Intraoperative arterial pressure time series forecasting algorithm in preoperative hypertension patients

Authors: Chuheng(Kevin) Yu, Xin(Ashely) Su, Weiming(Kevin) Wang, Tyler Christoforo*

Key words: Arterial pressure; Deep learning; Time series forecasting; Biosignal waves

Abstract

We present an in-depth analysis of forecasting intraoperative arterial pressure time series in patients with pre-existing hypertension. It draws inspiration from previous studies. By integrating deep learning methodologies, notably neural networks, and comparing their performance with traditional time series forecasting models like ARIMA and Facebook's Prophet. The research is grounded in the application of the VitalDB dataset, a comprehensive collection of intraoperative biosignals. The data was then filtered and preprocessed to fit each model's requirement. For performance, the MAE of ARIMA and SARIMA using 98 random events is 34.27 and 18.20. The final average MAE of Prophet is 24.129. And the test results MAE of LSTM neural network model is 11.39. The findings underscore the neural network model's enhanced performance in terms of consistency and complexity. This demonstrates its potential for more accurate predictions and integration into medical devices, offering a substantial contribution to the field of medical informatics and patient care. The study's methodological rigor and practical implications underscore its value in advancing predictive analytics in healthcare settings.

Introduction

Intraoperative hypotension(IOH) in simple terms, describes itself as the patient reaching low blood pressure during a non-cardiac surgery. In an attempt to arrive at a general definition, Laurence *et al* compiled studies and suggest that intraoperative hypotension should be defined by using the absolute values (POQI), with MAP < 60-70 mmHg or SBP < 100 mmHg [1]. Intraoperative hypotension is commonly associated with major kidney, neurological and cardiac events and has shown linkage to a higher risk of postoperative mortality, myocardial injury and much more[2]. Moreover, the risk factors that cause IOH may vary greatly, from old age to preoperative body conditions. Thus being able to predict IOH is an important task, which has been addressed by Lee *et al*[3]. Lee's team had utilized deep learning algorithms for real time predictions minutes before a hypotensive event The team separated surgery types by invasive and non-invasive, and used certain biosignal waves as inputs. The algorithms were able to reach MAEs of 7-8 mmHg, successfully predicting hypotension.

In light of this piece of work, our study fully recognizes the resourcefulness of Lee *et al*'s work, and shift the research focus into understanding the differences in performance for time series forecasting under different sub groups. Additionally, our study evaluates and compares how neural network models perform to how pre-packaged time series forecasting models, like Facebook's Prophet and ARIMA, perform.

Method

Data Source

The data source we chose for this study comes from VitalDB[4]. VitalDB is a dataset that captures intraoperative biosignals and clinical information. It is a high quality dataset of 6388 surgical patients, with more than 60 columns of related patient information to help interpret the signals. In total there are 557622 data tracks for these cases, each of which can be extracte from different files. VitalDB is available for access through PhysioNet, the downloadable files used are the clinical_data.csv and vital_files. These two files can be connected through the use of case_id and a function from the API. For clarity purposes, each row of data is 1 second.

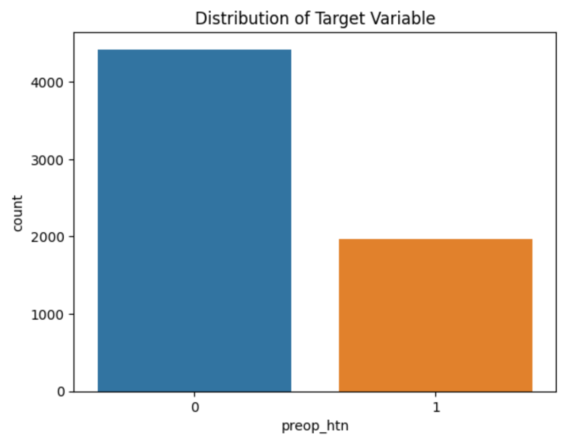
Data Pre-processing

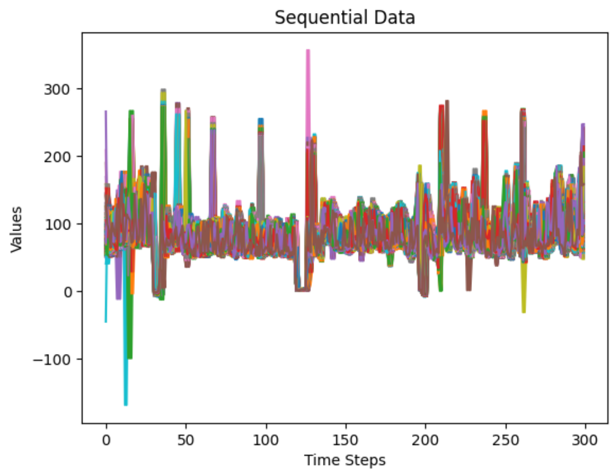
ARIMA & SARIMA

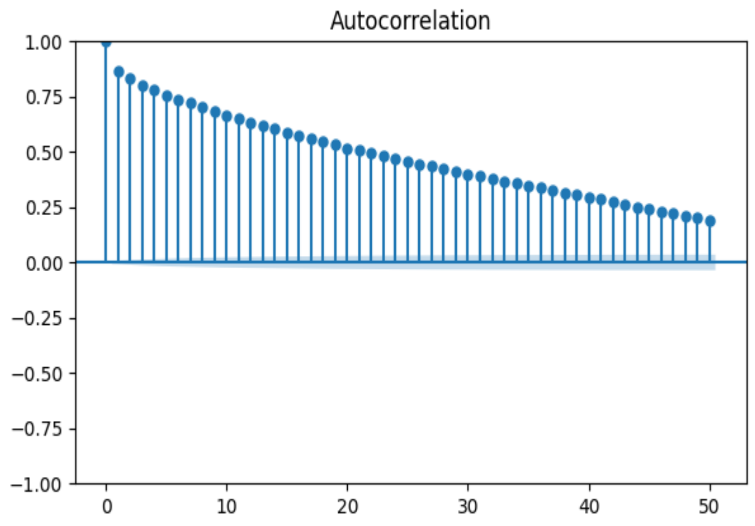
In order to comply with the ARIMA and SARIMA's requirements, a series of data pre-processing had been done. To start, our study narrows down the scope of the patients to the ones who had been recorded as having "preoperation hypertension", denoted as "preop_htn" in the dataset, the separation is shown in graph A**.** For these records, the case_ids were identified and used to filter records in the vital files, only records with arterial pressure wave (ANUADC/ART), ECG lead II wave (SNUADC/ECG_II), plethysmography wave (SNUADC/PLETH), capnography wave (Primus/CO2), and body temperature (Solar8000/BT) all simultaneously available will be selected. For a showcase of these biosignal, graph B demonstrates the waveforms in respect to time steps.

This gives us 11112 records of data. Since the models should be event id based, a group by procedure was carried out. Additionally, to standardize the comparisons among the models, it was essential to have all the record counts for each event id the same. This number was determined to be 75, in which 60 of the records are used for training and 15 used for testing.

Lastly, the goal is to use these records to predict the arterial pressure for each event id. In our case, it is to use 60s (rows) to predict 300s of future arterial pressure.







Graph A. Bar graph of the variable "preop-htn" distribution; Graph B. Sequential data of the biosignal waves in the form of line graph for Neural Network training; Graph C. Auto correlation graph on the arterial pressure column

Prophet

The input to Prophet is always a dataframe with two columns: ds and y. The ds (datestamp) column should be of a format expected by Pandas, ideally YYYY-MM-DD for a date or YYYY-MM-DD HH:MM:SS for a timestamp. The y column must be numeric, and represents the measurement we wish to forecast.

As our data does not stringently follow the required formats, we first preprocessed the data. Specifically, we are taking only the "art" column as our y column, and manipulating each record, namely each second of every event_id starting from 2023-01-01 00:00:00 as a timestamp column. Due to the absence of actual operation date and time information, we opted to assume a common start time for all operations.

Neural Network

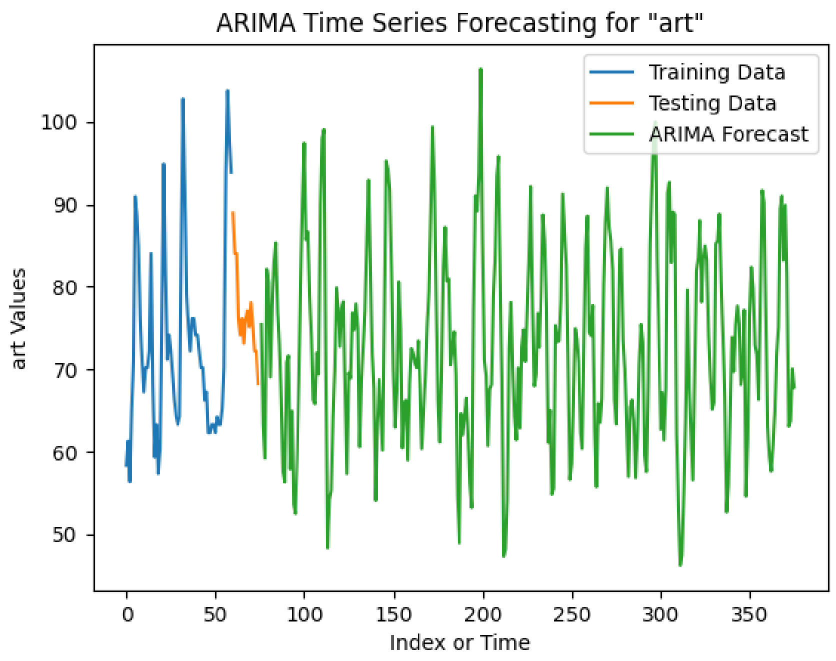
We opted for a neural network approach to forecast intraoperative hypotension in a time series context. Diverging from previously implemented pre-trained models, our approach successfully examined the impact of various hyperparameters and layers on performance. Utilizing the VitalDB dataset, our focus was on patients with preoperative hypertension, employing the same dataset and methodology. We selected arterial pressure wave (ANUADC/ART), ECG lead II wave (SNUADC/ECG_II), plethysmography wave (SNUADC/PLETH), capnography wave (Primus/CO2), and body temperature (Solar8000/BT) as model inputs. Our dataset comprised 1112 valid surgery cases, each with preoperative hypertension and all five waveforms recorded.

Model Construction

ARIMA

ARIMA stands for autoregression integrated moving average, this model is used to fit time series data. The name suggests 3 components: Auto Regression, Integrated and Moving Average. Auto regression refers to the function of modeling the relationship between an observation and several lagged observations. Integrated refers to the differencing of the time series in order to make it stationary. Moving Average is a modelling function that models the relationship between an observation and a residual error from a moving average model applied to lagged observations.

ARIMA has three parameters that are adjustable. Denoted as (p,d,q), where p represents the order of the Auto Regression, d represents the degrees of differencing, and q represents the order of the Moving Average. More specifically, p can be understood as including p number of most recent time points as predictors; And for q, it would mean the number of residual errors to include. For our study, p=60, d=1, q=1, autocorrection graph is shown in graph C. The model follows a standard 80-20 test train split for ARIMA fitting. The forecast function was set to 300 forecast steps. Graph D illustrates an example of ARIMA fitting, and the forecasting results.

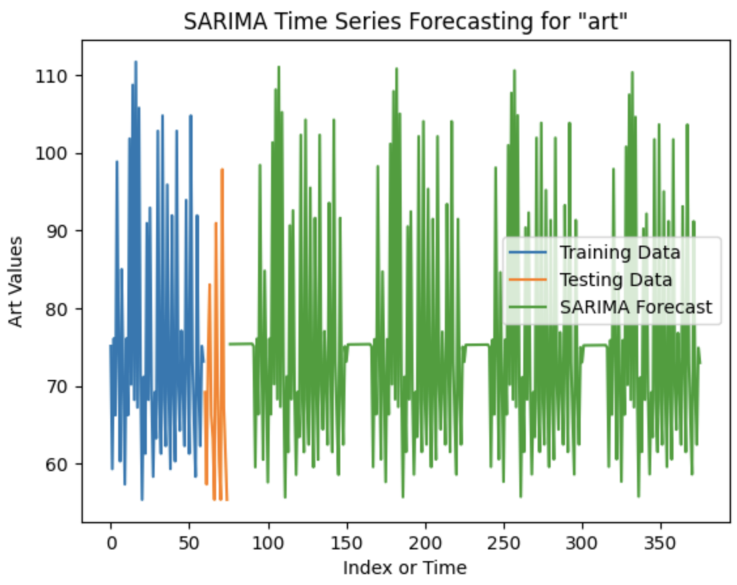


Graph D. Example of ARIMA model fitting and forecasting line graph for an event id

SARIMA

SARIMA stands out as Seasonal Autoregressive Integrated Moving Average. As an extension of the ARIMA model, SARIMA offers greater flexibility, allowing it to model a wider range of data with various levels of complexity. This includes trends, cycles and seasonal effects, enabling more comprehensive exploration of data.

Based on the ARIMA model, the SARIMA model includes four additional parameters, namely P, Q, D, and s. The P,Q,D parameters are similar to the pdq parameters of ARIMA but are specifically used for seasonal components of the time series. Before we run the model, our team employs Augmented Dickey-Fuller test to check the stationary of our time series. The test results, presented in Table A and B, indicate that the p-value is nearly 0. This suggests that the non-seasonal and seasonal components of the time series are stationary. Consequently, we set both d and D equal to 0. Moreover, parameter s indicates the length of a seasonal cycle. In our case, we set it as 75 as we extract each of the event id for 75 seconds, and consider this duration as one cycle. Therefore, we configure our SARIMA model as (1,0,1)(1,0,1) 75. Similar to the ARIMA model, we perform a test train split of 80 -20 for each one of the events to fit the model. Lastly, we set it to forecast 300 seconds arterial pressure. Graph E depicts an example of SARIMA fitting, and the forecasting results.



Graph E. Example of SARIMA fitting and forecasting for an event id

```
Test Statistic                    -29.136512
p-value                             0.000000
#Lags Used                         65.000000
Number of Observations Used     83265.000000
Critical Value (1%)                -3.430429
Critical Value (5%)                -2.861575
Critical Value (10%)               -2.566788
dtype: float64
```

Table A. Augmented Dickey-Fuller Test Result for Non-seasonal Time Series

```
Test Statistic                    -38.033317
p-value                             0.000000
#Lags Used                         65.000000
Number of Observations Used     83205.000000
Critical Value (1%)                -3.430429
Critical Value (5%)                -2.861575
Critical Value (10%)               -2.566788
dtype: float64
```

Table B. Augmented Dicky-Fuller Test Result for Seasonal Time Series

Prophet

Prophet is a forecasting model developed by the research team at Facebook for time series analysis and prediction. It is used in many applications across Facebook for producing reliable forecasts for planning and goal setting. Based on the nature of an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects, it works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

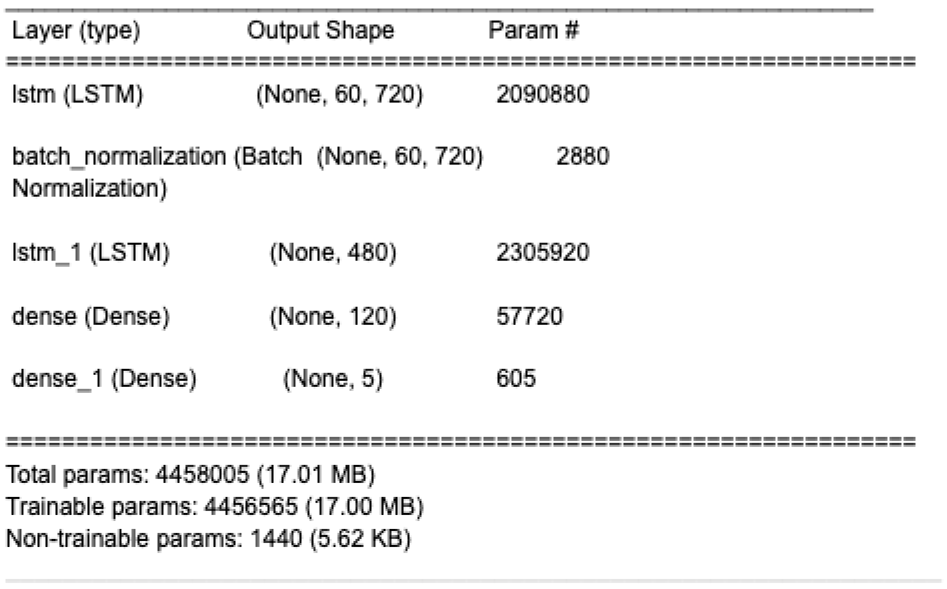
Given that Prophet is a univariate model, our approach involves training individual Prophet models for each event. This means that we predict based on the characteristics of each specific event. For each model, the dataset is split, with the initial 80% of records used as training data, and the remaining 20% kept as unfitted test data to assess model performance using mean absolute errors.

Neural Network

To facilitate dataset splitting, we enumerated the data into a pandas dataframe and assigned unique event_ids. Acknowledging that imputing missing values could introduce inaccuracies and biases, we chose to drop all null values to uphold the dataset's integrity and authenticity. We partitioned the dataset into 60% for training, 20% for validation, and 20% for testing, ensuring randomization in the allocation of cases across all three datasets. Subsequently, we filtered the dataframe based on case IDs and converted the datasets using the `timeseries_dataset_from_array` function from Keras for time series analysis. We set arterial pressure as the target variable and all other measurements as independent variables, with an offset determined by a prediction lag, referred to as ‘delay’. The sampling rate was fixed at 30Hz, with a prediction lag of 300 seconds (5 minutes). The 'delay' represents the time interval between the input data and the target variable, calculated by multiplying the sampling rate with the prediction lag. We also set the batch size to 256.

Moreover, we crafted a ‘create_sequences’ function to generate sequences and corresponding labels from time series data for all three datasets. This function efficiently transformed the data into a format conducive for training forecasting models. The forecast horizon, set to 5, directly influenced the model’s ability to learn temporal patterns and make accurate predictions.

In this approach, we employed a sequential LSTM neural network model, presented in Table C. The first LSTM layer, comprising 720 neurons, was configured to return the full sequence to the next layer, using 'tanh' as the activation function. This was followed by a batch normalization layer to enhance training stability, succeeded by another LSTM layer with 480 neurons, also employing 'tanh' activation. The model culminated in two Dense layers, with the final one outputting units equal to the ‘forecast_horizon’, aligning with the forecast horizon set earlier.



```
Layer (type)              Output Shape           Param #
=================================================================
lstm (LSTM)                (None, 60, 720)        2090880

batch_normalization (Batch  (None, 60, 720)         2880
Normalization)

lstm_1 (LSTM)              (None, 480)            2305920

dense (Dense)              (None, 120)            57720

dense_1 (Dense)             (None, 5)             605

=================================================================
Total params: 4458005 (17.01 MB)
Trainable params: 4456565 (17.00 MB)
Non-trainable params: 1440 (5.62 KB)
```

Table C. Neural network model with LSTM and Dense layers configuration

Results

ARIMA & SARIMA

Due to computing power, it was hard to generate an average of all the events, but it was possible to randomly choose event ids to test the performance of the model. The performance metric used is MAE. For the sake of comparison, both models randomly chose 10 event ids for the model to forecast, and the final mae is the average of all 100 event ids' MAE. The MAE of using ARIMA is 34.27, the MAE of using SARIMA is 18.20.

Prophet

The performance is evaluated by calculating the mean errors across all 1,112 models to derive the ultimate error measure for this Prophet model.The final mean absolute error of our Prophet model is 24.129.

Neural Network

The training and testing results of the neural network model offered insightful revelations about its performance in forecasting tasks. Over five epochs, the model exhibited a significant evolution in performance, as indicated by variations in loss (Mean Squared Error) and mean absolute error (MAE). The initial loss was recorded at 415.0760 with an MAE of 13.1318. However, as training advanced, these metrics demonstrated fluctuation, highlighting the model's progressive adaptation to the dataset. Notably, the final epoch registered a loss of 483.2746 and an MAE of 15.7490, suggesting potential overfitting or other complexities in the learning process.

In the testing phase, the model was assessed on an unseen dataset. The results were promising, showing a test loss of 350.3533 and a test MAE of 11.39. These outcomes indicate that despite the challenges observed during training, the model attained a commendable accuracy level in forecasting based on past observations. Nonetheless, the discrepancy between training and testing metrics underscores the need for further model refinement and data exploration to enhance predictive accuracy and generalizability.

Discussion

A comparison across all models' MAE might not seem far apart, but it is evident that the neural network is more consistent with the lower MAE score.Not only that, the neural network takes five inputs and can potentially take in more factors. But on the other hand, the simple models only consider the target variable itself, lacking much comprehensiveness and complexity.

The methodology of our study poses a couple of limitations. These models typically don't consider other factors that could affect arterial pressure, they only consider the art column itself. Tying into a greater limitation, where ARIMA fails to capture complex patterns, struggling to recognize the interactions in the data. Additionally, it can be seen that the larger the forecast steps, the harder it is for the model to be accurate. The simplicity of the parameters makes each choice of (p,d,q) quite different in performance, one proving the sensitivity of them, and two proving the inconsiderate part of the parameters. Further on the sensitivity, ARIMA is also sensitive to outliers, as seen in graph C, the 5 sharp peaks in the training data fluctuated forecasting results, where many sharp peaks are seen.

For SARIMA, the grid search is the most effective approach for our team to find the most ideal combination of the parameters. However, given the large size of our data, it is challenging to optimize all the parameters, which may result in less accurate prediction. Moreover, SARIMA is a one-dimensional forecasting tool, meaning it only uses one variable of historical data to predict its future values. Hence, SARIMA might not account for multivariate relationships. Furthermore, SARIMA can be suspected to force seasonality upon its forecast, as demonstrated by graph E.

The limitations of employing the Prophet model in our case are evident on several fronts. Firstly, it is typically employed for macro-level data that spans more extended periods and exhibits clear seasonal patterns. In our scenario, the focus is primarily on the duration of individual operations, rendering it too micro in nature for the model's conventional application. Secondly, the Prophet model conventionally utilizes only two columns for prediction: a timestamp column and the target variable (y). Consequently, the wealth of additional features derived from VitalDB remains untapped. Thirdly, in comparison to neural network models, the flexibility of the splitting process is notably restricted. We are constrained to setting a specific train-test split percentage, lacking the adaptability offered by techniques commonly applied to neural networks, such as setting sampling rate and sequence length inside the model.

As for our LSTM model, our studies suggest increasing model complexity to achieve a better performance. Simply by adding more layers combined with dropout layers, examining the learning rate, and increasing the number of neurons within existing layers, the model might better learn and generalize the intricate patterns. Moreover, expanding the prediction lag from 300 seconds to 600 seconds and 900 seconds could be beneficial, especially for surgical procedures where longer-term forecasts might be more clinically relevant. Lastly, extending the input length could let the model incorporate a more comprehensive history of the patient's arterial pressure behavior.

Conclusion

In our study, we extended our research off of Lee's work[3], utilizing VitalDB to predict time series arterial pressure among patients that were identified as preoperative hypertension. We examined three stats-based models, ARIMA, SARIMA and Prophet and tested the effectiveness of each. These models are inconsistent with each event id, with Prophet having the smallest average MAE from a batch of events. Thus prompting us to explore time series forecasting with LSTM neural networks. This achieved a MAE of 11.39, staying consistent,more comprehensive, more complex and more inclusive than the simple models. Although further tuning and layers in the neural network can be optimized even more, the base work in our study proves deep learning to be effective in intraoperative arterial pressure time series forecasting, shows potential in accurate predictions of arterial pressure and can eventually be implemented into medical devices and hospital monitor systems.

References

[1]Guarracino F, Bertini P. Perioperative hypotension: causes and remedies. J Anesth Analg Crit Care. 2022 Apr 14;2(1):17. doi: 10.1186/s44158-022-00045-8. PMID: 37386537; PMCID: PMC10245539.
[2]Weinberg, L., Li, S.Y., Louis, M. et al. Reported definitions of intraoperative hypotension in adults undergoing non-cardiac surgery under general anaesthesia: a review. BMC Anesthesiol 22, 69 (2022). https://doi.org/10.1186/s12871-022-01605-9
[3]Solam Lee, Hyung-Chul Lee, Yu Seong Chu, Deep learning models for the prediction of intraoperative hypotension, British Journal of Anaesthesia, Volume 126, Issue 4, 2021, Pages 808-817,ISSN 0007-0912
[4]Lee, H.-C., Park, Y., Yoon, S.B., Yang, S.M., Park, D., Jung, C.-W., 2022. VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients. Scientific Data 9.. https://doi.org/10.1038/s41597-022-01411-5
[5]Grannan, Chris. "Sarima Modeling." *Medium*, Medium, 19 Mar. 2021, chrisgrannan.medium.com/sarima-modeling-42ff700af29.