TIME SERIES FORECAST OF CALL ARRIVALS USING MACHINE LEARNING METHODS

Peerawit Surasai[1*], Vera Sa-Ing[2]

Abstract

This study focuses on enhancing workforce management in the Citizen Service Request (CSR) Call Center dataset of the government of Cincinnati, Ohio, by improving the accuracy of call arrival forecasts. Recognizing the pivotal role of precise call arrival predictions in optimizing call center operations, this the study conducts experiments by utilizing a range of forecasting models, including statistical, machine learning, and neural network approaches. Feature engineering was proposed to broaden the scope of features for forecasting. The top-performing models are evaluated based on key metrics such as Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and R-Squared (R^2) forecasting performance. The experimental results highlighted the comparative performance of various models, such as SARIMAX, Light Gradient Boosting Machine (Light GBM), Gradient Boosting Regressor (GBR), eXtreme Gradient Boosting (XGBoost), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU). Among these, Support Vector Regression (SVR) leads in accuracy with an MAE of 25.13, an MAPE of 6.15%, an RMSE of 34.46, and an R^2 of 90.56%. The features of abandon rate, answer speed, service level calls, and the 1st and 5th lags, were identified as the most importance feature in this research. These findings provide valuable insights for the improvement of workforce management strategies in call center operations, emphasizing the effectiveness of machine learning algorithms in achieving more accurate call arrival forecasts.

Keywords : Workforce Management, Machine Learning, Support Vector Regression

[1] Data Science, Faculty of Science, Srinakharinwirot University, Bangkok, 10110, Thailand

[2] Faculty of Science, Srinakharinwirot University, Bangkok, 10110, Thailand

* Corresponding author: Tel.: 080-5488355 E-mail address: aekpreya.baisani@g.swu.ac.th

Background

1. Introduction

Workforce Management (WFM) is a strategic approach that organizations employ to optimize the efficiency and productivity of their workforce. It encompasses various key components, including workforce planning, scheduling, time and attendance management, task assignment, performance management, and training and development. Workforce planning involves forecasting future needs and identifying talent gaps, while scheduling ensures that the right employees, with the right skills, are in the right place at the right time. Time and attendance management tracks working hours and ensures compliance with labor regulations. Task assignment optimizes productivity by matching employees to tasks based on skills and availability. Performance management aligns individual performance with organizational goals, and training and development enhance workforce skills. Technology, such as workforce management software, plays a crucial role in automating and streamlining these processes. The overarching goal of Workforce Management is to create a dynamic and agile workforce that can adapt to changing business needs, ultimately contributing to enhanced operational performance and organizational success.

Forecasting in a call center is a critical component of effective workforce management, aiming to predict and plan for future customer interaction volumes. It involves a comprehensive analysis of historical data, trends, and various factors influencing call volumes to make accurate predictions. By examining past call patterns and considering external variables such as marketing initiatives or economic factors, organizations can anticipate peak times and plan staffing levels accordingly. Regular monitoring and adjustment are essential for ensuring that forecasts remain aligned with real-time data and changing conditions. Accurate forecasting contributes to improved customer service by minimizing wait times, reducing the likelihood of abandoned calls, and enhancing overall operational efficiency.

2. Statistical method and Machine Learning Models

2.1 SARIMAX

SARIMAX, an extension of SARIMA **(Box & Jenkins, 1976)** known as Seasonal AutoRegressive Integrated Moving Average with Exogenous Factors," allows for incorporating variables into the modeling process. The factors listed here are examples of exogenous variables that could have an impact on the time series under study. Three elements that are comparable to SARIMA make up the SARIMAX model.

The SARIMAX model incorporates a Seasonal Component (S), acknowledging and reproducing recurring cycles or patterns in the time series data. It comprises AutoRegressive (AR) and Moving Average (MA) components, where the MA component models the relationship with residual errors, and the AR

component illustrates the connection with prior values. Additionally, SARIMAX introduces Exogenous Variables (X), external factors like holidays or advertising budget, enhancing the model's capacity to account for changes in the time series.

2.2 Support Vector Regression (SVR)

A machine learning technique called Support Vector Regression (SVR) was created especially for handling regression tasks involving complex and non-linear relationships between the target variable and the input features. In contrast to linear regression, support vector regression (SVR) uses a kernel trick to map the input features into a higher-dimensional space, which makes it possible to identify the best hyperplane for accurately representing the underlying relationships.

2.3 Light GBM (Light Gradient Boosting Machine)

A model called Light Gradient Boosting Machine, or Light GBM, is intended to effectively train big datasets. It is designed especially for gradient boosting and is a member of the learning algorithm family. Light GBM differs from other tree-growing methods in that it follows a leaf-based approach rather than a depth-first one. This method lowers the model's complexity.

2.4 Gradient Boosting Regression (GBR)

Another popular ensemble learning method for time series forecasting is gradient boosting regression (GBR). Usually, the process starts with establishing a prediction, which is the target variable's average. Then, by training them using the negative gradient of the loss functions predictions for the current model, the algorithm creates learners frequently using shallow decision trees. To account for the differences between the predicted values, residuals are computed, and the weak learners prediction is modified and incorporated into the current model. This process is repeated, with each weak learner concentrating on fixing mistakes in the previously constructed ensemble.

2.5 XGBoost (eXtreme Gradient Boosting)

When it comes to tasks like regression, classification, and even time series forecasting, XGBoost is a highly effective and popular machine learning algorithm. In order to produce a final prediction that is

both dependable and accurate, XGBoost functions as a learning technique that integrates the predictions from several weak models, most often decision trees.

The secret to the algorithms' performance is their capacity to manage relationships in the data while retaining a high degree of predictive accuracy. Boost converts data into features that include lag values and pertinent external factors in the context of time series forecasting. After that, a series of decision trees are built, each tree repairing the mistakes made by the previous ensemble.

2.6 Recurrent Neural Network (RNN)

Artificial neural networks that process sequential data by preserving an internal state or memory are known as recurrent neural networks, or RNNs. In contrast to feedforward neural networks, which process input data in a single pass, RNNs are capable of handling sequences of arbitrary length and maintaining information over time. **(Werbos, 1988)** is regarded as the most thorough source of information on recurrent neural networks (RNNs).

Nevertheless, the vanishing gradient problem presents a difficulty for conventional RNNs. This issue limits their capacity to accurately identify long-term dependencies. Advanced RNNs have been developed, such as the Gated Recurrent Unit (GRU) and Long Short Term Memory (LSTM), to get around this restriction. These versions overcome the limitations of RNNs and improve their ability to identify and leverage long-term dependencies.

2.7 GRU (Gated Recurrent Unit)

The GRU, a type of network (RNN) is widely used for tasks like predicting call arrival volume in sequence modeling and forecasting. It's a variant of RNN that addresses the vanishing gradient problem through gating techniques. An overview of the GRU model's elements is provided below:

- **Hidden State**: The GRU stores data from previous time steps in a hidden state that corresponds to the model's memory.
- **Update Gate**: The update gate controls how much old data should be. How much new information should be incorporated into the hidden state.
- **Reset Gate**: The reset gate determines how much of the state to ignore when calculating the current hidden state. It allows the reset or deletion of data.

- **Candidate Activation**: At each time step the candidate activation calculates a hidden state by combining information from the hidden state and the current input.
- **Final Hidden State**: is determined by using the update gate to blend the hidden state with the candidate activation. This represents updated memory or information, for that time step.

2.8 LSTM (Long Short-Term Memory)

Long-term dependencies in sequential data, like time series data, can be captured using long-term support graph networks, or LSTMs. They are appropriate for modeling patterns because they have the capacity to selectively remember or forget information over extended periods of time. In this research paper a new type of neural network (RNN) architecture was proposed by **(Hochreiter & Schmidhuber, 1997)**. This innovative architecture incorporated memory cells and gating mechanisms, which addressed the problem of vanishing gradients commonly encountered in RNNs. The elements of an LSTM network are as follows:

- **Forget gate**: The forget gate controls the flow of information from the previous hidden state to the current hidden state. It determines which past information is still relevant and should be retained while filtering out data.
- **Input gate**: The input gate controls how much new information is incorporated into the cell state from the current input. It decides which aspects of the current input are relevant and should be incorporated into the cell state.
- **Cell state update**: To update the cell state, the cell state update combines information from the forget gate, input gate, and current input. The cell state is the LSTM's core memory component, and it maintains long-term dependencies in sequential data.
- **Output gate**: At the current time step, the output gate determines which information from the cell state is used to produce the output. It determines how much information about the cell state is exposed to the network's subsequent layers.

Methodology

1. Data Understanding

(Services, 2023) contains information on Citizen Service Request (CSR) call center calls in Cincinnati, Ohio. The dataset includes information on the caller's type of service request, the time and date of the call, the location of the service request for help, and the request's status.

There are 856,405 rows representing 23 columns from 2015 to 2022. The data is updated daily and is able to filter by date, location, status, and service request type. **Figure 1** depicts the monthly trend to demonstrate the fluctuation. The dataset presents useful insights into the types of issues that Cincinnati citizens face, as well as the level of service offered by the CSR call center.

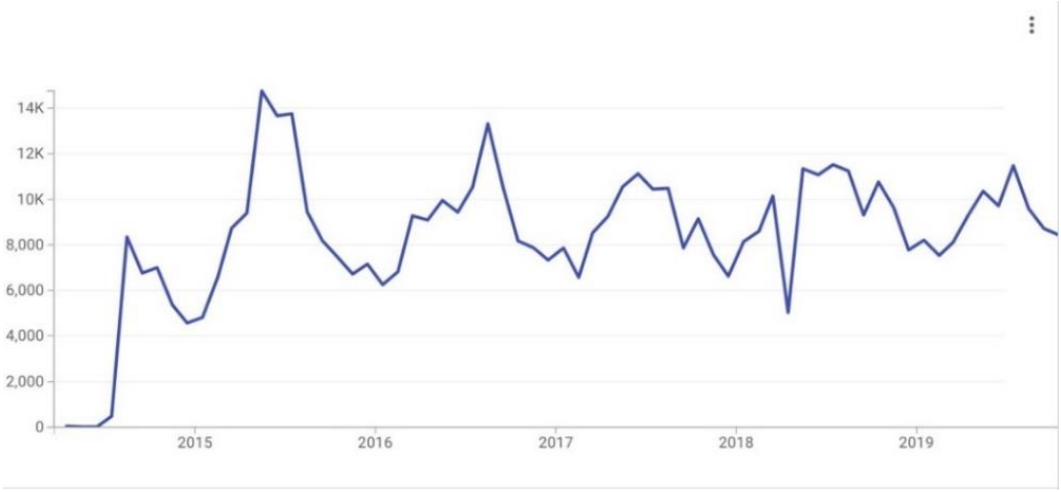

Figure 1 Monthly call trend of the Citizen Service Request (CSR) dataset from 2014 – 2021

From 2014 to 2022, the department's call arrivals follow a yearly pattern. Every July or Summer, the volume was higher than in other months. The number of calls ranged from 86 to 710, with an overall average of 403. Missing values are linearly interpolated in 2018. In early 2020, there was a drop trend, which could have been caused by a factor affecting call arrivals at the time.

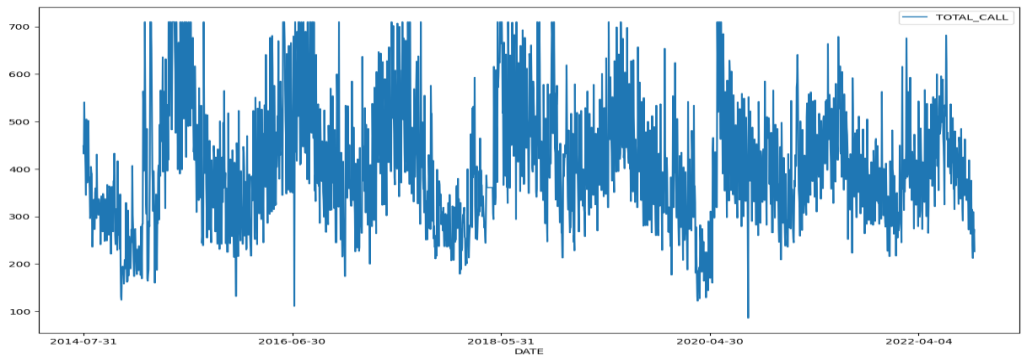

Figure 2 Total call per day visualization

The total calls in summer 2015, the second year of service, are highly fluctuated from the original values of over 1,400 calls. Outlier calls can be handled by setting minimum and maximum bounds, as well as zero values, which are repeated every weekend and holiday. Figure 9 depicts the results of the histogram of total daily calls.

2. Data Pre-processing

Pre-processing involved transforming raw data from 856,430 calls into a daily format. The series was refined by trimming both the initial and concluding segments to eliminate outliers. The resulting dataset spans from July 31, 2014, to October 14, 2022, covering a total of 2,135 days, with holidays excluded as depicted in **Figure 2** of the data frame.

Daily summaries of call data are generated by calculating the total number of calls, abandoned calls, and answered calls, as well as the average service level. Additionally, average values for relevant metrics like answer speed, talk time, and wrap time are computed. Finally, abandonment and answer rates are derived from the respective call counts. Data for weekdays (Monday to Friday) is used for this analysis, as days with no calls (holidays and days off) are excluded.

3. Feature Engineering

This research leverages time and lag features to grant the model the power to discern the intricate relationships between TOTAL_CALL and other features, ultimately revealing the key factors that contribute to accurate prediction. These features capture the essential temporal patterns and dependencies, significantly

enhancing the model's grasp of the data and its predictive capabilities. The correlation of all features shows in **Figure 3**.

- Temporal Features: DAY_OF_WEEK, MONTH_DAY, YEAR_DAY, WEEK_YEAR : These features directly extracted from the 'DATE' column provide contextual information about the day of the week, month, year, and week of the year for each entry.

- Seasonal Features: Binary indicators represent the seasons (FALL, SPRING, SUMMER, WINTER). Each feature is assigned 1 if the entry falls within the corresponding season and 0 otherwise.

- Monthly Features: Similarly, binary features represent each month of the year (JAN, FEB, ..., DEC). An entry receives 1 for its respective month and 0 for other months.

- Yearly Features: YEAR_2014, YEAR_2015, ..., YEAR_2022: features differentiate the specific year of each entry (YEAR_2014, YEAR_2015, ..., YEAR_2022). The feature corresponding to the entry's year is set to 1 and others to 0.

- Day of the Week Features: Each day of the week is represented by a binary feature (MONDAY, TUESDAY, ..., SUNDAY). The feature relevant to the entry's day is assigned 1 and the rest 0.

- Lag1 to Lag5: These features capture 5 days of seasonality (Monday to Friday) and aim to learn any potential weekly patterns affecting call volume.

- Lag10 to Lag30: This range of lag features (2 to 6 weeks) investigates the influence of longer-term historical data on future call volumes.

- TOTAL_CALL_LAG_1 to TOTAL_CALL_LAG_30: These features represent the specific lag observations created for 'TOTAL_CALL', providing the model with historical information at different timeframes.

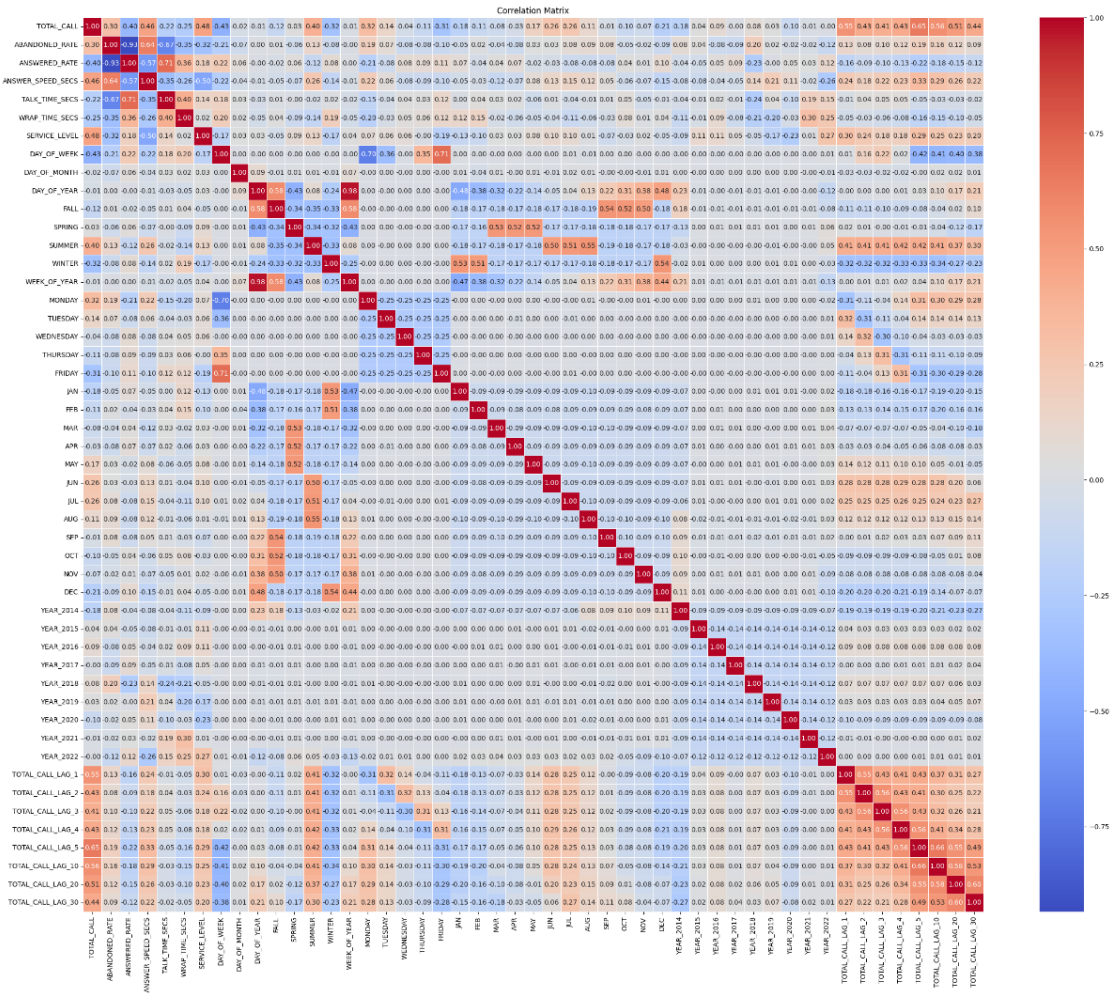

Figure 3 Correlation of total 48 features

4. Statistical Test

The Augmented Dickey-Fuller (ADF) : This test investigates at the presence of a "unit root," a sign of non-stationarity, in a time series. Potential stationarity is suggested by a rejected null hypothesis of non-stationarity, highlighting the significance of differencing in achieving stationarity.

KPSS Test (Kwiatkowski-Phillips-Schmidt-Shin Test): investigates the null hypothesis of stationarity around a deterministic trend, building on the findings of the ADF test. This reveals possible long-term trends or structural fractures within the data by assisting in the determination of whether a series is stationary around a trend.

The ADF and KPSS tests complement one another. The ADF test is especially useful for determining the need for differencing, whereas the KPSS test is more concerned with detecting stationarity around a trend. Analysts can make informed decisions about appropriate transformations and modes thanks to their combined application.

```
ADF Statistic: -4.4329157222535684
p-value: 0.000259451398892672
Critical Values:
1%: -3.433586587734614
5%: -2.8629697591275196
10%: -2.5675311411427995
Result: The time series is stationary (reject the null hypothesis).
```

Figure 5 Augmented Dickey-Fuller test results.

There is substantial evidence for stationarity in **Figure 5**, with an ADF Statistic of -4.43 and a p-value of 0.00026. Critical values of -3.43 (1%), -2.86 (5%), and -2.57 (10%) at various significance levels further corroborate this. This strongly rejects the null hypothesis of non-stationarity because the p-value is extremely low and the ADF Statistic is substantially smaller than these critical values.

As a result, the test's definitive result verifies that the time series is stationary. This suggests that stationarity was probably achieved without the need for differencing, which is important information for further study and modeling. By using this knowledge, models that are more precise and efficient can be created, improving time series data predictions, and understanding.

```
  result = kpss(timeseries, regression='c')
KPSS Statistic: 0.09957647225933565
p-value: 0.1
Lags Used: 27
Critical Values:
10%: 0.347
5%: 0.463
2.5%: 0.574
1%: 0.739
Result: The time series is stationary (fail to reject null hypothesis).
```

Figure 6 KPSS test results

The KPSS (**Figure 6**) test employed here yielded a statistic of approximately 0.10, a p-value of 0.1, and 27 lags. Notably, the critical values for different significance levels indicate a range of acceptable values: 0.347 (10%), 0.463 (5%), 0.574 (2.5%), and 0.739 (1%).

Given that the p-value is quite large (0.1) and the KPSS Statistic is below all of these key values, the null hypothesis of stationarity around a deterministic trend is not rejected. As a result, the test result supports the first hypothesis by confirming the stationarity of the time series. This result suggests that the time series shows signs of stationarity and may possibly point to an underlying trend. This important realization facilitates further research and model development by revealing details about the structural stability of the data and opening up a deeper knowledge of its long-term behavior.

5. ACF and PACF results

Through the examination of the ACF and PACF plots, analysts can obtain significant knowledge for determining the proper AR and MA orders (incorporating seasonal and non-seasonal elements) as well as pertinent exogenous variables that enhance the SARIMAX model's prediction ability.

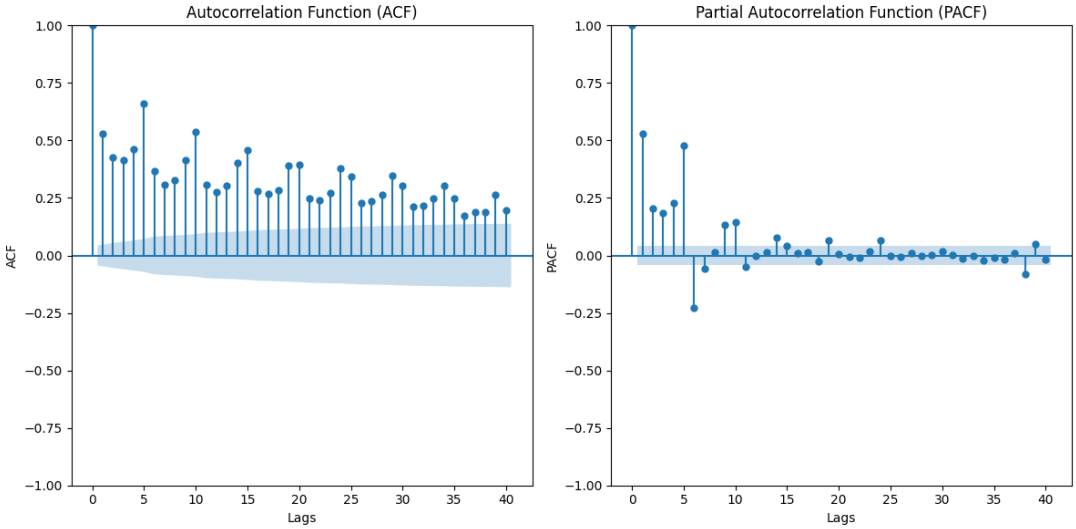

Figure 7 ACF and PACF of original data

Figure 7 reveals crucial information about the underlying patterns and trends within our time series data. Notably, the ACF and PACF plots exhibit a "long memory" effect, meaning the autocorrelation values at successive lags decay slower than expected for a purely random process. This suggests that first-order differencing is necessary to remove the trend-related structures and achieve stationarity. Furthermore, the presence of prominent spikes at every fifth lag in both the ACF and PACF plots signifies a clear seasonal pattern with an order of 5. Identifying this periodicity is crucial for selecting suitable parameters in time series models, particularly SARIMA. This valuable insight sheds light on the data's temporal characteristics, paving the way for future modeling and analysis efforts to be tailored accordingly.

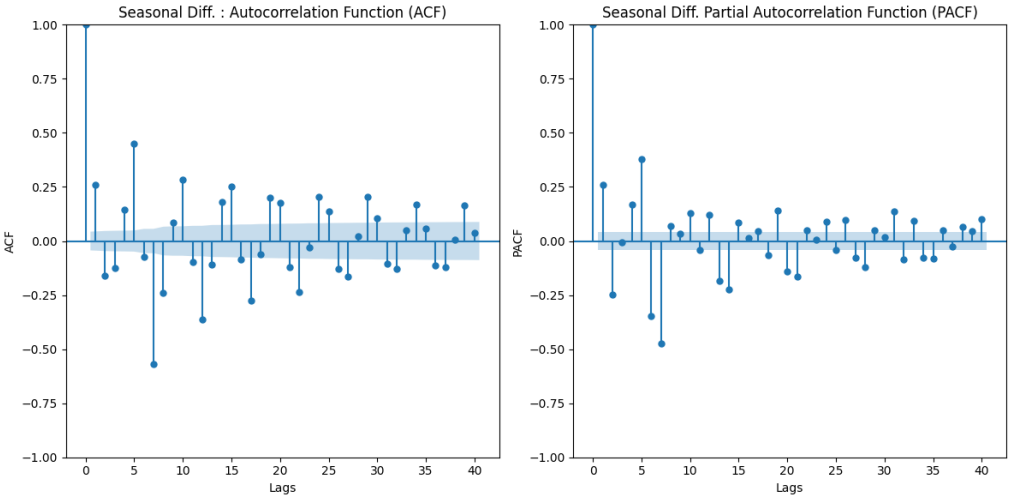

Figure 8 ACF and PACF of 1st differenced data

Following the application of first-order differencing and a seasonal difference of order 5, the ACF and PACF plots in **Figure 8** offer further insights into potential model parameters. The ACF plot specifically reveals autocorrelation potentially present at the first or second lag. This suggests that AR(1, 2, or 4) and MA(1, 2, or 4) models might be appropriate choices for further investigation.

6. Machine Learning

In our machine learning (ML) workflow (**Figure 9**), we follow a systematic approach to ensure robust and accurate analysis.

- Normalize or Standardize: Ensure all data shares the same scale for consistent analysis.
- Leverage Time Series Generator: Utilize a dedicated time series generator tool to efficiently split the data into training (70%) and testing (30%) sets.
- Employ the 30% testing set to train and modify the data for further analysis across different algorithms.
- Time Series Cross-Validation: Implement time series cross-validation with three folds (n_splits = 3) to assess model performance on various data splits.
- Performance Evaluation: Utilize diverse metrics such as MAE, RMSE, MAPE, R^2 to thoroughly evaluate the performance of different models.

- Continuous Improvement: Iterate through feature engineering and hyperparameter tuning to optimize model performance and uncover better options.

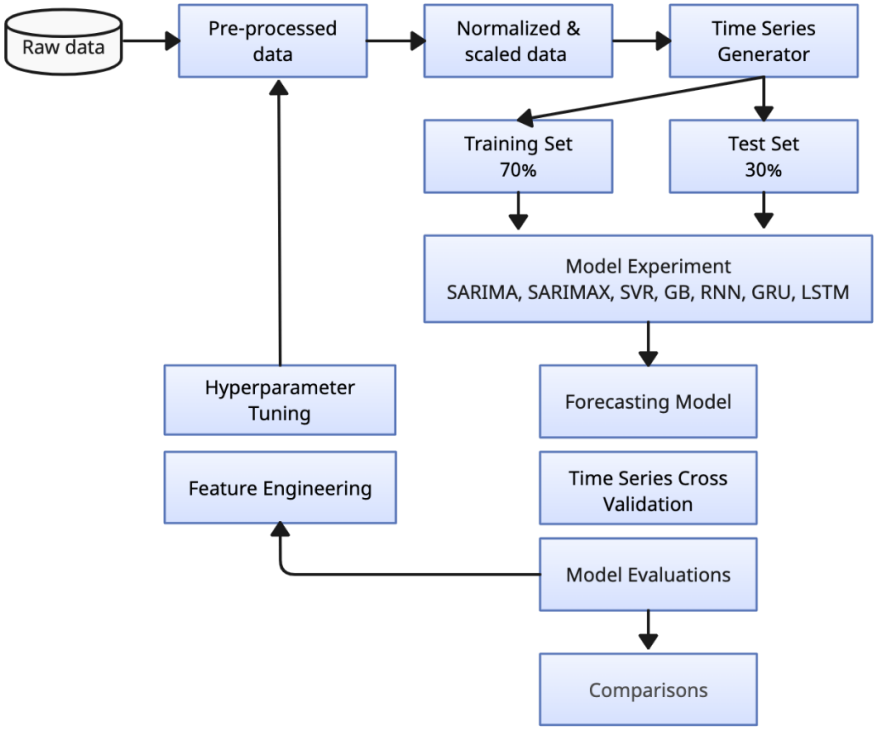

Figure 9 The data flow diagram of the experiment.

7. Error Metrics

In order to assess the accuracy of forecasting models, error metrics are necessary. These metrics quantify the variation between the expected and actual values in a time series. In time series analysis, a model's accuracy is determined by how well it can uncover hidden patterns and trends within the data. Several error metrics are used to assess a forecasting model's performance. Among the most often used error metrics are mean absolute error (MAPE), mean squared error (MSE), root mean square error (RMSE), and mean absolute percentage error (MAPE).

N represents the total amount of value in the time series,

Ai represents the actual value at time i,

Fi represents the forecasted value at time i,

Σ is represents the total of the squared discrepancies between the actual and anticipated values, and | i=1 to N | denotes the summation over all values of i from 1 to N.

7.1 Mean Absolute Error (MAE)

One popular error metric used in time series forecasting is mean absolute error (MAE). the time series' average absolute difference between the values of the forecast (Fi) and actual (Ai) values. Divide the total number of samples (N) by the absolute differences for each time point to get the MAE. The average size of the model's prediction error is measured by the MAE, which is expressed in the same units as the time series data. In general, a model with a lower MAE value is more dependable, whereas a larger MAE number indicates less accuracy.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} | A_i - F_i |$$

7.2 Root Mean Squared Error (RMSE)

In time series forecasting, another commonly used error metric is the Root Mean Squared Error (RMSE). This equation represents a square root of the average of the squared differences between the anticipated (Fi) and actual (Ai) values. To compute RMSE, you square the difference for each time point, sum up these squared differences, take the average, and then calculate the square root of that average. Larger errors are penalized more than smaller errors by the RMSE since the differences are squared. The Root Mean Squared Error (RMSE) can be computed using the formula below:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(A_i - F_i)^2}$$

7.3 Mean Absolute Percentage Error (MAPE)

In time series forecasting, mean absolute percentage error (MAPE) is another popular error metric. The average percentage difference in a time series between anticipated (Fi) and actual (Ai)

values. To compute MAPE, you calculate the absolute percentage difference for each time point, sum up these absolute percentage differences, and then divide by the total number of samples (N). The final value is expressed as a percentage by multiplying it by 100. A useful metric for comparing forecasting model performance across different time series, particularly when the actual values' magnitudes differ significantly, is the mean absolute percentage error (MAPE) of the model. This formula can be used to determine MAPE:

$$MAPE = \frac{100}{N}\sum_{i=1}^{N} \left|\frac{A_i - F_i}{A_i}\right|$$

Note that the MAPE is less appropriate for time series with large seasonal changes or outliers since it tends to overstate large errors and can produce endless or undefined results when the actual values are zero.

7.4 R-squared

The coefficient of determination, also known as R-squared (R2), expresses how closely the forecasts made by the model match the actual values. The following formula is used to calculate it:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(A_i - F_i)^2}{\sum_{i=1}^{N}(A_i - \overline{A})^2}$$

Where $\bar{A}$ denotes the average of the actual values. R^2 is a number between 0 and 1, with 0 indicating that the model explains no data variability and 1 indicating that it fits perfectly. It indicates how well the independent variable predicts the dependent variable's variance, with greater values suggesting greater model efficiency.

Experimental Results

The metrics of all models are summarized in **Table 1**. Among the correlated features, the SVR algorithm performed the best in terms of MAE with a value of 26.2, MAPE with a value of 6.34%, RMSE with a value of 36.6, and a high R^2 with a value of 89.40%. Other algorithms, such as LightGBM, GBR, and

XGBoost, also performed well. However, as compared to traditional machine learning techniques, recurrent neural network models such as Simple RNN, GRU, and LSTM produced larger mistakes and lower R^2 values.

Analyzing the full dataset, SVR emerged as the best-performing algorithm across all measures, with the lowest MAE (39.87), MAPE (10.22%), and RMSE (52.09), as well as a high R^2 of 78.85%. LightGBM, GBR, and XGBoost also performed well, in line with the linked feature group. Notably, recurrent neural network models struggled to capture patterns over the entire dataset, with significantly larger errors and lower R^2 values than other approaches.

In the selected feature group, SVR continued to outperform other algorithms (**Figure 10**), achieving the lowest MAE (25.13), MAPE (6.15%), and RMSE (34.66), along with the highest R^2 (90.56%). LightGBM, GBR, and XGBoost also demonstrated competitive results, reaffirming their effectiveness with selected features. Notably, the performance of recurrent neural network models improved in this feature group, with reduced errors compared to the full dataset, although SVR remained superior in overall predictive accuracy.

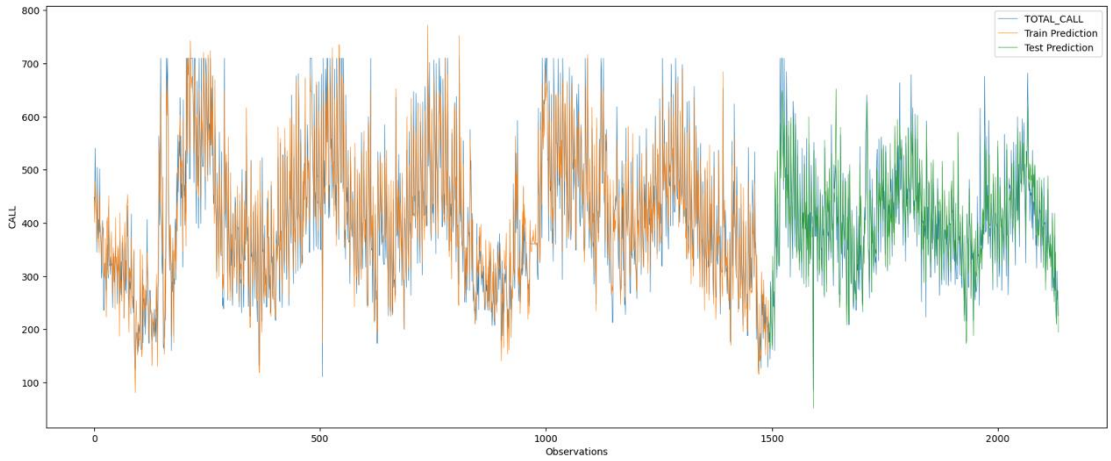

Figure 10 Train and Test Actual vs Prediction plot of Selected Features in SVR

Across all feature groups, SVR consistently stood out as the top-performing algorithm, demonstrating robust predictive capabilities. LightGBM, GBR, and XGBoost also proved effective in capturing patterns in the data, especially when specific features were selected. On the other hand,

recurrent neural network models, including Simple RNN, GRU, and LSTM, exhibited higher errors and lower R^2 values, indicating challenges in capturing the underlying patterns in the given data. These findings highlight the importance of algorithm selection and feature engineering in optimizing time series forecasting models.

Table 1 Summary metrics by model in 3 feature group

Group Feature	ML Algorithms	MAE	MAPE	RMSE	R^2
Correlated Feature	SARIMAX	32.11	8.31%	41.91	84.57%
	SVR	**26.2**	**6.34%**	**36.6**	**89.40%**
	LightGBM	30.73	7.73%	42.52	84.92%
	GBR	29.19	7.32%	39.89	86.62%
	XGBoost	30.53	7.79%	41.72	85.26%
	Simple RNN	58.92	15.75%	78.72	50.70%
	GRU	64.34	16.84%	84.89	43.44%
	LSTM	61.41	18.18%	90.58	57.76%
Full dataset	SARIMAX	57.73	15.69%	68.8	54.81%
	SVR	39.87	10.22%	52.09	78.85%
	LightGBM	31.07	7.91%	42.66	84.77%
	GBR	**29.35**	**7.45%**	**39.93**	**86.65%**
	XGBoost	31.16	7.98%	42.86	84.49%
	Simple RNN	83.22	21.40%	104.98	13.02%
	GRU	60.49	15.66%	81.53	48.17%
	LSTM	62.99	17.57%	87.69	60.35%
Selected Feature	SARIMAX	33.21	8.73%	43.03	84.01%
	SVR	**25.13**	**6.15%**	**34.66**	**90.56%**
	LightGBM	30.95	7.81%	42.25	84.95%
	GBR	28.77	7.18%	39.24	87.04%
	XGBoost	30.51	7.73%	41.72	85.12%
	Simple RNN	59.3	15.79%	80.00	49.25%
	GRU	58.88	15.83%	79.00	50.52%
	LSTM	54.53	15.20%	78.48	68.31%

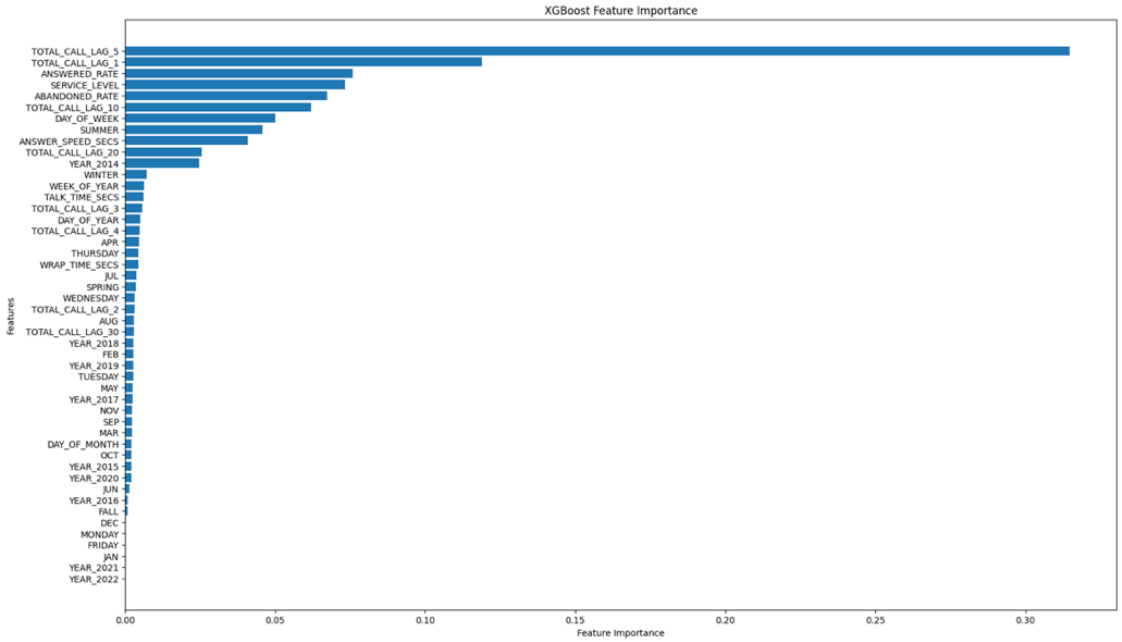

Figure 11 XGBoost Feature Importance (Full dataset)

Conclusion

The present study highlights the critical function of Workforce Management (WFM) in enhancing the efficacy of contact centers. It highlights the significance of optimizing call volume forecasting and agent scheduling procedures to achieve cost savings and operational efficiency. The experiment investigated the efficacy of sophisticated time series models, such as SARIMAX, SVR, Gradient Boosting, RNN, GRU, and LSTM, in forecasting contact volume, a crucial component of a successful WFM implementation. With the lowest Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and greatest R^2 values, SVR and GBR stood out as the best-performing models. The study used careful metrics translation, normalization, and data pretreatment to improve model performance by feature engineering and hyperparameter optimization.

Discussion

To systematically categorize data and investigate their impact on predictive model efficacy, the study adopts three unique feature groups—Full Dataset, Correlated Feature group, and Selected Feature group. Notably, the SARIMAX model demonstrates competitive performance with a concentrated selection of only five features, demonstrating the efficacy of focused feature selection over a more extensive method. SVR emerges as the best-performing model, excelling particularly with features derived from XGBoost's top important features, highlighting the importance of feature engineering in optimizing SVR's performance. Gradient Boosting models provide consistent top contributors, with differences in feature relevance among models such as LightGBM highlighting the intricacies of model-specific preferences. While RNNs perform poorly across the full dataset, GRU and LSTM perform similarly across feature sets, highlighting the necessity of careful feature selection to avoid overfitting and improve generalization. The fifth lag element, which captures patterns across a 5-day period, is critical for forecasting, especially on working days. Key measures such as abandoned rate, answer speed, and service level call are more than just process features (**Figure 11**); they are invaluable indicators of caller behavior, system performance, and operational efficiency. Recognizing the significance of these traits allows organizations to better prepare for swings in call volume, improving overall service quality and customer happiness.

Limitation

This study provides valuable insights into call volume forecasting, suggesting potential areas for further refinement and exploration. The inclusion of hourly data and additional features, such as top contact reasons, locations, and contact channels, could enhance the granularity of temporal patterns and improve the understanding of factors influencing the time series. Leveraging domain expertise within the company is recommended for better interpretability and contextual relevance in feature engineering and model selection. The study's scope could be broadened by examining a wider array of models and datasets to gain a more comprehensive understanding of the model's applicability across different contexts. Experimentation with various models may reveal alternative strategies that could outperform or complement existing models in specific situations.

Reference

Box, G. E. P., & Jenkins, G. M. (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day. https://books.google.co.th/books?id=1WVHAAAAMAAJ

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. *Neural computation*, *9*, 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Services, C. o. C.-D. o. P. (2023, December 6). *Citizen Service Request (CSR) Call Center Calls*. City of Cincinnati - Department of Public Services. https://data.cincinnati-oh.gov/Efficient-Service-Delivery/Citizen-Service-Request-CSR-Call-Center-Calls/k2qr-ck2v

Werbos, P. J. (1988). Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, *1*(4), 339-356. https://doi.org/https://doi.org/10.1016/0893-6080(88)90007-X