Investigating The Usage of Forecasting Models and Their Effectiveness in Our Daily Lives

What is the relative forecasting accuracy of SARIMA and PROPHET models for daily COVID-19 cases in South Korea?

A Computer Science Extended Essay

Word count: 3261

Table of Contents

I. Introduction

The principle of forecasting comes from the idea that data can be predicted prior to the real event occurring by identifying underlying patterns in a dataset. Forecasting has become possible by analyzing statistics evident within datasets that have one or more variables.

As research into forecasting has progressed over time, models such as neural networks and piecewise linear models have been published to the world wide web. These models have allowed for individuals to test and automate forecasting datasets.

For this specific research document, the SARIMA and PROPHET model will be investigated and compared in both performance and accuracy. Both models have been a popular choice for both individuals and businesses to forecast stocks and climate change. The SARIMA model works as a conjunction between multiple linear models that works as weight for each forecast completed. The PROPHET model also works by connecting simpler models together to process an accurate forecast.

The dataset in interest for this investigation is going to be the total number of new COVID-19 cases that has occurred from January 3rd, 2020, till May 3rd, 2023, in South Korea. Prior to finalizing the decision to using the COVID-19 dataset, there was a dilemma on whether climate change should be used instead due to its longevity compared to COVID-19. With the goal of acknowledging the growing pandemic and its direct impact on humanity, climate change was disregarded as it had less urgency and attention compared to COVID-19.

To summarize, the research will be answering, “What is the relative forecasting accuracy of SARIMA and PROPHET models for daily COVID-19 cases in South Korea?” Throughout this research, an analysis and understanding of what both models will be completed. While doing so, a method of utilizing these two models would be needed to understand how to forecast and test the relative accuracies that it has produced.

II. Literature Review

2.1.1 Non-seasonal ARIMA Model

The ARIMA model is made up of three components: auto-regressive (AR), integrated (I), moving-average (MA). These components are individual models that are assigned the variable p, d, and q. The parameters can be changed accordingly to fit the model and tested for its accuracy using the Akaike's Information Criterion (AIC), a branch of Akaike's Information Criterion (AICc), and Bayesian Information Criterion (BIC) (Smith 6).

AR is assigned to p, I is assigned to d, and MA is assigned to q. The p value indicates the number of lag observations. The lag observations shows us how many past values of the variable are used to predict the current value of the variable (Hyndman and Athanasopoulos 8.2, 8.3).

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

Equation 1: Mathematical expression for autoregressive models (Hyndman and Athanasopoulos 8.3)

The d value identifies the degree of differencing. Depending on the degree of differencing, it will tell us the number of times the current value is subtracted from the previous value (Hyndman and Athanasopoulos 8.1).

$$y'_t = y_t - y_{t-1}$$

$$y''_t = y'_t - y'_{t-1} = (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) = y_t - 2y_{t-1} + y_{t-2}$$

Equation 2: Mathematical expression for differencing (Hyndman and Athanasopoulos 8.1)

The q value identifies the order of the moving average which is used to add weights to compensate for the errors made which would alter the forecast accordingly (Hyndman and Athanasopoulos 8.4).

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

Equation 3: Mathematical expression for the moving average model (Hyndman and Athanasopoulos 8.4)

Hence, the formula for the non-seasonal ARIMA model can be derived by combining these three components into one equation.

$$y'_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

Equation 4: Mathematical expression for the ARIMA model (Hyndman and Athanasopoulos 8.5)

As previously mentioned, AIC acts as an estimator for predicting the prediction error. Therefore, this can be used to test the accuracy and validity of the model for model selection (BURNHAM and ANDERSON). However, the AIC is under the risk of overfitting and underfitting which would lead to false data results. Therefore, to minimize the issue of overfitting and underfitting in small sample spaces, AICc can be utilized. However, due to the increase in complexity of the AIC formula, it is more difficult to compute (BURNHAM and ANDERSON). The BIC's purpose is like an AIC, it works as an estimator and the lower the value, the more accurate it is (BURNHAM and ANDERSON).

2.1.2 Seasonal ARIMA Model

The SARIMA (Seasonal ARIMA) is an extension of the ARIMA model. The SARIMA model would detect seasonality and trends that can be found throughout the dataset. Additional parameters of the SARIMA are P, D, Q, and m (Hyndman and Athanasopoulos 8.9).

The P value shows the order of the seasonal autoregressive component. This means how many previous values within a season are used to predict the current value.

The D value is the degree of seasonal differencing, which shows how many times the data is differenced at a seasonal lag to make it stationary.

$$y'_t = y_t - y_{t-m}$$

$$y''_t = y'_t - y'_{t-m} = (y_t - y_{t-m}) - (y_{t-m} - y_{t-2m}) = y_t - 2y_{t-m} + y_{t-2m}$$

Equation 5: Mathematical expression for seasonal differencing (Hyndman and Athanasopoulos 8.1)

The Q value is the order of seasonal moving average, which shows how many previous errors are used as weights in a season to predict the current value (Hyndman and Athanasopoulos 8.9).

To determine what seasonal parameters are the most optimal for the SARIMA, it is ideal to see the seasonal lags of both partial autocorrelation function (PACF) and autocorrelation function (ACF). ACF measures how much the time series correlates with itself at different lags. For example, if the ACF at lag 12 has a high frequency, then the value of the time series at a given time point is like the value 12 lags ago. PACF measures how much the time series correlates with itself at different lags after removing the effect of previous lags. For example, when the ACF reads at lag 12 as high, the value of the time series at a given time point is similar to the value 12 lags ago after accounting for the values in between. When the ACF shows a gradual decay and the PACF shows a sharp cut off after a certain, it suggests that an AR component at that lag. If the ACF shows a sharp cut off and the PACF shows a gradual decay, it suggests an MA component at that lag. If either plot shows a significant spike at the seasonal lag, it suggests a seasonal component at that lag (Hyndman and Athanasopoulos 8.9).

2.2 PROPHET Model

The PROPHET model is made up of three components: trend ($g(t)$), seasonality ($s(t)$), holidays ($h(t)$), and the error term. These components are then combined to generate a formula for forecasting.

The trend component works to model the nonperiodic changes in the value of the time series (Taylor and Letham 39). The component can be either linear or logistic depending on the growth parameter. A linear trend is a slope from one point to another. When the linear slope has a change in direction it is often referred to as the changepoint. A logistic trend is a curved line that eventually approaches a limit, in other words it is called the carrying capacity, a maximum point which the forecast can reach. The mathematical formula for this component can be simplified into:

$$g(t) = \begin{cases} kt + m & if\ growth = 'linear' \\ \dfrac{C}{1 + e^{-k(t-m)}} & if\ growth = 'logistic' \end{cases}$$

Equation 6: Mathematical expression for trend component (Taylor and Letham 40)

C is the carrying capacity, k is the initial growth rate, and m is an offset parameter (Taylor and Letham 40).

The seasonality component models the periodic changes which can be weekly and yearly (Taylor and Letham 41). The component consists of the Fourier series to provide a dynamic model of periodic effects, for annual data $P = 365.25$, for weekly data $P = 7$. The mathematical formula for this component can be simplified into:

$$s(t) = \sum_{n=1}^{N} (a_n cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right))$$

Equation 7: Mathematical expression for seasonality component (Taylor and Letham 41)

P is the period expected for the time series to have, N is the order of the Fourier series, and a_n and b_n are weights to be estimated (Taylor and Letham 41).

The holidays component works to eliminate the effects of holidays as they could sometimes generate irregularities (Taylor and Letham 41). The component sets a range of dummy variables for each holiday which the user adjusts with dates. The mathematical formula for this component can be simplified into:

$$h(t) = \sum_{i=1}^{H} \gamma_i 1_{d(t)=d_i}$$

Equation 8: Mathematical expression for holidays component (Taylor and Letham 41)

H is the number of holidays, γ_i is the magnitude of the holiday effect, $d(t)$ is the date of time, and d_i is the date of holiday (Taylor and Letham 41).

The error terms identify points in data changes that are unique and doesn’t follow a trend (Taylor and Letham 44). The component acts as a weight for the forecast to adjust the forecast generated based on the previous value that it has forecasted.

$$\xi(h) = E[\phi(T, h)]$$

Equation 9: Mathematical expression for error term (Taylor and Letham 44)

The h value is used to represent the error made at a horizontal forecast, and the T shows the last point of historical data used to fit the model (Taylor and Letham 44).

2.3 Comparison

When fitting the SARIMA model, there needs to be a total of six parameters filled out and evaluated based on the performance of the forecast. This can be computationally expensive and time consuming to be complete. Creating a list of parameters for the SARIMA to run through and test is a solution to the time consumption of manually modifying the parameters, but it does not help improve the computational expense of testing every parameter. On the other hand, the PROPHET model does not require the modification of parameters but only needs adjustments to its holidays and seasonality. The PROPHET model already automatically determines the best fit of the model based on its training data. Therefore, the PROPHET is more intuitive than SARIMA.

Another comparison between the two models is that while the SARIMA assumes that the seasonality of its data is continuous, the PROPHET can generate forecasts based on multiple seasonalities. Based on this, an assumption can be made that the PROPHET will perform better for long-term forecasts than the SARIMA, but the SARIMA will perform better for short-term forecasts.

2.4 Relevant Studies

The first study has tested for the forecasting of seasonal influenza in Mainland China from 2005 to 2018 by utilizing the SARIMA model. Results for this experiment showed that the model fitted the seasonal fluctuation well with the predicted relative errors from 0.0010 to 0.0137 (Cong et al.). For example, when the relative error for July 2018 is 0.001, the predicted value of 1.65 is very close to the actual value, 1.64.

The second study investigated a minimalistic approach for evapotranspiration (ET) by using the PROPHET model. For comparison, the stochastic volatility (SVT) model was used against the

PROPHET model. Results showed that the PROPHET model generally performed better in high rainfall scenarios while the SVR model was more suitable for low rainfall scenarios (Hosono et al.). This may have been due to the PROPHET model being more robust to outliers in the data which may have been more common in high rainfall scenarios. There may also have been missing values and data gaps which the PROPHET model is able to fill out.

The third study investigated the forecasting of the air pollution in the city of Bhubaneswar located in India by comparing the SARIMA and PROPHET model. The approach to comparing the performance of these two models was by measuring their performance through root mean squared error (RMSE) and mean squared error (MSE). Results revealed that both models have provided a good quality of accuracy. However, the PROPHET model with a logarithmic data transformation did perform the best with the lowest RMSE and MSE value (Rani Samal et al.).

III. Methodology

3.1 Data Collection

The data required for this research was gathered from WHO. They provide data for the number of daily/total cases and vaccination per country in relation to COVID-19. For this experiment, as the investigations involves the accuracy in forecasting the number of daily COVID-19 cases, the dataset with the title of ‘Daily cases and deaths by date reported to WHO’ will be used. The dataset is made up of 8 columns: Date_reported, Country_code, Country, WHO_region, New_cases, Cumulative_cases, New_deaths, Cumulative_deaths.

Based on the trial that was run, the data was split. Different trials had different numbers of training and testing data. The testing data would be required to test the accuracy of the forecasts being made.

3.2 Notes

To build the SARIMA model and the PROPHET model, a fit and forecast method has been used. For the SARIMA, the pmdarima, developed by Taylor G Smith and Aaron Smith with other external contributor, has helped automate the process of building the forecast. For the PROPHET model, it already has a built-in automatic fit and forecast method provided by TensorFlow.

3.3.1 ARIMA Data Processing

To use ARIMA, the data needs to be steady, which means the data should not fluctuate too much over time. But many real data are not steady, because they have patterns or cycles. It is possible to make these kinds of data steady through different methods of transformation.

This program uses two ways of changing the data: Box-Cox and log. Box-Cox makes non-normal distributions into normal distributions. Log is a type of Box-Cox that makes the data less tilted and less wide by using Euler's number, e.

It is important for many math problems and models that the data are like a bell shape. It means the data have one peak and two sides that are the same. Normaltest from pmdarima checked if the data were like a bell shape after changing them. It measures how much the data are like a bell shape and gives a number. A small number (usually less than 0.05) means the data are not like a bell shape.

To find the best value of d, three ways of testing the data were used: Kwiatkowski-Phillips-Schmidt-Shin (KPSS), Augmented Dickey-Fuller (ADF), and Phillips-Perron (PP). They check if the data have something that makes them not steady. The KPSS test says the data are steady if they have a line but no curve (Shin and Schmidt). The ADF and PP tests say the data are steady if they do not have something that makes them change over time (Cheung and Lai) (Breitung and

Franses). If the KPSS test says no and the ADF or PP test says yes, it means the data have a curve and need to be taken away. If the KPSS test says yes and the ADF or PP test says no, it means the data do not have a curve and do not need to be taken away. The best value of d is the smallest number of times to take away the data that make all three tests agree on being steady.

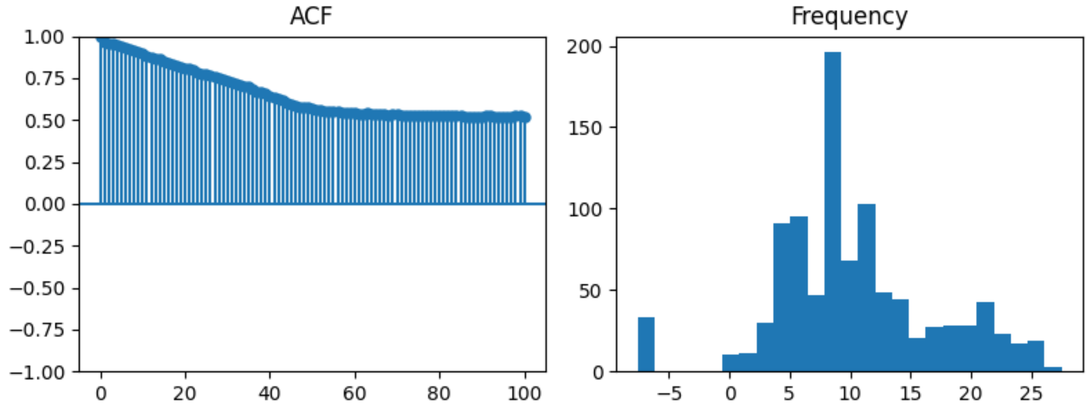

Figure 1: Logarithmic transformation to data for normalizing (75:25) (Author own)

Figure one shows the spread of the frequency after having the logarithmic transformation applied to its data. Overall, the data has been able to achieve a singular large peak with fluctuations.

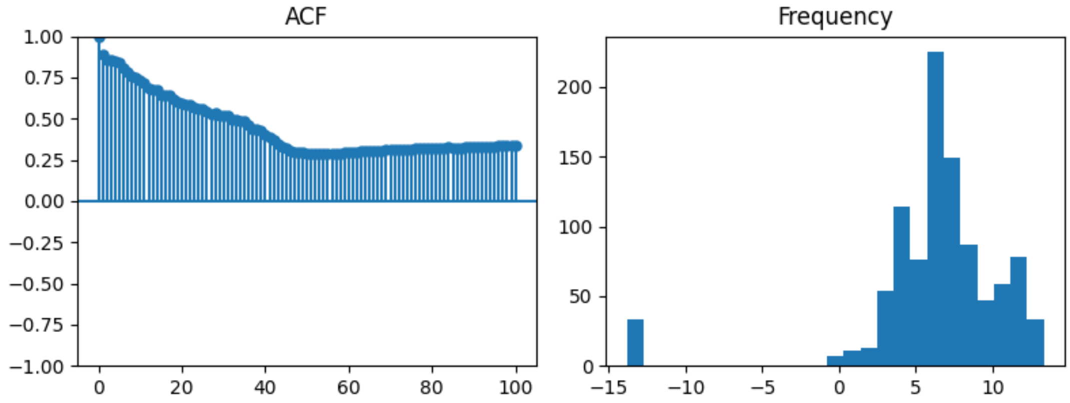

Figure 2: BoxCox transformation to data for normalizing (75:25) (Author own)

Figure two also shows the spread of the frequency after having the BoxCox transformation applied to its data. Overall, while the data has been able to achieve an unbalanced normal shape which may lead to a decrease in forecasting accuracy.

By comparing both results of transformation, the Logarithmic transformation should theoretically have a better forecast than the data with the BoxCox transformation.

3.3.2 SARIMA Model

Pmdarima already has two methods for endogenous and exogenous variables. The BoxCox and logarithm transformation are for endogenous transformations, and the DataFeaturizer and FourierFeaturizer are for exogenous transformations. For this experiment, the endogenous transformation was used as the dataset used does not consider any other variables that may have affected the data, it is a univariate data. It is purely just looking at the increase and decrease in the number of COVID-19 cases. If the experiment was to use exogenous transformation, the model would have considered outside variables that are not existent on the dataset and their effects.

To accomplish the most optimal results, both methods for the endogenous transformation were utilized into the SARIMA model and compared to find the superior result. The automatic parameter detector package from the pmdarima was utilized as it eliminates the need to manually change the parameter values. The package determined the most optimal parameters by calculating the AIC value each time. Once it found the lowest AIC value on a set of parameters, it calculated the MAE, MdAPE, and RMSE values.

3.4.1 PROPHET Data Processing

One of the main steps in preparing the data for the prophet model was to filter out the dates that had irregular effects on the time series. If the data has irregularities, the model might not be able to capture the true patterns and make accurate predictions. Therefore, the data was cleaned by removing the dates that were known to have irregular effects. The irregular effects were determined by the sudden increase or decrease in the data as they would act as weights for the PROPHET model (Taylor and Letham). These dates were then given as a list of holidays or outliers parameters in the prophet model, which made the model skip them when fitting the data.

3.4.2 PROPHET Model

The PROPHET model does not require mathematical transformations for it to fit and forecast data. Instead, it can automatically determine the seasonality and parameters necessary in the process of feeding the data into the model (Meta). All that was required was to simply define the model and have it fit with the data that has been processed beforehand. After the forecasted results, the MAE, MdAPE, and RMSE were calculated to evaluate the accuracy of the forecast.

IV. Results

4.1 SARIMA Results

	SARIMA (BoxCox)	SARIMA (Log)
80% (Training) 20% (Forecast)	ARIMA(2,1,2)(2,0,2)[7]	ARIMA(2,1,3)(2,0,2)[7]
MAE	60279.269	54900.033
MdAPE	4.897	4.451
SMAPE	123.37	122.353
RMSE	69505.634	63122.743

75% (Training) 25% (Forecast)	ARIMA(2,1,2)(2,0,2)[7]	ARIMA(2,1,3)(2,0,2)[7]
MAE	41903.115	36198.114
MdAPE	2.589	2.101
SMAPE	106.078	102.013
RMSE	50737.435	45327.182
70% (Training) 30% (Forecast)	ARIMA(2,1,2)(2,0,1)[7]	ARIMA(2,1,3)(2,0,2)[7]
MAE	887210.371	31716.685
MdAPE	27.245	0.966
SMAPE	155.183	97.811
RMSE	1221044.958	47788.971

Table 1: 80:20, 75:25, 70:30, Results of SARIMA Model (Author own)

Based on the metric evaluations it can be observed that 75% training data and 25% testing data performed most optimally compared to the other ranges of data. When comparing the two data transformation SARIMA models, it is evident that the logarithm data transformation has performed better in all metrics.

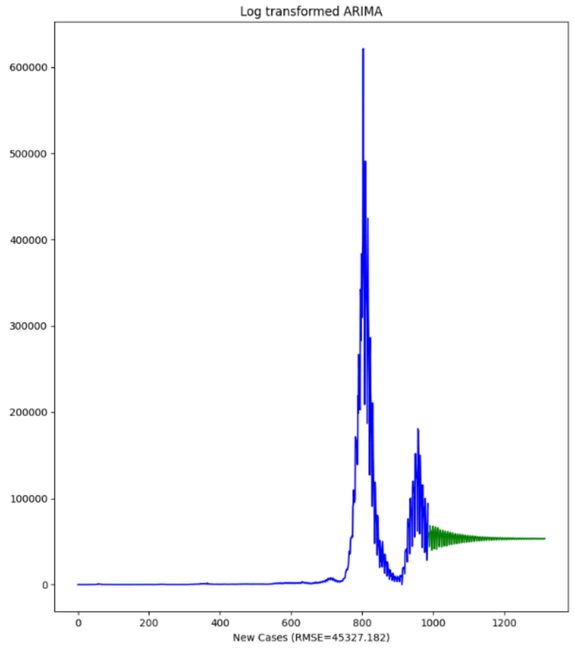

Figure 3: SARIMA(Log) Result (75:25) (Author own)

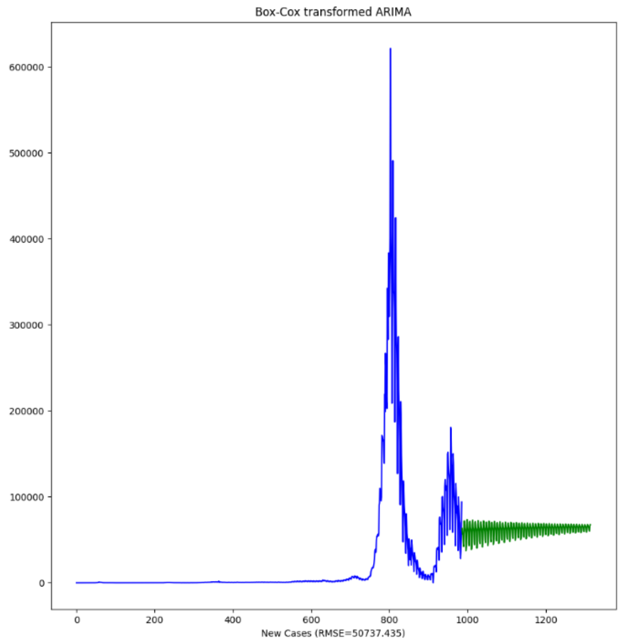

Figure 4: SARIMA(BoxCox) Result (75:25) (Author own)

Figure three and four shows the forecasts being displayed. The blue line indicates the real data, and the green line shows the forecast. Based on the graph, it is ideal to be more careful when handling forecasts that goes on for a long time.

4.2 PROPHET Results

	MAE	MdAPE	SMAPE	RMSE
80% (Training) 20% (Forecast)	58686.53	4.568	124.548	66476.6
75% (Training) 25% (Forecast)	127105.3	7.913	143.054	135686.54
70% (Training) 30% (Forecast)	32370.36	1.002	102.414	50397.04

Table 2: 80:20, 75:25, 70:30, Results of PROPHET Model (Author own)

Results show that the PROPHET model performed most optimally when given 70% training data and 30% testing data.

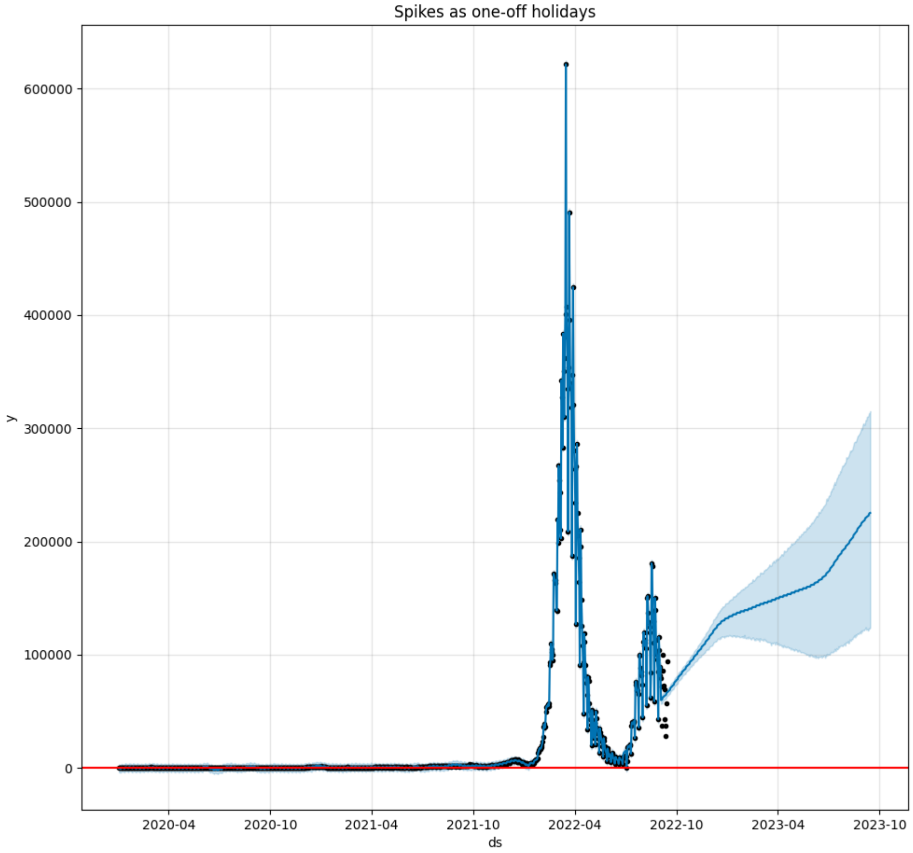

Figure 5: PROPHET Result (75:25) (Author own)

Figure five shows the forecast of the model with the inclusion of upper and lower error margins. Based on the forecast, the forecast is increasing exponentially in order to account for the large spike in data occurring at around 2022 March.

4.3 Comparing Results.

	SARIMA (Log)	PROPHET
80% (Training) 20% (Forecast)	ARIMA(2,1,3)(2,0,2)[7]	
MAE	54900.033	58686.53

MdAPE	4.451	4.568
SMAPE	122.353	124.548
RMSE	63122.743	66476.6
75% (Training) 25% (Forecast)	ARIMA(2,1,3)(2,0,2)[7]	
MAE	36198.114	127105.3
MdAPE	2.101	7.913
SMAPE	102.013	143.054
RMSE	45327.182	135686.54
70% (Training) 30% (Forecast)	ARIMA(2,1,3)(2,0,2)[7]	
MAE	31716.685	32370.36
MdAPE	0.966	1.002
SMAPE	97.811	102.414
RMSE	47788.971	50397.04

Table 3: Compared result between SARIMA and PROPHET (Author own)

Results show that the SARIMA(Log) has outperformed the PROPHET model in all data ranges.

V. Discussion

There were several limitations to this study. First, the data did not show a consistent seasonality throughout the years in South Korea. While there were 'waves' of COVID-19 cases occurring, they seemingly happened to occur during unprecedented times. For further testing in seasonality, the comparison between the number of cases occurring daily in America and South Korea was conducted. The data did not show any sort of correlation in trend, both countries were experiencing unique waves of the COVID-19. Secondly, it may be difficult to generalize the results from South Korea to the world as it seemed that all countries around the world experienced different effects of the COVID-19. However, as both models performed well, they could be used to help predict the number of cases within South Korea.

Prior to building the SARIMA model, the ARIMA model was used and tested but returned inaccurate forecasts which made the experiment not fair for comparison. Therefore, further research was conducted to use the SARIMA model and help improve the forecast.

VI. Conclusion

A further extension of the SARIMA model from the ARIMA model is the SARIMAX model. To summarize, the SARIMAX model has an additional component 'X' which accounts for exogenous variables. This component allows for the model to account for external variables that may have possible implications to the data. This in turn helps the model make more accurate forecasts than it could with just a single variable. While the ARIMA and SARIMA model are both univariate models the SARIMAX model is a multivariate model (Arunraj et al.).

Therefore, a continuation of this research could be completed with the usage of the SARIMAX model and another forecasting model such as the long-short term memory (LSTM) model or the light gradient boosting machine (LightGBM) model. These models are more complex than the models used in the research which would hypothetically return forecasts that are more accurate.

Overall, the usage of the SARIMA and PROPHET model for this experiment had been a success despite the limitations to the point where it was possible for real-life application. However, as both forecasting models were only forecasted on a 3-year record of data, it would require for a routine update on the dataset and modifications of parameters to be used for practical.

Further thoughts on the application of the two models has brought the idea to test these models for other applications such as stocks or climate changes. Stocks and climate changes have a larger dataset as they have recorded for over a decade. The enlargement in data would most definitely help improve the performance of the models and perhaps give different results as of this experience.

VII. Citations

Smith, Taylor G. “Pmdarima 2.0.3 Documentation.” *User Guide: Contents - Pmdarima 2.0.3 Documentation*, Feb. 2017, alkaline-ml.com/pmdarima/user_guide.html.

BURNHAM, KENNETH P., and DAVID R. ANDERSON. “Multimodel Inference - Understanding AIC and BIC in Model Selection.” *BBURN.DVI*, Nov. 2004, www.sortie-nd.org/lme/Statistical%20Papers/Burnham_and_Anderson_2004_Multimodel_Inference.pdf.

Hyndman, Rob J., and George Athanasopoulos. “Chapter 8 ARIMA Models | Forecasting: Principles and Practice (2nd Ed).” *Chapter 8 ARIMA Models | Forecasting: Principles and Practice (2nd Ed)*, May 2018, otexts.com/fpp2/arima.html.

Alonso Brito, Gustavo Reinel, et al. “Comparison Between SARIMA and Holt–Winters Models for Forecasting Monthly Streamflow in the Western Region of Cuba - SN Applied Sciences.” *SpringerLink*, 29 May 2021, https://doi.org/10.1007/s42452-021-04667-5.

Taylor, Sean J., and Benjamin Letham. “Forecasting at Scale.” *The American Statistician*, vol. 72, no. 1, Aug. 2017, pp. 37–45. https://doi.org/10.1080/00031305.2017.1380080.

Shin, Yongcheol, and Peter Schmidt. “The KPSS Stationarity Test as a Unit Root Test.” *Economics Letters*, vol. 38, no. 4, Elsevier BV, Apr. 1992, pp. 387–92. *Crossref*, https://doi.org/10.1016/0165-1765(92)90023-r.

Mushtaq, Rizwan. “Augmented Dickey Fuller Test.” *SSRN Electronic Journal*, Elsevier BV, Aug. 2011. *Crossref*, https://doi.org/10.2139/ssrn.1911068.

Breitung, Jörg, and Philip Hans Franses. “ON PHILLIPS–PERRON-TYPE TESTS FOR SEASONAL UNIT ROOTS.” *Econometric Theory*, vol. 14, no. 2, Cambridge UP (CUP), Apr. 1998, pp. 200–21. *Crossref*, https://doi.org/10.1017/s0266466698142032.

Cheung, Yin-Wong, and Kon S. Lai. “Lag Order and Critical Values of the Augmented Dickey–Fuller Test.” *Journal of Business & Economic Statistics*, vol. 13, no. 3, Informa UK Limited, July 1995, pp. 277–80. *Crossref*, https://doi.org/10.1080/07350015.1995.10524601.

Taylor, Sean J., and Benjamin Letham. “Handling Shocks.” *Prophet*, 6 Sept. 2022, facebook.github.io/prophet/docs/handling_shocks.html.

Meta. “Quick Start.” *Prophet*, 25 June 2022, facebook.github.io/prophet/docs/quick_start.html.

Cong, Jing, et al. “Predicting Seasonal Influenza Based on SARIMA Model, in Mainland China From 2005 to 2018.” *MDPI*, vol. 16, no. 23, Nov. 2019, p. 4760. *MDPI*, https://doi.org/10.3390/ijerph16234760.

Hosono, Takahiro, et al. “A Minimalistic Approach for Evapotranspiration Estimation Using the Prophet Model.” *Hydrological Sciences Journal*, vol. 65, no. 12, July 2020, pp. 1994–2006. *Taylor & Francis Online*, https://doi.org/10.1080/02626667.2020.1787416.

Rani Samal, K. Krishna, et al. “Time Series Based Air Pollution Forecasting Using SARIMA and Prophet Model.” *ITCC 2019: Proceedings of the 2019 International Conference on Information Technology and Computer Communications*, vol. 1, Aug. 2019, pp. 80–85. *ACM*, https://doi.org/10.1145/3355402.3355417.

Arunraj, Nari Sivanandam, et al. “International Journal of Operations Research and Information Systems.” *Application of SARIMAX Model to Forecast Daily Sales in Food Retail Industry.*, vol. 7, no. 2, 2016, pp. 1–21. *IGI Global*, https://doi.org/10.4018/IJORIS.2016040101.

VIII. Appendix

SARIMA (BoxCox): box-sarima.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

import pmdarima as pm
from pmdarima.model_selection import train_test_split
print(f"Using pmdarima {pm.__version__}")

df = pd.read_csv('south-korea-gathered-data.csv')
print(df.head())

dataSize = len(df)
train_size = int(0.75 * dataSize)

y_train = df['New_cases'][:train_size]
y_test = df['New_cases'][train_size:]
```

```
from pmdarima.utils import tsdisplay
from pmdarima.preprocessing import BoxCoxEndogTransformer


y_train_bc, _ = BoxCoxEndogTransformer(lmbda2=1e-6).fit_transform(y_train)
tsdisplay(y_train_bc, lag_max=100)


from scipy.stats import normaltest
print(normaltest(y_train_bc)[1])


from pmdarima.pipeline import Pipeline


fit2 = Pipeline([
    ('boxcox', BoxCoxEndogTransformer(lmbda2=1e-6)),
    ('arima', pm.AutoARIMA(trace=True,
                         suppress_warnings=True,
                         m=7,
                         seasonal=True,
                         seasonal_test='ocsb',
                         ))
])


fit2.fit(y_train)
print(fit2.summary())
```

```
from sklearn.metrics import mean_squared_error as mse

def plot_forecasts(forecasts, title, figsize=(8, 12)):
    x = np.arange(y_train.shape[0] + forecasts.shape[0])

    fig, axes = plt.subplots(2, 1, sharex=False, figsize=figsize)

    axes[0].plot(x[:y_train.shape[0]], y_train, c='b')
    axes[0].plot(x[y_train.shape[0]:], forecasts, c='g')
    axes[0].set_xlabel(f'New Cases (RMSE={np.sqrt(mse(y_test, forecasts)):.3f})')
    axes[0].set_title(title)

    resid = y_test - forecasts
    _, p = normaltest(resid)
    axes[1].hist(resid, bins=15)
    axes[1].axvline(0, linestyle='--', c='r')
    axes[1].set_title(f'Residuals (p={p:.3f})')

    plt.tight_layout()
    plt.show()

forecasts = fit2.predict(y_test.shape[0])
```

```
plot_forecasts(forecasts, title='Box-Cox transformed ARIMA') # Added this line


from sklearn.metrics import mean_absolute_error as mae
from sklearn.metrics import mean_absolute_percentage_error as mape
from sklearn.metrics import median_absolute_error as mdae
from pmdarima.metrics import smape


mae_value = mae(y_test, forecasts)
mdape_value = mdae(y_test, forecasts) / np.median(y_test)
smape_value = smape(y_test, forecasts)
mape_value = mape(y_test, forecasts)
rmse_value = np.sqrt(mse(y_test, forecasts))


print(f'MAE: {mae_value:.3f}')
print(f'MdAPE: {mdape_value:.3f}')
print(f'SMAPE: {smape_value:.3f}')
print(f'MAPE: {mape_value:.3f}')
print(f'RMSE: {rmse_value:.3f}')
```

SARIMA (Log): log-sarima.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


import pmdarima as pm

from pmdarima.model_selection import train_test_split

print(f"Using pmdarima {pm.__version__}")


df = pd.read_csv('south-korea-gathered-data.csv')

print(df.head())


dataSize = len(df)

train_size = int(0.7 * dataSize)


y_train = df['New_cases'][:train_size]

y_test = df['New_cases'][train_size:]


from pmdarima.utils import tsdisplay

from pmdarima.preprocessing import LogEndogTransformer


y_train_log, _ = LogEndogTransformer(lmbda=1e-6).fit_transform(y_train)

tsdisplay(y_train_log, lag_max=100)


from scipy.stats import normaltest

print(normaltest(y_train_log)[1])
```

```
from pmdarima.pipeline import Pipeline


fit3 = Pipeline([
    ('log', LogEndogTransformer(lmbda=1e-6)),
    ('arima', pm.AutoARIMA(trace=True,
                           suppress_warnings=True,
                           m=7,
                           seasonal=True,
                           seasonal_test='ocsb',
                           ))
])


fit3.fit(y_train)
print(fit3.summary())


from sklearn.metrics import mean_squared_error as mse


def plot_forecasts(forecasts, title, figsize=(8, 12)):
    x = np.arange(y_train.shape[0] + forecasts.shape[0])


    fig, axes = plt.subplots(2, 1, sharex=False, figsize=figsize)
    axes[0].plot(x[:y_train.shape[0]], y_train, c='b')

    axes[0].plot(x[y_train.shape[0]:], forecasts, c='g')

    axes[0].set_xlabel(f'New Cases (RMSE={np.sqrt(mse(y_test, forecasts)):.3f})')

    axes[0].set_title(title)


    resid = y_test - forecasts

    _, p = normaltest(resid)

    axes[1].hist(resid, bins=15)

    axes[1].axvline(0, linestyle='--', c='r')

    axes[1].set_title(f'Residuals (p={p:.3f})')


    plt.tight_layout()

    plt.show()


forecasts_log = fit3.predict(y_test.shape[0])


plot_forecasts(forecasts_log, title='Log transformed ARIMA')


from sklearn.metrics import mean_absolute_error as mae

from sklearn.metrics import mean_absolute_percentage_error as mape

from sklearn.metrics import median_absolute_error as mdae

from pmdarima.metrics import smape
```

```
mae_value_log = mae(y_test, forecasts_log)
mdape_value_log = mdae(y_test, forecasts_log) / np.median(y_test)
smape_value_log = smape(y_test, forecasts_log)
mape_value_log = mape(y_test, forecasts_log)
rmse_value_log = np.sqrt(mse(y_test, forecasts_log))

print(f'MAE: {mae_value_log:.3f}')
print(f'MdAPE: {mdape_value_log:.3f}')
print(f'SMAPE: {smape_value_log:.3f}')
print(f'MAPE: {mape_value_log:.3f}')
print(f'RMSE: {rmse_value_log:.3f}')
```

PROHPHET: prophetandseason.ipynb

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import mean_absolute_error, mean_absolute_percentage_error
from sklearn.metrics import median_absolute_error as mdae
from prophet import Prophet
from prophet.plot import plot_plotly, plot_components_plotly

spikes = pd.DataFrame([
    {'holiday': 'spike_1', 'ds': '2022-01-09', 'lower_window': 0, 'ds_upper': '2022-06-26'},
    {'holiday': 'spike_2', 'ds': '2022-07-03', 'lower_window': 0, 'ds_upper': '2022-09-02'},
])
for t_col in ['ds', 'ds_upper']:
    spikes[t_col] = pd.to_datetime(spikes[t_col])
spikes['upper_window'] = (spikes['ds_upper'] - spikes['ds']).dt.days
spikes

df = pd.read_csv('south-korea-gathered-data-prophet.csv')
df.head()

total_rows = df.shape[0]
train_rows = int(total_rows * 0.7)
test_rows = int(total_rows * 0.3)
train = df.iloc[:train_rows]
test = df.iloc[-test_rows:]
y_test = test['y'].values

m2 = Prophet(holidays=spikes)
m2.fit(train)
future2 = m2.make_future_dataframe(periods=365)
forecast2 = m2.predict(future2)

m2.plot(forecast2)
```

```
plt.axhline(y=0, color='red')
plt.title('Spikes as one-off holidays')
plt.show()

m2.plot_components(forecast2)

y_pred = forecast2['yhat'].values[-test_rows:]

rmse = np.sqrt(np.mean((y_test - y_pred)**2))
print(f'The RMSE value is {rmse:.2f}')

mae = mean_absolute_error(y_test, y_pred)
print(f'The MAE value is {mae:.2f}')

mape = mean_absolute_percentage_error(y_test, y_pred)
print(f'The MAPE value is {mape:.2f}')

mdape_value = mdae(y_test, y_pred) / np.median(y_test)
print(f'The MdAPE value is {mdape_value:.3f}')

from pmdarima.metrics import smape
smape_value = smape(y_test, y_pred)
print(f'The SMAPE value is {smape_value:.3f}')
```

[redacted]

Data 1: south-korea-gathered-data.csv

date	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
1/3/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/4/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/5/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/6/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/7/2020	KR	Republic of Korea	WPRO	0	0	0	0

Data 2: south-korea-gathered-data-prophet.csv

ds	Country_code	Country	WHO_region	y	Cumulative_cases	New_deaths	Cumulative_deaths
1/3/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/4/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/5/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/6/2020	KR	Republic of Korea	WPRO	0	0	0	0
1/7/2020	KR	Republic of Korea	WPRO	0	0	0	0

Data 3: usa-gathered-data.csv

Date_rep	Country_	Coun	WHO_re	New_c	Cumulative	New_de	Cumulative_

orted	**code**	**try**	**gion**	**ases**	**_cases**	**aths**	**deaths**
1/3/2020	US	United States of America	AMRO	0	0	0	0
1/4/2020	US	United States of America	AMRO	0	0	0	0
1/5/2020	US	United States of America	AMRO	0	0	0	0
1/6/2020	US	United States of	AMRO	0	0	0	0

		America					
1/7/2020	US	United States of America	AMRO	0	0	0	0