 87–97. Springer.

Sheng, Q., J. Smith, and A. Johnson (2019). Predicting homebuilders' stock returns: A machine learning approach. *Journal of Financial Analytics 25*(3), 45–60.

Staff, A. (2022, March). U.S. Foreclosure Activity In February 2022 Continues To Increase Steadily.

Taffese, W. Z. (2007). Case-based reasoning and neural networks for real estate valuation. In *Artificial Intelligence and Applications*, pp. 98–104.

Tetlock, P. C. (2008). Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance 63*(3), 1139–1168.

Vara-Miguel, A. (2020). Cross-national similarities and differences between legacy and digital-born news media audiences. *Media and Communication 8*(2), 16–27.

Vargas-Silva, C. (2008). Monetary policy and the US housing market: A VAR analysis imposing sign restrictions. *Journal of Macroeconomics 30*(3), 977–990. Publisher: Elsevier.

Vu, T. T., S. Chang, Q. T. Ha, and N. Collier (2012). An experiment in integrating sentiment features for tech stock prediction in twitter. In *Proceedings of the workshop on information extraction and entity analytics on social media data*, pp. 23–38.

Wang, H.-C., M.-C. Wu, C.-C. Chen, and I.-J. Chang-Chien (2011). Forecasting the reits' returns in us: New evidence from structural time series model. In *2011 IEEE International Summer Conference of Asia Pacific Business Innovation and Technology Management*, pp. 205–209. IEEE.

Wang, L., Y. Liu, and W. Chen (2021). Exploring deep learning techniques for forecasting home builders stock. *Journal of Real Estate Economics 40*(3), 415–432.

Wang, X., J. Wen, Y. Zhang, and Y. Wang (2014). Real estate price forecasting based on SVM optimized by PSO. *Optik 125*(3), 1439–1443. Publisher: Elsevier.

Wei, Y. and Y. Cao (2017). Forecasting house prices using dynamic model averaging approach: Evidence from China. *Economic Modelling 61*, 147–155. Publisher: Elsevier.

Weinberg, D. H. (2014). Data sources for US housing research, part 1: Public sector data sources. *Cityscape 16*(3), 131–148. Publisher: JSTOR.

Weinberg, D. H. (2015). Data sources for US housing research, part 2: private sources, administrative records, and future directions. *Cityscape 17*(1), 191–206. Publisher: JSTOR.

Wilson, I. D., S. D. Paris, J. A. Ware, and D. H. Jenkins (2002). Residential property price time series forecasting with neural networks. In *Applications and Innovations in Intelligent Systems IX*, pp. 17–28. Springer.

Wu, C.-H., C.-H. Li, I.-C. Fang, C.-C. Hsu, W.-T. Lin, and C.-H. Wu (2009). Hybrid genetic-based support vector regression with feng shui theory for appraising real estate price. pp. 295–300. IEEE.

Xue, Y., X. Li, and L. Zhang (2021). Investigating the use of sentiment analysis in financial markets. *International Journal of Finance & Economics 46*(2), 319–335.

Yasnitsky, L. N., V. L. Yasnitsky, and A. O. Alekseev (2021). The complex neural network model for mass appraisal and scenario forecasting of the urban real estate market value that adapts itself to space and time. *Complexity 2021*. Publisher: Hindawi.

Yuan, X., J. Yuan, T. Jiang, and Q. U. Ain (2020). Integrated long-term stock selection models based on feature selection and machine learning algorithms for china stock market. *IEEE Access 8*, 22672–22685.

Zhang, J., R. de Jong, and D. Haurin (2013). Are Real House Prices Stationary? Evidence from New Panel and Univariate Data. Technical report, working paper.

Zhang, L. (2013). *Sentiment analysis on Twitter with stock price and significant keyword correlation*. Ph. D. thesis.

Zhang, X., H. Fuehres, and P. A. Gloor (2011). Predicting stock market indicators through twitter "i hope it is not as bad as i fear". *Procedia-Social and Behavioral Sciences 26*, 55–62.

Zhou, J. (2017). Forecasting reit volatility with high-frequency data: a comparison of alternative methods. *Applied Economics 49*(26), 2590–2605.

Zhou, J. and Z. Kang (2011). A comparison of alternative forecast models of reit volatility. *The Journal of Real Estate Finance and Economics 42*, 275–294.

Zivot, E. and D. W. K. Andrews (2002). Further evidence on the great crash, the oil-price shock, and the unit-root hypothesis. *Journal of business & economic statistics 20*(1), 25–44. Publisher: Taylor & Francis.

Appendix A

Appendix to Chapter 1

A.1 Robustness Check

The FHFA house price index was utilized spanning from April 2014 to April 2024 to conduct a comprehensive analysis. In contrast to the data from Case-Shiller and Zillow, the FHFA index demonstrated stationarity, which means it was appropriate for classical statistical forecasting models in this forecast horizon.

The robustness check reproduced the original experiment by generating projections for 3-step and 12-step ahead. The outcomes over the 12-step timeframe were stable. Machine learning models consistently shown superior performance compared to traditional models, indicating their efficacy in long-term forecasting. However, a notable change became apparent in the projection for three steps forward. Although machine learning models had previously shown a short-term advantage, none of them surpassed classical models in the current experiment.

This difference can be attributable to the varying timeframes of the forecasts. The initial experiment, which involved predicting the final three periods of 2022, occurred simultaneously with the effects of the COVID-19 pandemic. Machine learning methods, with their flexibility, could have better represented this distinctive economic circumstance. On the other hand, the FHFA data from April 2014 to April 2024 is likely to reflect a period of economic stability, during which traditional models perform satisfactorily.

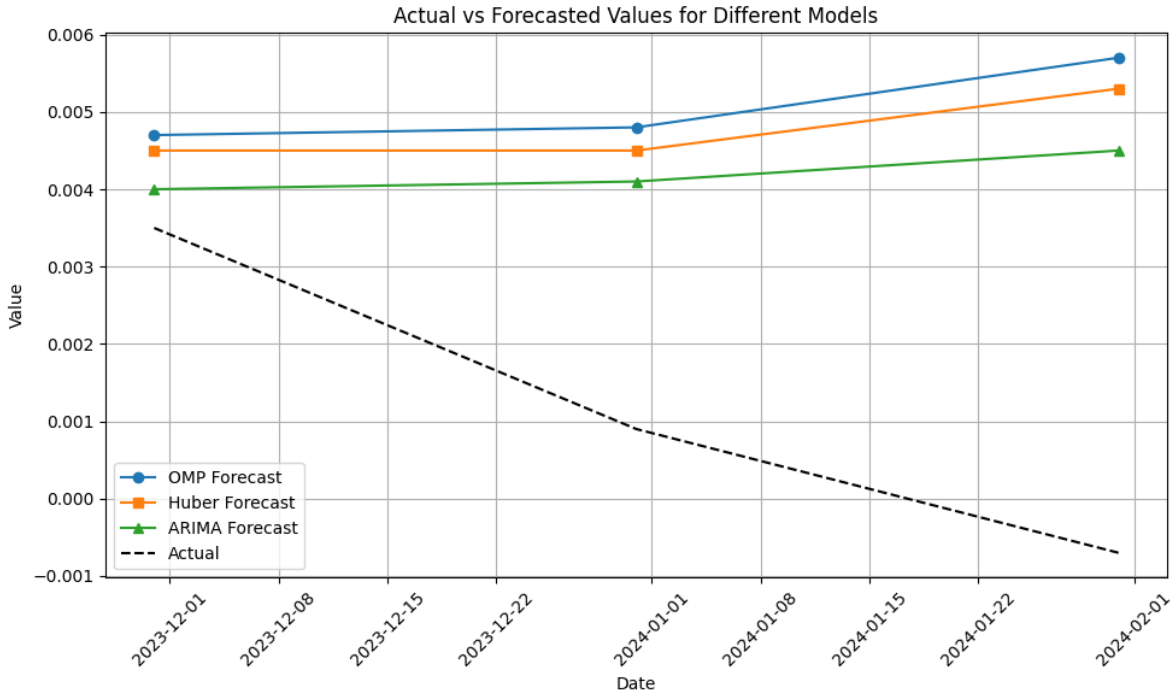
Additional research could explore the precise attributes of housing market data that impact the most suitable selection of forecasting models for short-term timeframes.

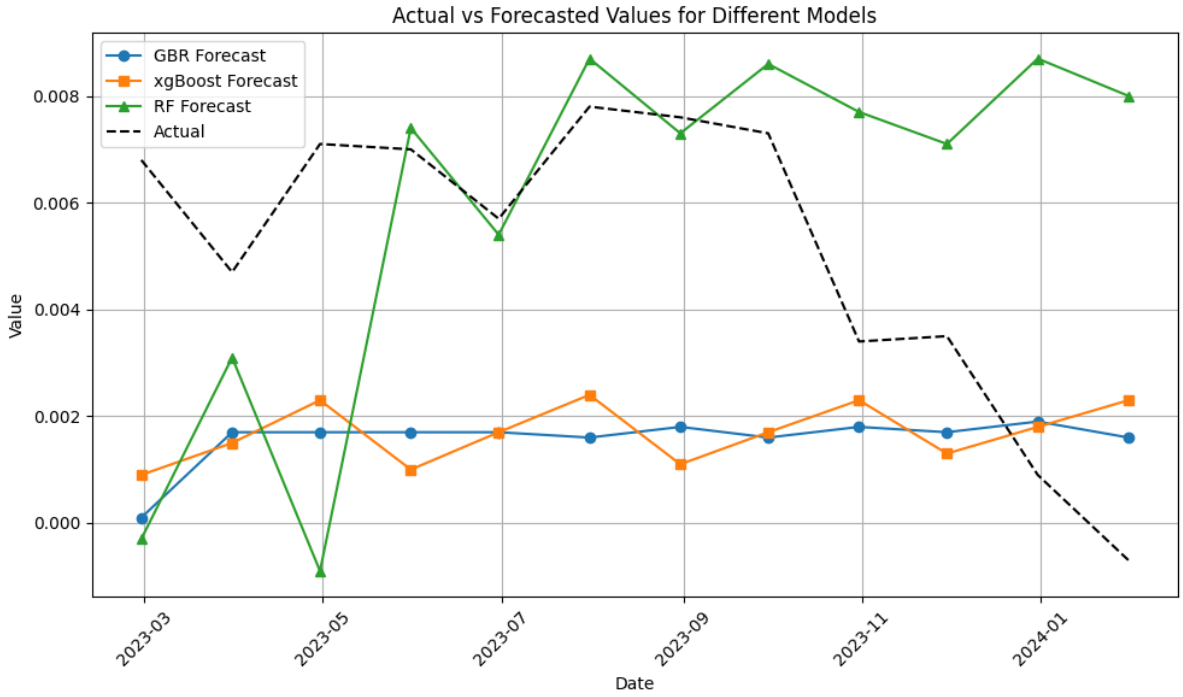
Table A.1.1: FHFA 3-step ahead rolling forecast result (% gain over random walk values)

Model	RMSE	MAPE	SMAPE
arima	44	70.85	58.33
huber_cds_dt	44	58.80	55.97
omp_cds_dt	36	50.42	49.55
lr_cds_dt	36	50.42	49.55
br_cds_dt	36	49.19	49.70
theta	28	67.25	41.05
naive	24	61.60	28.90
ets	24	64.85	35.22
exp_smooth	24	64.54	34.88
auto_arima	24	64.72	35.60

Table A.1.2: FHFA 12-step ahead rolling forecast result (% gain over random walk)

Model	RMSE	MAPE	SMAPE
rf_cds_dt	21.33	20.95	8.97
gbr_cds_dt	20	23.64	5.67
xgboost_cds_dt	17.33	5.88	8.83
et_cds_dt	13.33	2.92	9.49
naive	10.67	-9.78	14.32
theta	9.33	-8.00	8.96
dt_cds_dt	10.67	-2.55	7.99
ada_cds_dt	10.67	4.03	5
ets	8	-13.05	8.74
exp_smooth	5.33	-12.13	5.42





A.2 Models Used

- Naive Model : Considers what occurred in the preceding period and forecasts that it will occur again. Also known as random walk forecast.

- Classical models (sktime library)

- Grand Means Forecaster: Forecast of all future values are equal to the average of historical data.

- Seasonal Naive Forecaster: The forecasted value is equal to last observed value from the same season (i.e. same month of previous years) and useful when training high seasonal data.

- Polynomial Trend Forecaster: Forecast the series with a polynomial trend. The model runs a regression to extract polynomial features and transforms the time index according to polynomial features extracted to account for polynomial degree and intercept. The model is then trained after transformation to get forecasted result.

- ARIMA: Also known as Autoregressive integrated moving average model. In an Autoregressive model the forecasts correspond to a linear combination of past values of the variable. In a Moving Average model, the forecasts correspond to a linear combination of past forecast errors. Basically, the ARIMA models combine these two approaches. Since they require the time series to be stationary, differencing (Integrating) the time series may be a necessary step, i.e. considering the time series of the differences instead of the original one. An auto arima model is run to check for best p, q, d according to AIC and BIC and give forecasted result for ARIMA model with best parameters collected.

- Auto ARIMA: A SARIMAX model which accounts for multiple seasonal periods and exogenous factors missing in ARIMA model. The SARIMAX model (Seasonal ARIMA) extends the ARIMA by adding a linear combination of seasonal past values and/or forecast errors.

- Exponential Smoothening: Also known as Holt Winter or triple smoothening forecaster. The method involves three smoothening equations to update level, trend and seasonal component of time series. The smoothening equations use exponential weighting to give more weight to recent observations and less weight to old observations.

- ETS: A model equivalent to Holt-Winter smoothing method with capability to find optimal parameters selected based on goodness of fit criteria such as AIC & BIC.

- BATS (Box-Cox transformation, ARMA errors, Trend and Seasonal component): Extension of Holt-Winters method that incorporates Box-Cox transformation and ARMA error modeling to improve forecast.

- TBATS (Trigonometric Seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal component): An extension of BATS model with capability of modeling multiple seasonal patterns.

- Theta forecaster: Theta Forecaster (Assimakopoulos and Nikolopoulos (2000)) Involves fitting two θ lines, forecasting the lines using a Simple Exponential Smoother, and then combining the forecasts from the two lines to produce final forecast. It's similar to simple exponential smoothening with drift parameter. Also known as additive theta method it uses the idea of theta parameter to model the trend component and seasonal component separately.

- Croston Forecaster: A type of exponential smoothening model designed for intermittent demand forecasting. The method involves two smoothening equations to update the level and interval components of the time series. The model also uses exponential weighting to give more weight to recent observation. For more information on this model, you can check the original paper. (Croston (1972))

- Machine Learning Models (scikit-learn library)

- Linear Regression: Also known as OLS, fits a linear model to minimize residual sum of squares between observed and predicted.

- Least Angular regression: The LARS algorithm iteratively adds features to the regression equation that have the highest absolute correlation with the response variable, while simultaneously decreasing the coefficients of the other features towards zero. At each stage, the algorithm determines the direction that is most

correlated with the response variable and then shifts the coefficients in this direction until another variable becomes similarly correlated. Efron et al. (2004)

- Ridge regression: Addresses some problems of OLS by imposing L2 regularization. The ridge regression has hyperparameter alpha which accounts for shrinkage. The greater the shrinkage parameter, the more robustness towards collinearity for coefficients.

- Elastic Net Regression: Combination of L1 and L2 regularization giving best of ridge and lasso models.

- LASSO regression: Model with L1 regularization, technically it optimizes the same objective function as elastic net but with L2=0.

- Bayesian Ridge regression: The Bayesian Ridge algorithm is a regression algorithm that estimates regression coefficients using Bayesian statistics. Based on a Bayesian linear regression model that imposes a prior distribution on the regression coefficients, this method is utilized. The prior distribution is a zero-mean normal distribution with an alpha-dependent variance. The algorithm then calculates the posterior distribution of the regression coefficients, which is also a normal distribution who's mean and variance depend on the prior distribution and the observed data.

- LASSO LAR: LASSO model implemented using LARS algorithm. This yields the exact solution, which is piecewise linear as a function of the norm of its coefficients, unlike the implementation based on coordinate descent.

- Huber regression: The Huber algorithm is a robust regression technique that incorporates the most advantageous characteristics of the mean squared error (MSE) and the mean absolute error (MAE) loss functions. It is commonly used in machine learning and statistical modeling to deal with anomalies and high-variance data. The Huber algorithm is a gradient descent optimization technique that minimizes the Huber loss function to determine the regression

model's optimal coefficients. Updating the coefficients iteratively using the gradient of the Huber loss function until convergence is reached.

- Passive Aggressive Regression: Passive-Aggressive Regression is a type of algorithm used for regression analysis, where the goal is to predict a continuous numerical output variable based on a set of input features. The algorithm belongs to the family of online learning algorithms, meaning that it can adapt to new data as it arrives and can be updated incrementally.

  The Passive-Aggressive algorithm is called "passive" because it tries to avoid making mistakes, and "aggressive" because it makes corrections when it does make mistakes. The algorithm works by minimizing the loss function, which is the difference between the predicted output and the true output. The algorithm updates the weights of the model by applying a penalty term to the weight update proportional to the size of the error, which can be seen as the "aggressive" component.

- Orthogonal Matching Pursuit: Orthogonal Matching Pursuit (OMP) is an algorithm for finding a sparse solution to an underdetermined linear system of equations when solving linear regression problems with a sparsity constraint. Pati et al. (1993)

- Random Forest: Various subsets of data are used to train a significant number of decision trees in RF. Using a random subset of the features and a subset of the training data, each decision tree is constructed. During training, decision trees learn to predict the objective variable given the input feature values. After all the trees have been trained, their predictions are combined to produce a final forecast. The final prediction is typically the mean or median of all the trees' predictions. Breiman (2001)

- Decision Tree: Decision tree regression is an algorithm for supervised learning that predicts continuous numerical values. It entails segmenting the input space into regions using a set of decision principles inferred from the training data. A

decision tree is constructed by recursively dividing the input space into smaller regions, with each division based on a feature value and a threshold. At each tree node, the characteristic with the highest information gain is selected as the division criterion. The procedure is repeated until a stopping criterion is met, such as when the maximum depth of the tree is attained or when the number of samples in a leaf node falls below a predetermined threshold. Li et al. (1984)

- Extra Tree: Extra Trees Regression, also known as Extremely Randomized Trees Regression or Extra-Trees Regression, is a regression-specific ensemble learning method. It's an extension of the Random Forest algorithm and a member of the family of decision tree-based methods. Multiple decision trees are trained on various random subsets of the training data in Extra Trees Regression, and at each node, a random subset of features is considered for division. Extra Trees Regression randomizes the splitting process by selecting the dividing point at random, as opposed to selecting the optimal split based on impurity reduction.

  This additional randomization reduces the model's variance and makes it less sensitive to data noise and outliers. In addition, Extra Trees Regression requires fewer computational resources and can be trained more quickly than Random Forest. The final prediction is determined by averaging the predictions of all the ensemble trees. Extra Trees Regression hyperparameters include the number of trees in the ensemble, the number of features to consider at each split, and the dividing criteria. Geurts et al. (2006)

- Gradient Boosting: Gradient Boosting is a well-known machine learning technique that combines multiple feeble learners to produce a more accurate model. In Gradient Boosting Regression, specifically, a set of decision trees are trained sequentially, with each new tree attempting to remedy the errors of the preceding tree. Gradient Boosting Regression is based on the concept of fitting a model y = F(X) to a set of training data by minimizing the loss function L (y, F(X)). At each iteration, a new decision tree is trained to minimize the loss function

gradient with respect to the present model, and then added to the ensemble. The final prediction is the sum of all the predictions of the ensemble's decision trees. Friedman (2001)

- AdaBoost: Commonly used boosting algorithm in machine learning for regression assignments. AdaBoost is based on the concept of combining several weak regressors (regressors that are only marginally better than random guesswork) to create a strong regressor. To use AdaBoost for regression, we must slightly modify the algorithm. Instead of using weighted voting to make a binary classification decision, we predict the output value for a given input using weighted averaging. At each iteration, a weak regressor is fitted to the training set and the weighted error between the predicted and actual values is computed. The weights of the training examples are then adjusted based on the error so that the next regressor concentrates on the examples that were not accurately predicted by the preceding regressor. This procedure is repeated a predetermined number of times or until the error rate reaches an acceptable level.

- Light Gradient Boosting: Light Gradient Boosting Machine (LightGBM) is a framework for gradient boosting that employs tree-based learning algorithms. It is designed to be memory and computation time efficient, while achieving high predictive accuracy. LightGBM employs the innovative Gradient-based One-Side Sampling (GOSS) technique to reduce the number of instances used for gradient calculation. GOSS chooses instances based on their gradients and eliminates instances with minor gradients, resulting in a more efficient and accurate gradient calculation. LightGBM also employs a histogram-based approach to construct the decision trees, which increases the algorithm's efficacy. LightGBM constructs histograms of feature values and uses them to determine the optimal split sites, as opposed to the conventional method of dividing nodes based on feature values. Ke et al. (2017)

- KNN regression: KNN regression, also referred to as k-nearest neighbors'

regression, is a non-parametric regression algorithm. The value of a new observation is predicted by averaging the values of its k adjacent companions in the training set. In KNN regression, the algorithm identifies the k nearest neighbors of a new observation using a distance metric (such as the Euclidean distance). The target value of the new observation is then predicted by calculating the average of the target values of the neighboring observations. Cover and Hart (1967)

A.3 Model Evaluation Metrics

I) RMSE: Root Mean Square Error

RMSE metric calculates the standard deviation of residuals (predicted errors)

$$RMSE = \sqrt{\left(\frac{\sum_{i=1}^{n}(p_i - a_i)}{N}\right)^2}$$

where N is the total number of observations, p_i is the predicted value and a_i is the actual value.

where e_j is forecast error of given period and a_t is actual observation of time series.

II) R2: R squared

R2 can be interpreted as the "proportion of dependent variable variance explained". As opposed to how well a time series model matches historical values, we frequently judge it on how well it forecasts future values. Yet, R2 favors the latter over the former.

$$R2 = 1 - \left(\frac{SS_{residual}}{SS_{total}}\right)$$

III) sMAPE: symmetric Mean Absolute Percentage Error

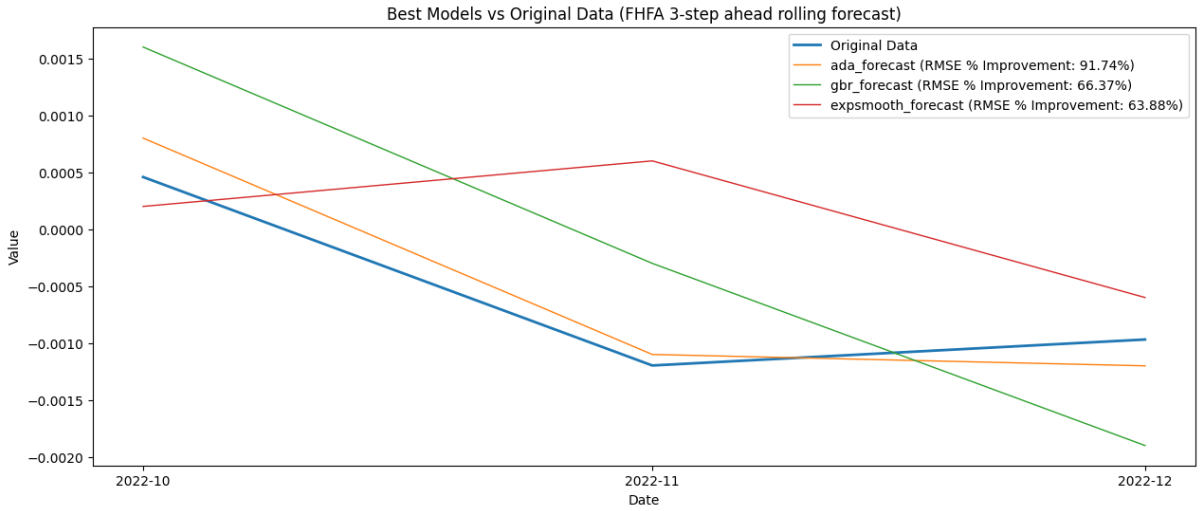
A measure of accuracy based on a percentage (or relative) mistake is called the symmetric mean absolute percentage error (SMAPE). The absolute error divided

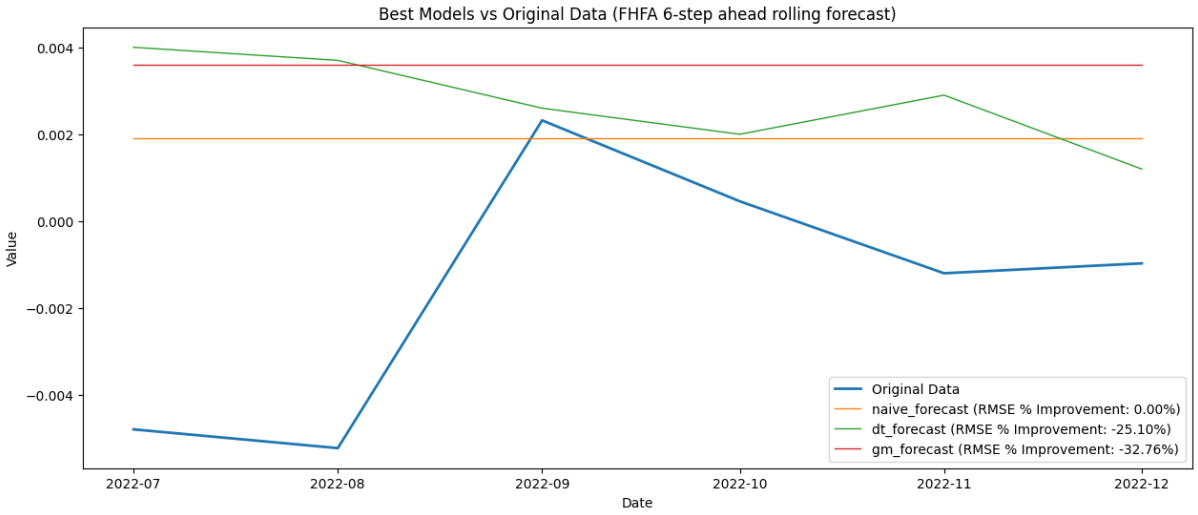
by the magnitude of the precise value yields the relative error. SMAPE has a lower bound and an upper bound, unlike the mean absolute percentage error, which only has an upper bound.

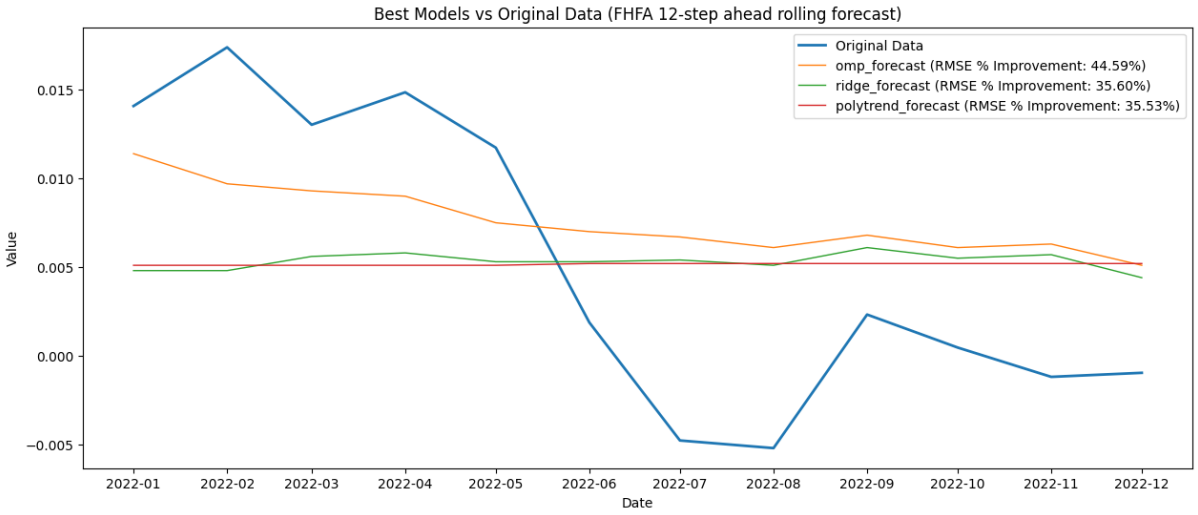
$$sMAPE = 100 \times \left[\frac{1}{n} \sum_{t=1}^{n} \frac{|p_i - a_i|}{(|a_i| + |p_i|)/2} \right]$$

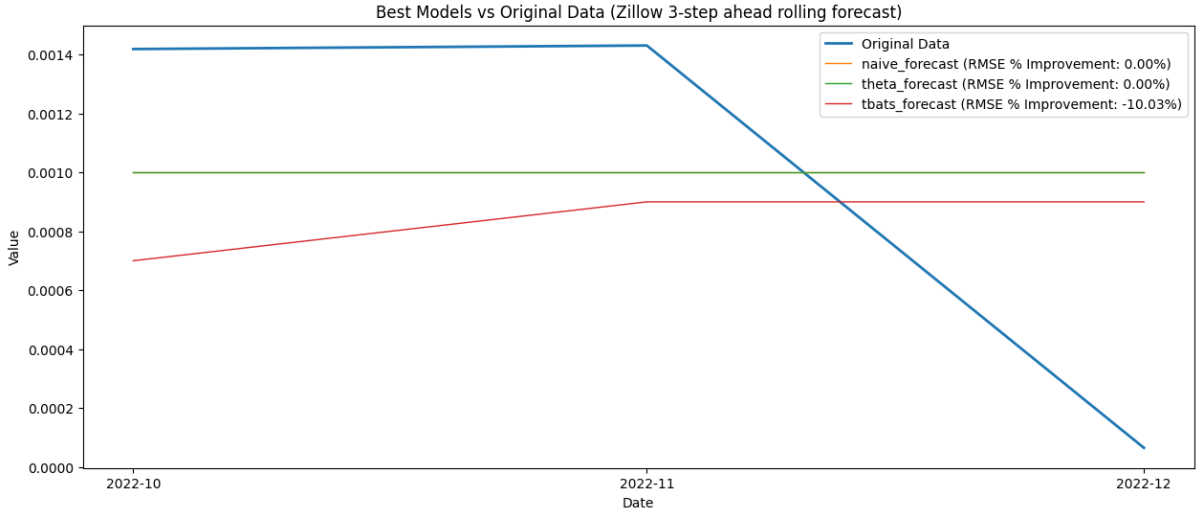
where n is the number of observations and a_i and p_i are actual and predicted values from the forecast.

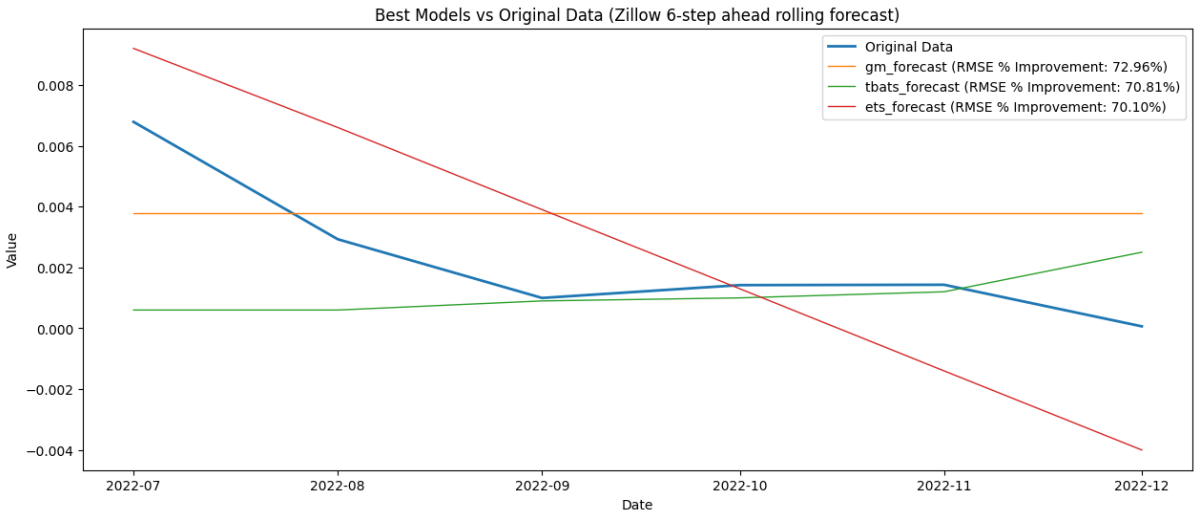
A.3.1 Graphs

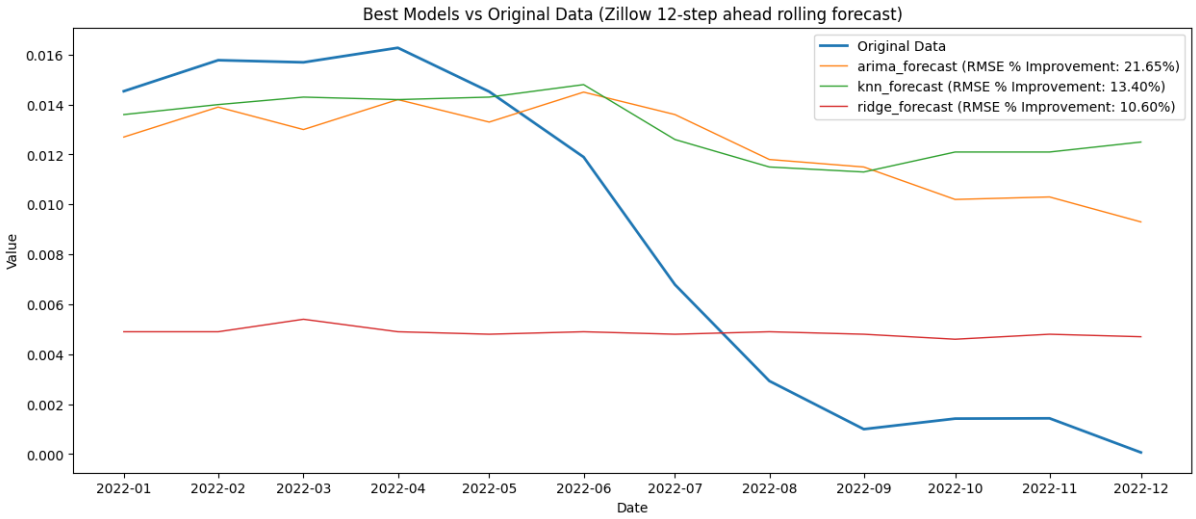


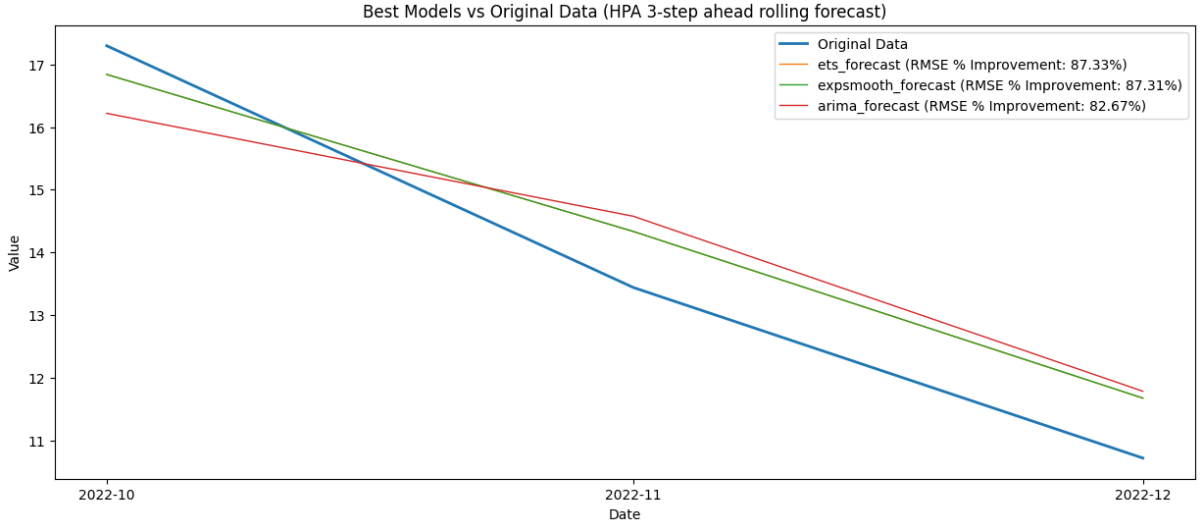


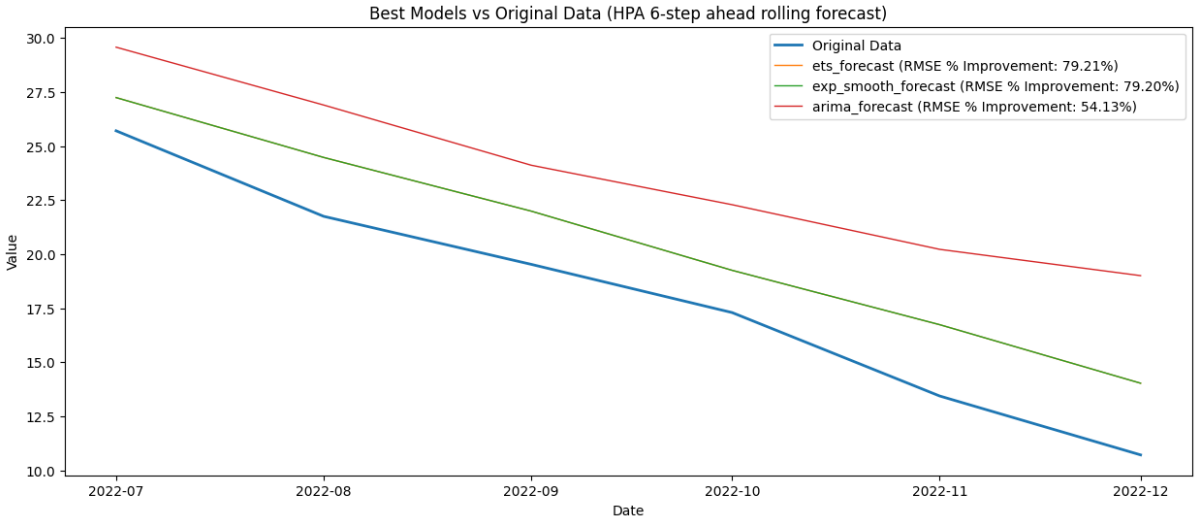

Best Models vs Original Data (FHFA 12-step ahead rolling forecast)
Original Data
omp_forecast (RMSE % Improvement: 44.59%)
ridge_forecast (RMSE % Improvement: 35.60%)
polytrend_forecast (RMSE % Improvement: 35.53%)
Value
Date
0.015
0.010
0.005
0.000
−0.005
2022-01
2022-02
2022-03
2022-04
2022-05
2022-06
2022-07
2022-08
2022-09
2022-10
2022-11
2022-12

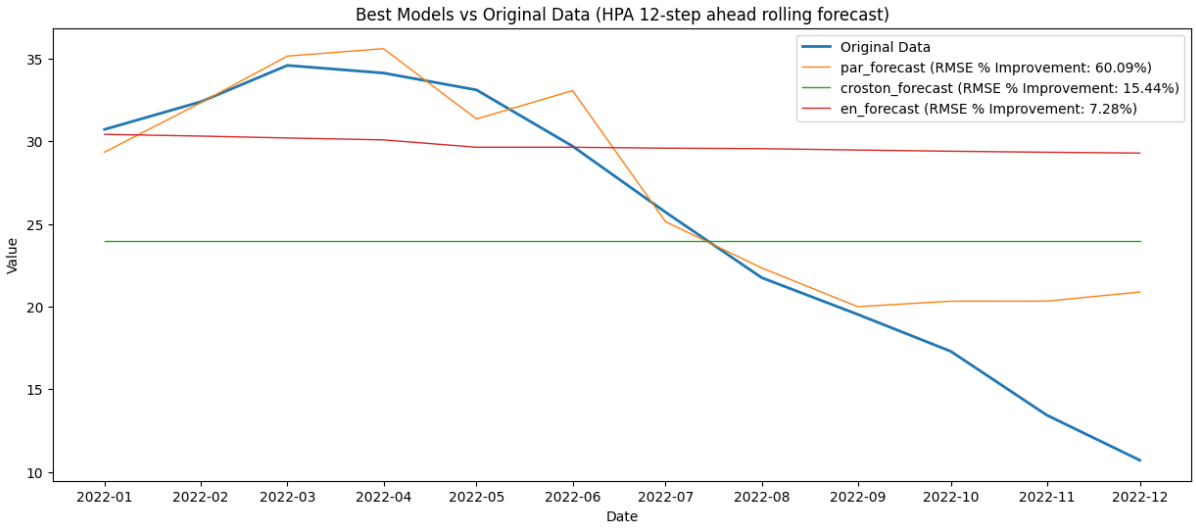

Best Models vs Original Data (Zillow 3-step ahead rolling forecast)
Original Data
naive_forecast (RMSE % Improvement: 0.00%)
theta_forecast (RMSE % Improvement: 0.00%)
tbats_forecast (RMSE % Improvement: -10.03%)
Value
Date
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
2022-10
2022-11
2022-12


Best Models vs Original Data (Zillow 6-step ahead rolling forecast)
Original Data
gm_forecast (RMSE % Improvement: 72.96%)
tbats_forecast (RMSE % Improvement: 70.81%)
ets_forecast (RMSE % Improvement: 70.10%)
Value
Date
0.008
0.006
0.004
0.002
0.000
−0.002
−0.004
2022-07
2022-08
2022-09
2022-10
2022-11
2022-12


Best Models vs Original Data (Zillow 12-step ahead rolling forecast)
Original Data
arima_forecast (RMSE % Improvement: 21.65%)
knn_forecast (RMSE % Improvement: 13.40%)
ridge_forecast (RMSE % Improvement: 10.60%)
Value
Date


Best Models vs Original Data (HPA 3-step ahead rolling forecast)
Original Data
ets_forecast (RMSE % Improvement: 87.33%)
expsmooth_forecast (RMSE % Improvement: 87.31%)
arima_forecast (RMSE % Improvement: 82.67%)
Value
Date


Best Models vs Original Data (HPA 6-step ahead rolling forecast)
Original Data
ets_forecast (RMSE % Improvement: 79.21%)
exp_smooth_forecast (RMSE % Improvement: 79.20%)
arima_forecast (RMSE % Improvement: 54.13%)
Value
Date



A.4 Results table

FHFA 3-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
ada_forecast	91.73	84.24	89.67
gbr_forecast	66.36	44.57	59.26
expsmooth_forecast	63.88	38.99	76.29
rf_forecast	59.49	41.97	63.09
ets_forecast	50.15	14.84	63.46
et_forecast	46.03	9.12	43.00
theta_forecast	38.44	9.59	47.11
tbats_forecast	22.21	9.59	35.87
bats_forecast	22.21	9.59	35.87
Auto-arima_forecast	-1.33	0	-0.80
dt_forecast	-3.44	46.16	-19.82
omp_forecast	-26.95	-1.22	-23.07
huber_forecast	-30.07	-2.22	-26.08
lr_forecast	-31.55	-3.06	-34.00
lars_forecast	-32.57	-3.06	-35.00
Bridge_forecast	-37.30	-3.31	-39.73
gm_forecast	-42.95	-3.98	-51.04
knn_forecast	-43.65	-1.91	-38.35
lightgbr_forecast	-49.09	-2.52	-45.01
par_forecast	-97.87	-6.59	-116.14
en_forecast	-98.95	-6.59	-117.14
lasso_forecast	-98.95	-6.59	-117.14
llars_forecast	-98.95	-6.59	-117.14
ridge_forecast	-102.18	-6.78	-122.18
polytrend_forecast	-103.09	-6.59	-121.71
croston_forecast	-230.83	-8.89	-270.91
arima_forecast	-296.44	-8.24	-296.88
snaive_forecast	-394.31	-9.92	-447.81

FHFA 6-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
dt_forecast	-25.09	0.56	-9.06
gm_forecast	-32.75	-5.93	-68.20
omp_forecast	-36.71	-9.12	-100.36
en_forecast	-74.14	-12.78	-152.91
lasso_forecast	-74.14	-12.78	-152.91
llars_forecast	-74.14	-12.78	-152.91
par_forecast	-74.14	-12.78	-152.91
polytrend_forecast	-74.79	-11.71	-152.98
ridge_forecast	-81.30	-13.31	-165.12
gbr_forecast	-101.51	-15.39	-193.96
lightgbm_forecast	-116.46	-16.40	-241.87
rf_forecast	-128.65	-16.63	-240.73
ada_forecast	-134.39	-16.41	-293.88
expsmooth_forecast	-147.90	-17.41	-290.95
huber_forecast	-163.41	-17.94	-348.77
theta_forecast	-168.45	-17.42	-332.48
tbats_forecast	-168.45	-17.42	-332.48
bats_forecast	-168.45	-17.42	-332.48
Auto-Arima_forecast	-169.30	-17.67	-344.72
lars_forecast	-172.38	-18.08	-353.35
ets_forecast	-172.97	-18.03	-342.39
lr_forecast	-173.59	-18.12	-358.27
Bridge_forecast	-176.14	-18.23	-360.15
et_forecast	-183.31	-18.56	-381.51
arima_forecast	-185.21	-18.46	-442.35
knn_forecast	-198.22	-18.41	-416.53
croston_forecast	-213.10	-18.86	-416.37
snaive_forecast	-229.06	-19.23	-477.70

FHFA 12-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
omp_forecast	44.58	-3.29	56.26
ridge_forecast	35.59	-19.56	59.65
polytrend_forecast	35.53	-19.26	60.44
par_forecast	34.84	-20.86	60.22
en_forecast	34.76	-20.86	60.20
lasso_forecast	34.76	-20.86	60.20
llar_forecast	34.76	-20.86	60.20
gma_forecast	34.28	-22.97	71.00
lr_forecast	18.02	0.99	21.48
lar_forecast	17.98	0.81	21.46
Bridge_forecast	17.83	0.85	21.30
huber_forecast	16.16	1.34	19.02
croston_forecast	14.52	-0.03	16.44
ada_forecast	12.84	1.05	17.30
snaive_forecast	12.33	-4.69	12.42
arima_forecast	8.65	-3.81	7.45
tbats_forecast	7.81	0.74	8.68
bats_forecast	7.81	0.74	8.68
theta_forecast	7.23	0.74	8.02
expsmooth_forecast	6.56	-0.83	8.07
ets_forecast	6.55	-0.03	7.32
dt_forecast	5.82	-3.18	2.26
knn_forecast	5.11	0.36	1.90
et_forecast	3.54	-0.17	-1.75
Auto-arima_forecast	3.44	-0.78	4.19
lightgbm_forecast	1.15	-0.08	0.94
rf_forecast	0.76	-2.07	-0.95
gbr_forecast	-9.29	-2.05	-11.92

Zillow 3-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
theta_forecast	0	0	0
tbats_forecast	-10.03	-16.59	8.40
omp_forecast	-10.16	6.01	-19.60
bats_forecast	-26.92	-15.06	-21.93
lightgbr_forecast	-54.25	-85.33	42.37
rf_forecast	-65.09	-80.84	21.84
dt_forecast	-73.13	-80.84	21.38
et_forecast	-75.24	-106.87	32.71
knn_forecast	-83.39	-31.90	-84.46
gbr_forecast	-98.99	6.25	-122.43
ada_forecast	-124.21	-6.32	-135.76
huber_forecast	-135.94	-143.96	-4.72
Bridge_forecast	-164.30	-143.96	-36.91
lr_forecast	-174.78	-143.96	-47.64
lar_forecast	-174.78	-143.96	-47.64
snaive_forecast	-190.21	-20.09	-200.00
Auto-arima_forecast	-228.56	-143.96	-111.56
gm_forecast	-351.64	-52.52	-305.54
expsmooth_forecast	-479.11	-143.96	-387.28
ets_forecast	-521.60	-143.96	-430.20
arima_forecast	-586.87	-143.96	-476.38
polytrend_forecast	-643.07	-77.08	-518.22
en_forecast	-644.67	-76.64	-528.03
lasso_forecast	-644.67	-76.64	-528.03
llar_forecast	-644.67	-76.64	-528.03
par_forecast	-650.92	-76.67	-538.29
ridge_forecast	-652.70	-76.24	-548.10
croston_forecast	-1433.78	-105.07	-1089.11

Zillow 6-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
gm_forecast	72.96	32.98	69.56
tbats_forecast	70.81	35.61	81.20
ets_forecast	70.10	25.95	67.31
expsmooth_forecast	69.51	24.70	65.08
polytrend_forecast	58.21	23.00	52.62
en_forecast	58.03	22.94	51.91
lasso_forecast	58.03	22.94	51.91
llars_forecast	58.03	22.94	51.91
par_forecast	57.55	22.92	50.45
ridge_forecast	57.35	23.12	49.73
lr_forecast	40.86	11.61	42.74
lars_forecast	40.86	11.61	42.74
Bridge_forecast	37.70	10.42	39.38
arima_forecast	35.31	8.64	42.48
Huber_forecast	32.34	8.73	32.84
snaive_forecast	6.15	2.20	0.39
omp_forecast	1.91	0.40	1.73
dt_forecast	0.64	1.34	-5.48
theta_forecast	0	0	0
rf_forecast	-1.62	0.02	-3.10
et_forecast	-3.26	-0.50	-6.09
lightgbm_forecast	-4.44	-0.78	-7.65
knn_forecast	-8.59	-1.84	-7.07
gbr_forecast	-8.72	-0.89	-14.73
ada_forecast	-13.07	-2.37	-12.10
croston_forecast	-15.85	-3.73	-13.82
Auto-arima_forecast	-435.89	-29.32	-372.51
bats_forecast	-550.24	-30.97	-475.21

Zillow 12-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
arima_forecast	21.64	2.32	30.19
knn_forecast	13.40	4.11	9.13
ridge_forecast	10.59	-26.24	65.21
huber_forecast	9.82	3.64	12.11
par_forecast	9.57	-27.34	63.90
en_forecast	9.47	-27.33	63.92
lasso_forecast	9.47	-27.33	63.92
llars_forecast	9.47	-27.33	63.92
polytrend_forecast	8.87	-28.35	63.77
croston_forecast	7.56	-1.28	8.15
snaive_forecast	5.67	-1.22	3.57
lars_forecast	4.96	3.17	7.63
omp_forecast	3.27	0.37	3.63
gm_forecast	0.34	-33.92	73.90
theta_forecast	0	0	0
Bridge_forecast	-1.19	2.14	0.44
lr_forecast	-2.93	1.85	-1.68
lightgbm_forecast	-9.89	-0.58	-10.91
ada_forecast	-10.14	0.54	-9.11
bats_forecast	-10.27	1.18	-9.65
rf_forecast	-12.59	0.18	-10.55
Auto-arima_forecast	-16.73	-1.01	-14.99
et_forecast	-24.75	-3.30	-29.95
dt_forecast	-34.55	-5.36	-18.07
gbr_forecast	-41.06	-8.55	-32.95
ets_forecast	-63.52	-9.95	-62.50
expsmooth_forecast	-70.52	-12.80	-66.65

HPA 3-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
ets_forecast	87.32	83.54	87.02
expsmooth_forecast	87.30	83.52	87.00
arima_forecast	82.67	77.60	82.44
gm_forecast	49.02	45.70	62.21
lr_forecast	39.69	36.85	41.94
lar_forecast	39.69	36.85	41.94
Bridge_forecast	39.07	36.28	41.32
ridge_forecast	37.51	34.83	39.75
huber_forecast	36.46	33.78	38.64
Auto-arima_forecast	29.81	27.84	32.01
bats_forecast	6.08	5.95	6.87
lightgbm_forecast	-1.13	2.74	1.27
rf_forecast	-1.43	-1.43	-1.66
dt_forecast	-1.87	-1.38	-1.81
gbr_forecast	-2.25	-1.73	-2.26
omp_forecast	-5.40	-4.20	-5.46
snaive_forecast	-6.01	-9.95	-9.14
theta_forecast	-6.33	-5.03	-6.49
en_forecast	-8.28	-6.44	-8.40
lasso_forecast	-8.36	-6.50	-8.48
llar_forecast	-8.36	-6.50	-8.48
et_forecast	-11.21	-9.17	-11.71
par_forecast	-11.83	-9.18	-12.08
ada_forecast	-16.43	-13.40	-17.18
knn_forecast	-16.68	-14.87	-18.23
polytrend_forecast	-46.60	-38.87	-50.27
croston_forecast	-102.74	-77.57	-109.34

HPA 6-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
ets_forecast	79.21	71.23	79.34
exp_smooth$_{f}orecast$	79.20	71.22	79.34
arima_forecast	54.12	42.50	53.81
polytrend_forecast	46.19	41.90	49.17
lars_forecast	42.33	34.49	42.95
lr_forecast	40.27	33.27	41.02
Bridge_forecast	38.90	32.08	39.66
gm_forecast	36.55	21.13	59.13
ridge_forecast	36.04	29.59	36.78
huber_forecast	34.09	27.88	34.82
Auto-arima_forecast	26.79	28.12	31.83
bats_forecast	14.45	17.58	18.59
croston_forecast	12.79	11.13	13.63
lasso_forecast	4.20	2.60	4.15
llar_forecast	4.20	2.60	4.15
en_forecast	1.90	0.81	1.78
snaive_forecast	-1.99	-3.47	-2.79
omp_forecast	-3.43	-2.44	-3.46
theta_forecast	-6.36	-3.92	-6.26
lightgbm_forecast	-17.26	-11.26	-17.38
et_forecast	-22.83	-14.32	-22.60
gbr_forecast	-22.89	-16.11	-23.15
rf_forecast	-24.41	-17.10	-24.72
dt_forecast	-26.24	-17.58	-26.42
ada_forecast	-27.37	-20.09	-27.86
knn_forecast	-29.02	-20.45	-29.39
par_forecast	-30.89	-20.35	-30.18

HPA 12-step ahead rolling forecast results (% improvement)

Model	***RMSE***	***sMAPE***	***MAPE***
par_forecast	60.09	57.44	63.59
croston_forecast	15.44	-4.45	20.01
en_forecast	7.27	4.40	7.10
lasso_forecast	6.88	4.07	6.67
llar_forecast	6.88	4.07	6.66
lightgbm_forecast	5.68	-8.14	2.61
bats_forecast	-0.81	-0.21	-0.65
huber_forecast	-1.79	-1.75	-2.10
arima_forecast	-7.12	-3.49	-6.56
lar_forecast	-7.90	-6.71	-8.67
polytrend_forecast	-9.06	-30.35	19.82
theta_forecast	-9.89	-5.98	-9.62
dt_forecast	-11.36	-6.37	-10.83
gbr_forecast	-11.57	-6.48	-10.99
rf_forecast	-12.05	-6.52	-11.32
et_forecast	-12.16	-6.19	-11.26
knn_forecast	-12.63	-6.91	-11.93
ada_forecast	-13.31	-7.96	-12.97
ridg_forecast	-13.98	-8.51	-13.66
Bridg_forecast	-14.08	-8.55	-13.75
lr_forecast	-14.31	-8.64	-13.95
ets_forecast	-14.69	-12.47	-15.70
expsmooth_forecast	-18.10	-14.02	-18.95
omp_forecast	-23.27	-12.37	-21.69
Auto-Arima_forecast	-23.63	-14.89	-23.45
gm_forecast	-72.23	-143.19	-13.39

Appendix B

Appendix to Chapter 2

B.1 Robustness Check: Varying Time Horizon and Hyperparameter Adjustment

In order to evaluate the resilience of our results in relation to the selection of a certain time frame, we replicated the experiment utilizing an alternative data window. In this instance, a time frame ranging from July 1, 2007 to June 15, 2015 was utilized, yielding a total of precisely 2000 data points. In order to mitigate a constraint acknowledged in the preceding experiment, we implemented a more stringent methodology by conducting hyperparameter tweaking for all employed machine learning (ML) models. Furthermore, we have expanded the application of hyperparameter tuning to deep learning models with the objective of enhancing their performance.

The findings shown enhanced predictive precision for both machine learning and deep learning models following hyperparameter optimization. In contrast to their non-tuned counterparts, deep learning models demonstrated a more substantial enhancement. This implies that meticulous hyperparameter tweaking may yield greater advantages for deep learning models. The precise enhancements in performance for each type of model are displayed in the subsequent table and accompanying graphics.

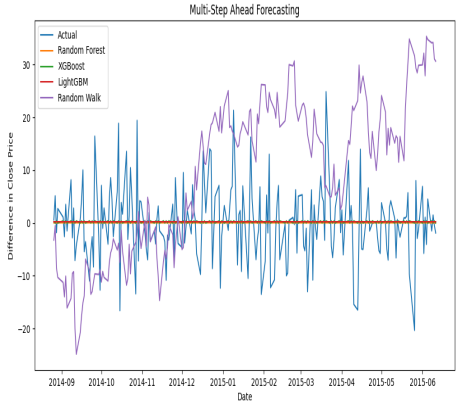
B.2 Results Table

REIT	RF (%)	LGBM (%)	XGB (%)	RNN (%)	LSTM (%)	GRU (%)	Bi-LSTM (%)
ALX	84.61	84.61	52.74	65.31	64.99	65.00	64.98
AMT	52.53	52.47	67.89	70.46	71.26	70.99	71.26
BRT	63.46	67.89	75.67	83.39	84.52	84.36	84.40
BXMT	75.63	75.65	88.66	90.66	90.66	90.65	90.66
EGP	88.75	88.63	81.41	88.14	88.08	88.10	88.11
EQIX	81.43	81.43	86.33	84.36	84.35	84.35	84.34
GTY	86.55	86.57	89.40	92.77	92.78	92.78	92.78
IHT	89.40	89.40	77.89	81.94	81.88	81.88	81.88
NHI	77.90	77.90	90.11	89.96	89.94	89.93	89.91
PCH	90.14	90.15	74.14	83.65	83.70	83.72	83.72
VNO	74.14	74.14	74.13	83.25	83.68	83.71	83.70

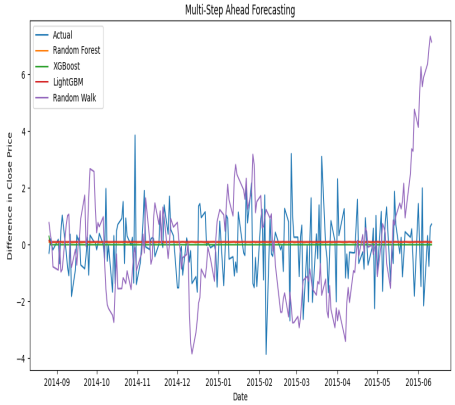
Table B.2.1: % Improvement Over Random Walk Model

B.3 Graphs

[H]



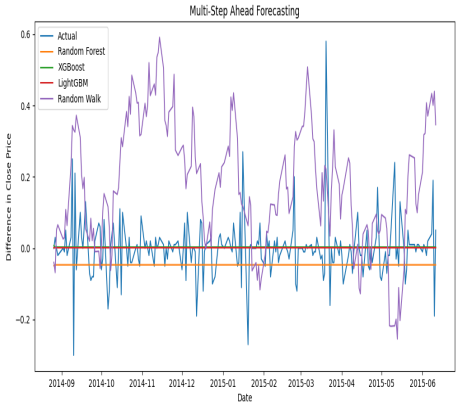
((a)) ALX



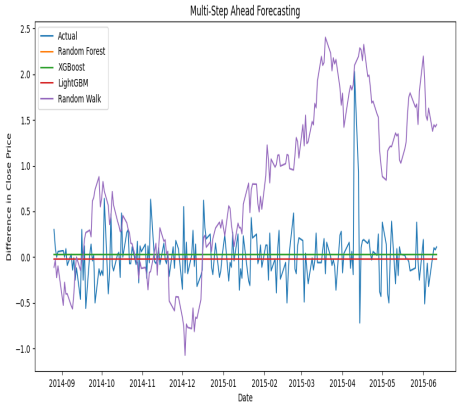
((b)) AMT

[H]

[H]



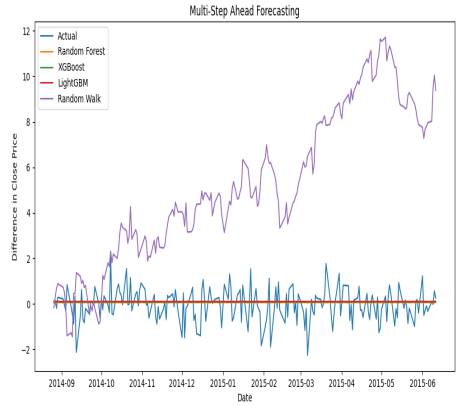
((c)) BRT



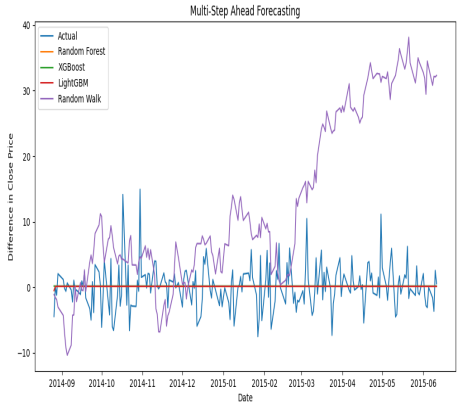
((d)) BXMT

[H]

[H]

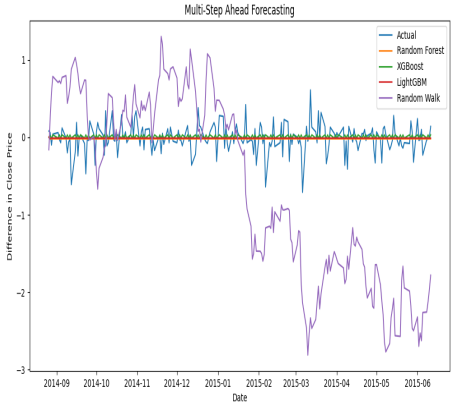


((e)) EGP



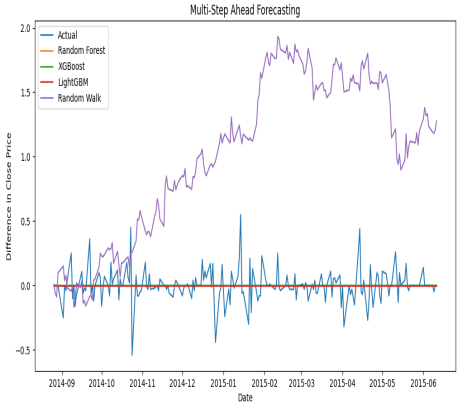
((f)) EQIX

[H]



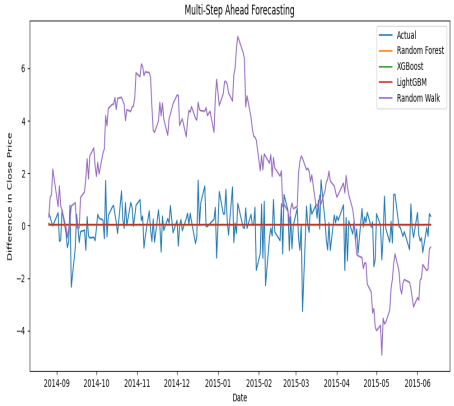
((a)) GTY

[H]



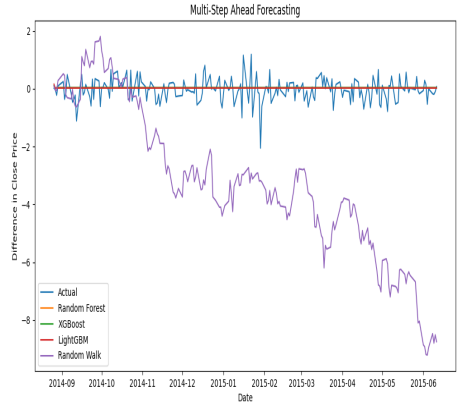
((b)) IHT

[H]



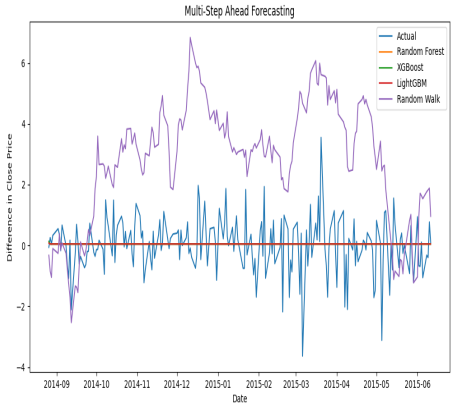
((c)) NHI

[H]



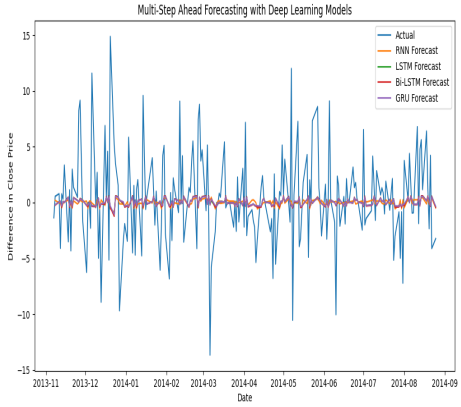
((d)) PCH

[H]

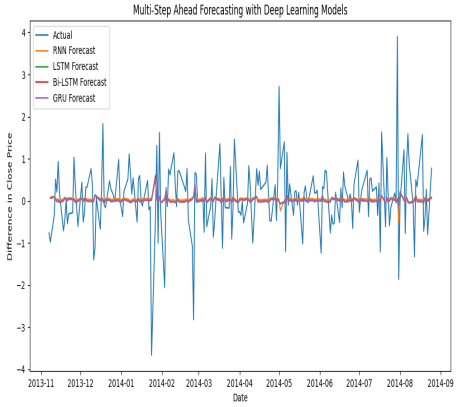


((e)) VNO

[H]



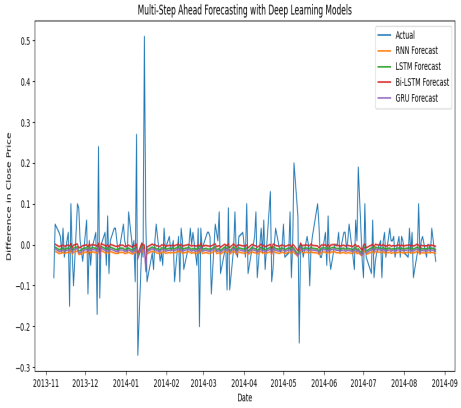
((a)) ALX



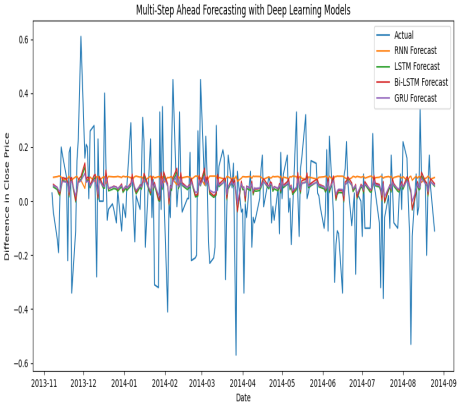
((b)) AMT

[H]

[H]



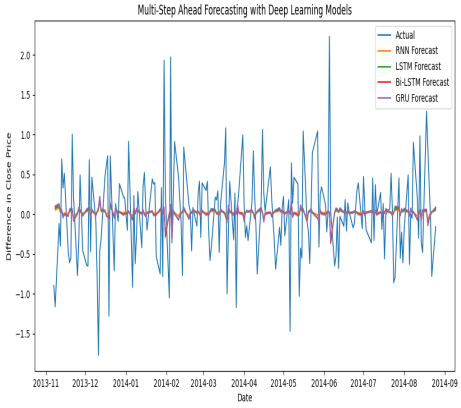
((c)) BRT



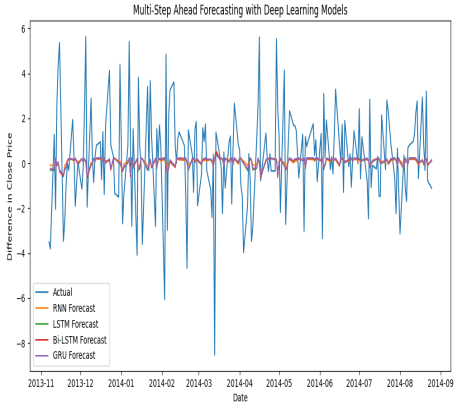
((d)) BXMT

[H]

[H]

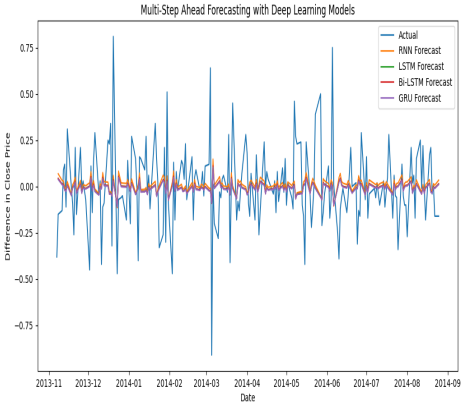


((e)) EGP



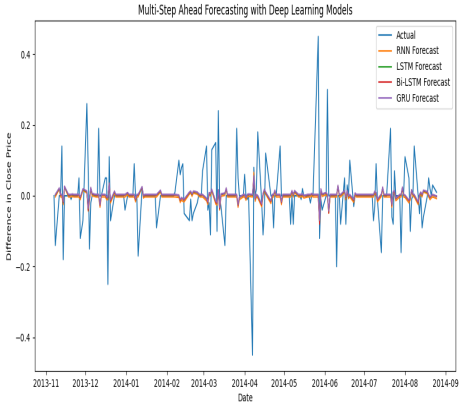
((f)) EQIX

[H]



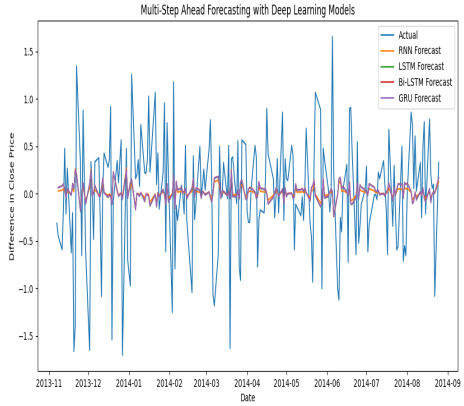
((a)) GTY

[H]



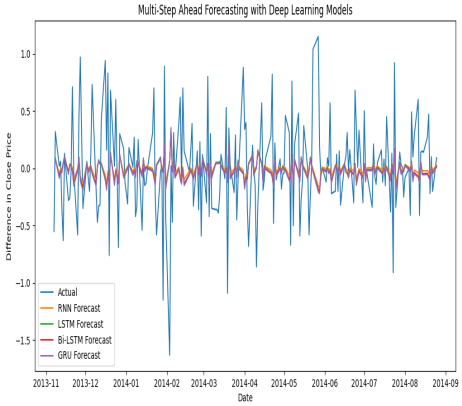
((b)) IHT

[H]



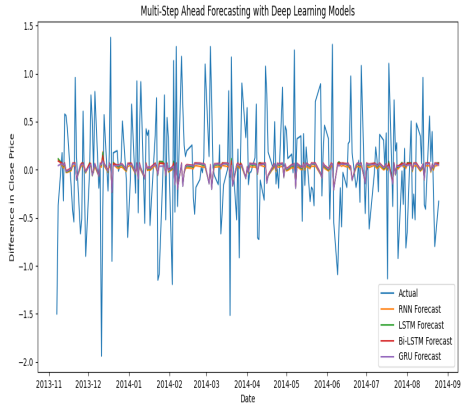
((c)) NHI

[H]



((d)) PCH

[H]



((e)) VNO

Appendix C

Appendix to Chapter 3

C.1 Robustness check

This section examines the resilience of the model's performance to two crucial factors: fine-tuning of hyperparameters and selection of features. We conducted hyperparameter optimization for all ensemble and boosting models using a randomized grid search methodology. In addition, the models were trained utilizing all five features together, rather than depending on individual features separately.

The findings presented in Table below, which displays the Normalized Root Mean Square Error (RMSE) for each dataset, indicate a varied impact on the performance of the model when all five features are combined. Some homebuilders' stock returns exhibited increased performance, as seen by a normalized RMSE value closer to 0, whereas for others, there was a modest deterioration. This implies that the impact of specific characteristics can change among various datasets.

Table below presents the normalized root mean square error (RMSE) achieved by different models for each dataset. Nevertheless, it is crucial to acknowledge that these outcomes are derived from models that have been fine-tuned with hyperparameters. The tuning method includes employing a randomized grid search approach to explore a range of alternatives for ensemble and boosting models, although the particular hyperparameter values are not displayed in this context. This strategy aids in reducing the likelihood of

overfitting and guarantees that the models are optimal for each specific dataset.

The last three models in Table 1 (Lasso, KNN Regressor, and SVM) are classified as linear models. Ensemble approaches have a larger number of hyperparameters in comparison to these models. Below is a concise overview of the hyperparameter values determined for each model:

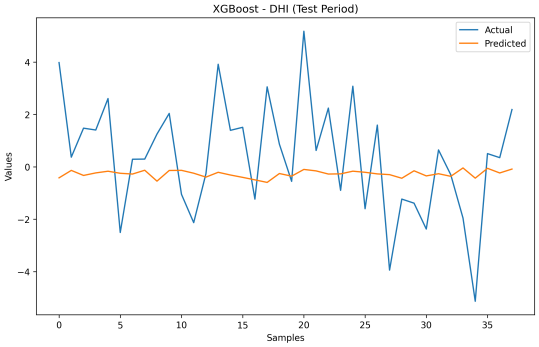
Lasso: Surprisingly, the ideal alpha value for all six datasets remained constant at 10, despite the fact that other values like as 0.1 and 1.0 were included in the range of possibilities. These findings indicate that implementing a robust L1 regularization technique yielded positive results for all datasets in this particular scenario. The KNN Regressor determined that the ideal value for the number of neighbors (n_neighbors) is 7 for all datasets. The value was selected from a search area that encompassed 3, 5, and 7 neighboring options. The Support Vector Machine (SVM) algorithm outperformed other methods in terms of performance, namely in three out of six datasets, as indicated by its reduced normalized Root Mean Squared Error (RMSE). The hyperparameter optimization for SVM involved evaluating both linear and RBF kernels, as well as three choices for the "C" parameter (0.1, 1.0, and 10). The Support Vector Machine (SVM) algorithm consistently chose the Radial Basis Function (RBF) kernel and a "C" value of 0.1 for all six datasets. This indicates a predilection for a non-linear correlation between characteristics and the target variable, along with a reduced regularization intensity. In summary, the robustness check emphasizes the significance of fine-tuning hyperparameters and conducting thorough feature selection to attain the best possible model performance. Although the combination of all five features had varying effects on different datasets, the process of hyperparameter tweaking was essential in maximizing the performance of each model for the unique data it was trained on.

Table C.1.1: Normalized RMSE for Each Dataset

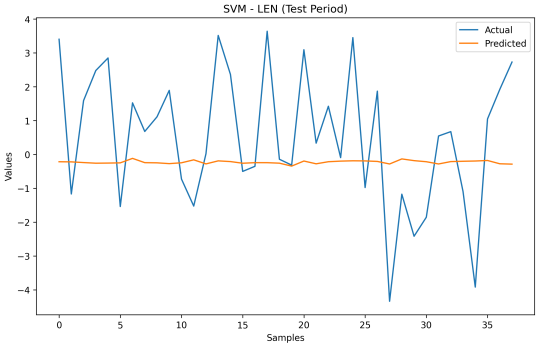
Dataset	Random Forest	Extra Trees	XGBoost	AdaBoost	LightGBM	Lasso	KNN	SVM
DHI	0.8106	0.7802	0.7699	0.9039	0.7764	0.7745	0.8447	0.7691
LEN	0.6133	0.5889	0.5942	0.7937	0.5906	0.5846	0.6391	0.5759
PHM	0.6864	0.6688	0.6612	0.7900	0.6662	0.6615	0.7230	0.6571
TOL	0.7209	0.7158	0.7118	0.8986	0.7067	0.7057	0.7656	0.6810
KBH	0.7996	0.7352	0.7522	0.9594	0.7513	0.7478	0.8292	0.7443
MTH	0.7720	0.7468	0.7378	0.9612	0.7391	0.7334	0.8289	0.7398

Figure C.1.1: Actual vs Predicted for Best Models

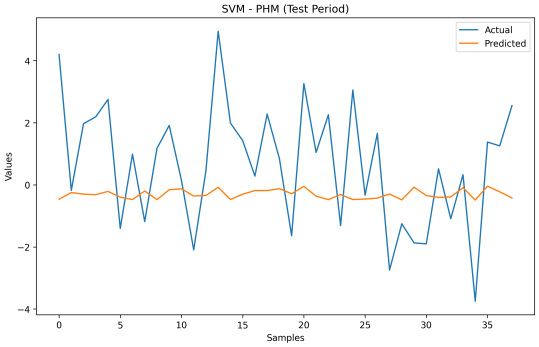
((a)) DHI



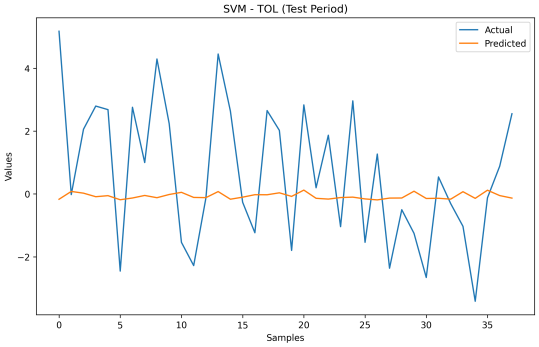
((b)) LEN



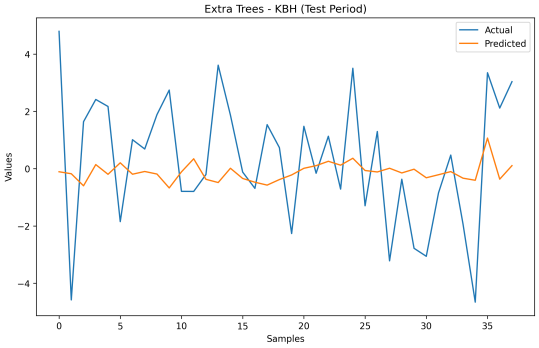
((c)) PHM



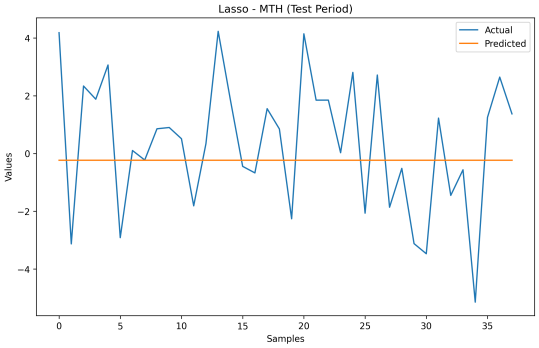
((d)) TOL



((e)) KBH



((f)) MTH



C.2 Graphs

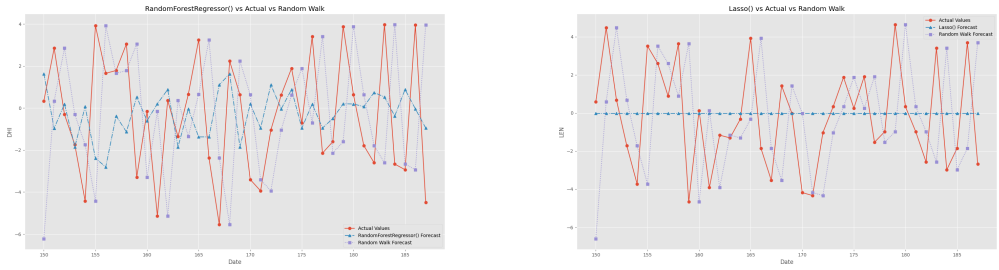


Figure C.2.2: Other better Performing Model vs. Random Walk vs. Actual Values [DHI & LEN]

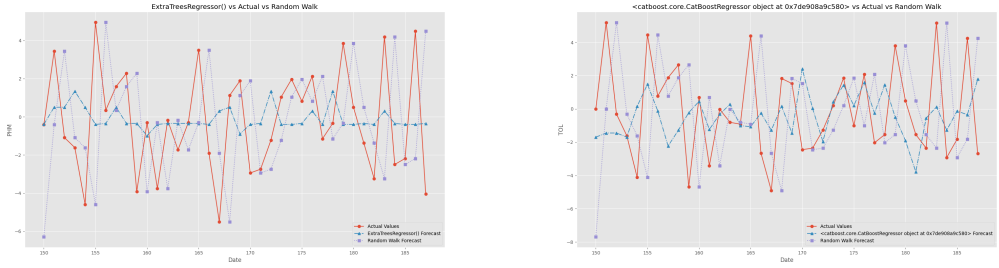


Figure C.2.3: Other better Performing Model vs. Random Walk vs. Actual Values [PHM & TOL]

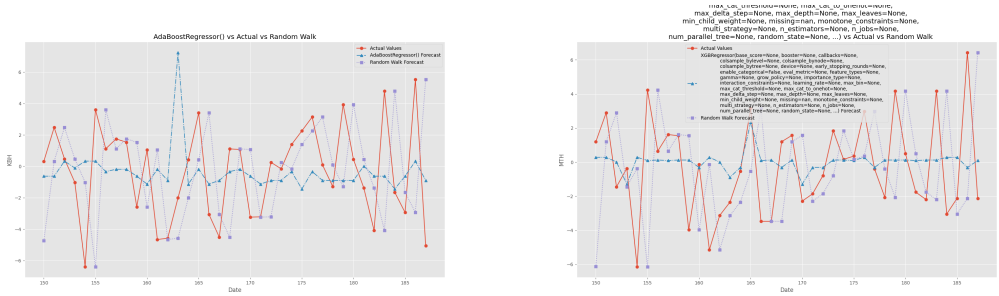


Figure C.2.4: Other better Performing Model vs. Random Walk vs. Actual Values [KBH & MTH]

Table C.1.2: Random Forest Hyperparameters for Each Dataset

Dataset	Hyperparameters
DHI	n_estimators=50, min_samples_split=2, min_samples_leaf=4, max_features='log2', max_depth=None
LEN	n_estimators=100, min_samples_split=2, min_samples_leaf=4, max_features='log2', max_depth=5
PHM	n_estimators=50, min_samples_split=10, min_samples_leaf=4, max_features='sqrt', max_depth=None
TOL	n_estimators=50, min_samples_split=10, min_samples_leaf=4, max_features='sqrt', max_depth=None
KBH	n_estimators=200, min_samples_split=5, min_samples_leaf=1, max_features='log2', max_depth=5
MTH	n_estimators=200, min_samples_split=2, min_samples_leaf=4, max_features='log2', max_depth=10

Table C.1.3: Extra Trees Hyperparameters for Each Dataset

Dataset	Hyperparameters
DHI	n_estimators=50, min_samples_split=2, min_samples_leaf=4, max_features='log2', max_depth=None
LEN	n_estimators=50, min_samples_split=10, min_samples_leaf=4, max_features='sqrt', max_depth=None
PHM	n_estimators=50, min_samples_split=10, min_samples_leaf=4, max_features='sqrt', max_depth=None
TOL	n_estimators=200, min_samples_split=2, min_samples_leaf=4, max_features='log2', max_depth=10
KBH	n_estimators=100, min_samples_split=2, min_samples_leaf=1, max_features='sqrt', max_depth=5
MTH	n_estimators=50, min_samples_split=10, min_samples_leaf=4, max_features='sqrt', max_depth=None

Table C.1.4: XGBoost Hyperparameters for Each Dataset

Dataset	Hyperparameters
DHI	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5
LEN	subsample=0.7, n_estimators=50, max_depth=3, learning_rate=0.01, colsample_bytree=0.7
PHM	subsample=0.7, n_estimators=50, max_depth=3, learning_rate=0.01, colsample_bytree=0.7
TOL	subsample=0.7, n_estimators=50, max_depth=3, learning_rate=0.01, colsample_bytree=0.7
KBH	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5
MTH	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5

Table C.1.5: LightGBM Hyperparameters for Each Dataset

Dataset	Hyperparameters
DHI	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5
LEN	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5
PHM	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5
TOL	subsample=0.7, n_estimators=50, max_depth=3, learning_rate=0.01, colsample_bytree=0.7
KBH	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5
MTH	subsample=0.5, n_estimators=50, max_depth=5, learning_rate=0.01, colsample_bytree=0.5