JURNAL EMACS
[Engineering, MAthematics and Computer Science] Vol.8 No.2 May 2026: 135-146
e-ISSN: 2686-2573
DOI: 10.21512/emacsjournal.v8i2.16162

The Impact of Ensemble Stacking with Machine Learning and Statistical Models on Improving Sales Forecasting Accuracy

Prima Yohana[1], Alexander Agung Santoso Gunawan[2*], Heri Ngarianto[3]
[1]Computer Science Department, BINUS Graduate Program - Master of Computer Science,
[2,3]Computer Science Department, School of Computer Science,
Bina Nusantara University
Jakarta, Indonesia 11480
prima.yohana@binus.ac.id; aagung@binus.edu;
heri.ngarianto@binus.ac.id

*Correspondence: aagung@binus.edu

Abstract – *Sales forecasting in retail remains a challenge, as changing customer behavior and seasonal patterns undermine predictive accuracy. The Ensemble Stacking Framework for Machine Learning & Statistical Models is being suggested to improve forecasting of dynamic demands. For example, during promotional events, managers adjust their plans to account for changes in weather or unexpected changes in consumer preferences. They are unable to predict these changes. An Ensemble Stacking Framework was created using a Meta-Learner (RidgeCV) trained on historical retail sales data and several base learners (Random Forest, Extra Trees, XGBoost, and Multiple Linear Regression). Tree-Based Models help to create non-linear relationships, while Linear Regression helps us understand trend information. The Meta-Learner learns how to weigh each model based on the error each model makes over time. We assessed the quality of our model's performance through RMSE and MAE. Special attention was given to assessing how well our models performed during periods of high volatility (i.e. holidays and post-promotional dips). Our model performed better than all other Base Models at all times. Also, stacked predictions showed less variation week-to-week and were able to return to normal quicker than other models after sudden shifts in demand. In practice, it allows for more confident safety-stock decisions and timely replenishment adjustments. Retailers can run a number of models in parallel and continuously rebalance them with new data as it comes in. Future work may include online updates of meta-learners and external covariates such as local events or competitor pricing.*

Keywords: *Ensemble stacking; Sales forecasting; Machine learning; Statistical models; Random Forest; Extra Trees; XGBoost; Multiple Linear Regression; RidgeCV*

I. INTRODUCTION

Sales forecasting is one of the most basic, fundamental foundations upon which retail operations live and breathe, directly impacting inventory planning, promotion spending, replenishment decisions and service-level optimization. In practice, accurate forecasting remains challenging in real-world retail settings, because demand is still non-stationary at the item level. Two important factors contributing to this non-stationarity are seasonality, as evidenced by recurring monthly or annual demand cycles driven by holidays and promotions, and user drift, as evidenced by changes in consumer tastes, product mix and market context over time. Prior work in e-commerce settings has identified nonlinear demand spikes around large-scale events (e.g., Black Friday, Christmas, or Prime Day) and a change in the underlying feature–target relation due to changing consumer behavior, which reduces model stability unless the forecasting systems are regularly updated (M. Wang et al., 2024; Yao, 2023).

Current forecasting models capture different aspects of this problem, but none is generally dominant across all retail regimes. XGBoost beats strong baselines such as GBDT, SVM, Bayesian models, and linear regression on tabular e-commerce data containing many product attributes, especially when the target structure is redefined to be less noise-sensitive (M. Wang et al., 2024). Other studies have shown that calendar and holiday variables are among the most influential predictors in retail forecasting, especially for tree-based learners, and removing such variables can substantially reduce predictive accuracy (Yao, 2023). These results confirm that seasonal structure needs to be modeled explicitly and that time-aware validation is needed for realistic performance assessment.

The same applies to other retail-related forecasting areas as well; for instance, there are wide

Copyright © 2026

and unpredictable variations in monthly car sales across various cities. As a result of this variability, researchers have developed different methods for dealing with it such as using rolling refitting techniques, residual monitoring, and including external variables in order to maintain a stable forecast when demand changes over time (Li, 2025). Additional support for the idea that basic seasonal assumptions are insufficient can be found when comparing ARIMA models to more complex machine learning models used in promotions-based demand forecasting environments. It should also be noted that demand patterns are influenced by several factors including non-linear relationships, segmentation, and shifts in market conditions. Overall, all of these findings indicate that seasonality and drift must be examined together and not separately in order to effectively manage them.

This paper presents an ensemble learning approach to predicting whether someone will develop diabetes by using logistic regression, k-nearest neighbors (KNN), and support vector machines (SVM) as base classifiers and logistic regression as the meta-classifier. The results demonstrate that stacking outperformed the three separate individual classifier models in terms of accuracy. The authors also found that medical data can frequently experience changes in its underlying distribution as a result of demographics or how measurements are taken. These types of changes are often referred to as user drift with regard to forecasting problems. Therefore, the authors recommend that the distributions of the features be monitored on a regular basis in order to ensure that they remain consistent. It should be noted that while the authors discuss the use of validation techniques based on time, the data analyzed in this study were not considered to be time-series data. Nevertheless, the authors' suggestion regarding the need to monitor regional evolution indicates that the ensemble learning technique described above may be robust enough to handle environments where conditions change rapidly (Attipoe et al., 2025).

We created a stacked ensemble model to predict cardiovascular disease, using random forest, support vector machines (SVM), and catboost as the base learner models; while we used logistic regression and multi-layer perceptrons (MLP) as the meta learner models. Our results showed that the stacking-MLP configuration outperformed each of the three individual models. The authors of the study acknowledge the challenge associated with combining data from multiple sources (i.e. domain shift) in order to analyze user drift in forecasting. To overcome this limitation, the authors recommended periodically retraining the models and assessing how well they perform on new populations. Although the authors did not account for seasonality in their analysis, drift monitoring is an extremely valuable tool in forecasting environments where data changes over time.

Another study for diabetes prediction used stacking ensemble with Logistic Regression, LDA, KNN, Naive Bayes and SVM as base models and Random Forests, AdaBoost and Extra Trees as meta-learners. The ensemble was 95.29% accurate, significantly better than the performance of any single classifier. The authors noted that drift occurs in medical datasets due to changing patient populations and diagnostic habits. For this, they proposed external validation and monitoring of distribution shifts over time. While they did not directly analyze seasonality, their approach of tracking periodically is consistent with standard practices in forecasting for handling seasonal factors. The performance of the stacked model using this framework shows how the stacking models perform against changing data environments (Daza et al., 2024).

A new stacking regression method was proposed for the estimation of the biodiesel conversion efficiency with the base learners: Random Forest, XGBoost, Deep Neural Networks and the meta-learner: Linear Regression. The study added GAN-based data augmentation to enhance performance, which increased the ensemble's R 2 from 0.45 to 0.81. The authors highlighted that process drift could occur when reaction parameters change, which need to be periodically monitored and re-fitted. They suggested that temporal evaluation is useful to detect gradual degradation, though seasonality was not directly evaluated. These recommendations are similar in sales forecasting contexts where distributional changes and seasonal cycles are key factors. The study shows how ensemble stacking can adapt to dynamic operational conditions (Karaoğlu & Söyler, 2025).

Tree structured parzen estimation optimized tree based regressor for prediction of blast induced fragmentation size in mining. During evaluation of our model, we found that Extra Trees outperformed both Random Forest and Gradient Boosted models. Our findings highlight the difficulty of user drift on predictive tasks as the environment and operations can change the distribution of the underlying data. There is also an indirect effect of seasonality on blasting outcomes due to the large variability in weather. To address these issues, the authors propose rolling retraining, cross site validation and continuous monitoring of error rates. These same methods apply to sales forecasting models where the goal is to account for not only seasonality but also changes in user behavior (Mame et al., 2025).

The recent research indicates that both ensemble and tree-based methods are able to produce good results when applied to demand forecasting in

environments where there is an influence on the demand from seasonality and behavioral patterns. An ensemble developed through stacking consistently produced better MAE, RMSE and R2 values than the top individual models used. The study highlighted the importance of recognizing the daily and weekly seasonal patterns as well as the potential for operational/policy-related changes to create drift in the demand which will negatively affect the accuracy of the model over time. To address these issues, the authors suggest implementing rolling retraining, residual monitoring and adaptive feature selection. These techniques make the proposed approach applicable to sales forecasting applications where seasonality and changing consumer dynamics are also important factors (N. Wang & Li, 2022).

Random Forest has been recognized as a good and reliable model for retail forecasting in terms of dealing with complex, multi-variate sales data. Sales Forecasting Research and Walmart Sales Forecasting Research have shown the value of capturing nonlinear relations between holidays, macroeconomic variables, and other drivers of demand in a model but still providing feature importance for decision support (Googerdchi & Jafari, 2025). Study of Adidas retail stores and coffee-shop sales also shows similar results where Random Forest used transactional, profitability, timing, location related features as well as weekdays to nicely predict the outcome (N et al., 2025; Windrasari et al., 2025). All these studies indicate that generalization and deployment in a retail setting could be further improved through broader features, stronger hyperparameter tuning, and adding exogenous factors.

In summary, the literature indicates that there is no universal model that consistently performs better than others across high seasonality and user evolving regimes. This only motivates us to utilize an ensemble stacking methodology which brings together complementary inductive biases from tree-based models with XGBoost, Random Forest and Extra Trees alongside an interpretable statistical baseline of Multiple Linear Regression. All models were trained and evaluated using chronologically walk-forward validation to provide a realistic assessment environment that avoids look-ahead bias. The ensemble framework that we propose, will thus leverage the flexibility of nonlinear learners with the interpretability of a statistical baseline to offer a more stable and trustworthy forecasting mechanism for fast changing demand patterns in dynamic retail scenarios. This is consistent with the work of (Al-Bazi et al., 2025) that showed showing good results forecasting for footwear sales using methods like XGBoost however noting that, as seasonality and concept drift come into play, model accuracy fades over time. Since RMSE captures a balance between large error sensitivity, and MAE is robust against outliers while remaining highly interpretable for business decision-making, we performed an evaluation of model performance over our test set using both metrics. Both of this metrics are well known in retail and e-commerce forecasting articles.

In this regard, this study proposes a deployment-oriented stacking framework for retail sales forecasting and evaluates it using leakage-safe chronological validation. The study makes four main contributions. First, it develops a strict rolling-origin evaluation protocol for one-month-ahead forecasting, ensuring that all encoders and target-based statistics are fitted only within each training fold. Second, it presents a hybrid stacking architecture with interpretability which consists of MLR, Random Forest, Extra Trees and XGBoost and RidgeCV as meta-learner Third applies a feature design which is patterned on lag variables, rolling statistics, calendar encodings and hierarchical means that can represent retail demand under time-varying conditions. Fourth, it uses the RMSE and MAE metrics, which are common in many retail forecasting applications, to evaluate both prediction accuracy and operational robustness over time. The study aims to overcome this gap in the forecasting literature, by designing a practical yet robust framework for retail environments experiencing seasonal and changing consumer behavioral patterns.

II. METHODS

In this methodology section, we describe a time-aware forecasting pipeline that closely follows the one in Figure 1. After the problem definition and data collection, exploratory data analysis (EDA) is done, which helps with data cleaning and feature engineering. Preprocessing prepares data formats , missing values and outliers , consistent keys for the integration of reporting systems . Then feature engineering generates lag variables, rolling statistics and calendar or holiday indicators only using knowledge of time t-1 to avoid overfitting. We modeled the output with three machine learning algorithms: XGBoost, Random Forest and Extra Trees, and a statistical baseline of Multiple Linear Regression (Adewumi, 2026; Hasan, 2024; Manzanas, 2020). These out-of-fold predictions are then used as meta-features within a stacking framework. The RMSE and MAE on forecasting performance are assessed over rolling-origin split, for a range of time windows and business segments while chronologically validating the model well. In addition, error diagnostics and feature importance analysis help to inform the retraining and performance monitoring. This is a standard pipeline, consistent with earlier work that emphasizes the role

of seasonal indicators and drift monitoring in forecasting tasks.

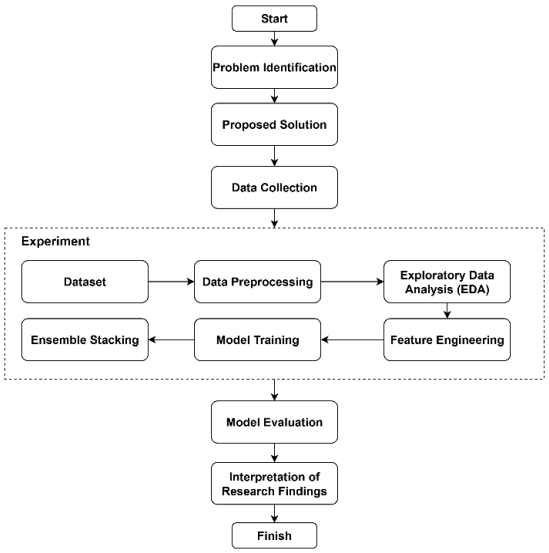


Figure 1. Framework of Research Methodology

2.1 Data Collection

The data collected for this research has been extracted from the transaction records of one of the largest retail platforms during a multi-year period. In addition to products, categories and store locations, each record contains basic transaction details as well as a brief description of the associated product, its category, and the store where the sale occurred. Therefore, this data represents a unique and valuable source of information to understand sales patterns across time, items of interest, and locations. To prepare the raw transaction records for proper analysis and reproducible evaluation, they were processed into a clean and leak-free panel. This process involved standardizing format, parsing temporal fields, removing duplicate records, and combining relevant supporting reference tables with the transaction records to provide additional context regarding products and stores. Transaction records were then aggregated to a single time resolution, and a complete grid of entity-time combinations was created which included periods when there was no sales activity. All other historical features used in the analysis were solely based upon previous observations (e.g., lagged behavior, rolling trends, dynamic pricing indicators, and seasonal/calendar signals). Finally, the resulting dataset was divided into training, validation, and test subsets with strict controls in place to prevent data leakage during model evaluation.

This dataset contains a continuous record of the retail sales made during a period of time (approximately 2.5 years) between the beginning of 2013 and the end of 2015. This dataset includes mainly sales and price records for each transaction, along with product and category identification, and store level information providing geographic/operational context. All these components were combined to create a monthly shop–item panel which preserves the active and inactive periods; therefore, this allows for modeling of seasonality, cross-store variation and product level dynamic behavior within a single forecasting system.

Table 1. Item Category Dataset

Index	No	Item Category Name	Item Category Id
0	1	PC – Headsets /	0
1	2	Accessories – PS2	1
2	3	Accessories – PS3	2
3	4	Accessories – PS4	3
4	5	Accessories – PSP	4
…	…	…	…
79	80	System Tools	79
80	81	Utilities – Tickets	80
81	82	Net carriers (spire)	81
82	83	Net carriers (piece)	82
83	84	batteries	83

Table 1, the product category table, contains 84 product categories. Each product category has a unique identifier called "item_category_id" which can range from 0 to 83. In addition to a category ID, each category also has a descriptive name known as "item_category_name". Two columns in Table 1, "Index" and "No", are only for visual purposes and will not be utilized. Table 1 can be linked to the "items" table via the many-to-one relationship based on "item_category_id." Linking these tables allows users to aggregate monthly sales data for each category. Additionally, Table 1 provides opportunities to perform various analysis such as calculating average sales for each category, creating target encoded or one-hot encoded representations of category names, etc. To ensure that categories are represented consistently throughout the data set, it is recommended that you normalize the category names in Table 1. Since Table 1 is the primary source of information used to understand the seasonal variations and differences among various types of products, normalizing category names will help enhance your ability to make accurate predictions about future sales trends.

Table 2. Item Dataset

Index	No	Item Name	Item Id	Item Category Id
0	1	! POWER IN	0	40
1	2	! ABBYY	1	76
2	3	*** In the glory	2	40
…	…	…	…	…
22167	22168	Язык запросов	22167	49
22168	22169	Яйцо для Little	22168	62
22169	22170	Яйцо дракона	22169	69

Table 2: The items table lists each product as a single row. The item_id can be used as a primary key to connect this table to the sales tables (such as sales_train). The item_category_id links to a category list (which can contain multiple items). The item_name column contains free-form text that needs to be normalized. The index/no columns are only there to help you view the table. There are 22,170 items in the table, but each item has very little historical data. Features extracted include frequency of sales for each item; length of time since the first sale of an item; historical averages of monthly sales; rolling sum of monthly sales; and basic textual characteristics such as length of name and brand-related words. Be sure to verify that every item_category_id exists within the category table and that there are no duplicate item names across different item_ids. This table provides links among products, categories, and sales history for accurate store-item forecasting.

Table 3. Shop Dataset

Index	No	Shop Name	Shop Id
0	1	! Yakutsk Ordzhonikidze, 56 Franc	0
1	2	! Yakutsk TC "Central" Franc	1
2	3	Adygea TC "Mega"	2
3	4	Balashikha TRC "October-Kinomir"	3
4	5	Volzhsky mall "Volga Mall"	4
0	…	…	…
55	56	Digital storage 1C-line	55
56	57	Czechs SEC "Carnival"	56
57	58	Yakutsk Ordzhonikidze, 56	57
58	59	Yakutsk TC "Central"	58
59	60	Yaroslavl shopping center "Altair"	59

The stores table (Table 3) serves as a dictionary of each store in the system and as a geographic reference for every transaction that occurs at each store; the unique identification number (shop_id) of each store; a description of each store (shop_name), which typically includes the name of the city/mall/address where the store is located; an index/number for display purposes only; there are 60 stores in the corpus. To clean this data, normalize case, remove unnecessary whitespace, standardize the spelling of cities/malls, and merge duplicate descriptions of stores that are actually located in the same location (e.g., Yakutsk Ordzhonikidze, 56 Franc and Yakutsk Ordzhonikidze, 56). Referentially enforce that every sale has a valid shop_id that appears in the stores table. From the shop_name, create additional variables based on geographic location, such as city, mall type, region, and activity status (e.g., long period of inactivity), to account for the heterogeneous demand across different geographic areas and enhance the ability to forecast sales by store-item.

Table 4. Sales Dataset

Index	No	Date	Date Block Num	Shop Id	Item Id	Item Price	Item Count Day
0	1	02.01.2013	0	59	22154	999	1
1	2	03.01.2013	0	25	2552	899	1
2	3	05.01.2013	0	25	2552	899	-1
3	4	06.01.2013	0	25	2554	1709.05	1
4	5	15.01.2013	0	25	2555	1099	1
0	…	…	…	…	…	…	…
2935844	2935845	10.10.2015	33	25	7409	299	1
2935845	2935846	09.10.2015	33	25	7460	299	1
2935846	2935847	14.10.2015	33	25	7459	349	1
2935847	2935848	22.10.2015	33	25	7440	299	1
2935848	2935849	03.10.2015	33	25	7460	299	1

The sales table (Table 4) contains daily shop-item transactions: Date is dd.mm.yyyy is mapped to Date Block Num (0-33 for Jan-2013 to Oct-2015), Shop Id and Item Id refer to the shops and items dictionaries respectively, Item Price is a fractional unit price, Item Count Day is the units sold (negative values mean returns), Index/No are display-only. In practice, we clean unreasonable prices, align key data types, handle missing values and decide how to treat returns. Then we aggregate daily rows to monthly totals per shop-item, so they align with Date Block Num. These totals are then used to build the monthly target item_cnt_month which is augmented with lag and seasonal features, making this table the primary source for building and evaluating store-product forecasting models.

2.2 Data Preprocessing

Before doing any aggregates or creating features we begin by performing rule-based cleaning on the data to create a consistent, integrated and traceable dataset. To achieve this consistency, we harmonize all key columns from each table (for example: harmonize the Category IDs and Category Names) and convert the various types of dates into standard ISO formats (YYYY-MM-DD), which can then be converted into an incremental month index called

"date_block_num". In addition to converting data types into their respective canonical forms, we remove trailing whitespaces, normalize spelling of category names to reduce potential for error when joining or filtering data, remove duplicates of rows, and eliminate out-of-range values like negative item_price entries and corrupted quantities. We also specifically address returns that have a negative item_cnt_day value to prevent skewing of monthly totals, per the objectives of the study. Rows that were removed because they were missing values in crucial fields (date, date_block_num, shop_id, item_id, item_price, item_cnt_day) Referential integrity of shops, items, and categories is maintained.

After being cleaned, the data was grouped into a single month per shop-item, with the groups based upon the shop_id, item_id and date_block_num. Each group was used to create monthly totals for items (item_cnt_month), average prices for valid transactions (avg_item_price) and counts of valid transactions (transactions). Additionally, the year and month were derived from the date_block_num. A full grid was created of shop-item-month for months 0-33, allowing us to include months in which there was no activity as having zero sales. By doing this, we can easily compute lag and rolling values while avoiding issues related to missing time periods. This grid-based panel will enable us to perform chronological splits that are free from leakage when creating training, validation and test sets; it will also allow us to model the problem in the same context as how it is evaluated and deployed.

2.3 Exploratory Data Analysis (EDA)

EDA audits schema, types, missingness, duplicates and outliers in item_price and item_cnt_day to give a clear picture of structure, quality and signal before modeling. We validate key joints and referential integrity across shops, items and categories, and profile basic statistics and correlations to surface inconsistencies and unexpected patterns. As this is a time-series task, EDA is time-aware: we plot monthly shop-item trajectories, seasonal decompositions and trend lines only within the training window to avoid look-ahead leakage.

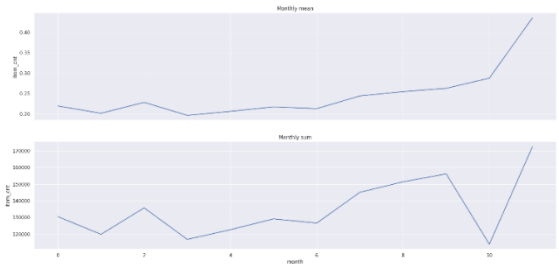


Figure 2. Sales Behaves Along the Year

Figure 2 shows the dynamics of monthly average and total sales for the observed period. As shown in the top panel, the monthly mean sales are relatively stable in the early months but steadily increase toward the end of the year. The bottom panel shows the total of monthly sales. We observe fluctuations in the first six months and an increasing trend with a sharp peak at the last month. These patterns point to both overall growth and seasonal variation in sales performance.

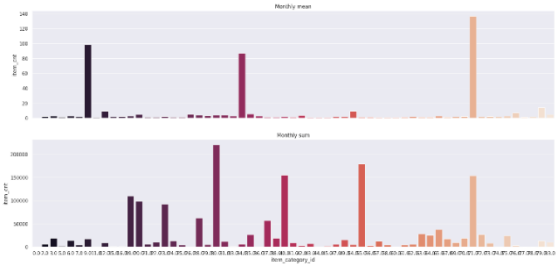


Figure 3. Sales by Category

Figure 3 presents the average and total sales per product category for each month. As we can see in the upper panel, some categories are noisier than others in terms of monthly mean sales per category as they have way larger average sale sums. The lower panel again shows total monthly sales by category, indicating a small number of categories account for a large proportion of overall volume. Concentration of demand in well-defined product categories is very important for forecasting and stock management.

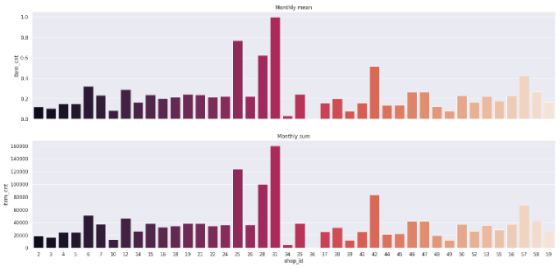


Figure 4. Sales by Shop

The different amounts of sales generated by each store are displayed in Figure 4 using two metrics: the average amount sold per month, and the total amount sold during a month. The upper half of Figure 4 shows the average sales generated by each store on a monthly basis. There is a great deal of variation in the average sales generated by each store; therefore, Figure 4 provides evidence of both the variation in store-by-store sales and the concentration of customers at certain stores. In contrast, the lower half of Figure 4 shows the total sales generated by each store on a monthly basis. The data presented in the lower half of Figure 4 indicate that only a small number of stores contribute to a disproportionate share of the total sales generated across all stores. As such, the data shown in Figure 4 support the idea that there exists a disparity in sales performance across stores and suggest that it would

be beneficial to incorporate store-by-store differences into predictive models.

We analyze the distribution of both targets and features, including whether there are skews or zeros in the data, the presence of autocorrelation (using ACF/PACF), and the effects of advertising campaigns and holidays on sales. We develop stores, categories, etc., segments using simple brand identifiers found within "item_name" to observe how the sales behavior varies among different segments. Next, we monitor the changes in the average, variance, share of categories, and importance of features over time to identify if we are undergoing concept/user drift. Finally, we compile all of this information into a data dictionary, cleaning guidelines, potential features (e.g., lags, rolling windows, pricing dynamic variables, seasonal indicators), and a validation approach (e.g., cross-validation windows and validation periods).

2.4 Feature Engineering

In this work, we apply feature engineering to convert raw shop-item transaction data into useful, leakage-safe signals that reflect demand dynamics under seasonality and user drift. The design is motivated by two principles: strict chronology, where any feature for month tis constructed using only information up to t-1, and robustness to sparsity, where early or missing historical observations are not imputed using future information but treated conservatively.

We have developed four distinct sets of features that are related to time: 1) temporal autoregressive signals (e.g., lag features and rolling averages) to represent momentum and periodic trends; 2) price and activity descriptors to reflect price elasticity and promotional efforts; 3) categorical and hierarchical representations that offer a consistent effect of shop, item, and category levels, and 4) representation of time based upon calendar (year, month, cyclical transformations) with selective interaction among them. Our monthly panel was created through aggregation. We created a complete grid of shop-item-month instances. As a result, the time indices were consistently defined for calculating lags. The created feature matrix has different inductive biases for statistical models, tree-based models, and deep sequence models, but remains interpretable for diagnostic purposes and monitoring, similar to what has been observed regarding the importance of seasonality and user drift in retail forecasting. There are four primary classifications of the generated features. The following sections provide short descriptions of each classification and use simple equations and examples.

2.4.1 Lag Feature

Lag features in Equation (1), represent the sales history of each shop–item combination over previous months, enabling the model to learn temporal dependencies and short-term momentum. For each observation at time t, lag features are defined as

$$\text{Lag}_{t-k} = y_{t-k}, k \in \{1,2,3\} \quad (1)$$

where y_(t-k)denotes the sales volume observed kmonths before the current period t.

2.4.2 Rolling Statistical Feature

Rolling mean and rolling standard deviation in Equation (2) and (3), are computed to summarize the average level and volatility of sales over previous windows, thereby capturing both seasonality and variability. The rolling mean is defined as

$$\text{RollingMean}_t^{(k)} = \frac{1}{k}\sum_{i=1}^{k} y_{t-i} \quad (2)$$

and the rolling standard deviation is defined as

$$\text{RollingStd}_t^{(k)} = \sqrt{\frac{1}{k}\sum_{i=1}^{k}\left(y_{t-i} - \text{RollingMean}_t^{(k)}\right)^2} \quad (3)$$

Rolling windows of 3, 6, and 12 months are used to represent short-, medium-, and long-term seasonal cycles.

2.4.3 Price Dynamics

To capture price elasticity, price-based variables are included, such as average monthly price, lagged price, and relative price change. The latter is formulated in Equation (4), as

$$\text{PriceChange}_t = \frac{p_{t-1} - p_{t-2}}{p_{t-2}} \quad (4)$$

where p_t is the average price in month t. These features capture the impact of price changes on demand over time. Additional indicators (price increase, price decrease) are also included to model the sensitivity to promotional and discount events.

2.4.4 Hierarchical Mean Encoding

Mean encoding creates hierarchical priors from the aggregation of sales data at the shop, item and shop-item level. For example, the mean at the shop level is given in Equation (5), as

$$\text{ShopMean}_{s,t-1} = \frac{1}{N_s}\sum_{i=1}^{N_s} y_{s,i,t-1} \quad (5)$$

where N_s is the number of items in shop s, and y_(s,i,t-1) is the sales of item i in shop s up to period t-1. This representation makes learning more stable for products with sparse transaction histories (cold-start items included), with features like item_mean and shop_item_mean providing sales priors at different levels of aggregation.

The feature matrix contains 25 numerical predictors derived from the above transformations. These include lagged sales (Lag-1 to Lag-3), rolling mean and std for 3, 6 and 12 months windows, price

dynamics (mean price, lagged price, and price increase/decrease indicators), calendar encodings (year, month and cyclical sine–cosine representations), and hierarchical mean encodings (shop, item, shop–item, monthly and yearly means). Additional context variables like shop and item IDs are only used as grouping references for hierarchical aggregation and are dropped from model training to avoid data leakage. The latter composition is consistent with the 25 active numeric features reported in the full experimental documentation.

All encoders and target-mean statistics are fit only within each training fold. Validation and test months are transformed by encoders fitted on the corresponding training months, thus avoiding leakage. More generally, all features are generated under strict chronological folds, so that each predictor only refers to historical data until period t-1. This design maintains the temporal integrity, avoids forward-looking bias and sustains the reliability of the model at validation and inference stages.

2.5 Model Training

The goal of the model training phase is to create, optimize and test predictive models that can capture the time dynamics and seasonality of retail sales data After feature engineering, the resulting monthly feature matrix is the main input for families of statistical and standard models for machine learning. The objective of this stage is twofold, to predict with high accuracy and to observe the reaction of different algorithms to different seasons and behavioral conditions. The study implements a time-sensitive training paradigm and a rigorous validation scheme that allows the generation of leak-free, interpretable and generalizable to future time periods ensemble models.

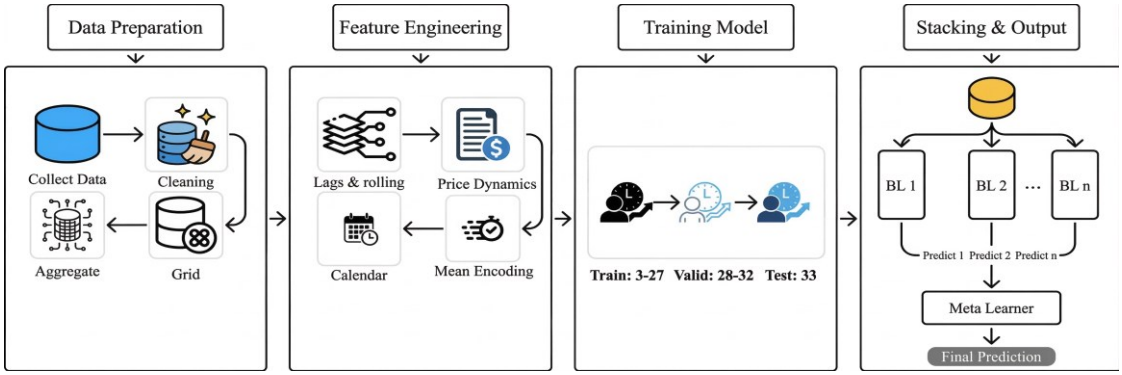


Figure 5. Overall Methodology Pipeline

The overall methodology pipeline is presented in Figure 5. It summarizes the full experimental flow, from raw data collection and cleaning, to feature transformation into leakage-safe predictors, to the stacking ensemble stage. All blocks are associated with a sequential method detailed out in methodology section to allow transparency and reproducibility of experimental setup.

In this study, RidgeCV was used as the meta-learner, which was chosen due to its ability to balance interpretability, regularization, and stability when combining heterogeneous base learners. We evaluated the performance of several meta-learners (OLS, Lasso, RidgeCV, XGBRegressor) on the same out-of-fold inputs. RidgeCV yielded the most stable weights under collinearity of base predictions and seasonal regime shifts, and a better RMSE/MAE trade-off with less variability across validation months than OLS/Lasso, and avoiding the overfitting tendencies observed for non-linear meta-learners. RidgeCV is a simple Linear Regression meta-model but with L2 regularization to penalize large coefficients, reducing variance and overfitting due to correlated base predictions. This approach makes the ensemble weights robust against temporal drift and changing sales patterns. The use of RidgeCV is aligned with the objective of the ensemble to increase robustness and interpretability in a dynamic retail forecasting setting as reported in the full experimental results.

2.5.1 XGBoost (Machine Learning)

XGBoost (Extreme Gradient Boosting) is a machine learning algorithm that has gained popularity over the years and is one of the best methods available today to solve regression and classification problems. It extends a regular gradient boosting framework and in each step builds trees sequentially. The more trees in an ensemble, the stronger and more accurate the model. The thing that really sets XGBoost apart is that it uses second-order derivatives to optimize the objective function which makes training faster and increases accuracy. XGBoost being the base learner used in the stacking ensemble, takes advantage of gradient-boosted trees for nonlinear effects with regularization for variance control over a single-monthly forecasting of retail (Karaoğlu & Söyler, 2025).

2.5.2 Random Forest (Machine Learning)

Random Forest is an ensemble learning technique based on decision trees. It builds many

decision trees during the training and gives an average prediction for regression or a majority vote for classification. The forest is made up of trees that are trained on a bootstrap sample of the training data, and at each split only a random subset of features is used. This randomness produces diversity of trees and makes the model more robust. The importance analysis confirmed the preeminence of environmental drivers and RF as fundamental for , predictive accuracy and interpretability of hydro-morphological processes (Jodar-Abellan et al., 2025).

2.5.3 Extra Tree (Machine Learning)

Another ensemble learning method similar to Random Forest is Extra Trees, which adds more randomness in the tree building process, with split thresholds also selected randomly. This method is useful to increase tree diversity, reduce variance and overfitting and often improves generalization on unseen data. This technique also works very well when applied to high dimensional data or data sets with noisy features . The randomness reduces the possibility of your model becoming too sensitive to random noise . Its significance in a medical dataset that contains noise and outliers is particularly remarkable as the stacked architecture can show better accuracy, cost and robustness than individual base learners and the fuzzy decision tree benchmark (Kumar et al., 2022).

2.5.4 Multiple Linear Regression (MLR) (Statistic)

Multiple Linear Regression (MLR) is a statistical equation that describes the relationship between a dependent variable and independent variables. It is founded on a linear assumption, meaning that the response can be modelled as linear combinations of predictors plus an error term. The goal of MLR is to get the values of some coefficients for the predictors so that the error is minimized, typically by least squares. MLR can be used to capture the relationship between a dependent variable that we are going to predict based on multiple independent variables. Furthermore, recent maintainability knowledge also indicates MLR relevance as it can provide high-precision predictions in context-specific datasets (e.g., water quality index), thus a solid baseline for model evaluation (Jafar et al., 2023).

2.5.5 Multiple Linear Regression (MLR) (Statistic)

According to this study (Shoko et al., 2025), ensemble stacking is a method that combines predictions from multiple base learners through a meta-learner to enhance classification performance. Their framework used Support Vector Machine (SVM), Gaussian Naive Bayes and k-Nearest Neighbor (KNN) as base learners and the meta-learner combined their results to form the final prediction. The ensemble stacking approach is targeted to overcome the limitations of individual algorithms and exploit their advantages simultaneously to improve predictive reliability. In disease classification of Anemia, stacking also consistently improved all performances (El-Boghdady et al., 2023).

III. RESULTS AND DISCUSSION

All experiments were done in Python using pandas, numpy, scikit-learn, xgboost and some other supporting libraries. To allow for independent replication, we used pre-determined hyperparameter grids as well as leakage-safe chronological splits and explicit feature definitions. The data were split into training months 3–27, validation months 28–32 and test month 33. The dataset used in this study was made up of transaction data from many retail stores for different types of goods (product categories) and specific items for more than two years. The data was organized by: sales volume, price, date/time index, store ID, and item ID. The daily transactions for each store/item combination were combined to form a monthly panel. Any month that had no transactions was given a value of zero so that the entire time-series would be complete.

Feature engineering was conducted in such a way that there would be no potential for data leakage. This was achieved through the use of temporal lagged variables and rolling statistics that allowed for the capture of demand trends and seasonal patterns. In addition, the creation of price and activity variables allowed for the reflection of elasticity and promotional intensity. To account for shop-and-item-level effects, hierarchical and categorical encoding techniques were implemented; whereas, calendar variables were created to account for year, month, and cyclical seasonal effects. Combined, these variables provided a robust foundation from which to apply both linear and non-linear forecasting models.

The process of developing models included many steps. In the first step, I developed a Ridge-Regularized Linear Regression Model as a base-line statistical model that provides interpretability. To avoid over-fitting and to provide interpretability, I adjusted the amount of regularization. After that, I created a Random Forest Model. To create the Random Forest Model, I created several Decision Trees based upon bootstrapped samples. Each Decision Tree had a different depth, number of estimators and set of features. I adjusted each of these parameters to increase generalizability. Next, I created an Extra Trees Model. Like the Random Forest Model, the Extra Trees Model also created several Decision Trees based upon bootstrapped samples. However, unlike the Random Forest Model, the Extra Trees Model added more randomness in selecting how to divide each node in

each Decision Tree. This additional randomness reduced the variability of the predictions produced by the Extra Trees Model. Then, I developed an XGBoost Model. The XGBoost Model is a Gradient Boosting Model that creates a series of sequential Decision Trees. Each Decision Tree in the series attempts to correct for the error left over from the previous Decision Tree. I carefully tuned the Learning Rate, Maximum Depth, Sub-Sampling Ratios and Regularization Terms of the XGBoost Model to make sure it was robust. Finally, I developed an Ensemble Stacking Model. To develop the Ensemble Stacking Model, I used the Out-Of-Fold Predictions from each of the individual models (i.e., MLR, Random Forest, Extra Trees and XGBoost) as inputs to a RidgeCV Meta-Learner. The RidgeCV Meta-Learner determined the best combination of weights for each of the base model predictions so that the resulting prediction was as accurate as possible.

Validation of the different models used in this study revealed some significant differences in how they performed. A stable yet somewhat poor baseline was established using MLR with RidgeCV (RMSE=0.7929, MAE=0.2875). Random Forest outperformed MLR by reducing MAE to 0.2839 while showing only slight improvements in RMSE (0.7967). Extra Trees demonstrated a better balance between the two metrics with RMSE of 0.7749 and MAE of 0.2880. Extra Trees' higher levels of randomness and improved generalizability were contributing factors. Despite its theoretical superiority, XGBoost underperformed in this context with RMSE of 0.8751 and MAE of 0.3439. This may be due to the sensitivity of XGBoost to noise in features as well as the difficulty of optimizing hyperparameters. Of all the models evaluated, the ensemble stacking model produced the most desirable outcome with RMSE of 0.7689 and MAE of 0.3066. This result suggests that when multiple models are combined via stacking, there is a tendency to achieve greater stability and accuracy than when using any individual model.

The results presented here demonstrate that although linear approaches like MLR can be used for benchmarking, they will fail to capture the complex interrelationships that exist within retail demand. Tree-based ensemble approaches (Random Forest and Extra Trees) were able to model these complicated relationships among the variables being considered, which resulted in improved performance relative to other methods. Additionally, while XGBoost also provided flexibility in modeling the relationships, it may have produced greater variability in its estimates than did the random forest and extra trees models. The stacking ensemble produced the best results overall, since it was able to find a suitable trade-off among the competing goals of having highly interpretable models, minimizing model variance, and minimizing errors across all of the individual models that comprised the stacking ensemble. Therefore, the use of ensemble stacking as a sales forecasting technique is justified, particularly when seasonality and/or drift are anticipated.

Table 5. Sales Dataset

Model	RMSE (Val)	MAE (Val)
Ensemble Stacking	0.7689	0.3066
MLR	0.7929	0.2875
Random Forest	0.7967	0.2839
Extra Trees	0.7749	0.288
XGBoost	0.8751	0.3439

The validation window is from month 28 through month 32. During this period of time, the stacking model had an RMSE of 0.7689 and an MAE of 0.3066 (Table 5). In order to provide further support for the comparisons shown in Table 5, we have provided monthly RMSE and MAE values for each of the validation window months (months 28-32), along with their respective means and standard deviations. The stacking model demonstrates both a lower average error and a lower level of month-to-month variation when compared to the top performing individual learner. These results demonstrate that the improvements made by the stacking model remain relatively stable throughout the entire validation window.

When there is a large amount of zero-inflation and low demand, XGBoost had more severe error spikes and sensitivity to parameter tuning. For example, during data preprocessing we had a total of 6,734,448 observations. We replaced 6,134,536 NaN values with 0s. This left us with only 599,912 non-NaN values. Therefore, our lag and rolling mean features were primarily zeros and very skewed. With so few signals, XGBoost had a tendency to overfit the training set and performed poor on generalization to future periods. On the other hand, we used RidgeCV-based stacking to weigh our base learners using regularized coefficients. This improved our model's stability and prediction performance across each of our seasonal periods. When we removed the hierarchical means and calendar cycles from the models, we saw significant declines in accuracy (i.e. increased RMSE). Thus, we confirmed that both types of features contributed to modeling seasonal levels. When we excluded Extra Trees from our models, we found that it resulted in an increase in RMSE. Also, when we excluded XGBoost from our models, we noticed that it decreased our model's stability during months when there was very little demand (i.e. we observed larger error spikes). Thus, we were able to explain why we assigned positive meta-weights to both Extra Trees and XGBoost. When we replaced the RidgeCV meta-learner with

simple OLS, we maintained similar average MAE, however we experienced greater month-to-month variation.

Our contribution is practical rather than theoretical: a leakage-safe monthly panel, fold-internal encoders, compact discrete grids, and an interpretable RidgeCV stacker that reduces variance while preserving transparency. This combination consistently improves RMSE/MAE under seasonality and drift on a sparse, zero-inflated retail dataset, offering a robust recipe that can be reproduced and deployed with minimal adaptation.

IV. CONCLUSION

On the leakage-safe monthly panel, Extra Trees emerged as the strongest single model, while Random Forest and ridge-regularized linear regression remained competitive under time-aware validation. XGBoost showed weaker performance in this setting. The proposed stacking framework, which combined MLR, Random Forest, Extra Trees, and XGBoost through a RidgeCV meta-learner, achieved the lowest RMSE (0.7689), indicating improved performance in reducing larger forecast errors compared with the individual models. This improvement, however, did not translate to MAE where several of the single models outperformed the stacked model and therefore indicates that the benefit from stacking in this study was larger with respect to stabilizing overall error than for absolute average error reduction. These results imply that forecast methods based on the principles of ensemble stacking under seasonal variation and evolving demand can be valuable retail forecasting strategy, especially given a validation by leakage-safe chronological validation of these methods. These results further imply that different non-homogeneous base learners enable the meta-learner to make use of predictive signals that are only imperfectly available to any one individual model.

This study has several limitations. Retail demand remains affected by zero inflation, sparse histories, and cold-start items, while the feature set does not yet incorporate richer exogenous information such as promotions, holidays, or macroeconomic factors. Future work should therefore examine adaptive retraining under drift, the inclusion of external covariates, and probabilistic forecasting approaches such as quantile prediction to better support inventory, replenishment, and staffing decisions.

AUTHOR CONTRIBUTION

AASG conceived and supervised the study. PY devised the methodology, developed the software, performed analysis, curated data, planned visualizations, supervised the project and wrote the manuscript. The validation, review and editing of the manuscript was performed by PY, AASG & HN. The authors have used ChatGPT for the purpose of language clarification and take full responsibility for the final content.

DATA AND MATERIAL AVAILABILITY

This study used the Predict Future Sales dataset, specifically the English-converted version curated by Darshan N and published on Kaggle. The dataset files are available through the corresponding Kaggle:
https://www.kaggle.com/datasets/ndarshan2797/english-converted-datasets/data.

REFERENCES

Adewumi, I. O. (2026). Predictive Analytics for Sales Forecasting in Emerging Market Retail: An Ensemble Learning and Time-Series Approach for A&Z Supermarket. Community and Social Development Journal, 27(1), 95–114. https://doi.org/10.57260/csdj.2026.286583

Al-Bazi, A., H. Al-Salami, Q., Zulfikar, N., Jerjees, Z., & A. Ahmad, M. (2025). Advanced Performance Analysis in Sport Footwear Sales Prediction Using Machine Learning. The 5th International Scientific Conference on Administrative and Financial Sciences (CIC-ISCAFS'2025), 14–21. https://doi.org/10.24086/icafs2025/paper.1582

Attipoe, E. K., Yussiff, A. S., Asante-Mensah, M. G., Tetteh, E. D., & Turkson, R. E. (2025). An ensemble learning approach for diabetes prediction using the stacking method. Computer Science and Information Technologies, 6(2), 102–111. https://doi.org/10.11591/csit.v6i2.p102-111

Bihri, H., Charaf, L. A., Azzouzi, S., & Charaf, M. E. H. (2025). A Robust Stacking-Based Ensemble Model for Predicting Cardiovascular Diseases. AI, 6(7), 160. https://doi.org/10.3390/ai6070160

Daza, A., Sánchez, C. F. P., Apaza-Perez, G., Pinto, J., & Ramos, K. Z. (2024). Stacking ensemble approach to diagnosing the disease of diabetes. Informatics in Medicine Unlocked, 44, 101427. https://doi.org/10.1016/j.imu.2023.101427

El-Boghdady, A. M., Kishk, S., Ashour, M. M., & AbdElhalim, E. (2023). Machine-learning based stacked ensemble model for accurate multi classification of CBC Anemia. Mansoura Engineering Journal, 49(3), 4. https://doi.org/10.58491/2735-4202.3144

Googerdchi, K. F., & Jafari, S. M. (2025). Sales Forecasting in Retail Using Random Forest Regression: A Case Study on Walmart. 2025 4th International Conference on Computing and Information Technology (ICCIT), 592–597. https://doi.org/10.1109/ICCIT63348.2025.10989314

Hasan, M. D. R. (2024). Addressing seasonality and trend detection in predictive sales forecasting: A machine learning perspective. Journal of Business and Management Studies, 6(2), 100–109. https://doi.org/10.32996/jbms.2024.6.2.10

Jafar, R., Awad, A., Hatem, I., Jafar, K., Awad, E., & Shahrour, I. (2023). Multiple Linear Regression and Machine Learning for Predicting the Drinking Water Quality Index in Al-Seine Lake. Smart Cities, 6(5), 2807–2827. https://doi.org/10.3390/smartcities6050126

Jodar-Abellan, A., Van Oudenhove, M., De Vente, J., Boix-Fayos, C., & Eekhout, J. (2025). Predicting bankfull channel dimensions through Stepwise Multiple Linear Regression and Random Forest in intermittent Mediterranean streams.   In EGU General Assembly Conference Abstracts (pp. EGU25-11060). https://doi.org/10.5194/egusphere-egu25-11060

Karaoğlu, A., & Söyler, H. (2025). A New Approach for Improving Biodiesel Conversion Efficiency: A Stacking Ensemble Model Based on Linear Regression Approach with GAN-Enhanced. Arabian Journal for Science and Engineering, 50(23), 19421–19441. https://doi.org/10.1007/s13369-025-10227-5

Kumar, M., Singhal, S., Shekhar, S., Sharma, B., & Srivastava, G. (2022). Optimized Stacking Ensemble Learning Model for Breast Cancer Detection and Classification Using Machine Learning. Sustainability, 14(21), 13998. https://doi.org/10.3390/su142113998

Li, X. (2025). A study on monthly sales forecasting of new energy vehicles in urban areas using the WOA-BiGRU model. PloS One, 20(4), e0320962. https://doi.org/10.1371/journal.pone.0320962

Mame, M., Huang, S., Li, C., & Zhou, J. (2025). Application of Extra-Trees Regression and Tree-Structured Parzen Estimators Optimization Algorithm to Predict Blast-Induced Mean Fragmentation Size in Open-Pit Mines. Applied Sciences, 15(15), 8363. https://doi.org/10.3390/app15158363

Manzanas, R. (2020). Assessment of Model Drifts in Seasonal Forecasting: Sensitivity to Ensemble Size and Implications for Bias Correction. Journal of Advances in Modeling Earth Systems, 12(3), e2019MS001751. https://doi.org/10.1029/2019MS001751

N, P. H., P, K. D., M, S., & R, S. M. (2025). Sales Data Visualization and Future Predictions Using Power BI and Machine Learning. 2025 IEEE International Conference on Compute, Control, Network & Photonics (ICCCNP), 1–6. https://doi.org/10.1109/ICCCNP63914.2025.11233722

Shoko, C., Sigauke, C., & Makatjane, K. (2025). An Application of Ensemble Stacking in Machine Learning to Predict Short-term Electricity Demand in South Africa. Statistics, Optimization & Information Computing, 13(6), 2412–2433. https://doi.org/10.19139/soic-2310-5070-2170

Wang, M., Liu, Y., Li, G., Yue, Y., & Man, K. L. (2024). Unlocking Your Sales Insights: Advanced XGBoost Forecasting Models for Amazon Products. International Conference on Advanced Multimedia and Ubiquitous Engineering, 181–187. https://doi.org/10.1007/978-981-95-1565-3_23

Wang, N., & Li, Z. (2022). A stacking-based short-term wind power forecasting method by CBLSTM and ensemble learning. Journal of Renewable and Sustainable Energy, 14(4). https://doi.org/10.1063/5.0097757

Windrasari, S. N., Margono, H., & Putra, Y. A. N. S. (2025). Predictive Sales Analysis in Coffee Shops Using the Random Forest Algorithm. MALCOM: Indonesian Journal of Machine Learning and Computer Science, 5(3), 1000–1011. https://doi.org/10.57152/malcom.v5i3.2023

Yao, B. (2023). Walmart Sales Prediction Based on Decision Tree, Random Forest, and K Neighbors Regressor. Highlights in Business, Economics and Management, 5, 330–335. https://doi.org/10.54097/hbem.v5i.5100