IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 1 March 2023, accepted 30 March 2023, date of publication 11 April 2023, date of current version 4 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3266275

RESEARCH ARTICLE

Time Series Forecasting and Modeling of Food Demand Supply Chain Based on Regressors Analysis

SANDEEP KUMAR PANDA[1], (Member, IEEE), AND SACHI NANDAN MOHANTY[2], (Senior Member, IEEE)

[1]Department of Artificial Intelligence and Data Science, Faculty of Science and Technology (IcfaiTech), The ICFAI Foundation for Higher Education (Deemed to be University), Hyderabad, Telangana 500029, India

[2]School of Computer Science and Engineering (SCOPE), VIT-AP University, Amaravati, Andhra Pradesh 522237, India

Corresponding author: Sandeep Kumar Panda (sandeeppanda@ifheindia.org)

ABSTRACT Accurate demand forecasting has become extremely important, particularly in the food industry, because many products have a short shelf life, and improper inventory management can result in significant waste and loss for the company. Several machine learning and deep learning techniques recently showed substantial improvements when handling time-dependent data. This paper takes the 'Food Demand Forecasting' dataset released by Genpact, compares the effect of various factors on demand, extracts the characteristic features with possible influence, and proposes a comparative study of seven regressors to forecast the number of orders. In this study, we used Random Forest Regressor, Gradient Boosting Regressor (GBR), Light Gradient Boosting Machine Regressor (LightGBM), Extreme Gradient Boosting Regressor (XGBoost), Cat Boost Regressor, Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM) in particular. The results demonstrate the potential of deep learning models in forecasting and highlight the superiority of LSTM over other algorithms. The Root Mean Squared Log Error(RMSLE), Root Mean Square Error(RMSE),Mean Average Percentage Error(MAPE), and Mean Average Error(MAE) reach 0.28, 18.83, 6.56%, and 14.18, respectively.

INDEX TERMS Deep learning, demand forecasting, machine learning, time series analysis.

I. INTRODUCTION

Due to the consumer's varying needs and increasing levels of competitiveness among companies, most companies in today's market are shifting their focus to demand forecasting for the effective demand-supply chain management. Demand forecasts are beyond the scope of any planning decisions, as they directly impact a company's profitability. Inaccurate approximation of demand can either cause too much inventory, which eventually results in a high risk of wastage and high costs to pay or too little inventory, leading to out-of-stocks which ultimately pushes the company's customers to seek services from its competitors. For these very reasons, the use of demand forecasting methods is one of the most fundamental components of the strategic planning and administration of a company's logistics.

Its importance becomes evident as its outcome is used by many subdivisions in the company: the financial department uses it to estimate costs, profit levels, and the required capital; the marketing department uses it to plan its course of action and analyze the impact of diverse marketing strategies on the volume of sales; the purchasing department may devise their plans of short- and long-term investments; and finally, the operations department can manage their plan of purchasing the necessary raw materials, machinery, and labor well in advance. It is, therefore, concordant that forecasts are beneficial, and their high accuracy has the potential to prove lucrative, improve demand-supply chain management, and reduce wastage.

Its importance becomes evident as its outcome is used by many subdivisions in the company: the financial department

The associate editor coordinating the review of this manuscript and approving it for publication was Sajid Ali.

uses it to estimate costs, profit levels, and the required capital; the marketing department uses it to plan their course of action and analyze the impact of diverse marketing strategies on the volume of sales; the purchasing department may devise their plans of short- and long-term investments; and finally, the operations department can manage their plan of purchasing the necessary raw materials, machinery, and labor well in advance. It is, therefore, concordant that forecasts are very useful, and their high accuracy has the potential to prove lucrative, improve demand-supply chain management, and reduce wastage.

A. MOTIVATION

Out of all the services, the biggest challenge faced by a meal delivery company is adjusting production and stock levels to minimize the loss of raw materials due to their short perishability, thus making the accuracy of the forecast of utmost importance [1]. Demand forecasting involves converting the time series problem to a regression problem. Currently, numerous models are available for both linear and nonlinear approaches to quantitative demand forecasting. The models adhere to various archetypes but have the same fundamental idea. Traditionally, forecasting models consisted of Linear Regression, Random Forest Regression, etc., suitable for short-term demand situations. But, several boosting algorithms like Gradient Boosting Regressor (GBR) [2], Light Gradient Boosting Machine Regressor (LightGBM) [3], Extreme Gradient Boosting Regressor (XGBoost) [4], and Cat Boost Regressor [5] perform better than the traditional algorithms when both numerical and categorical features are involved. Also, models like Long-Short Term Memory (LSTMs) [6] and Bidirectional LSTMs have good portability and application scenarios, as they can internally maintain the memory of the input, thus making them well suited for solving problems involving sequential data, such as a time series, and for long-term demand situations.

B. PROBLEM DEFINITION

The client is a meal delivery service with multiple locations. In these cities, they have several fulfillment centers for delivering meal orders to customers. As there is a high chance of food going to waste, the client needs a good model to predict how many orders will come in over the next few weeks so that suitable raw materials can be stocked.

Since most raw materials must be replenished on a weekly basis and are perishable, procurement planning is crucial. Second, accurate demand forecasts are very beneficial when staffing the center. The following data will be used to forecast demand for the subsequent 10 weeks:

1) Historical data of the number of sales of a particular meal for a specific center.
2) The meals' features include category, subcategory, current price, and discount.
3) Information about fulfillment centers, such as the region code, city code, etc.

C. NOVELTY OF THE WORK

As this is a forecasting problem, calculating lag features and Exponentially Weighted Moving Average (EWMA) for lag values play a vital role in improving prediction accuracy. Numerous experiments were carried out to determine the optimal parameters for each model. In the present article, we aim to compare 7 models: Random Forest Regressor, GBR, LightGBM, XGBoost, Cat Boost Regressor, LSTM, and Bi-LSTM, to analyze the adeptness of each of them.

D. CONTRIBUTIONS

The aim of this article is to predict the number of meals for the next 10 weeks using machine learning and the deep learning regressors mentioned above. Significant contributions of the work manifested in this paper include:

1) Traditional Random Forest Regressor is optimized and implemented as the baseline model.
2) Boosting algorithms like GBR, LightGBM, XGBoost and Cat Boost Regressor are applied since they are more adaptable to categorical and numerical features.
3) Only the lag and EWMA features are used with LSTMs and Bidirectional LSTMs because they are more reliable in analyzing historical data and forecasting using the same.
4) The Root Mean Squared Log Error (RMSLE), Root Mean Square Error (RMSE), Mean Average Percentage Error (MAPE) and Mean-Average Error (MAE) reach values 28.18, 18.83, 6.56%, and 14.18 respectively.

The rest of the article is organized as follows: in Section II, we present a literature survey of the forecasting methods used in time series analysis. In Section III, we describe and analyze the dataset and calculate the lag and EWMA features used for prediction. Next, the preprocessing techniques are described. Following that, in Section IV, we introduce the forecasting methods and the evaluation metrics as well. Next, in Section V, we compare the performance of the seven proposed models, and finally, in Section VI, we conclude the paper.

II. LITERATURE SURVEY

Forecasting demand of any commodity is one of the most important aspects to prevent wastage in any form. In fields like Agri - Food Supply Chains [7] where IoT technologies can be applied, it may not perform up to the mark unless there is an accurate forecast of the demand of the crop. J. Zheng et al. [8] addressed one of the challenges in Online Food Delivery (OFD) - food preparation timeby designing an iterated greedy algorithm with a decompositoon based strategy. Zhang et al. [10] predicted the forecast of wheat production on a country level using ensemble learning. I. Shah et al. [9], [10], [11], [12], [13], [14], [15] and Bibi et al. [16], have suggested many methods to predict the electricity demands and prices for various times, i.e., short term, medium term and long term as well. But, electricity demands may vary largely in restaurants or meal delivery canters as it in turn depends on the demand of their meals/products. The same is in the case of a factory producing pre-processed food products. Once the meal/product demand is known, it can give a huge increase in the accuracy of prediction of electricity demands and can also let the farmers estimate how much of the crop must be grown to meet the required demands. Hence, forecasting in the domain of meal demand prediction is of utmost importance. Statistical techniques like exponential smoothing [17], the Holt-Winters method [18], moving average, and the Auto-Regressive Integrated Moving Average (ARIMA) model [19] have traditionally been used in forecasting. Ramos et al. [20] used ARIMA and state-space models to predict the volume of upcoming commodity sales. SARIMA, or Seasonal Auto-Regressive Integrated Moving Average, was also frequently used. These techniques have a high adaptive capacity for dealing with linearity in problem solving. The field of time series analysis has gradually embraced more sophisticated techniques for dealing with both linearity and non-linearity in data. A decomposable time-series model with three main components—seasonality, trend, and holidays—is used by Facebook Prophet [21]. The best model, according to C. W. Chu and G. P. Zhang's comparative analysis of various linear and nonlinear models, was an Artificial Neural Network (ANN) constructed from deseasonalized time series data [22]. It was concluded that nonlinear models must be considered as well when forecasting sales, as they have the ability to represent nonlinear data by approximating them with a series of combinations of linear components. Chen et al. [23] considered the possibility of using a Back-Propagation Neural Network (BPNN) for forecasting. Apart from these, the use of Evolutionary Neural Networks (ENN) [24] and Extreme Learning Machine (ELM) [25] were also investigated in this field. In the field of food forecasting, Tarallo et al. [26] demonstrated the added benefits of applying machine learning models over conventional forecasting models. Krishna et al. [27] compared and contrasted various machine learning techniques like Linear Regression, Polynomial Regression, Lasso Regression, Ridge Regression, AdaBoost, Gradient Boost and XGBoost in the field of forecasting and deduced that boosting algorithms perform better than the rest. The authors of [28] have compared and concluded that CatBoost performs better than traditional machine learning methods in predicting sales. More recently, the use of Recurrent Neural Networks (RNNs), Long-Short Term Memory (LSTM), and Bi-directional Long-Short Term Memory (Bi-LSTM) has come into the spotlight due to their ability to model nonlinear functions and capture long-term time-dependent patterns. RNN forecasting models were subjected to a thorough empirical study by Hewamalage et al. [29]. RNNs were found to be capable of directly modeling seasonality if the series had uniform seasonal patterns; otherwise, deseasonalization must be performed. The LSTM network is a special kind of RNN model that is used to model the connections between longer input and output data. Xu and Wang [30] forecast sales based on univariate time series and the LSTM model. Bi-LSTM performance on multivariate time series data was examined by the authors of [31], who came to the conclusion that because most time series data have nonlinear trends, neural network based analysis and prediction methods are superior to statistical methods.

A summary of the above discussion is tabulated in Table 1.

TABLE 1. Comparison study of related works.

Author	Domain study	Model
S.Yadav et al. [7]	Food demand	IoT-based
J. Zheng et al. [8]	Food demand	Iterated greedy algorithm
Zhang et al. [10]	Food demand	Ensemble models
I. Shah et al. [11]	Electricity demand	Auto Regressive models
F Lisi et al.[12]	Electricity demand	regressive models
I. Shah et al. [13]	Electricity demand	Auto regression models
Shah et al. [14]	Electricity demand	Auto Regressive
Shah et al. [15]	Electricity demand	Auto regressive models
N.Bibi et al. [16]	Electricity demand	Ensemble machine learning
E. S. Gardner Jr. [17]	-	Exponential smoothing
Holt et al. [18]	-	Holt-Winters method
Box et al. [19]	-	ARIMA
Ramos P et al. [20]	Fashion	regressive models
Taylor et al. [21]	-	Prophet
C. W. Chu et al. [22]	Retail sales	machine learning
C.Y.Chen et al. [23]	Food demand	Cluster and Forecast Model
K.-F. Au et al. [24]	Fashion	ENN, SARIMA and ANN
Z.-L. Sun et al. [25]	Fashion	ELM and ANN
E Tarallo et al. [26]	Multidisciplinary	machine learning
A. Krishna et al. [27]	Food demand	regression models
J.Ding et al. [28]	Retail store	linear regression, SVM
Hewamalage et al. [29]	Multidisciplinary	deep learning
Xu et al. [30]	Food demand	deep learning
Kim et al. [31]	Food demand	Bi-LSTM model

III. DATA ANALYSIS AND PREPROCESSING

In this section, we describe our proposed methods by first introducing the existing and calculated features and then the preprocessing techniques.

A. DATASET DESCRIPTION

The "Food Demand Forecasting" dataset [22] released by Genpact, an American professional services firm, comprises 145 weeks' worth of weekly orders for 50 distinct meals. The data is divided into three files with a combined total of about 4,50,000 entries and 15 features, which are explained as follows:

1) Weekly Demand Data: includes the historical data of the number of sales of a particular meal in a particular center. The following details are included:
   a) id: unique order ID associated.
   b) week: week number (ranges from 1 to 145).
   c) meal_id: unique ID associated with each meal.
   d) center_id: unique ID associated with each fulfillment center.
   e) checkout_price: final price of the meal after all the discount, taxes, and delivery charges.
   f) base_price: base price of the meal as written in the menu.
   g) emailer_for_promotion: describes whether email is sent for the promotion of the meal or not (0/1 implies no/yes).

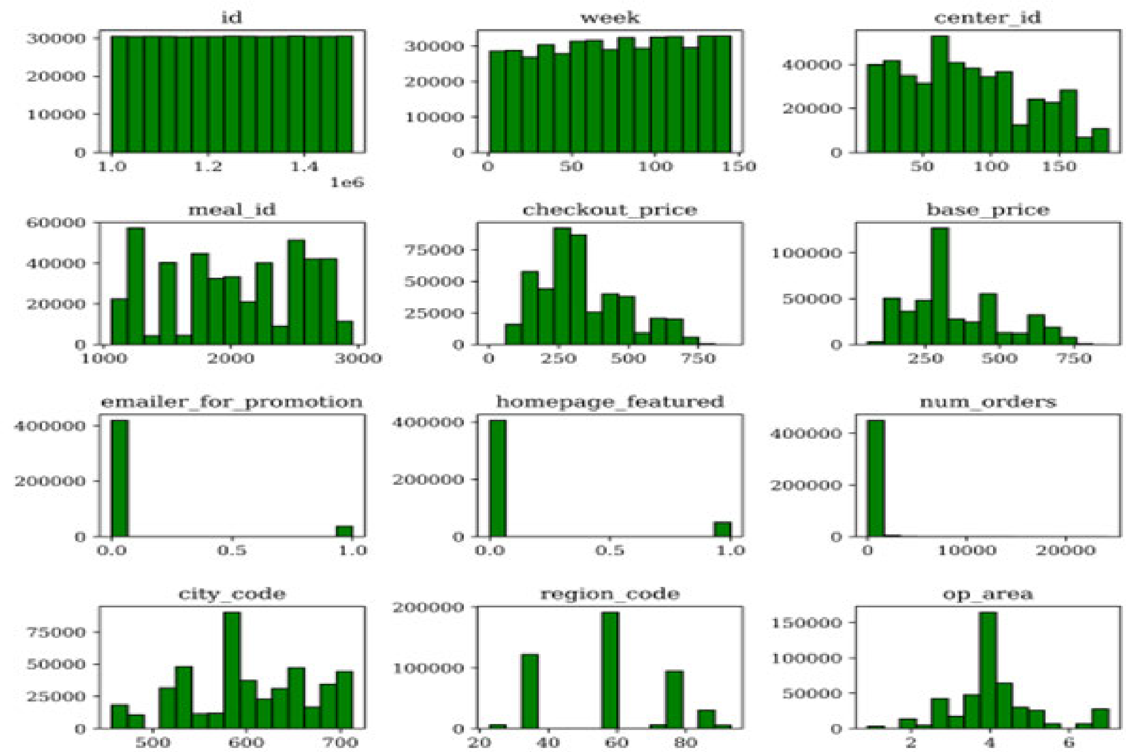

FIGURE 1. Spread of distribution of each feature.

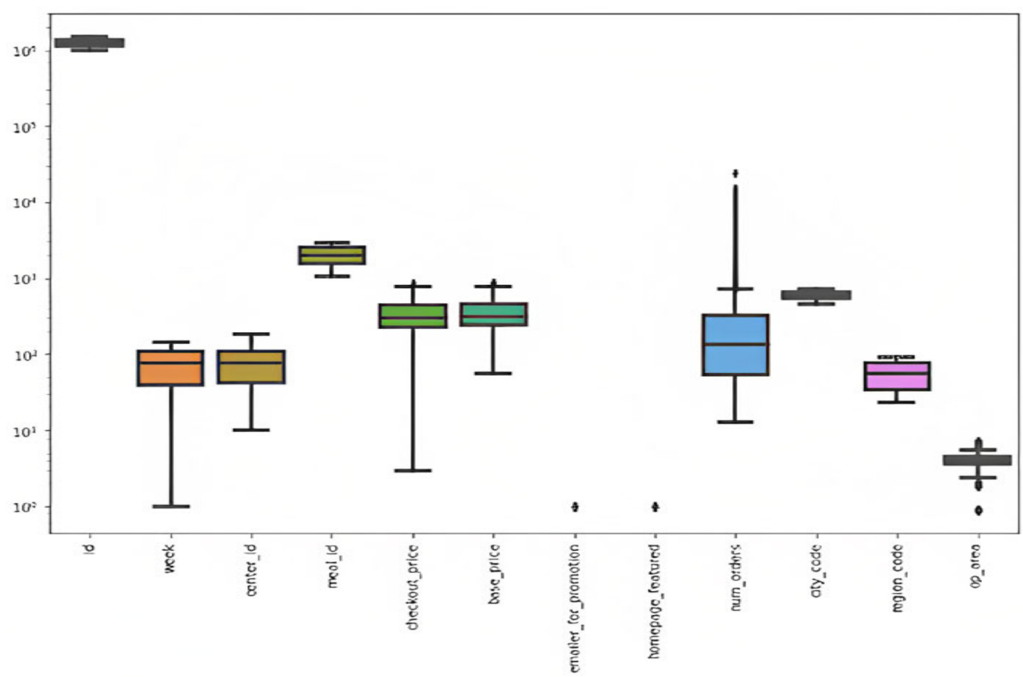

FIGURE 2. Box plot for all features.

h) homepage_featured: describes whether the meal was featured on the homepage of the website or not (0/1 implies no/yes).

i) num_orders: number of orders received.

2) fulfilment_center_info.csv: contains details for every fulfillment center. It includes the following:

a) center_id: ID of every fulfillment center.

b) op_area: area of service for each center (in km2).

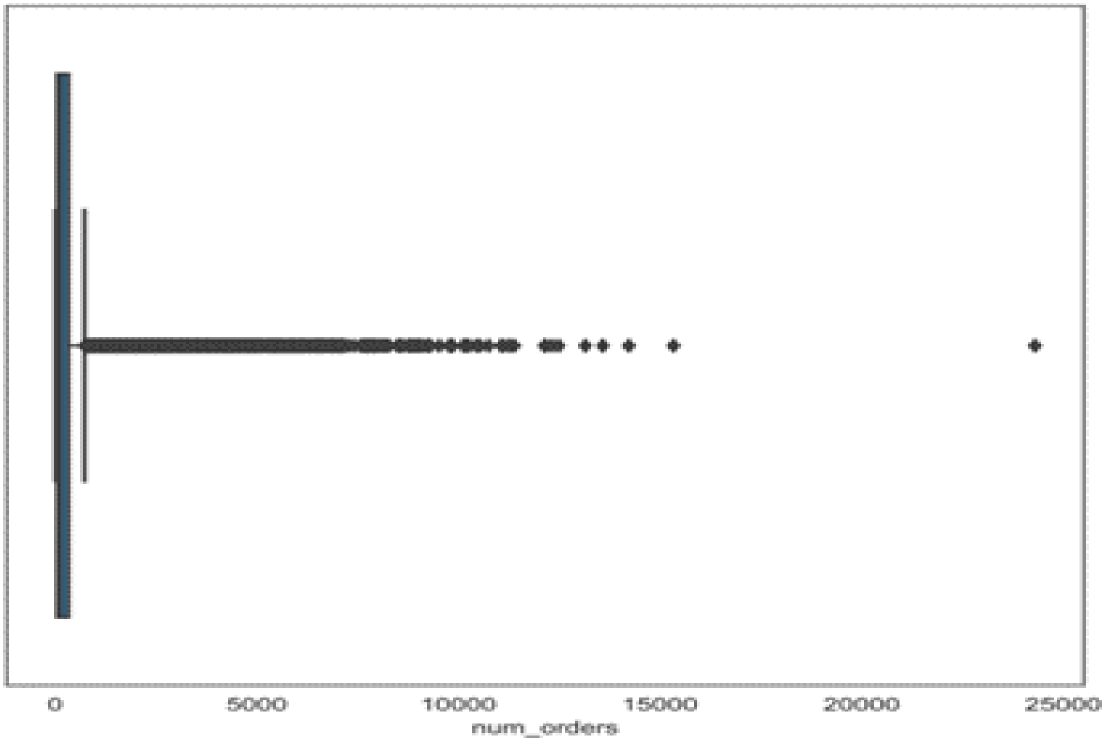

FIGURE 3. **Box plot for num_ orders.**

c) city_code: pin code associated with each city.
d) center_type: type of the center (type_A, type_B, type_C).
e) region_code: unique code associated with each region.

3) meal_info.csv: contains information of each fulfillment center. Specifically, it consists of:

a) meal_id: ID of each meal.
b) category: type of meal (beverages, snacks, soups, etc).
c) cuisine: Indian, Italian, etc.

B. DATASET ANALYSIS

In this section we will analyze the dataset. First, a single variable analysis is carried out, and then a multi variable analysis.

1) UNIVARIATE ANALYSIS

This section aims at analyzing each feature individually. Both categorical and numerical features are analyzed thoroughly.

Histograms are plotted to see the spread of the distribution for each feature. As observed for "num_orders" in Fig. 1, most of the values for this feature are concentrated around 0-1000, but the spread of the distribution is greater than 20000 which implies that there might be a possibility of an outlier. That can be made clear by plotting box plots for each of the features or only the feature of concern.

As seen in Fig. 2 and Fig. 3, there is only one outlier in the number of orders that may be a result of miswriting and is therefore removed as it may hinder the further analysis.

Fig.4 shows the total number of orders received by each type of center. Type_A centers received the highest number of orders, whereas Type_C centers received the least. In Fig. 4, it was found that center Type_A has the highest number of orders, but, according to Fig. 5, it is seen that center 13 of Type_B has the highest number of orders. The reason behind it is that center Type_A has the most centers as shown in Fig. 6.

Fig. 7 displays the number of orders received in each region. Region 56 received an astonishingly high number of orders when compared to other regions. According to Fig. 8, Beverages are the food category with the highest number of orders, and Biryani is the food category with the least number of orders. Meal ID 2290 has received the most orders, as shown in Fig. 9. The number of orders for various meal IDs does not differ significantly in many cases. Fig. 10 shows that City - 590 has the most orders with 18.5M, which is nearly 10M more than the city with the second-highest number of orders, City 526, with 8.6M.

Fig. 11 depicts the total number of orders received per week. It is observed that the highest number of orders

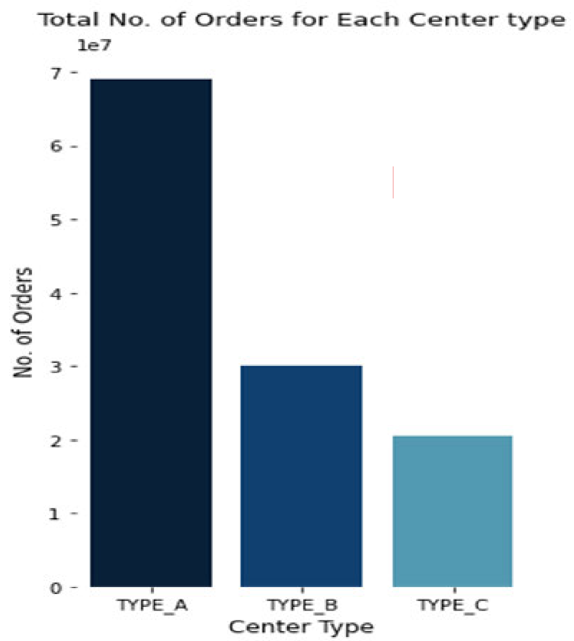

FIGURE 4. **Number of orders received by each center type.**

were received in week 48 and the lowest number of orders in week 62.

2) MULTIVARIATE ANALYSIS

In this section we aim at analyzing two or more features together. This helps in determining the dependency of one feature on another.

One of the most efficient methods for analyzing potential connections or correlations between data attributes is to use a pair-wise correlation matrix and visualize it as a heat-map.

Fig. 12 makes it clear that it is very easy to spot potential attributes that strongly correlate with one another because the gradients in the heat map change depending on how strong the correlation is. Scatter plots among attributes of interest are another way to visualize the same thing, but they can only be plotted between numerical values, as shown in Fig. 13.

Fig. 12 and 13 prove that base_price and checkout_price have a high correlation, so checkout_price must be removed. The individual distributions for the attributes can be seen using joint plots, which are particularly useful for looking for patterns and relationships. In Fig. 14, a new feature is created: a discount, which is the difference between the base price and the checkout price. We attempted to determine whether or not there is a relationship between the discount and the number of orders. Surprisingly, there is no strong link between them.

C. FEATURE ENGINEERING

The choice of lag characteristics, which hold the key to increasing the prediction's accuracy, is a crucial stage in any time series prediction. Target values from earlier periods are essentially what lagged characteristics are. For our problem, sales of a meal from the previous 10 weeks are utilized to forecast sales over the following 10 weeks.

However, only computing lag characteristics will result in an equal weighting of all the data from the previous week. In real life, recent data is typically given higher weight as it is thought to be more pertinent for future projections. Calculating the Exponentially Weighted Moving Average (EWMA) [23] for the target value over the past 10-15 weeks solves this problem as it assigns weights to the value. More recent data points are given more weight, which increases their relevance for forecasts in the future. EWMA for each target variable was calculated as follows:

$$EWMA(w) = \alpha * x(w) + (1 - \alpha) * EWMA(w - 1) \qquad (1)$$

where EWMA(w) means EWMA for week w and α is a parameter that decides the importance of the current observation. In the present scenario, α was set to 0.5.

Other features calculated include the rank of a meal based on its base price, meal count features in relation to a meal's category, cuisine, center, city, and region, as well as the highest, lowest, and average prices of the meal in a city and a region.

We compute a few more features from the existing ones, which are mentioned as follows:

1) base_price_max: maximum base price for a particular meal, irrespective of the week, center, city, or region.
2) base_price_mean: the average base price for a particular meal, irrespective of the week, center, city or region.
3) base_price_min: the minimum base price for a particular meal, irrespective of the week, center, city or region.
4) center_cat_count: a count of orders belonging to a particular category in a particular center.
5) center_cat_price_rank: rank of the price of a particular category of meals in a particular center.
6) center_cat_week_count: a count of orders belonging to a particular category in a particular center in a week.
7) center_cui_count: a count of orders belonging to a particular cuisine in a particular center.
8) center_price_rank: rank of the price of a particular meal in a particular center.
9) center_week_count: a count of orders in a particular center in the given week.
10) center_week_price_rank: rank of price of a particular meal in a particular center in a particular week.
11) city_meal_week_count: a total count of a certain meal sold in a city in a particular week.
12) meal_count: a count of a certain meal sold in all the weeks across all centers and regions.
13) meal_city_price_rank: rank of price of a particular meal in a particular city.
14) meal_price_max: maximum checkout price of the meal irrespective of the week,center,city or region.
15) meal_price_mean: average checkout price of the meal irrespective of the week,center,city or region.

these features marked in red are all calculated based on the 'num_orders' column which is the target variable.
This is wrong because in the real world, this data will be unavailable for the test data we will be making predictions on.
Hence, these new features marked in red will be impossible to create for the test dataset. So this is wrong.

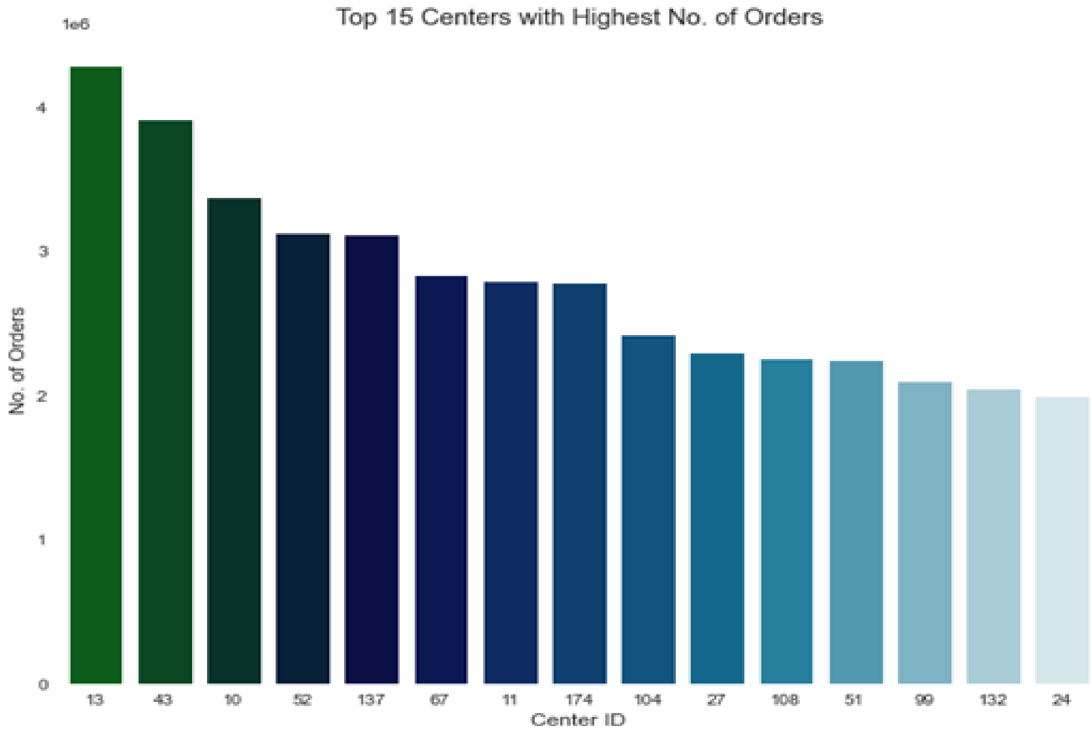

FIGURE 5. Top 15 centers with the highest number of orders.

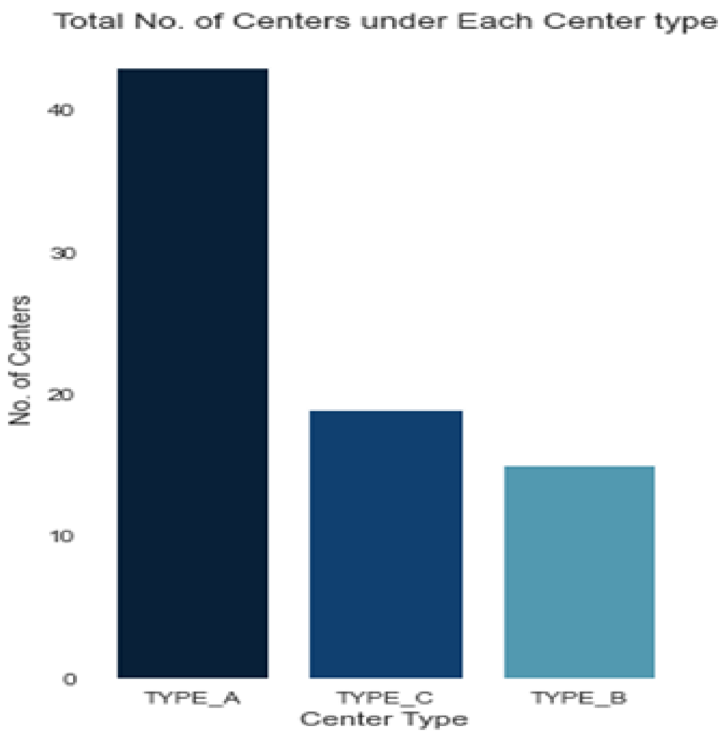

FIGURE 6. Number of centers under each center type.

16) meal_price_min: minimum checkout price of the meal irrespective of the week,center,city or region.
17) meal_price_rank: rank of price of a particular meal.
18) meal_region_price_rank: rank of price of a particular meal in a particular region.
19) meal_week_count: a total count of a certain meal sold in the given week.

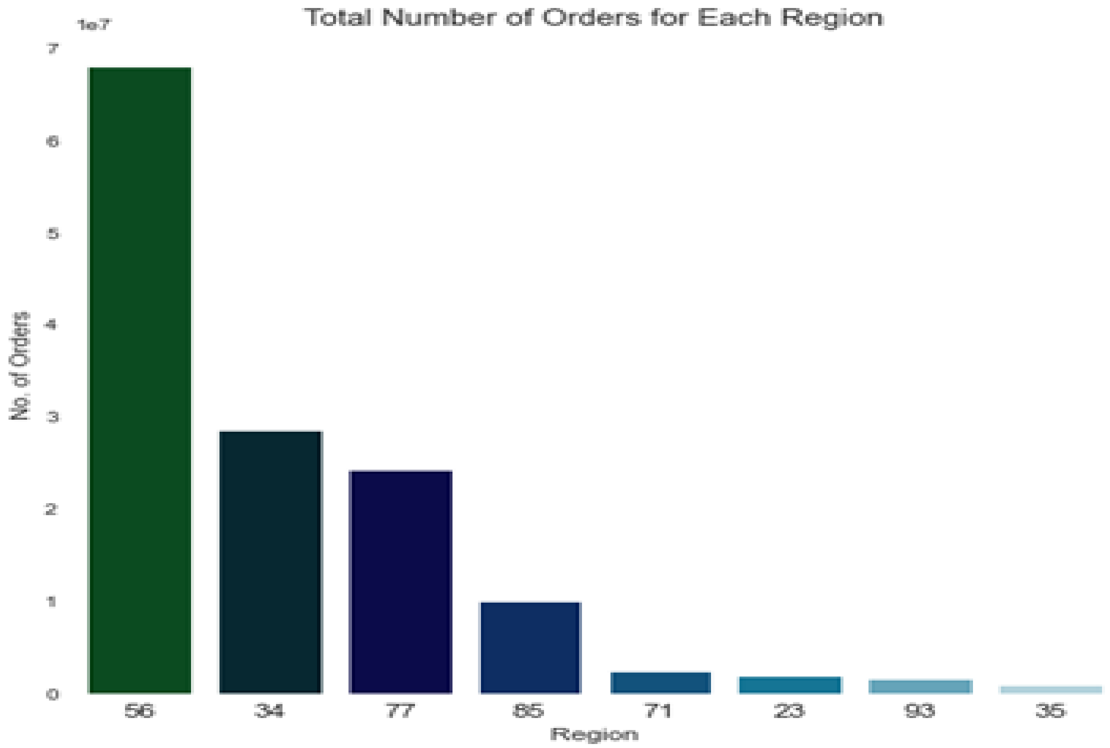

FIGURE 7. Number of orders for each region.

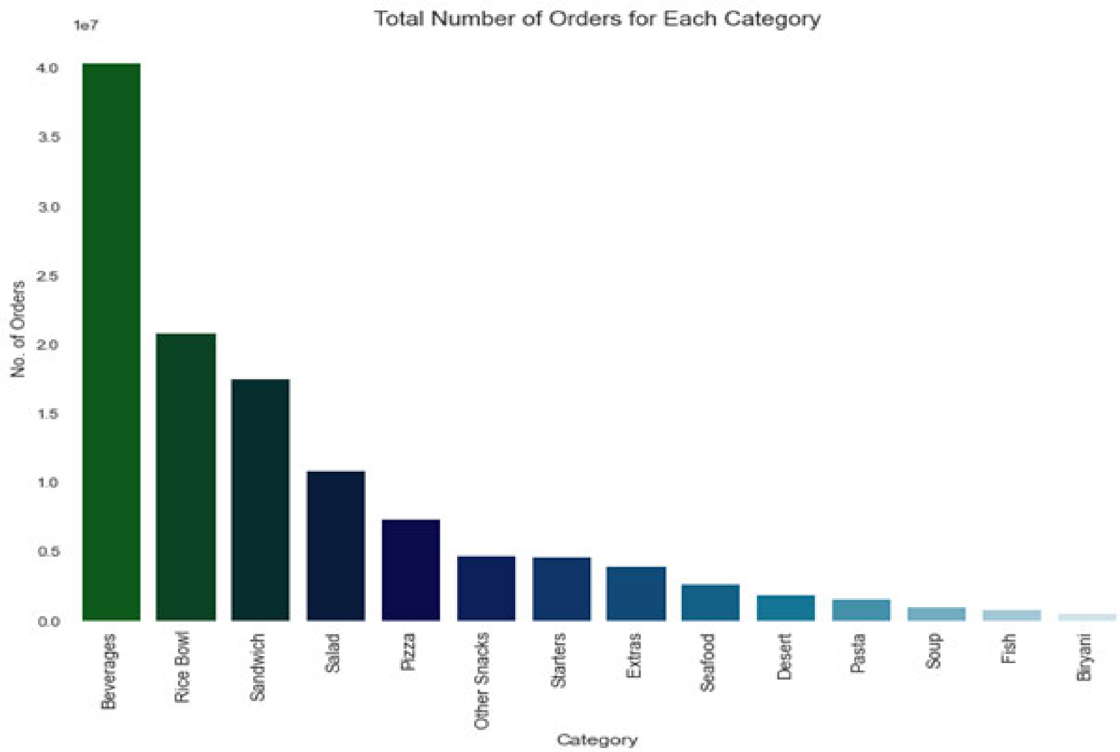

FIGURE 8. Number of orders for each category.

20) meal_week_price_rank: rank of price of a particular meal in a particular week.

21) region_meal_count: a region's total sales of a particular meal.

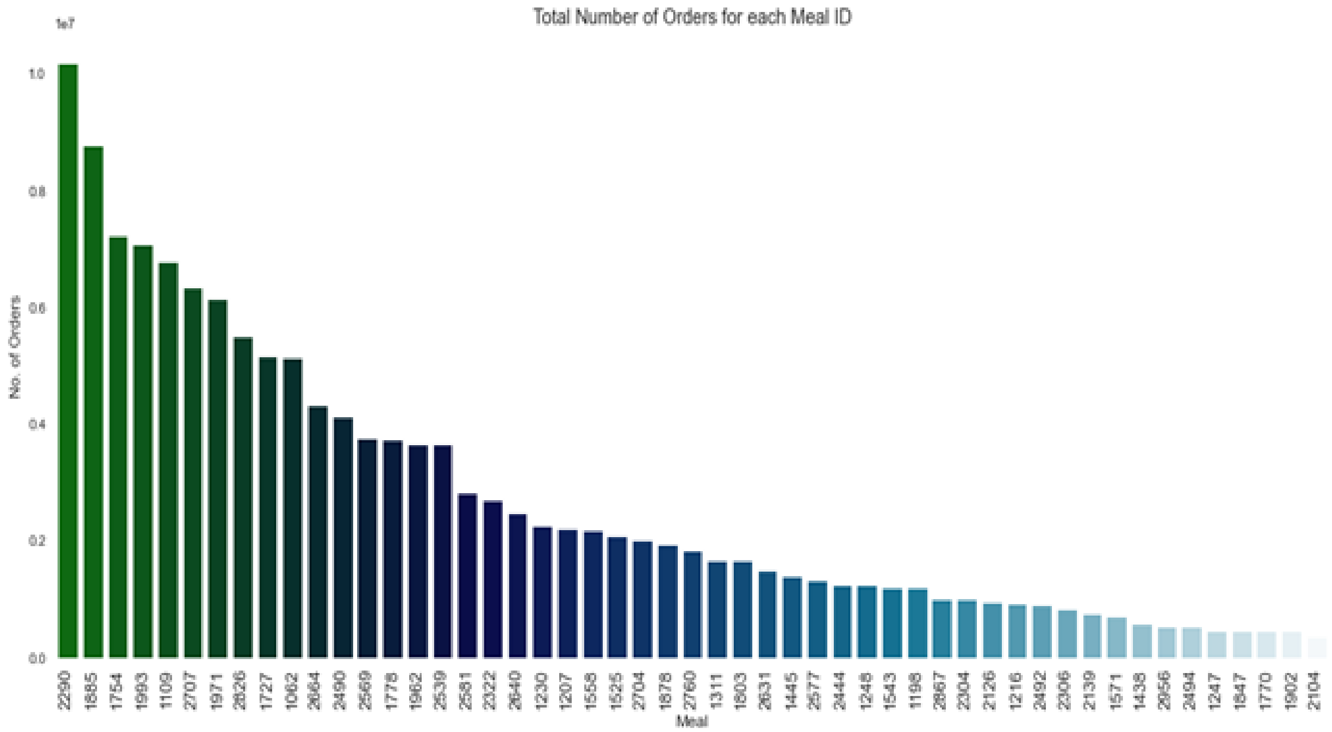

FIGURE 9. Number of orders for each meal ID.

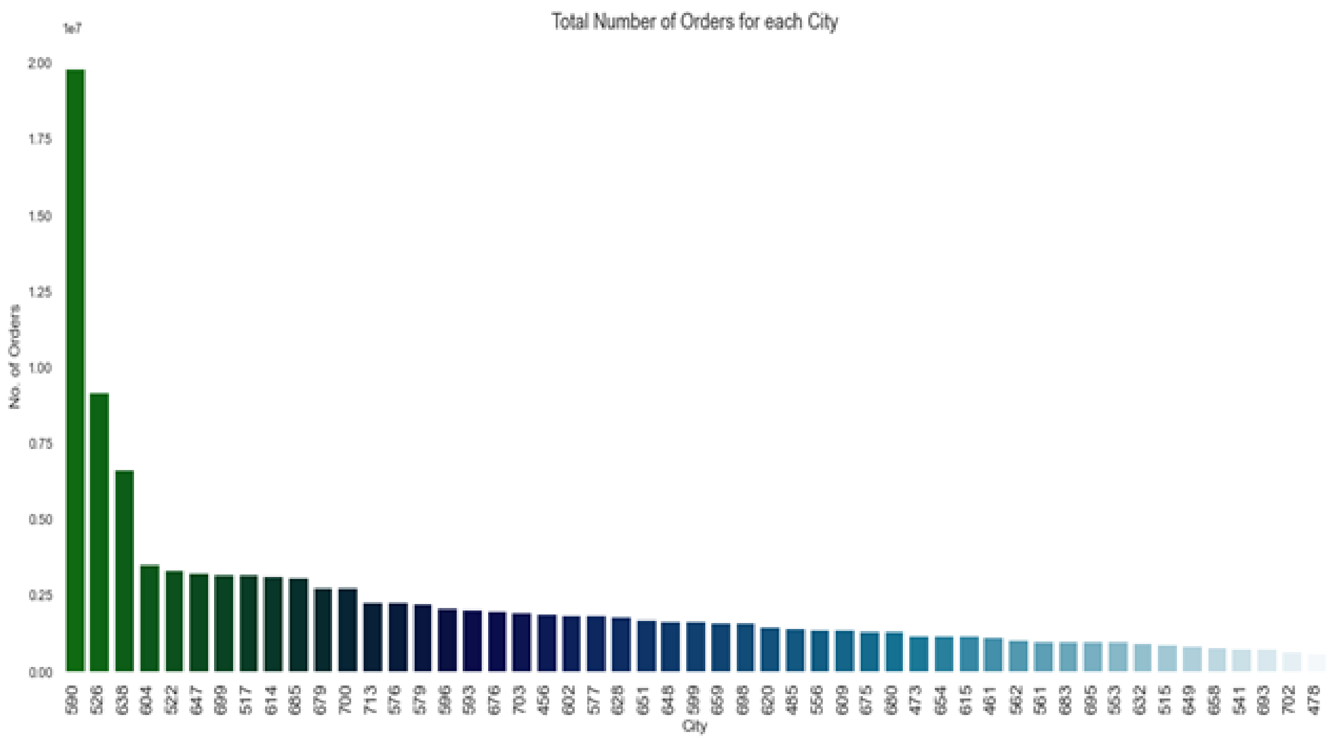

FIGURE 10. Number of orders for each city.

22) region_meal_week_count: the total number of a specific meal sold in a region during a specific week.

23) type_meal_week_count: the total amount of a specific meal sold in a center over the course of a particular week.

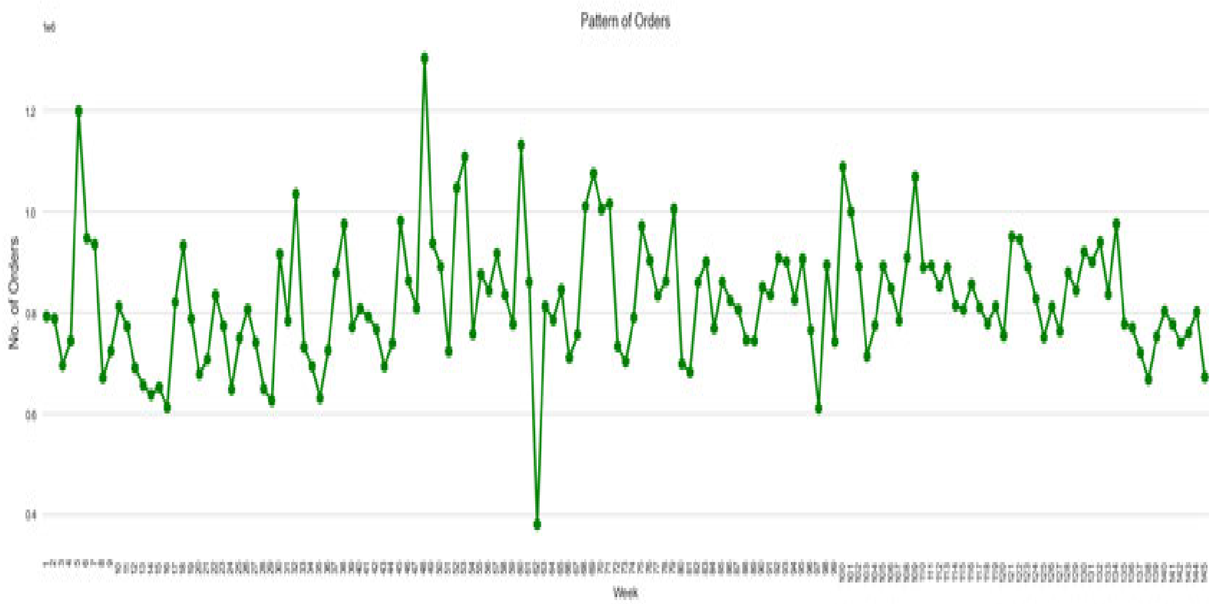

FIGURE 11. **Total number of orders received every week.**

D. PRE-PROCESSING TECHNIQUES

1) NUMERICAL FEATURES

The values for each feature are not on the same scale, and most of the numerical features in the dataset are heavily right-skewed. Skewness can be decreased by applying the quantile or power transformation. Both were used and compared, and Fig. 15 shows one such example where it is evident that the quantile transformation reduced skewness more than the power transformation in our case. However, this method merely transforms the distribution to a normal distribution. Since the data already has a normal distribution, standardizing the data is a better option for scaling than normalizing it.

2) CATEGORICAL FEATURES

Label Encoding and One-Hot Encoding are two different techniques with the same purpose of converting a categorical feature into a numerical feature. In Label Encoding, each label of a categorical feature is assigned a unique integer based on alphabetical ordering. In contrast, additional features in One-Hot Encoding are created based on the number of unique values in the categorical feature. This consumes a lot of memory and energy and is only useful when there are few features. We will use Label Encoding in this situation because there are so many features already available.

try onehot

IV. METHODOLOGY AND METRICS

In this section, we aim to describe the forecasting methods used in our study. All the models are trained and validated on the data from weeks 1-135, and they are tested on the data from weeks 136-145. The hardware included a 12 GB NVIDIA GeForce RTX 3060 GPU and a CPU with 64 GB of memory.This section's main goal is to list all of the models that were applied before comparing the metrics that each of them produced. The full architectural layout of our current study is shown in Fig. 16.

A. MACHINE LEARNING MODELS

In our study, we used 5 machine learning models: Random Forest Regressor, GBR, XGB, LightGBM, and CatBoost. The Random Forest Regressor is used as the baseline model for our investigation. The hyper-parameters for each model are tuned using the GridSearchCV method to enhance performance.

1) RANDOM FOREST

The random forest, an ensemble-learning algorithm, can be used for time series analysis. A random forest, as its name suggests, is made up of several decision trees. The results obtained from each of the decision trees are brought together and an estimation process is performed. A step-by-step procedure describing the construction of each of the trees Td, where d = 1, 2, ... D assuming, we grow D trees, is described as follows [24]:

1) The process begins with randomly drawing in observations with replacement thus creating a bootstrapped data. Using this data, several decision trees are constructed by choosing a set of random variables. This process is termed as bagging. Not all samples are included in the bootstrapped data. Such samples are named as out-of-bag-samples.
2) Now that the trees are constructed, they are modified by recursively repeating the following steps:

YOU COULD HAVE TRIED BINARY ENCODER, LABELENCODING MAKES THE MODELS THINK THE FEATURES HAVE AN ORDINAL RELATIONSHIP LIKE BIG, BIGGER, BIGGEST.

WHEREAS THIS IS NOT THE CASE FOR MOST CATEGORICAL FEATURES. USING BINARY ENCODER CREATES LESS COLUMNS

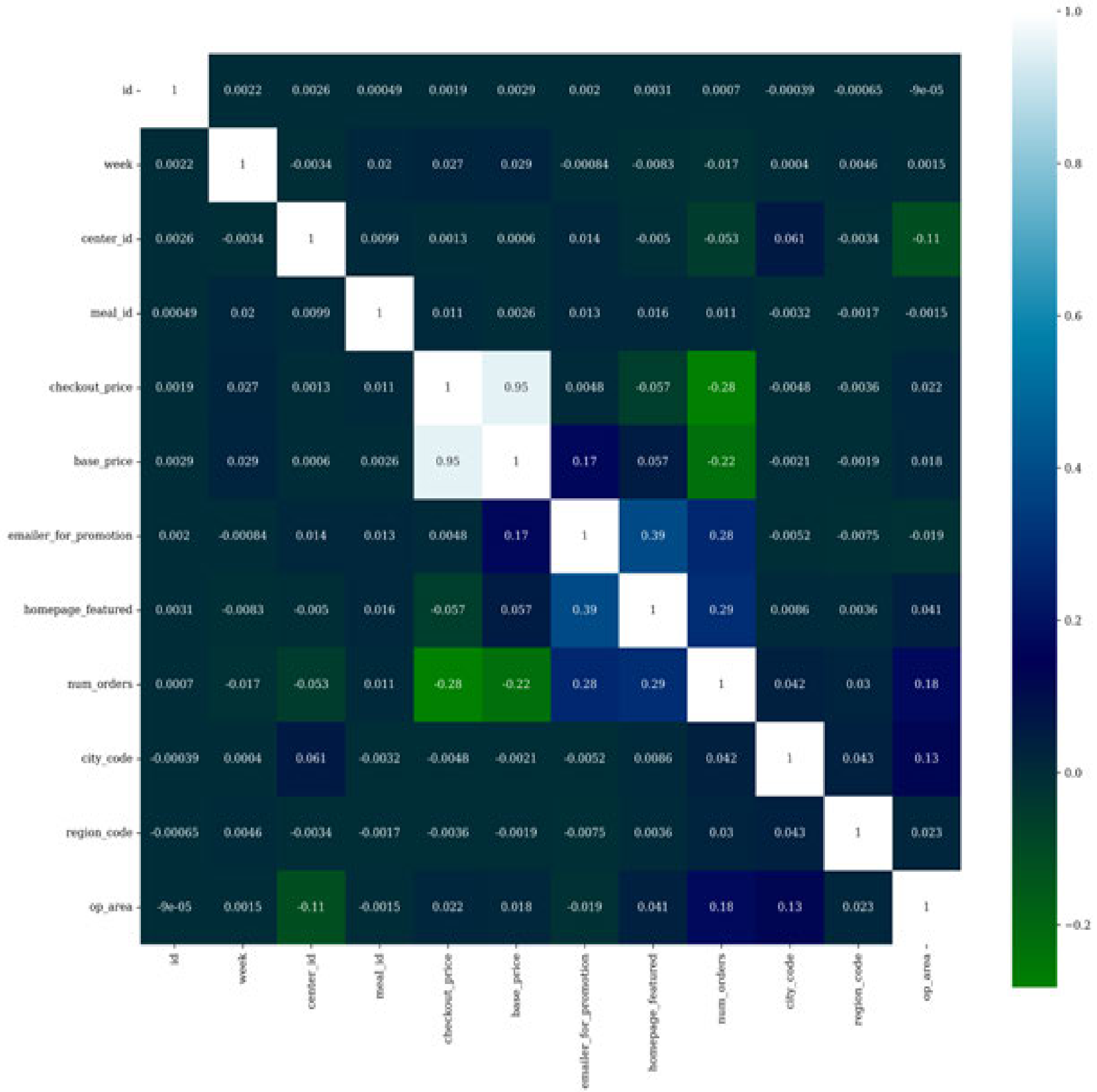

FIGURE 12. Heat-map for the given dataset.

a) Random subspace of k features, such that k ≤ l is chosen.
b) Variance reduction is used to select the best variable and split from the set of k features. This is defined as:

$$Var(N) = \frac{1}{|N|}\sum_{y \in N}(y - \bar{y})^2 \tag{2}$$

$$Reduction = Var(N) - \left(\frac{|N_l|}{|N|}Var(N_l) + \frac{|N_r|}{|N|}Var(N_r)\right) \tag{3}$$

where, N is the node, y is the target variable, $\bar{y}$ is the predicted value, N_l are the left nodes and N_r are the right nodes. Two child nodes are created from the original node.

c) Two child nodes are created from the original node.

3) The final prediction of the random forest is calculated by aggregating the results from the D trees as follows:

$$\bar{y}_t = \frac{1}{D}\sum_{d=1}^{D} T_d x(t) \tag{4}$$

We have used this model from the scikit-learn library [25] in Python. After tuning the parameters using GridSearchCV, the maximum depth of the tree is 9, and the maximum number of features used for construction of each decision tree is the square root of the total number of features. 150 trees are constructed in total. Moreover, the minimum number of samples required to split an internal node and the minimum

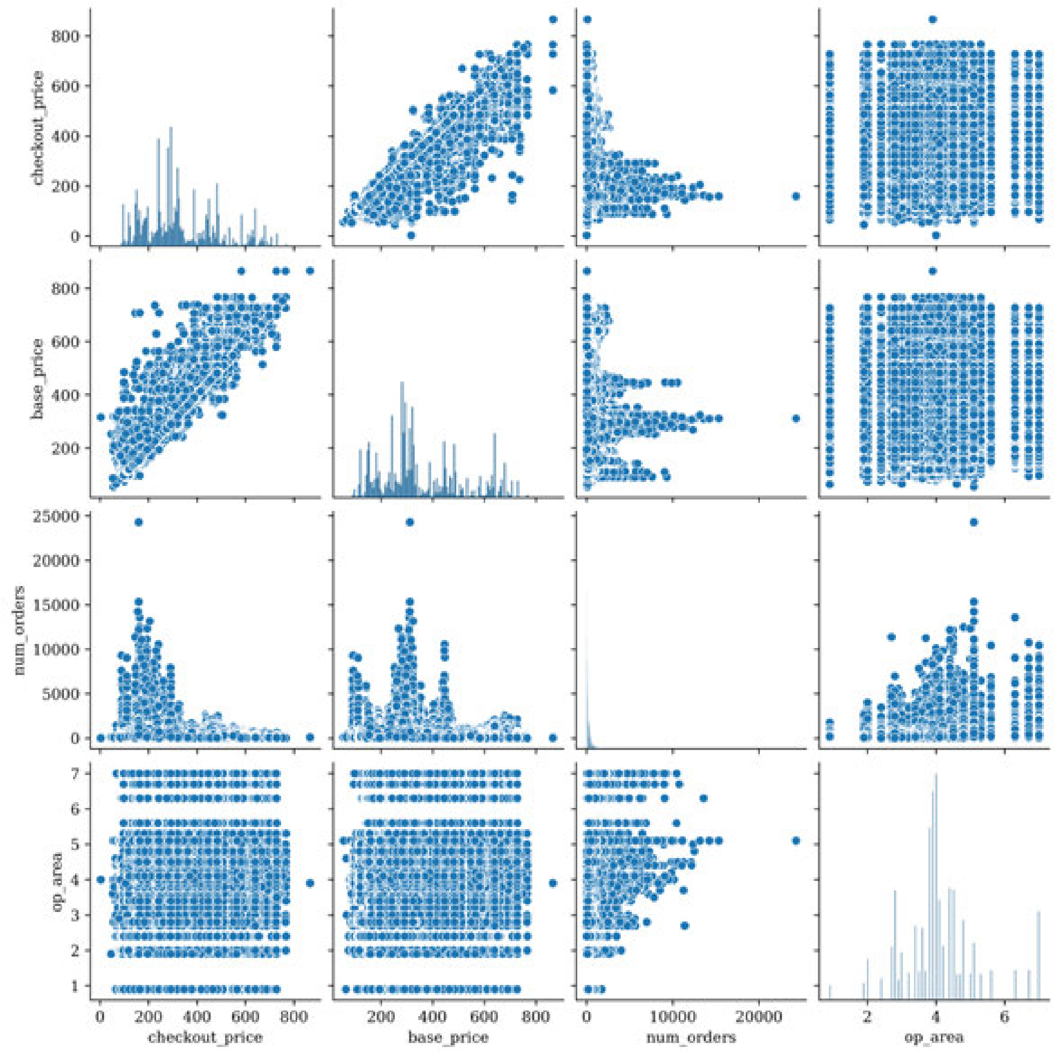

FIGURE 13. Scatter plot for numerical features.

number of samples required to be a leaf node are 2 and 3, respectively.

In general, cases with both categorical and numerical features can be handled well by the Random Forest.Moreover, it can handle datasets with higher dimensions. Even though Random Forest provides these benefits, this model is only used as a naive baseline model as it fails when extrapolation outside the training data is needed thus making it unsuitable for time series analysis.

2) GRADIENT BOOSTING MACHINE

GBMs are based on the idea of combining a group of weak learner regression trees with gradient boosting regression to create robust forecasting models. With this approach, the error rate of poorly learned models is decreased. Given the following input data, a brief description of the algorithm is provided:

$$T = (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \tag{5}$$

Loss function is, therefore:

$$L(y, f(x)) \tag{6}$$

1) Firstly, a constant value (fo(x)) that minimizes the loss function is used to initialize the GBM.

$$f_o(x) \ = argmin_c \sum_{i=1}^{N} L(y_i, c) \tag{7}$$

2) Then, in each iterative process (from 1 to M), the negative gradient of the loss function is assumed to be the residual value (r_{mi}) in the current model

$$r_{mi} = -\left[\frac{\delta L(y, f(x_i))}{\delta(f(x_i))}\right]_{f(x)=f_{(m-1)}(x)} \tag{8}$$

3) After that, a new regression tree (c_{mj}) is trained to fit the new residual value and get the leaf node area of the m^{th} tree (R_{mj}).

$$c_{mj} = argmin_c \sum_{x_i \in R_{mj}} L(y_i, f_{m-1}(x_i) + c) \tag{9}$$

FIGURE 14. Scatter plot of discount versus num_orders.

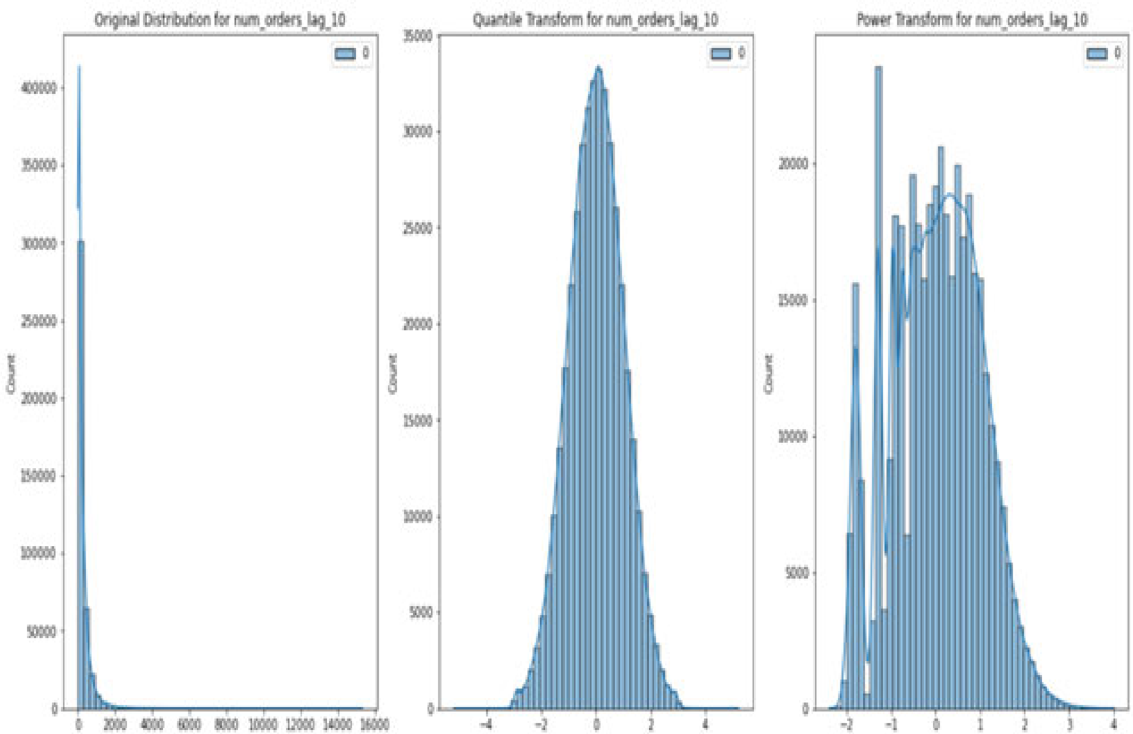

FIGURE 15. Comparison of original distribution, quantile transform and power transform of one of the numerical features.

4) Finally, the model is updated as follows:

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} c_{mj} I \tag{10}$$

These steps are repeated until the maximum number of iterations are reached. The final model is obtained by [33]:

$$\hat{f}(x) = f_m(x) = \sum_{m=1}^{M} \sum_{j=1}^{J} c_{mj} I \tag{11}$$

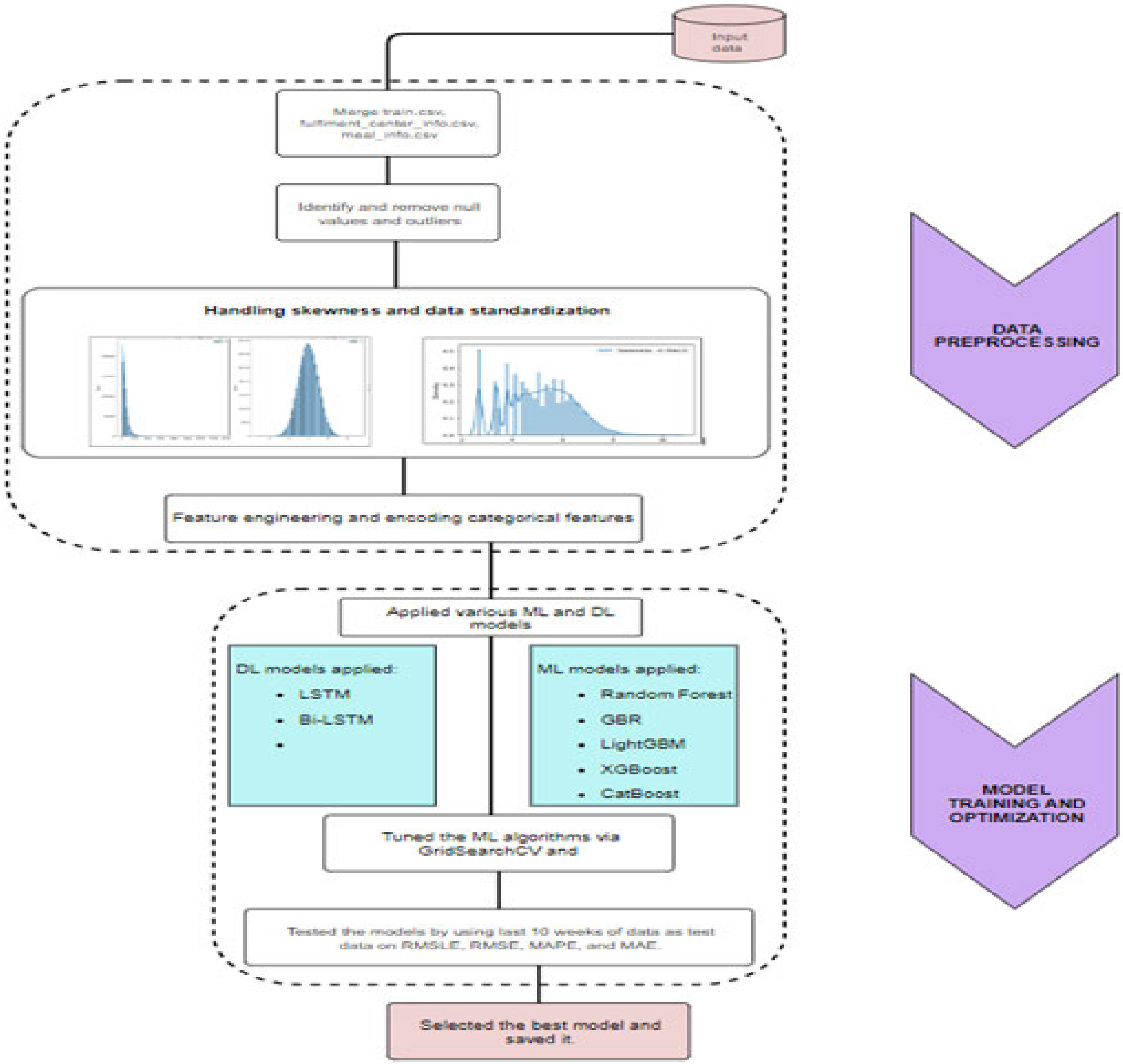

FIGURE 16. Architectural layout of the current study.

This model is borrowed from the scikit-learn library [32] python. After tuning the parameters, the maximum number of iterations are found to be 100, the maximum depth for each tree is set to 9, and the minimum number of samples required to split each node is 5. The default learning rate of 0.1 and loss function 'squared_error' are optimal for this case.

3) LIGHT GBM

The Light GBM algorithm is an optimization of the original Gradient Boosting Algorithm. It operates on the same principles as the boosting algorithm, but unlike its parent algorithm, it grows the decision trees using a leaf-wise policy. This technique divides the tree only along the best nodes that can minimize the loss the most [27]. Light GBM differs from the other algorithms in that it has modules that can deal with missing data, support parallelism, and support distributed computing.

We have used the Light GBM library in python and optimized the hyper-parameters with GridSearchCV. The default 'gbdt' boosting type was considered to be optimal along with learning rate of 0.13. The maximum depth of each tree was set to 8 and the L2 regularization term on weights was 3. The model was trained for 150 iterations with early stopping enabled. The model stopped after 100 iterations.

4) XGBoost

XGBoost is widely used because it efficiently cuts down on running time through parallel and distributed computing, as well as handling NaN values in the dataset. In order to minimize loss, XGBoost is made to define and optimize the objective function. The objective function is defined as

TABLE 2. Hyper-parameters for all the machine learning models applied.

Model	Parameter values modified by GridSearchCV
Random Forest Regressor (baseline model)	max_depth=9 max_features = 'sqrt' n_estimators =150 min_samples_leaf = 3
GBR	max_depth = 9 n_estimators = 100 min_samples_split = 5 loss = 'squared error'
LightGBM	max_depth = 8 learning_rate =0.13 n_estimators = 150 reg_lambda = 3
XGBoost	max_depth = 9 n_estimators = 100 learning_rate =0.1 tree_method = 'exact'
CatBoost	iterations = 2000 learning_rate = 0.01 max_depth = 9 l2_leaf_reg = 8 loss_function = 'RMSE'

follows [28]:

$$obj(\theta) = \sum_{i=1}^{N} L(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{12}$$

where, $f_k \in F$ and L is the loss function that calculates the deviation between the actual value y_i and the predicted value $\hat{y}_i$, Ω is the regularization function to prevent over-fitting. And f_k is a regression tree from the set of all possible regression trees F. We have employed the XGBoost package to use the XGBoost regressor in python. The tuned parameters include maximum depth of each tree set to 9, learning rate as 0.1, number of iterations as 100, and the default tree construction algorithm - exact- which follows a greedy method to reduce loss is selected.

5) CatBoost REGRESSOR

CatBoost is a machine learning algorithm that employs gradient boosting on decision trees. CatBoost gains significantly more efficiency in parameter tuning by using balanced trees to predict labels. To improve the model's robustness, it also builds an oblivious tree model on randomly shuffled training data using the greedy TS method [29]. The symmetry of the oblivious tree prevents the model from overfitting on one side, which prevents it from overfitting the training set. CatBoost employs an efficient method based on ordered boosting [29] that produces models that run faster and with greater accuracy with less memory storage.

The maximum depth of each tree is set to 9, the L2 regularization term was set to 8. The maximum number of iterations is the default value of 2000, and the learning rate was set to 0.01. The 'RMSE' loss function was used with early stopping enabled.

The tuned parameters for each of the models discussed above are briefly shown in Table 1.

B. DEEP LEARNING REGRESSORS

We have used LSTM and Bidirectional LSTM for our study. We have considered only the week number, lag features, and base price of a meal as input features for both LSTM and Bidirectional LSTM. As a result, we will have data from weeks 13-145 for training, validation, and testing of each meal. The data from weeks 13-125 (113 weeks of data) was used for training, weeks 126-135 (10 weeks of data) for model validation, and weeks 136-145 (10 weeks of data) for model testing.

1) LONG-SHORT TERM MEMORY (LSTM)

The LSTM network is chosen over the RNN network because it solves the issue of long-term dependencies that the RNN network cannot address and prevents the RNN network's gradient explosion. The four primary structural elements of the LSTM memory unit are the input gate, output gate, forget gate, and cell state (C(t)). The memory information at time t is stored in the cell state, which runs continuously along the entire chain and ensures that the information is unchanged during transmission. The gate structure will selectively add or remove information from the cell state. The following is the overall framework of the LSTM network [30]:

Forget gate:

$$f(t) = \sigma(w(f) * [s(t-1), x(t)] + b(f) \tag{13}$$

Input gate:

$$i(t) = \sigma(w(i) * [s(t-1), x(t)] + b(i) \tag{14}$$

$$C(t) = tanh(w(c) * [s(t-1), x(t)] + b(c) \tag{15}$$

Output gate:

$$o(t) = \sigma(w(o) * [s(t-1), x(t)] + b(o) \tag{16}$$

$$f(t) = o(t) * tanh(C(t)) \tag{17}$$

where *tanh* is the hyperbolic tangent activation function; w(f)and w(i)are the weight matrices for the forget and input gates, respectively; w(c) and w(o)are the weight matrices for the generated candidate joining information; and b(f), b(i), b(i), and b(o) are the bias terms for each layer. The input of an LSTM layer has the shape of (num_timesteps, num_features), which was (10, 13) in our case, meaning that each input sample has 10 timestamps and each timestamp has 13 features. For our study, we have used 3 layers of LSTM, each consisting of an LSTM cell, a ReLU layer, and to prevent the model from overfitting, a dropout layer with a dropout value of 0.25, as depicted in Table 2. The size of the memory units were set to 64, 32, and 16 respectively. The default recurrent activation - the hard-sigmoid function was used. We used mean squared error as the loss function and Adam as the optimizer. The number of epochs and batch size are 300 and 16, respectively. In order to avoid training the model on patterns it does not yet have access to, shuffle is set to False. This is necessary because the model should only be trained up to the point of data visibility. For instance, in our case, at timestep 20, the model should only be trained

TABLE 3. Proposed network architecture for LSTM.

Layer	Input size
LSTM layer 1	10 x 13
ReLU layer 1	10 x 64
Dropout layer 1	10 x 64
LSTM layer 2	10 x 64
ReLU layer 2	10 x 32
Dropout layer 2	10 x 32
LSTM layer 3	10 x 32
ReLU layer 3	16
Dropout layer 3	16
Dense layer 1	16

TABLE 4. Proposed network architecture for Bi-LSTM.

Layer	Input size
Bi-LSTM layer 1	10 x 13
Bi-LSTM layer 2	10 x 64
Dense layer 1	32

with data from 13 to 20 and not be made visible to data from 21 to 125.

2) BIDIRECTIONAL LSTM (BI-LSTM)

Only informational predictions from the front to the back can be made by the one-way LSTM network. The Bi-LSTM network performs forward calculation and backward calculation synchronously by applying two LSTMs to the input data, which can memorize the overall trend and fluctuation of demand. After the data is passed to the input layer, the model will generate new latent vectors $h(t)_f$ and $h(t)_b$ from the former latent vector $h(t-1)_f$ and $h(t+1)_b$ respectively. Then the output result $S(t)$ is calculated using $h(t)_f$ and $h(t)_b$ [31]. The formulae are as follows:

$$h(t)_f = LSTM(x(t), h(t-1)_f) \tag{18}$$
$$h(t)_b = LSTM(x(t), h(t+1)_b) \tag{19}$$
$$S(t) = tanh(w_{hs(f)} * h(t)_f + w_{hs(b)} * h(t)_b + b_y) \tag{20}$$

where $w_{hs(f)}$ and $w_{hs(b)}$ are weight matrices and b_y is the bias.

The input of an Bi-LSTM layer has the shape of (num_timesteps, num_features), which was (10, 13) in our case, meaning that each input sample has 10 timestamps and each timestamp has 13 features. For our study, we have used 2 layers of Bi-LSTM and the number of units in each layer was set to 32 and 16, respectively. We used tanh as the recurrent activation function rather than sigmoid. The recurrent dropout value was set to 0.25. Table 3 gives a brief overview of the Bi-LSTM architecture used. We used mean squared error as the loss function and Adam as the optimizer. The number of epochs and batch size are 50 and 16, respectively.

C. EVALUATION METRICS

Root Mean Square Error (RMSE), mean absolute percentage error (MAPE), mean absolute error (MAE), and root mean squared log error (RMSLE) were the metrics used to assess the forecasting models.

TABLE 5. A comparison of metrics produced by all the models applied.

Models / Metrics	RMSLE	RMSE	MAPE	MAE
Random Forest	0.35	228.92	10.822	115.90
Gradient Boost	0.31	226.46	10.03	123.48
Light Boost	0.31	194.68	10.08	104.49
XGB	0.30	205.31	9.97	106.17
CatBoost	0.29	186.68	9.76	99.40
LSTM	0.28	18.83	6.56	14.18
Bi-LSTM	0.37	31.88	12.8	28.33

$$RMSE = \frac{1}{N}\sqrt{\sum_{j=1}^{N}(y(actual)_j - y(pred)_j)^2} \tag{21}$$

$$MAPE = \frac{1}{N}\sum_{j=1}^{N}\frac{|y(actual)_j - y(pred)_j|}{y(actual_j)} \tag{22}$$

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|y(actual)_j - y(pred)_j| \tag{23}$$

$$RMSLE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\{log(y(pred)_j+1) - log(y(actual)_j+1)\}^2} \tag{24}$$

where N is the total number of observations, y(actual)is the actual value, and y(predicted) is the value predicted by the model.

V. EXPERIMENTAL RESULTS AND DISCUSSION

This section will compare the forecasting performances of the forecasting models, namely: Random Forest Regressor, Gradient Boost Regressor, Light Boost Regressor, XGBoost Regressor, CatBoost Regressor, LSTM, and Bi-Directional LSTM. All the models were evaluated on each meal, and Table.. 4 shows the average metrics taken. The results from the table attest to the LSTM model's superiority over other models. The RMSE value of the LSTM model, which represents the average separation between the observed and predicted data points, is 18.83, and the RMSLE value, which represents the average ratio of the predicted value to the actual value, is 0.28. These results indicate that the model is capable of making accurate predictions. In this case, the MAPE value indicates that the mean absolute percent error between the model's predicted sales and the actual sales is 6.56%, and the MAE value indicates that there will typically be a 28.33 difference between the predicted value and the actual value.

Fig. 17 shows the scatter plot between the actual and the predicted values for each of the model applied. The results of the scatter plot align with those in Table 4. LSTM performs better at forecasting even for those values which are scarce.

Figs. 18-24 demonstrate the line graph plotted between actual and predicted values for all the models applied. It is seen in Fig. 23 that LSTM not only follows the trend in

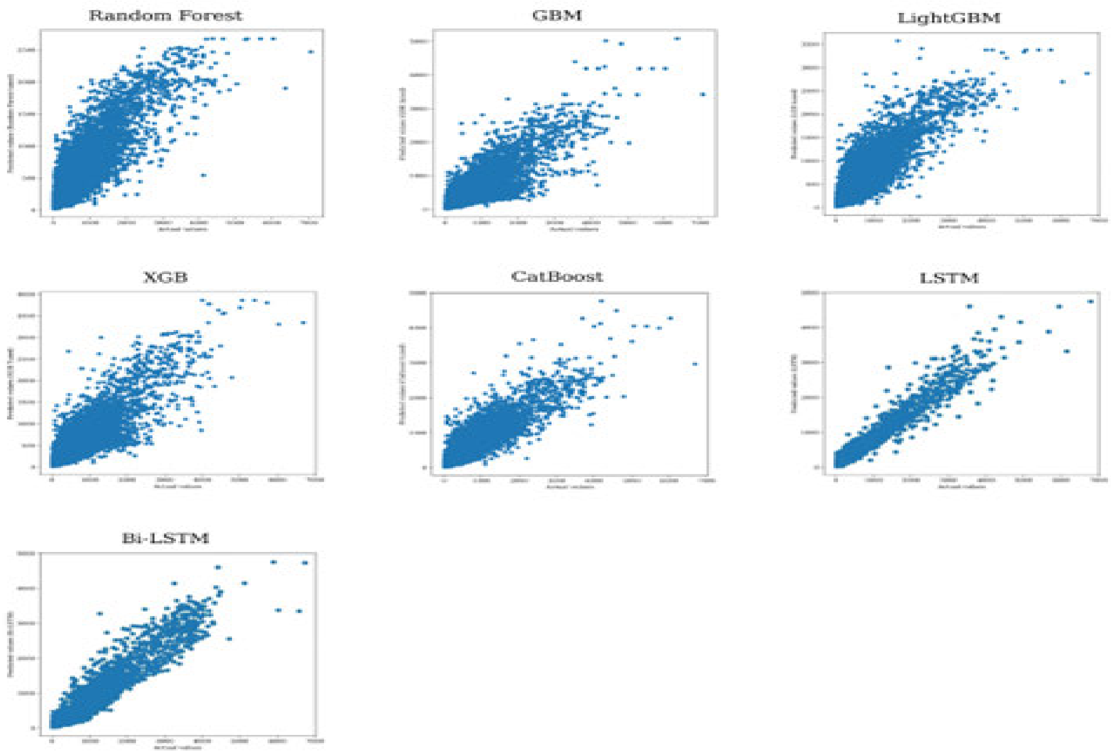

FIGURE 17. Scatter plot of actual versus predicted values of all the models.

TABLE 6. -values for Diebold and Mariano test.

Models	RFR	GBR	Light GBM	XGB	Cat Boost	LSTM	Bi-LSTM
RFR	-	0.00	0.00	0.00	0.00	0.04	0.32
GBR	0.99	-	0.90	0.00	0.42	0.08	0.02
Light GBM	0.99	0.10	-	0.00	0.20	0.03	0.00
XGB	0.99	0.99	0.99	-	0.00	0.03	0.09
CatBoost	0.99	0.58	0.80	0.99	-	0.03	0.02
LSTM	0.96	0.92	0.97	0.97	0.97	-	0.98
Bi-LSTM	0.68	0.98	0.99	0.91	0.98	0.02	-

TABLE 7. p-values of one way ANOVA.

Models	RMSLE	RMSE	MAPE	MAE
RFR + GBR + LightGBM + XGB + CatBoost + LSTM + Bi-LSTM	1	<0.0001	<0.0001	<0.0001

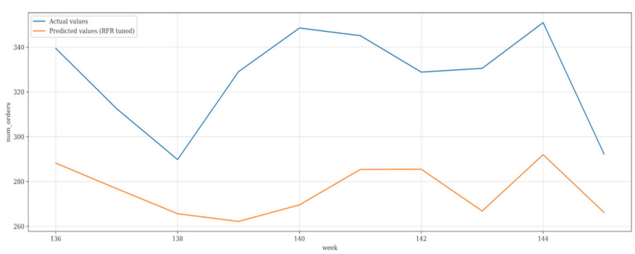

FIGURE 18. Actual versus predicted values for random forest.

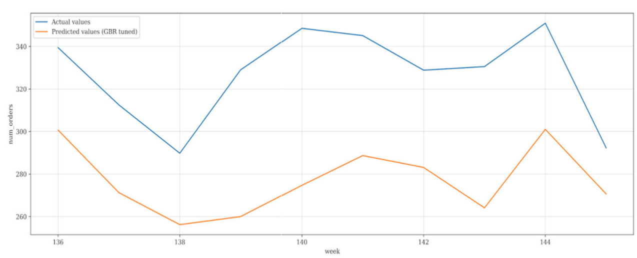

FIGURE 19. Actual versus predicted values for GBM.

the data, it predicts the values more accurately. Figs. 23 and 24 also highlight the potential of deep learning models over machine learning models for time series analysis.

To verify the superiority of the results listed in Table 4 and illustrations depicted from Fig. 17 - 24, Diebold and Mariano test is performed for each pair of models using

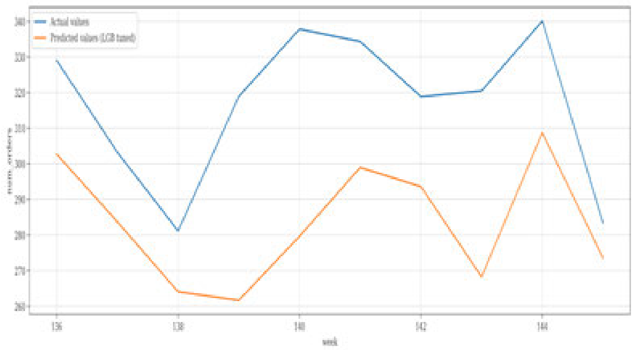

FIGURE 20. Actual versus predicted values for LightGBM.

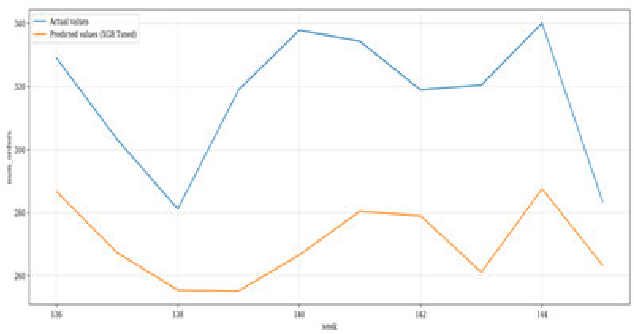

FIGURE 21. Actual versus predicted values for XGB.

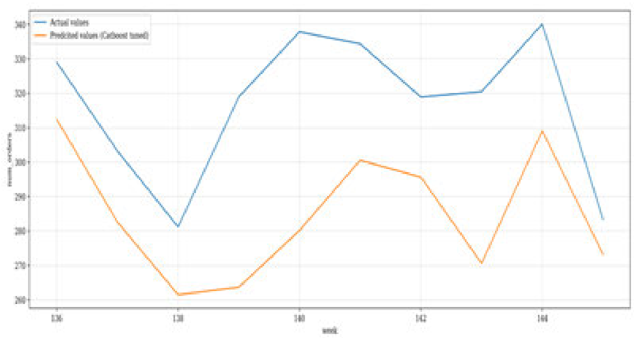

FIGURE 22. Actual versus predicted values for CatBoost.

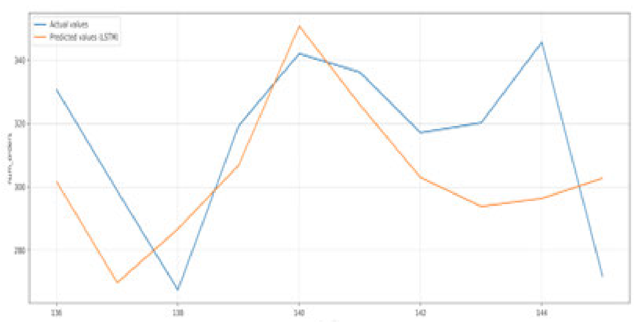

FIGURE 23. Actual versus predicted values for LSTM.

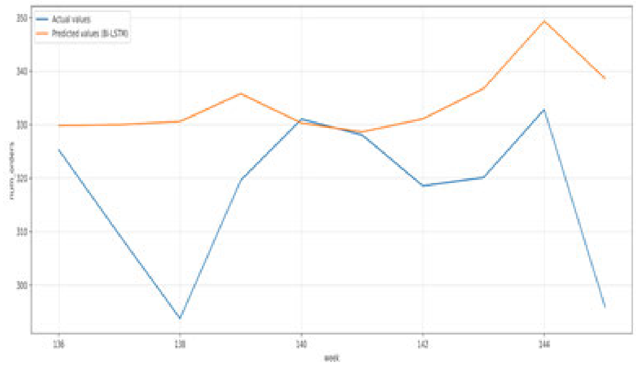

FIGURE 24. Actual versus Predicted values for Bi-LSTM.

TABLE 8. p-values of paired t-test.

Models	RMSLE	RMSE	MAPE	MAE
RFR and GBR	<0.0001	0.07	<0.0001	0.01
RFR and LightGBM	<0.0001	<0.0001	<0.0001	0.009
RFR and XGB	<0.0001	<0.0001	<0.0001	<0.0001
RFR and CatBoost	<0.0001	<0.0001	<0.0001	<0.0001
RFR and LSTM	<0.0001	<0.0001	<0.0001	<0.0001
RFR and Bi-LSTM	<0.0001	<0.0001	<0.0001	<0.0001
GBR and LightGBM	1	<0.0001	<0.0001	0.003
GBR and XGB	0.0002	<0.0001	<0.0001	0.001
GBR and CatBoost	<0.0001	<0.0001	<0.0001	0.0006
GBR and LSTM	<0.0001	<0.0001	<0.0001	<0.0001
GBR and Bi-LSTM	<0.0001	<0.0001	<0.0001	<0.0001
LightGBM and XGB	0.0002	0.0005	<0.0001	0.244
LightGBM and CatBoost	<0.0001	0.002	<0.0001	0.04
LightGBM and LSTM	<0.0001	<0.0001	<0.0001	0.0001
LighGBM and Bi-LSTM	<0.0001	<0.0001	<0.0001	0.0007
XGB and CatBoost	0.0002	<0.0001	<0.0001	0.0001
XGB and LSTM	<0.0001	<0.0001	<0.0001	<0.0001
XGB and Bi-LSTM	<0.0001	<0.0001	<0.0001	<0.0001
CatBoost and LSTM	0.0002	<0.0001	<0.0001	<0.0001
CatBoost and Bi-LSTM	<0.0001	<0.0001	<0.0001	<0.0001
LSTM and Bi-LSTM	<0.0001	0.0002	<0.0001	0.001

TABLE 9. p-values of one way ANOVA.

Models	p-values
RFR and GBR	0.49
RFR and LightGBM	0.01
RFR and XGB	<0.01
RFR and CatBoost	<0.01
RFR and LSTM	<0.01
RFR and Bi-LSTM	<0.01
GBR and LightGBM	0.08
GBR and XGB	<0.01
GBR and CatBoost	0.08
GBR and LSTM	<0.01
GBR and Bi-LSTM	<0.01
LightGBM and XGB	<0.01
LightGBM and CatBoost	0.35
LightGBM and LSTM	<0.01
LighGBM and Bi-LSTM	0.01
XGB and CatBoost	<0.01
XGB and LSTM	<0.01
XGB and Bi-LSTM	<0.01
CatBoost and LSTM	<0.01
CatBoost and Bi-LSTM	0.02
LSTM and Bi-LSTM	<0.01

squared loss function. The null hypothesis for this test is that it assumes that there is no significant difference in the accuracy of the regressor in the row/column. The alternative hypothesis assumes that the regressor in the column is more accurate than predictor in the row at 5% level of significance. From this table, it is evident that, among all the regressors, LSTM and Bi-LSTM statically outperform the others. Also, it can be seen than CatBoost performs statically better than other

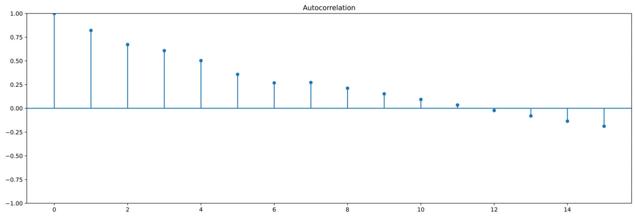

FIGURE 25. ACF graph of residuals of random forest regressor.

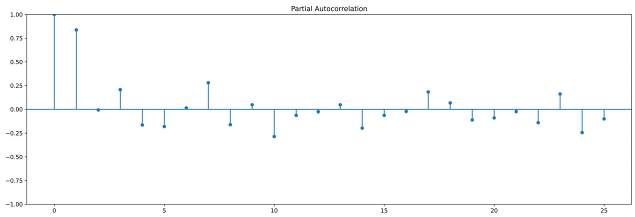

FIGURE 26. PACF graph of residuals of random forest regressor.

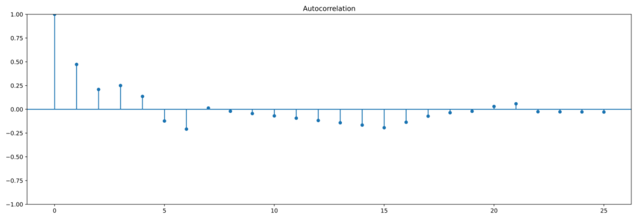

FIGURE 27. ACF graph of residuals of gradient boost regressor.

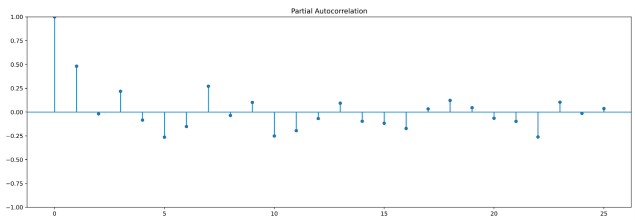

FIGURE 28. PACF graph of residuals of gradient boost regressor.

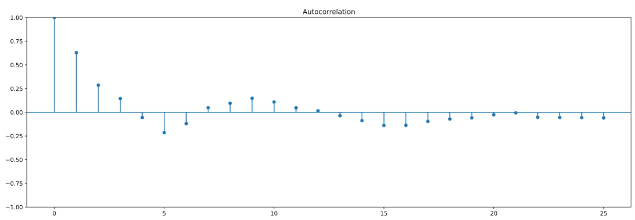

FIGURE 29. ACF graph of residuals of LightGBM.

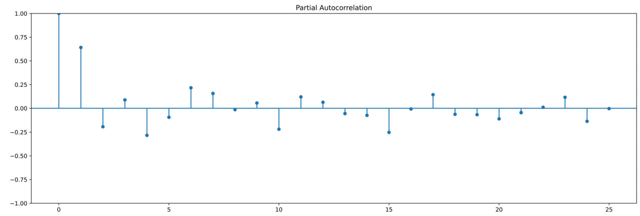

FIGURE 30. PACF graph of residuals of LightGBM.

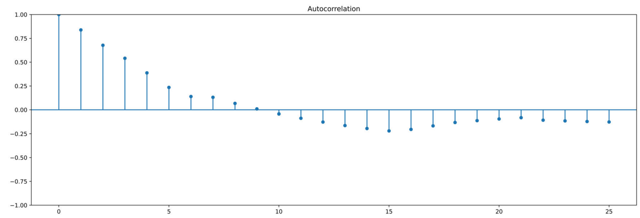

FIGURE 31. ACF graph of residuals of XGB.

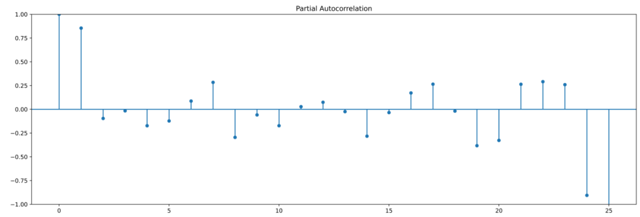

FIGURE 32. PACF graph of residuals of XGB.

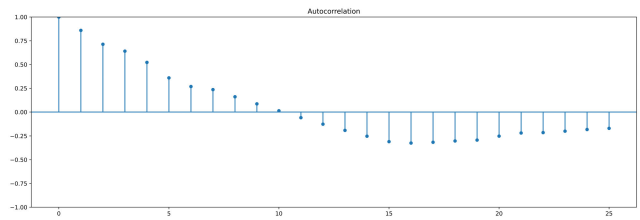

FIGURE 33. ACF graph of residuals of CatBoost.

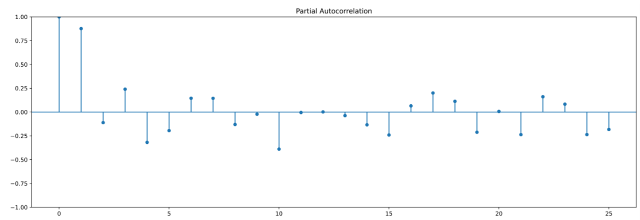

FIGURE 34. PACF graph of residuals of CatBoost.

machine learning models (RFR, GBR, LightGBM and XGB). CatBoost, LSTM and Bi-LSTM are the top three regressors in this case. LSTMs and Bi-LSTMs are known to handle time series data and predict accurate results due to their ability of handling historical data by solving the issue of long-term dependencies apart from making predictions based on the given feature values. CatBoost gave satisfactory results too owing to their ability of handling categorical data in a well manner.

To further ensure the quality of the proposed algorithm, one-way ANOVA is performed on the RMSLE, RMSE, MAPE and MAE of all the proposed models. To infer which models gave quality results, paired t-test is performed on the RMSLE, RMSE, MAPE and MAE of all the proposed models

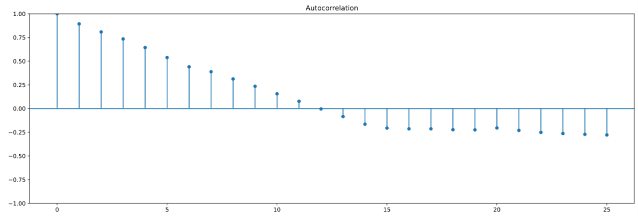

FIGURE 35. **ACF graph of residuals of LSTM.**

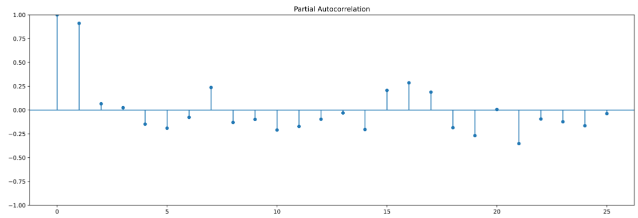

FIGURE 36. **PACF graph of residuals of LSTM.**

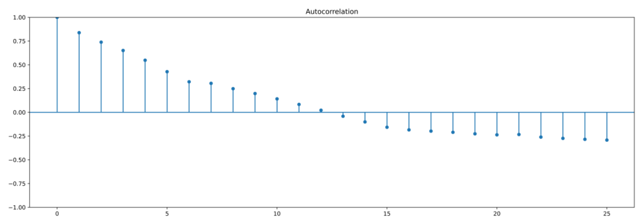

FIGURE 37. **ACF graph of residuals of Bi-LSTM.**

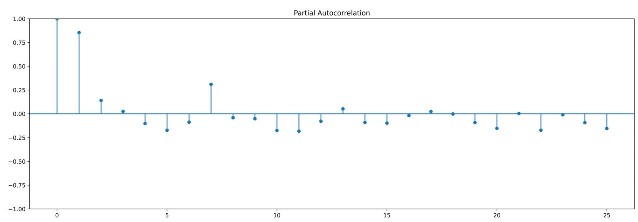

FIGURE 38. **PACF graph of residuals of Bi-LSTM.**

at a 5% level of significance. The null hypothesis in this case is that there is no significant difference between the metrics of the two models. If the t-test gives an output < 0.05, the null hypothesis is rejected. The results of ANOVA and t-test are present in Table 7 and 8 respectively. It turned out that the LSTM and Bi-LSTM produced similar performance when compared to the other models. In Table 7 it can also be seen that p-value for RMSLE is 1, which means that the RMSLE of any model is not statistically different from the other which is evident from Table 5 as well.

Moreover, pair wise Wilcoxon test is performed on the predicted values of all models pairwise to further ensure that the above results are in accordance with each other. The null hypothesis in this case is that there is no significant difference between the metrics of the two models. If the test gives an output < 0.05, the null hypothesis is rejected. The p-values from the Wilcoxon test are summarised in Table 9. It is evident that LSTM and Bi-LSTM perform better than other models.

The ACF and PACF plots for the residuals are plotted from Fig. 25-38. From these figures it is clear that the residuals are whitened and can be considered satisfactory.

The above discussion on the proposed models (RFR, GBM, LightGBM, XGB, CatBoost, LSTM and Bi-LSTM) along with the statistical tests and plots (Diabold and Mariano test, ANOVA, t-test, Wilcoxon test and ACF and PACF plots) have highlighted the performance of LSTM over other models in this study. The reason behind it is that they have the ability to capture long term dependencies which means that they can predict values based on the previous sequential data. This provides greater accuracy for demand forecasters thus resulting in better business decisions. As a result, there will be very little to no food wastage.

VI. LIMITATIONS

The proposed models undoubtedly perform well for the problem at hand, but, like any other models, there are a few limitations to them too. Random forests are generally slow and are ineffective for real time predictions as it may not be able to identify and formulate an increasing or decreasing trend. Gradient Boost are somewhat difficult to scale as every estimator is dependent on its predecessor. LightGBM and XGB are very sensitive to outliers. CatBoost works best on datasets with many categorical features, but is slow to execute with datasets containing too little categorical features. The deep learning models, LSTM and Bi-LSTM also have their share of limitations. Both LSTMs and Bi-LSTMs are complex and require more training data. Also, they are difficult to interpret.

VII. CONCLUSION AND FUTURE DIRECTIONS

The management of raw materials for a meal delivery service is significantly impacted by demand forecasting. Accurately forecasting the number of orders provides pertinent information to the concerned authority about the expected situation so that the inventory can be managed effectively without any waste. This study demonstrates the efficacy of deep learning and machine learning techniques for forecasting the volume of orders. In essence, these deep learning models are capable of identifying the time-variant characteristics and significant trends of historical data as well as predicting the future tendency of the given time-series data. On the basis of 135 weeks' worth of historical data, forecasts for the next 10 weeks are given. Each model's performance has been validated in terms of RMSLE, RMSE, MAPE, and MAE. Results and the statistical tests show that LSTM outperformed all other models in terms of forecasting performance. The dataset used in this study was restricted as it did not account for the date, month or any holidays. Without these factors, it was difficult to infer any trend or seasonality. Also, there was no mention of any event (like special discount or occasion) which may be able to explain sudden spikes of the target variable. The concept of applying transfer learning to time-series can also be explored as it may hold the key to improve the performance on smaller dataset. In future, these variables must be considered along with the limitations of the studied models to perform in depth analysis and propose a robust model.

FUNDING AND/OR CONFLICTS OF INTERESTS/COMPETING INTERESTS

The authors declare that they have no competing interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

[1] C. P. Veiga, "Analysis of quantitative methods of demand forecasting: Comparative study and financial performance," Ph.D. dissertation, Dept. Manag., Pontifical Catholic Univ. Paraná, Curitiba, Brazil, 2009.

[2] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

[3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 3146–3154.

[4] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2016, pp. 785–794.

[5] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: Gradient boosting with categorical features support," 2018, *arXiv:1810.11363*.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] S. Yadav, T.-M. Choi, S. Luthra, A. Kumar, and D. Garg, "Using Internet of Things (IoT) in agri-food supply chains: A research framework for social good with network clustering analysis," *IEEE Trans. Eng. Manag.*, vol. 70, no. 3, pp. 1215–1224, Mar. 2023, doi: 10.1109/TEM.2022.3177188.

[8] J. Zheng, L. Wang, L. Wang, S. Wang, J.-F. Chen, and X. Wang, "Solving stochastic online food delivery problem via iterated greedy algorithm with decomposition-based strategy," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 2, pp. 957–969, Feb. 2023, doi: 10.1109/TSMC.2022.3189771.

[9] V. K. Shrivastava, A. Shrivastava, N. Sharma, S. N. Mohanty, and C. R. Pattanaik, "Deep learning model for temperature prediction: A case study in New Delhi," *J. Forecasting*, vol. 43, no. 1, Feb. 2023, doi: 10.1002/for.2966.

[10] Y. Zhang, L. Wang, X. Chen, Y. Liu, S. Wang, and L. Wang, "Prediction of winter wheat yield at county level in China using ensemble learning," *Prog. Phys. Geogr., Earth Environ.*, vol. 46, no. 5, pp. 676–696, Oct. 2022.

[11] I. Shah, F. Jan, and S. Ali, "Functional data approach for short-term electricity demand forecasting," *Math. Problems Eng.*, vol. 2022, Jun. 2022, Art. no. 6709779.

[12] F. Lisi and I. Shah, "Forecasting next-day electricity demand and prices based on functional models," *Energy Syst.*, vol. 11, no. 4, pp. 947–979, Nov. 2020.

[13] I. Shah, H. Iftikhar, and S. Ali, "Modeling and forecasting electricity demand and prices: A comparison of alternative approaches," *J. Math.*, vol. 2022, Jul. 2022, Art. no. 3581037.

[14] I. Shah, S. Akbar, T. Saba, S. Ali, and A. Rehman, "Short-term forecasting for the electricity spot prices with extreme values treatment," *IEEE Access*, vol. 9, pp. 105451–105462, 2021.

[15] I. Shah, H. Bibi, S. Ali, L. Wang, and Z. Yue, "Forecasting one-day-ahead electricity prices for Italian electricity market using parametric and nonparametric approaches," *IEEE Access*, vol. 8, pp. 123104–123113, 2020.

[16] N. Bibi, I. Shah, A. Alsubie, S. Ali, and S. A. Lone, "Electricity spot prices forecasting based on ensemble learning," *IEEE Access*, vol. 9, pp. 150984–150992, 2021.

[17] E. S. Gardner Jr., "Exponential smoothing: The state of the art," *J. Forecasting*, vol. 4, no. 1, pp. 1–28, 1985.

[18] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *Int. J. Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.

[19] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA, USA: Holden Day, 1970.

[20] P. Ramos, N. Santos, and R. Rebelo, "Performance of state space and ARIMA models for consumer retail sales forecasting," *Robot. Comput.-Integr. Manuf.*, vol. 34, pp. 151–163, Aug. 2015.

[21] S. J. Taylor and B. Letham, "Forecasting at scale," *Amer. Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[22] C.-W. Chu and G. P. Zhang, "A comparative study of linear and nonlinear models for aggregate retail sales forecasting," *Int. J. Prod. Econ.*, vol. 86, no. 3, pp. 217–231, Dec. 2003.

[23] C.-Y. Chen, W.-I. Lee, H.-M. Kuo, C.-W. Chen, and K.-H. Chen, "The study of a forecasting sales model for fresh food," *Expert Syst. Appl.*, vol. 37, no. 12, pp. 7696–7702, Dec. 2010.

[24] K.-F. Au, T.-M. Choi, and Y. Yu, "Fashion retail forecasting by evolutionary neural networks," *Int. J. Prod. Econ.*, vol. 114, no. 2, pp. 615–630, Aug. 2008.

[25] Z.-L. Sun, T.-M. Choi, K.-F. Au, and Y. Yu, "Sales forecasting using extreme learning machine with applications in fashion retailing," *Decis. Support Syst.*, vol. 46, no. 1, pp. 411–419, Dec. 2008.

[26] E. Tarallo, G. K. Akabane, C. I. Shimabukuro, J. Mello, and D. Amancio, "Machine learning in predicting demand for fast-moving consumer goods: An exploratory research," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 737–742, 2019.

[27] A. Krishna, A. V. A. Aich, and C. Hegde, "Sales-forecasting of retail stores using machine learning techniques," in *Proc. 3rd Int. Conf. Comput. Syst. Inf. Technol. Sustain. Solutions (CSITSS)*, Dec. 2018, pp. 160–166.

[28] J. Ding, Z. Chen, L. Xiaolong, and B. Lai, "Sales forecasting based on CatBoost," in *Proc. 2nd Int. Conf. Inf. Technol. Comput. Appl. (ITCA)*, Dec. 2020, pp. 636–639.

[29] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *Int. J. Forecasting*, vol. 37, no. 1, pp. 388–427, 2021.

[30] H. Xu and C. Y. Wang, "Demand prediction of chain supermarkets based on LSTM neural network," *China Logistics Purchasing*, vol. 3, pp. 42–43, 2021.

[31] J. Kim and N. Moon, "BiLSTM model based on multivariate time series data in multiple field for forecasting trading area," *J. Ambient Intell. Hum. Comput.*, pp. 1–10, Jul. 2019.

[32] *Food Demand Forecasting Dataset*. Accessed: Aug. 10, 2022. [Online]. Available: https://www.kaggle.com/datasets/kannanaikkal/food-demand-forecasting

[33] J. Yu, S. B. Kim, J. Bai, and S. W. Han, "Comparative study on exponentially weighted moving average approaches for the self-starting forecasting," *Appl. Sci.*, vol. 10, no. 20, p. 7351, Oct. 2020.

[34] M. Markiewicz and A. Wyłomańska, "Time series forecasting: Problem of heavy-tailed distributed noise," *Int. J. Adv. Eng. Sci. Appl. Math.*, vol. 13, nos. 2–3, pp. 248–256, Sep. 2021.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, no. 10, pp. 2825–2830, Jul. 2017.

[36] M. Gong, Y. Bai, J. Qin, J. Wang, P. Yang, and S. Wang, "Gradient boosting machine for predicting return temperature of district heating system: A case study for residential buildings in Tianjin," *J. Building Eng.*, vol. 27, Jan. 2020, Art. no. 100950.

[37] S. Yao, A. Kronenburg, A. Shamooni, O. T. Stein, and W. Zhang, "Gradient boosted decision trees for combustion chemistry integration," *Appl. Energy Combustion Sci.*, vol. 11, Sep. 2022, Art. no. 100077.

[38] L. Ostroumova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[39] H. D. Nguyen, K. P. Tran, S. Thomassey, and M. Hamad, "Forecasting and anomaly detection approaches using LSTM and LSTM autoencoder techniques with the applications in supply chain management," *Int. J. Inf. Manage.*, vol. 57, Apr. 2021, Art. no. 102282.

[40] Z. Zheng, L. Feng, R. Liu, X. Wang, and Y. Sun, "Ultra-short-term forecast of multi-energy load for integrated energy system based on attention mechanism and BiLSTM," *J. Phys., Conf. Ser.*, vol. 2271, no. 1, May 2022, Art. no. 012018.

[41] F. X. Diebold and R. S. Mariano, "Comparing predictive accuracy," *J. Bus. Econ. Stat.*, vol. 13, no. 3, pp. 253–263, 1995.

SANDEEP KUMAR PANDA (Member, IEEE) is currently an Associate Professor and the Head of the Department of Artificial Intelligence and Data Science, Faculty of Science and Technology (IcfaiTech), The ICFAI Foundation for Higher Education (Deemed to be University), Hyderabad, Telangana, India. He has ten lakh seed money projects from IFHE. He has four edited books named *Bitcoin and Blockchain: History and Current Applications* (CRC Press, USA), *Blockchain Technology: Applications and Challenges* (Springer ISRL), *AI and ML in Business Management: Concepts, Challenges, and Case Studies* (CRC Press, USA), and *The New Advanced Society: Artificial Intelligence and Industrial Internet of Things Paradigm* (Wiley Press, USA), in his credit. He has published 50 papers in international journals and international conferences and book chapters in repute. He has 17 Indian patents on his credit. His research interests include blockchain technology, the Internet of Things, AI, and cloud computing. He is a member of ACM and a Life Member of IAENG. He received the "Research and Innovation of the Year Award 2020" from the MSME, Government of India, and the DST, Government of India, New Delhi, in 2020. He received the "Research Excellence Award" from Brand Honchos, in 2022.

SACHI NANDAN MOHANTY (Senior Member, IEEE) received the Ph.D. degree from IIT Kharagpur, India, in 2015, with MHRD Scholarship from the Government of India, and the Ph.D. degree from IIT Kanpur, in 2019. He has guided six Ph.D. scholars. He has published in 60 international journals of international repute. He has edited 24 books in association with Springer and Wiley. His research interests include data mining, big data analysis, cognitive science, fuzzy decision-making, brain–computer interface, cognition, and computational intelligence. He was elected as a fellow of the Institute of Engineers and a Senior Member of the IEEE Computer Society Hyderabad Chapter. He received three best paper awards during his Ph.D. degree at IIT Kharagpur from the International Conference in Beijing, China, and the other at the International Conference on Soft Computing Applications organized by IIT Roorkee, in 2013. He received the Best Thesis Award First Prize by the Computer Society of India, in 2015. He is also a Reviewer of *Journal of Robotics and Autonomous Systems* (Elsevier), *Computational and Structural Biotechnology Journal* (Elsevier), *Artificial Intelligence Review* (Springer), and *Spatial Information Research* (Springer).