Covid vaccines analysis(Time series)

Abstract

We investigated and predicted the future environmental circumstances of COVID-19 to minimize its effects using artificial intelligence techniques. The experimental investigation of COVID-19 instances has been performed in ten countries, including India, the United States, Russia, Argentina, Brazil, Colombia, Italy, Turkey, Germany, and France using machine learning, deep learning, and time series models. The confirmed, deceased, and recovered datasets from January 22, 2020, to May 29, 2021, of Novel COVID-19 cases were considered from the Kaggle COVID dataset repository. The country-wise Exploratory Data Analysis visually represents the active, recovered, closed, and death cases from March 2020 to May 2021. The data are pre-processed and scaled using a MinMax scaler to extract and normalize the features to obtain an accurate prediction rate. The proposed methodology employs KNN or K means clustering and Time series analysis of data.

Introduction

The aim is to conduct a Covid_vaccines analysis on the dataset given on Kaggle and to perform a time series analysis on the given dataset. We have done all required analysis and investigation to complete the process. The code was written in Python programming language over a platform called Google Collab. The program is also attached to this document which is done by our team for Naan Mudhalvan project submission. We uploaded the “ipynb” file which contains the code and it was also uploaded to the GitHub account for the Evaluators. We were provided the dataset with the code file to the GitHub account.

Procedure

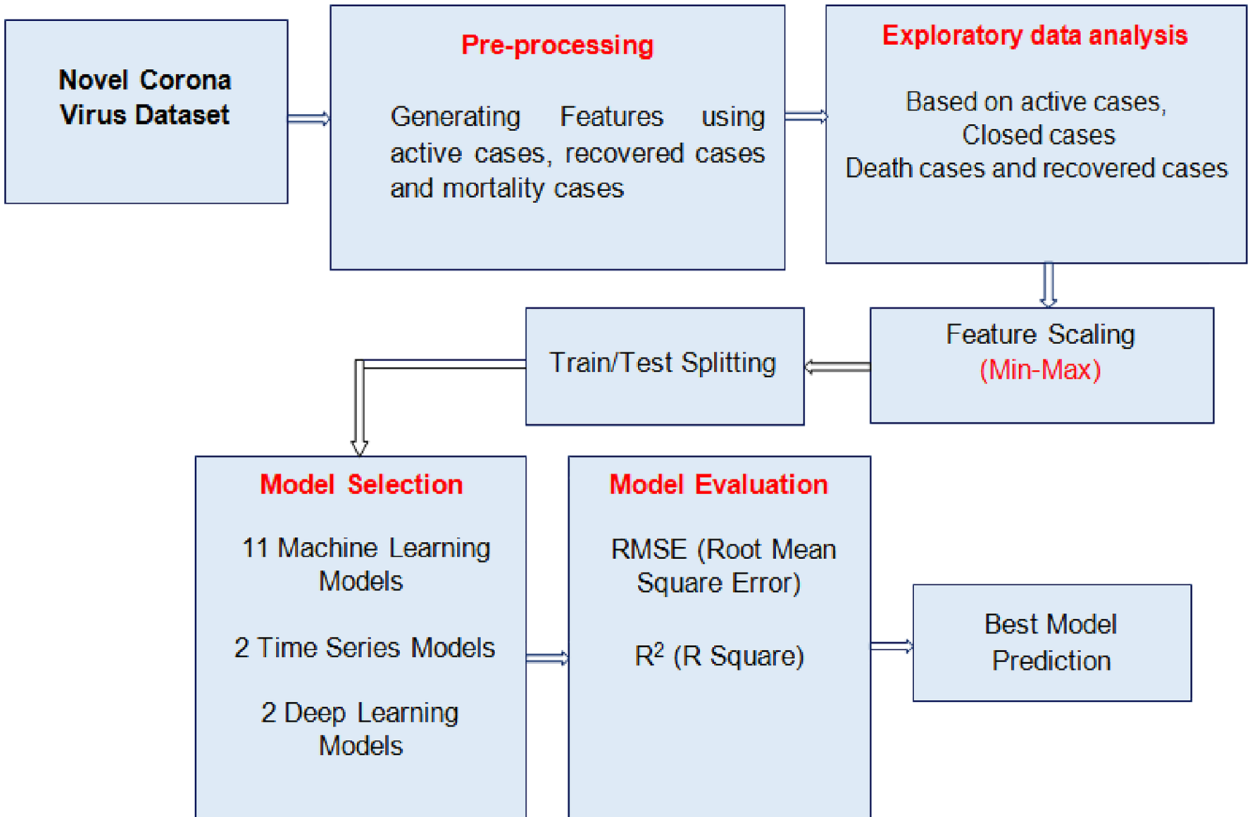
The overall goal of this research is to build models that can calculate two necessary evaluative measures: RMSE and R2 Score for confirmed, death, and recovered cases from ten different nations to help future forecasts. The steps are as follows:

Step 1: Initially, data are pre-processed to capture characteristics utilizing various variables, such as active cases, recovered cases, and COVID-19 fatality cases.

Step 2: Exploratory Data Analysis of COVID-19's active cases, closed cases, confirmed cases, recovered cases, and death cases are calculated to summarize or interpret the information that is hidden in rows or columns, and scaling techniques such as Min–Max have been applied to normalize each feature that is obtained from these attributes.

Step 3: Later, utilizing confirmed cases, recovered cases, and death cases from 10 different nations, the gathered data were used to anticipate the future conditions of a new CoronaVirus. To get the findings, several machine learning models, time series models, and deep learning models were used, and they were assessed using parameters, such as root-meansquare error and R square.

Step 4: Finally, all of the results have been ranked to choose the technique with the lowest root-mean-square error and the highest R-squared score.



Dataset

which the row is updated for the given province or country, the cumulative number of confirmed cases, the cumulative number of death cases, and the cumulative number of recovered patients from January 22, 2020, to May 29, 2021. The confirmed, dead, and recovered casesThis dataset is compiled daily to offer recent news on new coronavirus infections, fatalities, and recoveries for 2019. The data will be available from January 22, 2020, to May 29, 2021. This is a time series dataset with a total of 1248 time series datasets recorded for each day, while the count of time series datasets registered for each day indicates the cumulative total. The dataset contains a serial number, the observation date in the format MM/DD/YYYY, the province or state of observation, the country or region of compliance, and the time in UTC from ten different countries are included in this document.

Libraries

Several Python-based libraries, such as Pandas—a Python-based software toolkit that contains data structures and strategies for working with numerical tables and time series—were imported during the prediction of COVID-19 confirmed, death, and recovered cases, and Numpy— a Python array manipulation library. It also contains functions for working with linear algebra, the Fourier transform, and matrices, among other things. Matplotlib—a cross-platform data visualization and graphical plotting program built in Python for use with NumPy; Seaborn—a Python data visualization software based on Matplotlib. It uses a high-level interface to generate aesthetically beautiful and functional data visualizations, Plotly is a Python library that makes it easier to create professional-looking visualization by providing a flexible, open-source charting toolkit with over 40 chart types for a wide range of statistical, financial, geographic, scientific, and 3D use cases. Date–time is a module that mixes date and time and characteristics like the year, month, day, hour, minute, second, microsecond, and info. Sklearn—Scikit-learn is the most helpful Python machine-learning package.

Techniques

The pre-processing approach used to extract the characteristics is covered in the part of this work. This part also discusses the exploratory data analysis of the cleaned data, which is followed by the scaling approach. Following that, a section was presented in which many models from the COVID-19 testing dataset were described and shown.

Pre-processing

Data collected from the covid_vaccines 19 datasets have been pre-processed using various mathematical formulas, such as active cases, percentage of recovery rate, percentage of mortality, and week of days to generate features. There is a significant likelihood that the number of active topics has increased since some of the confirmed patients are now dead, and fewer new cases are being found. To calculate it, use Eq. (1). The recovery rate is the proportion of recovered instances, while the mortality rate is the percentage of death cases. Equations (2), and (3) display the formulas. The last parameter, the week of days, is calculated by importing the library named WEEKOFYEAR.

Formulae

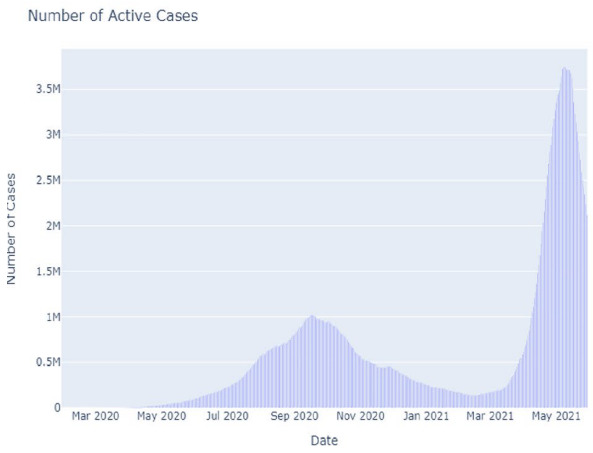
Active cases = Total number of confirmed cases − (total number of recovered cases + total number of death cases, (1)

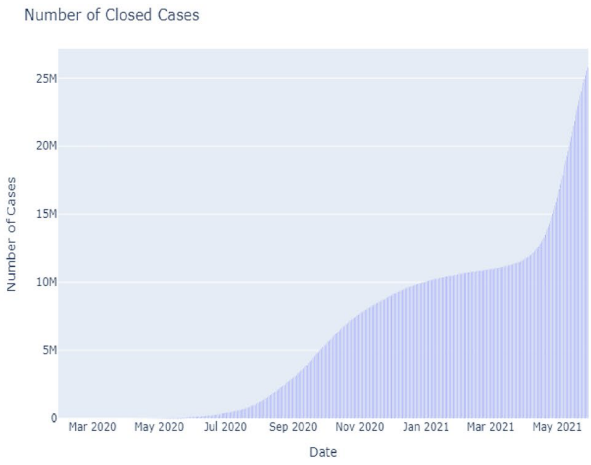
Recovery rate = (2)Number of recovered cases/number of confirmed cases * 100,

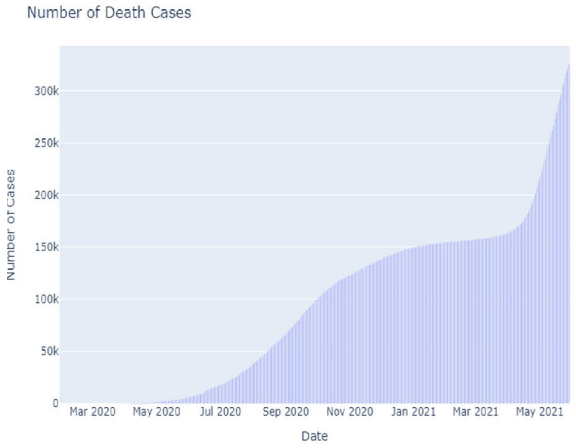
Mortality rate = (3)Number of death cases/Number of confirmed cases * 100.

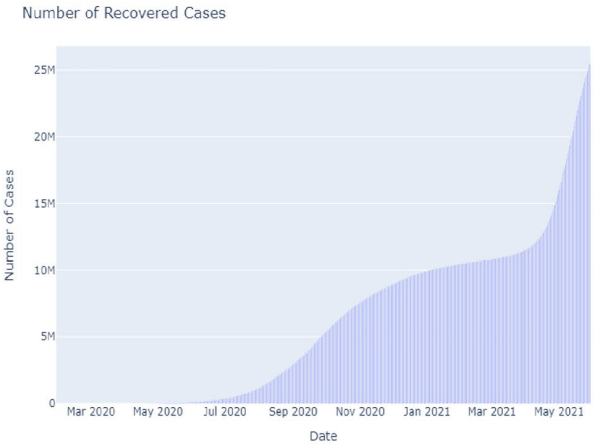
Exploratory Data Analysis

Exploratory Data Analysis is a vital process that entails performing preliminary analyses of data to uncover patterns, identify anomalies, test hypotheses, and verify assumptions using summary statistics and graphical representations. Some of the critical steps in exploratory data analysis are importing the data set in which we will get two data frames; one consisting of the data to be trained and the other for predicting the target value, identifying the number of features and columns, identifying the qualities or cues, identifying the data types of components, identifying the number of observations, checking if the dataset has empty cells or samples, identifying the number of empty cells by features or columns, and exploring categorical features. This work employed an exploratory analysis of ten different countries after pre-processing to assess their features via statistical graphs. The figures shown below depict the graphical analysis of active cases, death cases, closed points, and recovered cases that have been recorded from Jan 2020 to May 2021.









Feature Scaling

Normalizing the range of independent variables or features of data using feature scaling is a feature scaling approach. Min–Max scaling technique has been used to perform normalization on the parts obtained during data pre-processing. The Min–Max Normalization or Min–Max Scaling technique creates a scale that goes from 0 to 1 or from 1 to − 1. Deciding on a range of data to aim for relies on the type of data you are working with. Min–Max for the range[0,1] can be computed using Eq. (4)

Formulae

$x = x - \min(x)/\max(x) - \min(x)$ —--------(4)

After normalization, the data were split into two subsets: the training set, which would be used to assess machine learning methods, and the testing set, which would be used to evaluate deep learning techniques. It applies to issues involving classification or regression, as well as to any supervised learning technique. Following data partitioning, the first subset is utilized to fit the model; this is the training dataset. The second subset is used as an input element in the dataset supplied to the model, and predictions and comparisons to predicted values are performed. The test dataset is the second dataset. In a nutshell, the train data set is used to fit the machine learning model, while the test data set is used to verify the fit. The goal is to assess the performance of time series, machine learning, and deep learning models on new data. The most often used split percentages are as follows:

80% training, 20% testing.
67% training, 33% testing.
50% training, 50% testing.

Model selection

We have chosen the Time series model named the prophet model to perform analytics and the result visualization.

K Nearest Regressor

Non-parametric regression involves averaging nearby observations to determine if one or more independent variables are associated with a continuous result. For an analysis to be effective, the size of the neighborhood should be selected by the analyst. However, in some cases, it can be randomized to reduce the mean squared error. An algorithm that considers the K-nearest neighbor numerical objective is utilized to determine the average of the K target values.

KNN regression and KNN classification both utilize the same distance functions. KNN regression uses the same distance functions as KNN classification. The formulae to compute K-nearest regressor are shown in Eqs. (8)–(10)

$$\text{Euclidean formula} : \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}, \quad (8)$$

$$\text{Manhattan formula} : \sum_{i=1}^{k} |x_i - y_i|, \quad (9)$$

$$\text{Minkowski formula} : \left[\sum_{i=1}^{k} \left(|x_i - y_i|\right)^q\right]^{\frac{1}{q}}. \quad (10)$$

Kernel Ridge Regressor

Bayesian Regressor

Bayesian Regressor is a regression approach that uses Bayesian inference to do statistical analysis. This method enables a natural process to persist in the presence of limited or poorly dispersed data. It generates predictions based on the posterior probability of all feasible regression weights. With Bayesian Linear Regression, the aim is not to choose the "best" model parameter but to estimate the distribution of model parameters [39]. It is demonstrated by Eq. (25)

P(y, X) = P(y, X) × P(X) / P(yX) —----------(25)

Here, P(β|y, X) is the posterior probability distribution of the model parameters given the inputs and outputs. This is equal to the likelihood of the data, P(β|y, X), multiplied by the prior probability of the parameters and divided by a normalization constant.

Time Series Models: Facebook Prophet

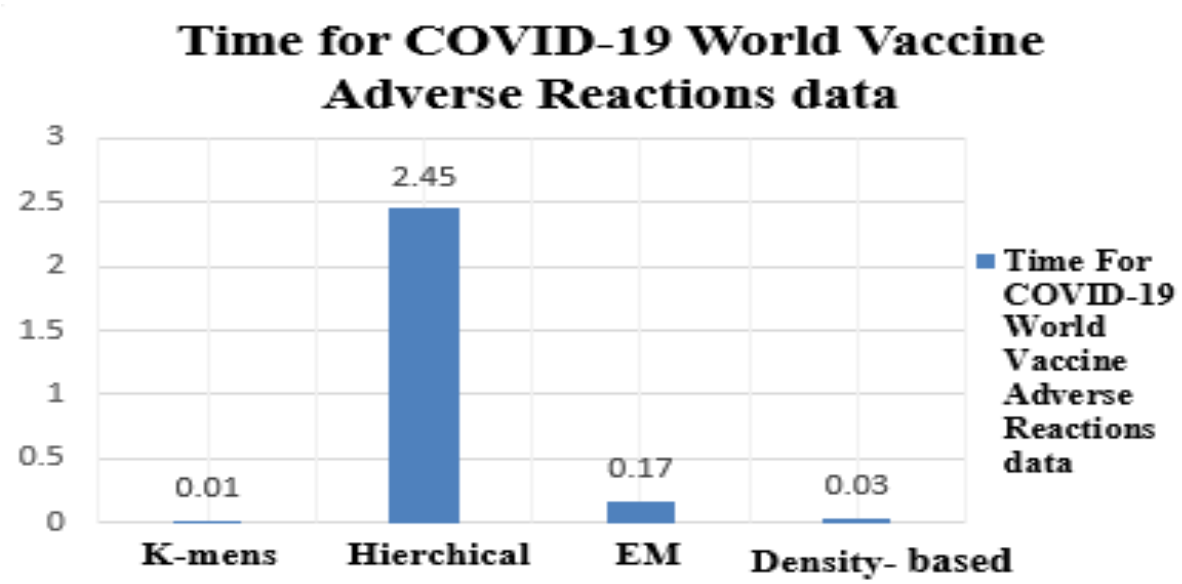
A forecasting approach based on an additive model known as a prophet is used to correlate nonlinear trends with seasonal and holiday impacts as well as yearly, weekly, and daily patterns. Time series with strong seasonal influences and extensive historical

data spanning many seasons do well with this approach. The Prophet works well with outliers, which makes it resistant to data and trend shifts. The time series model is built on a prophet, and it is fast, fully automated, and very exact. The trend, seasonality, and holidays from our time series model, which we break down into three key components: trend, seasonality, and holidays. They are merged in Eq. (5) as follows:

$$y(t) = g(t) + s(t) + h(t) + \in t, \text{ —----------(5)}$$

g(t): For modeling non-periodic changes in time series, a piecewise linear or logistic growth curve is used. s(t): changes on a regular basis (e.g., weekly/yearly seasonality).
h(t): The impact of vacations (supplied by the user) on individuals with irregular schedules. εt: The error term is used to account for any unforeseen changes that the model does not account for.

Time series results for adverse effects



K-means clustering algorithm for adverse effects of COVID vaccines

k-means algorithm is a clear partition process. It separated (N) data objects into (K) cluster sets to obtain low cross-similarity and high intracluster specificity. The mean diameter of a cluster's points, also known as the cluster's centroid, is used to evaluate cluster similarity [36,37,38]. It goes like this: first, choose k points at random as the cluster's mean (center). Following that, all artifacts are allocated to the k clusters with the shortest Distance measured between their centroids and objects. Mean is revised before it has been allocated to all of the artifacts. This update is continuing until the allocation is secure. Steps:

Step 1: Get started: As initial centers, select k data items at random from Data collection D. K is the number of clusters.
Step 2: Repetition:
a) Assume that each cluster is a centroid.
b) Evaluate the difference between all of the data points, as well as the centroids.
c) Allocate di to the cluster that is nearest to you as a data object.
Step 3: By each cluster j (1=j=k), create an update. Calculate the cluster center once more.
Step 4: Repeat until there is no difference in the cluster's core.
Step 5: Finish.

Performing K-means clustering on the topic of COVID-19 vaccine adverse effects involves using the K-means algorithm to group similar cases of adverse effects together based on certain features or attributes. Below, I'll provide a detailed step-by-step guide for conducting K-means clustering on this topic:

Step 1: Data Preparation

Gather a dataset that includes information about adverse effects reported for different COVID-19 vaccines. Your dataset should contain relevant features such as symptoms, demographics of patients, vaccine types, dosage, and timing of adverse events. Preprocess the data by cleaning it (removing duplicates, handling missing values, and outliers) and normalizing or standardizing the features if necessary.

Step 2: Feature Selection

Choose the features that you want to use for clustering. In the context of COVID-19 vaccine adverse effects, you may consider features such as specific symptoms, patient age, gender, vaccine manufacturer, and the time interval between vaccination and the onset of symptoms.

Step 3: Determine the Number of Clusters (K)

One of the critical decisions in K-means clustering is selecting the number of clusters (K). You can use techniques like the elbow method or silhouette analysis to determine an appropriate value for K. These methods help you identify the point at which adding more clusters doesn't significantly improve the clustering quality.

Step 4: Choose an Initialization Method

K-means is sensitive to the initial placement of centroids. You can choose between random initialization and more advanced methods like K-means++ to improve the chances of finding a good solution.

Step 5: Apply K-means Clustering

Run the K-means algorithm with the selected number of clusters (K) on your preprocessed data. The algorithm will assign each data point (representing an adverse event) to one of the K clusters.

Step 6: Interpretation of Clusters

Analyze the results of the clustering. You can do this in several ways:
Visualize the clusters: Use scatter plots or other visualization techniques to visualize how data points are grouped into clusters. Examine cluster centroids: For each cluster, look at the centroid or mean values of the features to understand what characterizes that cluster. This can help identify commonalities among adverse effects in each cluster. Explore cluster statistics: Calculate statistics for each cluster, such as the most frequently reported symptoms, the average age of patients, or the predominant vaccine types.

Step 7: Evaluate and Refine

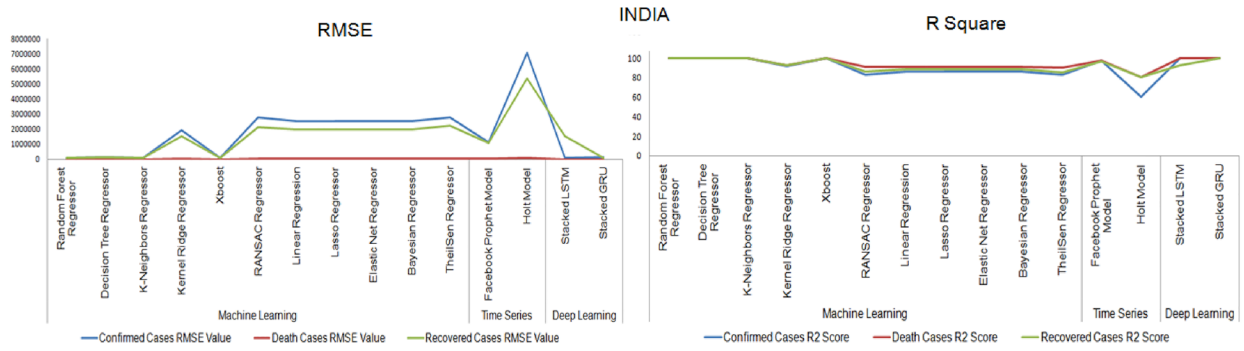
Evaluate the quality of your clusters using metrics like the silhouette score or Davies-Bouldin index. These metrics can help assess the separation and cohesion of clusters. If the results are not satisfactory, you can experiment with different values of K or try alternative clustering algorithms to see if they yield better results.

Step 8: Interpretation and Reporting

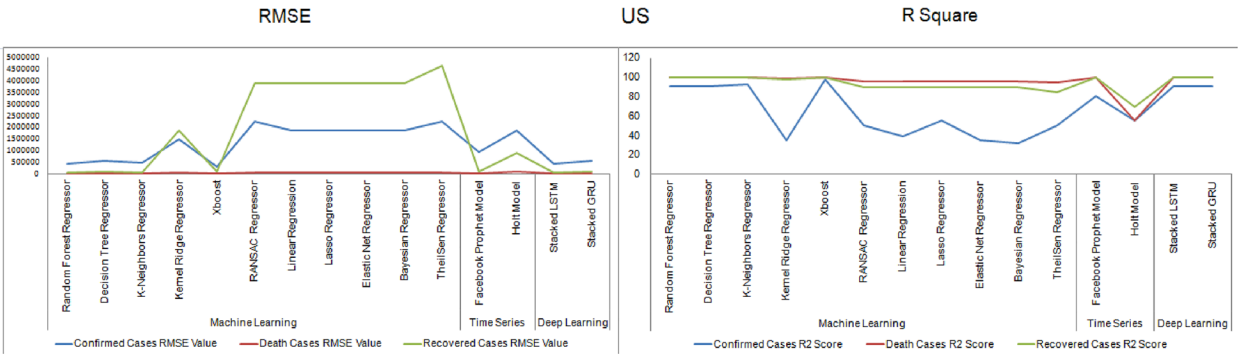
Interpret the clusters and their characteristics. This might involve identifying clusters with distinct patterns of adverse effects, such as a cluster with mostly mild symptoms or another with rare but severe adverse events. Report your findings in a clear and informative manner. Describe the characteristics of each cluster and any insights gained from the analysis. Communicate your results with relevant stakeholders, such as health authorities or medical professionals.

Step 9: Continuous Monitoring

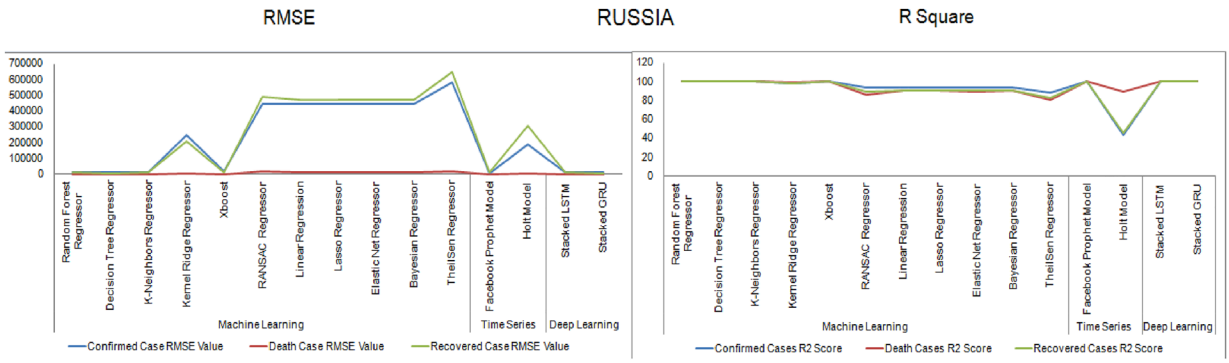
COVID-19 vaccine adverse effects may change over time as more data becomes available. Periodically update your dataset and re-run the clustering analysis to detect emerging patterns or changes in adverse effects.



(a)



(b)



(c)

Conclusion

Vaccines against COVID-19 (and its emerging variants) are a critical global intervention in the present pandemic situation. Vaccines also cause side effects. Occurring Adverse Event 139. Following Immunization (AEFI)for most doses of vaccines. Therefore, we retrieved data sets on COVID-19 world vaccine adverse reactions and processed this information using four algorithms found in machine learning. It is evident that machine learning can significantly improve the development of the covid-19 pandemic. Machine learning unsupervised type clustering algorithm is used to accurately assess the issues with the vaccines. In this paper, a comparison of the clustering algorithms k-means, expectation-maximization, heretical, and density-based was carried out. The COVID-19 world vaccine adverse reaction data were compared using the Weka tool, with the comparable outcomes in table and graph format the study is carried out on a time- and accuracy-based basis. After we applied our dataset on Weka software the results show that the simple k-means clusters type clustering algorithms with a lower time and higher accuracy than most other algorithms.