# A Comparative Analysis and Illustrative Study of Machine Learning Models for COVID-19 Prediction in India

**Aakanksha Patil[1], Urvashi Thakre[1], Mohd Zaki Aftab[2], Shrusti Warekar[1]**

*[1]Student, Dept. of Computer Engineering, Rajiv Gandhi Institute of Technology, Maharashtra, Mumbai*
*[2]Student, Dept. of Mechanical Engineering, NMIMS University, Maharashtra, Mumbai*

---------------------------------------------------------------------***---------------------------------------------------------------------

**ABSTRACT:** *COVID-19, also known as 2019-nCoV, is no longer a pandemic, but rather an endemic virus. The coronavirus disease has spread to a great extent globally and has reshaped our lives. More than 250 countries have been affected impetuously. India is one of the most affected countries with more than 28,694,879 cases and over 344,101 deaths. Scientists are baffled by how quickly the disease is spreading through India. Since early March, the daily case count has skyrocketed. India's high numbers have also aided in increasing the worldwide case count, which has nearly surpassed the January record. COVID-19 case counts began to decline in India in September, after peaking at roughly 100,000 infections per day. However, they clambered again in March, reaching a new apex that is more than double the previous one. "The second wave has made the first appear like a ripple in a bathtub," says Zarir Udwadia, a pulmonary medicine clinician-researcher at P D Hinduja Hospital.*

*COVID-19 is beyond a data science problem; it is a massive public health issue that has resulted in many deaths and is casting a harsh light on how we structure our society when it comes to important things like healthcare availability and affordability. We must nevertheless approach this problem from a data-science perspective. This paper presents the analysis and prediction of COVID-19 using various efficient machine learning models. The data has been collected for the period 30th January 2020 to 15th April 2021 (441 days). Machine Learning algorithms were used in this work to anticipate the spread of the new covid-19 virus. To get a better understanding of the current situation, we utilized a total of seven supervised machine learning models: linear regression, logistic regression, polynomial regression, support vector regression (SVR), random forest, decision trees, and multi-layer perceptron regressor (MLP). In this study, we discovered that these models could assist Indian doctors and the government in planning for the upcoming days. Further, these models can be tuned for forecasting over long-term intervals.*

**Keywords: Machine Learning, Data Science, Covid-19, Exploratory Data Analysis, Pandemic, Deep Learning.**

## 1. INTRODUCTION

The year 2020 would be looked back on as a catastrophic year for humankind on this planet earth. Coronavirus has affected folks all over the world. In humans, coronaviruses can cause respiratory disease. The virus gets its name, "corona," from the numerous crown-like spikes on its surface. Coronaviruses that cause disease in humans include severe acute respiratory syndrome (SARS), the Middle East respiratory syndrome (MERS), and the common cold. The novel strain of coronavirus, COVID-19, was first reported in Wuhan, China in December 2019. Since then, the virus has spread to every continent (except Antarctica). Covid-19 has caused millions of fatalities around the world as well as lasting health problems in some who survived the illness. It spreads from person to person, from an infected person's mouth or nose in small liquid particles when they cough, sneeze, speak, sing, or breathe. These particles range from larger respiratory droplets to smaller aerosols. **[1]**

The Coronavirus Disease 2019 (COVID-19) was declared as a global pandemic by World Health Organization (WHO) on 11-Mar-20. The disease has severely impacted globally across 188 countries and territories, with the confirmed cases count reaching 174.9 million. The disease was first detected in India on 30-Jan-20. Currently, India has the largest number of confirmed cases in Asia with 29.6 million reported cases of COVID-19 infection. The second wave, which began in March 2021, was significantly greater than the first, with shortages of vaccinations, hospital beds, oxygen cylinders, and other medications in several regions of the country. By the end of April, India had surpassed the rest of the globe in terms of new and active cases. It became the first country to report over 400,000 new covid-19 cases in 24 hours on April 30, 2021. Given the growing threat of COVID-19, it is critical to investigate and identify techniques and resources that can forecast the predicted number of COVID-19 cases. How will the curve of covid-19 cases slant be the foremost concerning issue for the government of India?

Machine Learning is a data analysis approach that automates the creation of analytical models. In this field of artificial intelligence, systems are taught to learn from data, detect patterns, and make choices with little or no human interaction. We think that reliable data sets and the precise use of machine learning models can assist us in predicting the real statistics that are the need of the hour.

In this work, we present several predictions based on India's Confirmed, Active, and Death Cases responding to COVID-19 and demonstrate how each of them may be handled by current advances in machine learning (ML). Viral pandemics are a serious threat. No, COVID-19 isn't the first and won't be the last of its type. But using ML we can collect data and analyze them effectively to develop efficacy solutions. The main aim of this paper is to predict the cases in the future as a precautionary measure and understand the pattern to avoid further risk based on the available data, we took from World Health Organization and Johns Hopkins University Center for Systems Science and Engineering (CSSE).

## 2. METHODOLOGY

### 2.1. Preparing the dataset

The time-series data of Coronavirus Disease 2019 (COVID-19) shows the number of cases confirmed, fatalities, recoveries, active cases, new confirmed cases, new cases recovered, and new fatalities. The data is disaggregated by country (and sometimes sub-region). Data is updated daily in CSV format. It comes from the upstream stock managed by the excellent team at the Johns Hopkins University Centre, which has been performing a tremendous public service from early times by collecting data from throughout the world. It comes from this upstream repository- https://github.com/laxmimerit/Covid-19-Preprocessed-Dataset/tree/master/preprocessed **[2].** The data for India was extracted from the dataset and used for the research. No preprocessing and cleaning of the data where required. The data has been collected for the period 30th January 2020 to 15th April 2021 (441 days).

The data was polynomial and had extremes from single-digit data to values ranging to tens of millions. This disparity was overcome by applying the logarithmic function to the values, which were later used for seamless prediction, thereby delivering values that were close to actual cases. Usage of log helped in a more linearized approach on such variable data. The values for prediction were calculated by removing the logarithmic value as and when required.

### 2.2. Machine Learning Models

#### 2.2.1. Linear Regression

There are several ways to predict the connection between variables, but linear regression is by far the most popular. This can be either a linear or a non-linear regression. The linear regression is described in the following equation:

$Y = a + bX$

In this equation 'y' is the independent variable which can be either a continuous or categorical value, 'X' is a dependent variable which is always a continuous value. The line's slope is 'b', and the intercept (the value of y when x = 0) is 'a'. It is analyzed with a probability distribution and mainly focused on conditional probability distribution with multivariate analysis.**[3]** A modeler, for example, could wish to use a linear regression model to connect people's weights to their heights. A modeler should first assess whether or not there is a correlation between the variables of interest before attempting to fit a linear model to observed data. A scatterplot can be used to determine the strength of a correlation between two variables. If there appears to be no relationship between the suggested independent and dependent variables (i.e., no rising or decreasing trends in the scatterplot), fitting a linear regression model to the data is unlikely to provide a meaningful model.

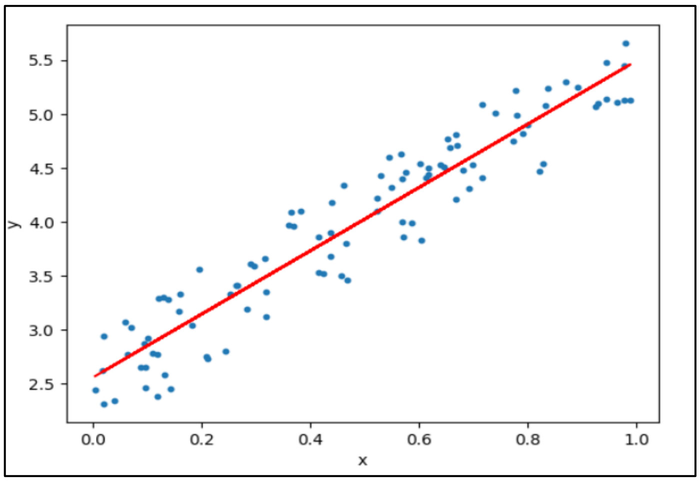


**Fig. 1 Linear Regression [13]**

**2.2.2. Support Vector Regressor**

A Support Vector Machine, or SVM, is a machine learning algorithm that analyses data and categorizes it into one of two groups. SVR is based on the Support Vector Machine (SVM) principle. It is one of the most often used Machine Learning models for classification and class assignment when the data is not linearly separable. For both linear and nonlinear regression types, support vector regression is a common choice for prediction and curve fitting. SVR is built on support vector machine (SVM) elements, where support vectors are essentially closer points towards the created hyperplane in an n-dimensional feature space that distinguishes the data points regarding the hyperplane. SVR has the advantage of allowing the building of a non-linear model without modifying the explanatory variables, resulting in a better interpretation of the final model. The generalized equation for hyperplane may be represented as:

Y = wX + b,

where w is weights and b are the intercept at X = 0. Epsilon ε represents the margin of tolerance. The SVR regression model is imported from the SVM class of sklearn python library.**[4]** In our prediction we have used multiple kernels, to gain various outputs to a single dataset. The kernels that yielded significant meaning were poly and linear. We used hyperparameter tuning through various methods, the values of significant importance were C=5000, degree = 2 and kernel = poly.

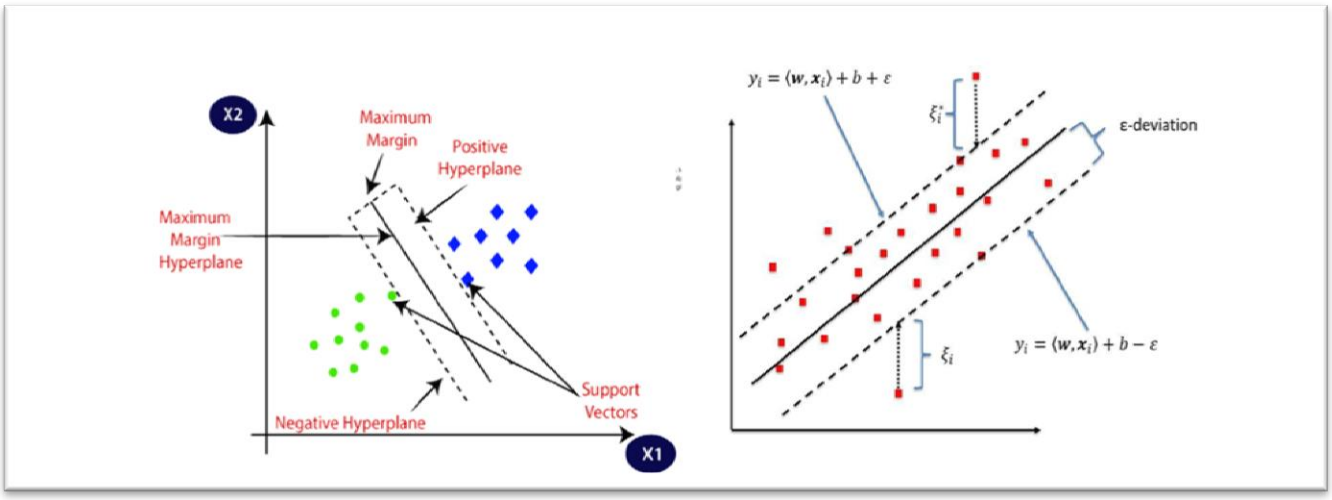

X2
Maximum Margin
Positive Hyperplane
Maximum Margin Hyperplane
Support Vectors
Negative Hyperplane
X1
$y_i = \langle w, x_i \rangle + b + \varepsilon$
$\xi_i^*$
ε-deviation
$\xi_i$
$y_i = \langle w, x_i \rangle + b - \varepsilon$

Fig 2. Support vector regression model for linear regression fitting where X1 = X and X2 = y are the features and labels in our case. [Image credit: https://www.researchgate.net/figure/Schematic-of-the-one-dimensional-support-vector-regression-SVR-model-Only-the-points_fig5_320916953] **[14]**

### 2.2.3. Decision Tree

The decision tree ML technique is used to split learning tasks, and the tree is built by dividing the dataset into smaller groups until each division is clean and pure, and data categorization is determined by the kind of data. The partition of the dataset attribute of numerical data type: (B)≤z, where z is the value of B domain for the entire categorical attribute of the data type partition C, form the values of (C), D∈E when E is a small set (B). To remove noise from the dataset, the pruning method process is used for the final construction of the tree when fully grown **[5].** In a decision tree method, each path starting from the root is representing a sequence of data splitting until a Boolean outcome is reached at the leaf node. In real-life practice, each path in the decision tree is a decision rule that can be easily translated into human languages or programming languages. The capability of decision support is not new in the studies of data mining and machine learning. Technically there are multiple methods capable of supporting decision makings. Nonetheless, a decision tree might be preferred because of its clearness and understandability. **[6]**

### 2.2.3. Random Forest

A random forest classifier is well known as an ensemble classification technique that is used in the field of machine learning and data science in various application areas. This approach employs "parallel ensembling," which involves fitting many decision tree classifiers in parallel on various data set sub-samples and using majority voting or averages to determine the conclusion or final result. As a result, it reduces the over-fitting problem while increasing forecast accuracy and control. As a result, an RF learning model with several decision trees is more accurate than a single decision tree-based model. To build a series of decision trees with controlled variation, it combines bootstrap aggregation (bagging) and random feature selection. It is adaptable to both classification and regression problems and fits well for both categorical and continuous values. **[7]**

For each decision tree, Scikit-learn calculates a node's importance using Gini Importance, assuming only two child nodes (binary tree) **[8]:**

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} - C_{right(j)}$$

$ni_{(j)}$= the importance of node j

$w_{(j)}$ = weighted number of samples reaching node j

$C_{(j)}$= the impurity value of node j

$left_{(j)}$ = child node from left split on node j

$right_{(j)}$ = child node from right split on node j

The importance for each feature on a decision tree is then calculated as:

$$fi_i = \frac{\sum_{j:node\ j\ splits\ on\ feature\ i} ni_j}{\sum_{k \in all\ nodes} n\,i_k}$$

- $fi_{(i)}$ = the importance of feature i
- $ni_{(j)}$ = the importance of node j

These can then be normalized to a value between 0 and 1 by dividing by the sum of all feature importance values:

$$normfi_i = \frac{fi_i}{\sum_{j \in all\ features} f\,i_j}$$

The final feature importance, at the Random Forest level, is its average over all the trees. The sum of the feature's importance value on each tree is calculated and divided by the total number of trees:

$$RFfi_i = \frac{\sum_{j \in all\ trees} nfi_{ij}}{T}$$

- $RFfi_{sub(i)}$ = the importance of feature i calculated from all trees in the Random Forest model
- $normfi_{sub(ij)}$ = the normalized feature importance for i in tree j

- T = total number of trees

### 2.2.4. Polynomial Regression

Polynomial regression is a type of regression analysis in which the connection between the independent variable x and the dependent variable y is a polynomial degree of nth in x rather than a linear relationship. Polynomial regression's equation is derived from the linear regression (polynomial regression of degree 1) equation, which is defined as follows:

y=b0 + b1x + b2x2 + b3x3 +⋯+ bnxn + e.

Here, *y* is the predicted/target output, b0, b1, bnb0, b1,.bn are the regression coefficients, *x* is an independent/ input variable. In simple words, we can say that if data are not distributed linearly, instead it is the nth degree of polynomial then we use polynomial regression to get desired output. **[9]**

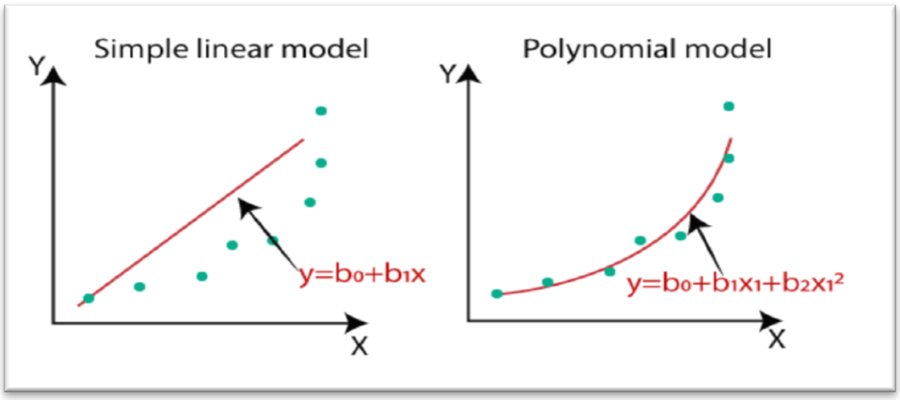


**Fig 3: Simple Linear Regression and Polynomial Regression [15]**

### 2.2.5. Multilayer Perceptron

The multilayer perceptron is the basic architecture of deep learning, commonly known as the feed-forward artificial neural network (MLP). In general, an MLP consists of an input layer, one or more hidden layers, and an output layer, and is completely linked. At a particular weight, each node in a layer is connected to each node in the layer below it. A neural network's most fundamental building element, "backpropagation," is used by MLP to internally modify weight values as the model is being built. There are a number of hyperparameters that may be adjusted in MLP that affect scaling and can result in a computationally costly model. Hypermeter tweaking resulted in an incredibly accurate model. The tanh and relu activation functions were utilized to make successful forecasts.

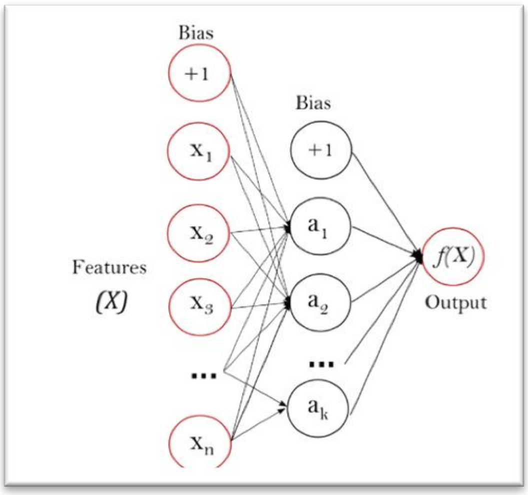


**Fig 4: Multilayer Perceptron [16]**

Artificial neural networks have been intensively explored and applied in real-world industrial applications over the previous four decades. Artificial neural networks are still in the early stages of development. As soon as they have been taught, artificial neural networks are very effective. The artificial neural network's training phase is the most difficult element of its use. Training is often too sluggish and inefficient. The multilayer perceptron is a weighted directed graph organized in layers.

Each neuron of one single layer is linked to each neuron of the next layer when the multilayer perceptron is fully connected. Neurons are the nodes in the network that weigh the connections between nodes. The weight of the multilayer perceptron is determined. The selection of suitable weight values is a worldwide challenge of optimization. The signals for the input are collected by each neuron. Signals are received in the multilayer perceptron environment from other neurons or external ones. Signals multiply by the weight of the connection and determine their total. By activation of each neuron, the number of weighted signals is generally normalized. This normalization is a fundamental characteristic of the multilayer perceptron since the input connections of neurons are variable in quantity and there is no weight constraint. **[10]**

Technically termed as artificial neurons, ANN imitates the functions and activities of the brain of a person. They exchange information in the form of 0s and 1s or combinations thereof, with each neuron having a specific weight assigned to it, which defines its duties and roles in the system. The structure of ANN is divided into layers, right from the data reception layer, input layer, middle or hidden layer to output layer which is called extraction or classification layer. Each layer has a specific function to perform and transform data into relevant information to get the ultimate and optimum result. The Activation and transfer function plays a critical role in the activities carried out by neurons. The transfer function added all the weighted input as **[11]**:

$$Z = \sum_{n=1}^{n} W_i X_i + W_b b$$

Thus, b represents the value of bias, which is often 1 value.

### 2.3 Error Calculation

#### 2.3.1 R Squared Error

The ratio of Sum of Squares Regression (SSR) to Sum of Squares Total (SST) is known as R-Squared (SST). The amount of variance explained by the regression line is known as the Sum of Squares Regression. R-Squared captures the fraction of response variance captured by the regression and tends to give a better picture of the quality of the regression model. The greater the SSR value, the more the variation covered by the regression / best fit line as a percentage of total variance (SST). A low number indicates a low level of correlation, implying a regression model that is not always valid. The r2 score typically varies from zero to 100 percent. It is similar to, but not identical to, the MSE (see below). The proportion of the variation in the dependent variable that is predicted by the independent variable(s) is defined as r2; if it is 100 percent, the two variables are completely correlated, i.e., there is no variance at all. Here's a graphic depiction to help you better grasp the principles of R-Squared.

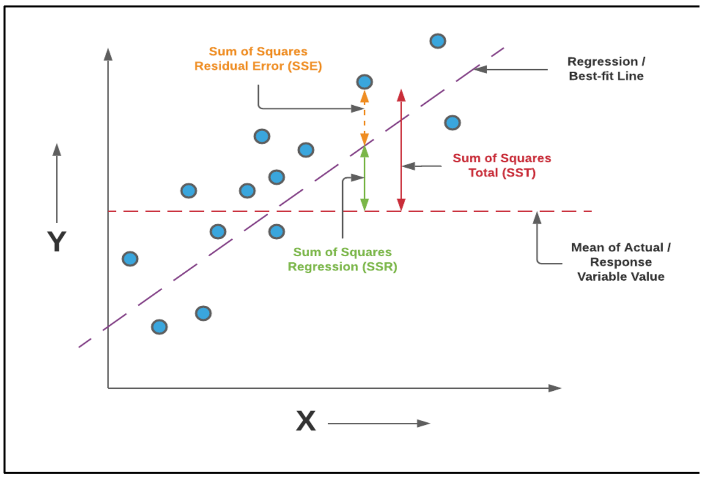

Sum of Squares Residual Error (SSE)
Regression / Best-fit Line
Sum of Squares Total (SST)
Y
Sum of Squares Regression (SSR)
Mean of Actual / Response Variable Value
X

Fig 5: Error Calculation [17]

$$R^2 = \frac{SSR}{SST} = \frac{\Sigma(\hat{y}_i - \bar{y})^2}{\Sigma(y_i - \bar{y})^2}$$

### 2.4 Hyperparameter Tuning.

A mathematical model containing several parameters that must be learned from data is referred to as a Machine Learning model. We can fit the model parameters by training a model using existing data. Hyperparameters, on the other hand, are a type of parameter that cannot be learned directly from the standard training procedure. They are normally fixed before the start of the training procedure. These parameters describe crucial aspects of the model, such as its complexity and learning rate. Hyper-parameters are variables that are not learned directly by estimators. They are supplied as arguments to the constructors of the estimator classes in scikit-learn. C, kernel, and gamma for Support Vector Classifier, alpha for Lasso, and so on are common examples. It is feasible and encouraged to look for the best cross-validation score in the hyper-parameter space. C- In SVM, it is a hypermeter for error control. For each misclassified data point, the C parameter applies a penalty. Because the penalty for misclassified points is modest when c is small, a large-margin decision boundary is chosen at the price of a greater number of misclassifications. If c is large, SVM tries to minimize the number of misclassified cases due to the high penalty, which results in a decision.

Kernels are a means for machine learning algorithms to get greater flexibility by expanding the dataset's polynomial degree without increasing the number of features. The kernel trick allows you to acquire the same result as if you had added numerous polynomial features, even with polynomials of extremely high degree, without really having to add them. The polynomial kernel is a machine learning kernel function that encodes the similarity of vectors in a feature space over polynomials of the original variables, allowing non-linear models to be learned. It is often used with support vector machines and other kernelized models. Intuitively, the polynomial kernel determines the similarity of input samples by looking at more than just the specified attributes. The kernel functions are used to map the original dataset (linear/nonlinear) into a higher dimensional space to transform it into a linear dataset. A polynomial kernel's (implicit) feature space is the same as that of polynomial regression, but without the combinatorial blow-up in the number of parameters to learn.

Polynomial kernels with a higher degree provide for a more flexible decision boundary. In the epsilon-SVR model, epsilon defines the epsilon-tube inside which no penalty is associated in the training loss function for points predicted within epsilon of the actual value (from documentation).The amount of accuracy of the estimated function is determined by the value of epsilon. It is fully dependent on the training set's target values. We cannot expect a favorable result if epsilon is larger than the intended value range. We can expect overfitting if epsilon is zero. As a result, Epsilon must be chosen to reflect the information in some way.

### 2.4 Cross-Validation

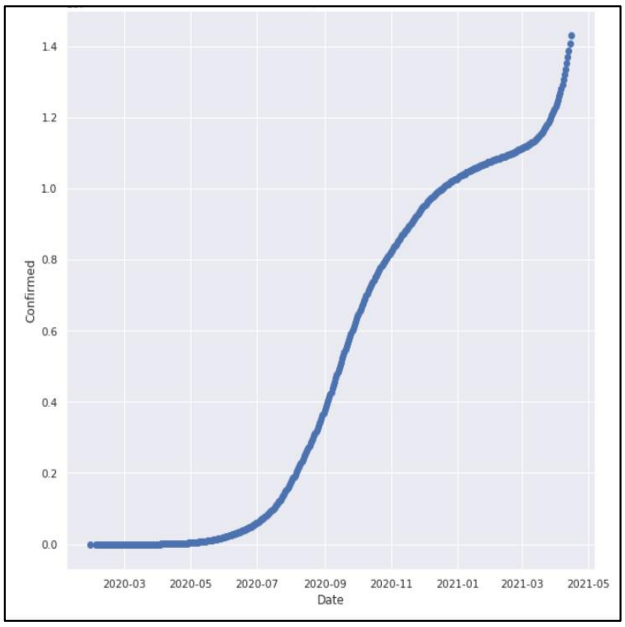
Cross-Validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. In typical cross-validation, the training and validation sets must cross over in successive rounds such that each data point has a chance of being validated against. In machine learning, it is a methodology for evaluating predictive models **[12]** and estimating a machine learning model's skill on unknown data. That is, to use a small sample to anticipate how the model will perform in general when used to generate predictions on data that was not utilized during the model's training. The process includes only one parameter, k, which specifies the number of groups into which a given data sample should be divided. As a result, the process is frequently referred to as k-fold cross-validation. When a precise value for k is specified, it can be substituted for k in the model's reference, for example, k=10 for 10-fold cross-validation. The K-Folds method is popular and simple to grasp, and it generally produces a less biased model than other methods. Because every observation from the original dataset has a probability of showing up in the training and test sets. Hence, we have used the k-fold cross-validation technique to validate our model.

## 3. RESULTS AND DISCUSIONS

### 3.1 Exploratory Data Analysis

Exploratory Data Analysis is an approach to analyze data, summarize the main characteristics of data, and better understand the data set. It also allows us to quickly interpret the data and adjust different variables to see their effect. The three main steps to get a perfect EDA are extracting the data from an authorized source, cleaning and processing the data, and performing data visualization on the cleaned data set.

In our research, we utilized EDA to quickly explore the covid-19 dataset and gain meaningful insights from it. For instance, to examine the weekly active cases or the growing trend of confirmed instances. With such insights, we may forecast the graph curve and employ it as a useful tool to some extent.



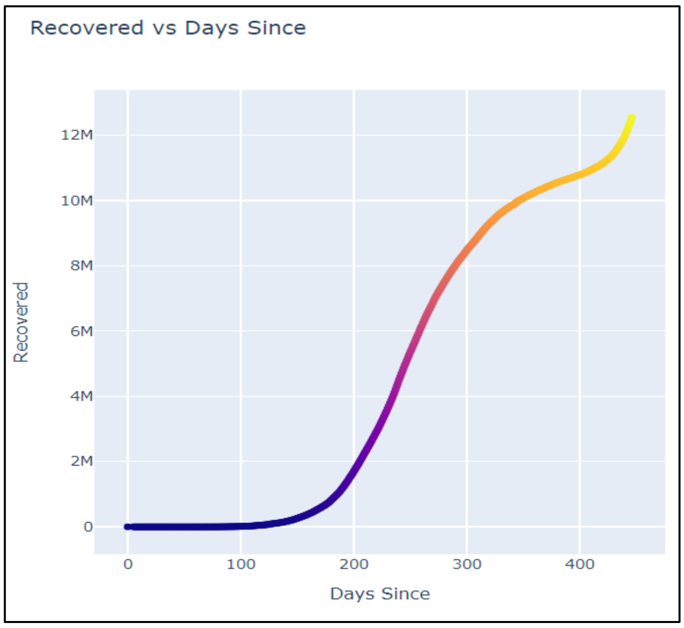


**Fig 6: Confirmed vs Days Since**

**Fig 7: Recovered vs Days Since**

In the Fig 6-line graph, the X-axis represents the "Dates" from March 2020 till today, with two-month intervals in between; and the Y-axis represents the number of "Confirmed "cases over the period. One can significantly observe from the above data that the number of confirmed cases exploded from June 2020, till the line became a bit even between January 2021 and March 2021. Furthermore, the second wave of covid-19 made the curve steeper from March 2021, and a sudden hike was registered in India.

In Fig 7, it is undeniable that the rate of 'Recovery' has gone up significantly, from a million to 12 million individuals, resulting in less risk.

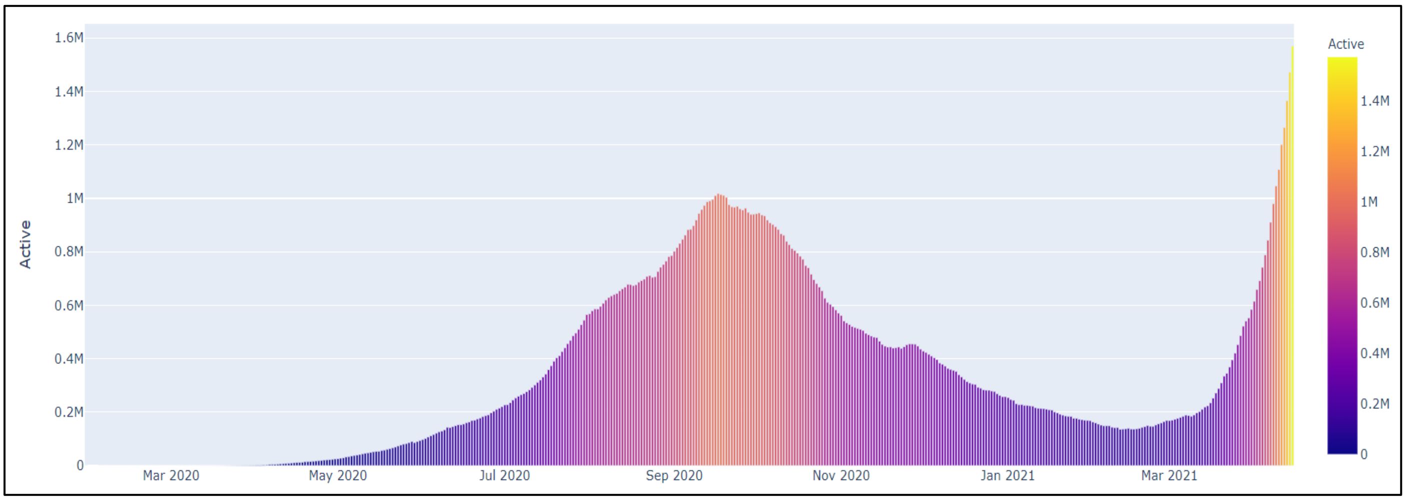


**Fig 8: Active vs Months**

According to the graph above, the number of 'Active' instances fluctuated throughout time, with an average ranging from 1 million to nearly 2 million depending on the circumstances.

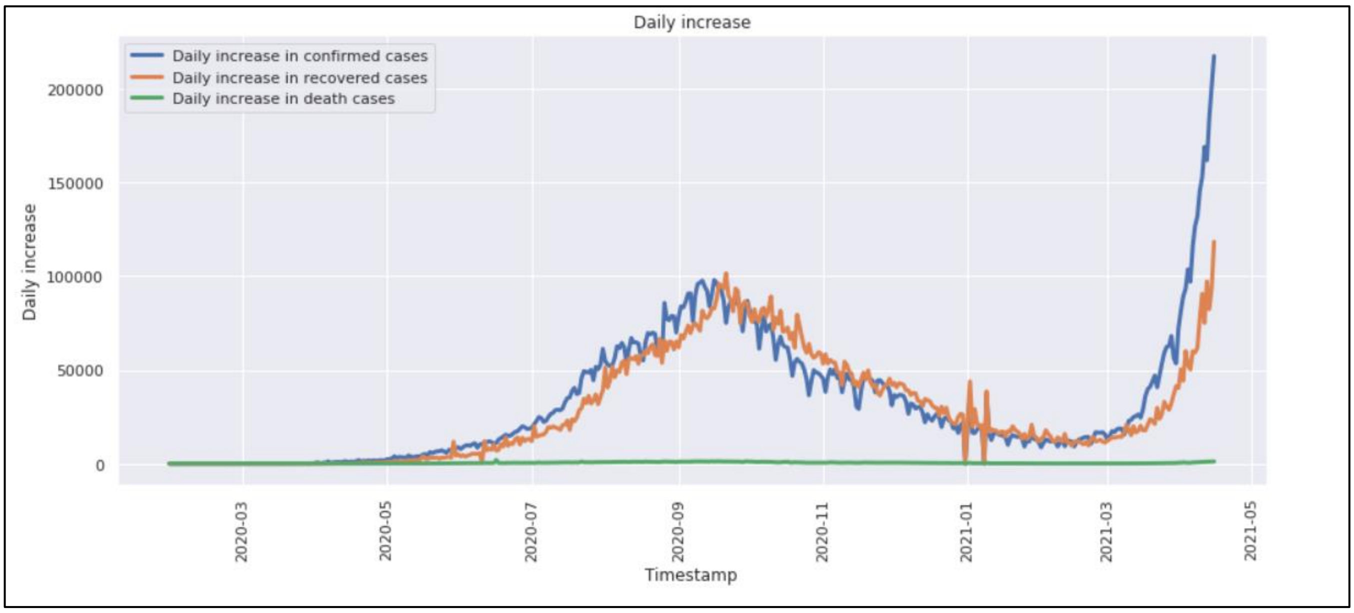


**Fig 9: Daily increase in the number of cases**

In the given line plot, the X-axis represents the Dates on an interval of 2 months from March 2020 till May 2021, and Y-axis represents the Daily increase in the number of cases (in lakhs). Blue Line specifies a daily increase in "Confirmed cases". The orange line shows a daily increase in "Recovered cases" and the green line shows a daily increase in "Deaths".

From the beginning of the year 2020, from March to May, there were little changes in the day-to-day growth of confirmed cases, and therefore recovered cases. From June 2020, "Confirmed" cases started to pop up in India. The number of confirmed and recovered cases increased until late September, following which it began to decline considerably until March 2021. Unfortunately, beginning in late March 2021, with the second wave of covid-19, the number of cases has increased exponentially up to the present day, followed by daily "Recovered" cases. We can also see that there has been no significant increase in the daily number of fatalities throughout the period.

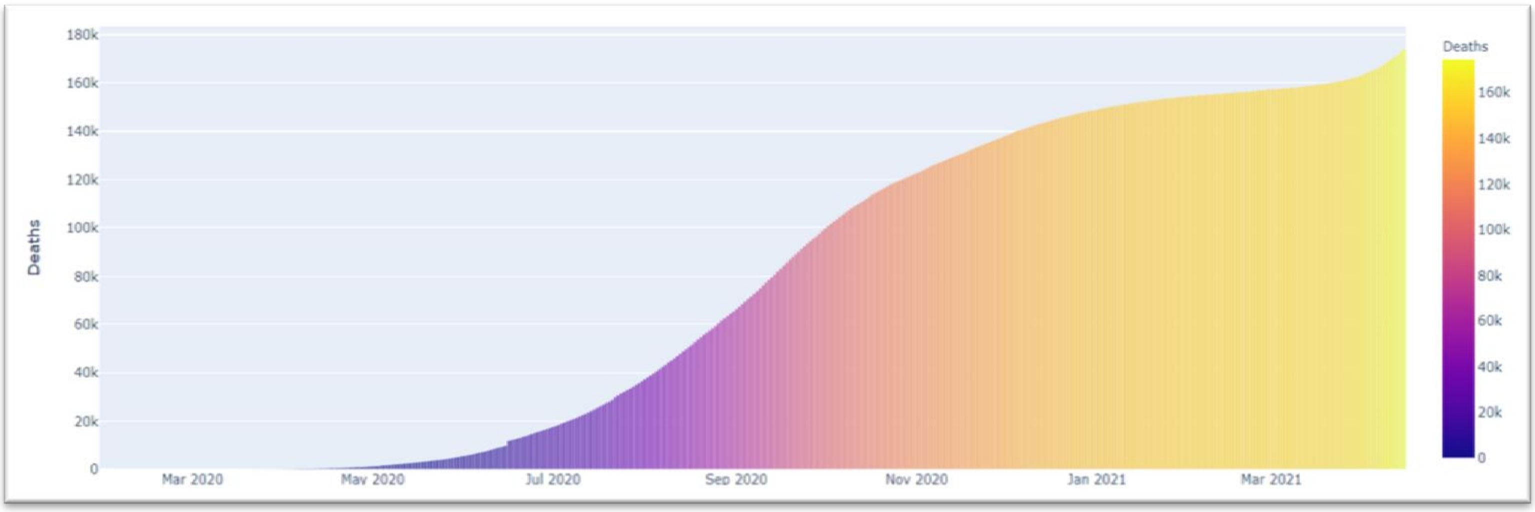


**Fig 10: Deaths vs Months**

From the above bar chart, it is perceptible that the number of casualties caused due to covid-19 has significantly increased from May 2020 until the present day.

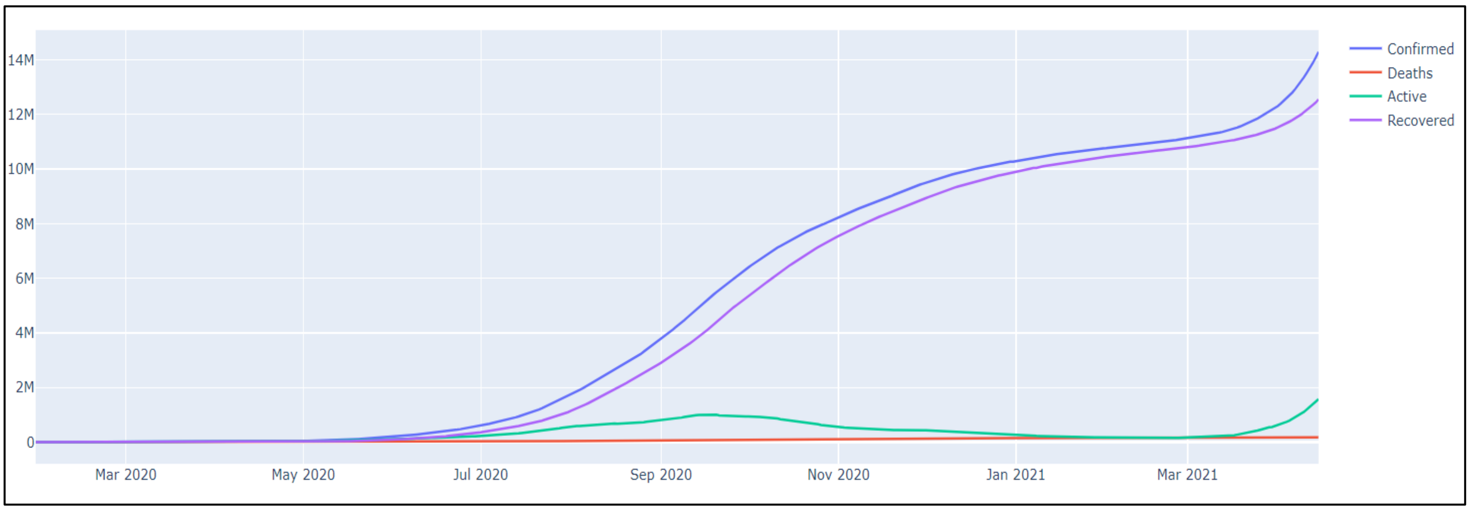


**Fig 11: A comparison of confirmed, deaths, active, and recovered cases.**

This diagram gives us a comparative perspective between the number of confirmed cases, active cases, recovered cases, and deaths. Approximately 14 million "Confirmed" cases have been registered till March 2021. Out of the 14 million confirmed cases, round about 12 million cases have been recovered and less than a million resulted in death leaving around almost 2 million active cases. In April 2021, India registered more than 12 million cases out of which 10 million have resulted in recovery on daily basis. It was seen that India's Recovery rate stands at 92.79%.

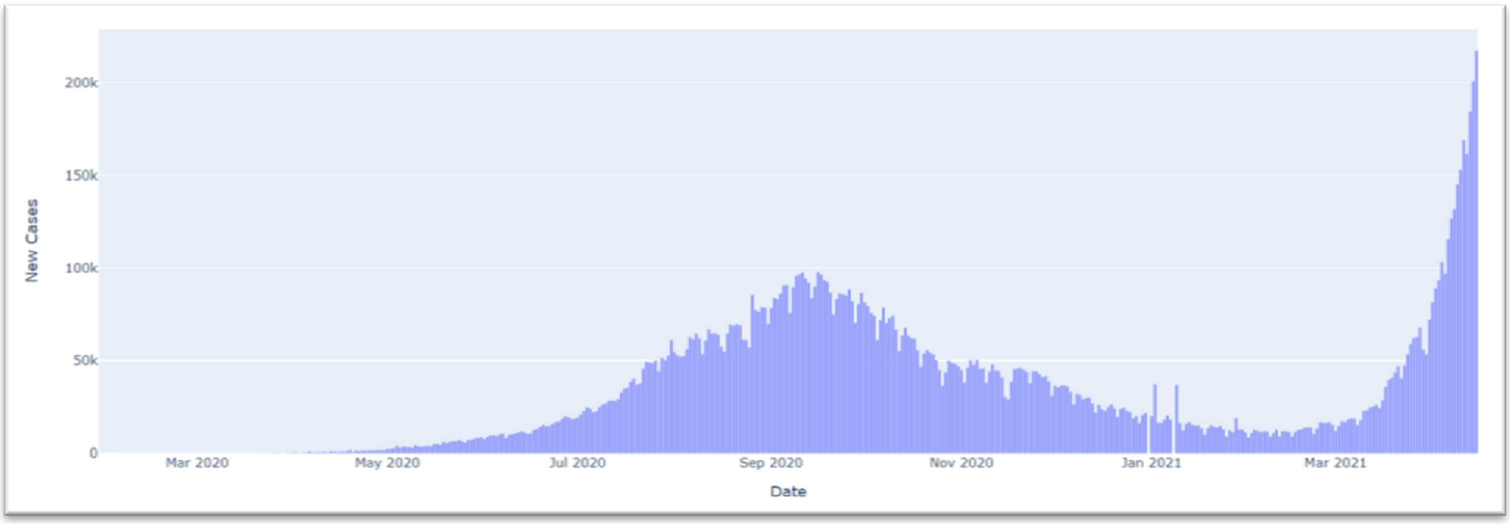


**Fig 12: New Cases vs Months**

This diagram manifests the new confirmed cases that pop up daily. It can be made out that during the first wave of covid-19, the cases significantly increased from May 2020 until late September 2020. After which there was a great amount of decrease in the daily cases found in India. But by March 2021, the daily cases suddenly spiked, making the curve 3 times steeper than the previous one.

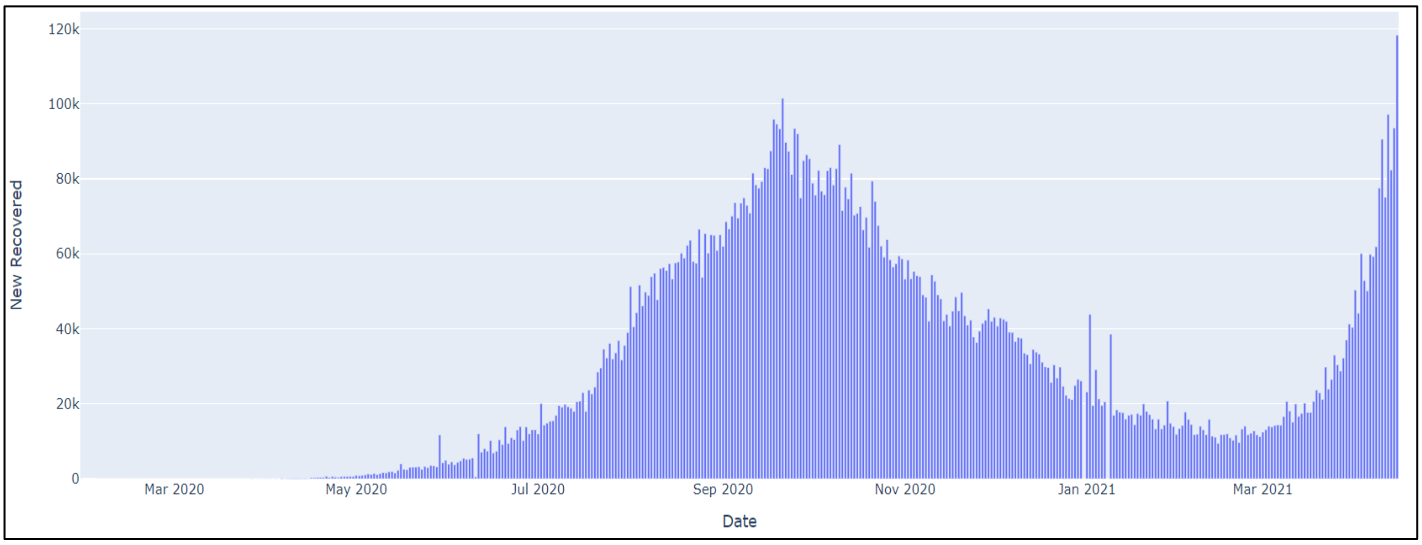


**Fig 13. New Recovered vs Months**

From the above chart, we can observe the great recovery rate of patients daily. The recovery rate has risen with the daily increase in confirmed cases and has significantly decreased with the decrease in confirmed cases. It was seen that India's Recovery rate stands at 92.79%.

### 3.2 Predicting the Covid-19 Curve

The models that were used have variable accuracies from 90% to more than 95%, these models were trained on data up to the 15th of April 2021, while predictions for further days was compared with real-world data and the models were able to closely predict the peak of second-wave in India. These models can be deployed in the real world and could be used to predict the next wave with proper inputs. Predicting the next wave is beyond the scope of this paper, we predicted expected confirmed cases through various models while including one or three features depending upon the correlation score. The models are discussed one by one:

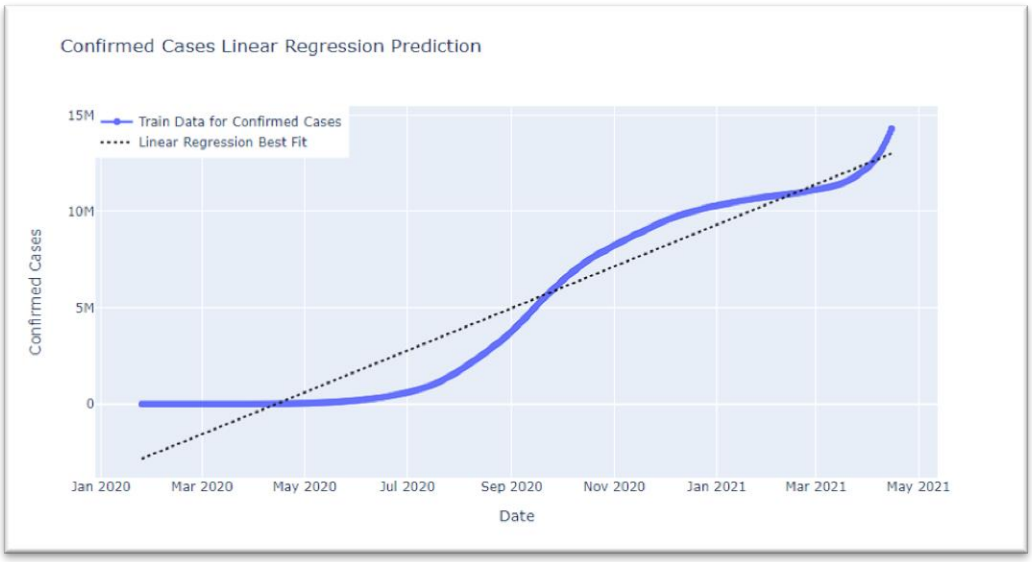
### 3.2.1 Linear Regression

Using a single feature i.e., the number of days, a benchmark model was created which reported an accuracy of 92.1%. It is to be noted that despite the variations in the data a model with logarithmic values was unable to deliver meaningful accuracy. We used this model to predict further cases and the numerical values are displayed in the figure below.

It is to be noted that the values predicted are not close enough to predict the second wave, as the graph was linear it was unable to predict the second wave accurately hence this model serves as a baseline for further predictions.

This model can be used to predict cases when there is a lockdown imposed in the country where the expectations of the rise of cases are much less.

The graph below shows the actual cases v our model, again in the longer perspective this model suits for usage. We have used validation techniques like K-Fold cross-validation having 10 splits to avoid over/underfitting.

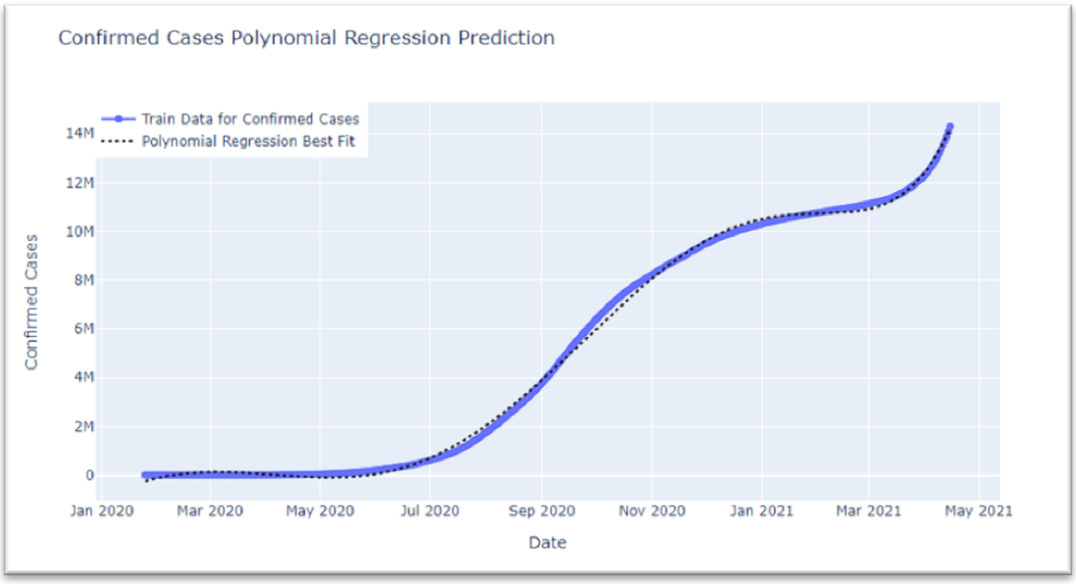


```
day [474] = 13994623
day [475] = 14030129
day [476] = 14065635
day [477] = 14101141
day [478] = 14136647
day [479] = 14172154
day [480] = 14207660
day [481] = 14243166
day [482] = 14278672
day [483] = 14314178
day [484] = 14349684
day [485] = 14385191
day [486] = 14420697
day [487] = 14456203
day [488] = 14491709
day [489] = 14527215
day [490] = 14562722
day [491] = 14598228
day [492] = 14633734
day [493] = 14669240
day [494] = 14704746
day [495] = 14740252
day [496] = 14775759
day [497] = 14811265
day [498] = 14846771
day [499] = 14882277
```

**Fig 14. Confirmed Cases vs Months.**

### 3.2.2 Polynomial Regression

As the next step to avoid linearity in the curve, polynomial regression was implemented upon optimizing various degrees the optimal degree was noted to be 6 after hyperparameter tuning. The cross-validation technique used was K-Fold with splits = 10. The resultant curve was optimum enough to fit the data extremely well, it is the best model seen so far with 98% accuracy on top of it, the model predicted the confirmed cases for the second wave extremely well. The degree selection was done by hyperparameter tuning and degree 6 was found to fit the curve and was able to avoid overfitting by predicting optimum values. The predicted values are listed.

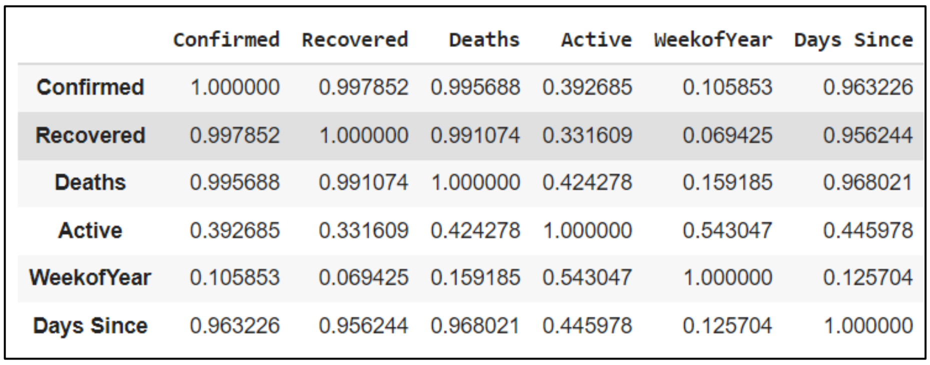


```
day [474] = 21553746
day [475] = 21942621
day [476] = 22342022
day [477] = 22752146
day [478] = 23173196
day [479] = 23605374
day [480] = 24048886
day [481] = 24503940
day [482] = 24970748
day [483] = 25449521
day [484] = 25940476
day [485] = 26443830
day [486] = 26959804
day [487] = 27488622
day [488] = 28030508
day [489] = 28585691
day [490] = 29154401
day [491] = 29736871
day [492] = 30333339
day [493] = 30944041
day [494] = 31569219
day [495] = 32209117
day [496] = 32863982
day [497] = 33534062
day [498] = 34219609
day [499] = 34920878
```

**Fig 15. Confirmed Cases vs Months.**

### 3.2.3 Support Vector Regressor

Up till now the models depended on a single feature namely the number of days to predict our data. We calculated the correlation score for each feature, the table for which is listed below:



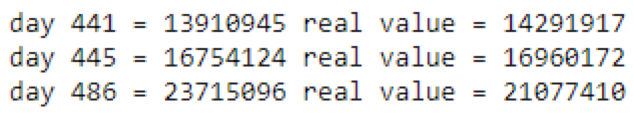
| | Confirmed | Recovered | Deaths | Active | WeekofYear | Days Since |
|---|---|---|---|---|---|---|
| **Confirmed** | 1.000000 | 0.997852 | 0.995688 | 0.392685 | 0.105853 | 0.963226 |
| **Recovered** | 0.997852 | 1.000000 | 0.991074 | 0.331609 | 0.069425 | 0.956244 |
| **Deaths** | 0.995688 | 0.991074 | 1.000000 | 0.424278 | 0.159185 | 0.968021 |
| **Active** | 0.392685 | 0.331609 | 0.424278 | 1.000000 | 0.543047 | 0.445978 |
| **WeekofYear** | 0.105853 | 0.069425 | 0.159185 | 0.543047 | 1.000000 | 0.125704 |
| **Days Since** | 0.963226 | 0.956244 | 0.968021 | 0.445978 | 0.125704 | 1.000000 |

**Fig 17: Correlation Score**

The features that were highly correlated for the confirmed column are Deaths, Days Since, and Recovered cases. This way it would give us a rough estimate on predicting the confirmed cases for a single day, on reversing the model to predict all these things just by knowing the confirmed cases would help too. The application can be vast enough for the current user process. Currently, we stick to predict the confirmed cases by using 3 features.

SVR Regressor has multiple kernels like RBF, poly, and linear out of this one can be tuned for its hyperparameter that is poly. The poly kernel has values of parameters like degrees and C value which can be tuned. This model was another type of regressor that helped us consider multiple features.

6 models were tried on SVR with different kernels and varying degrees for the poly kernel. The values of the 'c' hyperparameter is varied and the value 20000 was found to be optimum. The kernels used had varying degrees from 2 to 5, one of those gave an accuracy of 91.5% which is fair enough for predictions. On inputting values for deaths, recovered, and days. The value for predictions for a fortnight was found to be in acceptance with the real-world data.



```
day 441 = 13910945 real value = 14291917
day 445 = 16754124 real value = 16960172
day 486 = 23715096 real value = 21077410
```

The values are of 15th April, 19th April, and 5th May 2021, respectively. The cross-validation technique is the same as mentioned for the above models.

### 3.2.4 Multilayer Perceptron

This model had three features as an input had an accuracy of more than 99% suggesting that the model had overfitted the data, which was visible from the predictions, these are generalized models which are usually used for predictions, since they are mostly trained on similar type of data, the overfitting is expected. We used the same cross-validation techniques, and the results were expected. This model is a deep learning model with hidden layers to predict construed values, hence they fit this model perfectly. Multiple hidden layers were input along with different activation functions, multiple combinations were tested and tried to give out meaningful predictions, solvers like SGD and LBGFS were used while the learning rate was varied from 0.01 to 1, the optimum value was found out to be 0.01 yielding a strong accuracy. We tried to predict some data which fell well in line with the real-world trend, for example predicting data for 5 days after our input data gave results as 14262292 for confirmed cases whereas the real-world data was found out to be 15616130 which gave a percent error of 8.6% which is fair enough for such a model. This proves that overfitted models are credible enough to give good results. Multiple features are also good enough for recommendations.

Again, the scope of this paper is to analyze multiple models and report their findings while suggesting better models for usage and further research. The researchers can conclude that using polynomial regression for a single feature is suitable enough and for multiple features SVR is suitable.

### 3.2.5 Decision Tree and Random Forest

We applied three features as input in both decision tree and random forest, and the accuracy was around 98 percent and 97 percent, respectively. To get the corresponding results, we employed the same cross-validation technique. It was observed that the predicted values were close to real values on some days and a bit deviated on the other days. For instance, the random forest predicted 13934058 on the third day after our input data, whereas the decision tree model predicted 14291917 confirmed cases. The deviated results that we observed show that both the models are overfitted in paltry amounts. This study is intended to examine and report on multiple models while recommending preferable use and research models.

## 4. CONCLUSION

In this paper, we have proposed six machine learning models for the prediction of the COVID-19 outbreak in India. These models predict the confirmed cases for 26 days while including one or three features depending upon the correlation score. We have taken into consideration all the models that have variable accuracies from 80% to more than 95%. After evaluating the COVID-2019 outbreak in India dataset from 30th January 2020 to 15th April 2021 we forecasted the results for 26 days. SVR has been reported to outperform the consistency in predictability concerning other models when used to predict with multiple features. On the other hand, polynomial regression performs better when used to predict with a single feature. There seems to be very little difference between confirmed and predicted results for the COVID-2019 outbreak in India using these models. This suggested study will be extremely helpful to Indian doctors and the Indian government in managing the COVID-2019 outbreak over 26 days. Further, these models can be tuned for forecasting over long-term intervals. We can lower the spikes in the dataset and thus lower the rate of progression if appropriate containment measures with social distancing and hygiene are maintained.

| **Model** | **Accuracy (%)** |
|---|---|
| Linear Regression | 92.1 |
| Polynomial Regression | 98 |
| Support vector regressor | 84 |
| Multilayer perceptron | 99 |
| Decision tree | 98 |

| | |
|---|---|
| Random Forest | 97 |

**Table 1: Accuracies of different models used in the study.**

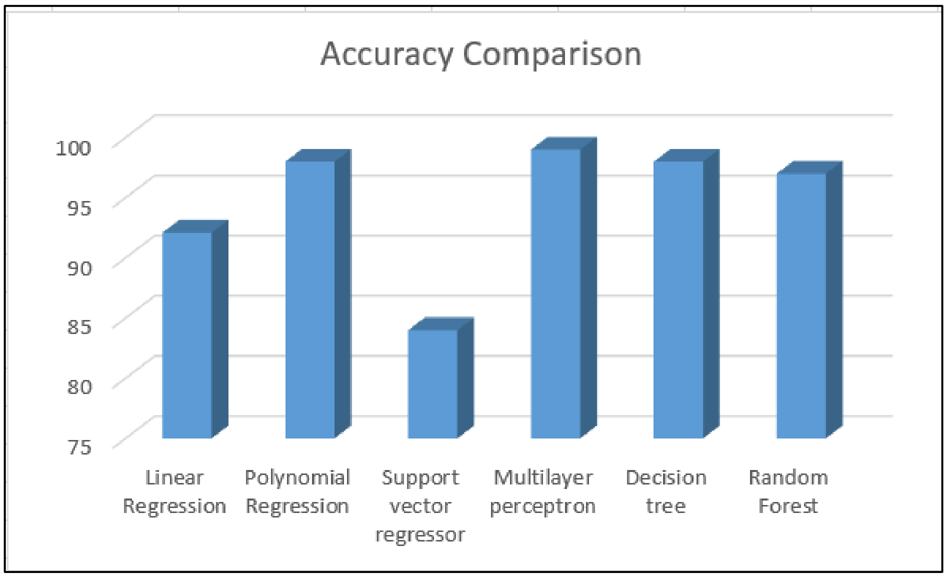


**Fig 17: Bar-Graph Representing Accuracies of Different Models**

## 5. REFERENCES

[1] (https://www.hopkinsmedicine.org/)

[2] World Health Organization (WHO): https://www.who.int/

[3] Kavitha S, Varuna S and Ramya R, "A comparative analysis on linear regression and support vector regression," 2016 Online International Conference on Green Engineering and Technologies (IC-GET), 2016, pp. 1-5, doi: 10.1109/GET.2016.7916627.

[4] Debanjan Parbat, Monisha Chakraborty, A python-based support vector regression model for prediction of COVID19 cases in India, Chaos, Solitons & Fractals, Volume 138,2020,109942, ISSN 0960-0779

[5] Muhammad, L.J., Algehyne, E.A., Usman, S.S. *et al.* Supervised Machine Learning Models for Prediction of COVID-19 Infection using Epidemiology Dataset. *SN COMPUT. SCI.* **2,** 11 (2021).

[6] F. Yang, "An Extended Idea about Decision Trees," 2019 International Conference on Computational Science and Computational Intelligence (CSCI), 2019, pp. 349-354, doi: 10.1109/CSCI49370.2019.00068.

[7] Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN COMPUT. SCI.* **2,** 160 (2021). https://doi.org/10.1007/s42979-021-00592-x

[8] https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3

[9] Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN COMPUT. SCI.* **2,** 160 (2021). https://doi.org/10.1007/s42979-021-00592-x

[10] Balabanov, Todor & Zankinski, Iliyan & Kolev, Kolyu. (2018). Multilayer Perceptron Training Randomized by Second Instance of Multilayer Perceptron.

[11] Muhammad, L.J., Algehyne, E.A., Usman, S.S. *et al.* Supervised Machine Learning Models for Prediction of COVID-19 Infection using Epidemiology Dataset. *SN COMPUT. SCI.* **2,** 11 (2021).

[12] Refaeilzadeh P., Tang L., Liu H. (2009) Cross-Validation. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9_565

[13] https://towardsdatascience.com/linear-regression-using-python-b136c91bf0a2

[14]https://www.researchgate.net/figure/Schematic-of-the-one-dimensional-support-vector-regression-SVR-model-Only-the-points_fig5_320916953

[15] https://www.javatpoint.com/machine-learning-polynomial-regression

[16] https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[17] https://vitalflux.com/linear-regression-explained-python-sklearn-examples/