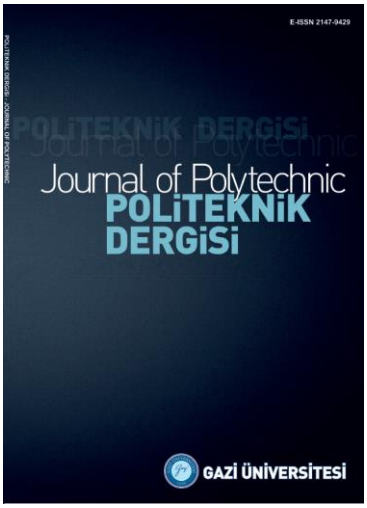

POLİTEKNİK DERGİSİ

JOURNAL of POLYTECHNIC

ISSN: 1302-0900 (PRINT), ISSN: 2147-9429 (ONLINE)
URL: http://dergipark.org.tr/politeknik

A comprehensive analysis of resilient multivariate forecasting models for steel plate price prediction

Çelik levha fiyat tahmini için esnek çok değişkenli tahmin modelleri

Yazar(lar) (Author(s)): *Mahmud ALsaideen[1], Zeynep Ertem[2]*

ORCID[1]: 0000-0003-0436-0372

ORCID[2]: 0000-0003-0632-0905

To cite to this article: Alsaideen M. and Ertem Z., "A Comprehensive Analysis of Resilient Multivariate Forecasting Models for Steel Plate Price Prediction", ***Journal of Polytechnic***, 28(2): 627-637, (2025).

Bu makaleye şu şekilde atıfta bulunabilirsiniz: Alsaideen M. ve Ertem Z., "A Comprehensive Analys of Resilient Multivariate Forecasting Models for Steel Plate Price Prediction", ***Politeknik Dergisi***, 28(2): 627-637, (2025).

Erişim linki (To link to this article): http://dergipark.org.tr/politeknik/archive

DOI: 10.2339/politeknik.1438983

A Comprehensive Analysis of Resilient Multivariate Forecasting Models for Steel Plate Price Prediction

Highlights

- *Makina öğrenmesi / Machine learning*
- *Çelik / Steel*
- *Tahmin bilimi / Forecasting*
- *Derin öğrenme / Deep learning*

Graphical Abstract

The global steel industry, holding paramount economic significance, is characterized by the inherent volatility of steel prices. This research article aims to predict the steel prices.

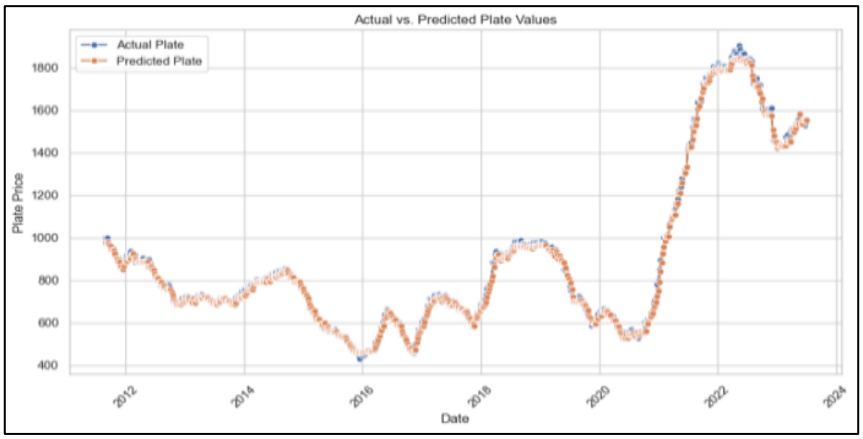

Figure. Graphical Abstract for Steel Prices

Aim

The global steel industry, holding paramount economic significance, is characterized by the inherent volatility of steel prices. Leveraging the reliable weekly steel plate price data from the Commodity Research Unit (CRU), this research employs sophisticated machine learning algorithms to forecast plate prices.

Design & Methodology

The dataset spans from July 27, 2011, to July 5, 2023, encompassing six key predictive factors. Notably, total inventory levels exhibit the highest correlation (0.88) with plate prices, with the finished goods inventory value of heavy machinery emerging as the most influential factor. A comprehensive training regimen is undertaken for machine learning models, incorporating Prophet, XGBoost, LSTM, and GRU.

Findings

Time Series Cross-Validation is implemented to maintain the temporal order of the data, and a Bayesian optimization function is employed for hyperparameter tuning. Demonstrating superior predictive accuracy, with a Mean Absolute Percentage Error (MAPE) of 0.94% and a Root Mean Squared Error (RMSE) score of 18.06, XGBoost establishes itself as the most effective model in steel plate price forecasting.

Conclusion

Outcomes underscore the efficacy of advanced machine learning methodologies in navigating the complexities of steel market dynamics for enhanced predictive insights.

Declaration of Ethical Standards

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

A Comprehensive Analysis of Resilient Multivariate Forecasting Models for Steel Plate Price Prediction

Araştırma Makalesi / Research Article

Mahmud ALSAIDEEN[1], Zeynep ERTEM[1*]
[1]Systems Science and Industrial Engineering Department, Binghamton University,
State University of New York, NY, United States
(Geliş/Received : 18.02.2024 ; Kabul/Accepted : 15.04.2024 ; Erken Görünüm/Early View : 04.09.2024)

ABSTRACT

The global steel industry, holding paramount economic significance, is characterized by the inherent volatility of steel prices. Leveraging the reliable weekly steel plate price data from the Commodity Research Unit (CRU), this research employs sophisticated machine learning algorithms to forecast plate prices. The dataset spans from July 27, 2011, to July 5, 2023, encompassing six key predictive factors. Notably, total inventory levels exhibit the highest correlation (0.88) with plate prices, with the finished goods inventory value of heavy machinery emerging as the most influential factor. A comprehensive training regimen is undertaken for machine learning models, incorporating Prophet, XGBoost, LSTM, and GRU. Time Series Cross-Validation is implemented to maintain the temporal order of the data, and a Bayesian optimization function is employed for hyperparameter tuning. XGBoost emerges as the top-performing model, yielding the lowest Mean Squared Error (MSE) of 332.25 and Mean Absolute Error (MAE) of 14.55. Demonstrating superior predictive accuracy, with a Mean Absolute Percentage Error (MAPE) of 0.94% and a Root Mean Squared Error (RMSE) score of 18.06, XGBoost establishes itself as the most effective model in steel plate price forecasting. This outcome underscores the efficacy of advanced machine learning methodologies in navigating the complexities of steel market dynamics for enhanced predictive insights.

Keywords: Machine Learning, Steel, Forecasting, Deep Learning.

Çelik Levha Fiyat Tahmini İçin Esnek Çok Değişkenli Tahmin Modelleri

ÖZ

Büyük bir ekonomik öneme sahip olan küresel çelik endüstrisi, çelik fiyatlarındaki doğal değişkenlik ile karakterize edilmektedir. Emtia Araştırma Birimi'nin (CRU) güvenilir haftalık çelik levha fiyatı verilerinden yararlanan bu araştırma, levha fiyatlarını tahmin etmek için gelişmiş makine öğrenimi algoritmaları kullanıyor. Veri seti 27 Temmuz 2011 ile 5 Temmuz 2023 arasındaki dönemi kapsıyor ve altı temel tahmin faktörünü içeriyor. Özellikle, toplam stok seviyeleri levha fiyatlarıyla en yüksek korelasyonu (0,88) sergilerken, ağır makinelerin nihai ürün stok değeri en etkili faktör olarak ortaya çıkıyor. Makine öğrenimi modelleri için Prophet, XGBoost, LSTM ve GRU'yu içeren kapsamlı bir eğitim rejimi yürütülmektedir. Verilerin zamansal sırasını korumak için Zaman Serisi Çapraz Doğrulama uygulanır ve hiperparametre ayarı için bir Bayesian optimizasyon işlevi kullanıldı. XGBoost, 332,25 ile en düşük Ortalama Karesel Hatayı (MSE) ve 14,55 ile Ortalama Mutlak Hatayı (MAE) sağlayan en iyi performansa sahip model olarak ortaya çıkıyor. %0,94 Ortalama Mutlak Yüzde Hata (MAPE) ve 18,06 Ortalama Karekök Hata (RMSE) puanıyla üstün tahmin doğruluğu sergileyen XGBoost, çelik levha fiyat tahmininde en etkili model olarak kendisini kanıtlıyor. Bu sonuç, gelişmiş tahmine dayalı içgörüler için çelik piyasası dinamiklerinin karmaşıklıklarını yönetmede gelişmiş makine öğrenimi metodolojilerinin etkinliğini vurgulandı.

Anahtar Kelimeler: Makina öğrenmesi, Çelik, Derin öğrenme, Tahmin bilimi

1. INTRODUCTION

The steel industry has been the backbone of the modern world's economy and on a large scale. Many nations depend on steel production for boosting their economy. China supplied roughly one billion tons of steel which accounts for over 55% of the world's overall steel production in 2020 [1]. The importance of steel grew even more after the second industrial revolution. According to John Mclean's book *History of Western Civilization II*, steel became one of the new areas that has a mass-production principle as the machinery industry started booming. Crude steel is often referred to as steel when it is in its first solidified state. Crude steel is then turned into a variety of steel products under two categories, longitudinal products, and flat products.

According to commodity market analyses, the price is usually determined based on a number of obvious factors like supply and demand, inventory and production rates, and other economic indicators [2]. Commodity market research involves gathering, analyzing, and interpreting information related to a specific commodity [3]. Commodity market research involves conducting price analysis by tracking price and seeking expert opinion to make data-driven buying decisions. It also includes following the updates of the commodities supply flow and interruptions and doing risk assessment if necessary. A developing part of commodity market research is

* *Sorumlu Yazar (Corresponding Author)*
e-mail : zeynep@binghamton.edu

forecasting the price of the commodity by using traditional statistical models and more recently using machine learning algorithms [2].

For time series forecasting tasks, statistical models have been widely used until some machine learning algorithms entered the mix and proved to give more accurate forecasts. Some statistical models that are still widely used are ARIMA and Vector Autoregressive (VAR). Some machine learning algorithms that are commonly used for time series forecasting are Prophet and XGBoost. The algorithms detect the patterns of historical data, learn the trends, and capture the seasonality [4].

Our study uses two machine learning and two deep learning algorithms to predict the steel price. A dataset containing the historical price data for the plate is compiled with factors that have a relationship with the price and correlate with it. The study uses state-of-the-art algorithms for time series forecasting and utilizes various machine learning techniques to achieve highly accurate long-term and short-term forecasts for the price.

1.1 Background

Forecasting is the process of generating predictions or estimates of future values by leveraging historical data [5]. Forecasting is a tool that has been used for a while in a variety of fields like economics, finance, business, and metrology. Depending on the objective and the domain knowledge, the best method is decided [5]. Forecasting relies heavily on historical data to make future predictions. Forecast horizon is the time or period to be predicted. It can be short – term which can be days to weeks, medium - term spanning months to years, and long - term which could extend from years to decades). The rising categories depend on the target to be forecasted.

Forecasting started with traditional time series models. The term "time series models" is used because the data fitted into the model adheres to a time series format, exhibiting a sequential and temporal order. The traditional models are Moving Average, Exponential Smoothing, Autoregressive integrated moving average (ARIMA), and Seasonal Decomposition. ARIMA is considered to be the most popular statistical model and it works through incorporating autoregressive and moving average components along with making the time series stationary [6]. ARIMA is shown in equation 1.

$$y_t = c + \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + .. + \varphi_p y_{t-p} + \epsilon_t \quad (1)$$

Moreover, machine learning algorithms are now in wide use to conduct time series forecasting. Regression is used to predict the requested horizon period. Some of the algorithms that have been proven to be effective for time series data include Support Vector Machine, Long Short-Term Memory (LSTM), and Random Forests.

When it comes to consumption rates, China is the largest consumer. Especially with the current increase in its manufacturing activity. Other leading countries include India, United States and Russia. Steel production is one of the largest causes of air pollution [1]. Environmental regulations might cause major supply chain disruptions in the future due to the drastic impact the seal production industry is leaving on the environment. The demand for steel is increasing with the years and it's expected to be increasing given that many countries are developing their infrastructure in many countries are boosting their manufacturing industry. With increased demand, production is increased to meet the demand. The Commodity Research Unit (CRU) is a well-known and respected research firm that many commodity managers are using to make informed buying decisions. They specialize in the analysis and research of commodity markets. They provide data, insights, and a variety of resources regarding various commodities. They also released some commodity prices on a weekly basis. Their price is used by a majority of exporters and is reliable. For our study we use the CRU's plate price historical data to make predictions.

Predictive and prescriptive analytics represent the latest trend in business intelligence. Establishing a knowledge-driven organization that utilizes the forecasting of steel prices, as well as the associated risks and opportunities, is paramount. Moreover, machine learning algorithms are rapidly advancing to boost their performance and address the challenges they encounter. This section underlines the research's goals and the key inquiries upon which this study is based. Predictive and prescriptive analytics encompasses the use of data analysis techniques to not only predict steel prices (Predictive Analytics) but also to provide recommendations and solutions for optimizing decision-making in the steel market (Prescriptive Analytics). This approach goes beyond forecasting and aims to help firms make more informed and strategic choices regarding steel pricing, procurement, and risk management.

We start by introducing the methods in the old literature and where the autoregressive models started. [7] worked around forecasting long term horizons of commodity price volatility. Their research includes three approaches: the first one is making forecasts that are based on option prices which means that the forecasts are generated exclusively by analyzing the pricing of financial options such as call (buy) and put (sell) options. The second approach is forecasting using time series modeling analysis. Forecasting the price of steel accurately can return the organization significant savings by planning buying according to the forecast. While working with time series data presents inherent challenges, the potential rewards can be substantial. As we will see in the literature, there is a decent number of research studies on time series analysis in general and forecasting specifically. But the number of studies on using machine learning to forecast commodities and especially steel studies is limited. There exist numerous research studies on how to accurately forecast commodities. Xu & Zhang's [2] claim to be the first study to use machine learning algorithms to forecast the steel price. To the best of our knowledge, our work serves as one of the first research studies to use machine learning algorithms to forecast plate steel prices in the United States. The final

category is a combination of market expectations and time series models. The authors suggested approach yields the best results. Their model combined the generalized Autoregressive Conditional Heteroskedasticity (GARCH) and Implied Standard Deviations (ISD). Both of those models combine time series forecasting and options prices in that order. Their model is denoted as COMB and is shown in equation 2 steel price specifically.

In another study, Xiong et. al., [8] worked to forecast agricultural commodity prices. They use a model called Vector Error Correction Model (VECM) which is a statistical time series model used in econometrics and Multioutput Support Vector Regression (MSVR) which follows a linear and nonlinear modeling framework. Their specific objective was to predict the price range of some agricultural commodities. For agricultural commodities, the price is a range or an interval rather than a fixed value. Therefore, the authors refer to it as an interval dash valued future price. The framework of using VECM and MSVR models, successfully captures the patterns in the price, be it linear or non-linear. It can statistically outperform other competitors like ARIMA. Palazzi et. al.,[9] forecasts commodity prices in Brazil using a hybrid approach of Singular Spectrum Analysis (SSA) and complex seasonality models. Their aim was to predict the monthly price of some agricultural commodities. SSA is a data analysis technique that decomposes time series data into its underlying components which reveals trends, seasonal patterns, and noise. Their hybrid model outperformed single models they used to compare with their approach.

$$\boldsymbol{h_t = \omega + \alpha\varepsilon_{t-1}^2 + \beta h_{t-1} + \delta\sigma_{t-1}^2} \qquad (2)$$

Deep learning techniques are widely used in time series analysis tasks. Time series data is complex and challenging to work with in machine learning. The traditional data preprocessing techniques are not always enough. Deep learning techniques can learn relevant patterns from time series data which often eliminates the need for manual feature engineering. Another deep learning algorithm and a variant of Recurrent Neural Networks (RNN) that is famous for its accurate results is Gated Recurrent Unit (GRU). Its ability to process sequential data makes it one of the top performers when it comes to time series forecasting.

Ameur et. al., [10] provides an overview on deep learning algorithms. Their study gives a thorough investigation of the capability of some RNN models for predicting commodity prices. Their investigation results show the resilience and robustness of Long Short-Term Memory Algorithm (LSTM) in predicting several commodity indices. The algorithm shell has an input gate, output gate and forget gate. As for the information the input gate controls, it includes the information that is accepted by the cell, then transferred to the cell. Any other information that is neglected, goes to the forget gate. The information is then transferred to the output gate which generates the output and provides the state [11].

The performance metrics used were Mean Absolute Deviation (MAD), R-squared (R2), Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE). Furthermore, the algorithms [10] used were Recurrent Neural Networks (RNN), Long Short-Term Memory Algorithm (LSTM), Convolutional Neural Networks (CNN), Gated Recurrent Unit (GRU), Artificial Neural Networks (ANN) and Autoregressive Fractionally Integrated Moving Average Model (AFIMA).

eXtreme Gradient Boosting commonly referred to as XGBoost is an effective machine learning algorithm when it comes to processing time series data and its prediction accuracy. XGBoost is designed to generate predictions for regression and classification tasks, and it is preferred for regression tasks for two main features; gradient boosting where it creates a strong prediction model based on multiple decision trees (ensemble learning) and it's L1 and L2 regularization techniques [12] An example of use of this impressive algorithm is the research of Oukhouya and El Himdi [13] where they used it to predict the Moroccan stock market. Another Machine learning algorithm that is designed for time series forecasting is Prophet created by Facebook. It can process and work with any form of time-based data. Some of its prominent features include automatic seasonality detection, customizable holidays, trend modeling, additive and multiplicative components, and scalability [14].

2. METHODOLOGY

A data set was built to support the objective of forecasting the price of steel plate. The data was in a monthly format and then transformed to weekly format to match the plate data. Another factor that was found to be helpful was the inventory levels of finished goods of heavy machinery. The levels of finished goods were also obtained from The United States Census Bureau. From the literature it was found that the United States dollar strength has predictive power over the steel price. The strength of the United States dollar was represented in the dollar index that is maintained by Yahoo finance. Daily data from Yahoo finance was obtained and then transformed into weekly. The U.S. Dollar along with the Australian dollar were found to be highly correlated to the steel price. As mentioned before, the term "Commodity Currency" refers to currencies that are affected by the price of a certain commodity that the country's economy relies on heavily. The literature also led this research to investigate the fuel price as a factor. The data for fuel price was pulled from the United States Energy Information Administration (EIA). To conclude this section all data sources are found to be trusted and reliable.

In this research, utilizing machine learning and deep learning algorithms constitutes a use of a variety of data preparation techniques. Therefore, several techniques were used but not necessarily on every model. It is important to lay out the algorithms in this work to

understand what technique was used on what model and why some techniques were not used on some models. The first model was Facebook's time series forecasting model, Prophet. This model is constructed to handle time series data and does not require any data preprocessing steps as mentioned in the literature review. After Prophet, XGBoost was used. This model requires stationary data, so a differencing technique is often required. Long Short – Term Memory (LSTM) was then used which can handle sequential data. A prerequisite however is to scale the data if the data is not stationary to make the model less complex which eventually produces more accurate predictions. Another Recurrent Neural Network (RNN) model that was used is Gated Recurrent Unit (GRU). It is also recommended to use a scaling technique and use a time series generator to fit the model which is also recommended for LSTM as well. The following sections will address all the techniques used. Figure 1 shows how the plate price is plotted and non-stationarity can be observed.

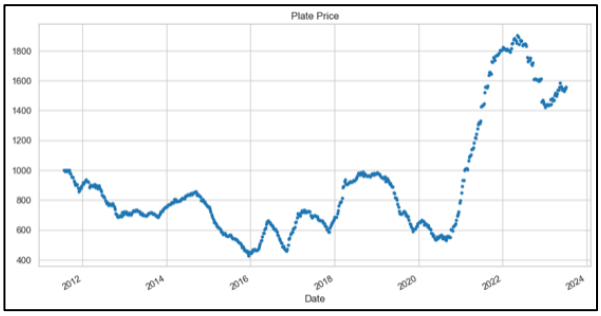

Figure 1. Plot of plate price.

2.1 Feature Engineering – Lag Features

A feature engineering technique that is used in time series analysis and other sequential data modeling tasks. It involves creating new features based on the values of a variable at previous time points. They are very useful in capturing temporal dependencies and historical patterns in the data. In our case, we created lag features out of the target variable. The value is shifted backward in time. For the play price, six lag features were created which are the previous price of the plate six weeks back in time. Their purpose was to capture and represent the history overtime series. They allow machine learning algorithms to learn the influence of historical data of the target variable when the algorithm is not designed to do so. The six lag features were used with XGBoost, LSTM and GRU. Prophet is designed to consider the influence of the target (Y) historical data unlike the other algorithms used where the model only considers the target value with respect to the independent variables used.

2.2 Time Series Cross Validation

The data was initially split 90% for training and 10% for testing the reason this was chosen as a split was because of the spike of price towards the end of the time series and then the drop is noticed in Figure 1. The spike started right after COVID-19 pandemic had slowed down a little in 2021 therefore app, after trial and error, 90/10 split was found optimal. For XGBoost, the time series cross validation that worked best was Expanding Window Cross – Validation. This technique allows the assessment of performance of machine learning models as it adapts and learns from an increasing amount of historical data. Initially, part of the data is used for training and part of the 10% holdout set is considered as the test set for the first fold. In the first iteration, the model is trained on the initial training data set and then evaluated on the corresponding validation set. After the first iteration is done, the training window expands to include the testing set from the first iteration and so on. In our case, we used five folds. Some of the key characteristics of Expanding Window Cross – Validation is adaptive learning. It implements this adaptation technique in the model to be able to progressively adapt to new data. It also maintains the temporal order of data. Figure 2 shows how the plate data split observed the expanding training set.

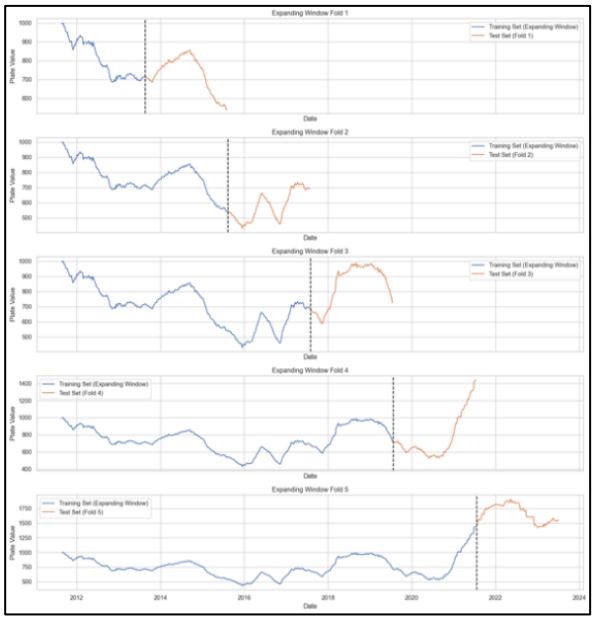

Figure 2. Expanding window cross – validation.

For the other models a similar technique was implemented. TimeSeriesSplit [15] from Sci-Kit Learn was used. it is very similar to Expanding Window Cross – Validation. However, it works differently. It extends the traditional K - Fold Cross - Validation to handle sequential data. It begins by sequentially splitting the data in a forward manner. It is more of a sliding window rather than an expanding window. It first begins by splitting the first fold into a training set and a testing set. Once the first iteration is done, the entire first fold becomes a training set and so on. Figure 3 shows how the Time Series Split was used.

2.3 Time Series Generator

Two time series generators were created. One is the training generator, and the other is testing generator. It is a formidable tool for creating batches of sequential data that can be used for training RNN and other sequential models. Time Series Generator is part of the Keras library for deep learning tasks[16].

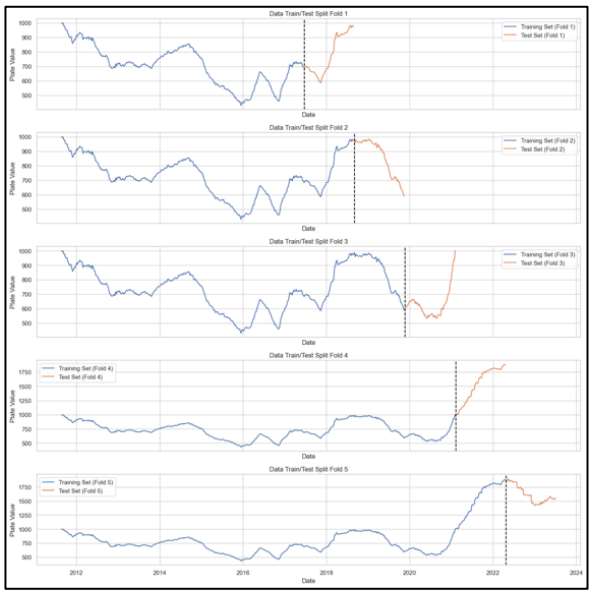

Figure 3. Time series cross – validation

The purpose of a generator is to help generate batches of temporal sequences from a given time series. It is especially helpful when working with LSTM and GRU. The generator takes the time series data, a specified length for input sequences, and a length for target sequences. The batch size can be specified as a parameter within the generator, and it was set to be 1 after trial and error. As for the length of steps, it was set to be 4 steps per batch. The x shape was (557, 4, 12) meaning 557 samples, number of time steps and the number of features.

2.4 Hyperparameter Tuning

For tuning the hyperparameters, HyperOpt was used to optimize the search. HyperOpt is a hyperparameter optimization and a Python library [17], and it works much faster than GridSearch. It uses a Bayesian optimization technique. It consists of two important components, the objective function, and the search space. The objective function takes a combination of hyperparameters in and sends back a score (Mean Squared Error in our case) as the score to minimize. The search space is defined for the hyperparameters that are to be optimized. It specifies the possible values or ranges for each hyperparameter. It is important to know that there are some hyperparameters that are manually tuned. To prevent overfitting, time series cross – validation was used to ensure the models' generalization across diverse temporal patterns. Regularization techniques, including L1 and L2 regularization, were applied. The integration of early stopping, coupled with cautious adjustments to learning rates, further contributed to preventing overfitting.

2.5 Exploratory Data Analysis (EDA)

An EDA was carried out to outline the data shape, relationships, and characteristics. The goal was to understand the data distribution and variability. The EDA helped in showcasing the correlation of the features with the plate price. It is important to investigate such cases to prevent collinearity between the regressors (features). It allowed for dimension reduction which evidently should improve the model's performance. There were no missing values nor any underlying issues with the data as a whole as it was constructed consistently and carefully. Figure 4 shows a decomposition of the time series. Which is a statistical visual used to break down parts of the time series into its underlying factors which include the residual factor, seasonal factor, and trend factor. The trend factor shows the long-term systematic movement in the data over time as seen in Figure 4. On the other hand, the seasonal factor shows the repeating patterns which occur at fixed intervals. This was key in creating the holiday parameter for the Prophet model and determining the number of steps and batch sizes for LSTM and GRU. Lastly, the residual factor resembles the error factor and accounts for the irregular and unpredictable fluctuations in the time series that have nothing to do with the trend and seasonal factor. It captures the noise in the data. All the data columns are continuous. Basic statistics are outlined in Table 1

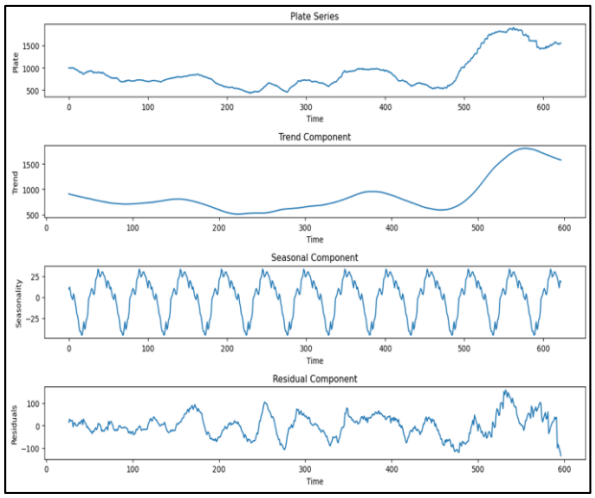

Figure 4. Time series decomposition

2.6 Model Construction and Performance Metrics

In this section, the construction of the models is reviewed. The performance metrics used for model evaluation are discussed. An overview of how each model generates predictions. The models' performance was evaluated on the holdout set (Test Set) after the Time Series Cross – Validation and parameter optimization were done. All models were optimized on the MSE score. The models were evaluated on the holdout set using the following metrics: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE). The MSE was also the metric that the models were optimized by. MSE is a common statistical metric that is used in regression tasks. It quantifies the average squared difference of the observed value and the predicted value. MSE represents the accuracy of predictive models. MAE is also a common metric to evaluate regression models. It measures the absolute difference between actual and predicted values. Just like the last two, RMSE also evaluates the accuracy of a predictive model.

Table 1. Data descriptions

	Count	Mean	Std	Min	Max
Plate	623	900.16	383.14	429	1904
USD Index	623	92.41	8.33	73.81	113.31
Capacity Utilization	623	75.33	4.95	52.92	84.22
Gasoline Price	623	2.94	0.63	1.73	4.91
Back Orders Value	623	16252.63	3784.86	10114	23713
Total Inventory	623	21363.42	3178.63	16269	28854
Finished Goods Inventory	623	18529.22	2423.17	14680	23942

It provides a measure of the square root of the average squared difference between actual and predicted values. MAPE is a common statistical metric for measuring the accuracy of forecasts, in our case, time series forecasting. It quantifies the average percentage difference of actual and predicted values.

2.7 Prophet

Facebook's Prophet was the first model used in this research. Its robustness to messy data, handling missing data (if any) and outliers made it a good choice to start with [18]. The model decomposes the sequential data into its key components: trend, seasonality, holidays and error or noise [19]. For the trend component, Prophet primarily utilizes a piecewise linear function to capture different patterns and trends in the data. The model also uses a logistic growth curve to handle seasonal component and holidays. The model automatically identifies changepoints in the data when the growth rate changes. The piecewise linear curve was used for the growth.

The seasonal components Prophet captures are yearly, weekly, and daily seasonality. Depending on the data horizon, it can be custom to capture monthly or hourly even [20]. A holiday effect was added for COVID-19 to accommodate the drastic changes in the price around that period. Prophet allows the incorporation for several events (holidays) and their dates to make the model account for those events. The equation for forecasting is as follows:

$$y(t) = \mathrm{g(t)} + s(t) + h(t) + \varepsilon_t \qquad (3)$$

Where:

$y(t)$ is the observed value at time t.

$\mathrm{g(t)}$ is the trend component.

$s(t)$ is the seasonal component.

$h(t)$ represents the holiday effect.

ε_t is the error term.

2.8 XGBoost

eXtreme Gradient Boosting commonly referred to as XGBoost is an ensemble learning algorithm. The predictive model is built by combining the predictions of multiple individual models, which typically are decision trees [21]. It can be utilized for classification tasks or regression tasks. The decision trees are base learners and are often shallow and have a limited depth to prevent overfitting. For boosting, XGBoost employs a great technique to build the ensemble of decision trees. It does so by training a new decision tree to correct the previous decision tree [12]. The objective function can be defined to accommodate the task at hand. For our purposes, "reg:squarederror" was used. The model generates the predictions by combining the predictions of the decision trees into a final ensemble prediction [12]. Each tree corrects the errors made by the previous tree, and so on. The prediction generated by the model can be represented as:

$$y_i = \varphi(x_i) = \sum_{k=1}^{1} f_k(x_i) \qquad (4)$$

Where:

y_i is the prediction for datapoint x_i.

K is the total number of trees in the ensemble.

$f_k(x_i)$ is the prediction made by the kth tree for the datapoint x_i.

2.9 Long Short – Term Memory (LSTM)

LSTM, which is a Recurrent Neural Network architecture that particularly handles time series data. LSTM is highly effective and reliable in various tasks, especially time series forecasting. The model is constructed as a network of interconnected cells which are cells designed to capture patterns and dependencies in sequential data [22]. The cells are the core block of an LSTM model. Each cell can maintain an internal state that can update or forget depending on the input data and the former internal state. The layers and units' aspect of an LSTM network is as important as the LSTM cells. An LSTM model can consist of multiple layers and within each layer, there can be one or more LSTM cells [23]. The layers can be stacked to make what is called a deep LSTM network. Depending on the goal and objective of the model built, the number of layers and units is determined. Data plays a role in determining so as well. The model used had three layers. Two LSTM layers with 64 cells in the first one and 32 cells in the second one. The third layer was a dropout layer.

Since the data is time series data, it was divided into time steps. Each time step corresponds to a specific point in

time. LSTM was specifically designed to address the vanishing gradient problem as mentioned before [24]. The goal was to make it more effective in learning long-range dependencies in the time series.

Within every LSTM model, a pivotal component is the memory cell, designed to retain and store information across numerous time steps. It could read, write, or discard information. The functionality of the memory cell is controlled by three gates that regulate the flow of information and they are input gate, forget gate, and output gate. The input gate decides what information from the present time step to be included to the memory cell. Moreover, the forget gate controls what information from the former state is to be removed from the memory cell. The output gate decides what information should be read from the memory cell to generate as an output for the current step. An LSTM model has a hidden state and a cell state. The hidden state passes from one cell to the subsequent in the series. The cell state can store longer-term information [25]. The final output of the LSTM model is generated based on the contents in the hidden state. An LSTM network uses backpropagation to adjust the model's parameters to reduce the error. Below is a simplified representation of the LSTM cell's computation for one time step:

Input Gate (i): $\boldsymbol{i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)}$ (5)

Forget Gate (f): $\boldsymbol{f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)}$ (6)

Output Gate (o): $\boldsymbol{o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o)}$ (7)

Cell State (c): $\boldsymbol{c_t = f_t \cdot c_{t-1} + i_t \cdot tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)}$ (8)

Hidden State (h): $\boldsymbol{h_t = o_t \cdot tanh(c_t)}$ (9)

Where:

i_t is the input gate.

f_t is the forget gate.

c_t is the cell state.

o_t is the output gate.

h_t is the hidden state.

x_t is the input at time step t.

W represents the weights.

b represents the biases.

σ represents the sigmoid activation function.

tanh is the hyperbolic tangent activation function.

2.10 Gated Recurrent Unit (GRU)

The last model used was the GRU which is a type of RNN model that handles sequential data. Similar to LSTM, it is designed to capture long-term patterns. It is a computationally efficient variant of the LSTM [26]. Some of what is different from LSTM will be discussed in this section. GRU also has a set of cells that process sequential data as well as having a hidden state and gates. The GRU cells have a hidden state and other components that will be explained. GRU takes sequential data input and divides it into time steps. Like LSTM, a GRU network can consist of multiple layers with each layer also containing cells.

A hidden state in a GRU network is maintained in a GRU cell and is updated at each time step. It captures relevant information from the past states and assists in modeling dependencies. Unlike LSTM, a GRU model has two gates only and they are the update gate and the reset gate. Both regulate the flow of information through the cell [27]. The reset gate determines what information from the prior state should be removed. Conversely, the update gate controls how much of the present hidden state should be updated with new information. Which is referred to as the candidate state. When the previous hidden state and the candidate state are combined, it is called the final hidden state [28]. Below is the simplified form of how the gates and the states are formulated:

Reset Gate (r_t): $r_t = \sigma(W_r \cdot [ht-1, x_t] + b_r)$ (10)

Update Gate (z_t): $z_t = \sigma(W_z \cdot [ht-1, x_t] + b_z)$ (11)

Candidate State (h'_t): $h'_t = tanh(W \cdot [r_t \cdot ht - 1, x_t] + b)$ (12)

Final Hidden State (h_t): $ht = (1 - zt) \cdot ht - 1 + zt \cdot h'_t$ (13)

Where:

r_t is the reset gate.

z_t is the reset gate.

h'_t is the candidate state.

ht is the final hidden state.

x_t is the input at time step *t*.

W represents the weights.

b represents the biases.

σ is the sigmoid function.

tanh is the hyperbolic tangent activation function.

LSTM and GRU are very similar in their architecture but there are few differences that make GRU computationally faster and less complex than the LSTM.

3. RESULTS and DISCUSSION

The Bayesian function HyperOpt was set up for each model. Some parameters were given their default values. Some parameters were manually tuned based on trial and error. All functions were set up to optimize based on the MSE score. For each model's parameters. For the Prophet model, when the changepoint_range parameter was set to 0.8 the model performed better than when it was automatically tuned. The n_estimators parameter is a key parameter in gradient boosting algorithms. It defines the number of base learners (Trees or models). While experimenting, it was found that manually setting the number of estimators yielded better performance in general which is why it was set to 200. The same case was with the number of RNN cells for the RNN models. When the number of cells was in the space parameter, the models tend to overfit and perform worse on the test set. It was noticed that the less cells the layer had, the better. As the number of the RNN cells decreases, the complexity and capacity of the model decrease. The capacity in this medium means the extent of the model's capability to capture the more complex relationships in the data. While that also is true, the model will also capture the noise and become sensitive to outliers which

in turn overfits the training data. A simply balanced model is often more robust and generalizes very well to new data. That is why it is important to take the time to find the right number of RNN cells for the model.

The model's performance was determined by evaluating the model on the test set which was 10% of the data. Initially, we evaluated the model's performance on the training set and then on the testing set to make sure that the model's performance on both data sets is somewhat similar. Because if the model is performing significantly worse on the testing set than the training set, it means that the model is not generalizing well to new data and is overfitting. Each model tremendously improved in performance after the hyperparameter optimization and after the techniques mentioned in the previous sections were applied. After testing and monitoring the model is performing in each iteration, the number of evaluations for the Bayesian optimization function was set to 50 for Prophet and XGBoost. While LSTM was 15 for and 20 for GRU. The reason is the algorithm converged to a solution before reaching the maximum number of evaluations. As for Epochs, the LSTM performed best 25 Epochs.

Table 2. Models' performance results

	MAE	MSE	MAPE	RMSE
Prophet	24.42	676.73	1.71%	26.04
XGBoost	14.55	332.25	0.94%	18.22
LSTM	19.08	589.31	1.32%	24.28

All models had comparable results. However, some models showed better performance in some respects. Prophet for instance had the highest MSE of all models with an error score of (676.73). Which indicates that the model predictions had the worst outliers of all other models' predictions. The MAE is the average absolute difference of the actual value, and the predicted value was (24.42) which is an acceptable score given the complexity of the data and the number of events. However, the MAPE score which is the percentage error was (1.71%) which is the highest of all models as well. The RMSE score for Prophet was also the highest of them all. Although Prophet's performance is acceptable, it was the worst performing model of the group. Figure 5 shows Prophet's prediction of the entire dataset.

The model seems to generalize appropriately well to unseen data by looking at the last period. That dramatic change in price towards the end of the year 2020 was challenging to overcome. Prophet's parameter, holiday data helped tremendously in assisting parameters like number of changepoints in mitigating the drastic changes like the spikes and dips over time. In comparison to Prophet, the GRU and LSTM models performed very closely to one another and to Prophet.

Although this is what the XGBoost model used to train and make predictions, it does not mean that the other models had the same order. For instance, the gas price seemed to have a significant importance when using Prophet. Finished goods inventory refers to the levels of finished goods inventory of heavy machinery. The lag features as seen show a significant influence on the model. Six lag features were created, but only the values of the first four weeks show on the plot. With the values of week five and six not even in the plot. The plot shows the ten most important features. The value of the 3rd previous week is the second most important. This result drove the decision of using 4 steps only for the time series generator used for LSTM and GRU because it highlighted the time dependency. Moreover, the rest of the features that showed less importance score despite having strong correlation with the plate were not excluded from the data.

The models generally performed very well and showed robustness in the ability to capture long-term and short-term dependencies. Table 2 shows how each model performed across all four metrics.

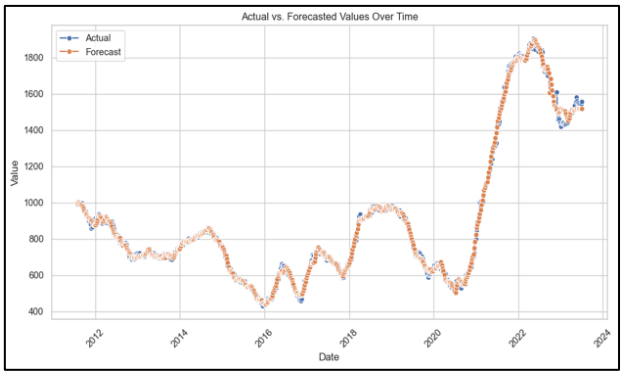

Figure 5. Forecast plot of entire dataset by prophet

GRU has a close MAE value to Prophet and LSTM has a close value of RMSE to Prophet. What is worth mentioning is that GRU is the second-best model when it comes to squared errors. It handled outliers better than LSTM and Prophet. That can be attributed to the complexity reduction implemented in GRU. Figures 6 and 7 show the forecast of the entire dataset by LSTM and GRU respectively.

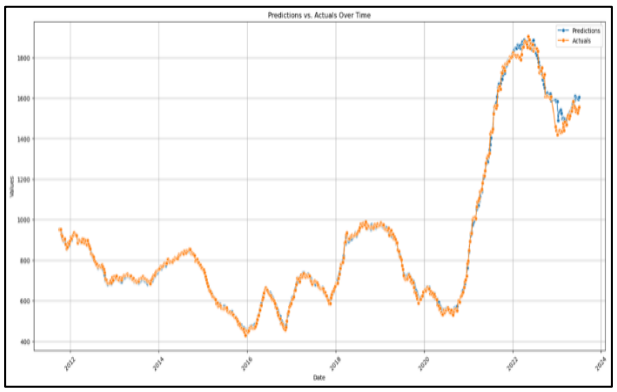

Figure 6. Forecast plot of entire dataset by LSTM

Both models excelled in their ability to capture long-term and short-term patterns and dependencies. As can be noticed towards the start of the test set, the model starts predicting less accurately yet enough to be acceptable. All three forecasts showed the models' ability to generalize well to new data and follow the trend of actual data. Given the difficulty of the test set where the test set starts towards the end of the price spike right before 2022, the models still managed to predict the trend correctly. In addition to keeping with that event, the models managed to capture the trend, seasonality, and the noise of that period.

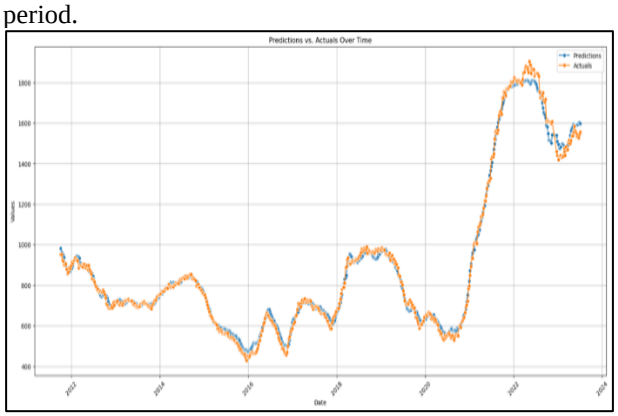

Figure 7. Forecast plot of entire dataset by GRU

That is an indication that the models can produce reliable long horizon forecasts. There's a tradeoff between the models. By looking at the performance metrics of each model, we can say that the GRU produces better long-term forecasts and LSTM produces better short-term forecasts. Nevertheless, the model that performed the best is XGBoost with the lowest errors of all models. The XGBoost models would be the best choice for long and short – term horizons. The MSE score for XGBoost is roughly half of the score of the other models. A low MAE score of (14.55) can be an indication of high accuracy in short – term forecasting. While an MSE score of (332.25) can be an indication of the model handling the outliers significantly well. Figure 7 shows the forecast plot produced by XGBoost. In general, all models performed well when it came to capturing patterns and dependencies and the accuracy of prediction.

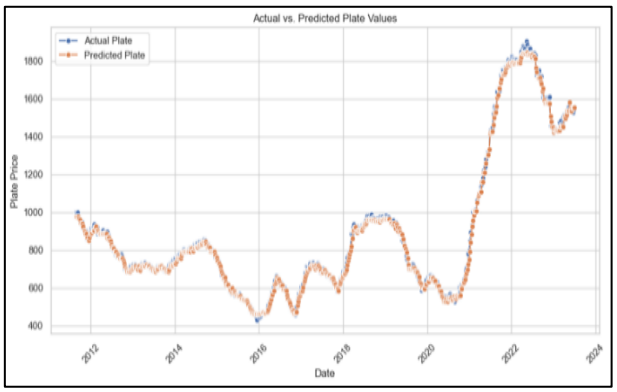

Figure 8. Forecast plot of entire dataset by XGBoost

4. CONCLUSION

The steel industry is one of the biggest manufacturing industries in today's world. It is also one of the most impactful industries economically. Some global economies depend heavily on steel production and consumption, making it an important commodity. The Commodity Research Unit (CRU) posts the price of steel products like the plate on a weekly basis. That price is a reliable price and is used globally. The plate price fluctuates usually in an unpredictable pattern. That is why the ability to incorporate machine learning algorithms in forecasting the price to make buying decisions could reap generous rewards. In this work, we proposed using data science techniques to forecast the price of steel plate. For this research, a dataset was built using the CRU plate price with a time frame of 7/27/2011 - 7/5/2023 and 6 factors that have predictive power over the plate price. Features were obtained from different reliable sources. The total inventory levels value had a 0.88 score which is the highest correlation score with the plate price across all features. Nonlinear testing was also done to determine relationships with the plate price like the Granger's Causality Test. We used feature engineering techniques like data scaling and creating lag features which helped in simplifying the data for the models and fully utilized the historical data for the target variable. Time Series Cross - Validation and Hyperparameter tuning was done to robustly train the model on capturing the patterns and dependencies, generalizing well to new data, and being less sensitive to outliers. The models used were Facebook's Prophet which is designed for forecasting tasks, Long Short - Term Memory (LSTM) is a deep learning Recurrent Neural Network (RNN) model that processes sequential data like text and time series, Gated Recurrent Unit (GRU) a is RNN architecture which handles sequential data, and XGBoost which is a decision tree-based model. Prophet had the highest MSE score of (676.73) while XGBoost had the lowest MSE score of (332.25). LSTM and GRU had somewhat close MSE scores of (589.31) and (506.48) respectively. XGBoost had the lowest MAE score of (14.55) while GRU and Prophet had the highest MAE of (24.42) and (24.75) in that order. All models had a low percentage error of less than 2% with XGBoost having the lowest MAPE of 0.94%. For the RMSE score

the models had a score ranging from 18 to 26 Which is satisfactory. The XGBoost model we developed performed extremely well and better than other models. The Prophet model had the highest error scores across all models. Overall, the diverse set of models displayed their capabilities, and the success of XGBoost underscores the effectiveness of advanced machine learning techniques in addressing the forecasting challenge at hand.

After building the models, the models should be put to another round of performance evaluation. This evaluation can be running long and short – term forecasts of after 7/5/2023 and comparing it with the readily available price data was posted in that same manner. The creation of a data frame with those dates that are to be forecasted would be done along with either using the same regressors data or retrieving new data for the regressors and retraining the models. It's a challenging task that needs to be performed to utilize the models' forecasting abilities. Another future work is acquiring data from the CRU like the lead time, the scrap index and other available data that may be helpful in our forecasting tasks. Along with acquiring data, some other time series machine learning algorithms should be experimented with. For example, DeepAR which was developed by Amazon. Temporal Fusion Transformer is another deep learning model that is gaining popularity in handling time series data. N-Beats is also another deep learning model designed specifically to handle time series data. All those models; proposed and potential, can produce predictions that can be aggregated using an ensemble model to produce more robust and accurate results.

DECLARATION of ETHICAL STANDARDS

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

AUTHORS' CONTRIBUTIONS

Mahmud ALSAIDEEN: Performed the experiments, analyze the results and wrote the manuscript.

Zeynep ERTEM: Performed the experiments, analyze the results and wrote the manuscript.

CONFLICT of INTEREST

There is no conflict of interest in this study.

5. REFERENCES

[1] T. Zhu, X. Wang, Y. Yu, C. Li, Q. Yao, and Y. Li, "Multi-process and multi-pollutant control technology for ultra-low emissions in the iron and steel industry," ***Journal of Environmental Sciences***, 123:83-95, (2023).

[2] John McLean, "History of Western Civilization II: Industrial Revolution," in *History of Western Civilization II*, ***Lumen Learning***, (2006).

[3] R. Waheeb, "Quality Control of Steel in Steel Iron Product Factory System," ***SSRN Electronic Journal***, (2023).

[4] X. Xu and Y. Zhang, "Regional steel price index forecasts with neural networks: evidence from east, south, north, central south, northeast, southwest, and northwest China," ***J Supercomput,*** 79: 13601:13619, (2023).

[5] J. Horák and M. Jannová, "Predicting the Oil Price Movement in Commodity Markets in Global Economic Meltdowns," ***Forecasting***, 5: 374:389, (2023).

[6] A. Page, "Covid-19 Effects on Commodity Pricing," ***East Carolina University***, (2023).

[7] X. Xu and Y. Zhang, "Regional steel price index forecasts with neural networks: evidence from east, south, north, central south, northeast, southwest, and northwest China," ***J Supercomput***, 79: 13601:13619, (2023).

[8] C. J. Haug and J. M. Drazen, "Artificial Intelligence and Machine Learning in Clinical Medicine, 2023," ***New England Journal of Medicine,*** 13: 1201-1208, (2023).

[9] S. Makridakis, E. Spiliotis, V. Assimakopoulos, A.-A. Semenoglou, G. Mulder, and K. Nikolopoulos, "Statistical, machine learning and deep learning forecasting methods: Comparisons and ways forward," ***Journal of the Operational Research Society,*** 74: 840-859, (2023).

[10] R. Ospina, J. A. M. Gondim, V. Leiva, and C. Castro, "An Overview of Forecast Analysis with ARIMA Models during the COVID-19 Pandemic: Methodology and Case Study in Brazil," ***Mathematics,*** 11(14), (2023).

[11] Statista, "World crude steel production from 2012 to 2022."

[12] K. F. Kroner, K. P. Kneafsey, and S. Claessens, "Forecasting volatility in commodity markets," ***J Forecast***, 14: 77-95, (1995).

[13] T. Xiong, C. Li, Y. Bao, Z. Hu, and L. Zhang, "A combination method for interval forecasting of agricultural commodity futures prices", ***Knowl Based Syst***, 77: 92-102, (2015).

[14] R. B. Palazzi, P. Maçaira, E. Meira, and M. C. Klotzle, "Forecasting commodity prices in Brazil through hybrid SSA-complex seasonality models", ***Production***, 33, (2023).

[15] N. Son and Y. Shin, "Short-and Medium-Term Electricity Consumption Forecasting Using Prophet and GRU", (2023).

[16] H. Ben Ameur, S. Boubaker, Z. Ftiti, W. Louhichi, and K. Tissaoui, "Forecasting commodity prices: empirical evidence using deep learning tools", ***Ann Oper Res***, (2023).

[17] K. E. ArunKumar, D. V Kalaga, Ch. M. S. Kumar, M. Kawaji, and T. M. Brenza, "Forecasting of COVID-19 using deep layer Recurrent Neural Networks (RNNs) with Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) cells," ***Chaos Solitons Fractals***, 146, (2021).

[18] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in ***Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining,*** (2016).

[19] H. Oukhouya and K. El Himdi, "Comparing Machine Learning Methods—SVR, XGBoost, LSTM, and MLP—For Forecasting the Moroccan Stock Market," ***Computer Sciences & Mathematics Forum***, MDPI, (2023).

[20] A. K. Gupta, V. Singh, P. Mathur, and C. M. Travieso-Gonzalez, "Prediction of COVID-19 pandemic measuring criteria using support vector machine, prophet and linear regression models in Indian

scenario," ***Journal of Interdisciplinary Mathematics***, 24: 89-108, (2021).

[21] T. Kriechbaumer, A. Angus, D. Parsons, and M. Rivas Casado, "An improved wavelet–ARIMA approach for forecasting metal prices", 39: 32-41, ***Resources Policy***, (2014).

[22] Y.-C. Chen, K. S. Rogoff, and B. Rossi, "Can Exchange Rates Forecast Commodity Prices?", ***Quarterly Journal of Economics***, 125: 1145-1194, (2010).

[23] V. I. Kontopoulou, A. D. Panagopoulos, I. Kakkos, and G. K. Matsopoulos, "A review of ARIMA vs. machine learning approaches for time series forecasting in data driven networks", 15(8), ***Future Internet,*** (2023).

[24] M. B. Priestley and T. S. Rao, "A test for non-stationarity of time-series," ***J R Stat Soc Series B Stat Methodol,*** 31: 140-149, (1969).

[25] A. M. Nyangarika, A. Y. Mikhaylov, and B. Tang, "Correlation of oil prices and gross domestic product in oil producing countries," ***International Journal of Energy Economics and Policy,*** 8: 42-48, (2018).

[26] L. Buitinck *et al.*, "API design for machine learning software: experiences from the scikit-learn project," *arXiv preprint arXiv:1309.0238*, (2013).

[27] C. Shivani, B. Anusha, B. Druvitha, and K. K. Swamy, "RNN-LSTM Model Based Forecasting of Cryptocurrency Prices Using Standard Scaler Transform," ***J. Crit. Rev***, 10: 144–158, (2022).

[28] S. Kamal, "An Analysis of Machine Learning Techniques for Economic Recession Prediction," (2021).