A Comparative Analysis of Sales Prediction Models: Evaluating the Efficacy of PHP-ML's SVR against Python's SVR and Linear Regression

Md Sayem Iftekar
Department of Computing and Games
Teesside University, Middlesbrough TS1 3BA.

Mohammed Aqib Zeeshan
Department of Computing and Games
Teesside University, Middlesbrough TS1 3BA

ABSTRACT
This research paper delves into the realm of business forecasting, specifically focusing on the implementation of Support Vector Regression (SVR) using the PHP-ML library. The study rigorously compares the performance of PHP-ML's SVR with Python's SVR and Linear Regression models, aiming to enhance understanding in the domain of sales prediction. Motivated by the need for accurate predictions in data-driven business environments, the research explores the practical effectiveness of PHP-ML as an alternative within the PHP ecosystem. The goal is to provide valuable insights into the viability of PHP-based machine learning solutions for improving business forecasting, addressing questions of accuracy, efficiency, and real-world applicability. The study employs historical sales data, preprocesses the dataset, and implements parameter tuned SVR models. Evaluation metrics such as Mean-Absolute-Error (MAE) and Root-Mean-Squared-Error (RMSE) are utilized for comparative analysis. The findings aim to contribute to the ongoing discourse on programming language and library selection for machine learning applications, providing practical guidance for businesses navigating predictive analytics complexities. Ultimately, the research assists decision-makers in making informed choices regarding the adoption of PHP-based machine learning solutions in sales prediction contexts.

Keywords
Comparative Analysis of Sales Prediction Models

1. INTRODUCTION
In the ever-evolving landscape of business forecasting, the quest for accurate sales prediction models remains paramount. This research paper undertakes a comprehensive exploration of the implementation of Support Vector Regression (SVR) using the PHP-ML library, positioning it against the established Python counterparts—SVR and Linear Regression. The primary objective is to assess the performance and accuracy of PHP-ML's
SVR in comparison to Python's SVR and Linear Regression, with a keen focus on enhancing business forecasting capabilities. By conducting a rigorous scientific analysis, this study not only contributes to the ongoing discourse on machine learning applications but also sheds light on the viability of PHP-based ML solutions in the domain of sales prediction. The insights gleaned from this comparative evaluation promise to inform practitioners and researchers alike, fostering a deeper understanding of the effectiveness of distinct machine learning techniques in the nuanced realm of sales forecasting.

2. MOTIVATION
In the dynamic landscape of data-driven business, accurate sales prediction is crucial for informed decision-making. This research compares PHP-ML's Support Vector Regression (SVR) with established Python models like SVR and Linear Regression. While Python dominates machine learning, PHP-ML offers an intriguing alternative within the PHP ecosystem. The study aims to understand PHP-ML's SVR strengths and limitations for sales prediction, providing insights to decision-makers. By bridging the gap between PHP and Python, it assists organizations in making informed choices for machine learning endeavours.

3. GOAL
This research aims to conduct a comparative analysis of sales prediction models, specifically comparing the Support Vector Regression (SVR) algorithm implemented using PHP-ML with Python's SVR and Linear Regression models. The study will evaluate their performance and accuracy metrics to assess the effectiveness of PHP-ML's SVR in sales prediction. The goal is to understand the practicality of PHP-based machine learning solutions for business forecasting and how they compare to established Python models in terms of accuracy, efficiency, and applicability. The insights from this comparison will contribute to the discourse on the selection of programming languages and libraries for machine learning applications, particularly in sales forecasting, and aid decision-makers in choosing PHP-based machine learning solutions for predictive analytics in business.

4. METHOD
This study utilizes the Support Vector Regression (SVR) algorithm from the PHP-ML library to predict future sales, employing historical sales data for training and evaluation. The dataset undergoes preprocessing to handle missing values and ensure feature normalization. The PHP-ML SVR model is implemented with parameter tuning for optimal performance. Comparative analysis is conducted against Python's SVR and Linear Regression models, considering Mean-Absolute-Error (MAE) and Root-Mean-Squared-Error (RMSE) as evaluation metrics. Results are interpreted to identify the most accurate and reliable model for predicting future sales based on historical data, with practical implications discussed for integrating the chosen model into ERP systems. The research concludes by summarizing findings and advocating for the adoption of the most effective model, as determined by the comparative analysis of PHP-ML's SVR against Python counterparts.

5. METHODOLOGIES
The Support Vector Regression (SVR) algorithm from the PHP-ML library to predict future sales based on historical data. To evaluate the effectiveness of the PHP-ML library's SVR model, its performance is compared with the SVR and Linear Regression models implemented in Python. The comparison metrics used are Mean-Absolute-Error (MAE) and Root-Mean-Squared-Error (RMSE), which provide insights into the accuracy of the models. The results of this comparison will help determine the most suitable model for sales prediction.

5.1 Support-Vector-Regression (SVR)

Utilizing the PHP-ML library's Support Vector Regression (SVR), this research aims to predict sales for the next five years using two datasets. SVR, as a machine learning model, captures the relationship between input variables and a continuous target variable, making it suitable for both linear and non-linear regression. Past studies indicate the efficacy of SVR in sales forecasting, particularly when optimized with the Grey Wolf Optimizer (GWO) method. However, its effectiveness is contingent on the quality of sales data and its adherence to SVR's assumptions, emphasizing the importance of data quality for accurate predictions. [1]

5.2 Linear-Regression in Python

Linear Regression in Python is a key machine learning algorithm used to predict a dependent variable (y) based on an independent variable (x), establishing a linear relationship between them. It's implemented using the `sklearn` library, which also supports other types of regression, classification, and clustering tasks. Models are typically fitted using the least squares approach, but other methods like ridge regression and lasso regression are also used. Applications include predicting house prices, stock prices, weather, and sales forecasting. [2]

5.3 Why choose PHP ML Library

The PHP-ML library's Support Vector Regression (SVR) is chosen for regression tasks in a PHP environment due to its easy integration with PHP applications, support for different kernels like linear, polynomial, and radial basis function (RBF), and customizable parameters. It's a powerful tool for predicting continuous outcomes in various applications, including sales forecasting and stock price prediction. [3]

5.4 Model formation

There are two models applied for the research such as Support-Vector-Regression (PHP-ML & Python) and Linear-Regression (Python).

5.4.1 Support-Vector-Regression in PHP-ML

The implementation's script first connects to the SQLite database, fetches the relevant data, and then splits it into training and testing sets. For each unique product, a separate SVR model is trained using the training data. After training, the script evaluates the model's performance on the test set, calculating metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and provides predictions for the next five years. The predictions are organized in a table, displaying the anticipated quantities for each product on a monthly basis, along with the total predicted quantity for each month. [4]

5.4.2 Support-Vector-Regression in Python

The script connects to an SQLite database, retrieves sales data, and organizes it into feature vectors and corresponding labels. After splitting the data into training and testing sets, separate SVR models are trained for each unique product. The script evaluates the models on the test set, calculating metrics such as Mean Absolute Error (MAE), R^2 Score, and Root Mean Squared Error (RMSE). Predictions are made for the next five years, with results presented in a tabular format. The implementation leverages scikit-learn's SVR and train-test split functionalities, demonstrating a concise approach to building and evaluating a predictive model for sales forecasting. The predictions are then visualized in an HTML table for easy interpretation.

5.4.3 Linear-Regression in Python

The Linear Regression model trained separately for each unique product in the dataset. The training data for each product consists of the month and year of sales, and the labels are the quantities sold. The model is then used to predict sales quantities on a test set, and the Mean Absolute Error (MAE), R^2 Score and Root Mean Square Error (RMSE) score are calculated for these predictions. Finally, the model is used to predict sales for the next 5 years for each product. The predictions are stored in a Data Frame and displayed in a table. The model is formatted as a dictionary where the keys are the unique products, and the values are the trained Linear Regression models. [2]

5.5 Model Deployment

In the research implementation, both Support-Vector-Regression (SVR) and Linear-Regression model are used to predict future sales quantities for various products based on historical sales data. Here's an explanation of how the both SVR and Linear-Regression models are used to resolve the problem:

5.5.1 Model applied with PHP-ML(SVR) and Python SVR implementation

The PHP-ML (SVR) and Python SVR implementations for predicting sales quantities based on historical data from an SQLite database follow a parallel structure. Both connect to the database, retrieve sales data, and preprocess it by organizing features and labels. Data splitting is done uniformly, with training and testing sets established. Model training involves employing SVR with a linear kernel for each unique product. Prediction on the test set is conducted, and various evaluation metrics are calculated. Future predictions extend over five years for each product. While the implementations differ in syntax, libraries, and presentation details, they converge in their overarching structure, demonstrating a consistent approach to solving the sales prediction problem with support for both PHP-ML and Python SVR. [5]

The overall model formation and flow between PHP-ML(SVR) and Python SVR and Linear regression implementation is same.

5.6 Sales prediction data format

The historical data, stored in an SQLite database, serves as the model's input. It is a lightweight relational database management system, offers a small binary footprint, minimal disk space usage, and less runtime configuration, yet can handle large data volumes. It stores data systematically in tables, rows, and columns within a single file, which can handle databases up to 2 terabytes and is portable across systems and networks. [6]

Two datasets have used for the experiments, such as dummy data demonstrating consistent upward trends and data on 'Supermarket sales' from Kaggle. The same data format used to train and predict the sales quantity. [7]

5.7 Data conversion

In the PHP implementation, the data conversion process begins with establishing a connection to an SQLite database and fetching records from the sales_data table. The fetched data is organized into arrays for features, labels, and product information. Subsequently, the dataset is split into training and testing sets, and separate Support Vector Regression (SVR) models are trained for each unique product. Predictions are made for the test set, and evaluation metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are calculated. Future predictions for the next 5 years are generated for each product. The final step involves outputting the predictions in an HTML table format. On the other hand, the Python implementation uses SQLite and machine learning libraries, such as scikit-learn and pandas. The data is fetched through SQL queries and processed directly into lists, arrays, and dictionaries. Train-test splitting, SVR model training, predictions, and performance metric calculation is performed, with results stored in appropriate data structures. The final predictions are displayed in an HTML table using a pandas DataFrame for enhanced visualization. Both implementations involve systematic data conversion, transforming raw database results into structured formats suitable for training, evaluating, and presenting SVR models.

5.8 ML based sales prediction data cleaning

While data cleaning is an essential step in many data analysis and machine learning projects, it is not explicitly addressed in this module as it will be trained, and predictions will be performed with dummy data. If the source data in the database is known to be potentially dirty or requires cleaning, additional data cleaning steps should be performed before using it for modelling. Data cleaning operations are highly dependent on the specific characteristics and quality of the dataset, and they typically involve domain-specific knowledge and decisions. So, data will be carefully examined, such as study the generated data to understand the data types and the structure of the dataset, removing any duplicates which assist reducing bias, handling missing values, make sure the correct data format on each column. In both the PHP and Python implementations, the data cleaning process involves fetching sales data from an SQLite database and organizing it for further analysis. The data is retrieved from the 'sales_data' table, containing information about month, year, quantity, and product. In PHP, a PDO connection is established, and data is fetched using a query. The fetched data is then processed by populating arrays for features (`$data` in PHP, and `data`, `labels`, and `products` in Python), filtering out null quantities, and creating separate arrays for training and testing sets. In Python, the `train_test_split` function is used to split the data into training and testing sets. Both implementations involve filtering and organizing data based on products and then training Support Vector Regression (SVR) models for each product. The cleaning process ensures that the data is suitable for training and evaluating the regression models.

5.9 ML based sales prediction Data size

The data set contains sales data of last twelve years (144 months) of two sample items. Items are identified as 'Product A' and 'Product B'. The data table contains the following columns such as id, month, year, product, quantity, and branch. There will be about 288 rows of data in the dataset. The sales quantities of both "Product A" and "Product B" have generally increased over the years, indicating a growing demand. Despite monthly fluctuations, "Product A" sales grew from 50 in January 2012 to 380 in December 2023, and "Product B" sales rose from 60 in January 2012 to 390 in December 2023, demonstrating consistent upward trends.

5.10 Data Size justification

The data size for training models, such as Support-Vector-Regression (SVR) in PHP-ML and Linear Regression in Python, are typically determined based on the specific requirements of the model and the problem at hand. Here are some factors to consider:

5.10.1 Complexity of the Problem

More complex problems may require more data for the model to learn effectively. Perhaps, for complex non-linear predictions, more data is often necessary compared to simple linear relationships.

5.10.2 Variability in Data

If there is a lot of variability in the data, then it probably needs more data to capture all the different variations.

5.10.3 Model Performance

It is a better idea to start with a smaller dataset and gradually add more data while monitoring the performance of the model. If the performance continues to improve with more data, it might be beneficial to add more.

5.10.4 Computational Resources

Training on larger datasets requires more computational resources and time. So, the available resources might also dictate the size of the dataset. [8]

5.11 Parameter settings (for PHP-ML & Python)

In both the PHP and Python implementations, the parameter settings for the Support Vector Regression (SVR) model are relatively straightforward. Both implementations use the linear kernel for the SVR model, which is specified as Kernel::LINEAR in PHP and kernel='linear' in Python. This choice indicates that a linear relationship between features and the target variable is assumed. Additionally, both implementations use the default settings for other SVR parameters, such as regularization parameters (C and epsilon), as these are not explicitly specified in the code. The training of individual models for each product is performed in a loop, ensuring that the SVR models are tailored to the specific characteristics of each product. While the implementations differ in syntax and specific library usage (PHP-ML in PHP and scikit-learn in Python), the fundamental parameter choices for the SVR model remain consistent, with an emphasis on a linear kernel for this regression task. [9]

5.11.1 Justification for the parameter's settings

The selection of parameter settings for Support Vector Regression (SVR), specifically employing a linear kernel with default regularization parameters (C and epsilon), is grounded in the presumption of a linear association between input features and the target variable. The linear kernel, denoted as `Kernel::LINEAR` in PHP and `kernel='linear'` in Python, is appropriate when anticipating a linear pattern in the data, simplifying the model without undue complexity. This choice is common for regression tasks, particularly when there is a reasonable belief in a linear relationship.

Default settings for other SVR parameters, including regularization parameters C and epsilon, are adopted under the assumption that default values in libraries (PHP-ML in PHP and scikit-learn in Python) generally suit a broad spectrum of scenarios, providing balanced and stable performance across datasets without manual tuning.

The strategy of training individual models for each product in a loop supports the use of default parameter settings. Tailoring SVR models to each product's unique characteristics through individualized training allows adaptation to diverse datasets, and the linear kernel serves as a flexible yet straightforward choice for capturing linear relationships within each product's data.

In essence, the rationale for these parameter settings lies in the anticipation of a linear relationship, aligning with the choice of a linear kernel. Default values for regularization parameters enhance this choice by offering a stable starting point, and individualized training in a loop ensures adaptability to each product's distinctive features.

6. EXPERIMENTS & RESULTS

6.1 Sales predictions

The section details experimentation with PHP-ML's SVR, Python's implementation with SVR and Linear-Regression (LR) for sales predictions. Variations are such as, yearly predictions for all items and each item, yearly and monthly predictions for each item, all projected for the next five years from the last available year in the dataset in tabular form and a bar chart graphically represents the yearly predictions for all items.

6.2 Sales-Prediction Accuracy Measure

6.2.1 Python-implementations vs PHP-ML(MAE):

The Mean Absolute Error (MAE) is a common metric for evaluating the accuracy of regression models, measuring the average absolute difference between predicted and actual values. In PHP-ML and Python, the MAE is calculated by averaging the absolute differences between the actual and predicted values. An MAE of PHP-ML-SVR = 5.08 / Python's Linear Regression = 5.10 / Python-SVR = 5.78 means that, on average, the model's

predictions for sales quantity are about 5.08 / 5.10 / 5.78 units away from the actual values. Smaller MAE values indicate a more accurate model, while larger values suggest less accuracy. The MAE can be used to compare different models or track the performance of the model over time.
The formula for MAE is:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

where:

- n is the number of predictions,
- y_i is the actual quantity for the i-th prediction, corresponds to $productLabels[$i],
- $\hat{y}_i$ is the predicted quantity for the i-th prediction obtained from the regression model.

6.2.2 *Python-implementations vs PHP-ML(RMSE)*

Python implementations with SVR and Linear-Regression showed near 99% identical results on all three prediction types such as total sales quantity by year for all products and each product, every month of each year for each product. The PHP-ML library's SVR model has the lowest MSE, RMSE by percentage and therefore is the most accurate among the three models.

It is calculated as: RMSE = $\sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$

Where:

- n is the number of data points (total number of squared errors).
- y_i is the actual label (ground truth) for the i-th data point($productLabels).
- $\hat{y}_i$ is the predicted label for the i-th data point ($regressions model). [10]

6.3 Dataset vs Predictions trends

Models are predicting sales over the course of the year, represented by green bars, based on historical sales data from both datasets with dates and quantities that show an upward pattern in the data pattern with the dummy dataset and the "Supermarket Sales" dataset shows a little ups and downs but consistent sales over the years, which are represented by blue bars.

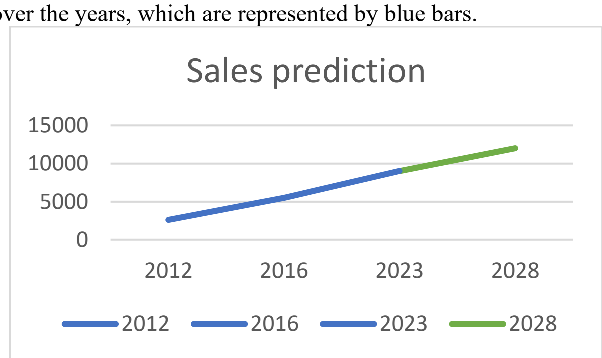

Fig 1: Historical vs prediction data with Dummy dataset.

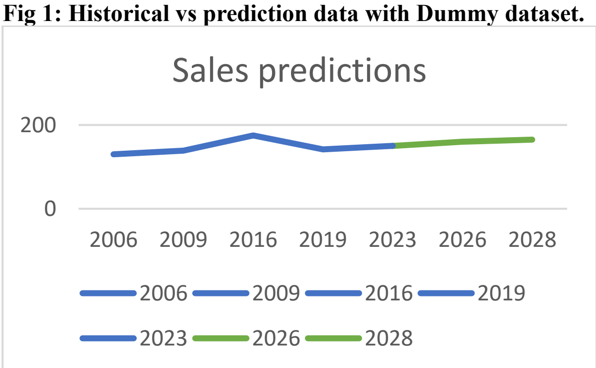

Fig 2: Historical vs prediction data with Dummy data with 'Supermarket Sales' dataset.

6.4 Evaluations upon the experimental results

The models showed slightly different accuracy between PHP-ML Library's (SVR) and Python's implementation of SVR, Linear-Regression (LR), predicting higher sales over the years in line with the increasing trend in the historical data. However, these models provide predictions, not guarantees, and their accuracy can be affected by significant changes or external factors. The accuracy improves with larger, more precise, and comprehensive data. PHP-ML's SVR model shows MAE score 5.08 where Python's SVR implementation shows 5.78 respectively which is approximately 12.92% and Python's Linear-Regression implementation shows 5.10 which is 0.39% difference against PHP-ML's SVR. An R-squared score of 99 indicates that 99% of the variation in the dependent variable is explained by the independent variables, suggesting a good model fit and highly predictable, but it could also imply potential overfitting, which may affect performance on new data.
The model scores are as follows: PHP-ML SVR Model scored 2.44, Python SVR Model achieved 2.31, and Python Linear Regression Model attained 2.30. These scores reflect the performance of each model, with lower values indicating better accuracy or less error. Again, the bar graph for the Kaggle's 'Supermarket Sales' dataset illustrates that Python Linear Regression has the lowest Mean Absolute Error (MAE) among three different machine learning models.

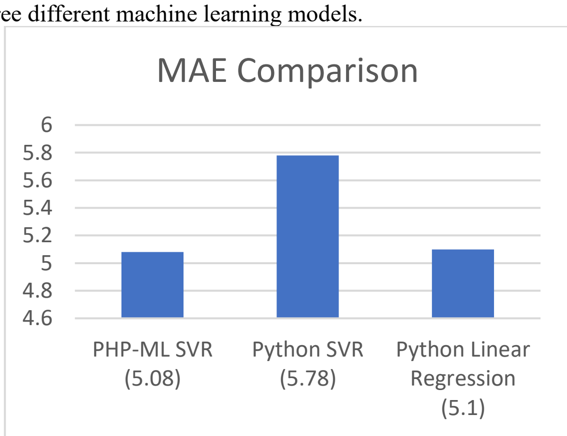

Fig 3: MAE scores using with Dummy dataset.

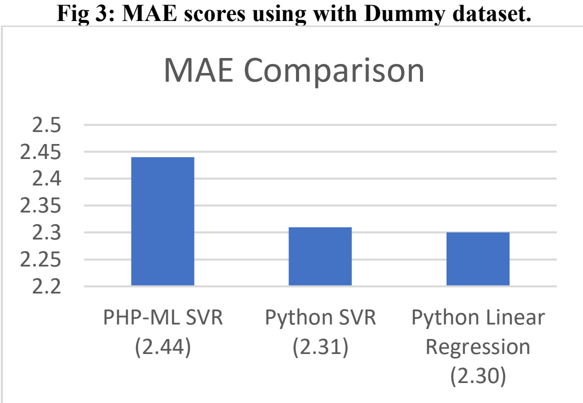

Fig 4: MAE scores using with 'Supermarket Sales' dataset.

The Root Mean Square Error (RMSE) is a measure of a model's accuracy in predicting quantitative data, with lower values indicating better fit. With dummy dataset, among the three models, the PHP-ML library's SVR model has the lowest RMSE score of 7.14, making it the best fit. Then, Python's SVR scored 7.4, and at last Python's linear regression scored 7.68. However, RMSE values don't provide a direct measure of the model's coefficient of determination (R^2), which would need to be calculated separately. Also, a lower RMSE doesn't necessarily mean the model is good, as context and domain understanding are crucial when interpreting these values.

The bar graph compares the performance of three machine learning models on the 'Supermarket Sales' dataset. Python Linear Regression achieved the lowest RMSE score of 2.74, followed closely by Python SVR with a score of 2.75, while PHP-ML SVR had the highest score of 2.88.

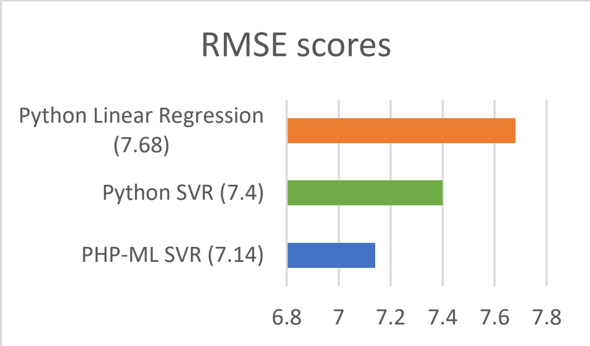

Fig 5: RMSE scores with dummy dataset

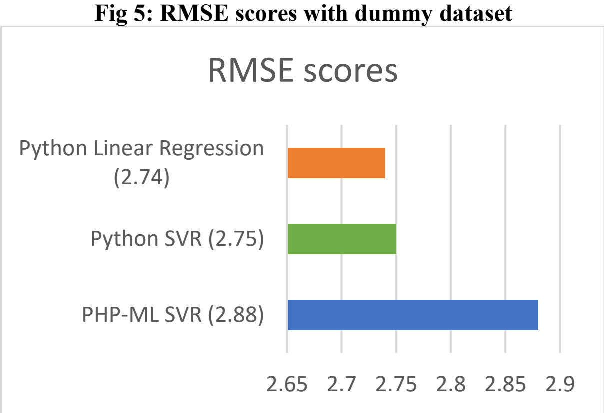

Fig 6: RMSE scores with 'Supermarket Sales' dataset.

7. VERDICT

Even though after keeping the same model formation, parameter settings, feature selection and dataset the identical ML algorithms between the two language and library showing different accuracy. There could be several other reasons why the MAE scores are different between the PHP-ML library's Support vector regression and Python's library of support vector regression. Here are some possible factors that could affect the results:

- The implementation details of the SVR algorithm in each library. Different libraries may use different optimization methods, numerical precision, or default settings for some parameters that are not explicitly specified by the user. For example, the PHP-ML library uses the LIBSVM library under the hood, while the Python library uses the scikit-learn package. These two libraries may have different approaches to solving the SVR problem, which could lead to different outcomes.

- The data preprocessing and feature scaling steps. Before applying SVR, it is important to preprocess and scale the data properly, as this can affect the performance and accuracy of the model. Different libraries may have different ways of handling missing values, outliers, categorical variables, or normalization. For example, the PHP-ML library requires the user to manually scale the data using the StandardScaler class, while the Python library can automatically scale the data if the parameter scale is set to True. These differences could result in different input values for the SVR model, which could affect the MAE and RMSE scores.

- The random seed or initialization of the SVR model. SVR is a stochastic algorithm, which means that it involves some randomness in its execution. This randomness can come from the initialization of the model parameters, the selection of the support vectors, or the optimization process. Different libraries may use different random seeds or initialization methods, which could lead to different results. To ensure reproducibility, it is advisable to set the random seed or the initialization method to a fixed value in both libraries and compare the results using the same seed or method.

- Another reason can be 'Random Shuffling' cannot be applied in PHP-ML's implementation by default as it does not provide a library like scikit-learn thus it creates programming difficulty as it is needed to be applied by coding. The train_test_split function from scikit-learn, when used without specifying the shuffle parameter, shuffles the data by default. This means that the function will randomly shuffle the data before splitting it into training and testing sets.

A thorough examination of each library's code and documentation was conducted to compare the parameters and steps involved in the SVR process and determine the precise cause of the variation in the MAE and RMSE scores.

8. CONCLUSION

In summary, the comparative analysis of sales prediction models between PHP-ML's SVR and Python's SVR and Linear Regression revealed notable discrepancies in MAE and RMSE scores, despite maintaining consistent model parameters and dataset. Variations in the underlying SVR algorithm implementation, data preprocessing methods, and the absence of default 'Random Shuffling' in PHP-ML contribute to contradictory outcomes. The utilization of LIBSVM in PHP-ML and scikit-learn in Python introduces nuanced differences in optimization methods and numerical precision, while manual scaling requirements in PHP-ML versus scikit-learn's automatic scaling add another layer of distinction. The stochastic nature of SVR, coupled with differences in random seed or initialization methods, emphasizes the importance of ensuring result reproducibility through consistent practices across libraries. These findings underscore the necessity for meticulous consideration when applying machine learning models, offering valuable insights for researchers and practitioners working with diverse programming languages and libraries in the realm of sales prediction.

9. REFERENCES

[1] Qu, F. et al. (2022) 'Forecasting of Automobile Sales Based on Support Vector Regression Optimized by the Grey Wolf Optimizer Algorithm', Mathematics, 10(13) Available at: https://doi.org/10.3390/math10132234 (Accessed: 9 November 2023)

[2] T., G., Choudhary, R. & Prasad, S. 2018, "Prediction of Sales Value in Online shopping using Linear Regression", IEEE, , pp. 1.

[3] PHP-ML (2020) Support Vector Regression. Available at: https://php-ml.readthedocs.io/en/latest/machine-learning/regression/svr/ (Accessed: 9 November 2023).

[4] Kondas, A. (2020) 'Epsilon-Support Vector Regression,' PHP-ML [online]. Available at: https://php-ml.readthedocs.io/en/latest/machine-learning/regression/svr/

[5] Vashistha Nitish (2023) 'PHP-ML – Machine Learning library for PHP', Available at: https://webkul.com/blog/php-ml-machine-learning-library-for-php/

[6] Gaffney, K.P., Prammer, M., Brasfield, L., Hipp, D.R., Kennedy, D. & Patel, J.M. 2022, "SQLite: past, present, and future", Proceedings of the VLDB Endowment, vol. 15, no. 12, pp. 3535-3547.

[7] Pyae, A. (2023) Supermarket sales. Available at: https://www.kaggle.com/datasets/aungpyaeap/supermarket-sales?resource=download (Accessed: 15/03/2024).

[8] Fahrudin, T., Wisna, N., Telnoni, P.A. & Wijaya, D.R. 2022, "Sales Forecasting Web Application in Small and Medium Enterprise", IEEE, pp. 213.

[9] Bhattacharyya, I. (2018) Support Vector Regression Or SVR. Available at: https://medium.com/coinmonks/support-vector-regression-or-svr-8eb3acf6d0ff (Accessed: 09/11/2023).

[10] Hodson, T.O. 2022, "Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not", Geoscientific Model Development, vol. 15, no. 14, pp. 5481-5487.