REPUBLIC OF TURKEY
ADANA ALPARSLAN TÜRKEŞ SCIENCE AND TECHNOLOGY UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF INDUSTRIAL ENGINEERING

PREDICTION OF COVID-19 CASE AND DEATH NUMBERS IN TURKEY BY USING ARTIFICIAL INTELLIGENCE TECHNIQUES

NAZLI NUR KARABULUT

MASTER OF SCIENCE

ADANA 2022

REPUBLIC OF TURKEY
ADANA ALPARSLAN TÜRKEŞ SCIENCE AND TECHNOLOGY UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF INDUSTRIAL ENGINEERING

PREDICTION OF COVID-19 CASE AND DEATH NUMBERS IN TURKEY BY USING ARTIFICIAL INTELLIGENCE TECHNIQUES

NAZLI NUR KARABULUT
MASTER OF SCIENCE

SUPERVISOR
ASSOC. PROF. DR. MUSTAFA GÖÇKEN

ADANA 2022

I hereby declare that all information in this thesis has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all information that is not original to this work.

Nazlı Nur KARABULUT

ABSTRACT

PREDICTION OF COVID-19 CASE AND DEATH NUMBERS IN TURKEY BY USING ARTIFICIAL INTELLIGENCE TECHNIQUES

Nazlı Nur KARABULUT

Department of Industrial Engineering

Supervisor: Assoc. Prof. Dr. Mustafa GÖÇKEN

February 2022, 79 pages

The coronavirus, which emerged in China in the last months of 2019, soon turned into a pandemic that affected the whole world. The countries of the world, which were caught unprepared for such a pandemic, carried out data analyzes to be able to take correct and timely measures at that time by using artificial intelligence techniques to analyze the current situation and to prevent the handicaps arising in many areas due to the virus.

In this study, using the Turkey coronavirus data set, predictive analysis was performed with three different boosting algorithms: AdaBoost, CatBoost, and XGBoost algorithms. In addition, predictions related to the considered data set were made by using ANN algorithms whose hyperparameters were optimized with the grid search algorithm. Basic performance metrics are used to evaluate and interpret the results of the analysis correctly. Necessary comparisons are made for four different analyzes on the same data set and the most appropriate algorithm is determined for the Turkey coronavirus data set.

Keywords: Coronavirus, Prediction, Boosting Algorithms, Artificial Neural Networks

ÖZET

TÜRKİYE COVİD-19 VAKA VE ÖLÜM SAYILARININ YAPAY ZEKÂ TEKNİKLERİ İLE TAHMİN EDİLMESİ

Nazlı Nur KARABULUT
Endüstri Mühendisliği Anabilim Dalı
Danışman: Doç. Dr. Mustafa GÖÇKEN
Şubat 2022, 79 sayfa

Çin'de 2019 yılının son aylarında ortaya çıkan koronavirüs, kısa zamanda tüm dünyayı etkisi altına alan bir pandemiye dönüşmüştür. Böylesi bir pandemiye hazırlıksız yakalanan dünya ülkeleri, hem mevcut durumu analiz etmek hem de virüs sebebi ile pek çok alanda ortaya çıkan handikapları önlemek, doğru ve zamanında önemler alabilmek için yapay zekâ tekniklerinden faydalanarak veri analizleri gerçekleştirmişlerdir.

Bu çalışmada Türkiye koronavirüs veri seti kullanılarak, yapay zekâ teknikleri ile tahmin analizleri yapılmıştır. Kullanılacak yapay zekâ teknikleri belirlenmiştir. Üç farklı artırma algoritması ile tahmin analizi gerçekleştirilmiştir. Bunlar, AdaBoost, CatBoost ve XGBoost algoritmalarıdır. Ayrıca, ızgara arama algoritması ile hiperparametreleri optimize edilen yapay sinir ağı algoritmaları kullanılarak göz önüne alınan veri setine ait tahminler gerçekleştirilmiştir. Analiz sonuçlarının değerlendirilmesi ve doğru yorumlanabilmesi için temel performans ölçütleri kullanılmıştır. Aynı veri seti üzerinde yapılan dört farklı analiz için gerekli karşılaştırmalar yapılıp Türkiye koronavirüs veri setine en uygun algoritma belirlenmiştir.

Anahtar Kelimeler: Koronavirüs, Tahminleme, Artırma Algoritmaları, Yapay Sinir Ağları

To my family

ACKNOWLEDGEMENTS

I would like to thank my invaluable supervisor Assoc. Prof. Dr. Mustafa GÖÇKEN, for his endless support, suggestions and patience throughout the process. I would like to express my sincere thanks for the support and understanding given during the times when it is most difficult to stay motivated, especially during this pandemic.

I would also like to express my gratitude to Assoc. Prof. Dr. Ayşe Tuğba DOSDOĞRU for all her feedback and help.

I would also like to thank my family, especially my mother, and all my friends who have always supported me.

TABLE OF CONTENTS

TABLE OF CONTENTS viii

LIST OF FIGURES x

LIST OF TABLES xii

NOMENCLATURE xiv

1. INTRODUCTION 1

2. LITERATURE REVIEW 4

3. MATERIALS AND METHODS 12

3.1. Artificial Intelligence 12

3.2. Machine Learning 13

3.3. Artificial Neural Networks 13

3.3.1. Types of ANNs 16

3.3.1.1. Feedforward ANNs 16

3.3.1.2. Feedback ANNs 16

3.3.2. Data Training in ANNs 16

3.3.2.1. Supervised Learning 16

3.3.2.2. Unsupervised Learning 16

3.3.2.3. Reinforcement Learning 17

3.4. Ensemble Learning 18

3.4.1. Ensemble Methods 18

3.4.2. Boosting Algorithms 20

3.4.3. AdaBoost (Adaptive Boosting) Algorithm 21

3.4.4. CatBoost (Categorical Boosting) Algorithm 23

3.4.5. XGBoost (Extreme Gradient Boosting) Algorithm 26

3.5. Error Measures 29

4. RESULTS AND DISCUSSION 32

4.1. Description of Turkey Coronavirus Dataset 32

4.2. Prediction with AdaBoost Algorithm 36

4.2.1. Predicted number of the cases with AdaBoost algorithm 36

4.2.2. Predicted number of the deaths with AdaBoost algorithm 39

4.3. Prediction with CatBoost algorithm 41

4.3.1. Predicted number of the cases with CatBoost algorithms 41

4.3.2. Predicted number of the deaths with CatBoost algorithm 43

4.4. Prediction with XGBoost algorithm 46

4.4.1. Predicted number of the cases with XGBoost algorithm 46

4.4.2. Predicted number of the deaths with XGBoost algorithm 48

4.5. Comparison of prediction results 51

4.5.1. Comparison with daily case and cumulative case results 51

4.5.1.1. Evaluation of case predictions according to the CV value 53

4.5.2. Comparison with daily death and cumulative death results 54

4.5.2.1. Evaluation of death predictions according to the CV value 56

4.6. Evaluation of the best performing CatBoost algorithm with Turkey data 57

4.7. Prediction with ANN 58

4.7.1. Hyperparameters optimization 58

5. CONCLUSIONS 69

6. RECOMMENDATIONS 70

REFERENCES 71

LIST OF FIGURES

Figure 3. 1 Structure of subsets 14
Figure 3. 2 Layers of ANN 15
Figure 3. 3 Basic structure of neuron 15
Figure 3. 4 Basic types of ensemble methods 18
Figure 3. 5 Pseudocode of the Bagging algorithm 19
Figure 3. 6 The Stacking algorithm 20
Figure 3. 7 The Boosting algorithm (Zhang & Haghani, 2015) 21
Figure 3. 8 Pseudocode of the AdaBoost.R algorithm (Freund & Schapire, 1997) 22
Figure 3. 9 Pseudocode of the CatBoost algorithm 24
Figure 3. 10 The BuildTree function 25
Figure 3. 11 The XGBoost algorithm ("XGBoost", 2019) 27
Figure 4. 1 Daily patients 33
Figure 4. 2 Daily cases 34
Figure 4. 3 Daily deaths 35
Figure 4. 4 Total cases 36
Figure 4. 5 Daily cases prediction with AdaBoost algorithm 37
Figure 4. 6 Cumulative cases prediction with AdaBoost algorithm 37
Figure 4. 7 Daily deaths prediction with AdaBoost algorithm 39
Figure 4. 8 Cumulative deaths prediction with AdaBoost algorithm 39
Figure 4. 9 Daily cases prediction with CatBoost algorithm 41
Figure 4. 10 Cumulative cases prediction with CatBoost algorithm 42
Figure 4. 11 Daily deaths prediction with CatBoost algorithm 44
Figure 4. 12 Cumulative deaths prediction with CatBoost algorithm 44
Figure 4. 13 Daily cases prediction with XGBoost algorithm 46
Figure 4. 14 Cumulative cases prediction with XGBoost algorithm 47
Figure 4. 15 Daily deaths prediction with XGBoost algorithm 49
Figure 4. 16 Cumulative deaths prediction with XGBoost algorithm 49

Figure 4. 17 The daily case hyperparameters for the ANN model.....61
Figure 4. 18 The cumulative case hyperparameters for the ANN model.....61
Figure 4. 19 The daily deaths hyperparameters for the ANN model 62
Figure 4. 20 The cumulative deaths hyperparameters for the ANN model 62
Figure 4. 21 Daily case ANN model 63
Figure 4. 22 Cumulative case ANN model 64
Figure 4. 23 Daily death ANN model 64
Figure 4. 24 Cumulative death ANN model.....65

LIST OF TABLES

Table 4. 1 Data description and data range 33
Table 4. 2 Performance results of the AdaBoost algorithm on daily cases and cumulative cases. 38
Table 4. 3 Performance results of the AdaBoost algorithm on daily deaths and cumulative deaths. 40
Table 4. 4 Performance results of the CatBoost algorithm on daily cases and cumulative cases. 42
Table 4. 5 Performance results of the CatBoost algorithm on daily deaths and cumulative deaths. 45
Table 4. 6 Performance results of the XGBoost algorithm on daily cases and cumulative cases. 47
Table 4. 7 Performance results of the XGBoost algorithm on daily deaths and cumulative deaths. 50
Table 4. 8 Performance results of daily case predictions 51
Table 4. 9 CPU times of boosting techniques with daily case data. 52
Table 4. 10 Performance results of cumulative case predictions 52
Table 4. 11 CPU times of boosting techniques with cumulative case data. 52
Table 4. 12 Some instances of the Covid-19 predictions using cumulative case data 53
Table 4. 13 CV values for case predictions 53
Table 4. 14 Performances of boosting algorithms on daily death predictions 54
Table 4. 15 CPU time performances of boosting algorithms on daily death predictions 55
Table 4. 16 Performances of boosting algorithms on cumulative death predictions 55
Table 4. 17 CPU time performance of boosting algorithms on cumulative death predictions 56
Table 4. 18 A sample dataset of actual vs. predicted number of Covid-19 cumulative deaths. 56
Table 4. 19 CV values for death predictions 57
Table 4. 20 Performance results of CatBoost predictions 58
Table 4. 21 The grid search hyperparameter sets for the ANN model 59
Table 4. 22 The best hyperparameters for the ANN model 60
Table 4. 23 CPU time of ANN case predictions 65

Table 4. 24 Performance results of ANN predictions 66
Table 4. 25 CPU time of ANN death predictions.....67
Table 4. 26 Performance results of ANN predictions 67

NOMENCLATURE

AdaBoost	: Adaptive Boosting
ANN	: Artificial Neural Network
ARIMA	: Autoregressive Integrated Moving Average
CatBoost	: Categorical Boosting
CV	: Coefficient of Variation
EDA	: Exploratory Data Analysis
EVS	: Explained Variance Score
LSTM	: Long Short-Term Memory
MAE	: Mean Absolute Error
MAPE	: Mean Absolute Percentage Error
MERS	: Middle East Respiratory Syndrome
MSE	: Mean Squared Error
MSLE	: Mean Squared Logarithmic Error
R^2	: Coefficient of Determination
RMSE	: Root Mean Square Error
SARS	: Severe Acute Respiratory Syndrome
SIR	: Susceptible, Infective and Recover/Removed
SVM	: Support Vector Machine
WHO	: World Health Organization
XGBoost	: Extreme Gradient Boosting

1. INTRODUCTION

Throughout the ages, there have been occurred many pandemics which affected people's lives in many ways such as economic, social, educational etc. in the world. If looked at the root of the word pandemic in the literature, pandemic originates from the Greek pan meaning "all" and demos "the people", and generally, it refers to a contagious epidemic that is common across the country or on one or more continents at the same time (Honigsbaum, 2009). According to the World Health Organization (WHO), A new disease common around the world is a pandemic (WHO, 2010).

Looking at the twentieth century, there have been happened three flu pandemics at certain intervals. The most serious of which was the supposed "Spanish Flu" (brought about by an A (H1N1) infection), assessed to have caused 20–50 million passing in 1918–1919. Milder pandemics happened accordingly in 1957–1958 (the "Asian Flu" brought about by an A (H2N2) infection) and in 1968 (the "Hong Kong Flu" brought about by an A (H3N2) infection), which were assessed to have caused 1–4 million passing each (WHO).

The Severe Acute Respiratory Syndrome (SARS) epidemic in 2003 was characterized as an epidemic, yet not a pandemic. In 2009, swine flu (H1N1) was recognized as a pandemic, however was accordingly viewed as a gentle form. In 2009, remarks and investigates expressing that "the current epidemic has not eliminated the threat of a more deadly bird flu epidemic in the near future" unmistakably exhibited the significance of pandemic arranging to defeat the infectious illness hole (Maital and Barzani, 2020).

Nowadays, peoples of the world face to new catastrophic which is called "coronavirus". As of the last months of the 2019, the coronavirus has unexpectedly entered people's lives, affecting daily life as well. Covid-19 which emerged in Wuhan city of China caused the death of thousands of people in a short period of time. Corona viruses are a huge group of infections that are known to cause ailment going from the normal virus to more extreme illnesses like Middle East Respiratory Syndrome (MERS) and SARS (WHO, 2020). It is acceptable fact that the virus

causes to spread in other countries other than China as of 11 March 2020. Therefore, covid-19 described as an epidemic disaster by WHO (WHO, 2020). The first coronavirus case in Turkey has been emerged on 11 March 2020. This unexpected epidemic has negative effects in many areas such as health, economic, and psychological. The countries of the world, caught unprepared for the epidemic, develop various policies to cope with the epidemic and to minimize its damage.
Human activity is one of the most significant factors that are effective in the spread of infectious diseases such as coronavirus. For this reason, many governments take similar measures which are related people social life like social distance, lockdown etc. These precautions play a crucial role to decrease the detrimental effects of epidemic. Besides these significant precautions, the governments required accurate and robust predictions related to progress of this devastating pandemic. Thanks to the predictions, managers can have opportunities to take right decisions which have effects related to the social life, using health equipment, economics, and education life and so on. Therefore, in this thesis, we will try to tackle the problem of getting accurate and robust predictions related to progress of coronavirus pandemic by using artificial intelligence techniques. These accurate and robust predictions would enable decision markers to make the right decisions at the right times especially during such together times.

The aim of this thesis is to predict the number of coronavirus case and death in the Turkey. It was performed via boosting algorithms which are adaptive boosting (AdaBoost), categorical boosting (CatBoost), and extreme gradient boosting (XGBoost). In addition, artificial neural network (ANN) algorithm is also utilized to be able to make comparisons between the boosting algorithms and the native ANN algorithm. The estimation was made using real data from Turkey from the TURCOVID19 website data source by Uçar et al. (2020). The main goal is to determine the number of cases and deaths.

The main objective of this thesis is to determine with the artificial intelligence algorithm with the best prediction performance in terms of the considered performance measurements.
The motivation of the study is to get the most accurate and robust predictions by using artificial intelligence techniques. So, this research focuses on determining the best algorithm for predicting Covid-19 in Turkey.

The study consists of six main sections:

- In the first section, a short summary and general information are included.
- The second section contains information about various methods used in the prediction of epidemic diseases in literature.
- The third section contains detailed information about the methods to be used in experimental studies.
- The fourth section includes experimental studies. Information and process steps regarding the input data are explained in this section. Findings are interpreted appropriately.
- The fifth section consists of the results obtained, all the analysis and discussions about the methods, and the contributions of the thesis to the current literature.
- In the sixth and last section, the conclusion and recommendations for future studies are provided.

2. LITERATURE REVIEW

There are many different methods to predict the coronavirus epidemic. In this part of the thesis study, we can look at the background studies which are used different prediction methods. During the years, people have faced with various types of epidemics. Hence, scientists try to develop several models to help predictions of disasters. For instance, one of the epidemic models is Susceptible, Infective and Recover/Removed (SIR) model which was developed by Kermack and McKendrick in 1927. The SIR model is very useful to predict future of epidemics. Lounis and Bagal (2020) conducted a study on prediction in Algeria by using SIR model. They obtained results according to dataset from Johns Hopkins Institute and Algeria health ministry and estimated different parameters such as the number of case, rate of infection, and reproduction number. Lounis and Bagal's findings were replicated in Turkey by Özdinç, Şenel, Öztürkcan, and Akgül, (2020), who performed SIR model to a dataset of 43 days from the beginning of the Covid-19 pandemic. Unlike other work, they observed positive impact of social distancing measures by comparing model outputs. On the other hand, Yadav (2020) preferred Euler's method to solve the SIR model in his study in India. The author aims to find the prevalence rate of the epidemic with the help of the SIR model. In addition, he aimed to find India's daily, monthly or even one year pandemic predictions. Furthermore, the same approach is utilized in six cities of Europe by Nesteruk (2020), who proposed the SIR model to find some reliable estimates for Covid-19. The SIR model is one of the simplest epidemic mathematical models, and many models are derivatives of this simple form. For example, Rajesh, Pai, Roy, Samanta, and Ghosh, (2020), included the number of deaths due to Covid-19 and applied the SIR (D) model on dataset for India. Also, the study incorporates optimistic and pessimistic scenarios which aim to find the effects of lockdown and social distance. Their results show that the measures are effective in reducing the number of deaths. Another model derived from the SIR mathematical model in its basic form is the SEIR model which divides the population into four parts; susceptible, exposed, infectious, and recovered. Pandey, Chaudhary, Gupta, and Pal, (2020) used the SEIR model and regression model in the Covid-19 prediction study. In this paper, both models are used to predict the number of Covid-19 confirmed cases in India and aim at taking right decisions to protect from Covid-19. Also, the SEIR model seems to be more advantageous in two models for which

the RMSLE performance criterion is considered as the main performance measurement. On the other hand, Ala'raj, Majdalawieh, and Nizamuddin, (2021) proposed a SEIRD model with autoregressive integrated moving average (ARIMA) corrections. General purpose of the study is to use SEIRD and ARIMA model to minimize the difference between actual and predicted data to improve classic SEIR model. Also, some parameters of the SEIRD model are estimated using the bass-hopping algorithm. Mean absolute error (MAE), mean squared error (MSE), MLSE, Normalized MAE, and Normalized MSE were used as performance criteria in the study. Liu, Magal, Seydi, and Webb, (2020) used SIR based model in China. In this paper, the authors considered Covid-19 cases in two different forms such as reported Covid-19 cases or unreported Covid-19 cases by government, respectively. Authors aim to show the effects of unreported virus cases on the epidemic prediction. It is reported that total positive cases significantly increase as varying the rate of unreported virus cases. According to the results, if government wants to take the control of epidemic, they should not ignore the unreported coronavirus cases. One of the models used to predict coronavirus behavior is the logistic model. Estimation of disease spread in Honduras using the logistic model and the SIR model was represented by Guerrero, (2020). In the study, it is reported that both models give complete and robust results.

When it comes to forecasting time series, one of the important techniques is ARIMA modeling. One of the preferred techniques in which studies are examined is the ARIMA model. For instance, India by Behl, and Mishra, (2020), the authors performed an analysis of the ten most infected states of India from a different perspective. They utilized SIR model and ARIMA models in their study. With the SIR model, they can predict the state-by-state Covid-19 end date, as well as the possible peak number of infections. They used the ARIMA model to estimate the number of Covid-19 susceptible populations in each state. It is clearly seen from the relevant literature that ARIMA modelling is commonly used for prediction purposes. To illustrate, Tandon, Ranjan, Chakraborty, and Suhag, (2020), performed ARIMA model to forecast near future of Covid-19 in India. They applied an ARIMA model to the time series data of confirmed cases. On the other hand, for short-term forecasting, a study performed by using an ARIMA model in Pakistan by Yousaf, Zahir, Riaz, Hussain, and Shah, (2020). The number of confirmed cases, deaths and recoveries in Pakistan is predicted. Moreover, Ceylan (2020), used the ARIMA model to estimate prevalence of Covid-19 in Italy, Spain, and France which are the most affected

countries of Europe. This study is used three performance criteria which are root mean square error (RMSE), MAE, and, mean absolute percentage error (MAPE). Also, the best model is selected by considering the lowest MAPE values. Sahai, Rath, Sood, and Singh, (2020) considered five countries most affected by Covid-19 in the world by using ARIMA model. In the study, the ARIMA model features were estimated using Hannan and Rissanen algorithms. MAD and MAPE performance criteria are considered for model accuracy. In another similar study, Ilie, Cojocariu, Ciobica, Timofte, Mavroudis, and Doroftei, (2020) utilized ARIMA model to predict the prevalence trend in nine countries which are Ukraine, Romania, Republic of Moldova, Serbia, Bulgaria, Hungary, USA, Brazil, and India. RMSE, MAE, and MAPE are the considered performance measurements in the study. Dehesh, Mardani-Fard, and Dehesh (2020) predicts the confirmed cases using the ARIMA model in China, Italy, South Korea, Iran, and Thailand. A 17-day forecast was made by using the developed model. Kırbaş, Sözen, Tuncer, and Kazancıoğlu, (2020) performed a comparative study using ARIMA, NARNN, and long short-term memory (LSTM) models. The purpose of the study is to predict the total number of coronavirus cases in European countries which are Denmark, Belgium, Germany, France, United Kingdom, Finland, Switzerland, and, Turkey. The best model is determined by using performance criteria such as MSE, PSNR, RMSE, NRMSE, MAPE, and, SMAPE. According the results the prediction performance of LSTM is found to be superior than that of others.

Regression, one of the methods used for estimation, is a popular method in academic studies. As for epidemic diseases, there are many papers that are performed by using regression or derived from regression. For instance, Ogundokun, Lukman, Kibria, Awotunde, and Aladeitan (2020), conducted a study which is used linear regression model to predict the rate of Covid-19 in Nigeria. In addition, the authors aimed to measure travel history and the effect of contact on cases. According to their results, they observed that travel history and contact have an impact on the cases. Another form of regression analysis is multiple linear regressions which is used for predictive analysis. Rath, Tripathy, and Tripathy (2020) applied two regression models which are linear regression and multiple linear regression to estimate the proportion of active cases in India. Menon (2020) applied both logistic growth model and genetic algorithm in their studies. The article aims to predict the number of infected cases due to the coronavirus. On the other hand, Amar, Taha, and Mohamed (2020) conducted their study by using seven different regression-

based models which are exponential polynomial, quadratic, third degree, fourth-degree, fifth-degree, sixth-degree, and logit growth for the Covid-19 dataset in Egypt. Authors aim to estimate the rate of the spread of the coronavirus and how many people will get the virus. Also, these models are beneficial for the Egyptian government to manage the coronavirus in the future. Support vector machine (SVM) is one of the most utilized machine learning algorithms for prediction purposes. When looked at the literature, the SVM model which is based on statistics was performed to get predictions related to coronavirus. To give an example, Parbat, and Chakraborty (2020) applied the SVM model to estimate total recovery people, cumulative cases number, and daily case number. They also used MSE and RMSE as the performance criteria. Again, a study using the SVM model was performed by Singh et al (2020). Researchers aim to estimate confirmed cases of Covid-19. Also, their results indicate that there is a positive relationship between confirmed cases and regular mortality. Among the academic studies, there are a lot of papers which used different models and compared them. Gupta, Singh, Mathur, and Travieso-Gonzalez (2021) conducted a study which consist of SVM, linear regression model, and, prophet forecasting model to estimate the active rate, the death rate, and the cured rate in India. They compare the three models to find out which model is the best predictive model based on the considered performance criteria. MAE, MSE, and RSME are considered as the performance criteria in the study. According to the results they obtained, the prophet forecasting model was determined as the best forecasting model. On the other hand, Ayyoubzadeh, Ayyoubzadeh, Zahedi, Ahmadi, and Kalhori (2020) conducted a prediction study which utilized a linear regression and LSTM models in Iran. The purpose of the investigation is to estimate the incidence of coronavirus in Iran. In addition to daily data, they also benefited from google trends data. They used RMSE as the performance metric.

From the related literature it is apparently seen that artificial intelligence plays an important role in the predictive analysis. In literature various type of artificial intelligence techniques are proposed. ANN is one of the artificial intelligence prediction methods inspired by the human brain. The ANN-based model was applied to estimate the confirmed case of coronavirus by Niazkar and Niazkar (2020). Within the scope of this study, estimates were made for 7 countries (China, Japan, Singapore, Iran, Italy, South Africa, and the USA). The authors proposed 14 different models all based on the ANN. The results revealed that the models that take into

account the incubation period give better results. Study applied in Turkey by Eroglu (2020) is used two methods which are the ANN and LSMT. The proposed models predict coronavirus cases in 7 days before.

The results differ according to forecasting periods depending on their being long-term or short-term. Therefore, Covid-19 estimation related academic studies are consisting of many different methods and different time periods as well. To illustrate, Roosa et al. (2020), conducted a study which shows estimates of coronavirus in China for 5, 10, and 15 consecutive days. Three different models which are generalized logistic growth model, the Richards growth model, and sub-epidemic wave model developed for estimation purpose. This paper indicates short-term estimations of coronavirus. Also, they used MSE as performance criteria. In addition, another paper provides us instant R value in Turkey. Senel, Özdinç, Öztürkcan and Akgül (2020), performed the estimation of Covid-19 by using Bayesian statistical inference. The researchers aim to estimate instantaneous R value which helps preventive protection for coronavirus. Moreover, Chimmula and Zhang (2020) performed a LSTM model to forecast the possible ending time of coronavirus in Canada. RMSE is considered as the performance criterion to be minimized. There are also other studies based on LSTM in the literature. To illustrate, Arora, Kumar, and Panigrahi (2020) proposed LSTM based models which are Deep LSTM, Convolutional LSTM, and Bi-directional LSTM to estimate how many people covid-19 positive reported in India. Also, according to maximum accuracy the best LSTM model is bi-directional LSTM, while convolutional LSTM underperforms in terms of the same measurement. They used MAPE as a performance criterion. Moreover, Wang, Zheng, Ai, Liu, and, Zhu (2020) performed the LSTM model which uses a rolling update mechanism approach to predict during the 150 days in Russia, Peru, and, Iran. It is one example of long-term prediction. With a rolling update mechanism, they can use the prediction data for input data. Also, this study uses a diffusion index (DI) to measure the effects of precautions on coronavirus spread. Another study that uses LSTM was conducted in India by Tomar, and Gupta (2020). It is aimed to predict the positive number of cases of coronavirus in India and to measure the effects of precautions. They test the effects of measure by changing transmission rate which affected by social distancing and lockdown. This paper highlights the importance of precautions. Other than these studies, one of the papers focuses on the statistical analysis. Al-Rousan, and Al-Najjar (2020) applied statistical analysis

that tries to find the effects of different factors which are sex, region, infection reasons, birth year, and, released or diseased date on Covid-19 in South Korea. According to their results, these factors affect just deaths cases. There are plenty of methods based on statistics for data analysis and help understand the data. One of the methods is exploratory data analysis (EDA) which provides us to describe main features in dataset. It is performed for evaluating current status of coronavirus. Dey, Rahman, Siddiqi, and Howlader (2020) performed EDA and visual exploratory data analysis in China for getting better understanding of different Covid-19 scenarios.

Highly developed software technology enabled researchers to use various kinds of estimation tools and techniques. In literature there exist plenty of papers which utilize deep learning approach. Zhu et al. (2020) used deep learning algorithm to determine the most significant variables to estimate number of deaths accurately. They used 181 confirmed Covid-19 patients' data from China. The most important clinical elements determined among the independent variables utilizing the deep learning neural network algorithm. The purpose of identifying these clinical factors is to assist in determining the number of deaths later on.

In today's digitalized world, it is undeniable fact that artificial intelligence plays an important role in our daily live. Artificial intelligence techniques enable quick estimations and analysis in many different areas. After the coronavirus pandemic, a tremendous increase has been observed in the studies on coronavirus using artificial intelligence techniques. Artificial intelligence is known to a subset of machine learning. Also, there are various types of algorithms which are subset of machine learning. Moreover, each algorithm analyzes the data with different approaches. For example, boosting algorithms which are one of the popular machine learning algorithms, are known to have very good predictive performances. These algorithms gained a widespread use in academic studies since their ability to reduce the rate of errors significantly. XGBoost, GradientBoost, AdaBoost, and CatBoost algorithms are the most known and used boosting algorithms. There are lots of studies that apply these algorithms that contribute to Covid-19 related predictions. In addition, different methods that are applied together with boosting algorithms exist in the literature. For example, Luo, Zhang, Fu, and Rao (2021) conducted a study to estimate daily confirmed infected cases in Amerika. The authors used two methods which are the LSTM and XGBoost algorithm. Also, MAE, MSE, RMSE, and MAPE are

considered as the performance criteria to evaluate the efficiency of model. On the other hand, Iwendi et al. (2020) proposed a random forest model supported by the AdaBoost algorithm. Results revealed that, positive relationships exist between patient's sex and deaths, the majority of patients are aged between 20 and 70. Another paper which is based on XGBoost machine learning algorithm suggests a model that predicts the risk of mortality in patients in Wuhan by Yan et al. (2020). In the study, Multi-tree XGBoost is used to determine the key clinical characteristics of the patients. Aljameel, Khan, Aslam, Aljabri, and Alsulmi (2021) performed a study to determine which clinical factors are more effective on mortality or survival in Saudi Arabia. In the study, logistic regression, random forest, and XGBoost are used for classification purposes. Gumaei, Al-Rakhami, Al Rahhal, Albogamy, Al Maghayreh, and AlSalman (2021) used gradient boosting regression method to predict confirmed cases of Covid-19. Authors performed three different methods which are extreme gradient boosting regression, support vector regression, and decision tree regression to compare with gradient boosting regression. The model is trained using 10-fold cross-validation for all models. Results revealed that gradient boosting regression outperforms the others in terms of RMSE, MAE, and coefficient of determination.

To date, many studies have been carried out in the literature on the prediction and determination of the current status of Covid-19. Researchers used historical data and developed different type of models for estimating the number of cases, deaths etc. Then, they used these estimations for improving the decision making process at that tough times. In this thesis, boosting algorithms are used to forecast the number of coronavirus cases and deaths in Turkey. Boosting algorithms are used in various fields as well. For example, Liu, Wu, Liu, Li, Hu, and Li (2021) predicted mortality rate of patients who have acute kidney injury using XGBoost algorithm. In the study, logistic regression, SVM, and random forest machine learning algorithms are utilized other than XGBoost for comparative purposes. All these models' performance is measured in terms of accuracy, precision, recall, and F1 score. According to the results, XGBoost outperforms the others in terms of considered performance metrics. Osman, Ahmed, Chow, Huang, and El-Shafie (2021) predicted the groundwater levels in Selangor Malaysia. In the study, XGBoost, ANN, and SVM are used. MAE, RMSE and, coefficient of determination (R^2) are used as the performance measurements. The results revealed that XGBoost outperforms to the others in terms of the

considered performance measurements. Al-Rakhami, Gumaei, Alsanad, Alamri, and Hassan (2019) suggested a prediction model to predict energy load in residential buildings using XGBoost algorithm. In the study, RMSE, R^2, MAE, and MAPE values are considered as the performance criteria.

3. MATERIALS AND METHODS

The world struggles with the Covid-19 pandemic in many areas these days. In order to overcome the process with the least damage, analyzes are carried out using the Covid-19 data. In this study, we performed predictive analyzes using Turkey coronavirus data. In this section, details about artificial intelligence, machine learning and ANN, AdaBoost, CatBoost, and XGBoost algorithms which are utilized to make Covid-19 related predictions.

3.1. Artificial Intelligence

Artificial intelligence is a computer-based or digitally controlled system that can perform the work of intelligent beings (Copeland, B. 2020). Artificial intelligence is very comprehensive and large-scale. The three types of learning strategies will be briefly discussed: supervised learning, unsupervised learning, and reinforcement learning.

Supervised Learning: One of the most popular uses of machine learning in healthcare systems is supervised learning. This learning method analyzes through the learning strategy in recognizing similar features and uses the information to find precise results (Hussain, Bouachir, Al-Turjman, & Aloqaily, 2020). Algorithms such as linear regression, random forest, SVM etc. are included in the supervised learning method.

Unsupervised Learning: No information or sign is used in this technique. This method finds the hidden structures of the data and divides it into close groups. It is a promising type of prediction (Hussain et al., 2020). K-means clustering, k-nearest neighbor (k-NN), Hierarchical clustering etc. are included in the unsupervised learning method.

Reinforcement Learning: “This learning technique is neither of the two stated above; it is, considerably progressive, a kind of a crossbreed approach. In this technique, there is an alternate or single authority that makes sense of acceptable behavior in a circumstance with the end goal that they endeavor to increase their immense total prize or score.” (Hussain et al., 2020).

3.2. Machine Learning

Machine learning is one of the developing branches of computational algorithms. It is designed to pretend human intelligence by learning from the environment (El Naqa & Murphy, 2015). Using machine learning computer algorithms, it learns the relationships between different data items to find results (Zhu et al. 2020).

In time, machine learning methods are utilized in many different areas and increased in popularity. One of these areas is health services. The inclusion of smart devices in health services enabled gathering data in different areas and making predictions with this data of the patients. Especially these days when the world is fighting with the coronavirus, many methods of machine learning have been benefited. We will briefly mention about the ensemble method, which is only one of these methods.

Ensemble Methods: Classical learners try to build one learner from training data. On the other hand, the ensemble learner approach creates a group of learners and combines them to solve the same problem. Additionally, it is named ensemble learning, committee-based learning, or multiple classifier systems (Zhou, 2019). To illustrate, boosting is one of the algorithms based on ensemble learning.

3.3. Artificial Neural Networks

ANNs, inspired by the neural networks of the human brain, are used for prediction, optimization, etc. (Jain, Mao, & Mohiuddin, 1996). It is one of the machine learning algorithms that solves plenty of problems. This algorithm can learn from input data and then gives predictions or classifications as an output.

As mentioned in the above definition of ANN, it is one of the foremost machine learning algorithms. In literature, machine learning algorithms are included in the artificial intelligence set. Figure 3.1 illustrates this structure of subsets.

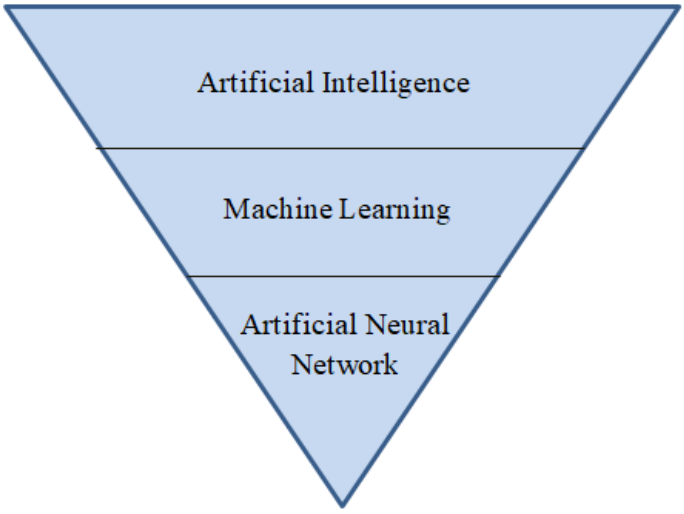

Figure 3. 1 Structure of subsets

The working principle of ANNs is similar to the biological neuron structure from which it is inspired. The human brain is made up of numerous numbers of interconnected neurons that form a network. Each neuron, like a cell performs a simple duty; responding to an incoming signal. Algorithm consists of layers and these layers depend on the type of ANN (Alaloul & Qureshi, 2020). Figure 3.1 shows basic structure of ANN. The structure consisting of three basic layers which are input layer, hidden layer and output layer which is the general architecture of the ANN. Also, these layers have connections between them.

Input Layer: this layer is the starting point of the algorithm; it is the entry point of the data which is called input data. This data which is to be used as an input to the network can be in various forms such as texts, numbers, sound files, image pixels, etc.. No changes are made on this input data; it is copied and sent to the hidden layer or layers (the number of hidden layers can change with respect to the type of ANN).

Hidden Layer: the number of hidden layers which is located between input layer and output layer can be single or more. Transformation of the data transferred by copying from the input layer takes place in the hidden layer. Therefore, hidden layer plays a crucial role in the analysis to obtain more accurate result.

Output Layer: it converts the link it receives from the hidden layer into an output variable.

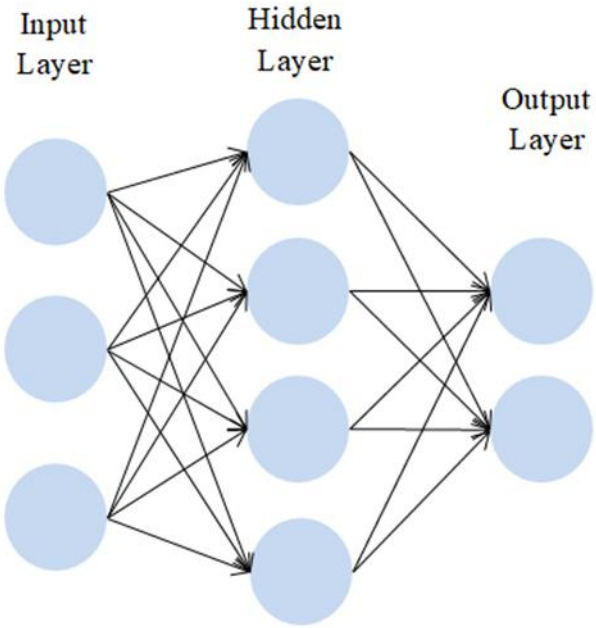

Figure 3. 2 Layers of ANN

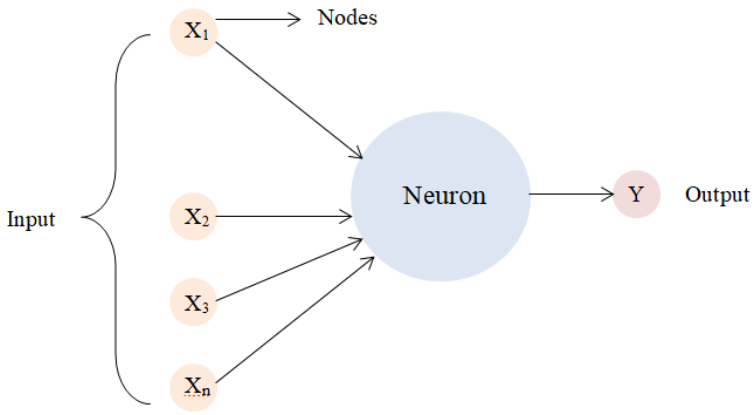

Figure 3. 3 Basic structure of neuron

3.3.1. Types of ANNs

Types of ANNs have some strengths and weaknesses depending on where they are used. In general, two basic types of ANNs are utilized. They are separated according to the shape of transmission between the layers.

3.3.1.1. Feedforward ANNs

The feedforward neural network is the first and most basic type of ANN to be created. There is no loop in this network; just forward transmission occurs between layers.

3.3.1.2. Feedback ANNs

Nodes in a feedback network have backward connected loops, and the output of the nodes might be the input to the same level or preceding nodes in these connections.

3.3.2. Data Training in ANNs

When the literature is examined, there are three basic learning types to train data in ANNs. These are named supervised learning, unsupervised learning, and reinforcement learning.

3.3.2.1. Supervised Learning

It is used to train the network associated with the input, output, and goal or intended model in supervised learning. The expected correct output determines the errors that can be used to change network parameters to assist improve performance during a comparison between the computed network and the expected correct output during a comparison between the calculated networks.

3.3.2.2. Unsupervised Learning

This learning strategy does not offer the output, which is the aim. Because it is anticipated that there would be no teacher to give the desired training in unsupervised learning, it learns by self-learning by discovering the characteristics and structures in the patterns that are provided as input to the system.

3.3.2.3. Reinforcement Learning

In reinforcement learning, the teacher will not offer the expected response, but will signal when it is computed and whether it is accurate or erroneous, while the information provided will aid the network in learning. A correct answer is rewarded, whereas a poor answer incurs a penalty. Reinforced learning is the least effective of all the learning methods (Gupta, Salau, Chaturvedi, Akinola, & Nwulu, 2019).

ANN algorithm has become popular in the academic area for different type of analysis. The reason behind this popularity is the algorithm which has several significant advantages.

General advantages of ANN:

- Information is saved on the entire network, rather than in a database, as it is in traditional programming. Thanks to this advantage of the algorithm, the network continues to function despite the loss of a few pieces of information in one location.
- The algorithm can work with incomplete information. The lack of performance here may vary depending on the importance of the incomplete information.
- ANNs have fault tolerance. The deterioration of a certain amount of cells in the algorithm structure does not prevent the algorithm from producing results. The networks become fault resistant as a result of this characteristic.
- Having a distributed memory: in order for an ANN to learn, the examples must be determined and the network must be taught according to the desired output by giving these examples to the network. The network's success is proportional to the instances chosen, and if the event cannot be shown to the network in all of its dimensions, the network may generate erroneous results.
- Gradual corruption occurs when a network slows down and degrades over time. The network fault does not appear to be corroding right away.
- Machine learning capability: ANNs examine events, learned from them and make decisions based on them.
- ANNs can parallel processing. ANNs have the numerical energy to handle multiple tasks simultaneously (Mijwel, 2018).

3.4. Ensemble Learning

An ensemble is made up of a group of learners known as base learners. An ensemble's generalization ability is usually substantially higher than that of base learners. Ensemble learning is appealing because it can elevate poor learners who are only marginally better than a random guess to strong learners who can make extremely precise predictions (Zhou, 2019). As a result, "weak learners" are often referred to as "base learners."

3.4.1. Ensemble Methods

There are three basic types of ensemble learning methods in literature. These are named bagging, stacking, and boosting. In this part, bagging and stacking are explained briefly. Also, boosting algorithms will be explained in detail since these types of algorithms are used as one of the main models in this study.

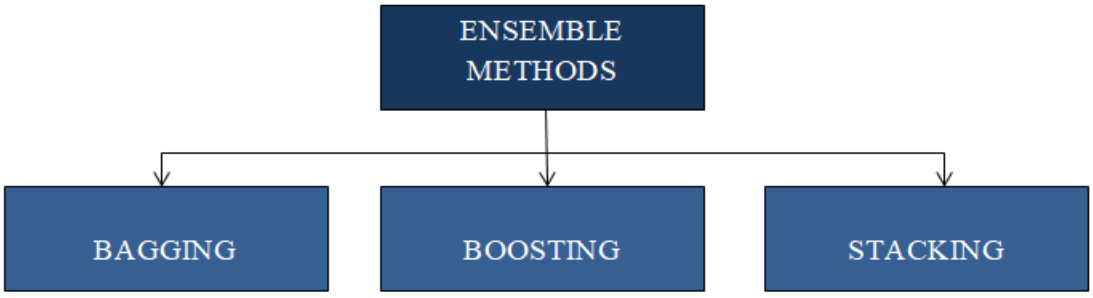

Figure 3. 4 Basic types of ensemble methods

Bagging: bagging predictors is a technique for creating numerous versions of a predictor and then combining them into a single aggregated prediction. When forecasting a numerical conclusion, the aggregate takes an average of the versions, and when predicting a class, it uses a plurality vote. Making bootstrap duplicates of the learning set and using these as new learning sets creates many versions. Bagging has been shown to improve accuracy in actual and simulated data sets utilizing classification and regression trees, as well as subset selection in linear regression (Breiman, 1996).

The pseudo-code of bagging algorithm, whose definition is given above, is shown in figure 3.5 below.

Input: Data set $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$;

Base learning algorithm L;

Number of learning rounds T.

Process:

for $t = 1, \dots, T$:

$D_t = Bootstrap(D)$; % Generate a bootstrap sample from D

$h_t = L(D_t)$ % Train a base learner h_t from the bootstrap sample

end.

Output: $H(x) = argmax_{y \in Y} \sum_{t=1}^{T} 1(y = h_t(x))$ % the value of 1(α) is 1 if α is *true* and 0 otherwise

---

Figure 3. 5 Pseudocode of the Bagging algorithm

Stacking: stacking, which frequently takes into account diverse weak learners, learns them in parallel and then combines them by training a meta-model to output a prediction based on the predictions of the various weak models (Rocca, 2019).

Input: Data set $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$;

First-level learning algorithms $L_1, \dots, L_T$;

Second-level learning algorithm L.

Process:

for $t = 1, \dots, T$:

$h_t = L_t(D)$ %Train a first-level individual learner h_t by applying the first-level

end; %learning algorithm L_t to the original data set D

$D' = \emptyset$; %Generate a new data set

for $i = 1, \dots, m$:

for $t = 1, \dots, T$:

$z_{it} = h_t(x_i)$ % Use h_t to classify the training example x_i

end;

$D' = D' \cup \{((z_{i1}, z_{i2}, \dots, z_{iT}), y_i)\}$

end;

$h' = L(D')$. % Train the second-level learner h' by applying the second-level

% learning algorithm L to the new data set D'

Output: $H(x) = h'(h_1(x), \dots, h_T(x))$

Figure 3. 6 The Stacking algorithm

3.4.2. Boosting Algorithms

Boosting is one of the categories of machine learning. The logic behind this algorithm is the idea that a weak classifier can show more success than any classifier (Ferreira & Figueiredo, 2012). It is aimed to do this by applying this method by giving more weight while training weak learners. In today, this algorithm gives very successful results in both classification and estimation.

Given a data sample distribution D. Determine the total number of base models as M:

Define the initial training sample distribution as $D_1 = D$

For $m = 1\ to\ M\ do$

Train a base model $B_m(x)$ from the training sample distribution D_m.

Compute the error of the model.

Adjust the distribution D_m. to D_{m+1}. to make the mistake of the model more evident.

Output the constructed base model $B_m(x)$.

End;

Output the prediction of the ensemble trees for a given new input x: $\frac{1}{M}\sum_{m=1}^{M} B_m(x)$;

Figure 3. 7 The Boosting algorithm (Zhang & Haghani, 2015)

Advantages of Boosting Algorithms

- The boosting technique accounts for the weighting of the higher and lower accuracy samples before combining the findings.
- Each learning step is assessed for net error. Interactions are well-suited to it.
- When dealing with bias or under fitting in a data set, the boosting techniques comes in handy.
- There are a variety of methods for boosting. To illustrate: AdaBoost, XGBoost, GradientBoost (Ragini, 2019).

3.4.3. AdaBoost (Adaptive Boosting) Algorithm

AdaBoost algorithm is the oldest of the boosting algorithms and is the short for Adaptive Boosting. The AdaBoost algorithm, introduced in 1995 by Freund and Schapire, solved many of the practical difficulties of the earlier boosting algorithms.

Figure 3.8 illustrates the iterations of the algorithm.

Algorithm: *AdaBoost.R*

Input: sequence of N examples $(x_1, y_1) \ldots, (x_N, y_N)$ with labels $y_i \in Y = [0,1]$

distribution D over the examples

weak learning algorithm **WeakLearn**

integer T specifying number of iterations

Initialize the weight vector:

$$w_{i,y}^1 = \frac{D_{(i)}\,|y - y_i|}{Z}$$

for $i = 1, \ldots, N, y \in Y$, where

$$Z = \sum_{i=1}^{N} D_{(i)} \int_0^1 |y - y_i|\, dy.$$

Do for $t = 1,2, \ldots, T$

1. Set

$$\boldsymbol{p^t} = \frac{\boldsymbol{w^t}}{\sum_{i=1}^{N} \int_0^1 w_{i,y}^t dy}.$$

2. Call **WeakLearn,** providing it with the density p^t; get back a hypothesis $h_t: X \times Y$.
3. Calculate the loss of h_t:

$$\varepsilon_t = \sum_{i=1}^{N} \left| \int_{y_i}^{h_t(x_i)} p_{i,y}^t \, dy \right|.$$

If $\varepsilon_t > 1/2$, then set $T = t - 1$ and abort the loop.

4. Set $\beta_t = \varepsilon_t/(1 - \varepsilon_t)$.
5. Set the new weights vector to be

$$W_{i,y}^{t+1} = \begin{cases} W_{i,y}^t & if\ y_i \le y \le h_t(x_i)\ or\ h_t(x_i) \le y \le y_i \\ W_{i,y}^t \beta_t & otherwise. \end{cases}$$

for $i = 1, \ldots, N, y \in Y$.

Output the hypothesis

$$h_f(x) = \inf \left\{ y \in Y: \sum_{t:\, h_t(x) \le y} \log\left(\frac{1}{\beta_t}\right) \ge \frac{1}{2} \sum_t \log\left(\frac{1}{\beta_t}\right) \right\}.$$

Figure 3. 8 Pseudocode of the AdaBoost.R algorithm (Freund & Schapire, 1997)

The purpose of the AdaBoost.R algorithm, whose formula is given above, is to find the $h: X \rightarrow Y$ hypothesis, which gives an approximate y value when an x value is given.

Looking at the steps of the algorithm in general;

In step 1, density $\boldsymbol{p^t}$ is defined by normalizing the weights $\boldsymbol{w^t}$.

In step 2, it is provided to the weak learner.

The weak learner's purpose is to reduce the ε_t defined in step 3 to lowest possible value.

Finally, the weights are updated as needed in step 5.

General advantages of AdaBoost algorithm:

- It is quite handy and also simple to program.
- The algorithm has no parameters which need to adjust (exception of T round number).
- It can be combined with a different method to find the weak hypothesis due to do not need foreknowledge. (Freund et al., 1999).

3.4.4. CatBoost (Categorical Boosting) Algorithm

CatBoost processes categorical data and is much more accomplished than other open-source gradient boosting applications (Dorogush, Ershov, & Gulin, 2018).

Algorithm: *CatBoost Algorithm*

Input : $\{(x_i, y_i)\}_{i=1}^{n}$, $\boldsymbol{I}$, *α, L, s, Mode*

$\sigma_r \leftarrow$ random permutation of $[1, n]$ for $r = 0..s$;

$M_0(i) \leftarrow 0$ for $i = 1..n$;

if *Mode = Plain* **then**

    $M_r(i) \leftarrow 0 \; for \; r = 1..s, i : \; \sigma_r(i) \; \leq \; 2^{j+1}$;

if *Mode = Ordered* **then**

    for $j \leftarrow 1$ **to** $[\log_2 n]$ **do**

        $M_{r,j}(i) \leftarrow 0$ for $r = 1..s, i = 1..2^{j+1}$;

for $t \leftarrow 1$ **to** I **do**

    $T_t, \{M_r\}_{r=1}^{s} \leftarrow BuildTree(\{M_r\}_{r=1}^{s}, \{(x_i, y_i)\}_{i=1}^{n}, \alpha, L, \{\sigma_i\}_{i=1}^{s}, Mode)$;

    $leaf_0(i) \leftarrow GetLeaf(x_i, T_t, \sigma_0)$ for $i = 1..n$;

    $grad_0 \leftarrow CalcGradient(L, M_0, y)$;

    foreach *leaf j in* T_t ***do***

        $b_j^t \leftarrow -avg(grad_0(i)$ for $i : \; leaf_0(i) = j)$;

    $M_0(i) \leftarrow \; M_0(i) + \; \alpha b_{leaf_0(i)}^t$ for $i \; = 1..n$;

return $F(x) \; = \sum_{t=1}^{I} \sum_j \alpha b_j^t \; 1_{\{GetLeaf(x, T_t, ApplyMode) = j\}}$;

Figure 3. 9 Pseudocode of the CatBoost algorithm

The above algorithm illustrates step-by-step CatBoost procedure (Prokhorenkova, Gusev, Vorobev, Dorogush, & Gulin, 2017). Furthermore, the algorithm structure of BuildTree, which is an important function in the CatBoost method, has been detailed.

Function *BuildTree*

Input : $M, \{(x_i, y_i)\}_{i=1}^{n}, \alpha, L, \{\sigma_i\}_{i=1}^{s}, Mode$

$grad \leftarrow CalcGradient(L, M, y)$;
$r \leftarrow random(1, s)$;
if $Mode = Plain$ **then**
    $G \leftarrow (grad_r(i)$ for $i = 1..n)$;
if $Mode = Ordered$ **then**
    $G \leftarrow \left(grad_{r,\lfloor \log_2(\sigma_r(i)-1) \rfloor}(i)\right.$ for $\left. i = 1..n\right)$;
$T \leftarrow empty\ tree$;
foreach $step\ of\ topdown\ procedure$ **do**
    foreach $candidate\ split\ c$ **do**
        $T_c \leftarrow$ add split c to T;
        $leaf_r(i) \leftarrow GetLeaf(x_i, T_c, \sigma_r)$ for $i = 1..n$;
        if $Mode = Plain$ **then**
            $\Delta(i) \leftarrow avg\left(grad_r(p)\ for\ p : leaf_r(p) = leaf_r(i)\right)$ for $i = 1..n$;
        if $Mode = Ordered$ **then**
            $\Delta(i) \leftarrow avg\left(grad_{r\lfloor \log_2(\sigma_r(i)-1) \rfloor}(p)\right.$ for p: $\left. leaf_r(p) = leaf_r(i), \sigma_r(p) < \sigma_r(i)\right)$ for $i = 1..n$;
        $loss(T_c) \leftarrow \cos(\Delta, G)$
    $T \leftarrow \text{argmin}_{T_c}\left(loss(T_c)\right)$
$leaf_{r'}(i) \leftarrow GetLeaf(x_i, T, \sigma_{r'})\ for\ r' = 1..s,\ i = 1..n$;
if $Mode = Plain$ **then**
    $M_{r'}(i) \leftarrow M_{r'}(i) - \alpha\ avg\left(grad_{r'}(p)\ for\ p : leaf_{r'}(p) = leaf_{r'}(i)\right)\ for\ r' = 1..s, i = 1..n$;
if $Mode = Ordered$ **then**
    for $j \leftarrow 1$ **to** $\lceil \log_2 n \rceil$ **do**
        $M_{r',j}(i) \leftarrow M_{r',j}(i) - \alpha\ avg\left(grad_{r',j}(p)\right.$ for $p : leaf_{r'}(p) = leaf_{r'}(i), \sigma_{r'}(p) \leq \left. 2^j\right)$ for $r' = 1..s, i : \sigma_{r'}(i) \leq 2^{j+1}$;
return T, M

Figure 3. 10 The BuildTree function

Advantages of CatBoost algorithm:

- CatBoost produces cutting-edge results that compete with any major machine learning algorithm in terms of performance.

- To transform categories into numbers, CatBoost can be used without any prior preprocessing. CatBoost uses multiple statistics on categorical and numerical aspects to transform categorical values to numbers.
- It eliminates the need for intensive hyper-parameter adjustment and decreases the risk of overfitting, resulting in more generic models. CatBoost, on the other hand, includes a number of parameters to tune, including the number of trees, learning rate, regularization, tree depth, fold size, bagging temperature, and others (Ray, 2017).

3.4.5. XGBoost (Extreme Gradient Boosting) Algorithm

XGBoost, is machine learning algorithm that is defined as a scalable tree boosting system. XGBoost has a scalability feature which is one of the main benefits (Chen, & Guestrin, 2016).

Algorithm : *A generic unregularized xgboost*

Input: training set $\{(x_i, y_i)\}_{i=1}^{N}$, a differentiable loss function $L\big(y, F(x)\big)$, a number of weak learners M and a learning rate α.

1. Initialize model with a constant value.

$$\hat{f}_{(0)}(x) = arg_{\theta} min \sum_{i=1}^{N} L(y_i, \theta).$$

2. For m = 1 to M:
    1. Compute the 'gradients' and 'hessians':

$$\hat{g}_m(x_i) = [\frac{\partial L\big(y_i, f(x_i)\big)}{\partial f(x_i)}]_{f(x)=\hat{f}_{(m-1)}(x)}.$$

$$\hat{h}_m(x_i) = [\frac{\partial^2 L\big(y_i, f(x_i)\big)}{\partial f(x_i)^2}]_{f(x)=\hat{f}_{(m-1)}(x)}.$$

    2. Fit a base learner (or weak learner, e.g. tree) using the training set $\{x_i, -\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\}_{i=1}^{N}$ by solving the optimization problem below:

$$\hat{\varphi}_m = argmin \sum_{i=1}^{N} \frac{1}{2} \hat{h}_m(x_i) \; [- \frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)} - \varphi(x_i)]^2.$$

$$\hat{f}_m(x) = \alpha \hat{\varphi}_m(x).$$

    3. Update the model:

$$\hat{f}_m(x) = \hat{f}_{(m-1)}(x) + \hat{f}_m(x).$$

3. **Output**

$$\hat{f}(x) = \hat{f}_{(M)}(x) = \sum_{m=0}^{M} \hat{f}_m(x).$$

Figure 3. 11 The XGBoost algorithm ("XGBoost", 2019)

XGBoost has many advantages and some of them are given below;

- It is quite adaptable.
- It takes advantages of parallel processing.
- It is less time consuming than Gradient Boosting.
- It is designed to deal with missing data.
- After each iteration, the user can perform a cross-validation.

3.5. Error Measures

The performance criteria which are used to evaluate the performance of predictions algorithms are given below.

A_j - Actual values; $\bar{A}$ - the mean of the actual values; P_j - predicted values;

Error = $A_j - P_j = e_j$

n – Size of the data set

The explained variance score (EVS) is used to calculate the ratio of error variance to true value variance. This score also indicates how effectively the model can explain variations in the dataset. The EVS metric is calculated by the formula given in equation 1 below.

$$EVS = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(e_j - \bar{e}_j\right)^2} \tag{1}$$

The MAE calculates the average of the error distances between each true and predicted value. In this metric, negatively oriented score, that is, predictors with lower values, perform better. The MAE metric is calculated by the formula given in equation 2 below.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}\left|e_j\right| \tag{2}$$

The MAPE calculates the mean absolute percent error in a model. It is one of the basic metrics often used to compare model accuracy. The MAPE metric is calculated by the formula given in equation 3 below.

$$MAPE = \frac{100}{n}\sum_{j}^{n}\frac{\left|e_j\right|}{\left|A_j\right|} \tag{3}$$

The MSE measures the measurement performance of the model predictor. It is always positive value and it is preferable if the value is as low as possible. The MSE metric is calculated by the formula given in equation 4 below.

$$MSE = \frac{1}{n}\sum_{j=1}^{n} e_j^2 \tag{4}$$

The mean squared logarithmic error (MSLE) is a measurement of the difference between true and predicted values. The percentage difference is all that matters to MSLE. This metric is calculated by the formula given in equation 5 below.

$$L(A,P) = \frac{1}{n}\sum_{i=0}^{n}\left(\log\left(A_j + 1\right) - \log\left(P_j + 1\right)\right)^2 \tag{5}$$

The RMSE gives an absolute number of how the predicted results differ from the actual number. Negatively oriented scores, that is, predictors with lower values, perform better. The RMSE metric is calculated by the formula given in equation 6 below.

$$RMSE = \sqrt{\frac{\sum_{j=1}^{n} e_j^2}{n}} \tag{6}$$

The R^2 Score is a metric for how well a model can predict future values. The R^2 score metric is calculated by the formula given in equation 7 below.

$$R^2(A,P) = 1 - \frac{\sum_{i=0}^{n-1}\left(A_j - P_j\right)^2}{\sum_{i=0}^{n-1}\left(A_j - \overline{A}\right)^2} \tag{7}$$

The coefficient of variation (CV) is a statistical measure of a data series' relative distribution around the mean. The coefficient of determination value has no units. The CV metric is calculated by the formula given in equation 8 below.

$$CV = \frac{\sigma}{\mu} \tag{8}$$

4. RESULTS AND DISCUSSION

As mentioned in the previous sections four algorithms which are AdaBoost, CatBoost, XGBoost, and ANNs are utilized on the coronavirus data of Turkey. All the algorithms are compared to each other with respect to considered performance measurements. Also, note that grid search optimization algorithm is used for optimizing the hyperparametres of ANN. The analysis performed on Turkey data set obtained from the TURCOVID19 website by Uçar et al. (2020). In addition, the data set is randomly divided into 80% train set and 20% test set for all analysis. All analyzes in this thesis were performed on a computer with Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz 16.0 GB RAM.

4.1. Description of Turkey Coronavirus Dataset

The first official coronavirus patient in Turkey was reported on 11 March 2020. Therefore, our analysis was carried out with the data obtained as of 11 March. The data set includes the number of patients, the number of cases, and the number of deaths. Individuals who are evaluated as patient are those who are treated at home or in a health institution that show symptoms of illness as well as being positive. Every person with a positive PCR test result is accepted as a case by the Turkish Ministry of Health. Although individuals who are considered as cases do not show symptoms, they infect the virus to their environment. The number of cases started to be announced by the Ministry of Health as of 25 November 2020. In this thesis, the number of patients was ignored in order to evaluate the performed analyzes more accurately. Data ranges and descriptive statistics are summarized in the table below. Also, Skewness and kurtosis values for daily number of patients, daily number of cases, and cumulative number of cases, daily death, and cumulative death data are shown in the Table 4.1.

Table 4. 1 Data description and data range

	Daily patient	New cases	Total cases	New deaths	Total deaths
Count	481	338	338	591	591
Mean	1780.11	19845.61	3246381.1	118.43	25936.74
Std	1433.1	13299.65	1862983.34	90.05	21227.94
Min	0	4418	28351	1	1
25%	856	7902.25	1357452.25	50.5	5882
50%	1370	18852	3824512	86	22070
75%	2253	28216	4586992.75	193	47825
Max	7381	63082	6707818	394	69998
Skewness	1.8032	1.1028	0.00041	0.7602	0.4794
Kurtosis	3.1621	1.1330	-1.2654	-0.4237	-1.1860
Data range	11.3.2020 4.7.2021	25.11.2020 28.10.2021	25.11.2020 28.10.2021	17.3.2020 28.10.2021	17.3.2020 28.10.2021

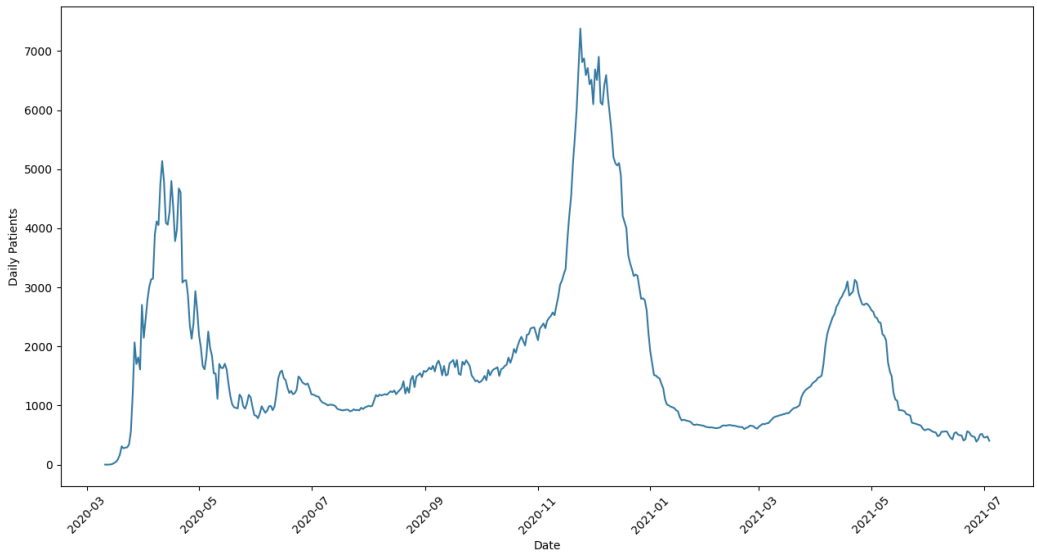

Figure 4. 1 Daily patients

The figure 4.1 illustrates the trend in the number of daily patients in Turkey between 11 March 2020 and 4 July 2021. The number of patients per day until 4 July 2021 was announced by the Ministry. It seems that the coronavirus has an upward trend from its start date. The coronavirus reached its first peak level month of April in the Turkey. After that, with the effect of the measures taken such as lockdown, wearing mask etc., the number of patients has started to fluctuate in a decreasing trend. Considering the decreasing trend, the normalization process has started in the country. With the arrival of the summer months, the reopening of businesses such as cafes, gyms, hairdressers, and the relaxation of the measures, the epidemic has started to rise again, as can be seen in the figure 4.1. Then it reached the 2nd peak again, then went into a rapidly decreasing trend, after being stagnant for a while, it reached the 3rd peak and went into a fluctuating decline.

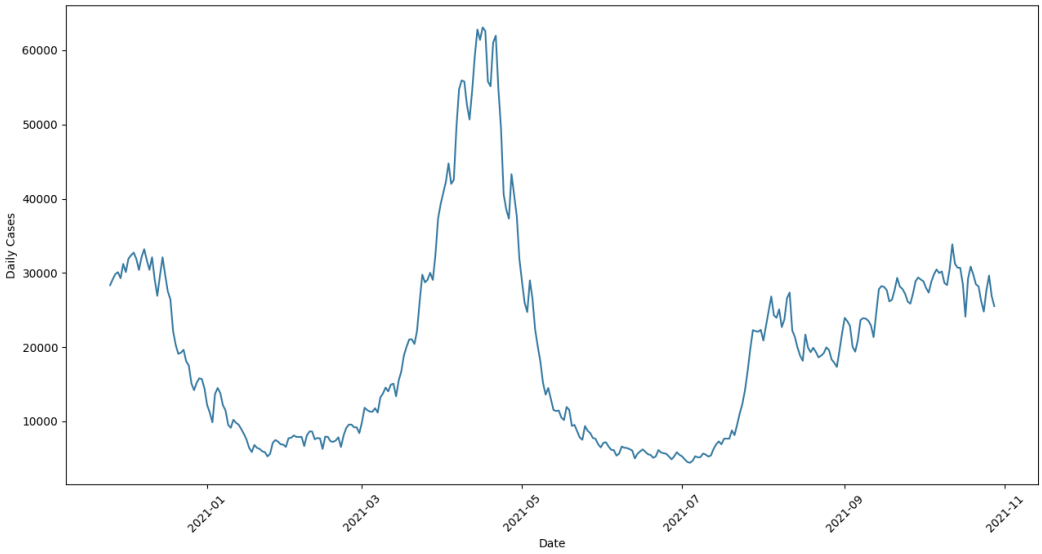

Figure 4. 2 Daily cases

The number of daily cases was shown on the figure 4.2 between 25 November 2020 and 28 October 2021 in Turkey. Since the number of cases directly affects the infection rate, the effects of the measures taken by the ministry can be seen in the figure, for example, a nationwide shutdown was announced between 29 April 2021 and 17 May 2021, followed by a downward trend in the number of cases.

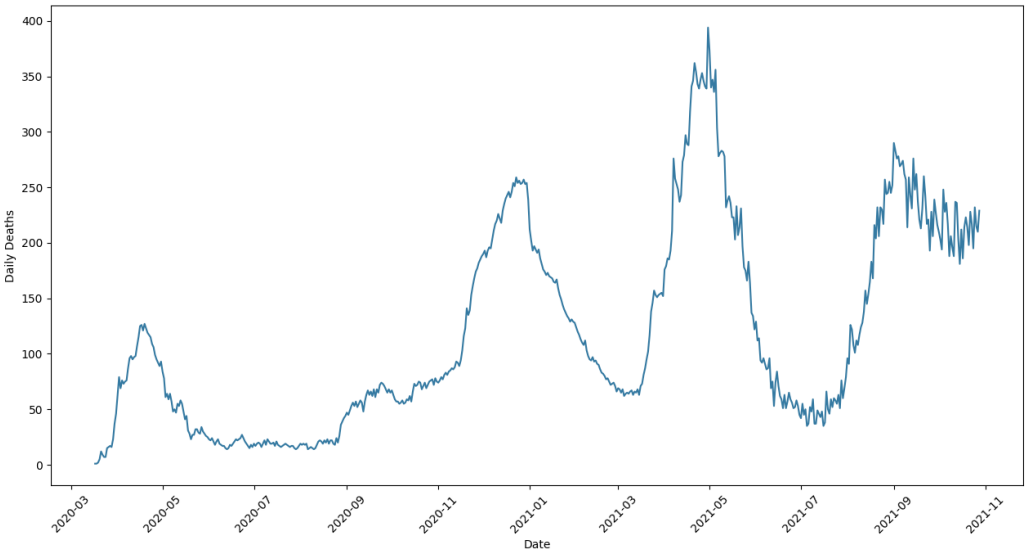

Figure 4. 3 Daily deaths

Figure 4.3 provides information about on the daily deaths trend between 17 March 2020 and 28 October 2021 in Turkey. Turkey reported its first coronavirus death on 17 March 2020. The daily death trend made an increasing start and reached the peak of the first wave and then followed a fluctuating decrease. This nonlinear trend then reached wave 2 and 3 peaks and the fluctuation continues.

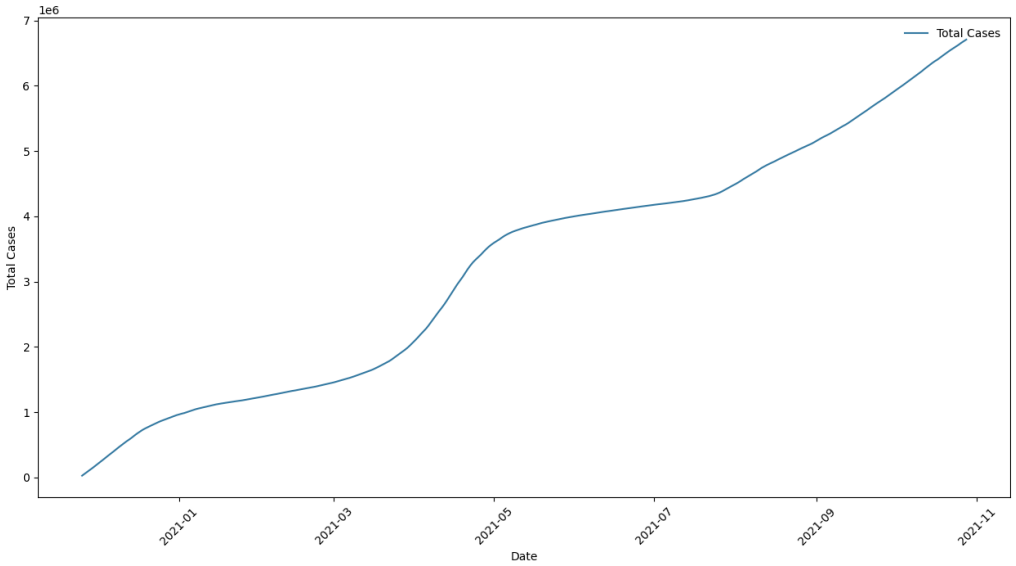

Figure 4. 4 Total cases

The total number of cases in Turkey was shown on the figure 4.4 between 25 November 2020 and 28 October 2021. In general, there is an upward trend. The trend started linearly and continued nonlinearly for a while in the middle of 2021, and then it had an increasing linear trend again.

4.2. Prediction with AdaBoost Algorithm

Firstly, the results of AdaBoost algorithm are discussed. In this analysis not only daily data but also cumulative data were evaluated. According to the obtained results, the performance of AdaBoost will be evaluated with respect to predetermined performance criterions.

4.2.1. Predicted number of the cases with AdaBoost algorithm

The AdaBoost algorithm is used to perform predictive analysis on daily and cumulative cases. Figure 4.5 and 4.6 show the daily and cumulative cases predictions, together with actual numbers. Although there are considerable differences on some days, the projected and actual numbers typically very close to each other.

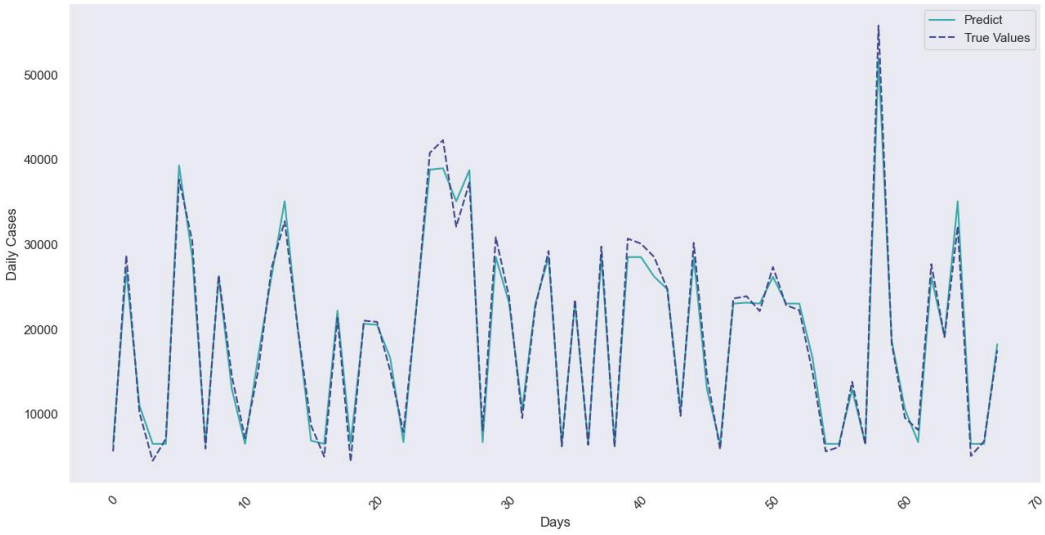

Figure 4. 5 Daily cases prediction with AdaBoost algorithm

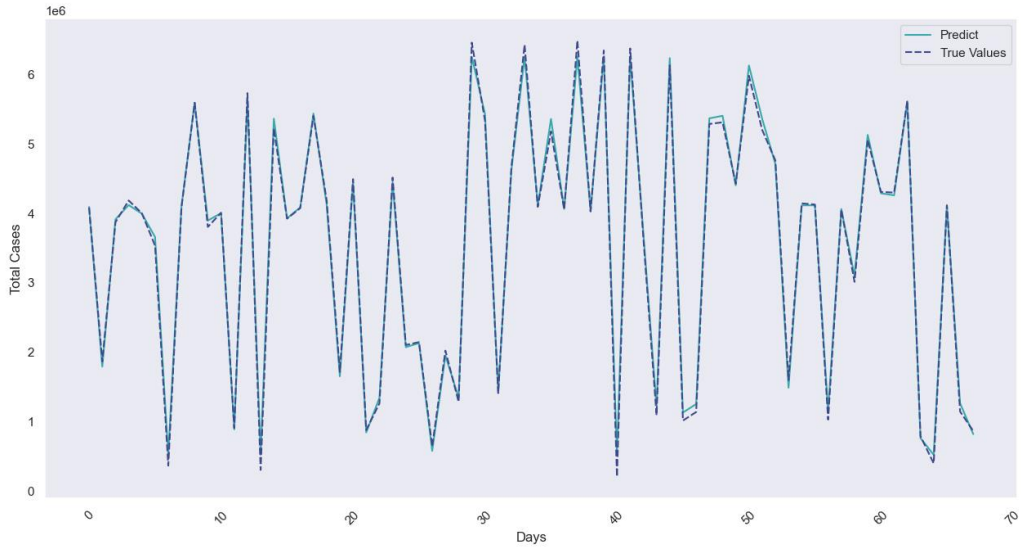

Figure 4. 6 Cumulative cases prediction with AdaBoost algorithm

The AdaBoost algorithm's performance is shown in Table 4.2. Daily cases and cumulative cases are evaluated using the performance criteria given by equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 2 Performance results of the AdaBoost algorithm on daily cases and cumulative cases.

Metrics	**Daily cases**	**Cumulative cases**
EVS	96.4523%	99.7363%
MAE	2.4792%	0.9622%
MAPE	8.1956%	3.1534%
MSE	0.1154%	0.0153%
MSLE	0.0541%	0.007%
RMSE	3.397%	1.2355%
r^2 score	96.4416%	99.73%

EVS value shows that 96.45% of the variations in the daily cases and 99.73% of the variations in the cumulative cases can be explained by the AdaBoost model.

MAE value shows that average size of the errors in daily cases is 2.4792% and average size of the errors in cumulative cases is 0.9622%.

MAPE value reveals that means absolute percentage of error in daily cases is 8.1956% and mean absolute percentage of error in cumulative cases is 3.1534%.

The mean square error is 0.154 percent in daily cases and 0.0153 percent in cumulative cases, according to the MSE value.

MSLE value shows that the ratio of between prediction and true value in daily cases is 0.0541% and the ratio of between predicted and actual value in cumulative cases is 0.007%.

RMSE value gives that the root mean squared error in daily cases is 3.397% and the root mean squared error in cumulative cases is 1.2355%.

r^2 score shows that the agreement of the prediction with the actual values in daily cases is 96.4416% while the agreement of the prediction with the actual values in cumulative cases is 99.73%.

In conclusion, considering the values of performance metrics AdaBoost algorithm shows superior results on the cumulative cases.

4.2.2. Predicted number of the deaths with AdaBoost algorithm

Numbers of daily and cumulative deaths are predicted using AdaBoost algorithm. Figure 4.7 and figure 4.8 provide us both predicted values and the actual values for number of daily and cumulative deaths respectively.

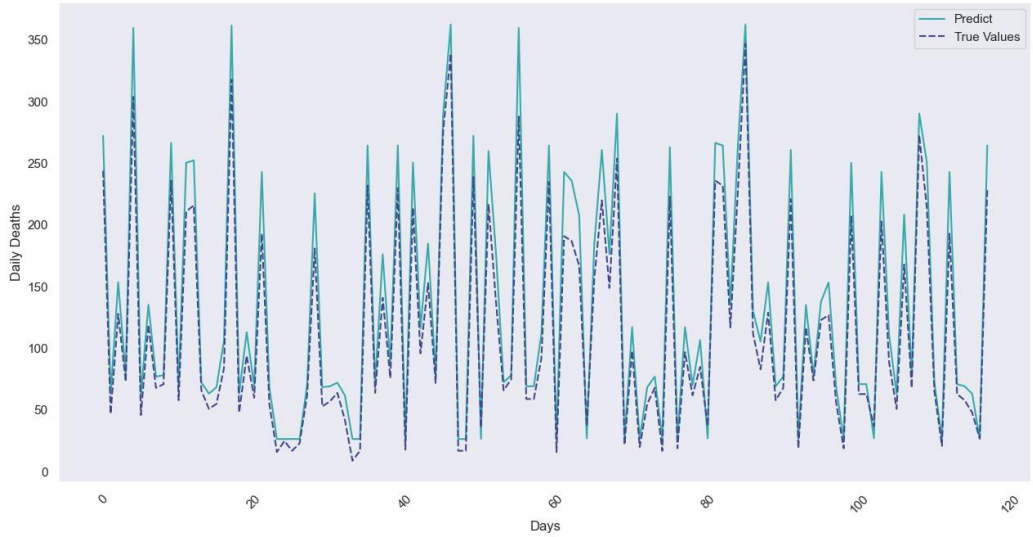

Figure 4. 7 Daily deaths prediction with AdaBoost algorithm

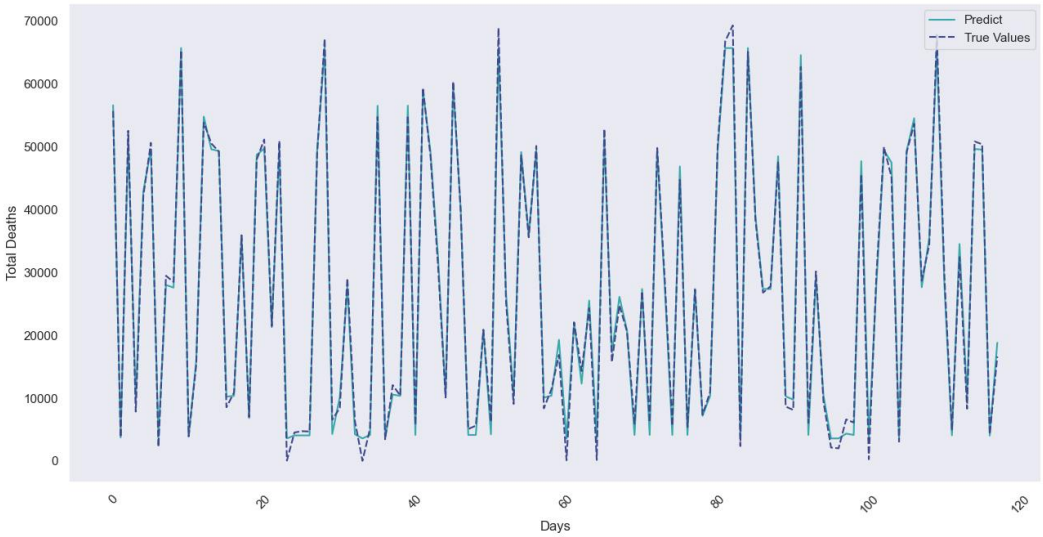

Figure 4. 8 Cumulative deaths prediction with AdaBoost algorithm

Table 4.3 illustrates the performance of the AdaBoost algorithm. Here, number of daily deaths and cumulative deaths are evaluated using the performance criteria given by equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 3 Performance results of the AdaBoost algorithm on daily deaths and cumulative deaths.

Metrics	**Daily deaths**	**Cumulative deaths**
EVS	95.1317%	99.5365%
MAE	4.4501%	1.3624%
MAPE	14.0681%	5.8579%
MSE	0.321%	0.0281%
MSLE	0.147%	0.0167%
RMSE	5.6658%	1.6756%
r^2 score	90.3138%	99.536%

The EVS scores indicate that 95.1317 percent of the variations of the daily deaths and 99.5365 percent of the variations of the cumulative deaths can be explained by the AdaBoost model.

MAE values reveals that the average size of the errors in daily deaths is 4.4501% and the average size of the errors in cumulative deaths is 1.3624%.

MAPE values shows that the mean absolute percentage of error in daily deaths is 14.0681% and the mean absolute percentage of error in cumulative deaths is 5.8579%.

MSE values reveal that the mean square error is 0.321% in daily deaths and 0.0281% in cumulative cases.

MSLE valued indicates that the ratio of between the predicted and the actual values in daily deaths is 0.147% and the ratio of between the predicted and the actual values in cumulative deaths is 0.0167%.

RMSE values indicate that the root mean squared error in daily deaths is 5.6658% and the root mean squared error in cumulative deaths is 1.6756%.

r^2 score shows that the agreement of the predicted and the actual values in daily deaths is 90.3138% while the agreement of the predicted and the actual values in cumulative deaths is 99.536%.

Consequently, considering the values of performance metrics AdaBoost algorithm show superior performance when the number of the cumulative deaths are used as an input to the model.

4.3. Prediction with CatBoost algorithm

Secondly, we applied the CatBoost algorithm which is one of the boosting algorithms to Turkey coronavirus data for prediction.

4.3.1. Predicted number of the cases with CatBoost algorithms

The second boosting algorithm used to predict the number of cases is CatBoost. The line graphs of the daily and cumulative prediction algorithm are shown in figures 4.9 and 4.10.

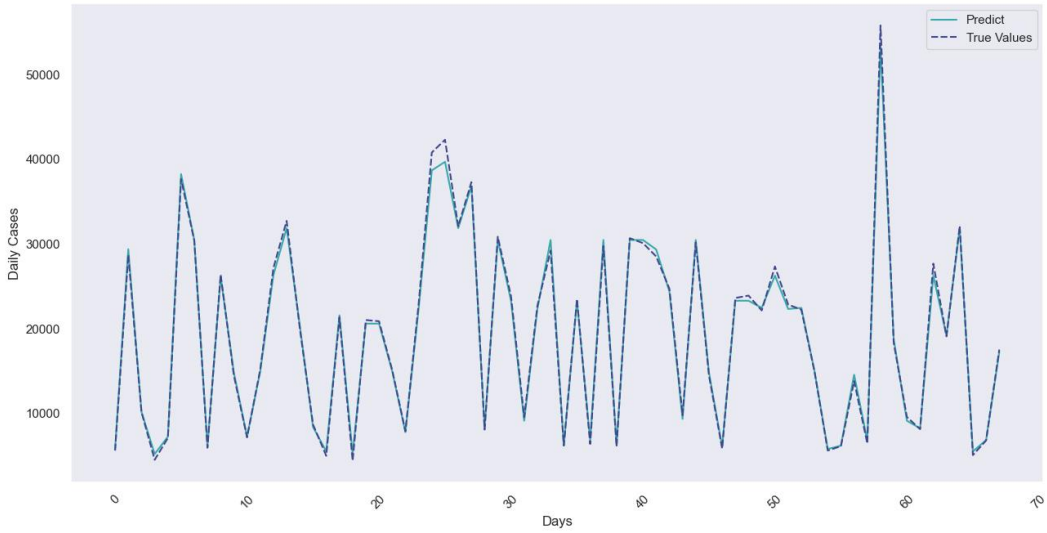

Figure 4. 9 Daily cases prediction with CatBoost algorithm

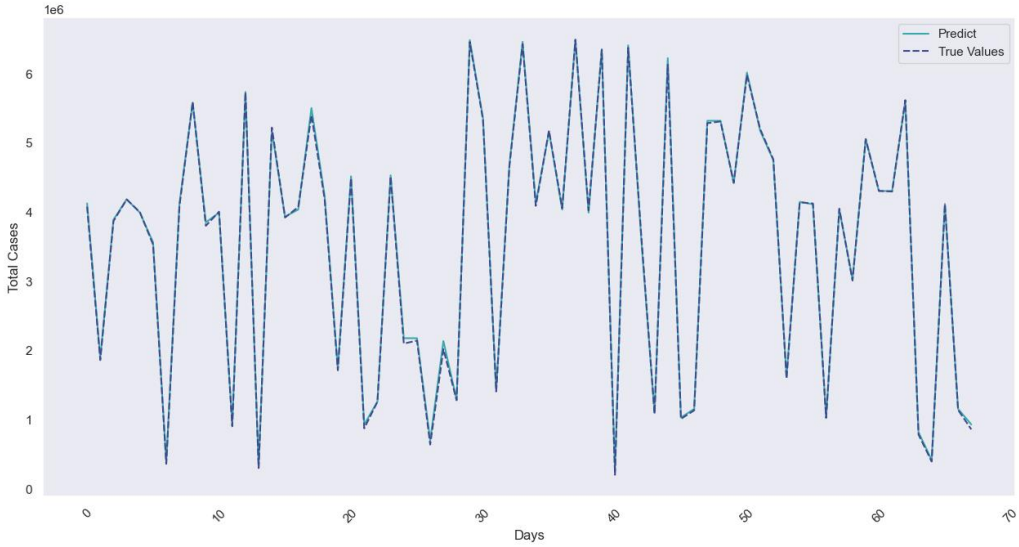

Figure 4. 10 Cumulative cases prediction with CatBoost algorithm

The results of the CatBoost algorithm can be seen in Table 4.4. As can be seen, daily cases and cumulative cases are evaluated by calculating the performance criteria equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 4 Performance results of the CatBoost algorithm on daily cases and cumulative cases.

Metrics	**Daily cases**	**Cumulative cases**
EVS	97.3092%	99.9753%
MAE	2.0864%	0.2766%
MAPE	6.2124%	0.6892%
MSE	0.0876%	0.0014%
MSLE	0.0406%	0.0006%
RMSE	2.9589%	0.378%
r^2 score	97.3002%	99.9753%

EVS values reveal that 97.3092 percent of the variations in daily cases and 99.9753 percent of the variations in cumulative cases can be explained by the CatBoost algorithm.

According to the MAE value, average size of the errors in daily cases is 2.0864 percent and average size of the errors in cumulative cases is 0.2766 percent.

MAPE values reveal, the mean absolute percentage of error in daily cases is 6.2124 percent, while the mean absolute percentage of error in cumulative cases is 0.6892 percent. So, we can infer that the accuracy of the prediction model with CatBoost algorithm is quite high especially for cumulative case data.

The mean square error in daily cases is 0.0876 percent, whereas the mean square error in cumulative cases is 0.0014 percent. The MSE value for cumulative cases is quite lower than that of daily cases which means one should get better predictions for cumulative cases data.

The MSLE number indicates that the ratio of predicted to actual value in daily cases is 0.0406 percent, while the ratio of predicted to actual value in cumulative instances is 0.0006 percent.

The root mean squared error in daily cases is 2.9589 percent, whereas the root mean squared error in cumulative cases is 0.378 percent, according to the RMSE value.

The agreement of the predicted with the actual values in daily cases is 97.3002 percent, while the agreement with the actual values in cumulative cases is 99.9753 percent, according to the r^2 score. As a result, when the values of performance metrics are considered, it is clear that the CatBoost algorithm performs better in cumulative circumstances.

4.3.2. Predicted number of the deaths with CatBoost algorithm

Two types of number of deaths input data are used for prediction purposes: number of daily deaths and number of cumulative deaths. Figures 4.11 and 4.12 show predicted values together with actual values for the daily and cumulative forms, respectively.

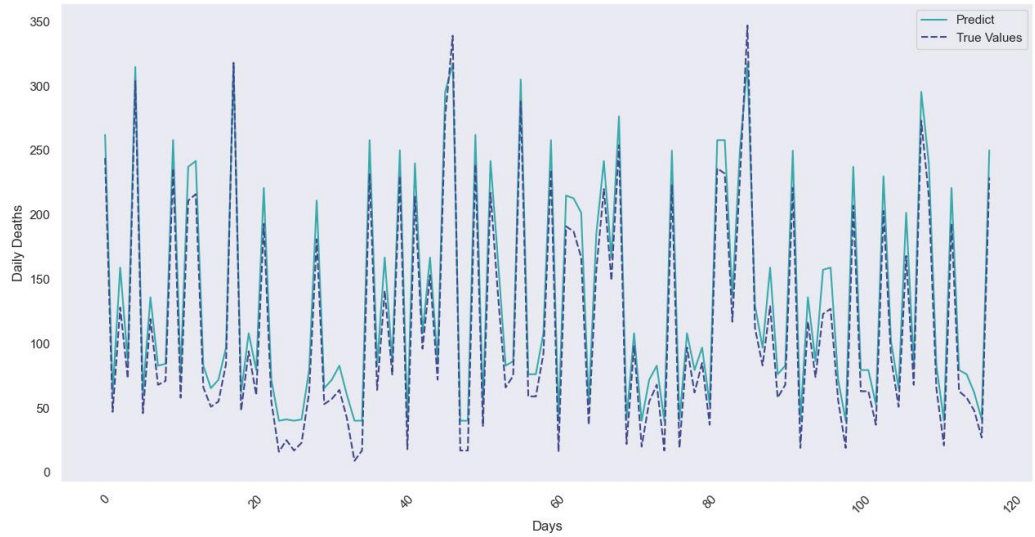

Figure 4. 11 Daily deaths prediction with CatBoost algorithm

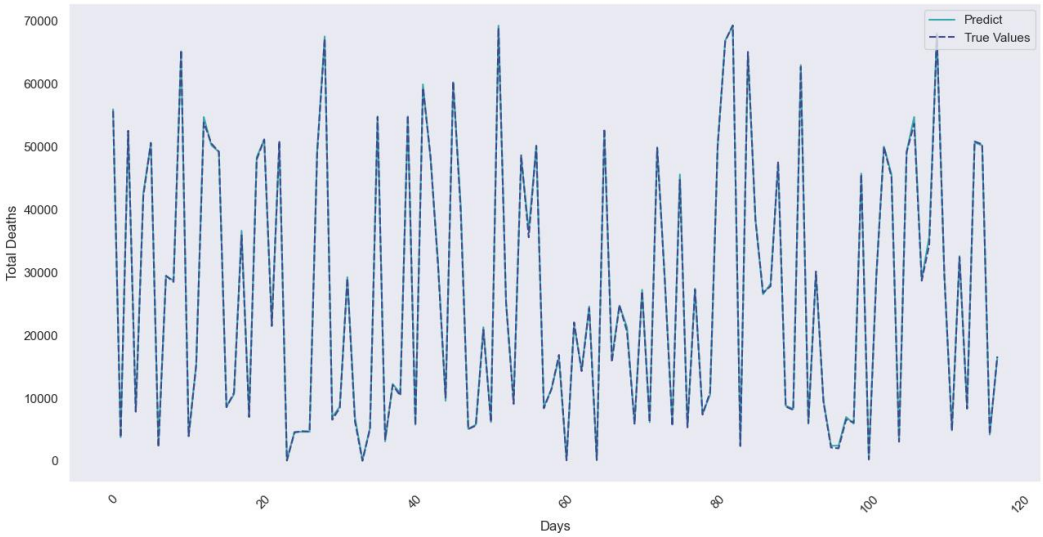

Figure 4. 12 Cumulative deaths prediction with CatBoost algorithm

Daily deaths and cumulative deaths are evaluated using performance criteria, which are calculated according to the equations' (1), (2), (3), (4), (5), (6), and (7). The CatBoost algorithm's performance results are given in Table 4.5.

Table 4. 5 Performance results of the CatBoost algorithm on daily deaths and cumulative deaths.

Metrics	**Daily deaths**	**Cumulative deaths**
EVS	96.2435%	99.9817%
MAE	4.4451%	0.246%
MAPE	17.1218%	0.7575%
MSE	0.2677%	0.0011%
MSLE	0.1368%	0.0005%
RMSE	5.1744%	0.3329%
r^2 score	91.9211%	99.9817%

EVS values reveal that 96.2435 percent of the variations in daily deaths, and 99.9817 percent of the variations in cumulative deaths can be explained by CatBoost algorithm.

According to the MAE value, average size of the errors in daily fatalities is 4.4451 percent and average size of the errors in cumulative deaths is 0.246 percent.

MAPE values reveals, the mean absolute percentage of error in daily fatalities is 17.1218 percent, while the mean absolute percentage of error in cumulative deaths is 0.7575 percent. So, we can infer that the accuracy of the prediction model with CatBoost algorithm is quite high especially for cumulative death data.

The mean square error in daily fatalities is 0.2677 percent, whereas the mean square error in total deaths is 0.0011 percent. The MSE value for cumulative deaths is quite lower than that of daily deaths which means one should get better predictions for cumulative deaths data.

MSLE valued reveals that the ratio of between the predicted and the actual values in daily fatalities is 0.1368% and the ratio of between the predicted and the actual values in cumulative deaths is 0.0005%.

RMSE values indicate that the root mean squared error in daily deaths is 5.1744 percent and the root mean squared error in cumulative deaths is 0.3329 percent.

The agreement of the predicted with the actual values in daily deaths is 91.9211 percent, while the agreement with the actual values in cumulative deaths is 99.9817 percent, according to the r^2 score.

In summary, from these results it can be seen that the CatBoost algorithm performed better on cumulative deaths.

4.4. Prediction with XGBoost algorithm

Finally, XGBoost algorithm is applied to Turkey coronavirus data for prediction.

4.4.1. Predicted number of the cases with XGBoost algorithm

Line graphs of XGBoost algorithm results for daily and cumulative situations are shown. Figures 4.13 and 4.14 show both the predicted values and the actual values for the daily and cumulative forms, respectively.

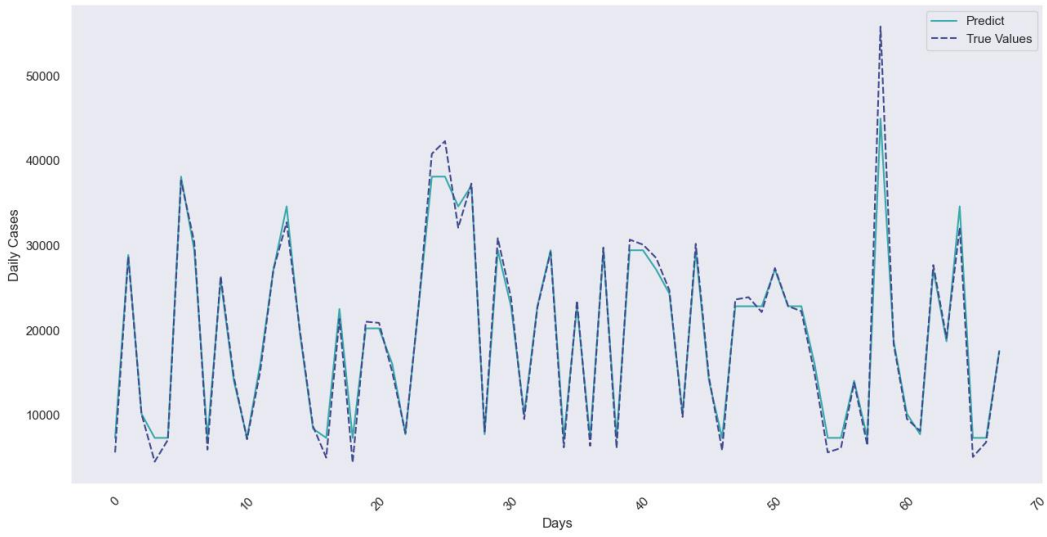

Figure 4. 13 Daily cases prediction with XGBoost algorithm

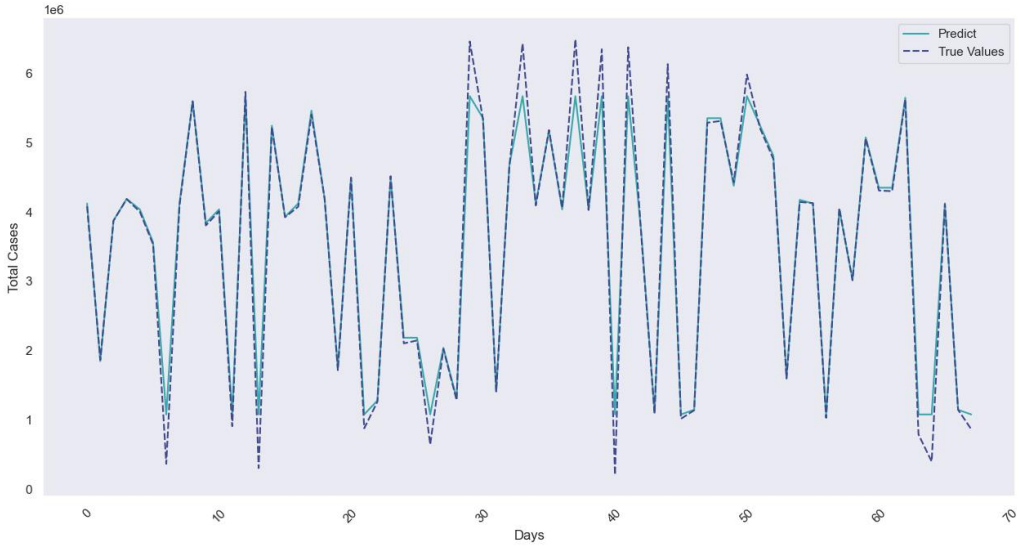

Figure 4. 14 Cumulative cases prediction with XGBoost algorithm

Table 4.6 displays the performance of XGBoost algorithm. The performance of XGBoost algorithm on both number of daily and cumulative cases data is measured by using performance criteria which are calculated using equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 6 Performance results of the XGBoost algorithm on daily cases and cumulative cases.

Metrics	**Daily cases**	**Cumulative cases**
EVS	95.4863%	97.5625%
MAE	2.7754%	1.8703%
MAPE	9.9445%	7.6233%
MSE	0.1465%	0.1412%
MSLE	0.0651%	0.068%
RMSE	3.8271%	3.7573%
r^2 score	95.4836%	97.5609%

The EVS scores indicate that 95.4863 percent of the variations of the daily cases and 97.5625 percent of the variations of the cumulative cases can be explained by the XGBoost model.

MAE values reveals that the average size of the errors in daily cases is 2.7754% and the average size of the errors in cumulative cases is 1.8703%.

MAPE values show that the mean absolute percentage of error in daily cases is 9.9445% and the mean absolute percentage of error in cumulative cases is 7.6233%.

MSE values reveal that the mean square error is 0.1465% in daily cases and 0.1412% in cumulative cases.

MSLE valued indicates that the ratio of between the predicted and the actual values in daily cases is 0.0651% and the ratio of between the predicted and the actual values in cumulative cases is 0.068%.

RMSE values indicate that the root mean squared error in daily cases is 3.8271 percent, and the root mean squared error in cumulative cases is 3.7573 percent.

r^2 score shows that the agreement of the predicted and the actual values in daily cases is 95.4836% while the agreement of the predicted and the actual values in cumulative cases is 97.5609%.

In summary, according to considered performance metrics XGBoost algorithm shows superior prediction performance with cumulative cases data.

4.4.2. Predicted number of the deaths with XGBoost algorithm

Line graphs of predicted and actual values with XGBoost algorithm for daily and cumulative deaths are shown in Figures 4.15 and 4.16.

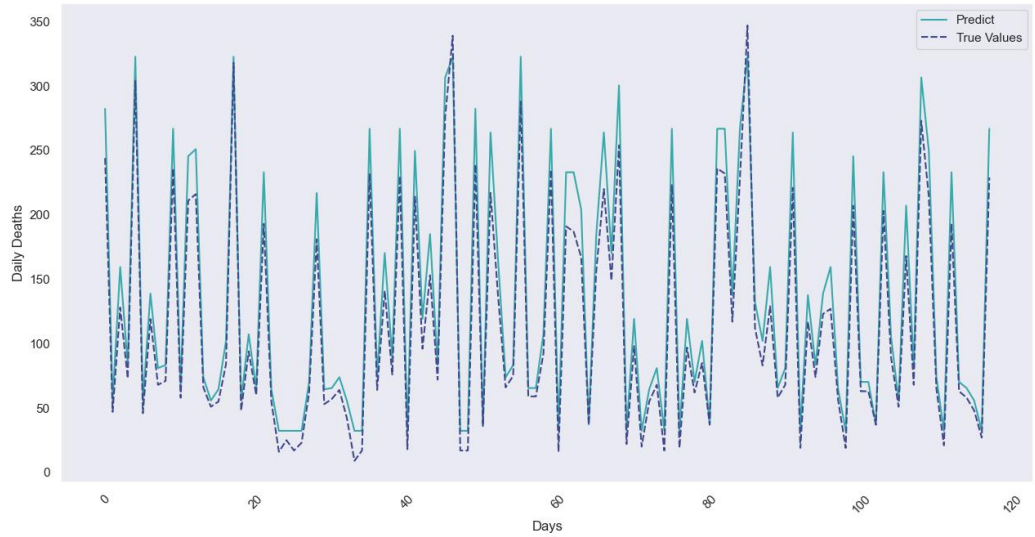

Figure 4. 15 Daily deaths prediction with XGBoost algorithm

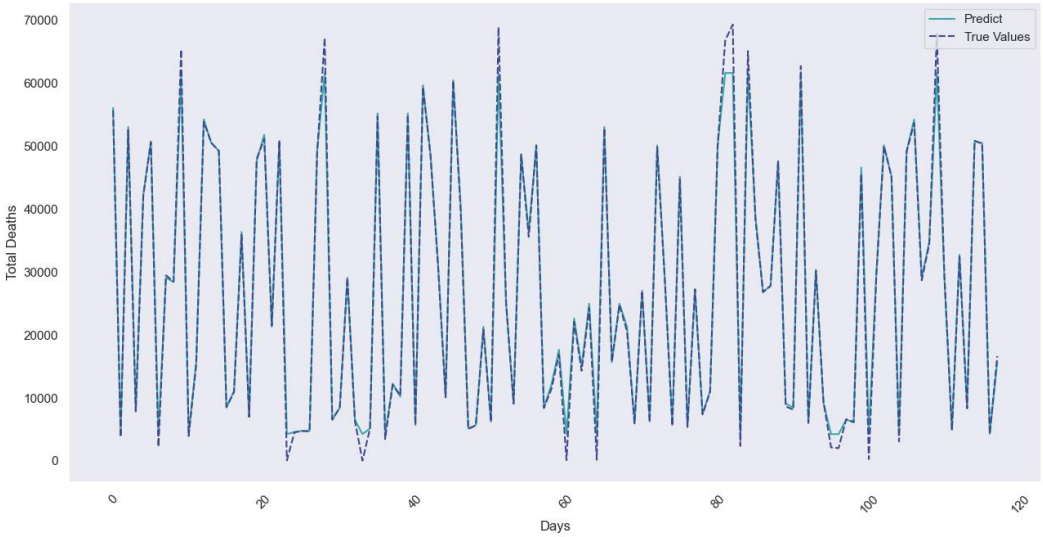

Figure 4. 16 Cumulative deaths prediction with XGBoost algorithm

The performance metrics calculated for XGBoost algorithm are given in Table 4.7. Daily deaths and cumulative deaths are evaluated by considering performance criteria which are calculated by equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 7 Performance results of the XGBoost algorithm on daily deaths and cumulative deaths.

Metrics	**Daily deaths**	**Cumulative deaths**
EVS	94.4853%	99.3325%
MAE	4.4912%	0.8788%
MAPE	14.6178%	3.6861%
MSE	0.3291%	0.0404%
MSLE	0.15%	0.0176%
RMSE	5.7368%	2.0109%
r^2 score	90.0693%	99.3317%

The EVS scores indicate that 94.4853 percent of the variations of the daily deaths and 99.3325 percent of the variations of the cumulative deaths can be explained by the XGBoost model.

MAE values reveal that the average size of the errors in daily deaths is 4.4912% and the average size of the errors in cumulative deaths is 0.8788%.

MAPE values show that the mean absolute percentage of error in daily deaths is 14.6178% and the mean absolute percentage of error in cumulative deaths is 3.6861%.

MSE values reveal that the mean square error is 0.329% in daily deaths and 0.0404% in cumulative cases.

MSLE valued indicates that the ratio of between the predicted and the actual values in daily deaths is 0.15% and the ratio of between the predicted and the actual values in cumulative deaths is 0.0176%.

RMSE values indicate that the root mean squared error in daily deaths is 5.7368% and the root mean squared error in cumulative deaths is 2.0109%.

r^2 score shows that the agreement of the predicted and the actual values in daily deaths is 90.0693% while the agreement of the predicted and the actual values in cumulative deaths is 99.3317%.

In summary, according to the considered performance metrics XGBoost algorithm shows superior prediction performance on the cumulative deaths data.

4.5. Comparison of prediction results

The goal of this section is to compare prediction performances of AdaBoost, CatBoost, and XGBoost algorithms.

4.5.1. Comparison with daily case and cumulative case results

According to the results, it is apparent that CatBoost algorithm outperformed the other algorithms in terms of considered performance metrics. The performance results of daily case predictions and cumulative case predictions for all boosting algorithms are given in Table 4.8 and 4.10; respectively. As a result, it is clearly seen that the CatBoost algorithm's prediction performance is better than the others on case data. Also, the results revealed that using cumulative data as input to the boosting algorithms gives better prediction performance than using daily data as an input.

Table 4. 8 Performance results of daily case predictions

Metrics	**AdaBoost**	**CatBoost**	**XGBoost**
EVS	96.4523%	97.3092%	95.4863%
MAE	2.4792%	2.0864%	2.7754%
MAPE	8.1956%	6.2124%	9.9445%
MSE	0.1154%	0.0876%	0.1465%
MSLE	0.0541%	0.0406%	0.0651%
RMSE	3.397%	2.9589%	3.8271%
r^2 score	96.4416%	97.3002%	95.4836%

The CPU times for AdaBoost, CatBoost, and XGBoost are listed in Table 4.9. From Table 4.9, we can get minimum CPU time with AdaBoost for daily case predictions. Still, the differences between the CPU time performances of all algorithms are negligible. It should be noted that in spite of being slowest CatBoost algorithm gives the best MAPE performance. Thus, having the best prediction accuracy CatBoost algorithm can be considered as the best performer among others.

Table 4. 9 CPU times of boosting techniques with daily case data.

CPU time (s)	AdaBoost	CatBoost	XGBoost
	0.2898066	0.6674096	0.3735122

Table 4. 10 Performance results of cumulative case predictions

Metrics	AdaBoost	CatBoost	XGBoost
EVS	99.7363%	99.9753%	97.5625%
MAE	0.9622%	0.2766%	1.8703%
MAPE	3.1534%	0.6892%	7.6233%
MSE	0.0153%	0.0014%	0.1412%
MSLE	0.007%	0.0006%	0.068%
RMSE	1.2355%	0.378%	3.7573%
r^2 score	99.73%	99.9753%	97.5609%

CPU time performances of AdaBoost, CatBoost, and XGBoost algorithms with cumulative case data are given in Table 4.11. From Table 4.11, XGBoost algorithm achieved the best CPU time with cumulative case data, while the CatBoost algorithm is the slowest. On the other hand, with a MAPE value of 0.6892%, the CatBoost algorithm achieves the best prediction accuracy. Since the CPU time performances of all boosting algorithms are all close to each other CPU time performances of the algorithms can be ignored. Thus, having the best prediction accuracy CatBoost algorithm can be considered as the best performer among others.

Table 4. 11 CPU times of boosting techniques with cumulative case data.

CPU time (s)	AdaBoost	CatBoost	XGBoost
	0.4827561	0.6004251	0.3821334

Table 4.12 gives sample predicted values of the tree boosting algorithms using actual cumulative case data values.

Table 4. 12 Some instances of the Covid-19 predictions using cumulative case data

True Value	AdaBoost	CatBoost	XGBoost
4082356	4093451	4129654	4122782
1866296	1793904	1897155	1847677
3873831	3915762	3893419	3875505
4186651	4120970	4188406	4189411
4002799	3996425	3996376	4037302
3534983	3660391	3566893	3562688
367379	522883	390562	1079016
4088311	4098961	4129228	4122782
5593542	5596164	5586955	5600553
3805716	3897279	3854476	3841097

4.5.1.1. Evaluation of case predictions according to the CV value

The CV value gives information about how reliable the prediction is. The smaller the value of CV is the more reliable the prediction. The CV values of the prediction errors calculated separately for daily and cumulative cases using equation (8) and are shown in table 4.13.

Table 4. 13 CV values for case predictions

CV value	AdaBoost	CatBoost	XGBoost
Daily case	1.236	1.4514	1.707
Cumulative case	1.304	1.1424	1.9647

From Table 4.13, the CV values of daily cases reveal that the predictions with AdaBoost algorithm are less risky with a CV value of 1.236 compared to the other algorithms. In terms of

MAPE, CatBoost is better than all other algorithms. However, the standard deviation of predictions with CatBoost is a little bigger; causing the prediction values to scatter more around the actual values, but this is negligible.

Also, CatBoost has the smallest CV value amongst consider algorithms for cumulative case. The predictions with CatBoost algorithm have a CV value of 1.1424. The CatBoost algorithm not only best performer in terms of MAPE but also provides the least risky predictions for cumulative cases.

4.5.2. Comparison with daily death and cumulative death results

The performances of the considered boosting algorithms are compared with respect to several performance metrics and the results are summarized in Table 4.14 and 4.16. From table 4.14, it is clearly seen that all algorithms provide accurate and robust predictions. However, CatBoost algorithm has the best prediction performance with respect to others in terms of both daily and cumulative data. It should be noted that with cumulative data set, the prediction performances of the three algorithms are close to each other.

Table 4. 14 Performances of boosting algorithms on daily death predictions

Metrics	**AdaBoost**	**CatBoost**	**XGBoost**
EVS	95.1317%	96.2435%	94.4853%
MAE	4.4501%	4.4451%	4.4912%
MAPE	14.0681%	17.1218%	14.6178%
MSE	0.321%	0.2677%	0.3291%
MSLE	0.147%	0.1368%	0.15%
RMSE	5.6658%	5.1744%	5.7368%
r^2 score	90.3138%	91.9211%	90.0693%

Table 4.15 shows the CPU times for AdaBoost, CatBoost, and XGBoost. Although CatBoost is the slowest algorithm according to CPU time, the differences between the CPU times of all algorithms are negligible.

Table 4. 15 CPU time performances of boosting algorithms on daily death predictions

CPU time (s)	AdaBoost	CatBoost	XGBoost
	0.2438143	0.618658	0.3939434

Table 4. 16 Performances of boosting algorithms on cumulative death predictions

Metrics	AdaBoost	CatBoost	XGBoost
EVS	99.5365%	99.9817%	99.3325%
MAE	1.3624%	0.246%	0.8788%
MAPE	5.8579%	0.7575%	3.6861%
MSE	0.0281%	0.0011%	0.0404%
MSLE	0.0167%	0.0005%	0.0176%
RMSE	1.6756%	0.3329%	2.0109%
r^2 score	99.536%	99.9817%	99.3317%

Resulting CPU times of AdaBoost, CatBoost, and XGBoost algorithms with cumulative death prediction are listed in table 4.17 below. From Table 4.17, it is clearly seen that the AdaBoost approach is the fastest, while the CatBoost algorithm is the slowest on cumulative death prediction. Note that the differences between CPU times among all considered boosting algorithms are negligible. From table 4.16 CatBoost has the best MAPE performance with a value of 0.7575 percent which indicates that the accuracy of the predictions with CatBoost algorithm is the highest among others. Also, from Table 4.13 it is seen that CatBoost algorithm has the best CV performance with a value of 1.1424 which indicates that the CatBoost algorithm gives the most robust predictions. So, in order to get accurate and robust predictions one should use CatBoost algorithm.

Table 4. 17 CPU time performance of boosting algorithms on cumulative death predictions

CPU time (s)	AdaBoost	CatBoost	XGBoost
	0.3738054	0.6615411	0.3957915

A snapshot of actual values and corresponding prediction values of each boosting algorithm using actual cumulative death values is given in Table 4.18.

Table 4. 18 A sample dataset of actual vs. predicted number of Covid-19 cumulative deaths.

True Value	AdaBoost	CatBoost	XGBoost
55713	56627	55983	56104
3786	3741	3777	4247
52565	51359	52556	53057
7858	9403	7868	7979
42187	42894	42622	42300
50650	49624	50591	50767
2259	3593	2431	4247
29489	28025	29466	29118
28503	27602	28727	28423
65373	65740	65225	61644

4.5.2.1. Evaluation of death predictions according to the CV value

Equation (8) is used to calculate the CV value of the number of death predictions. The CV value of the prediction errors of the AdaBoost, CatBoost, and XGBoost algorithms are summarized in Table 4.19. Note that daily and cumulative death data are used as inputs to each boosting algorithm.

Table 4. 19 CV values for death predictions

CV value	AdaBoost	CatBoost	XGBoost
Daily death	0.7429	0.4263	0.698
Cumulative death	1.2246	1.3204	2.1537

From Table 4.19, it is seen that the CatBoost algorithm has the smallest CV value of 0.4263 which indicates that CatBoost algorithm makes more reliable predictions and the prediction values are closely scattered around the actual values.

From the cumulative death predictions row of Table 4.19, with a CV value of 1.2246 the predictions with AdaBoost are less risky than that of CatBoost. However, the differences between the CV values of the boosting algorithms are negligible which means the robustness of the predictions of all algorithms are close to each other. Remember that CatBoost algorithm has the best MAPE performance with a MAPE value of 0.7575 percent (see Table 4.16) which indicates that the accuracy of the predictions with CatBoost algorithm is the highest among others. So, in order to get accurate and robust predictions one should use CatBoost algorithm while predicting number of cumulative cases.

4.6. Evaluation of the best performing CatBoost algorithm with Turkey data

Having determines that the CatBoost algorithm has better prediction performance than that of others Table 4.20 compares the performance of CatBoost algorithm itself on two different datasets.

Table 4. 20 Performance results of CatBoost predictions

Metrics	Cumulative case	Cumulative death
EVS	99.9753%	99.9817%
MAE	0.2766%	0.246%
MAPE	0.6892%	0.7575%
MSE	0.0014%	0.0011%
MSLE	0.0006%	0.0005%
RMSE	0.378%	0.3329%
r^2 score	99.9753%	99.9817%

4.7. Prediction with ANN

Besides boosting algorithms, ANN algorithm utilized on the coronavirus data set. Hyperparameters optimization is carried out to increase the performance of ANN hyperparameters optimization is carried out.

Hyperparameters are variables whose values influence the learning process and affect the model parameters that a learning algorithm learns (Nyuytiymbiy, 2020). Hyperparameters do not updating during model training. For example, the number of hidden in ANN model, number of clusters in K-means, etc. are hyperparameters. However, to make predictions parameters are required by model. Parameters learned from data. In ANN model, daily case, cumulative case, daily death, and cumulative death are inputs.

4.7.1. Hyperparameters optimization

Hyperparameter optimization is the selection of the best hyperparameters to obtain better results by directly affecting the model performance in machine learning algorithms. Hyperparameter optimization is one of the most significant steps in machine learning algorithms to increase the accuracy and robustness of prediction results. It is as important as choosing the right algorithms. There is a strong relationship between hyperparameters and model performance. So, it is undeniable fact that hyperparameters optimization plays a significant role in model performance.

By applying the optimization of the hyperparameter, the accuracy of the result performance can be increased. There exist several different methods to optimize the hyperparameters in the literature.

The most basic hyperparameters optimization algorithms are: grid search, random search, and Bayesian model-based optimization. Grid search is one of the search optimization algorithms. Grid search evaluates all possible alternatives utilizes cross-validation to determine best alternative. The random search method randomly selects the hyperparameters and then uses the best combination for the training the model. In bayesian hyperparameter optimization, the hyperparameters are evaluated with a probability model and the best hyperparameters are selected.

In this thesis, model hyperparameters are adjusted before making predictions. Grid search tries all possible combinations using cross-validation. In this optimization 3fold cross-validation is used. Hyperparameter sets for the ANN model are given in Table 4.21 below.

Table 4. 21 The grid search hyperparameter sets for the ANN model

	Hyperparameter Values
Epochs	[100, 200, 500]
Optimizer	[Adadelta, Adagrad, Adam, Adamax, Ftrl, RMSprop, SGD]
Activation	[elu, relu, selu, linear]
Neurons	[8, 16, 32, 64, 128]

Prediction analysis performed by using ANN based models with the same datasets previously considered in the study. Hyperparameters were optimized with grid search for each analysis. The best hyperparameters for each model are given in Table 4.22 below.

Table 4. 22 The best hyperparameters for the ANN model

	Epochs	**Optimizer**	**Activation**	**Number of Neurons**
Daily case	500	Adam	elu	128
Cumulative case	500	Adam	relu	32
Daily deaths	100	RMSprop	selu	8
Cumulative deaths	500	Nadam	relu	128

ANNs are shown with feature distribution graphs of each of the model hyperparameters. Hyperparameters of each analysis were visualized by using the violin plot from the seaborn library of python. Hintze and Nelson (1998) define the violin plot as follows: violin plot shows the combination of box plot and density curve of data on a single chart. Looking at the structure of the violin plot, it is seen that the median is not a straight line. Note that, in box plot median is shown as a straight line within the interquartile range. The white dot on the violin represents the median. Violin plots make alternatives easily comparable. The width of the violin corresponds to the frequency of data points in each region. The ends of the box in the middle of the violin represent the first and third quarters. The tail of the violin shows outliers. Violin plot provides fast and meaningful information about descriptive statistics for each data. The graphs of all hyperparameter alternatives evaluated in grid search are shown below. Figure 4.17-20 visualizes each hyperparameter performed for daily case, cumulative case, daily death, and cumulative death, respectively.

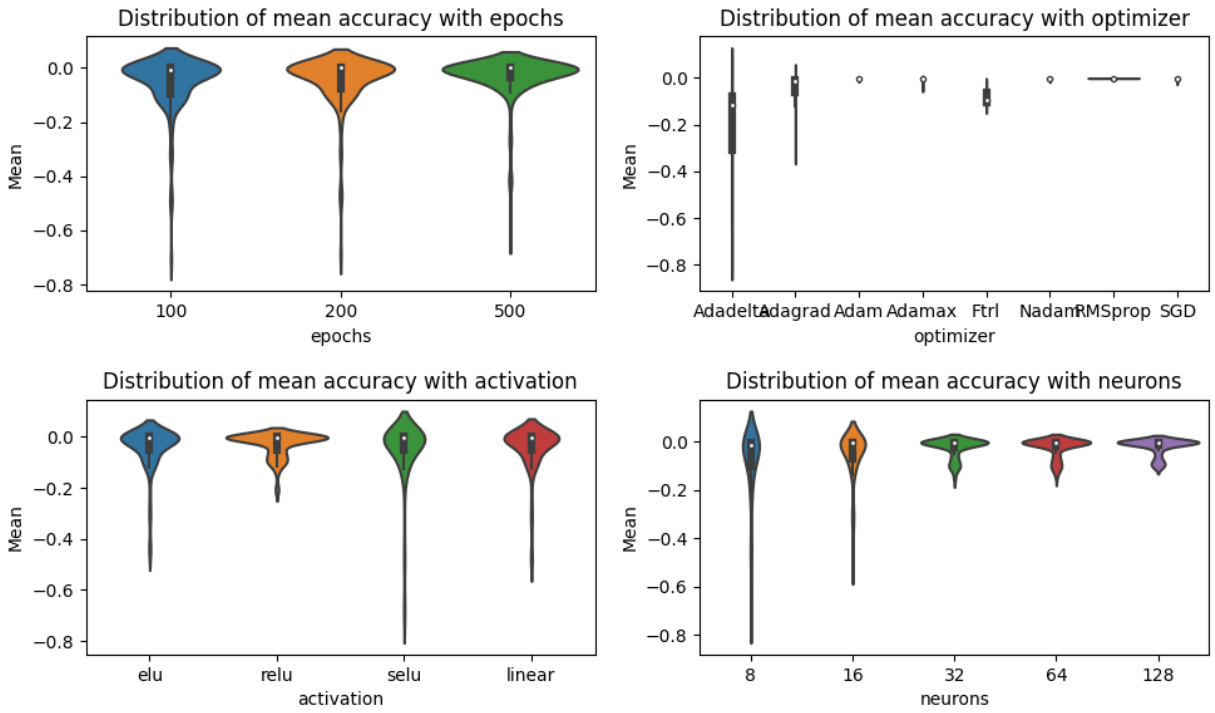

Figure 4. 17 The daily case hyperparameters for the ANN model

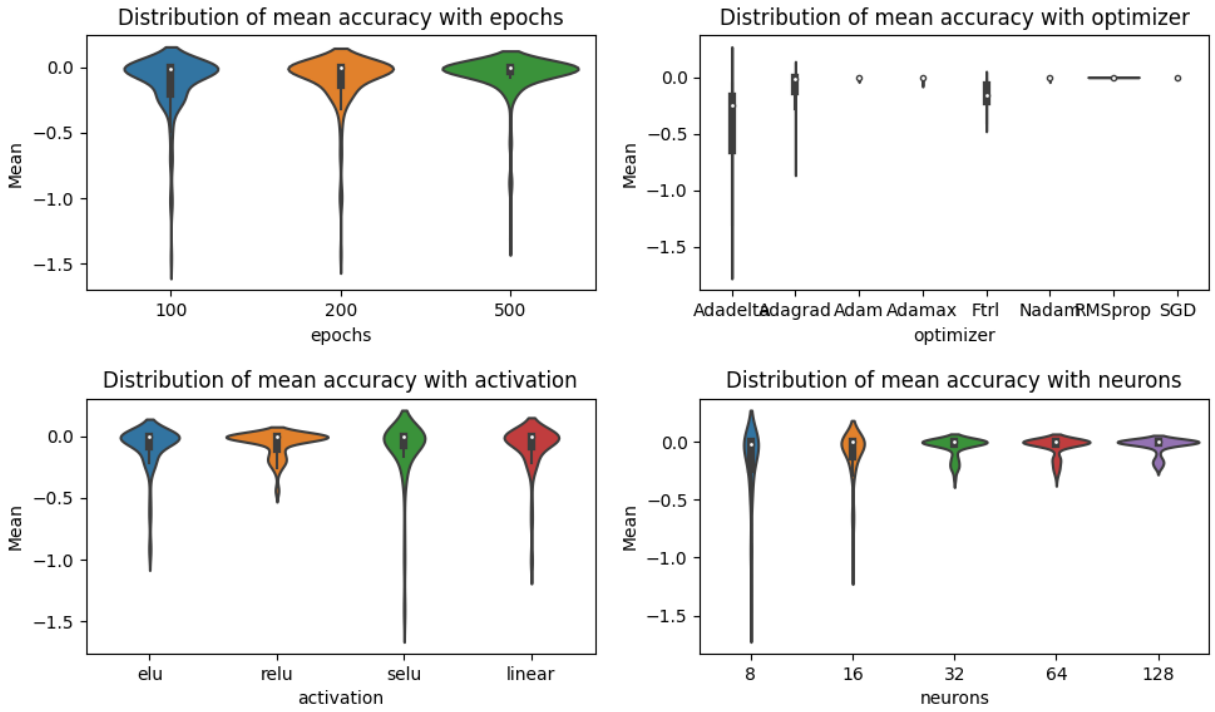

Figure 4. 18 The cumulative case hyperparameters for the ANN model

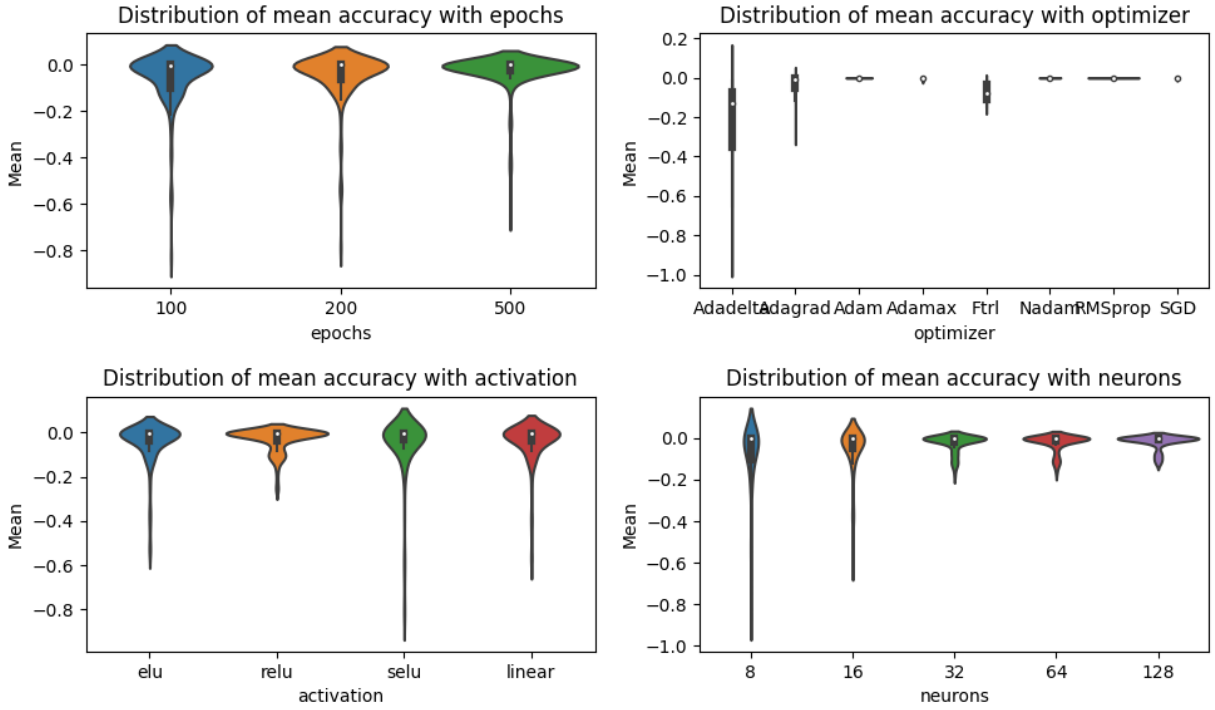

Figure 4. 19 The daily deaths hyperparameters for the ANN model

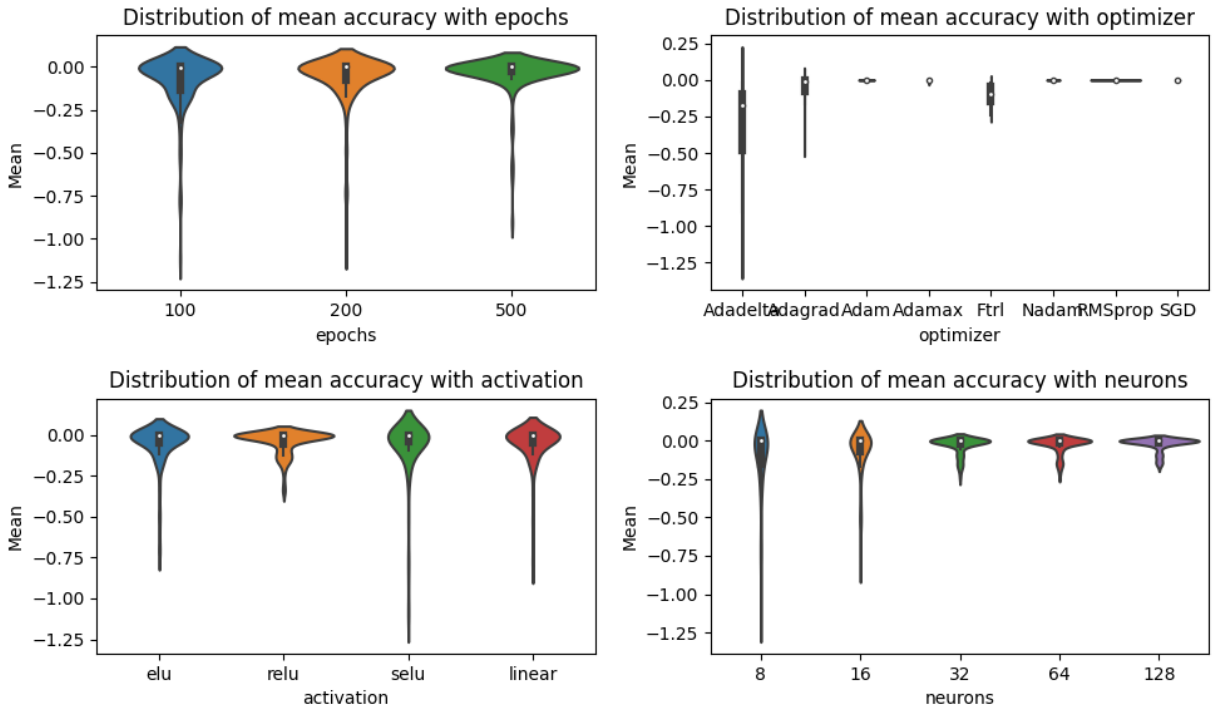

Figure 4. 20 The cumulative deaths hyperparameters for the ANN model

Graphs play a significant role in understanding and interpreting the relationship between variables. These graphs may vary depending on whether the data is categorical or numerical. In the literature, there are many different representations that vary according to the data type or the feature to be compared. When these graphs are read correctly, they are both memorable and help to summarize a lot of information effectively. The hyperparameters of the ANN model are optimized by grid search algorithm. The considered alternatives for activation function are determined to be elu, relu, selu, and linear. Also, considered alternatives for optimizer are determined to be Adadelta, Adagrad, Adam, Adamax, Ftrl, RMSprop, and SGD. Note that the data types of both activation and optimizer are categorical. In order to show the relationship of categorical data, the position of the activation and optimizer relative to the mean on the y-axis is shown with the catplot, that is, the categorical plot of the seaborn library of python. Figure 4.21-24 visualizes the categorical data of each hyperparameter optimization performed for daily case, cumulative case, daily death, and cumulative death, respectively. The closeness of the variables to deviations around the mean is directly proportional to their optimality.

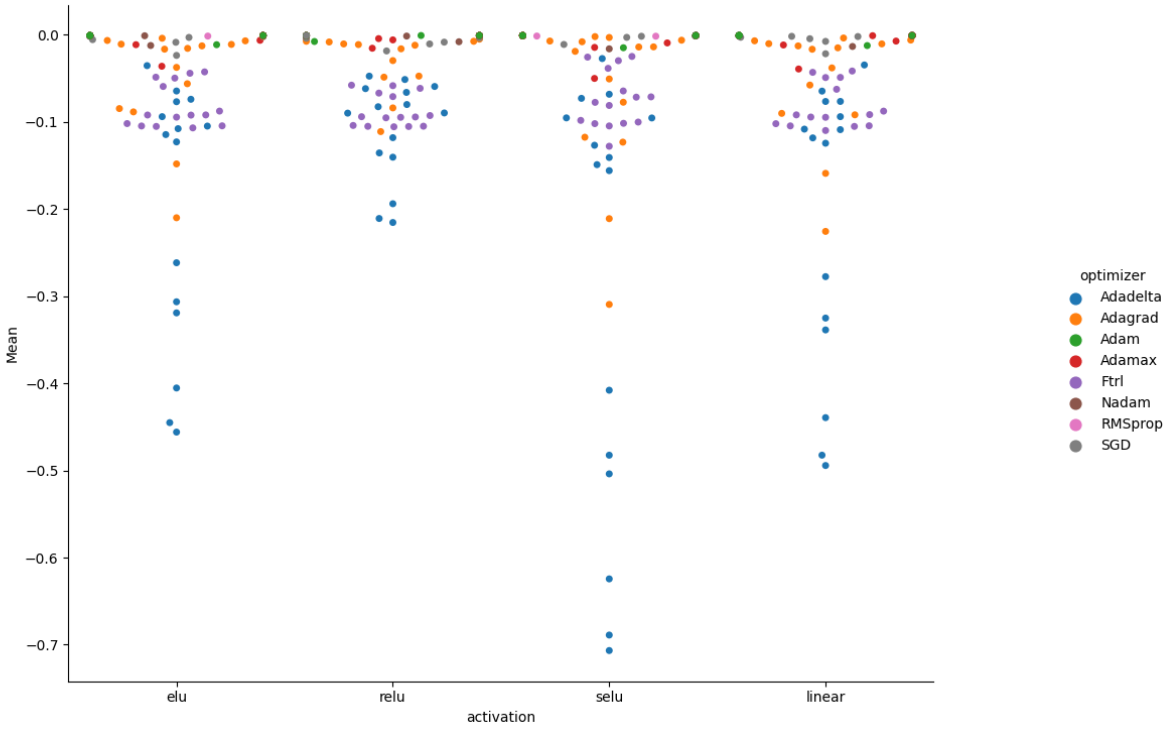

Figure 4. 21 Daily case ANN model

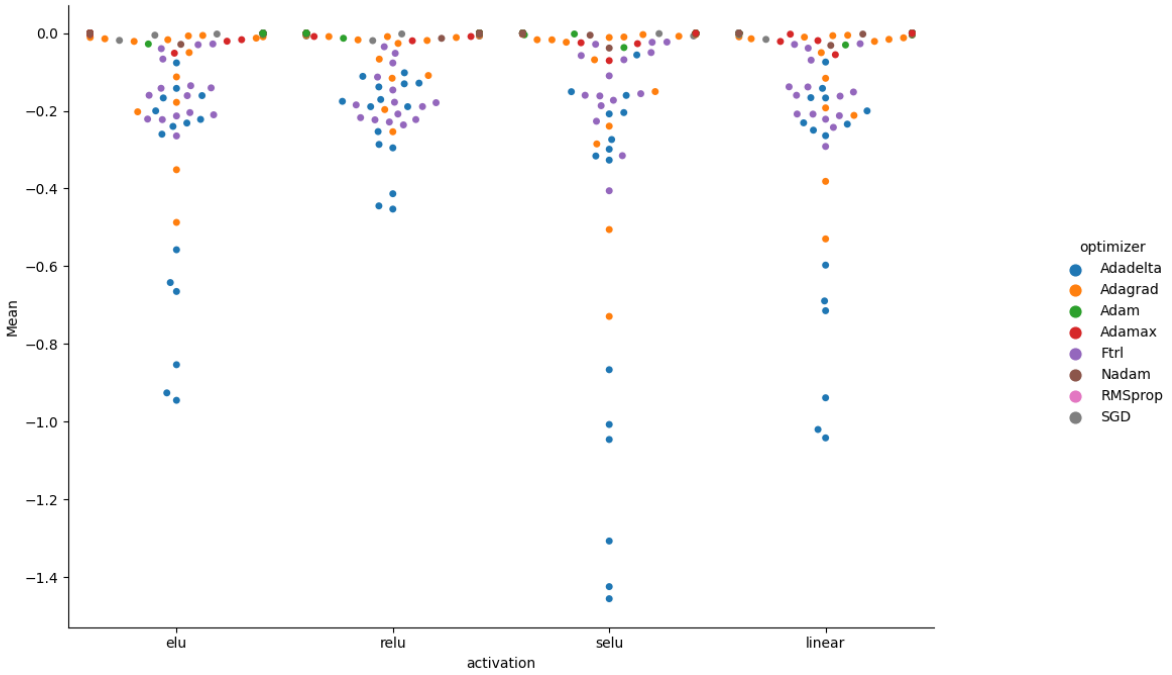

Figure 4. 22 Cumulative case ANN model

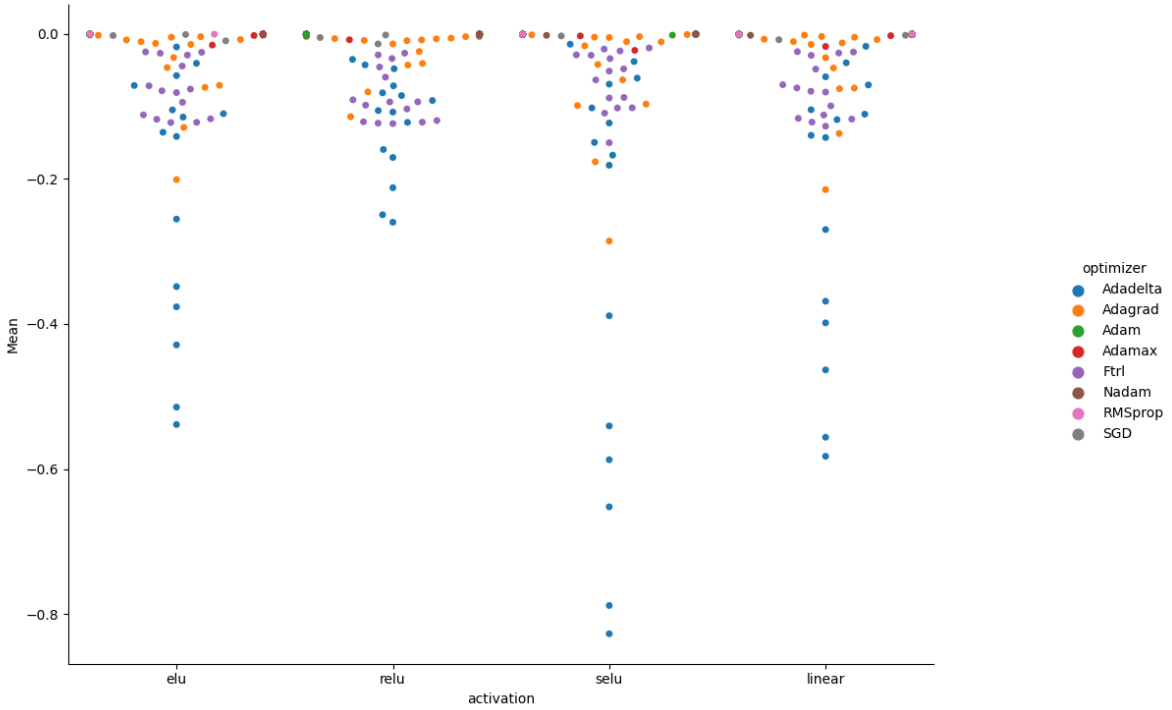

Figure 4. 23 Daily death ANN model

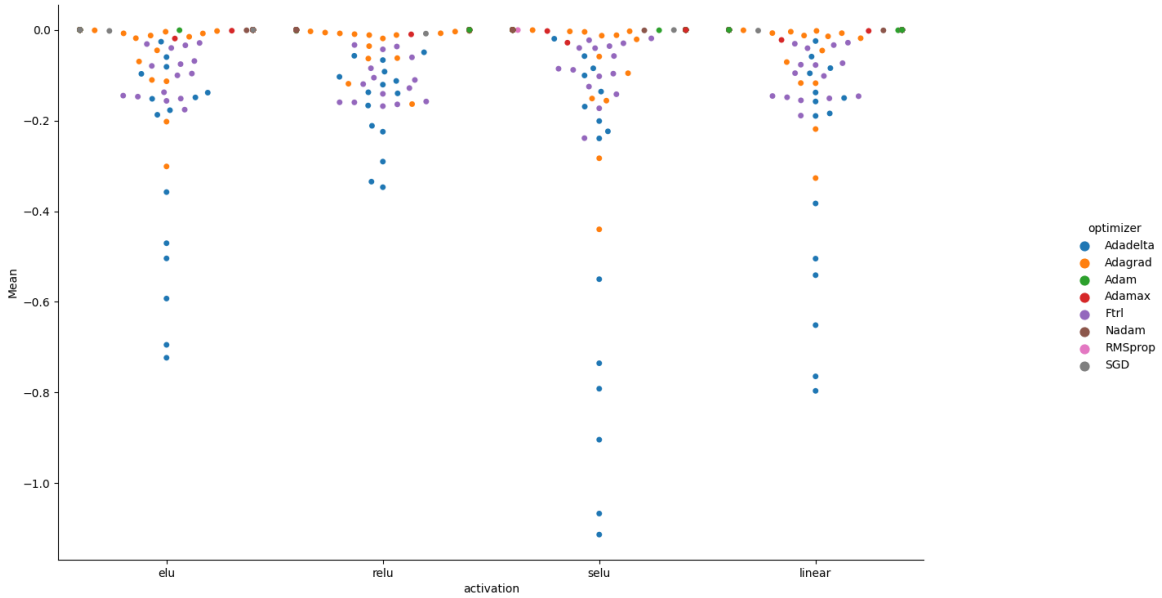

Figure 4. 24 Cumulative death ANN model

In Table 4.23, the CPU time of the prediction on daily and cumulative cases using ANNs is given. Note that the difference between the CPU time performance of ANN with daily cases and ANN with cumulative cases is negligible.

Table 4. 23 CPU time of ANN case predictions

CPU time (s)	**Daily cases**	**Cumulative cases**
	14.8017517	14.7001215

Table 4.24 illustrates the results of the ANN algorithm. Note that, daily case and cumulative case are evaluated by using different performance criteria which are calculated using equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 24 Performance results of ANN predictions

Metrics	Daily cases	Cumulative cases
EVS	98.0328%	99.9088%
MAE	1.8301%	0.6005%
MAPE	5.4951%	1.7907%
MSE	0.0641%	0.0052%
MSLE	0.0306%	0.0023%
RMSE	2.5320%	0.7264%
r^2 score	98.0230%	99.9088%

The EVS scores indicate that 98.0328 percent of the variations of the daily cases and 99.9088 percent of the variations of the cumulative cases can be explained by the ANN model.

MAE values reveal that the average size of the errors in daily case is 1.8301% and the average size of the errors in cumulative case is 0.6005%.

MAPE values show that the mean absolute percentage of error in daily case is 5.4951% and the mean absolute percentage of error in cumulative case is 1.7907%.

MSE values reveals that the mean square error of 0.0641 percent in daily cases and 0.0052 percent in cumulative cases.

MSLE valued indicates that the ratio of between the predicted and the actual values in daily cases is 0.0306% and the ratio of between the predicted and the actual values in cumulative cases is 0.0023%.

RMSE values indicate that the root mean squared error in daily cases is 2.5320% and the root mean squared error in cumulative cases is 0.7264%.

r^2 score shows that the agreement of the predicted and the actual values in daily cases is 98.0230% while the agreement of the predicted and the actual values in cumulative cases is 99.9088%.

Consequently, according to these results it is clear that ANN algorithm shows superior prediction performance on the cumulative case dataset.

Table 4.25 gives the CPU time of the prediction analysis on daily and cumulative deaths using ANNs. When the analysis times of daily death and cumulative death are examined, it can be seen that the ANN algorithm spends close time for both data forms, but this time difference can be ignored with a MAPE value of 0.34033 percent in the cumulative death data analysis.

Table 4. 25 CPU time of ANN death predictions

CPU time (s)	Daily deaths	Cumulative deaths
	14.6800636	15.1888473

Table 4.26 shows the prediction performance of the ANN. Note that, daily deaths and cumulative deaths are evaluated with respect to predetermined performance metrics which are calculated using the equations (1), (2), (3), (4), (5), (6), and (7).

Table 4. 26 Performance results of ANN predictions

Metrics	Daily deaths	Cumulative deaths
EVS	94.4638%	99.9945%
MAE	4.0708%	0.1337%
MAPE	11.3272%	0.34033%
MSE	0.3226%	0.0003%
MSLE	0.1386%	0.0001%
RMSE	5.6798%	0.1822%
r^2 score	90.2655%	99.9945%

EVS values reveal that 94.4638 percent of variations in the daily deaths and 99.9945 percent of the variations in the cumulative deaths can be explained by the ANN algorithm.

According to the MAE value, average size of the errors in daily deaths is 4.0708 percent and average size of the errors in cumulative deaths is 0.1337 percent.

MAPE values reveal, the mean absolute percentage of error in daily deaths is 11.3272 percent, while the mean absolute percentage of error in cumulative deaths is 0.34033 percent. So, we can infer that the accuracy of the prediction model with ANN algorithm is quite high especially for cumulative death data.

The mean square error in daily fatalities is 0.3226 percent, whereas the mean square error in cumulative deaths is 0.0003 percent. The MSE value for cumulative deaths is quite lower than that of daily deaths which means one should get better predictions for cumulative deaths data.

The MSLE number indicates that the ratio of predicted and actual value in daily deaths is 0.1386% while the ratio of predicted and actual value in cumulative deaths is 0.0001%.

The root mean squared error in daily deaths is 5.6798%, whereas the root mean squared error in cumulative deaths is 0.1822%, according to the RMSE value.

r^2 score shows that the agreement of the predicted and the actual values in daily deaths is 90.2655% while the agreement of the predicted and the actual values in cumulative deaths is 99.9945%.

Overall, considering the values of performance metrics ANN algorithm shows more superior performance when the number of cumulative deaths is used as an input to the model.

5. CONCLUSIONS

Coronavirus has become a severe problem all around the world. Many countries including Turkey suffer from the virus. Because of the virus, many areas of life have been paralyzed. In such emergency situations, the predictions related to spread of the virus played crucial role to take precautions. In this thesis, a comprehensive study using Turkey Covid-19 data to get accurate and robust predictions is conducted. After an extensive survey of the literature, it is seen that many different algorithms utilized in many countries of the world.

Within the scope of the study, ANN algorithm and boosting algorithms which are AdaBoost, CatBoost, and XGBoost are determined to make predictions. Hyperparameters of ANN optimized by using grid search algorithm. Predictions made using number of daily cases, number of cumulative cases, number of daily deaths, and number of cumulative deaths data from Turkey. Results reveal that using cumulative data as input all the considered algorithms' prediction performance increase. The boosting algorithms results were first evaluated in them and the CatBoost algorithm gave more successful results when results daily and cumulative cases were taken into account.

CatBoost algorithm performs better daily and cumulative prediction performance among the boosting algorithms. Also, it is observed that optimized ANN's prediction performance is very close to CatBoost algorithm. For example, for the cumulative case, MSE value with the CatBoost algorithm is 0.0014%, while MSE value with ANN algorithm is 0.0052%. When the same algorithms utilized on cumulative death data, the MSE value with CatBoost algorithm is 0.0011%, while MSE value with the ANN algorithm is 0.0003%.

In conclusion, considering all prediction results it is apparent that accurate and robust predictions achieved by using artificial intelligence techniques on Turkey coronavirus data.

6. RECOMMENDATIONS

This thesis, which we conducted, using data from Turkey, is quite comprehensive and pioneering. There are plenty of predictions in the literature. However, boosting algorithms have become popular to use prediction day-by-day. In this thesis presents a satisfactory study which benefits from boosting algorithms. By taking this study as a reference, the study can be expanded and a predictive analysis can be performed to measure the impact of the vaccine in the Covid-19 epidemic with artificial intelligence techniques.

REFERENCES

Alaloul. W. S., & Qureshi, A. H. (2020). Data processing using artificial neural network. In *Dynamic Data Assimilation-Beating the Uncertainties*. IntechOpen.

Ala'raj, M., Majdalawieh, M., & Nizamuddin, N. (2021). Modeling and forecasting of COVID-19 using a hybrid dynamic model based on SEIRD with ARIMA corrections. *Infectious Disease Modelling, 6*, 98–111. https://doi.org/10.1016/j.idm.2020.11.007

Aljameel, S. S., Khan, I. U., Aslam, N., Aljabri, M., & Alsulmi, E. S. (2021). Machine Learning-Based Model to Predict the Disease Severity and Outcome in COVID-19 Patients. *Scientific Programming, 2021*.

Al-Rakhami, M., Gumaei, A., Alsanad, A., Alamri, A., & Hassan, M. M. (2019). An ensemble learning approach for accurate energy load prediction in residential buildings. *IEEE Access, 7*, 48328–48338. https://doi.org/10.1109/ACCESS.2019.2909470

AL-Rousan, N., & AL-Najjar, H. (2020). Data analysis of coronavirus COVID-19 epidemic in South Korea based on recovered and death cases. *Journal of Medical Virology*, *92*(9), 1603–1608. https://doi.org/10.1002/jmv.25850

Amar, L. A., Taha, A. A., & Mohamed, M. Y. (2020). Prediction of the final size for COVID-19 epidemic using machine learning: a case study of Egypt. *Infectious Disease Modelling*, *5*, 622–634. https://doi.org/10.1016/j.idm.2020.08.008

Arora, P., Kumar, H., & Panigrahi, B. K. (2020). Prediction and analysis of COVID-19 positive cases using deep learning models : A descriptive case study of India. *Chaos, Solitons and Fractals, 139*, 110017. https://doi.org/10.1016/j.chaos.2020.110017

Ayyoubzadeh, S. M., Ayyoubzadeh, S. M., Zahedi, H., Ahmadi, M., & Kalhori, S. R. N. (2020).

Predicting COVID-19 incidence through analysis of google trends data in Iran: data mining and deep learning pilot study. *JMIR public health and surveillance, 6(2),* e18828.

Behl, R., & Mishra, M. (2020). COVID-19 lifecycle: Predictive modelling of states in India. *Global Business Review, 21*(4), 883–891. https://doi.org/10.1177/0972150920934642

Breiman, L. (1996). Bagging predictors. *Machine learning, 24*(2), 123-140.

Ceylan, Z. (2020). Estimation of COVID-19 prevalence in Italy, Spain, and France. *Science of the Total Environment, 729,* 138817. https://doi.org/10.1016/j.scitotenv.2020.138817

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).

Chimmula, V. K. R., & Zhang, L. (2020). Time series forecasting of COVID-19 transmission in Canada using LSTM networks. *Chaos, Solitons and Fractals, 135,* 109864. https://doi.org/10.1016/j.chaos.2020.109864

Copeland, B. (2020, August 11). artificial intelligence. *Encyclopedia Britannica*. https://www.britannica.com/technology/artificial-intelligence

Dehesh, T., Mardani-Fard, H. A., & Dehesh, P. (2020). Forecasting of covid-19 confirmed cases in different countries with arima models. *MedRxiv*, 1–12. https://doi.org/10.1101/2020.03.13.20035345

Dey, S. K., Rahman, M. M., Siddiqi, U. R., & Howlader, A. (2020). Analyzing the epidemiological outbreak of COVID-19: A visual exploratory data analysis approach. *Journal of Medical Virology, 92*(6), 632–638. https://doi.org/10.1002/jmv.25743

Dorogush, A. V., Ershov, V., & Gulin, A. (2018). CatBoost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363.*

El Naqa, I., & Murphy, M. J. (2015). What is machine learning?. In *machine learning in radiation oncology* (pp. 3-11). Springer, Cham.

Eroğlu, Y. (2020). Forecasting models for Covid-19 cases of Turkey using artificial neural networks and deep learning. *Endüstri Mühendisliği,31*(3), 353–372.

Ferreira, A. J., & Figueiredo, M. A. (2012). Boosting algorithms: A review of methods, theory, and applications. *Ensemble machine learning*, 35-85.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences, 55*(1), 119-139.

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence, 14*(771-780), 1612.

Guerrero, D. (2020). Spread of Covid-19: a study case of honduras, forecasting with logistic model and SIR model. *UpnfmEduHn*, 1–13.

Gumaei, A., Al-Rakhami, M., Al Rahhal, M. M., Albogamy, F. R., Al Maghayreh, E., & AlSalman, H. (2021). Prediction of COVID-19 confirmed cases using gradient boosting regression method. *Comput Mater Continua, 66,*315-329.

Gupta, A. K., Singh, V., Mathur, P., & Travieso-Gonzalez, C. M. (2021). Prediction of COVID-19 pandemic measuring criteria using support vector machine, prophet and linear regression models in Indian scenario. *Journal of Interdisciplinary Mathematics, 24*(1), 89-108. https://doi.org/10.1080/09720502.2020.1833458

Gupta, A., Salau, A. O., Chaturvedi, P., Akinola, S. A., & Nwulu, N. I. (2019, April). Notice of

Violation of IEEE Publication Principles; Artificial Neural Networks: Its Techniques and Applications to Forecasting. In *2019 International Conference on Automation, Computational and Technology Management (ICACTM)* (pp. 320-324). IEEE.

Hintze, J. L., & Nelson, R. D. (1998). Violin plots: a box plot-density trace synergism. *The American Statistician, 52*(2), 181-184.

Honigsbaum, M. (2009). Pandemic. *The Lancet, 373*(9679), 1939.

Hussain, A. A., Bouachir, O., Al-Turjman, F., & Aloqaily, M. (2020). Al techniques for COVID-19. *IEEE Access, 8*, 128776-128795.

Ilie, O. D., Cojocariu, R. O., Ciobica, A., Timofte, S. I., Mavroudis, I., & Doroftei, B. (2020). Forecasting the spreading of COVID-19 across nine countries from Europe , Asia , and the American continents using the ARIMA models. *Microorganisms, 8*(8), 1158.

Iwendi, C., Bashir, A. K., Peshkar, A., Sujatha, R., Chatterjee, J. M., Pasupuleti, S., … & Jo, O. (2020). COVID-19 patient health prediction using boosted random forest algorithm. *Frontiers in public health, 8*, 357. https://doi.org/10.3389/fpubh.2020.00357

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer, 29*(3), 31-44.

Kırbaş, İ., Sözen, A., Tuncer, A. D., & Kazancıoğlu, F. Ş. (2020). Comparative analysis and forecasting of COVID-19 cases in various European countries with ARIMA, NARNN and LSTM approaches. *Chaos, Solitons and Fractals, 138,* 110015. https://doi.org/10.1016/j.chaos.2020.110015

Liu, J., Wu, J., Liu, S., Li, M., Hu, K., & Li, K. (2021). Predicting mortality of patients with acute kidney injury in the ICU using XGBoost model. *Plos one, 16*(2), e0246306.

Liu, Z., Magal, P., Seydi, O., & Webb, G. (2020). Predicting the cumulative number of cases for the COVID-19 epidemic in China from early data. *arXiv preprint arXiv:2002. 122298.*

Lounis, M., & Bagal, D. K. (2020). Estimation of SIR model's parameters of COVID-19 in Algeria. *Bulletin of the National Research Centre*, *44*(1), 1-6.

Luo, J., Zhang, Z., Fu, Y., & Rao, F. (2021). Time series prediction of COVID-19 transmission in America using LSTM and XGBoost algorithms. *Results in Physics*, *27*, 104462. https://doi.org/10.1016/j.rinp.2021.104462

Maital, S., & Barzani, E. (2020). The global economic impact of COVID-19: A summary of research. *Samuel Neaman Institute for National Policy Research*, *2020*, 1-12.

Menon, V. K. (2020). Prediction of number of cases expected and estimation of the final size of coronavirus epidemic in India using the logistic model and genetic algorithm. *arXiv preprint arXiv:2003. 12017.*

Mijwel, M. M. (2018). Artificial neural networks advantages and disadvantages. *Retrieved from Linkedln https//WWW. linkedin.com/pulse/artificial-neuralnet Work.*

Nesteruk, I. (2020). SIR-simulation of corona pandemic dynamics in Europe. *MedRxiv*. https://doi.org/10.1101/2020.04.22.20075135

Niazkar, H. R., & Niazkar, M. (2020). Application of artificial neural networks to predict the COVID-19 outbreak. *Global Health Research and Policy*, *5*(1), 1-11. https://doi.org/10.1186/s41256-020-00175-y

Nyuytiymbiy, K. (2020) Parameters and Hyperparameters in Machine Learning and Deep Learning. *Retrieved from https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac.*

Ogundokun, R. O., Lukman, A. F., Kibria, G. B., Awotunde, J. B., & Aladeitan, B. B. (2020). Predictive modelling of COVID-19 confirmed cases in Nigeria. *Infectious Disease Modelling*, *5*, 543–548. https://doi.org/10.1016/j.idm.2020.08.003

Osman, A. I. A., Ahmed, A. N., Chow, M. F., Huang, Y. F., & El-Shafie, A. (2021). Extreme gradient boosting (Xgboost) model to predict the groundwater levels in Selangor Malaysia. *Ain Shams Engineering Journal*, *12*(2), 1545–1556. https://doi.org/10.1016/j.asej.2020.11.011

Özdinç, M., Şenel, K., Öztürkcan, S., & Akgül, A. (2020). Predicting the progress of COVID-19: the case for Turkey. *Turkiye Klinikleri Journal of Medical Sciences*, *40*(2), 117–119. https://doi.org/10.5336/medsci.2020-75741

Pandey, G., Chaudhary, P., Gupta, R., & Pal, S. (2020). SEIR and Regression Model based COVID-19 outbreak predictions in India. *arXiv preprint arXiv:2004.00958.*

Parbat, D., & Chakraborty, M. (2020). A python based support vector regression model for prediction of COVID19 cases in India. *Chaos, Solitons and Fractals*, *138*, 3–7. https://doi.org/10.1016/j.chaos.2020.109942

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2017). CatBoost: unbiased boosting with categorical features. *arXiv preprint arXiv:1706.09516.*

Ragini, R. (2019, June 6). https://medium.com/@ruhi3929/bagging-and-boosting-method-c036236376eb

Rajesh, A., Pai, H., Roy, V., Samanta, S., & Ghosh, S. (2020). CoVID-19 prediction for India from the existing data and SIR(D) model study. *medRxiv*. https://doi.org/10.1101/2020.05.05.20085902

Rath, S., Tripathy, A., & Tripathy, A. R. (2020). Prediction of new active cases of coronavirus

disease (COVID-19) pandemic using multiple linear regression model. *Diabetes and Metabolic Syndrome: Clinical Research and Reviews*, *14*(5), 1467–1474. https://doi.org/10.1016/j.dsx.2020.07.045

Ray, S. (2017, August 14). https://www.analyticsvidhya.com/blog/2017/08/catboost-automated-categorical-data/

Rocca, J. (2019, Apr 23). https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

Roosa, K., Lee, Y., Luo, R., Kirpich, A., Rothenberg, R., Hyman, J. M., …& Chowell, G. B. (2020). Real-time forecasts of the COVID-19 epidemic in China from February 5th to February 24th, 2020. *Infectious Disease Modelling*, *5*, 256–263. https://doi.org/10.1016/j.idm.2020.02.002

Sahai, A. K., Rath, N., Sood, V., & Singh, M. P. (2020). ARIMA modelling & forecasting of COVID-19 in top five affected countries. *Diabetes and Metabolic Syndrome: Clinical Research and Reviews*, *14*(5), 1419–1427. https://doi.org/10.1016/j.dsx.2020.07.042

Şenel, K., Özdinç, M., Öztürkcan, S., & Akgül, A. (2020). Instantaneous R for COVID-19 in Turkey: estimation by Bayesian statistical inference. *Turkiye Klinikleri Journal of Medical Sciences*, *40*(2), 127–131. https://doi.org/10.5336/medsci.2020-76462

Singh, V., Poonia, R. C., Kumar, S., Dass, P., Agarwal, P., Bhatnagar, V., & Raja, L. (2020). Prediction of COVID-19 corona virus pandemic based on time series data using Support Vector Machine. *Journal of Discrete Mathematical Sciences and Cryptography*, *23*(8), 1583-1597. https://doi.org/10.1080/09720529.2020.1784535

Tandon, H., Ranjan, P., Chakraborty, T., & Suhag, V. (2020). Coronavirus (COVID-19): ARIMA based time-series analysis to forecast near future. *arXiv preprint arXiv:2004.07859*.

Tomar, A., & Gupta, N. (2020). Prediction for the spread of COVID-19 in India and effectiveness of preventive measures. *Science of the Total Environment, 728*, 138762. https://doi.org/10.1016/j.scitotenv.2020.138762

Uçar, A., Arslan, Ş., Manap, H., Gürkan, T., Çalışkan, M., Dayıoğlu, A., Efe, H., Yılmaz, M., İbrahimoğlu, A., Gültekin, E., Durna, R., Başar, R., Osmanoğlu, F. & Ören, S. (2020). *Türkiye'de Covid-19 Pandemisinin Monitörizasyonu Için Interaktif Ve Gerçek Zamanlı Bir Web Uygulaması: TURCOVID19. Anatolian Clinic the Journal of Medical Sciences, Anadolu Kliniği Tıp Bilimleri Dergisi (COVID19 Özel Sayısı), 154-155.* DOI: 10.21673/anadoluklin.726347

Wang, P., Zheng, X., Ai, G., Liu, D., & Zhu, B. (2020). Time series prediction for the epidemic trends of COVID-19 using the improved LSTM deep learning method: Case studies in Russia , Peru and Iran. *Chaos, Solitons & Fractals, 140*, 110214. https://doi.org/10.1016/j.chaos.2020.110214

World Health Organization (WHO), Past Pandemics, https://www.euro.who.int/en/health-topics/communicable-diseases/influenza/pandemic-influenza/past-pandemics

XGBoost. (2020, September 9). In *Vikipedia.* https://en.wikipedia.org/wiki/XGBoost.

Yadav, R. S. (2020). Mathematical Modeling and Simulation of SIR Model for COVID-2019 Epidemic Outbreak: A Case Study of India. *INFOCOMP Journal of Computer Science, 19*(2), 01–09. https://doi.org/10.1101/2020.05.15.20103077

Yan, L., Zhang, H. T., Xiao, Y., Wang, M., Guo, Y., Sun, C., … & Yuan, Y. (2020). Prediction of criticality in patients with severe Covid-19 infection using three clinical features: a machine learning-based prognostic model with clinical data in Wuhan. *MedRxiv.*

Yousaf, M., Zahir, S., Riaz, M., Hussain, S. M., & Shah, K. (2020). Statistical analysis of forecasting COVID-19 for upcoming month in Pakistan. *Chaos, Solitons & Fractals, 138*,

109926.

Zhang, Y., & Haghani, A. (2015). A gradient boosting method to improve time travel prediction. *Transportation Research Part C: Emerging Technologies, 58*, 308-324.

Zhou, Z. H. (2019). *Ensemble methods: foundations and algorithms.* Chapman and Hall/CRC.

Zhu, J. S., Ge, P., Jiang, C., Zhang, Y., Li, X., Zhao, Z., … & Duong, T. Q. (2020). Deep-learning artificial intelligence analysis of clinical variables predicts mortality in COVID-19 patients. *Journal of the Amerikan College of Emergency Physicians Open, 1*(6), 1364-1373.