BINDURA UNIVERSITY OF SCIENCE EDUCATION

FACULTY OF SCIENCE AND ENGINEERING

DEPARTMENT OF STATISTICS AND MATHEMATICS

FORECASTING COVID 19 IN ZIMBABWE USING TIME SERIES

BY MANGOZA PRAISE

B1852411

A DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS OF THE BACHELOR OF SCIENCE HONOURS DEGREE IN STATISTICS AND FINANCIAL MATHEMATICS

SUPERVISOR: MS J.C. PAGAN'A

JUNE 2022

APPROVAL FORM

I Mangoza Praise, do hereby declare that this submission is my work apart from the references of other people’s work which has been acknowledged. I hereby declare that this work has neither been presented in whole nor in p for any degree at this university.

MANGOZA PRAISE …………………………18/06/2022

Certified by

J.C. PAGAN’A J. …..17/06/2022……..

Supervisor

MR MAPUWEI ………………………………………… …………….

Chairperson

DEDICATION

I dedicate this project to my beloved family and friends for their passionate efforts and emotional and financial support.

ACKNOWLEDGEMENTS

I would want to convey my gratitude to Ms J.C Pagan'a, my supervisor, for her patience, encouragement, and wisdom. The supervisor's constructive comments, suggestions, corrections, and remarks were quite beneficial to the learner. Without her constant guidance, I would not have been able to finish this dissertation.

My sincere gratitude to my classmates for the fantastic time we shared. Last but not least, I would want to express my gratitude to the Department of Statistics and Mathematics staff, particularly Mr Mapuwei, Mr Basira, and Mr Kusotera only to mention a few for fostering an environment that allowed me to finish my studies.

ABSTRACT

Many people all over the world have been infected, and many others have died as a result of the ongoing new Coronavirus outbreak around the world (Covid 19). In order to avert deaths, it is critical to anticipate future infection cases and the viral transmission rate so that healthcare providers can prepare ahead of time. The research community has an analytical and difficult real-world difficulty in accurately projecting Covid 19 instances, hence the need to employ various models to predict the dynamics of Covid 19 infection. In this study, the researcher used daily level Covid 19 cumulative cases data for the entire country of Zimbabwe to model the dynamics of Covid 19 infection. The data used was from March 2020 to January 31, 2022. ARIMA and LSTM forecasting models were used to model the trends and make predictions. The mean absolute percentage error was used to assess the models' effectiveness. Findings from this study have revealed that the LSTM model is more effective at forecasting Covid 19 cases. The forecasting results have the potential to help countries devise actions to stop the virus from spreading.

Contents

Figures

Chapter 1

1.1 INTRODUCTION

The Covid 19 virus is a disease-causing coronavirus that caused a global pandemic in the year 2019. The virus quickly spread over the globe and has now become a huge security threat to humanity. While the virus poses a threat to people's lives and safety, it has also wreaked havoc on the economics of several countries. Several businesses have closed, it has become difficult to find work, and the lives of individuals have been significantly impacted. As a result, many people have begun to use time series models to forecast how the virus would spread, notably, Bin Zhao in Wuhan Province, where Covid 19 is claimed to have started, analyzed the novel coronavirus's spreading tendency and short-term forecast, which will aid governments in better understanding the epidemic's development trend and ensuring more suitable preparation and prompt action.

According to Al-Turaiki and Alrasheed (2020), anticipating the pandemic's route is critical for tracking its spread. Covid 19 data is a wonderful example of time series data that may be forecasted using different methods. Even though there are many models used for forecasting, drawing broad theoretical conclusions about their relative advantages is problematic.

Different time series models for projecting the spread of Covid 19 in Zimbabwe are evaluated. Time series are created when processes change over time and are recorded. A time series is a collection of data acquired at different times or across different periods MA, AR, ARIMA, GARCH, TARCH, EGARCH, FIGARCH, CGARCH, and other time series models are among them (Engle 2018).

The Covid 19 has not spared Zimbabwe as a country, several people in Zimbabwe have died as a result of the pandemic since it began. The Zimbabwean government devised and implemented methods to contain and reduce the pandemic's impact on human health and the economy. It imposed a lockdown from mid-March to the end of June 2020, prohibiting most industrial, commercial, and transportation activity save for critical services to reduce pollution (Zimbabwean *Ministry of health and population, 2021*). The models were created using Covid 19 data from March 2020 to January 2022, which was reported in Zimbabwe. COVID 19 has been the subject of numerous investigations since the emergence of the coronavirus pandemic. There appears to be limited literature on time series analysis for COVID 19 however. This chapter provides an overview of the primary elements under investigation, as well as the study's goal, objectives, and assumptions. It also defines the situation at hand, as well as its significance and justification. Furthermore, the chapter defines the study's bounds, as well as its restrictions and delimitations.

There are five chapters in this study. Following this chapter, the second examines the relevant and related theoretical and empirical literature on the use of time-series analysis in forecasting Covid 19. The third chapter looks into time series methods in greater depth, including the collection and explanation of the datasets and models used. The fourth chapter provides a thorough examination of the data as well as a thorough discussion of the research findings. Finally, in Chapter 5, the findings are summarized, recommendations are offered, and opportunities for further investigation are identified.

1.2 BACKGROUND OF THE STUDY

The coronavirus, since its discovery, in China Wuhan in 2019 has infected millions of individuals all over the world. The Covid 19 disease causes serious respiratory challenges and is rapidly spreading. This virus can make you sick if you have a chronic illness like heart disease or diabetes, or if your immune system is weak. Lockdowns, curfews, and travel bans were enacted all across the world to stem the spread of the disease.

According to the New York Times, the Maryland Coronavirus Map and Case Count the Covid 19 virus is caused by the SARS-COV-2 virus (WHO 2022). Many people will only experience minor symptoms and will recover without the need for further treatment. Some, on the other hand, will progress to the point where they will require medical attention. If you are close to someone who has Covid 19, you can spread it by inhaling the virus or encountering a contaminated surface.

A fundamental problem is a lack of historical data that can help experts assess the disease's impact and estimate its future dynamics. Forecasting Covid 19's evolution is critical for public health planning and decision-making. One approach to do this is to estimate the number of active cases at any one time properly. Patients with conditions like hypertension, diabetes, heart failure, chronic kidney disease, and cancer were found to have increased mortality and fatality rates when they developed COVID 19. Ferdinand and Nasser argued that the prevalent cardiovascular disease among African Americans which is direct links to poor health care conditions in the community is to be blamed for the coronavirus cases in the African Americans.

Predicting a future value or category at a specific point in time is done using time series forecasts. Today, time-series forecasting is used to estimate possible demand for a product, resource allocation, financial performance, predictive maintenance, and a variety of other applications in a wide range of industries, from energy and retail to transportation and banking. Even though time series forecasting can revolutionize business models and increase revenues, it has yet to be implemented. (Anais Dotis, Georgiou 2021) Time series forecasting is a way of trying to establish what will happen in the future. To make these forecasts historical trends are utilized and an assumption that future trends will be similar is made. Forecasting is the process of using models to predict future values based on historical data. For circumstances with a time component, time series forecasting, a data-driven method of effective prediction, is necessary. Many writers have utilized time series analysis to forecast Covid 19 mortality patterns and instances.

The Director-General of the World Health Organisation on January 30, 2020, has designated the outbreak of the coronavirus disease 2019 (COVID 19) as a public health emergency. Worldwide hundreds of people lost their loved ones because of this disease. Following the

WHO announced lockdown measures to minimize the spread of this disease. Through statutory Instrument 83 of 2020, Section 4(1) of SI 83 of 2020.

1.3 STATEMENT OF THE PROBLEM

Although time-series analytic approaches have yielded promising results in the study of infectious diseases, finding an appropriate methodology to analyze Covid 19 remains a challenge, given that it is one of the most significant and deadly health crises in recent years. The spread of Covid 19 is difficult to model accurately as compared to other diseases because of the large number of overall cases, the high contagiousness, quick dissemination, and the difficulties in reliably testing and tracking patients. Covid 19 analysis is more difficult than other diseases because of the large number of overall cases, the high contagiousness and quick dissemination, and the difficulties in reliably testing and tracking patients. The challenge is if those time series analysis techniques are still available and valid in the Covid 19 scenario, and if so, whether we can figure out the optimal model (if one exists) for representing case patterns and so predicting future trends.

1.4 AIM OF THE STUDY

The sole objective of the study is to forecast the dynamics of Covid 19 in Zimbabwe using time series models.

1.5 RESEARCH OBJECTIVES

The goal of the research is to:

- To establish trends of Covid 19 in Zimbabwe
- To predict the future trends of Covid 19 using time series models
- To determine the best performing model between ARIMA and Deep Learning models in terms of mean absolute percentage error

1.6 RESEARCH QUESTION

The study seeks to answer the following questions:

1. Is the trend in the Covid 19 cases increasing or decreasing?
2. To what extent can time series models be used in forecasting the spread of Covid 19?
3. Which model between ARIMA and Deep Learning is the best performing as measured by mean absolute percentage error?

1.7 SIGNIFICANCE OF THE STUDY

This study is important for the following:

1.7.1 GOVERNMENT OF ZIMBABWE

The research findings may help the government in the sense that forecasting the dynamics and impact of Covid 19 is establishing itself as the backbone for maximizing the utilization of available resources in hospitals and improving management options for infected patients and decision making.

1.7.2 BINDURA UNIVERSITY OF SCIENCE EDUCATION

It is regarded as probable that the study is an additional element to the library material for other researchers as reference materials to those who would like to carry out research in this area. Henceforth, the research may give an idea for further studies related to this study.

1.7.3 THE RESEARCHER

The researcher 's understanding of how to use different software models will be broadened and also improve the researcher's skills for future use in the same area.

1.7.4 LITERATURE

There seems to be a gap in the literature on a time series analysis specifically on forecasting COVID 19. This research contributes to the existing literature on the evaluation of time series models in forecasting Covid 19.

1.8 SCOPE OF THE STUDY

The study focuses on using statistical and mathematical modeling in machine learning to forecast Covid 19. The research will use historic data obtained from the official website of the Ministry of Health and Ministry of information. The data is imported into the Python package and analyzed.

1.9 ASSUMPTIONS

- The data utilized in this study comes from reputable sources and has not been tampered with.
- The sample used is enough to model the whole population
- There is no missing data, and the month is defined as a time variable

1.10 LIMITATIONS

- The study focuses on Zimbabwe exclusively.
- The study spans from March 2020 to January 31, 2022

1.11 DEFINITION OF TERMS

COVID 19 -Coronavirus Disease 2019

SARS - severe acute respiratory syndrome

Time series - It is a sequence of numbers collected at regular intervals over some time

1.12 CONCLUSION

The background checks that led to the researcher's decision to research this topic, are covered in this section of the study and this chapter detailed the investigation's primary findings. The backdrop, problem description, study purpose, limitations, and assumptions were all addressed in this chapter. In the chapter, the value of research to the government was also emphasized.

CHAPTER TWO

2.1 INTRODUCTION

In this chapter, we will quickly discuss the available literature on the applicability of time series models in forecasting Covid 19, which will serve as a review of the concepts from earlier studies and associated works of literature to briefly. A research case framework is presented and covers the main focus of the dissertation described here. Many different models have been used to try and forecast Covid 19 cases. Most of this research, however, was carried out in developed countries and may or may not be empirical to third world countries and developing countries.

2.2 THEORETICAL LITERATURE REVIEW

The epidemic has had a significant impact on the workplace and lifestyle. Lockdowns and movement limitations imposed by Covid 19 have spawned e-learning and opened up new possibilities for GIS applications. The lockout had a good impact on the environment, particularly in densely populated and industrialized countries with significant levels of air pollution. A good number of review papers related to this study are available. Several time series models have been developed and used to forecast Covid 19 in many parts of the world.

When it comes to analyzing time series, it can be traced back to the employment of time series models in the study of infectious illnesses, and since then, various models have been created to explain the disease transmission process, and are still being developed. Numerous statistical or data-based models have been employed for purposes of modeling several infectious diseases, including Hepatitis A virus infection, HIV, and ovine John's disease, among others (Brown and Ozanne, 2019). Time series theories can be used to model data in addition to regular mathematical models. Time series models have a long history of being used to analyze data. Several models for predicting illness trends have been created, and a family of models known as the Generalized Autoregressive Conditional Heteroscedastic (GARCH) models have been extensively employed in the study of volatile clusters with suitable modifications and expansions. Time series model features have a multiplicity of uses and have thus been used by researchers in other fields of study (Tyagi 2021). Similarly, the Autoregressive Integrated Moving Average (ARIMA) class of models and the Random walk model have been widely used in practically all domains for modeling time series data.

Most of the time, these models are used to fit historical data and estimate the future. This ability of a model boosts the model's underlying theory's believability. These models produce the best projections when we need to forecast the future behavior of a system based on prior observations, which has led to their widespread application in studying disease trends.

The main aim of this research is to show that with a better comprehension of the data, established asset return prediction models can be utilized to anticipate and forecast pandemic diseases (Covid 19 in this case).

2.3 TIME SERIES ANALYSIS

The term 'time series' refers to recordings of processes that change over time (Robinson, 2020). A recording can be a continuous line r a set of collection of individual observations. (Ross Ihaka 2005). According to Madsen (2008), Time series analysis deals with statistical methods for analyzing and modeling an ordered sequence of observations. In general time series seeks to understand the underlying context of the relevant data points through the derivation forecast of future values from recorded past values. This study mainly focuses on forecasting Covid 19 cases using ARIMA and LSTM models.

ARIMA models are one of the most extensively utilized approaches. Three basic time series models are used to create ARIMA models.

a) Autoregressive (AR)

b) Moving Average (MA)

c) Autoregressive Moving Average (ARMA)

In the MA model, the current value of the time series is a linear function of its current and prior residual values. The ARMA model combines AR and MA, taking into account both past values and residuals. The time series necessary for AR, MA, and ARMA models are stationary processes, which means that the series' mean and covariance does not change over time.

The Auto-Regressive Integrated Moving Average (ARIMA) model moves away from deep learning models and into the realm of traditional machine learning (Hayes,2021). This is a Linear Regression model that excels in time series forecasting, which makes it ideal for our Covid 19 research. The ARIMA model differs from other machine learning models in that it does not include exogenous variables as features. Instead, it makes predictions based purely on the target variable's past values. As a result, it's a model that's heavily mathematical and statistically driven (Yiu 2020).

AR: Auto-Regressive

The target variable in this model is regressed on its past, which is referred to as autoregressive. As the X variable, the target's lagged data are used.

$$Y = B0 + B1 * Y_{lag1} + B2 * Y_{lag2} + \cdots + Bn * Y_{lagn}$$

The above equation shows that the current value Y is a linear function of the past n values. The regression beta values that are fitted to the model during training are known as B values. This equation can be changed to forecast the future, as seen in the equation below.

$$Y_{forward1} = B0 + B1 * Y + B2 * Y_{lag1} + B3 * Y_{lag3} + \cdots Bn * \ Y_{lag\,(n-1)}$$

I: Integrated

Integrated signifies that a separate equation is used to apply a differencing step to the data:

$$Y_{foward\,1} - Y = B0 + B1 * \left(Y - Y_{lag1}\right) + B2 * \left(Y_{lag1} - Y_{lag2}\right) + \cdots$$

The preceding equation demonstrates that Y's future value is linearly related to its previous values. During time series forecasting, this is done to make the Y values mean-variance stationery.

MA: Moving Average

A moving average model is given by the following equation

$$Y = B0 + B1 * E_{lag1} + B2 * \ E_{lag2} + \cdots + Bn * E_{lagn}$$

In the moving average model, E represents the random residual discrepancies between the model and the target variable. This means that E is the difference between the exact value and the model's estimated value. Due to its basic structure and lack of exogenous variables, the ARIMA model serves as a solid baseline model when all of this is taken into account.

The formulation of ARIMA is a complicated process, but in summary, it consists of four steps:

1. Identification of the ARIMA (p, d, q) structure
2. Estimating the coefficients of the formulation
3. Fitting test on the estimated formulation
4. Predicting future results using historical data

The ARIMA model is a stochastic process with three parameters: p, d, and q, where p represents the Auto-Regressive AR (p) process, d stands for the integration (which is required for the transformation into a stationary stochastic process), and q is for the Moving Average MA (q) process.

Because it uses lagged moving averages to smooth time series data, the ARIMA model is useful for predicting. However, predicting turning points is tough, and long-term predictions are even more challenging.

2.3.1 DEEP LEARNING MODELS

This study, which uses deep learning techniques, aims to propose a promising solution for enhancing the proactive prediction of epidemic expansion. Long short-term memory is used in a deep learning strategy (LSTM). In the literature, deep learning algorithms showed significant performance increases in a variety of applications. This section is devoted to quickly outlining the basic concept of the deep learning model that will be used for Covid 19 time series forecasting in the future, particularly (LSTM).

2.3.2 LONG SHORT-TERM MEMORY (LSTM)

In the year 1977, Hochreiter and Schmidhuber created LSTM networks to overcome the problem of long-term reliance. Because significant occurrences in a time series may have unpredictable lags, LSTM networks are well suited to categorize, process, and generate predictions based on time series data. Although LSTMs look like a chain of repeating modules, each repeating module has its structure. LSTM networks are a sort of recurrent neural network that can learn order dependence and are used in sequence prediction tasks. Deep learning has gotten a lot of press since it outperforms traditional methods in a variety of fields. The LSTM network is a recurrent neural network with an LSTM layer in deep learning. The input gate, forget gate, memory cell, and output gate are comprised of the LSTM cell and used to learn a better model by utilizing knowledge from the past.

2.3.3 RECURRENT NEURAL NETWORK (RNN)

A recurrent neural network (RNN) is an artificial neural network having recurrent connections or coupled neurons that can cycle. These networks are useful in modeling problems such as automatic speech recognition, handwriting recognition, computer vision, and time series forecasting.

The output of any layer in an RNN is determined by the set of previously reported inputs as well as the current input.

$$h_{\mathrm{t}} = fw(h_{\mathrm{t-1}} \cdot x_{\mathrm{t}})$$

As presented in the above equation, this function gives it a significant, benefit over other neural networks by taking advantage of previously collected inputs and estimating outputs at a later level.

2.3.4 ARTIFICIAL NEURAL NETWORKS

Deep learning is a type of machine learning that is based on learning data representations rather than task-specific algorithms. Deep learning models employ a network of multi-layered nonlinear processing units known as neurons, which can automatically extract and transform features. Artificial Neural Networks (ANN) are a type of non-parametric information representation in which the output is a nonlinear function of the input variables. An ANN is an interconnected group of nodes that simulate the structure of neurons present in the human brain (Abhinav. T. 2018). These neurons are organized into layers, with the output of the current layer of neurons serving as the input to the subsequent layer.The interconnections between the layers of neurons in a feed-forward do not create a cycle. Each layer of a feed-forward neural network applies a function to the output of the preceding layer. The output layer applies activations to its inputs for final predictions, whereas the hidden layer turns its inputs into something that the output layer can use.

LSTMs can forecast future values based on recent sequential data, which gives demand forecasters more accuracy and leads to better business decisions. However, LSTMs take longer to train, require more memory, and dropout is considerably more difficult to apply**.**

2.4 EMPIRICAL LITERATURE REVIEW

Several recent studies have offered several time series methods for calculating Covid 19's distribution. Saba and Elsheikh, for example, developed rudimentary autoregressive neural networks for projecting the incidence of the Covid 19 outbreak in Egypt, which displayed rather high performance when compared to publicly reported instances. Yousaf *et al* used the auto-regressive integrated moving average (ARIMA) model to forecast Covid 19 in Pakistan. The model predicted that the number of confirmed cases would increase by a factor of 2.7 by the end of May 2020, yielding a 95 percent prediction interval of 5681 33079 cases. However, by the end of May 2020, Pakistan had reported almost 70 000 instances, suggesting that the model failed to predict accurately. To reduce the pandemic's suffering, a variety of studies are being done.

Other investigators also anticipated the Covid 19 outbreak in Saudi Arabia based on a total of 49176 infected patients and data collected between March 2 and May 15, 2020 (Alboaneen et al., 2020). The Susceptible-Infected-Recovered Model and the Logistic Growth Model were both used. The authors found that both models had certain limitations because the outcomes were varied. (Vaishya et al., 2020) investigated artificial intelligence applications for the Covid 19 pandemic, citing seven key applications, one of which was the prediction of cases and mortality. They pointed out that artificial intelligence (AI) can track and anticipate the virus characteristics, as well as to detect positive instances and death predictions, as well as determine which places, people, and countries are most vulnerable, and can take preventive measures.

Arko Barman (2020) investigated time series analysis and forecasting of COVID-19 instances using LSTM and ARIMA models. The results of different Long Short-Term Memory (LSTM) models and the Auto-Regressive Integrated Moving Average (ARIMA) model in projecting the number of confirmed cases in the United States, Italy, and Germany were investigated, and different Long Short-Term Memory (LSTM) models and the Auto-Regressive Integrated Moving Average (ARIMA) model were used to project the number of confirmed cases in the United States, Italy, and Germany.

Using time series of daily cumulative cases of Covid 19, several LSTM models and ARIMA were utilized to construct 1-day, 3-day, and 5-day forecasts. Two new k-period performance metrics, k-day mean absolute percentage error (kMAPE) and k-day Media Symmetric Accuracy (kMDSA), were developed to evaluate the models' ability in projecting time series values for multiple days. Prediction errors for LSTM models were both as high as 0,05 percent, while those for ARIMA were 0,07 percent and 0,06 percent, respectively, according to the data. The figures are slightly underestimated by LSTM models, whereas they are slightly overestimated by ARIMA models.

Forecasting Covid 19 confirmed cases, deaths, and recoveries," Abdallah M. Sammy (2021) wrote in his journal, the Holt model, Autoregressive Integrated Moving Average (ARIMA) model, Trigonometric Exponential smoothing state-space model, and cubic smoothing spline model were all considered by this author.

In the model, the day was the unit of time. Mean absolute error and mean absolute percentage error was used to evaluate the predicting performance of all of these models. All of the models were forecasted using the "R" program, and forecast durations were checked weekly to measure model performance as the data set grew larger. The research revealed that all models came to similar findings, with minor changes between them. If only one model is to be used, the ARIMA model was the most dependable and might be preferred over the others. When compared to the Holt and TBATS models, ARIMA and cubic smoothing spline models displayed lower prediction errors and narrower PIs. The TBATS model had much wider PI widths, especially when there was a small amount of data, which is undesirable in applications with constant data.

The Holt model, on the other hand, always had a substile greater AIC, indicating that the model fit was poor. All four models' forecasting accuracy metrics can account for significant changes in the market.

Luo (2021) researched "Time series prediction of Covid 19 transmission in America using LSTM and XGBoost algorithms". The author used the XGBoost model to perform a sensitivity analysis to determine the predictive model's robustness to parameter features. The findings revealed that lowering the contact rate between infected people can effectively reduce the number of daily confirmed cases. The models used in the study were based on data up to September 30, 2020. The LSTM model has a lower metrics value than XGBoost, according to the data. The author noted that the presented models have some limitations. For example, the sample dimensions are rather small and need to be increased if the model is to be generalized. Furthermore, several smoothing models can be used to obtain a better fitting curve and a more accurate forecast.

In an article,

S Singh (2020) established the efficiency of ARIMA models as an early warning method capable of generating accurate Covid 19 forecasts despite minimal data points. Data for the study were collected from 22 March 2020 to 31 January 2022, and the time-series database and prediction models were created using the Statistical Package for the Sciences (SPSS) version 24.0. The results demonstrated that ARIMA models are not only effective but also a simple and easy method for predicting Covid 19 trends based on open access data. Furthermore, the use of smoothed data and independent covariates improved model accuracy.

Various earlier researches using typical time series forecasting models to forecast future Covid 19 cases around the world also have been investigated.

2.5 ESTABLISHING SPACE IN KNOWLEDGE AND HOW THE STUDY WILL TRY AND BRIDGE THE GAPS

Most researchers looked at the Covid 19 instances time series analysis to predict and forecast its spread and to discover an appropriate time series to explain the changes in the trend. However, the current study aims to assist the government in projecting the dynamics and impact of Covid 19 cases as a backbone for maximizing the use of available resources in hospitals and planning for the future, as well as determining whether Zimbabwe would be hit by Covid again or not.

2.6 SUMMARY

A detailed theoretical and empirical literature review has been laid on previous studies written under the applicability of time series models in forecasting Covid 19. The methodology of the study , as well as the data collection and analysis methodologies, will be discussed in the following chapter.

CHAPTER THREE

3.1 INTRODUCTION

This chapter outlines the methods used to conduct the research. It includes the research type, research design, information about the population, sampling and data analysis, and the model building procedures.

3.2 RESEARCH TYPE

The research is a quantitative one. Secondary data was collected, analyzed, and used to develop an ARIMA and a Deep Learning Model to forecast the spread of Covid 19.

3.3 RESEARCH DESIGN

(Zach Claybaugh, 2020) The term "research design" refers to the overall technique for conducting research that includes data gathering, interpretation, analysis, and discussion consistently and logically. A mix of exploratory research designer and action research design was implemented. An exploratory research design was chosen because they are few studies in this area of forecasting Covid 19 using Deep Learning. This research design provides flexibility for the definition of new terms and clarification of existing concepts and theories. Action research design focuses on practical, solution-oriented research rather than verifying theories.

3.4 RESEARCH INSTRUMENT

Research instruments refer to any tools that were used to collect and analyze data. Python 3 programming language was used to scrape data from the Ministry of Health website. Web scraping is a process of collecting and pre-processing data that is hosted on the World Wide Web. Google Collaboratory, which is an open-source platform created by Google Inc., to perform Data Science using python programming language, was the code editor used to analyze data and build the models. The idea of the Google Collaboratory Notebook was taken from Jupyter Notebook which is distributed by Anaconda.

3.5 DATA COLLECTION

The data used by the researcher was collected from the Ministry of Health's official website. The Ministry of health records the number of new confirmed Covid 19 cases every day. The first case was recorded on 20 March 2020. A process called web scraping in python 3 was used to retrieve information from the website. Web scraping is an automated retrieval of information from the web using a programming language. The data collected comprised cumulative

Zimbabwe Covid 19 cases from March 2020 to 31 January 2022. The data set was constructed with consecutive dates and respective Covid 19 case counts, and it was ready to be used for time-series forecasting. The information used is classified as secondary data. Secondary data is information that has been gathered and compiled by other entities and is ready to be analyzed.

3.6 DATA CLEANING

Data will not always be in the form that is best suitable for model development. Data cleaning aims to make messy data tidy. According to (Wickham, 2014) a tidy dataset is described by three things which are:

- A variable from the column
- Each observation forms a row
- Each observational unit forms a table

The daily data were combined to be in a tabular format which contains all the data from 20 March 2020 to 31 January 2022. The data was pre-processed to be in a tidy format where each observation forms a row and each variable forms a column. There are two variables in the dataset which are the cumulative cases and the date. Data were checked for completeness and also for the presence of duplicates.

3.7 DATA ANALYSIS

The data were analyzed to get an as in-depth understanding of the data. The cumulative Covid cases data were differenced to get daily Covid cases. Descriptive statistics were calculated on the daily cases. These are the mean, maximum and minimum number of daily cases. The data were also checked for outliers using the interquartile range proximity rule and boxplots. The distribution of the data was found by plotting histograms and kernel density estimations. Time series data is made up of 3 components which are the trend, seasonality, and the residuals. The data was decomposed into its principal components mentioned above. The trend in the movement of the Covid cases was observed from the results of decomposing the time series. Also, the line plot of the cumulative cases, as well as the daily cases, helped to determine the trend in the data.

3.8 TESTS FOR STATIONARITY

Time series models require data to be stationary before it is fit to the model. Stationary data has a constant mean, constant variance, and covariance which only depend on the lag *k*. Several statistical tests were done on the data to determine if it was stationary or not. The tests were:

- Kwiatkowski–Phillips–Schmidt–Shin (KPSS) tests

- augmented Dickey-Fuller test (ADF)
- Phillips–Perron test

3.9 PLOTTING THE ACF AND THE PACF

The ACF and PACF analyses shows how time series data observations e connected t order to figure otomodel for a particular time-series data. The charts aid in determining the AR and MA phrases' order.

3.10 MODEL IDENTIFICATION

Two models were developed and these are the ARIMA and LSTM models.

3.10.1 ARIMA MODEL

A module in python called pmdarima was used to build the ARIMA model. The module has a function called Auto ARIMA. It iteratively tries different combinations of the ARIMA parameters until it finds the best set of parameters. In the order of importance ARIMA, considers three primary parameters denoted as (p, d, q), which include the autoregressive lag order (p), the degree of differencing (d), and the moving average window size (q). The autoregressive lag order (p) represents the number of previous values to be used and the moving average window size (q) represents the data points that are used to compute a weighted average. The best set of parameters is the ones that give the least mean squared error among the pool of parameters tried. The researcher gives the function the start and the stop values for each of the ARIMA parameters. The start and the stop values are determined from the exploratory data analysis, testing for stationarity, and the plotting of the ACF and PACF that was done.

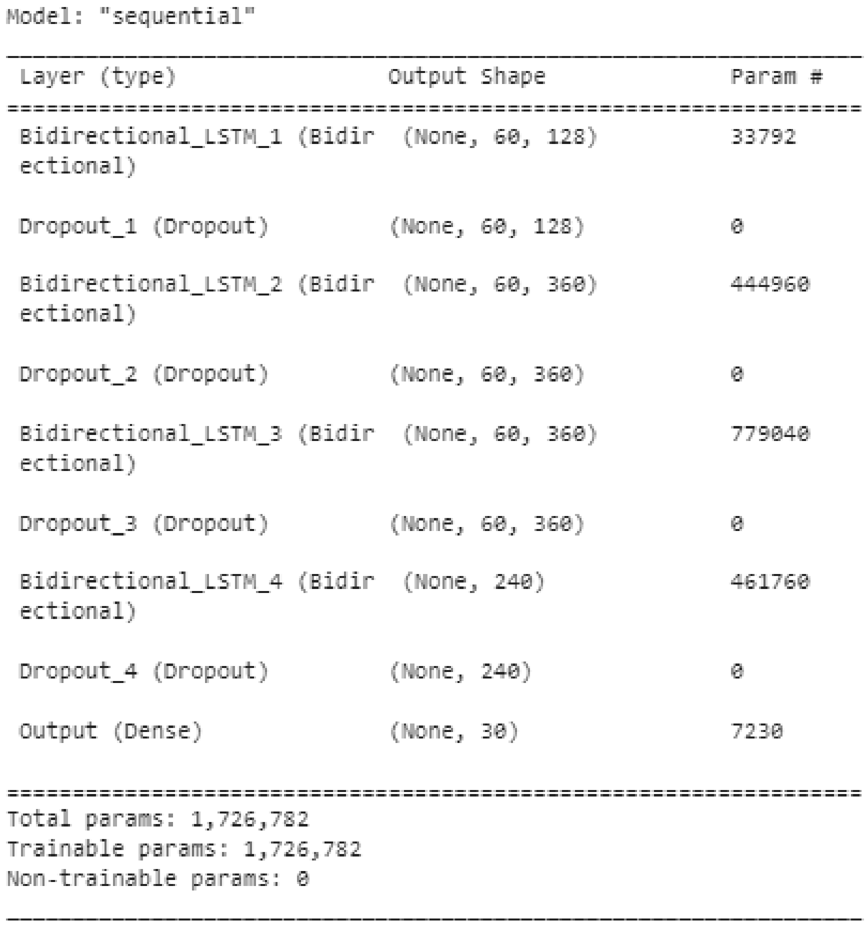
3.10.2 LSTM

LSTM is short for Long Term Short Memory. The input data which is differenced to become daily cases are structured in a form required by the LSTM model that is *n* previous values being used to predict the next *m* values where *n* and *m* are integers which are obtained from a process called hyper parameter tuning. The data was normalized. Normalization is transforming data to be on a scale of 0 to 1. The formula for normalization used is shown be$x_{norm} = \frac{x - \min(X)}{\max(X) - \min(X)}$

Where:

- X is the whole dataset
- x is an individual sample

- x_norm is the individual sample after normalization



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Bidirectional_LSTM_1 (Bidir  (None, 60, 128)          33792
 ectional)

 Dropout_1 (Dropout)         (None, 60, 128)           0

 Bidirectional_LSTM_2 (Bidir  (None, 60, 360)          444960
 ectional)

 Dropout_2 (Dropout)         (None, 60, 360)           0

 Bidirectional_LSTM_3 (Bidir  (None, 60, 360)          779040
 ectional)

 Dropout_3 (Dropout)         (None, 60, 360)           0

 Bidirectional_LSTM_4 (Bidir  (None, 240)              461760
 ectional)

 Dropout_4 (Dropout)         (None, 240)               0

 Output (Dense)              (None, 30)                7230

=================================================================
Total params: 1,726,782
Trainable params: 1,726,782
Non-trainable params: 0
_________________________________________________________________
```

Figure 1 LSTM Architecture Developed

The architecture of the LSTM model shown above is developed through an iterative process of experimentation to find the best architecture in terms of the least mean squared error. 4 Bidirectional LSTM layers were stacked together. In between them, Dropout Layers will be put to provide regularization to avoid overfitting. Overfitting refers to a situation where the model performs well on training data but performs poorly on test data. The model would have failed to generalize the patterns in the training data.

The model takes 60 days of previous daily cases and uses them to predict the next 30 days' data. The output layer has a shape of (None, 30) so that when given input it predicts the next 30 days. The first layer has a shape of (None, 60, 128) so that it takes 60 days of data as input. 64 LSTM units which are bidirectional to make them 128.

An LSTM model is a special form of a Recurrent Neural Network Model (RNN). It is a class of Deep Learning models. It is used to predict sequential data. Time series data falls into this category of sequential data. It was designed to solve the shortcomings of the RNN model. The RNN model has an architecture shown in the image below:

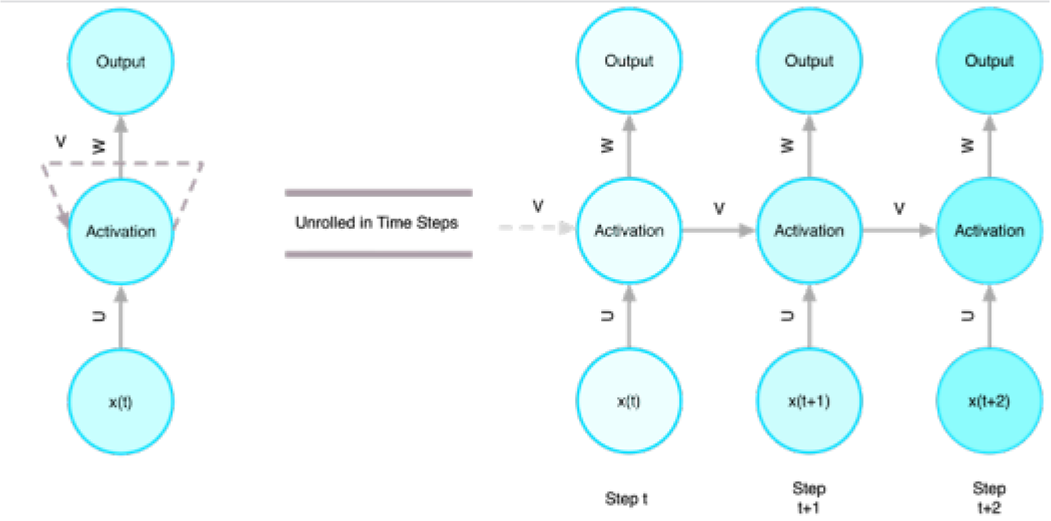


Figure 3.2 RNN Unit Inner Workings

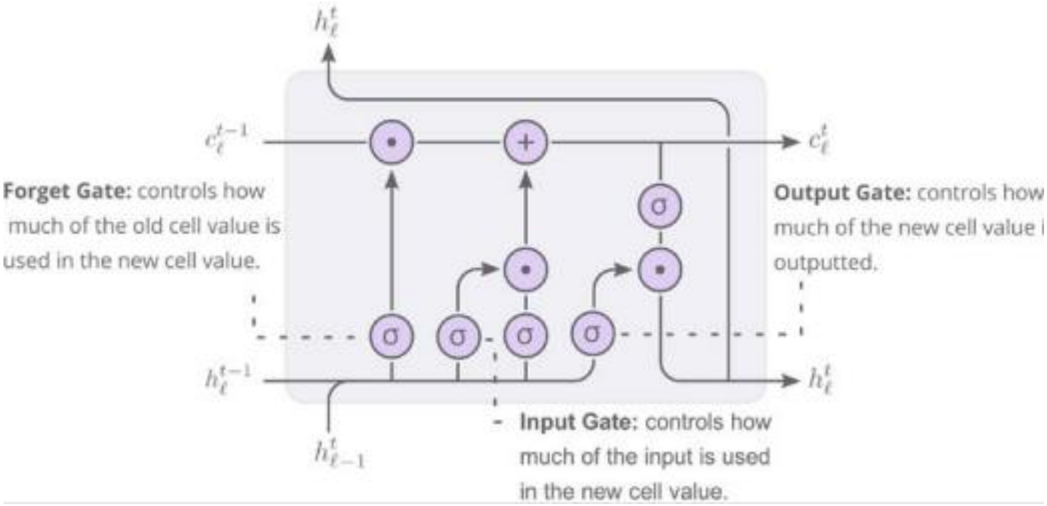
The input at time *t* is multiplied by a weight *u* and the result is passed into the activation on the center. The input from the previous time stamp is multiplied by the weight v and the result is also passed to the activation. The activation is then multiplied by the weight *w* to get output at time *t*. At first, the weights *u*, *v*, and *w*, are initialized by random numbers from the normal distribution. Training an LSTM model is finding the optimal values for the 3 weights which result in the lowest mean squared error of the model. This network can be unrolled in terms of sequence or time steps. This unfolding is all imaginary. It is, however, depicted on the right side of the figure. Activation at time step t can be expressed mathematically as:

$$a_t \ = U.x_t \ + V.x_t$$

Output at the same step can be presented by:

$$O_t = W.a_t$$

The RNN suffers from vanishing or exploding gradients and the inability to manage long-term dependencies. As a result, some modifications were done to come up with the LSTM which can solve these problems. The architecture of the LSTM is shown below:



h_ℓ^t
c_ℓ^{t-1}
c_ℓ^t
h_ℓ^{t-1}
h_ℓ^t
$h_{\ell-1}^t$
Forget Gate: controls how much of the old cell value is used in the new cell value.
Output Gate: controls how much of the new cell value i outputted.
Input Gate: controls how much of the input is used in the new cell value.

Figure3.3: LSTM Unit Inner Workings

The single unit of an LSTM model called a cell has 4 main parts which are explained below:

- Cell Value which is the current value or memory of the cell
- Forget the gate which controls how much of the old cell value is used in the new cell value
- Input gate which controls how much of the input is used in the new cell value
- Output gate which controls how much of the new cell value is outputted

These 4 main parts make the LSTM model robust to the problems presented by the RNN which is why it was chosen.

3.11 MODEL TRAINING

The data was split into two that is train set and the test set. The train set comprised data from March 2020 to 1 January 2022. The test set comprised data from 2 January 2022 to 31 January 2022. For the ARIMA Model, data were fitted to the model that was obtained from the model identification process. The resulting model was evaluated on the test set and used to predict the movement of cases.

The LSTM model was trained for 1000 epochs. An epoch is a unit of time used to train a neural network with all the training data for a single cycle. Two call-backs were used. A call back controls how the model trains. Early stopping call back was used to stop the model if it did not improve for 20 epochs. An improvement in the model means a reduction in the model loss function. In this case, the mean absolute percentage error was used as the loss function. It was chosen because it penalizes higher errors more as compared to other loss functions like the mean absolute error. However, it doesn’t have a natural interpretation like the mean absolute error. The mean absolute show how on average the model predictions deviate from the true value. Model Checkpoint call back was used to save and update the best model after each epoch.

3.12 COMPONENTS OF TIME SERIES

One should consider the types of data patterns obtained from time plots in t order for them to select and choose the most appropriate forecasting method. The main types of time series patterns are trend (T), cyclical (C), seasonal (S), horizontal (H), and irregular (I).

TREND (T)

According to Methodological information (2005), a trend is the component of a time series that represents variations of low frequency in a time series, the high and medium frequency fluctuations having been filtered out. The trend is normally referred to as the long-term movement in a cyclical context.

CYCLICAL(C)

(Makridakis, Wheelwright, & Hyndman, 1998), noted that a cyclical pattern exists when the data exhibit rises and cyclical falls that are not of a fixed period. A cycle occurs when the data exhibits rise and falls that are not of a fixed frequency. These fluctuations are usually due to economic conditions, and are often related to the "business cycle". The duration of these fluctuations is usually at least 2 years.

SEASONAL (S)

A seasonal pattern occurs when a time series is affected by seasonal factors such as the time of the year or the day of the week. Seasonality is always of a fixed and known frequency.

HORIZONTAL (H)

Makridakis, Wheelwright, & Hyndman, 1998 explained that this type of pattern exists when the data values fluctuate horizontally around a constant mean and such a series is called “stationarity” in its mean.

IRREGULAR (I)

The irregular component is an unpredictable component that remains after the seasonal and trend components of a time series have been estimated and removed

3.12 MODEL VALIDITY

After training the models, the researcher evaluated the models to check for validity and to compare the models to determine the one which is the best suitable prediction of covid cases movement. The following metrics were used in the evaluation of the models:

- $MAE = \frac{\sum|actual-predicted|}{n}$
- $MSE = \frac{\sum(actual-predicted)^2}{n}$
- $MAPE = \frac{1}{n}\sum|\frac{actual-predicted}{actual}|$

Where:

- MAE = Mean Absolute Error
- MSE = Mean Squared Error
- MAPE = Mean Absolute Percentage Error

The model was deemed valid if they scored a mean absolute percentage error of less than 1. A better model is the one with the least mean absolute percentage error between the two models.

3.13SUMMARY

The methodology for conducting the research was written out in response to comments given by researchers in Chapter 2. The study's methodology and data sources were also highlighted. The chapter moved on to discuss data analysis processes, which will be covered in the following chapter. The data representation and analysis will be discussed in the following chapter. The discussion then moves on to the ultimate conclusions and recommendations based on the findings.

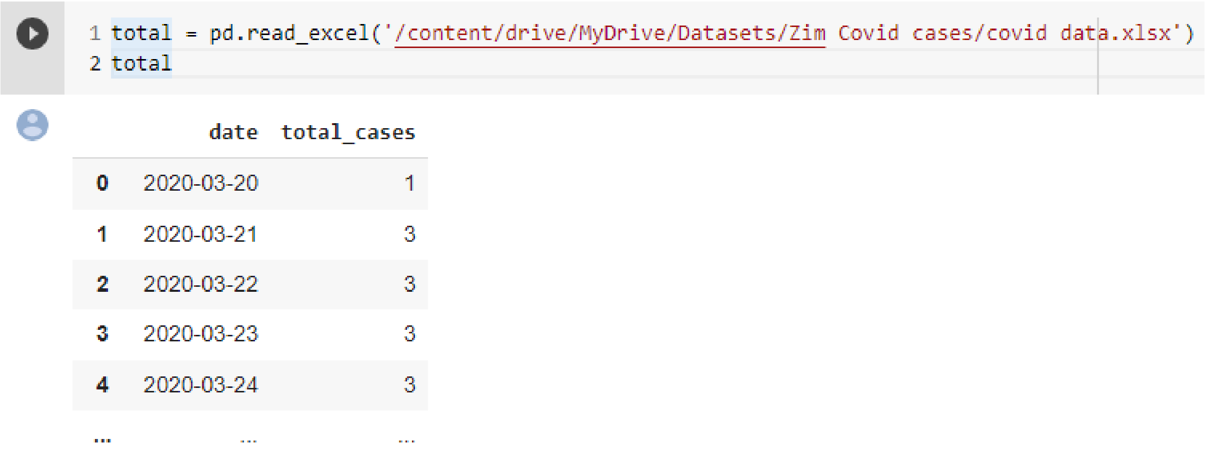
CHAPTER FOUR

4.1. INTRODUCTION

This chapter presents the results of the data analysis. It outlines the steps followed by the researcher in analyzing data. The chapter also presents the results of the ARIMA and LSTM model built by the researcher. The pictures inserted in this section are screenshots from Google Collaboratory Notebook. The whole code for performing data analysis and building the model is found in the appendix section.

4.2. UPLOADING THE DATA ONTO PYTHON

After secondary data was obtained from the Ministry of Health website, it was uploaded into google drive. Google drive works as a backend data repository for Google Collaboratory. The data was a comma-separated values file. Code in Appendix A

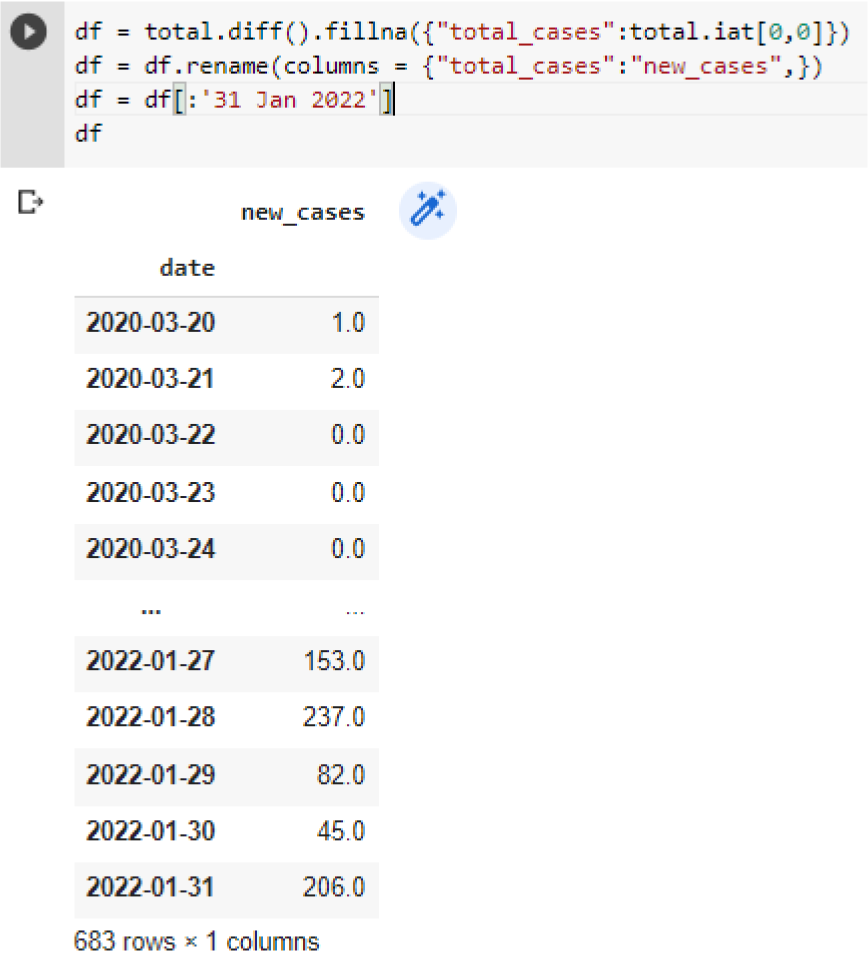


```
total = pd.read_excel('/content/drive/MyDrive/Datasets/Zim Covid cases/covid data.xlsx')
total
```

```
            date  total_cases
0     2020-03-20            1
1     2020-03-21            3
2     2020-03-22            3
3     2020-03-23            3
4     2020-03-24            3
...          ...          ...
```

Figure 4.1: Reading Data

4.3. DIFFERENCING THE DATA TO GET DAILY CASES

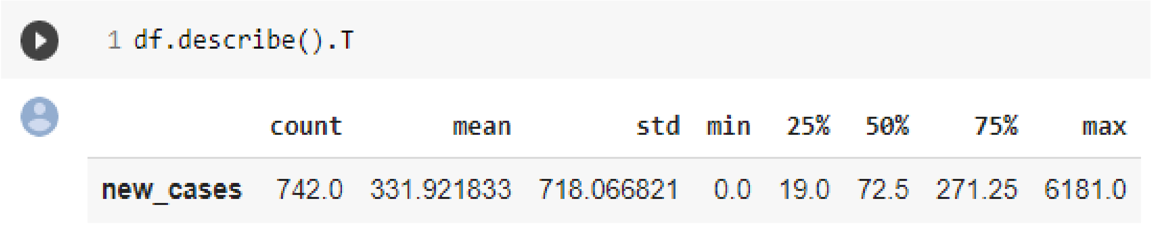


```
df = total.diff().fillna({"total_cases":total.iat[0,0]})
df = df.rename(columns = {"total_cases":"new_cases",})
df = df[:'31 Jan 2022']
df
```

```
            new_cases
date
2020-03-20        1.0
2020-03-21        2.0
2020-03-22        0.0
2020-03-23        0.0
2020-03-24        0.0
...               ...
2022-01-27      153.0
2022-01-28      237.0
2022-01-29       82.0
2022-01-30       45.0
2022-01-31      206.0
683 rows × 1 columns
```

Figure 4.2: Converting Cumulative to Daily Cases

4.4. DESCRIPTIVE STATISTICS

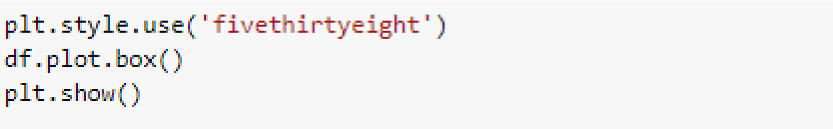


```
df.describe().T
```

```
           count  mean        std         min  25%   50%   75%     max
new_cases  742.0  331.921833  718.066821  0.0  19.0  72.5  271.25  6181.0
```

Figure 4.3: Descriptive Statistics

The table above shows the summary statistics for the daily cases. The results indicate the presence of outliers. There is a huge difference between the upper quartile and the maximum value. A box plot was drawn to further explore this. It showed that there were a lot of outliers.



```
plt.style.use('fivethirtyeight')
df.plot.box()
plt.show()
```

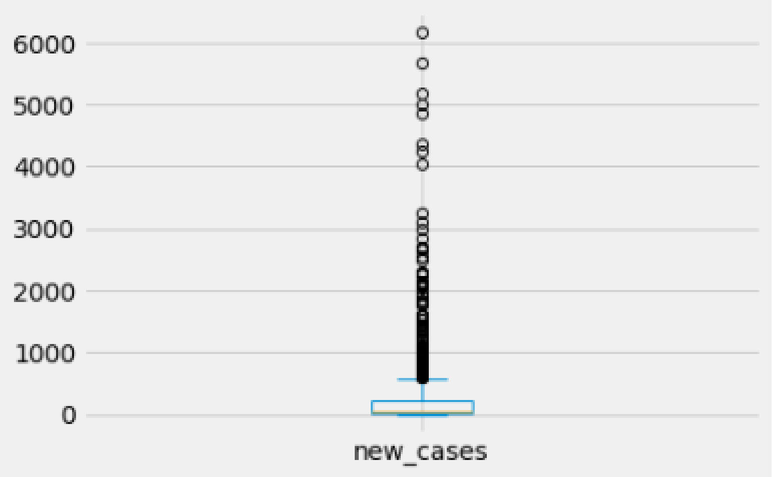
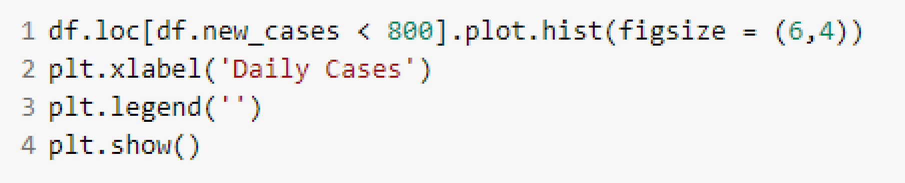


Figure 4.4: Daily Cases Box Plot

A histogram was also plotted to check the distribution of the data. Values greater than 800 were removed so that the histogram shows the distribution of where the majority of the data is.



```
df.loc[df.new_cases < 800].plot.hist(figsize = (6,4))
plt.xlabel('Daily Cases')
plt.legend('')
plt.show()
```

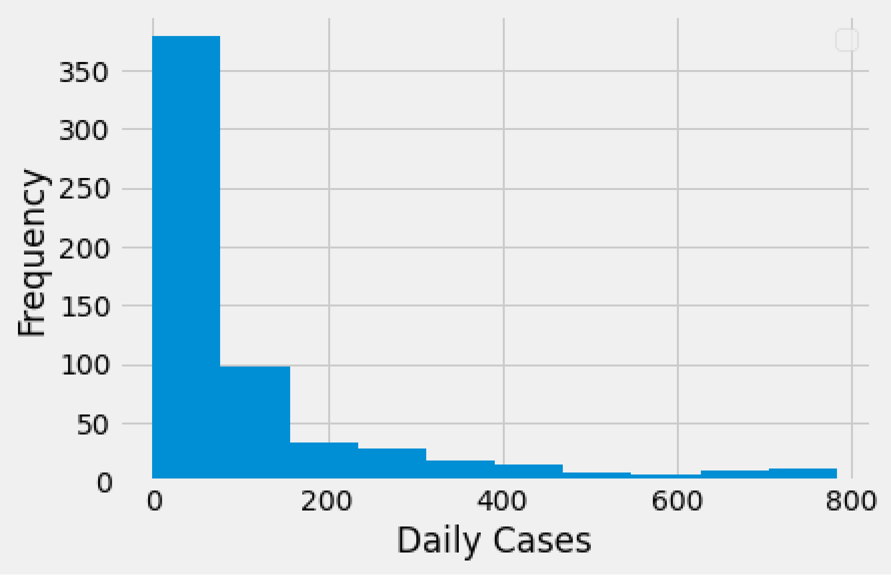
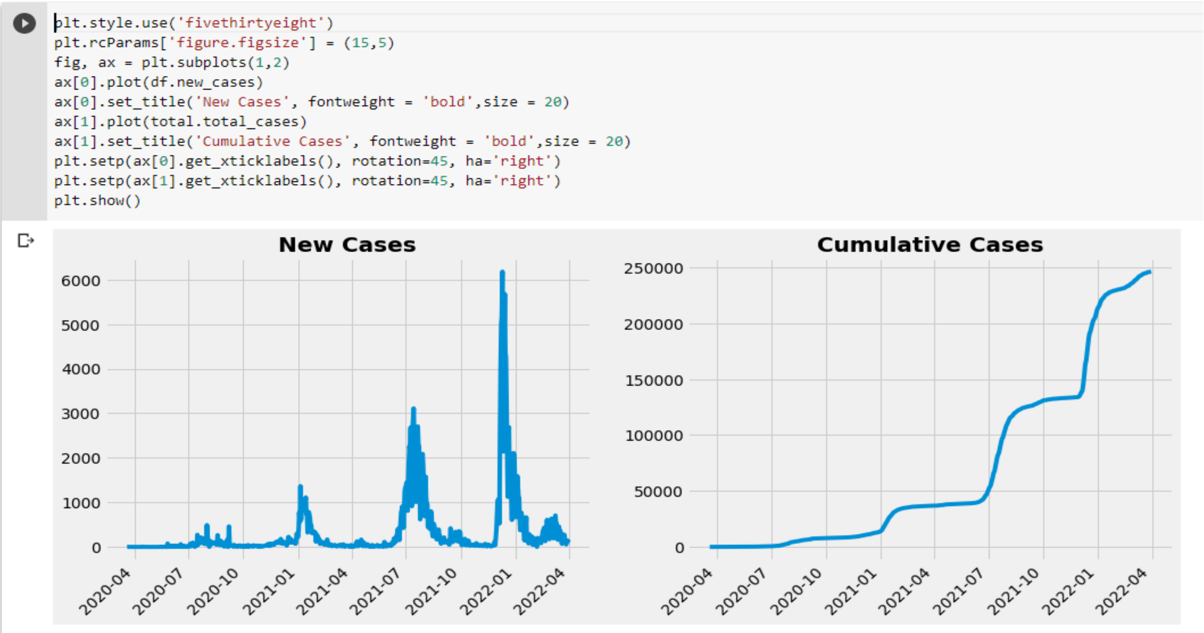


Figure 4.5: Daily Cases Truncated Histogram

The plot below shows the daily new cases and cumulative cases of Covid 19 from March 2020 to January 2022.

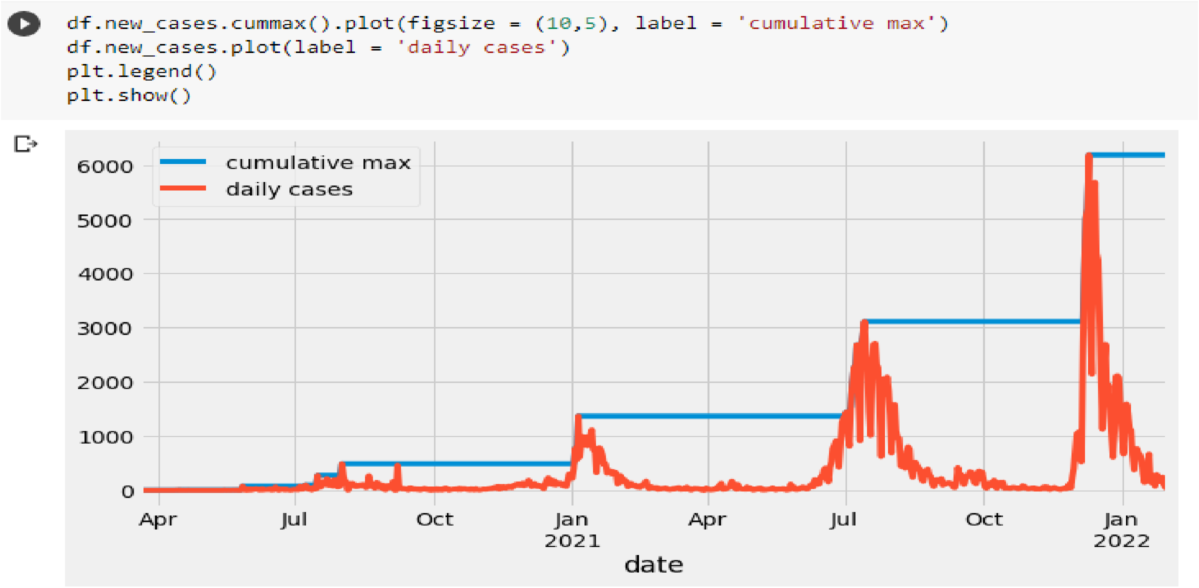


```
plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = (15,5)
fig, ax = plt.subplots(1,2)
ax[0].plot(df.new_cases)
ax[0].set_title('New Cases', fontweight = 'bold',size = 20)
ax[1].plot(total.total_cases)
ax[1].set_title('Cumulative Cases', fontweight = 'bold',size = 20)
plt.setp(ax[0].get_xticklabels(), rotation=45, ha='right')
plt.setp(ax[1].get_xticklabels(), rotation=45, ha='right')
plt.show()
```

Figure 4.6: Daily and Cumulative Cases Line Plot

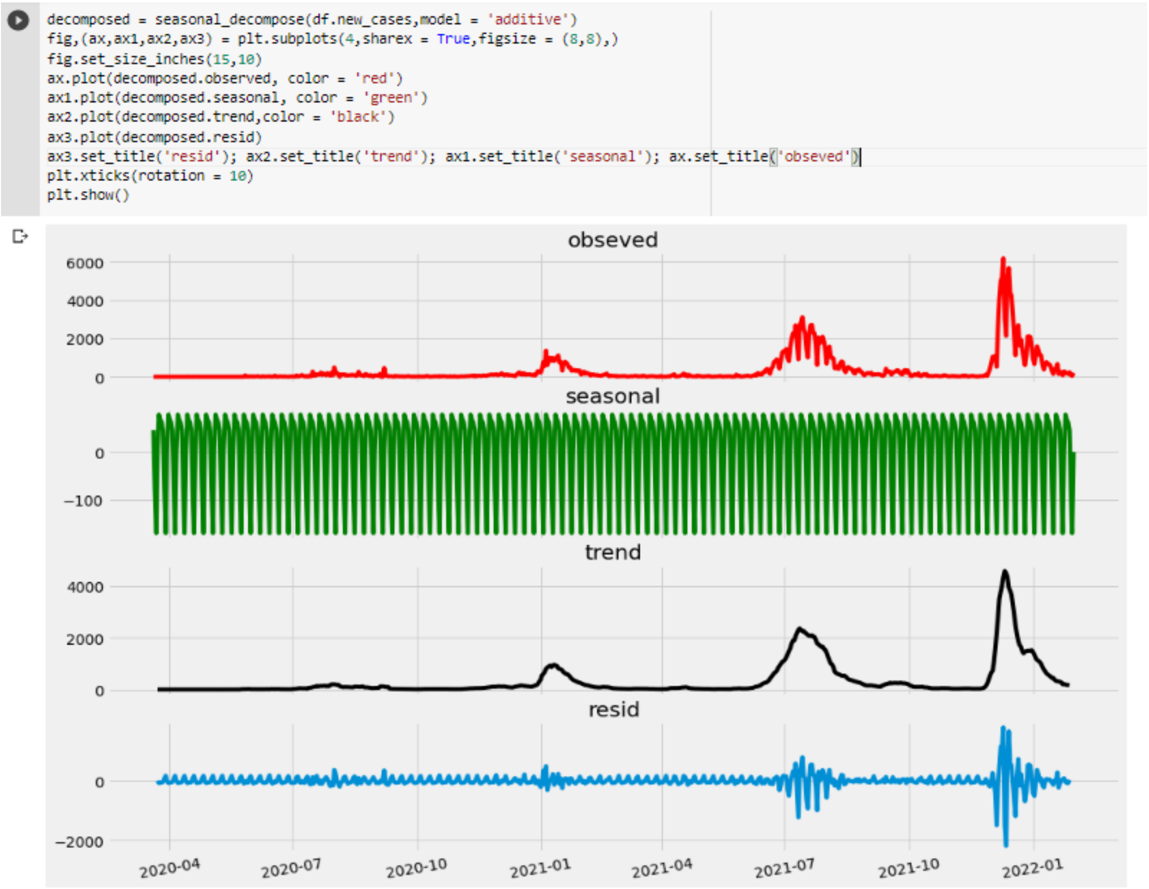
There was a huge jump in cases in January 2021, July 2021, and November 2021. These are the outliers detected by the summary statistics as well as the box plot.

Time series data is composed of 3 components which are the trend, the seasonality, and the residuals. The dashboard in figure 4.5 shows the individual components of the daily Covid 19 cases. The trend component helps to determine the trend in the movement of Covid daily cases. The line plots shown earlier were also used to determine the trend. As can be seen in the figure below, the daily cases are fairly constant at low values with some cyclical spikes in July 2020, January 2021, June 2021, and November 2021



```
df.new_cases.cummax().plot(figsize = (10,5), label = 'cumulative max')
df.new_cases.plot(label = 'daily cases')
plt.legend()
plt.show()
```

Figure 4.7: Identification of Spikes in Daily Cases



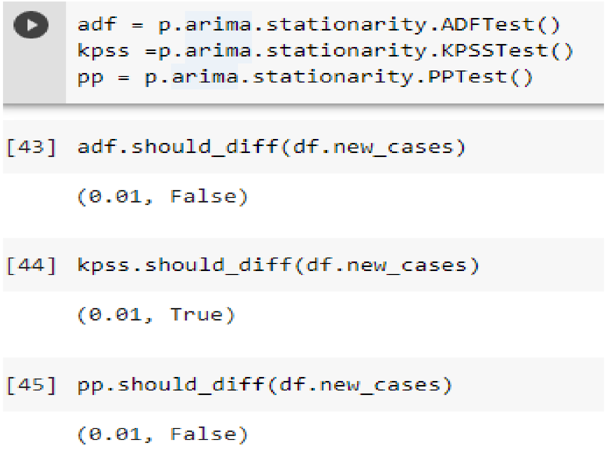
```
decomposed = seasonal_decompose(df.new_cases,model = 'additive')
fig,(ax,ax1,ax2,ax3) = plt.subplots(4,sharex = True,figsize = (8,8),)
fig.set_size_inches(15,10)
ax.plot(decomposed.observed, color = 'red')
ax1.plot(decomposed.seasonal, color = 'green')
ax2.plot(decomposed.trend,color = 'black')
ax3.plot(decomposed.resid)
ax3.set_title('resid'); ax2.set_title('trend'); ax1.set_title('seasonal'); ax.set_title('obseved')
plt.xticks(rotation = 10)
plt.show()
```

Figure 4.8: Time Series Components

Before the data is fitted into either the LSTM or the ARIMA models, normalization was done to make it easier to train.

4.5. TEST FOR STATIONARITY

The daily case data was tested for stationarity using methods described in chapter 3. The results were as follows:



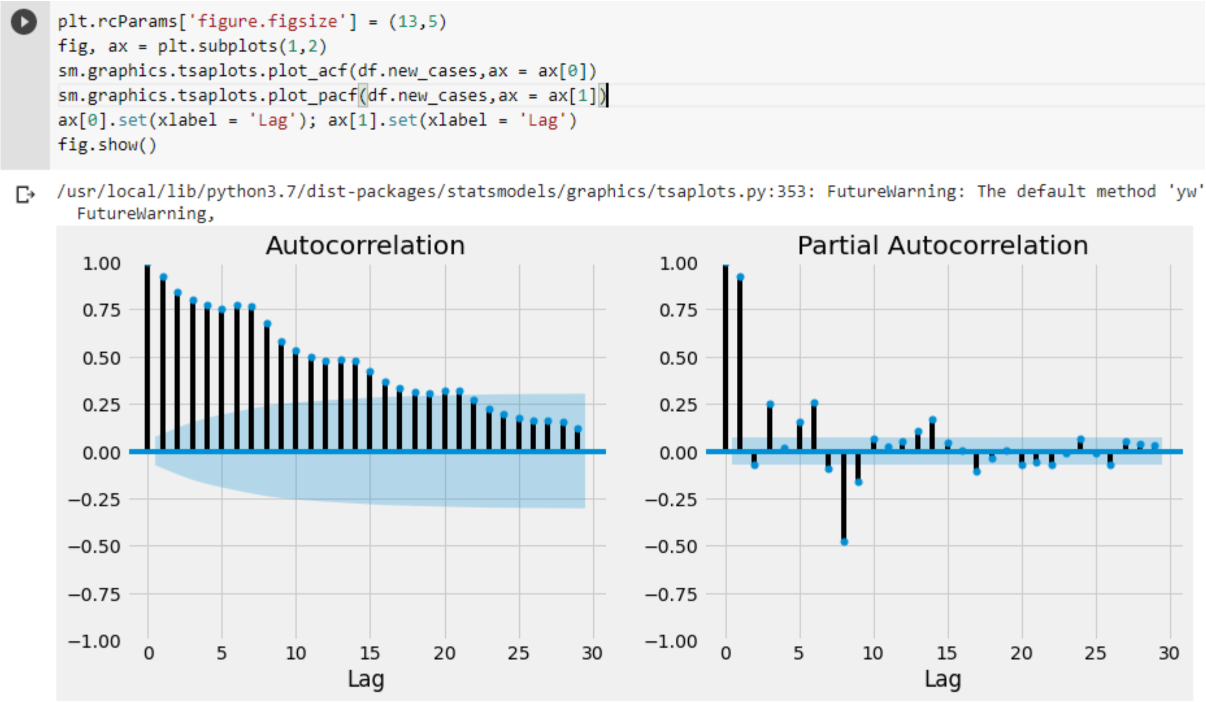
```
adf  = p.arima.stationarity.ADFTest()
kpss =p.arima.stationarity.KPSSTest()
pp  = p.arima.stationarity.PPTest()

[43] adf.should_diff(df.new_cases)

     (0.01, False)

[44] kpss.should_diff(df.new_cases)

     (0.01, True)

[45] pp.should_diff(df.new_cases)

     (0.01, False)
```

Figure 4.9: Testing for Stationarity

Method	**Results**
Dickey-Fuller test (ADF)	Stationary
Kwiatkowski–Phillips–Schmidt–Shin (KPSS)	Not Stationary
Phillips–Perron test	Stationary

4.6. ACF AND PACF

The ACF and PACF were plotted to further understand the results of the stationarity tests and help choose the starting points and end points for the search space of the ARIMA parameters.



```
plt.rcParams['figure.figsize'] = (13,5)
fig, ax = plt.subplots(1,2)
sm.graphics.tsaplots.plot_acf(df.new_cases,ax = ax[0])
sm.graphics.tsaplots.plot_pacf(df.new_cases,ax = ax[1])
ax[0].set(xlabel = 'Lag'); ax[1].set(xlabel = 'Lag')
fig.show()
```

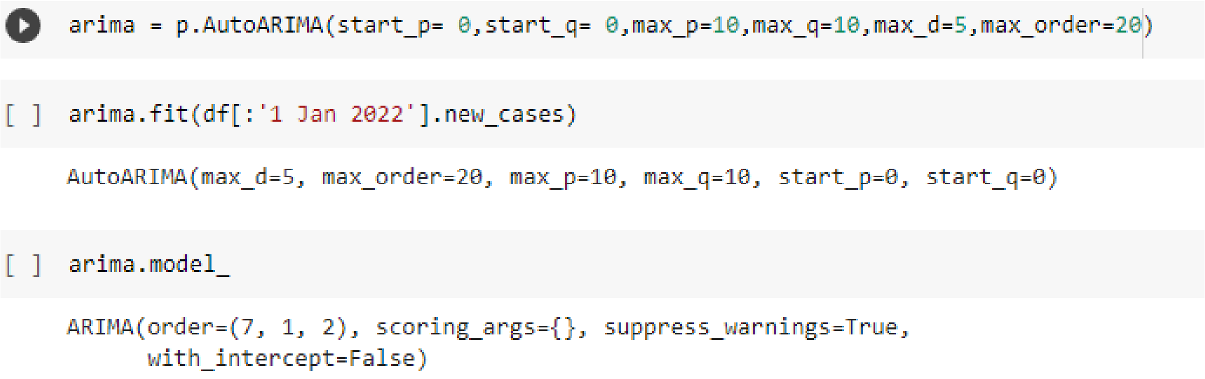
```
/usr/local/lib/python3.7/dist-packages/statsmodels/graphics/tsaplots.py:353: FutureWarning: The default method 'yw'
  FutureWarning,
```

Figure 4.10: ACF and PACF Plots

There is significant autocorrelation up to lag 30 which supports the results of the KPSS test for non-stationarity of the data. This implies a need for further differencing to make the data stationary. The PACF graphs are cut off which means an MA term should be included in the model

4.7. ARIMA MODEL RESULTS

The daily cases data from 20 March 2020 to 1 January 2022 was fitted to the Auto ARIMA function in the PmdArima module in Python. The start and maximum parameter values for *p*, *d*, and *q* were specified in the model definition. The fit function was used to train the model to obtain the best set of parameters. The result of training the Auto ARIMA was an ARIMA (7, 1, 2).



```
arima = p.AutoARIMA(start_p= 0,start_q= 0,max_p=10,max_q=10,max_d=5,max_order=20)
```

```
arima.fit(df[:'1 Jan 2022'].new_cases)
```

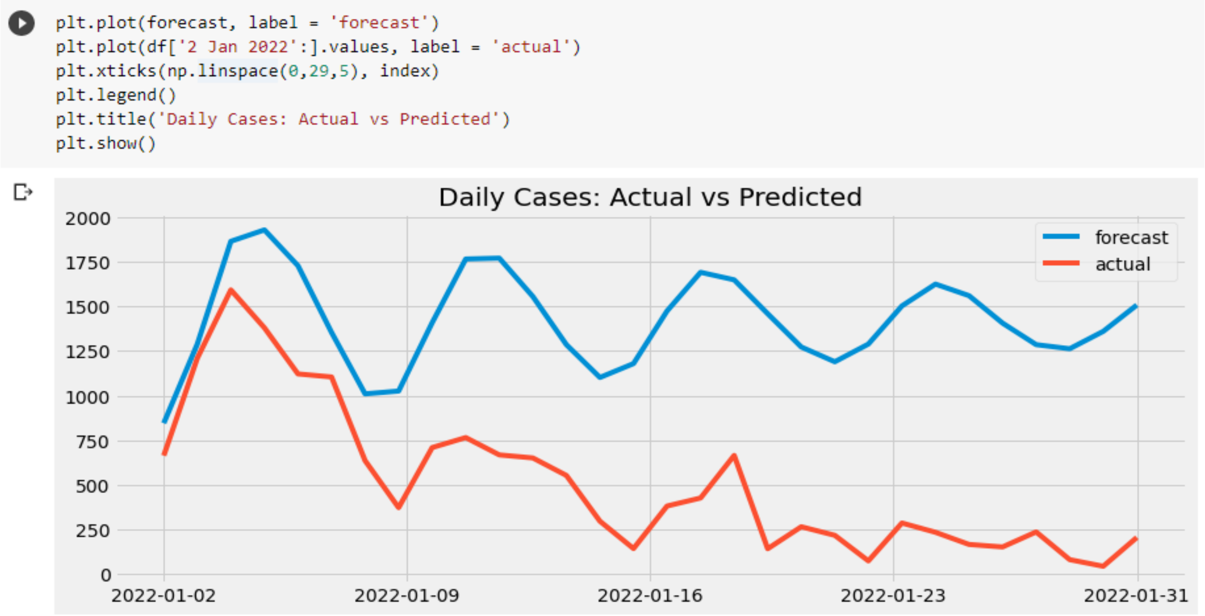
```
AutoARIMA(max_d=5, max_order=20, max_p=10, max_q=10, start_p=0, start_q=0)
```

```
arima.model_
```

```
ARIMA(order=(7, 1, 2), scoring_args={}, suppress_warnings=True,
      with_intercept=False)
```

Figure 4.11: Training an ARIMA Model

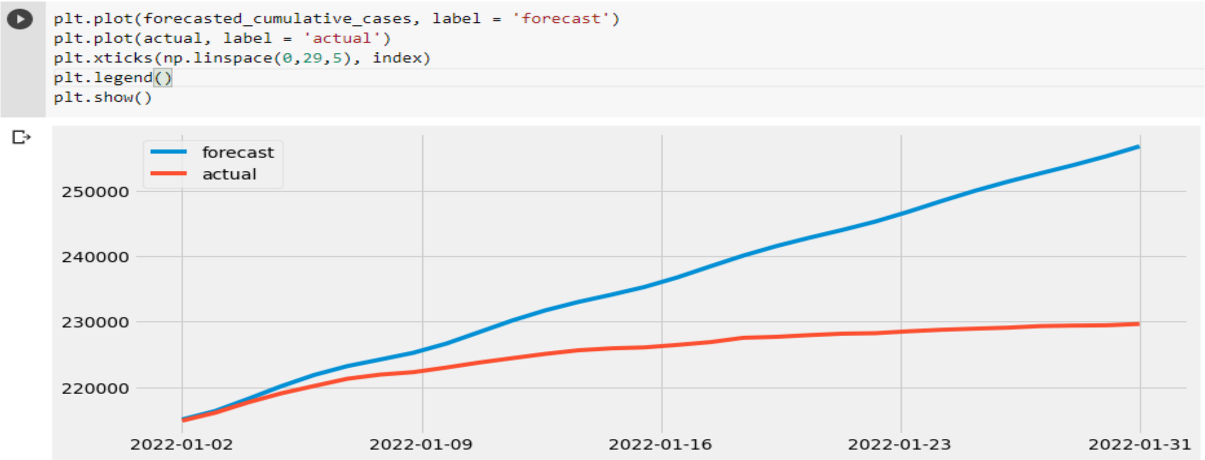
The Auto ARIMA model found the ARIMA (7, 1, 2) to be the best model for the Zimbabwe Covid data. The graph below shows the predicted vs the actual results on the test sets. These are daily movements. The test set was data from 2 January 2022 to 31 January 2022.



```
plt.plot(forecast, label = 'forecast')
plt.plot(df['2 Jan 2022':].values, label = 'actual')
plt.xticks(np.linspace(0,29,5), index)
plt.legend()
plt.title('Daily Cases: Actual vs Predicted')
plt.show()
```

Figure 4.12: ARIMA Actual vs Predicted – Daily Cases (ARIMA)

The graph below shows the Actual vs Predicted after the data was converted back to cumulative cases:

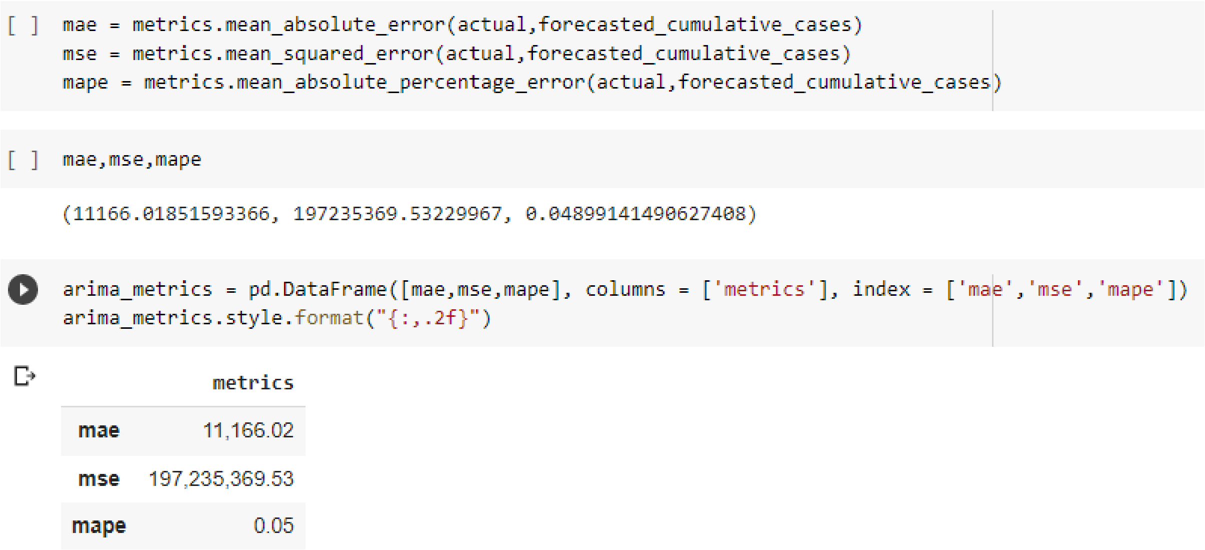


```
plt.plot(forecasted_cumulative_cases, label = 'forecast')
plt.plot(actual, label = 'actual')
plt.xticks(np.linspace(0,29,5), index)
plt.legend()
plt.show()
```

Figure 4.13: Actual vs Predicted – Cumulative Cases (ARIMA)

The graphs show a significant disparity between the actual and the predicted. This signifies a poor fit. To further understand how well the model performs, some evaluation metrics were calculated. These metrics are mean absolute error (MAE), mean squared error (MSE), and mean absolute percentage error (MAPE).

The table below shows the metrics for the ARIMA model



```
[ ]  mae = metrics.mean_absolute_error(actual,forecasted_cumulative_cases)
     mse = metrics.mean_squared_error(actual,forecasted_cumulative_cases)
     mape = metrics.mean_absolute_percentage_error(actual,forecasted_cumulative_cases)
```

```
[ ]  mae,mse,mape

     (11166.01851593366, 197235369.53229967, 0.04899141490627408)
```

```
arima_metrics = pd.DataFrame([mae,mse,mape], columns = ['metrics'], index = ['mae','mse','mape'])
arima_metrics.style.format("{:,.2f}")
```

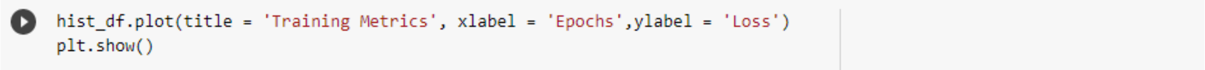
	metrics
mae	11,166.02
mse	197,235,369.53
mape	0.05

Figure 4.14: ARIMA Metrics

Mean absolute percentage error shows how wrong the model is on a percentage scale. The model is 5% wrong. On average, the model is wrong by 11166 cases when predicting the cumulative cases as shown by the MAE.

4.8. LSTM MODEL RESULTS.

The data were normalized and fitted to the LSTM model. The graph below shows the training metrics. There was a steep decrease in the mean squared error and mean absolute error for the first 50 epochs. Afterward, there was a steady decrease up to 300 epochs where the model was stopped by an Early Stopping Call back. The best model was saved using the Model Check Point Call Back. The Actual vs Predicted graph shows how well the model performed on data that it was not trained on. From the graph, we can conclude that model is a good fit.



```
hist_df.plot(title = 'Training Metrics', xlabel = 'Epochs',ylabel = 'Loss')
plt.show()
```

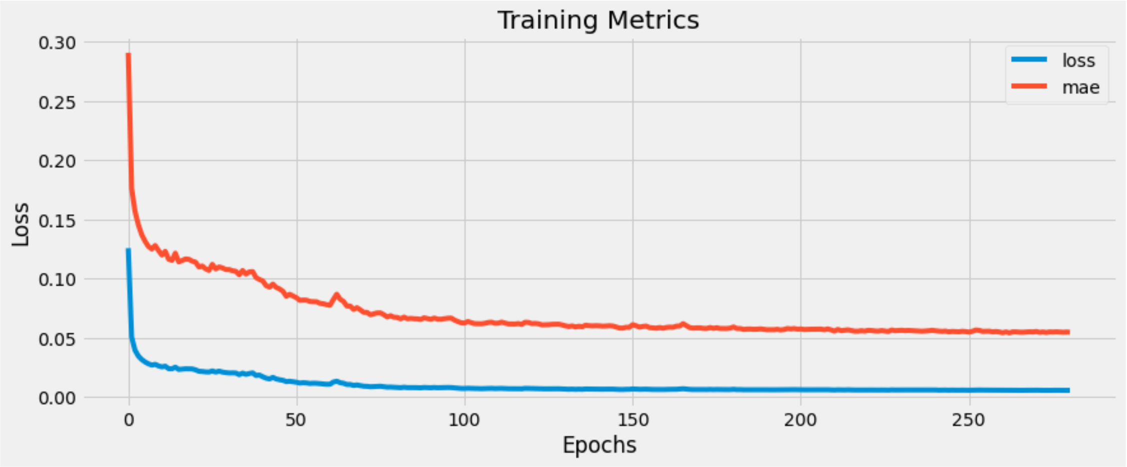
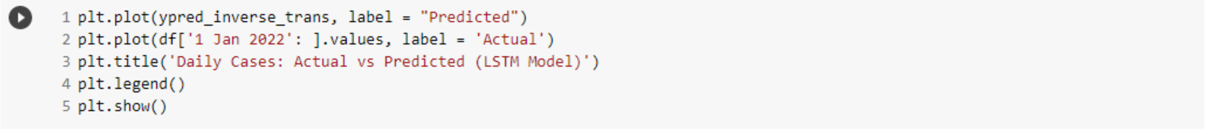


Figure 4.15: LSTM Training Metrics



```
plt.plot(ypred_inverse_trans, label = "Predicted")
plt.plot(df['1 Jan 2022': ].values, label = 'Actual')
plt.title('Daily Cases: Actual vs Predicted (LSTM Model)')
plt.legend()
plt.show()
```

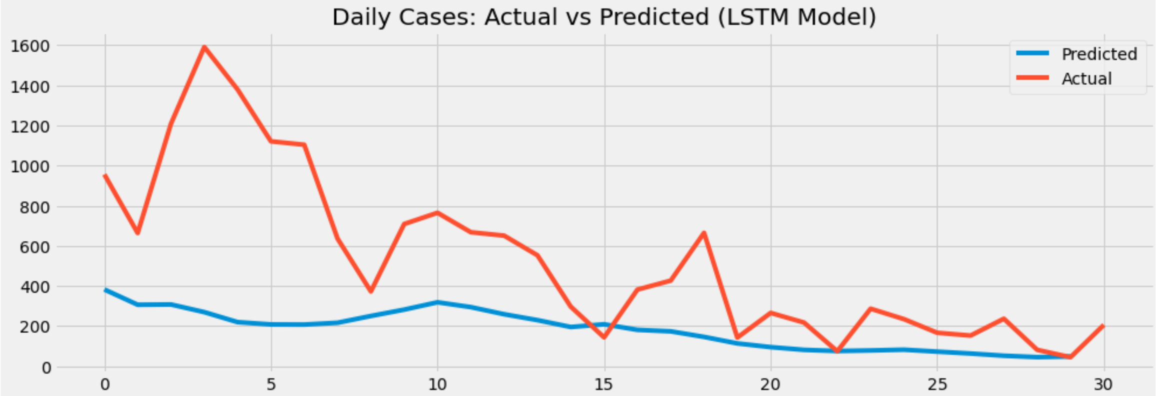
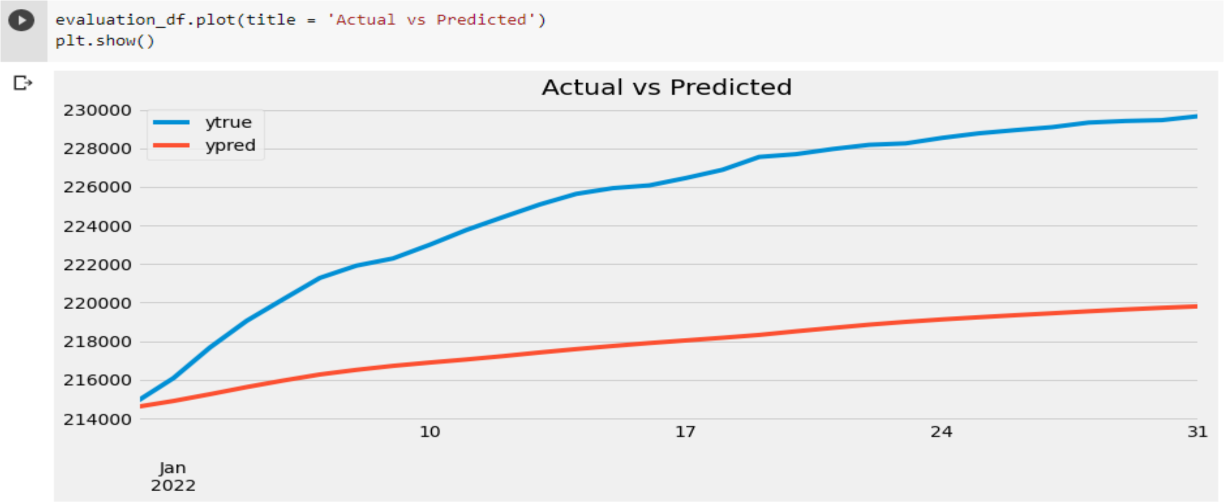


Figure 4.16: Actual vs Predicted – Daily Cases (LSTM)

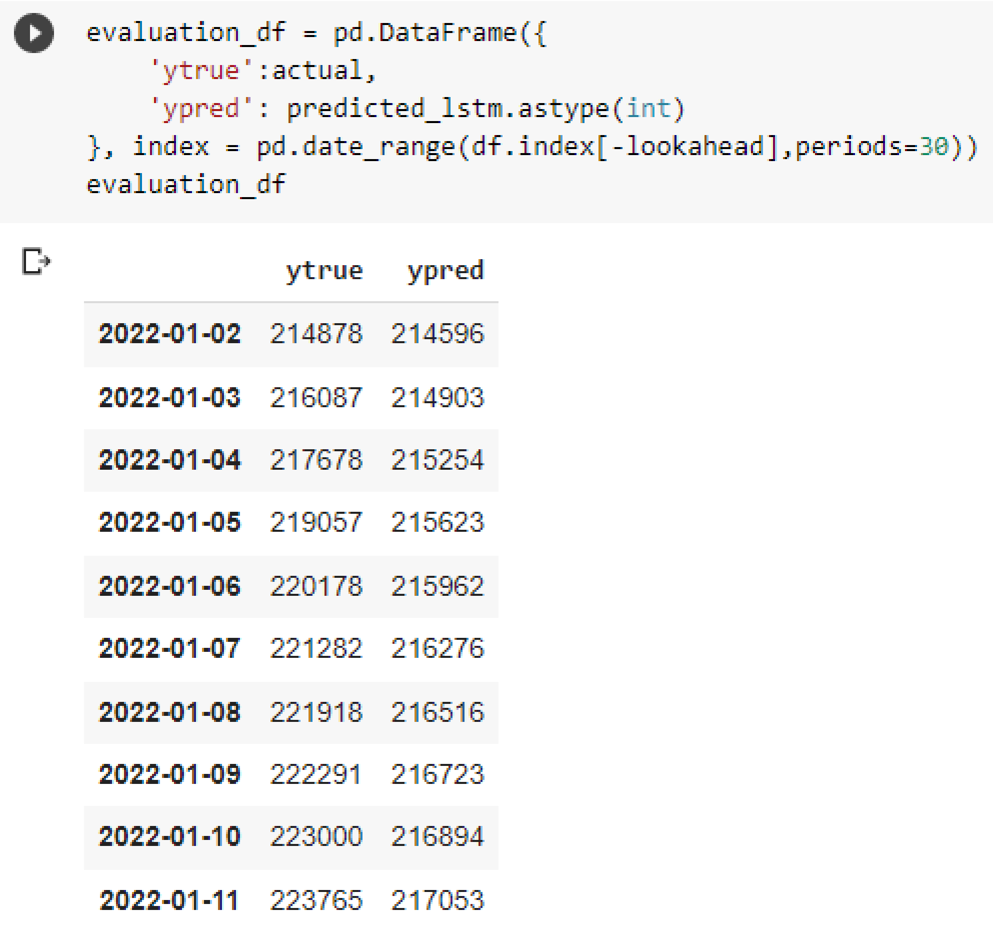
The graph below shows the Actual vs Predicted after the data was converted back to cumulative cases:



```
evaluation_df.plot(title = 'Actual vs Predicted')
plt.show()
```

Figure 17: Actual vs Predicted – Cumulative Cases (LSTM)

The predicted values graph closely resembles the actual values for values after 15 January. A sample of the actual versus the predicted values is shown in the table below.

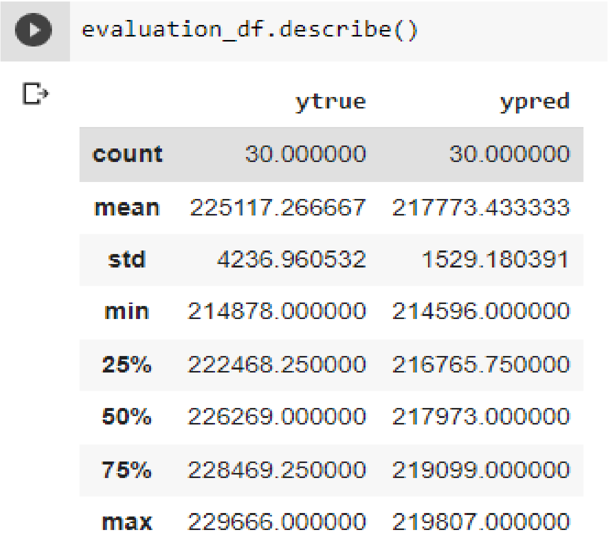


```
evaluation_df = pd.DataFrame({
    'ytrue':actual,
    'ypred': predicted_lstm.astype(int)
}, index = pd.date_range(df.index[-lookahead],periods=30))
evaluation_df
```

	ytrue	ypred
2022-01-02	214878	214596
2022-01-03	216087	214903
2022-01-04	217678	215254
2022-01-05	219057	215623
2022-01-06	220178	215962
2022-01-07	221282	216276
2022-01-08	221918	216516
2022-01-09	222291	216723
2022-01-10	223000	216894
2022-01-11	223765	217053

Figure 4.18: LSTM Actual vs Predicted Sample Data

The descriptive statistics of the actual and the predicted are shown below



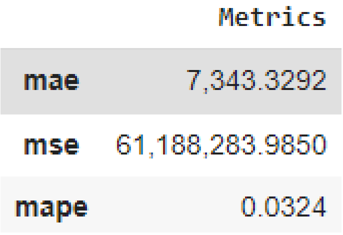
```
evaluation_df.describe()
```

	ytrue	ypred
count	30.000000	30.000000
mean	225117.266667	217773.433333
std	4236.960532	1529.180391
min	214878.000000	214596.000000
25%	222468.250000	216765.750000
50%	226269.000000	217973.000000
75%	228469.250000	219099.000000
max	229666.000000	219807.000000

Figure 4.19: LSTM Actual vs Predicted Descriptive Statistics

The descriptive statistics together with the cumulative plots show that the model constantly underpredicts the cumulative Covid 19 cases

To further understand the model, some metrics were calculated. These are the mean absolute error (MAE), mean squared error (MSE), and mean absolute percentage error (MAPE). The metrics for the LSTM model are shown below.



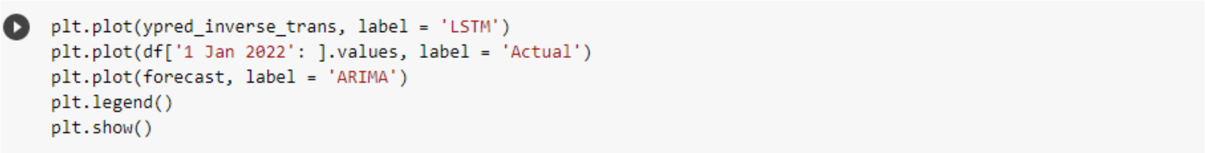
	Metrics
mae	7,343.3292
mse	61,188,283.9850
mape	0.0324

Figure 4.110: LSTM Metrics

The metrics show that the LSTM model is relatively good. On average it is wrong by 7343 cumulative cases.

4.9. LSTM VS ARIMA

The graph below shows the comparison of the LSTM and ARIMA models for the prediction of new cases.



```
plt.plot(ypred_inverse_trans, label = 'LSTM')
plt.plot(df['1 Jan 2022': ].values, label = 'Actual')
plt.plot(forecast, label = 'ARIMA')
plt.legend()
plt.show()
```

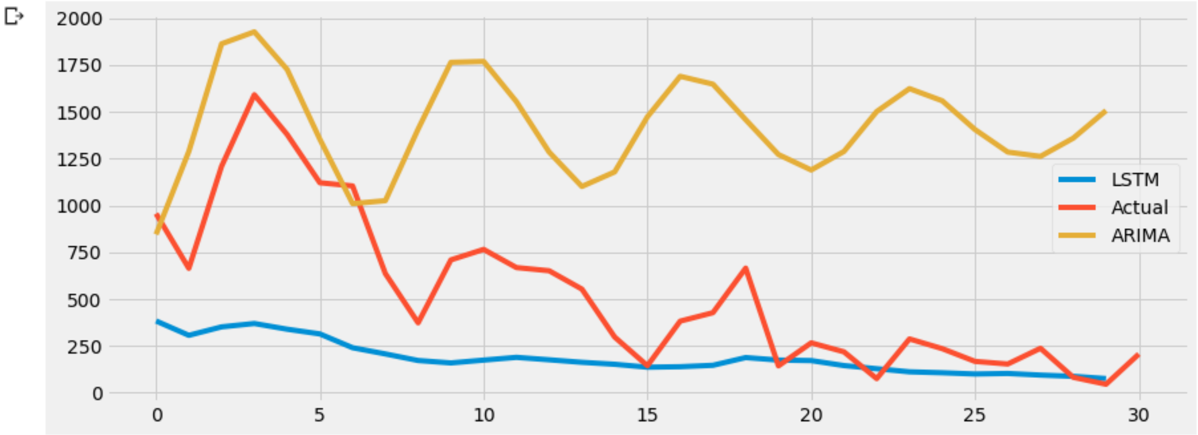
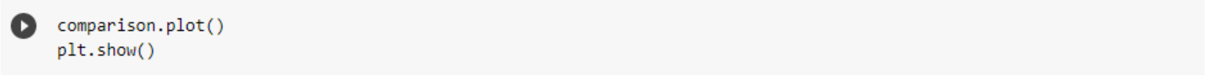


Figure 4.20: ARIMA vs LSTM – Daily Cases Predictions

The ARIMA line plot is relatively far away from the actual cases. The more predictions are in the future, the more wrong the ARIMA becomes. It is good for predicting fewer days into the future. For the LSTM, it predicts better in the future. Overall it is a better model as shown by the table and graphs below on cumulative cases.



```
comparison.plot()
plt.show()
```

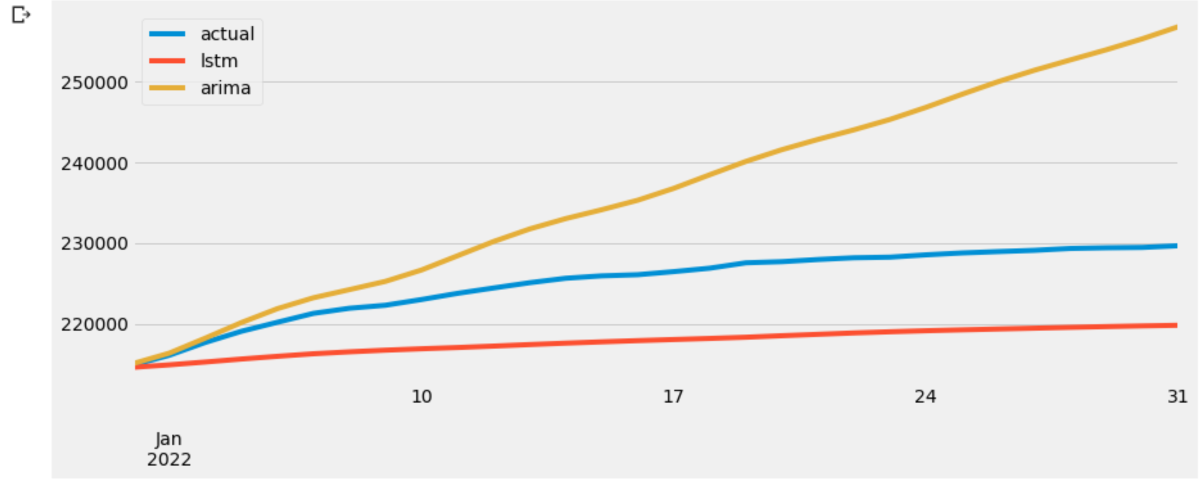
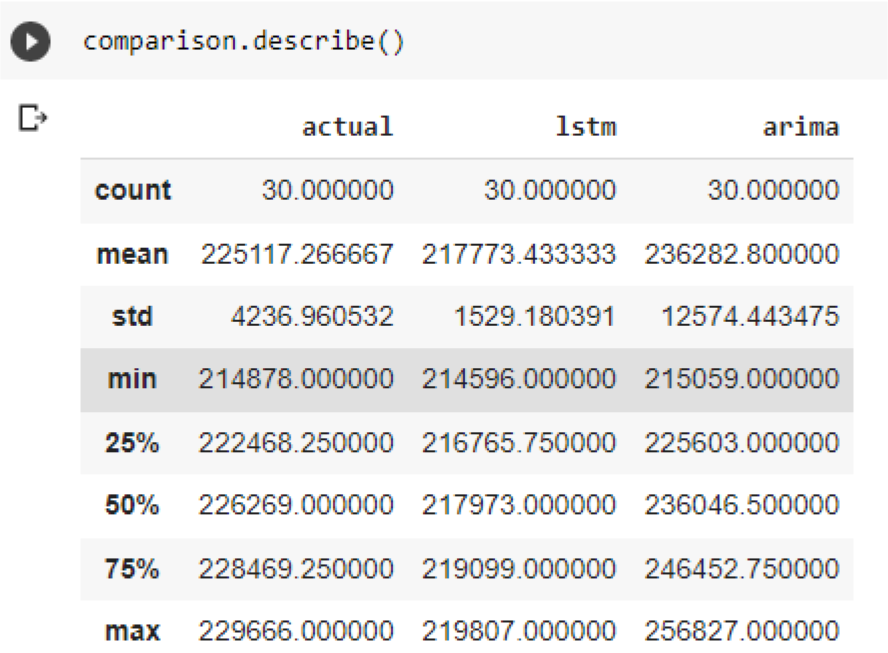


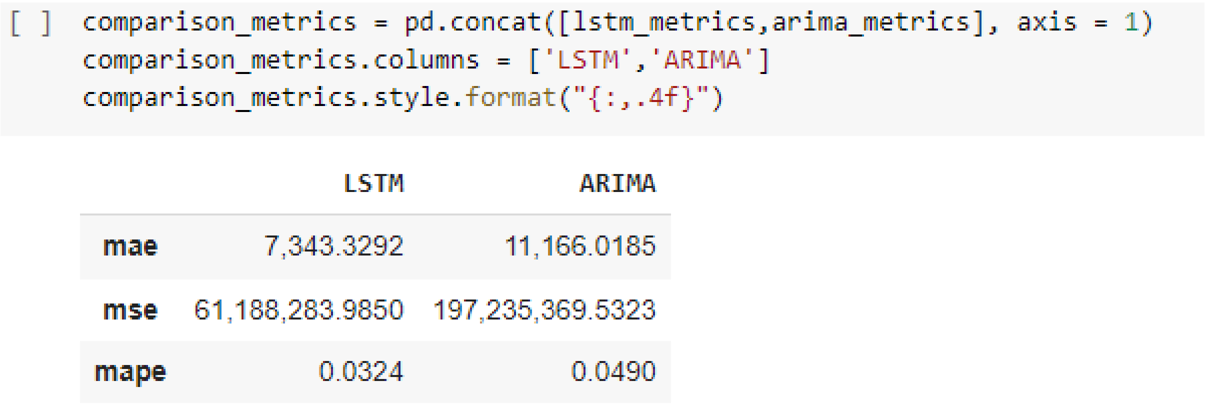
Figure 4.21: ARIMA vs LSTM – Cumulative Cases Predictions



```
comparison.describe()
```

	actual	lstm	arima
count	30.000000	30.000000	30.000000
mean	225117.266667	217773.433333	236282.800000
std	4236.960532	1529.180391	12574.443475
min	214878.000000	214596.000000	215059.000000
25%	222468.250000	216765.750000	225603.000000
50%	226269.000000	217973.000000	236046.500000
75%	228469.250000	219099.000000	246452.750000
max	229666.000000	219807.000000	256827.000000

Figure 4.11: Predictions Descriptive Statistics



```
[ ]  comparison_metrics = pd.concat([lstm_metrics,arima_metrics], axis = 1)
     comparison_metrics.columns = ['LSTM','ARIMA']
     comparison_metrics.style.format("{:,.4f}")
```

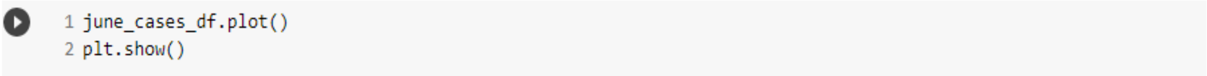
	LSTM	ARIMA
mae	7,343.3292	11,166.0185
mse	61,188,283.9850	197,235,369.5323
mape	0.0324	0.0490

Figure 4.23: ARIMA vs LSTM metrics

The LSTM model has better values across all metrics.

4.10. JUNE PREDICTIONS

The LSTM model was used to predict Covid 19 daily cases from 23 May to 21 June. The model can only predict 30 days into the future when given 60 days of previous data as input. The graph below shows the predictions for the 30 days.



```
june_cases_df.plot()
plt.show()
```

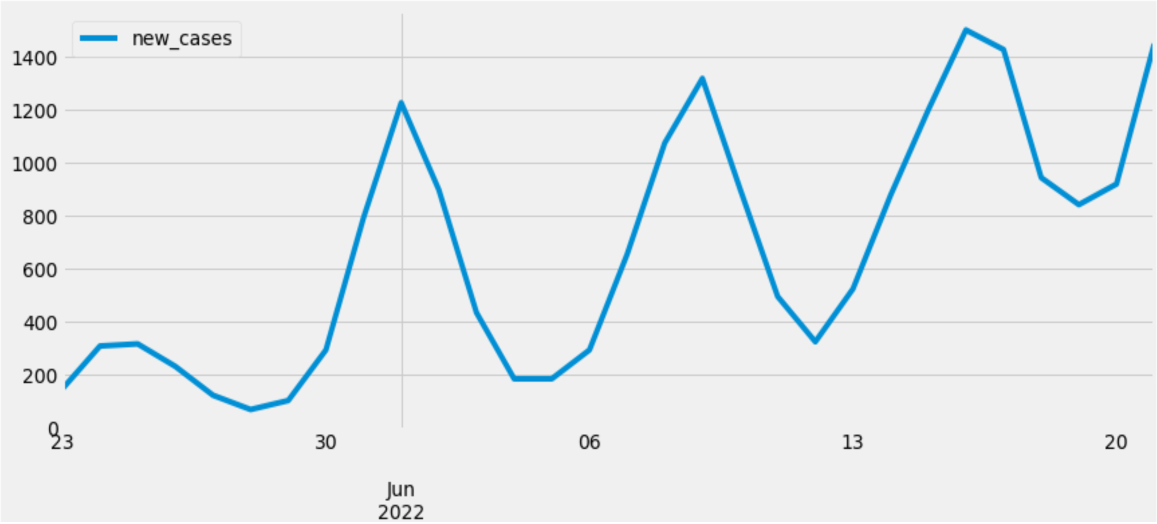
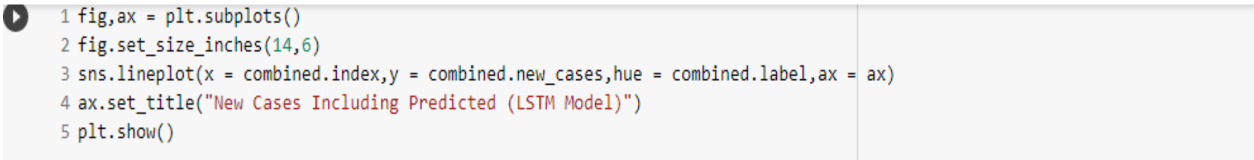


Figure 4.12: LSTM 30 days predictions

The graph below shows the predicted values plotted together with the historical data.



```
fig,ax = plt.subplots()
fig.set_size_inches(14,6)
sns.lineplot(x = combined.index,y = combined.new_cases,hue = combined.label,ax = ax)
ax.set_title("New Cases Including Predicted (LSTM Model)")
plt.show()
```

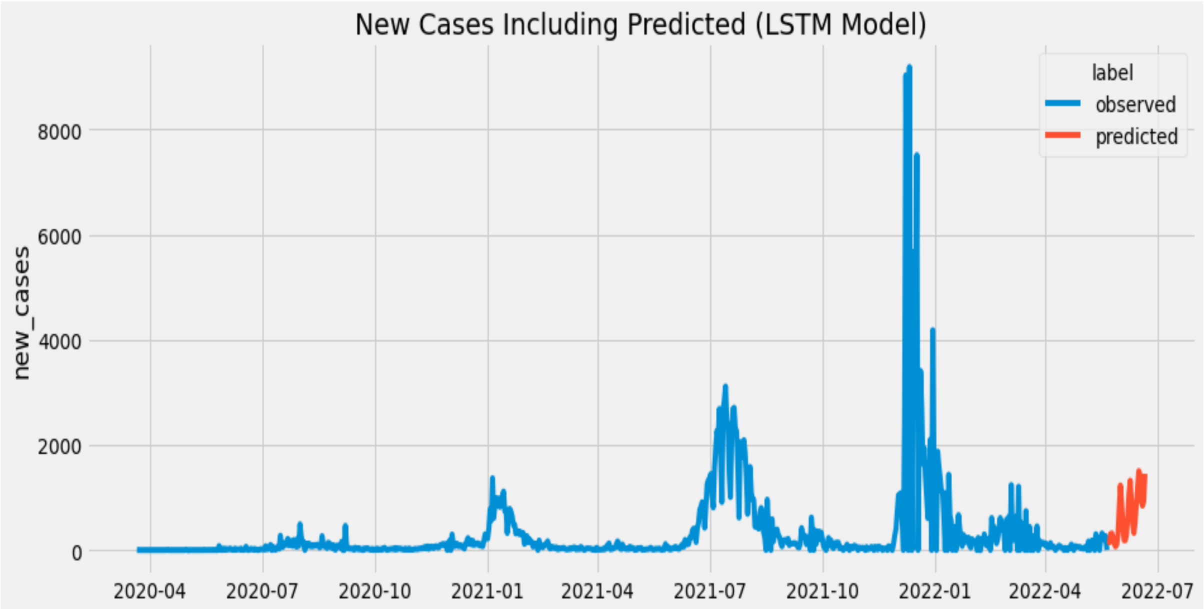


Figure 4.13: LSTM New Cases including Predicted

The red part shows the predicted values and the blue line shows the historical data. The LSTM model is predicting a positive trend in the daily Covid 19 cases over the month of June

Conclusion and Recommendations

5.1 INTRODUCTION

This chapter presents the summary conclusions and recommendations of the research.

5.2 ARIMA

The PMDArima module selected the parameters p = 7, d = 1, and q =2 as the best parameters for the ARIMA model. The ARIMA model was good in forecasting the near future. The more it went into the future the more wrong it became. ARIMA is weak in predicting data that does not have a defined pattern.

5.3 LSTM

The LSTM model is good for forecasting the long term. In its structure, it has the forget gate which deals with long-term dependencies. To further improve the predictive power of the LSTM model, one can increase the number of hidden layers and hidden neurons per layer. However, there is a trade-off between the increasing complexity of the model and the total time it will take to train the model. The more complex the model is the more time it will take to train it. Some hardware accelerators may even be needed.

4.11. CONCLUSION

The LSTM model was the better model as compared to the ARIMA model. It had a lower mean squared error, mean absolute error, and mean absolute percentage error. The model was then chosen to forecast daily cases from 23 May to 21 June. The model forecasted an upward trend in daily cases.

5.4 RECOMMENDATIONS

Time series models can be used to forecast the spread of Covid 19 with a reasonable amount of accuracy. To improve the accuracy of time series models, some exogenous variables may need to be incorporated into the models. Exogenous variables are independent variables whose values are generated outside the confines of the model. Some of the exogenous variables that may be tried are the number of vaccinated at each point in time, the number of deaths, and the presence of Covid measures like lockdowns. There are also other methods in the class of SIR models that can be used to forecast the spread of Covid 19. These models make some assumptions about the underlying population. If the assumptions are good, the models may outperform the time series models. However, setting the correct assumptions is difficult.

REFERENCES

Al-Turaiki I, Almutlaq F, Alrasheed H, Alballa N. Empirical Evaluation of Alternative Time-Series Models for COVID-19 Forecasting in Saudi Arabia. Int J Environ Res Public Health. 2021 Aug 16;18(16):8660. DOI: 10.3390/ijerph18168660. PMID: 34444409; PMCID: PMC8393561.

Zhao B, Mingzhe E, Cao J (2020) Time series analysis of Holt model and the Arima model facing Covid 19. *Ann Math Phys3(1)*

Xuan Z (2019) Epidemiological characteristics and time series analysis of hand, foot, and mouth diseases in *Wuzhou city*

Dr. Bin Zhao, 2020, Time series analysis of Holt model and the Arima facing Covid 19

“Zimbabwe starts Covid 19 vaccinations, Vice President Chiwenga gets first shot “*News 24*

R. Salgotra,2020, Time series analysis and forecast of the Covid 19 pandemic in India

J.D. Cryer, &Chan, K, -s (2008 in) Time-series Analysis with application R. Lowa *city USA Springer Texts in Statistics*

Ayoob N, Sharifrazi D, Alizadehsani R, et all, (2021*)* Time series forecasting of new cases and new deaths rate for Covid 19 using deep learning methods

Kafieh R, Arian R, Saeedizadeh N, Amini Z, Serej N, Minaee S, Yadav S, Vaezi A, Rezaei N, (2021) Covid 19 in Iran: Forecasting pandemic using deep learning *Computational and Mathematical Methods in Medicine*

Zhang R, Guo Z, Meng Y, Wang S, Li S, Niu R, Wang Y, Guo Q, Li Y (2021) Comparison of Arima and lstm in forecasting the incidence of hfmd combined and uncombined with exogenous meteorological variables in Ningbo China *International Journal of Environment Research and public health*

Chandra R, Jain A, Chauhan D (2021) Deep learning via LSTM models for Covid 19 infection forecasting in India

Singh S, Sundram B, Rajendran K, Lawk, Aris T, Ibrahim H, Dass S, Gill B (2021) Forecasting daily confirmed Covid 19 cases in Malaysia using ARIMA models *Journal of infection in developing countries*.

Murekachiro, D. (2016). Time series volatility forecasting of the Zimbabwe stock exchange. The *International Journal of Business and Management* (ISSN 2321 -8916). Vol 4 Issue 3, 41-42

Gecili E, Ziady A, Szczesniak R (2020) Forecasting Covid 19 confirmed cases, deaths and recoveries: Revisiting established time series modeling through novel applications for the USA and Italy

Hyndman, R J, & Athanasopoulos, G. (2014), Forecasting: Principles and practice. OTexts

IBM. (2013). IBM SPSS Forecasting 22

Ocaya.Bruno, Ruranga. Charles, &Kaberuka. William (2013). Foreign Direct Investment and Economic growth in Rwanda. A time-series analysis. *Journal of business management corporate affairs*, 11-18

Ramasubramanian. (2015). Time series analysis. New Dehi: I.A.S.R.I library avenue

Naresh K, Seba S (2020) Covid 19 pandemic prediction using time series forecasting model.*11th International conference on computing, communication, and networking technology*

Junling Luo, Zhongliang Zhang, Yao Fu, Feng Rao, (2021), Time series prediction of COVID-19 transmission in America using LSTM and XGBoost algorithms, *Results in Physics, Volume 27104462, ISSN 2211-3797,*

https://doi.org/10.1016/j.rinp.2021.104462.

(https://www.sciencedirect.com/science/article/pii/S2211379721005775)

Madini O. Alassafi, Mutasem Jarrah, Reem Alotaibi, (2022) Time series predicting of COVID-19 based on deep learning, *Neurocomputing, Volume 468, Pages 335-344, ISSN 0925-2312,https://doi.org/10.1016/j.neucom.2021.10.035.(https://www.sciencedirect.com/science/article/pii/S0925231221015150)*

Kibria, H. B., Jyoti, O., & Martin, A. (2022). Forecasting the spread of the third wave of COVID-19 pandemic using time series analysis in Bangladesh. *Informatics in medicine unlocked*, *28*, 100815. https://doi.org/10.1016/j.imu.2021.100815 Al-Turaiki, I.; Almutlaq, F.; Alrasheed, H.; Alballa, (2021) N. Empirical Evaluation of Alternative Time-Series Models for COVID-19 Forecasting in Saudi Arabia. *Int. J. Environ. Res. Public Health 2021, 18, 8660. HTTPS:// doi.org/10.3390/ijerph18168660 Academic Editor: Paul B. Tchounwou*

Gebretensae YA, Asmelash D. (2021) Trend Analysis and Forecasting the Spread of COVID-19 Pandemic in Ethiopia Using Box–Jenkins Modelling Procedure. *Int J Gen Med*.; 14:1485-1498https://doi.org/10.2147/IJGM.S306250

Fadoua Khennou, Moulay A. Akhloufi Perception, Robotics, and Intelligent Machines Research Group (PRIME) Department of Computer Science Université de Moncton Moncton, NB, Canada {fadoua. khennou, moulay.akhloufi}@umoncton.ca

Appendix

The whole code used in the data analysis and model building is be found on the following link:

Covid 19 Cases Prediction Model

```
import pandas as pd
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import pmdarima as p
from statsmodels.tsa.seasonal import seasonal_decompose
import statsmodels as sm
import seaborn as sns
import datetime
from sklearn import preprocessing, metrics
```

Read Data

```
total = pd.read_excel('/content/drive/MyDrive/Datasets/Zim Covid cases/
covid data.xlsx')
total = total.set_index('date')
df = total.diff().fillna({"total_cases":total.iat[0,0]})
df = df.rename(columns = {"total_cases":"new_cases",})
df = df[:'31 Jan 2022']
```

Descriptive Statistics

```
df.describe().T

plt.style.use('fivethirtyeight')
df.plot.box()
plt.show()

df.loc[df.new_cases < 800].plot.hist(figsize = (6,4))
plt.xlabel('Daily Cases')
plt.legend('')
plt.show()

df.loc[df.new_cases < 800].plot.hist(figsize = (6,4))
plt.xlabel('Daily Cases')
plt.legend('')
plt.show()
plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = (15,5)
fig, ax = plt.subplots(1,2)
ax[0].plot(df.new_cases)
ax[0].set_title('New Cases', fontweight = 'bold',size = 20)
ax[1].plot(total.total_cases)
ax[1].set_title('Cumulative Cases', fontweight = 'bold',size = 20)
plt.setp(ax[0].get_xticklabels(), rotation=45, ha='right')
plt.setp(ax[1].get_xticklabels(), rotation=45, ha='right')
plt.show()

labels = pd.date_range('2020-03-20','2022-01-31',15)
df.new_cases.cummax().plot(figsize = (10,5), label = 'cumulative max')
df.new_cases.plot(label = 'daily cases')
plt.legend()
plt.show()

df.plot.hist(figsize = (6,4), legend = False)
plt.xlabel('New Cases')
plt.show()
```

Autocorrelation and Partial AutoCorrelation

```
plt.rcParams['figure.figsize'] = (13,5)
fig, ax = plt.subplots(1,2)
sm.graphics.tsaplots.plot_acf(df.new_cases,ax = ax[0])
sm.graphics.tsaplots.plot_pacf(df.new_cases,ax = ax[1])
ax[0].set(xlabel = 'Lag'); ax[1].set(xlabel = 'Lag')
fig.show()
```

Decompose Time Series

```
decomposed = seasonal_decompose(df.new_cases,model = 'additive')
fig,(ax,ax1,ax2,ax3) = plt.subplots(4,sharex = True,figsize = (8,8),)
fig.set_size_inches(15,10)
ax.plot(decomposed.observed, color = 'red')
ax1.plot(decomposed.seasonal, color = 'green')
ax2.plot(decomposed.trend,color = 'black')
ax3.plot(decomposed.resid)
ax3.set_title('resid'); ax2.set_title('trend'); ax1.set_title('seasonal
'); ax.set_title('obseved')
plt.xticks(rotation = 10)
plt.show()
```

ARIMA Model

Tests

```
adf = p.arima.stationarity.ADFTest()
kpss =p.arima.stationarity.KPSSTest()
pp = p.arima.stationarity.PPTest()
```

```
adf.should_diff(df.new_cases)
kpss.should_diff(df.new_cases)
pp.should_diff(df.new_cases)
```

Model

```
arima = p.AutoARIMA(start_p= 0,start_q= 0,max_p=10,max_q=10,max_d=5,max
_order=20)
arima.fit(df[:'1 Jan 2022'].new_cases)
arima.model_
```

Predictions

```
forecast = arima.predict(30)
index = pd.date_range('2 jan 2022','31 jan 2022',5)
index = [pd.Period(i, 'D') for i in index]

plt.plot(forecast, label = 'forecast')
plt.plot(df['2 Jan 2022':].values, label = 'actual')
plt.xticks(np.linspace(0,29,5), index)
plt.legend()
plt.title('Daily Cases: Actual vs Predicted')
plt.show()

forecasted_cumulative_cases = total.loc['1 jan 2022'].values + np.cumsu
m(forecast)
actual = total.loc['2 jan 2022': '31 jan 2022'].total_cases.values
```

Actual vs Predicted

```
plt.plot(forecasted_cumulative_cases, label = 'forecast')
plt.plot(actual, label = 'actual')
plt.xticks(np.linspace(0,29,5), index)
plt.legend()
plt.show()
```

ARIMA Metrics

```
mae = metrics.mean_absolute_error(actual,forecasted_cumulative_cases)
mse = metrics.mean_squared_error(actual,forecasted_cumulative_cases)
mape = metrics.mean_absolute_percentage_error(actual,forecasted_cumulat
ive_cases)

arima_metrics = pd.DataFrame([mae,mse,mape], columns = ['metrics'], ind
ex = ['mae','mse','mape'])
arima_metrics.style.format("{:,.2f}")
```

LSTM Model

Data Transformation

```
class TimeSeriesDataPreprocessor:
  """
      Create supervised training data from a time series dataframe.
```

```
    """

    def __init__(self,df,lookback,lookahead):
      """
      Parameters:
        1. df = dataframe with only one column or  which is the series to
   be modelled
        2. look = the previous data points that we want to use as inputs
                        or the future values we want to forecast
                        for ahead put look + 1
      """
      self.df = df
      self.lookback = lookback
      self.lookahead = lookahead

    def create_features(self):
      self.x = pd.concat([self.df.shift(i) for i in range(self.lookback)]
  , axis = 1)
      self.x.columns = list(range(self.x.shape[1],0,-1))
      self.x = self.x.sort_index(axis = 1)
      return self.x

    def create_response(self):
      self.y = pd.concat([self.df.shift(-
  i) for i in range(self.lookahead)],axis = 1)
      self.y.columns = list(range(self.y.shape[1]))
      self.y = self.y.drop(0,axis = 1)
      return self.y

    def create_input(self):
      """
      This function is a continuation of create features and labels
      It removes the np.nans created by using the shift function in creat
  e__features(and response)
      It then return x and y dataframes of the same length without np.nan
  s
      The starting index is from x since it dropped np.nans from the top
      The ending index is from y since it dropped np.nans from the bottom
      """

      self.x = self.create_features().dropna()
      self.y = self.create_response().dropna()

      self.x = self.x.loc[self.x.index[0]:self.y.index[len(self.y) - 1]]
      self.y = self.y.loc[self.x.index[0]:self.y.index[len(self.y) - 1]]

      return self.x,self.y
```

```
data = df.copy()
df = df[:'31 jan 2022']
# Train Test split
lookahead = 30
lookback = 60
train = df.loc[:df.index[-lookahead]]
ytest = df.loc[df.index[-lookahead]:]
xtest = df.loc[df.index[-lookback - lookahead + 1]:df.index[-
lookahead]]

#Transform using log. We add 1 because there are zeros in the data
#Minmax is put data in the scale 0 to 1
train = np.log(train + 1)
minmax = preprocessing.MinMaxScaler()
train = pd.DataFrame(minmax.fit_transform(train))

#Instantiate the TrainData Class
data = TimeSeriesDataPreprocessor(train,lookback,lookahead + 1)
x,y = data.create_input()
xtrain = np.reshape(x.values,(x.values.shape[0],x.values.shape[1],1))
xtrain.shape
```

Hyperparameters

```
batch_size = 64
epochs = 1000
units1 = 64
units2 = 180
units3 = 120
output = 30
```

Model

```
def lstm_model(units1,units2,units3,output):
  model = tf.keras.models.Sequential([
          tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(units = un
its1, return_sequences=True,input_shape = (xtrain.shape[1],1), name = '
LSTM_1'), name = 'Bidirectional_LSTM_1'),
          tf.keras.layers.Dropout(0.2,name = 'Dropout_1'),

          tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(units = un
its2, return_sequences=True,name = 'LSTM_2'), name = 'Bidirectional_LST
M_2'),
          tf.keras.layers.Dropout(0.2, name = 'Dropout_2'),

          tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(units = un
its2, return_sequences=True,name = 'LSTM_3'), name = 'Bidirectional_LST
M_3'),
          tf.keras.layers.Dropout(0.2, name = 'Dropout_3'),
          tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(units = units3,name = 'LSTM_4'), name = 'Bidirectional_LSTM_4'),
          tf.keras.layers.Dropout(0.2, name = 'Dropout_4'),

          tf.keras.layers.Dense(units = output,activation='relu', name = 'Output')
  ])
  return model

model = lstm_model(units1,units2,units3,output)
model.compile('adam',loss = 'mse',metrics = ['mae'])

mc = tf.keras.callbacks.ModelCheckpoint('/content/drive/MyDrive/DataSets/Keras Models/covid_new_cases3.hdf5',monitor = 'loss',verbose = 1,save_best_only=True)
es = tf.keras.callbacks.EarlyStopping('loss',patience=20,verbose=1)

history = model.fit(xtrain,y,epochs = epochs,batch_size=batch_size, callbacks=[mc,es])

model.summary()
tf.keras.utils.plot_model(model, show_shapes=True)
```

Analysing the Training History

```
hist_df = pd.DataFrame(history.history)
hist_df.plot(title = 'Training Metrics', xlabel = 'Epochs',ylabel = 'Loss')
plt.show()
```

Evaluation

```
full_model = tf.keras.models.load_model('/content/drive/MyDrive/DataSets/Keras Models/covid_new_cases3.hdf5')
#preprocess the test set
xtest_df = TimeSeriesDataPreprocessor(pd.DataFrame(minmax.transform(np.log(xtest + 1))),lookback,None).create_features().dropna()
xtest_array = np.reshape(xtest_df.values,(xtest_df.values.shape[0],xtest_df.values.shape[1],1))
xtest_array.shape

ypred = full_model.predict(xtest_array)
ypred

#Inverse transform the predicted
ypred_inverse_trans = (np.exp(minmax.inverse_transform(ypred)) - 1)[0]
ypred_inverse_trans
plt.plot(ypred_inverse_trans, label = "Predicted")
plt.plot(df['1 Jan 2022': ].values, label = 'Actual')
plt.title('Daily Cases: Actual vs Predicted (LSTM Model)')
plt.legend()
plt.show()

plt.plot(ypred_inverse_trans, label = 'LSTM')
plt.plot(df['1 Jan 2022': ].values, label = 'Actual')
plt.plot(forecast, label = 'ARIMA')
plt.legend()
plt.show()

lstm_metrics = pd.DataFrame([metrics.mean_absolute_error(actual, predic
ted_lstm),metrics.mean_squared_error(actual, predicted_lstm), metrics.m
ean_absolute_percentage_error(actual, predicted_lstm)],
             columns = ['Metrics'], index = ['mae','mse', 'mape'])
lstm_metrics.style.format("{:,.4f}")

evaluation_df = pd.DataFrame({
    'ytrue':actual,
    'ypred': predicted_lstm.astype(int)
}, index = pd.date_range(df.index[-lookahead],periods=30))
evaluation_df

evaluation_df.plot(title = 'Actual vs Predicted')
plt.show()

evaluation_df.describe()
comparison = evaluation_df.assign(arima = forecasted_cumulative_cases.a
stype(int))
comparison.columns = ['actual','lstm','arima']
comparison
comparison.plot()
plt.show()

comparison_metrics = pd.concat([lstm_metrics,arima_metrics], axis = 1)
comparison_metrics.columns = ['LSTM','ARIMA']
comparison_metrics.style.format("{:,.4f}")
```

Forecast the Future

```
all_data = pd.read_csv('/content/drive/MyDrive/Datasets/Zim Covid cases
/covid.csv')
all_data.Date = pd.to_datetime(all_data.Date, format='%d/%m/%Y')
all_data.set_index('Date', inplace = True)
all_data = all_data.diff()
all_data.columns = ['new_cases']
all_data

# forecast
test_set = np.log(all_data + 1).tail(60)
test_set = pd.DataFrame(minmax.transform(test_set))
```

```
test_set_obj = TimeSeriesDataPreprocessor(test_set,60,None)
test = test_set_obj.create_features().dropna()
test = np.reshape(test.values,(test.values.shape[0],test.values.shape[1
],1))
test.shape

new_cases_pred = full_model.predict(test)
june_cases = minmax.inverse_transform(new_cases_pred)
june_cases = np.exp(june_cases) - 1
june_cases_df = pd.DataFrame(np.int32(june_cases)).T
june_cases_df.index = pd.date_range('2022/05/23',periods=30)
june_cases_df.columns = ['new_cases']
june_cases_df

june_cases_df.plot()
plt.show()

combined = pd.concat([all_data,june_cases_df])
combined.loc[:all_data.index[-1],'label'] = 'observed'
combined.loc[all_data.index[-1]:,'label'] = 'predicted'
combined

fig,ax = plt.subplots()
fig.set_size_inches(14,6)
sns.lineplot(x = combined.index,y = combined.new_cases,hue = combined.l
abel,ax = ax)
ax.set_title("New Cases Including Predicted (LSTM Model)")
plt.show()
```