CONTENTS

ABSTRACT

Machine learning is a subset of artificial intelligence (AI). It allows the system to be able to learn automatically from its previous experiences or analysis without being programmed explicitly. The main idea of machine learning is to focus on the development of computer programs that can access data and use it to learn on their own. Machine learning algorithms are built on a model that depends on sample data. This sample data is known as training data. This data is further used to make various predictions as well as helpful in decision making without being programmed explicitly. Various prediction models are popularly being used to handle various future forecasting problems. This study demonstrates how capable machine learning models are to predict the number of expected cases or patients is affected by COVID-19. Coronavirus disease (COVID-19) is a very infectious disease that has become a serious threat to mankind. Machine learning algorithms play a crucial role in pandemic investigation and forecasting. Furthermore, machine learning techniques help on exposing the epidemic patterns. Mainly, two standard forecasting models, such as linear regression (LR) and support vector machine (SVM) have been used in this study to forecast the factors of COVID-19. Each model makes three types of predictions, such as the number of infected cases, the number of deceased cases, and the number of cases recovered in the coming 10 days. The results obtained by the study shows which mechanism will be the most accurate and reliable to use these techniques for the present scenario of the COVID19 pandemic.

Keywords: COVID-19, future forecasting, supervised machine learning.

LIST OF FIGURES

1. INTRODUCTION

Machine learning (ML) has proved itself as a prominent field of study over the last decade by solving many very complex and sophisticated real-world problems. The application areas included almost all the real-world domains such as healthcare, autonomous vehicle (AV), business applications, natural language processing (NLP), intelligent robots, gaming, climate modeling, voice, and image processing. ML algorithms' learning is typically based on trial and error method quite opposite of conventional algorithms, which follows the programming instructions based on decision statements like if-else. One of the most significant areas of ML is forecasting, numerous standard ML algorithms have been used in this area to guide the future course of actions needed in many application areas including weather forecasting, disease forecasting, stock market forecasting as well as disease prognosis. Various regression and neural network models have wide applicability in predicting the conditions of patients in the future with a specific disease. There are lots of studies performed for the prediction of different diseases using machine learning techniques such as coronary artery disease, cardiovascular disease prediction, and breast cancer prediction. In particular, the study is focused on live forecasting of COVID-19 confirmed cases and study is also focused on the forecast of COVID-19 outbreak and early response. These prediction systems can be very helpful in decision making to handle the present scenario to guide early interventions to manage these diseases very effectively. This study aims to provide an early forecast model for the spread of novel coronavirus, also known as SARS-CoV-2, officially named as COVID-19 by the World Health Organization (WHO). COVID-19 is presently a very serious threat to human life all over the world. At the end of 2019, the virus was first identified in a city of China called Wuhan, when a large number of people developed symptoms like pneumonia. It has a diverse effect on the human body, including severe acute respiratory syndrome and multiorgan failure which can ultimately lead to death in a very short duration. Hundreds of thousands of people are affected by this pandemic throughout the world with thousands of deaths every coming day. Thousands of new people are reported to be positive every day from countries across the world. The virus spreads primarily through close person to person physical contacts, by respiratory droplets, or by touching the contaminated surfaces. The most challenging aspect of its spread is that a person can possess the virus for many days without showing symptoms. The causes of its spread and considering its danger, almost all the countries have declared either partial or strict lockdowns throughout the affected regions and cities. Medical researchers throughout the globe are currently involved to discover an appropriate vaccine and medications for the disease. Since there is no approved medication till now for killing the virus so the governments of all countries are focusing on the precautions which can stop the spread. Out of all precautions, ‘‘be informed’’ about all the aspects of COVID-19 is considered extremely important. To contribute to this aspect of information, numerous researchers are studying the different dimensions of the pandemic and produce the results to help humanity.

Coronavirus spreads are categorized into four stages. The first stage starts with the cases recorded for the people who traveled to or from affected countries or cities, whereas in the second stage, cases are reported regionally among family, friends, and groups who came into contact with the person coming from the affected countries. Therefore, the affected people are identifiable. Next, the third stage causes the circumstance severely as the infected person becomes undetectable and flattens across the individuals who neither have any travel records nor came in connection with the affected person. This condition obliges immediate lockdown across the nation to reduce the social contacts between individuals to measure the movement of the virus. Finally, stage four starts when the transmission converts to endemic and uncontrollable. China is the first country that felt under stage four of the COVID-19 transmission, while most of the developed countries are now in this stage of the transmission and bearing a further number of epidemics and losses compared to China.

To contribute to the current human crisis our attempt in this study is to develop a forecasting system for COVID-19. The forecasting is done for the three important variables of the disease for the coming 10 days: 1) the number 0f New confirmed cases. 2) the number of death cases 3) the number of recoveries. This problem of forecasting has been considered as a regression problem in this study, so the study is based on some state-of-art supervised ML regression models such as linear regression (LR), least absolute shrinkage and selection operator (LASSO), support vector machine (SVM), and exponential smoothing
(ES). The learning models have been trained using the COVID-19 patient stats dataset provided by Johns Hopkins. The performance evaluation has been done in terms of

important measures including R-squared score (R2 score), mean square error (MSE), mean absolute error (MAE), and root mean square error (RMSE).

This study has some key findings which are listed below:

- Different ML algorithms seem to perform better in different class predictions.
- Most of the ML algorithms require an ample amount of data to predict the future, as the size of the dataset increases the model performances improve.
- ML model based forecasting can be very useful for decision-makers to contain pandemic like COVID-19.

2. LITERATURE SURVEY

Pandemic spread has recurred very regularly on this planet. Records of such information as old as from the 3000 Century BC are available in the literature. During times of health emergencies, it becomes imperative to study the spread, the preventive measures, and the future actions. Every time during such pandemics, researchers and experts have come forward to provide various solutions to tackle such a situation, —COVID-19 pandemic is no exception for the same. Expert studies during the pandemic varies and is multidisciplinary. From the cause to mitigate effects and prevent/protect the future, researchers across the globe have put in their efforts to guide the stakeholders. The current situation has added one more dimension to this and that is the world is a connected place now and hence the dimension is Data. The internet and its connectivity has enabled individuals with access to huge amounts of data. The data generated, till date records the number of people infected, recovered and died. This nature and sequence of such collected data helps in building robust models to predict the spread of the pandemic in future. Statistical methods like time series analysis and mathematical models with least error and better fit have been reported in the literature.

Our project is based on the paper titled "COVID-19 Future Forecasting Using Supervised Machine Learning Models" which has been published in the Institute of Electrical and Electronics Engineers (IEEE) , which is a part of the Scopus Indexed List.

The main motivation behind the paper of "COVID-19 Future Forecasting Using Supervised Machine Learning Model" is the work of several other journals and researches which have been done previously by people renowned for their dedication and progressive work in the field of forecasting.

As mentioned ,our topic was based on the paper from the IEEE journal. Before that, there were many other instances of citations and conference papers where this topic has been involved. A few examples of these references is as:

- S. Makridakis, E. Spiliotis, and V. Assimakopoulos, ‘‘Statistical and machinelearningforecastingmethods:Concernsandwaysforward,’’PLoS ONE, vol. 13, no. 3, Mar. 2018, Art. no. e0194889.
- F. Petropoulos and S. Makridakis, ‘‘Forecasting the novel coronavirus COVID19,’’ PLoS ONE, vol. 15, no. 3, Mar. 2020, Art. no. e0231236.

3. PROBLEM ANALYSIS

Corona Virus disease (COVID-19) is an infectious disease caused by a newly discovered virus, which emerged in Wuhan, China in December of 2019. Most people infected with the COVID-19 virus will experience mild to moderate respiratory illness and recover without requiring special treatment.

3.1. PROBLEM STATEMENT

To forecast the number of upcoming patients affected by COVID-19 which is presently considered as a potential threat to mankind. Three types of predictions are made by each of the models, such as the number of newly infected cases, the number of deaths, and the number of recoveries in the next 10 days.

3.2. ROOT CAUSE

Older people and those with underlying medical problems like cardiovascular disease, diabetes, chronic respiratory disease, and cancer are more likely to develop serious illness. The COVID-19 virus spreads primarily through droplets of saliva or discharge from the nose when an infected person coughs or sneezes, so you might have heard caution to practice respiratory etiquette (for example, by coughing into a flexed elbow).

3.3. BOUNDARIES AND CONSTRAINTS

In particular, we are going to implement two standard forecasting models, such as linear regression (LR), and support vector machine (SVM) have been used in this study to forecast the threatening factors of COVID-19. Future predictions are made in order to obtain the number of confirmed cases, the number of deaths, and the number of recovered cases in the upcoming 10 days. The results produced by the study proves it to be a promising mechanism to use these methods for the current scenario of the COVID-19 pandemic.

4. DESIGN

4.1. NUMPY

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

The Python programming language was not originally designed for numerical computing, but attracted the attention of the scientific and engineering community early on. In 1995 the special interest group (SIG) matrix-sig was founded with the aim of defining an array computing package; among its members was Python designer and maintainer Guido van Rossum, who extended Python's syntax (in particular the indexing syntax) to make array computing easier.

An implementation of a matrix package was completed by Jim Fulton, then generalized[further explanation needed] by Jim Hugunin and called Numeric (also variously known as the "Numerical Python extensions" or "NumPy"). Hugunin, a graduate student at the Massachusetts Institute of Technology (MIT), joined the Corporation for National Research Initiatives (CNRI) in 1997 to work on JPython, leaving Paul Dubois of Lawrence Livermore National Laboratory (LLNL) to take over as maintainer. Other early contributors include David Ascher, Konrad Hinsen and Travis Oliphant.

A new package called Numarray was written as a more flexible replacement for Numeric.Like Numeric, it too is now deprecated. Numarray had faster operations for large arrays, but was slower than Numeric on small ones,so for a time both packages were used in parallel for different use cases. The last version of Numeric (v24.2) was released on 11 November 2005, while the last version of numarray (v1.5.2) was released on 24 August 2006.

There was a desire to get Numeric into the Python standard library, but Guido van Rossum decided that the code was not maintainable in its state then.

In early 2005, NumPy developer Travis Oliphant wanted to unify the community around a single array package and ported Numarray's features to Numeric, releasing the result as NumPy 1.0 in 2006.[9] This new project was part of SciPy. To avoid installing the large SciPy package just to get an array object, this new package was separated and called NumPy. Support for Python 3 was added in 2011 with NumPy version 1.5.0.

In 2011, PyPy started development on an implementation of the NumPy API for PyPy. It is not yet fully compatible with NumPy.

4.1.1. <u>FEATURES</u>

NumPy targets the CPython reference implementation of Python, which is a nonoptimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted,[18] and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLABlike plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging**.**

The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. These arrays are strided views on memory.[9] In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries (notably BLAS and LAPACK). NumPy has built-in support for memory-mapped ndarrays.

4.2. PANDAS

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

4.2.1. FEATURES

- Fast and efficient Data Frame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

4.3. MATPLOTLIB

Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers also.

Matplotlib has a procedural interface named the Pylab, which is designed to resemble MATLAB, a proprietary programming language developed by MathWorks. Matplotlib along with NumPy can be considered as the open source equivalent of MATLAB.

Matplotlib was originally written by John D. Hunter in 2003. The current stable version is 2.2.0 released in January 2018.

Pyplot is a Matplotlib module which provides a MATLAB-like interface.

Matplotlib is designed to be as usable as MATLAB, with the ability to use Python, and the advantage of being free and open-source.

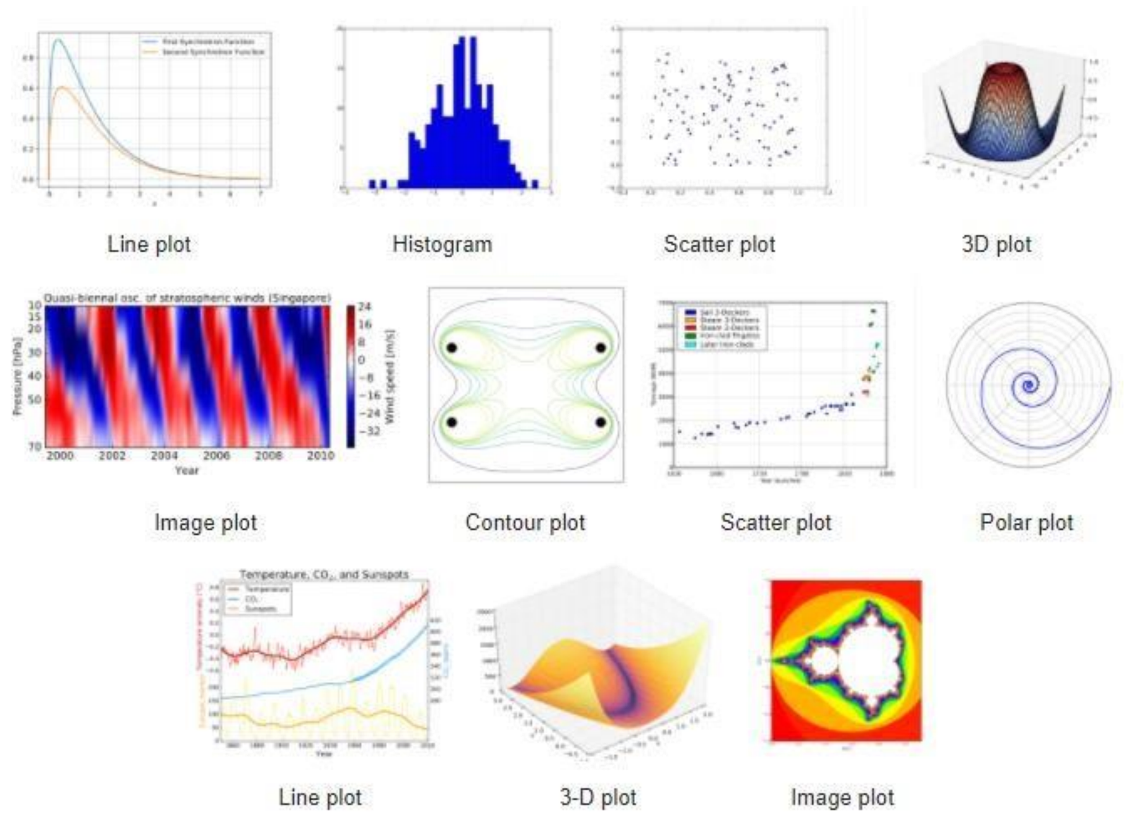


Figure 4.3.1 : Types of plotting

4.3.1. <u>TOOLKIT</u>

Several toolkits are available which extend Matplotlib functionality. Some are separate downloads, others ship with the Matplotlib source code but have external dependencies.

- **Basemap:** map plotting with various map projections, coastlines, and political boundaries
- **Cartopy:** a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities. (Matplotlib v1.2 and above)
- **Excel tools:** utilities for exchanging data with Microsoft Excel
- **GTK tools:** interface to the GTK+ library

- **Qt** interface
- **Mplot3d:** 3-D plots
- **Natgrid:** interface to the natgrid library for gridding irregularly spaced data.
- **matplotlib2tikz:** export to Pgfplots for smooth integration into LaTeX documents·
- **Seaborn:** provides an API on top of Matplotlib that offers sane choices for plot style and color defaults, defines simple high-level functions for common statistical plot types, and integrates with the functionality provided by Pandas.

Matplotlib is a multi-platform data visualization library built on NumPy arrays, and designed to work with the broader SciPy stack. It was conceived by John Hunter in 2002, originally as a patch to IPython for enabling interactive MATLAB-style plotting via gnuplot from the IPython command line. IPython's creator, Fernando Perez, was at the time scrambling to finish his PhD, and let John know he wouldn't have time to review the patch for several months. John took this as a cue to set out on his own, and the Matplotlib package was born, with version 0.1 released in 2003. It received an early boost when it was adopted as the plotting package of choice of the Space Telescope Science Institute (the folks behind the Hubble Telescope), which financially supported Matplotlib's development and greatly expanded its capabilities.

One of Matplotlib's most important features is its ability to play well with many operating systems and graphics backends. Matplotlib supports dozens of backends and output types, which means you can count on it to work regardless of which operating system you are using or which output format you wish. This cross-platform, everythingto-everyone approach has been one of the great strengths of Matplotlib. It has led to a large user base, which in turn has led to an active developer base and Matplotlib's powerful tools and ubiquity within the scientific Python world.

In recent years, however, the interface and style of Matplotlib have begun to show their age. Newer tools like ggplot and ggvis in the R language, along with web visualization toolkits based on D3js and HTML5 canvas, often make Matplotlib feel clunky and oldfashioned. Still, I'm of the opinion that we cannot ignore Matplotlib's strength as a welltested, cross-platform graphics engine. Recent Matplotlib versions make it relatively easy to set new global plotting styles (see Customizing Matplotlib: Configurations and Style Sheets), and people have been developing new packages that build on its powerful internals to drive Matplotlib via cleaner, more modern APIs—for example, Seaborn (discussed in Visualization With Seaborn), ggpy, HoloViews, Altair, and even Pandas itself can be used as wrappers around Matplotlib's API. Even with wrappers like these, it is still often useful to dive into Matplotlib's syntax to adjust the final plot output. For this reason, I believe that Matplotlib itself will remain a vital piece of the data visualization stack, even if new tools mean the community gradually moves away from using the Matplotlib API directly.

Matploitlib is a Python Library used for plotting, this python library provides and objected-oriented APIs for integrating plots into applications.

Matplotlib is not a part of the Standard Libraries which is installed by default when Python, there are several toolkits which are available that extend python matplotlib functionality. Some of them are separate downloads, others can be shipped with the matplotlib source code but have external dependencies.

4.4. RANDOM

The Python random module functions depend on a pseudo-random number generator function random(), which generates the float number between 0.0 and 1.0.

There are different types of functions used in a random module which is given below:

- **random.random()**

This function generates a random float number between 0.0 and 1.0.

- **random.randint()**

This function returns a random integer between the specified integers.

- **random.choice()**

This function returns a randomly selected element from a non-empty sequence.

- **random.shuffle()**

This function randomly reorders the elements in the list.

- **random.randrange(beg,end,step)**

This function is used to generate a number within the range specified in its argument. It accepts three arguments, beginning number, last number, and step, which is used to skip a number in the range.

4.5. MATH

The **math module** is a built-in Python library. It helps in numerous *mathematical* functions following the C standard. This works for *real* numbers only.

4.5.1. BASIC FUNCTIONS

Functions	Description
math.ceil(value)	It returns the smallest integer greater than or equal to *value*.
math.copysign(value1,value2)	It returns the float with magnitude of *value1* and sign of *value2*.

math.fabs(value)	It returns the absolute of *value*.
math.factorial(value)	It returns the factorial of *value*.
math.floor(value)	It returns the largest integer greater than or equal to *value*.
math.fsum(iterable)	It returns the float sum of the values of the iterable.

4.6. SKLEARN

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon **NumPy, SciPy** and **Matplotlib**.

It was originally called ***scikits.learn*** and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael

Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

Let's have a look at its version history −

- May 2019: scikit-learn 0.21.0
- March 2019: scikit-learn 0.20.3
- December 2018: scikit-learn 0.20.2
- November 2018: scikit-learn 0.20.1
- September 2018: scikit-learn 0.20.0
- July 2018: scikit-learn 0.19.2
- July 2017: scikit-learn 0.19.0
- September 2016. scikit-learn 0.18.0
- November 2015. scikit-learn 0.17.0
- March 2015. scikit-learn 0.16.0
- July 2014. scikit-learn 0.15.0
- August 2013. scikit-learn 0.14

4.6.1. FEATURES

Rather than focusing on loading, manipulating and summarising data, Scikit-learn library is focused on modeling the data. Some of the most popular groups of models provided by Sklearn are as follows −

- **Supervised Learning algorithms** − Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree etc., are the part of scikit-learn.
- **Unsupervised Learning algorithms** − On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.
- **Clustering** − This model is used for grouping unlabeled data.
- **Cross Validation** − It is used to check the accuracy of supervised models on unseen data.

- **Dimensionality Reduction** − It is used for reducing the number of attributes in data which can be further used for summarisation, visualisation and feature selection.
- **Ensemble methods** − As name suggest, it is used for combining the predictions of multiple supervised models.
- **Feature extraction** − It is used to extract the features from data to define the attributes in image and text data.
- **Feature selection** − It is used to identify useful attributes to create supervised models.
- **Open Source** − It is open source library and also commercially usable under BSD license.

4.7. DATETIME

In Python, date and time are not a data type of its own, but a module named **datetime** can be imported to work with the date as well as time. **Datetime module** comes built into Python, so there is no need to install it externally.

Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so when you manipulate them, you are actually manipulating objects and not string or timestamps.

The datetime classes are categorize into 6 main classes –

- **date** – An idealized naive date, assuming the current Gregorian calendar always was, and always will be, in effect. Its attributes are year, month and day.
- **time** – An idealized time, independent of any particular day, assuming that every day has exactly 24*60*60 seconds. Its attributes are hour, minute, second, microsecond, and tzinfo.
- **datetime** – Its a combination of date and time along with the attributes year, month, day, hour, minute, second, microsecond, and tzinfo.
- **timedelta** – A duration expressing the difference between two date, time, or datetime instances to microsecond resolution.
- **tzinfo** – It provides time zone information objects.

- **timezone** – A class that implements the tzinfo abstract base class as a fixed offset from the UTC (New in version 3.2).

4.8. OPERATOR

The operator module supplies functions that are equivalent to Python's operators. These functions are handy in cases where callables must be stored, passed as arguments, or returned as function results. The functions in operator have the same names as the corresponding special methods (covered in Special Methods). Each function is available with two names, with and without leading and trailing double underscores (e.g., both operator.add(*a*,*b*) and operator._ _add_ _(*a*,*b*) return *a*+*b*).

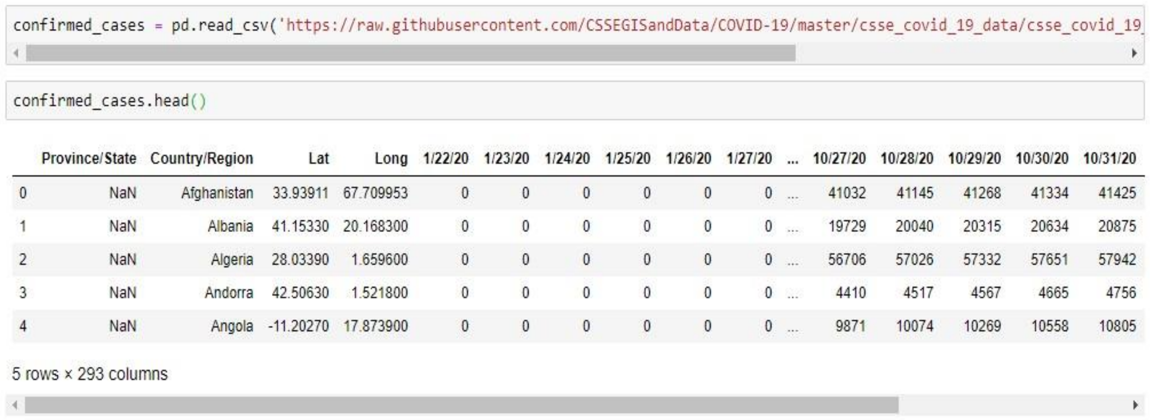
5. IMPLEMENTATION

5.1. DATASET

The aim of this study is the future forecasting of COVID-19 spread focusing on the number of new positive cases, the number of deaths, and the number of recoveries. The dataset used in the study has been obtained from the GitHub repository provided by the Center for Systems Science and Engineering, Johns Hopkins University [12]. The repository was primarily made available for the visual dashboard of 2019 Novel Coronavirus by the university and was supported by the ESRI Living Atlas Team. Dataset files are contained in the folder on the GitHub repository named 'csse_covid_19_time_series'. The folder contains daily time series summary tables, including the number of confirmed cases, deaths, and recoveries. All data are from the daily case report and the update frequency of data is one day.

Note:

- The total number of columns present in the time-series summary tables include Province/State, Country/Region, Lat (Latitudinal position), Long (Longitudinal position), and day-to-day case report starting from 1/22/20 (i.e., 22nd January, 2020) till date.
- The total number of rows are equivalent to the first row (i.e., column name), number of countries and states.



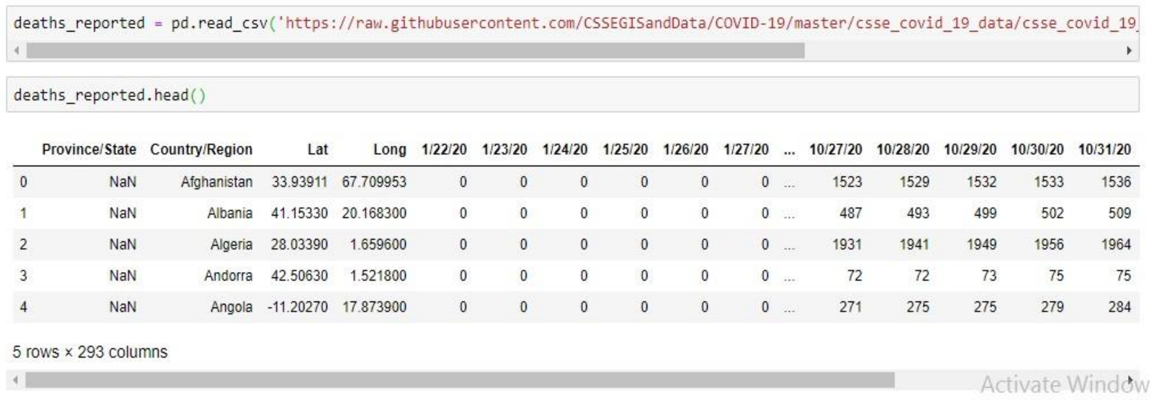
```
confirmed_cases = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_
```

```
confirmed_cases.head()
```

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/27/20	10/28/20	10/29/20	10/30/20	10/31/20
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...	41032	41145	41268	41334	41425
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...	19729	20040	20315	20634	20875
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	...	56706	57026	57332	57651	57942
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	...	4410	4517	4567	4665	4756
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	...	9871	10074	10269	10558	10805

5 rows × 293 columns

Figure 5.1.1 : COVID-19 patient confirmed cases time-series worldwide



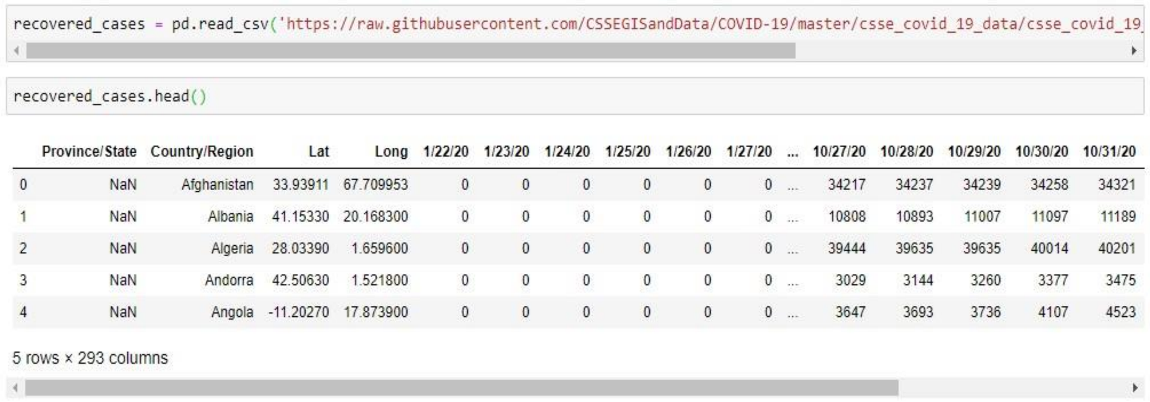
```
deaths_reported = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_
```

```
deaths_reported.head()
```

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/27/20	10/28/20	10/29/20	10/30/20	10/31/20
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...	1523	1529	1532	1533	1536
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...	487	493	499	502	509
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	...	1931	1941	1949	1956	1964
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	...	72	72	73	75	75
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	...	271	275	275	279	284

5 rows × 293 columns

Figure 5.1.2 : COVID-19 patient death cases time-series worldwide



```
recovered_cases = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_
```

```
recovered_cases.head()
```

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/27/20	10/28/20	10/29/20	10/30/20	10/31/20
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...	34217	34237	34239	34258	34321
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...	10808	10893	11007	11097	11189
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	...	39444	39635	39635	40014	40201
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	...	3029	3144	3260	3377	3475
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	...	3647	3693	3736	4107	4523

5 rows × 293 columns

Figure 5.1.3 : COVID-19 patient recovered cases time-series worldwide

5.2. ARCHITECTURE

The study is about novel coronavirus also known as COVID-19 predictions. The COVID-19 has proved a present potential threat to human life. It causes tens of thousands of deaths and the death rate is increasing day by day throughout the globe. To contribute to this pandemic situation control, this study attempts to perform future forecasting on the death rate, the number of daily confirmed infected cases and the number of recovery cases in the upcoming 10 days.

The forecasting has been done by using four ML approaches that are appropriate to this context. The dataset used in the study contains daily time series summary tables, including the number of confirmed cases, deaths, and recoveries in the past number of days from which the pandemic started. Initially, the dataset has been preprocessed for this study to find the global statistics of the daily number of deaths, confirmed cases, and recoveries.

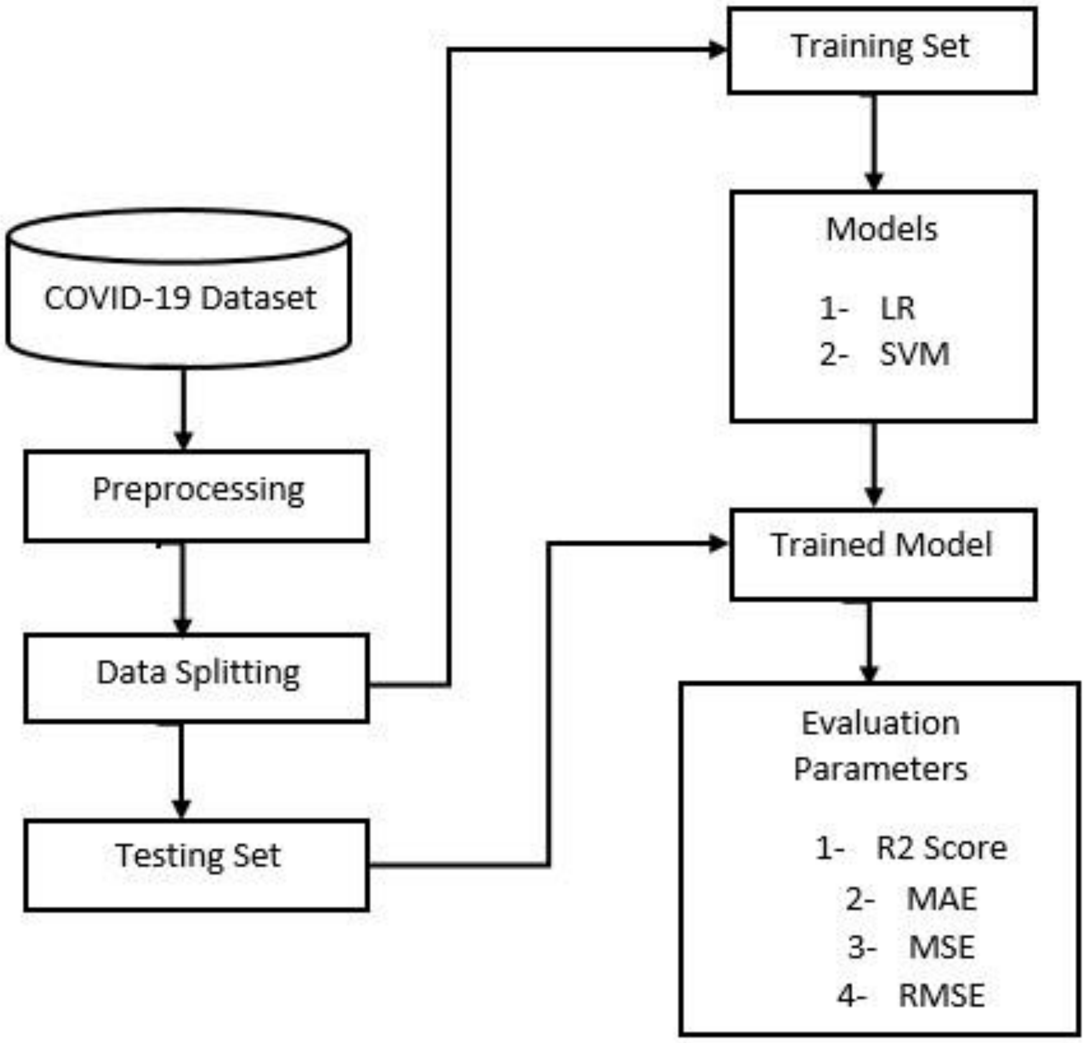


Figure 5.2.1 : Workflow

At the initial step, preprocessing of the data is done. After this, the data has been further divided into two subsets: the training set and the testing set. The models such as Linear Regression (LR) and the Support Vector Machine (SVM) have been used in this study. These models have been trained on the days and the confirmed cases, the number of deaths, and the number of recovered cases are predicted for the upcoming 10 days. These models are then evaluated against some parameters such as R^2-score, MSE, MAE, and RMSE and recorded in the results.

5.3. SUPERVISED MACHINE LEARNING MODELS

Supervised learning is the most common subbranch of machine learning today. Typically, new machine learning practitioners will begin their journey with supervised learning algorithms. Therefore, the first of this three post series will be about supervised learning.

Supervised machine learning algorithms are designed to learn by example. The name "supervised" learning originates from the idea that training this type of algorithm is like having a teacher supervise the whole process.

When training a supervised learning algorithm, the training data will consist of inputs paired with the correct outputs. During training, the algorithm will search for patterns in the data that correlate with the desired outputs. After training, a supervised learning algorithm will take in new unseen inputs and will determine which label the new inputs will be classified as based on prior training data. The objective of a supervised learning model is to predict the correct label for newly presented input data. At its most basic form, a supervised learning algorithm can be written simply as:

$$Y = f(x)$$

Where *Y* is the predicted output that is determined by a mapping function that assigns a class to an input value *x*. The function used to connect input features to a predicted output is created by the machine learning model during training.

Supervised learning can be split into two subcategories: Classification and regression.

Regression is a predictive statistical process where the model attempts to find the important relationship between dependent and independent variables. The goal of a regression algorithm is to predict a continuous number such as sales, income, and test scores. The equation for basic linear regression can be written as so:

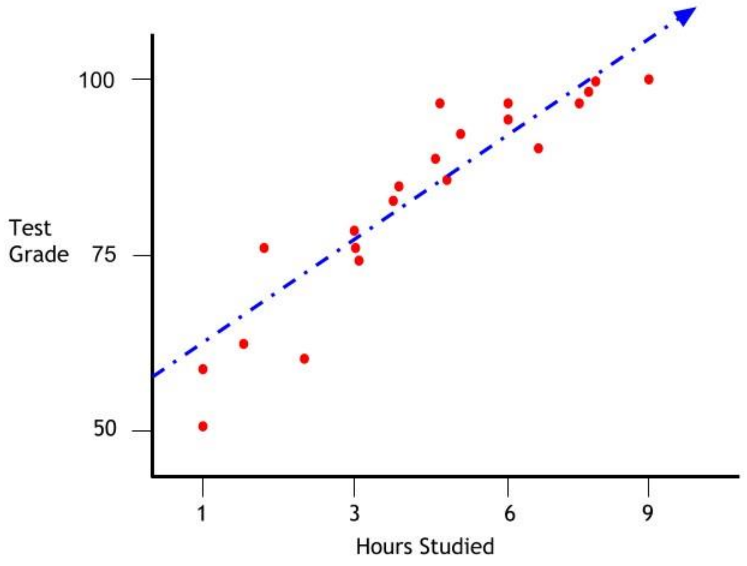
$$\hat{y} = w[0] * x[0] + w[1] * x[1] + \cdots + w[i] * x[i] + b$$

Where *x[i]* is the feature(s) for the data and where *w[i]* and *b* are parameters which are developed during training. For simple linear regression models with only one feature in the data, the formula looks like this:

$$\hat{y} = wx + b$$

Where *w* is the slope, *x* is the single feature and *b* is the y-intercept. Familiar? For simple regression problems such as this, the models predictions are represented by the line of best fit. For models using two features, the plane will be used. Finally, for a model using more than two features, a hyperplane will be used.

Imagine we want to determine a student’s test grade based on how many hours they studied the week of the test. Lets say the plotted data with a line of best fit looks like this:



There is a clear positive correlation between hours studied (independent variable) and the student’s final test score (dependent variable). A line of best fit can be drawn through the data points to show the models predictions when given a new input. Say we wanted to know how well a student would do with five hours of studying. We can use the line of best fit to predict the test score based on other student’s performances.

A supervised learning model is built to make a prediction when it is provided with an unknown input instance. Thus in this learning technique, the learning algorithm takes a dataset with input instances along with their corresponding regressor to train the regression model. The trained model then generates a prediction for the given unforeseen input data or test dataset. This learning method may use regression techniques and classification algorithms for predictive models' development.

Regression models used in this study of COVID-19 future forecasting:

• Linear Regression

• Support Vector Machine

5.3.1. LINEAR REGRESSION

In regression modeling, a target class is predicated on the independent features. This method can be thus used to find out the relationship between independent and dependent variables and also for forecasting. Linear regression a type of regression modeling is the most usable statistical technique for predictive analysis in machine learning. Each observation in linear regression depends on two values, one is the dependent variable and the second is the independent variable. Linear regression determines a linear relationship between these dependent and independent variables. There are two factors (x, y) that are involved in linear regression analysis. The equation below shows how y is related to x known as regression.

$$y = \beta_0 + \beta_1 x + \varepsilon \qquad (1)$$

or equivalently

$$E(y) = \beta_0 + \beta_1 x \qquad (2)$$

Here, ε is the error term of linear regression. The error term here uses to account the variability between both x and y, β0 represents y-intercept, β1 represents slope.

To put the concept of linear regression in the machine learning context, in order to train the model x is represented as input training dataset, y represents the class labels present in the input dataset. The goal of the machine learning algorithm then is to find the best values for β_0 (intercept) and β_1 (coefficient) to get the best-fit regression line. To get the best fit implies the difference between the actual values and predicted values should be minimum, so this minimization problem can be represented as:

$$minimize \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2 \qquad (3)$$

$$g = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2 \qquad (4)$$

Here, g is called a cost function, which is the root mean square of the predicted value of y ($pred_i$) and actual y (y_i), n is the total number of data points.

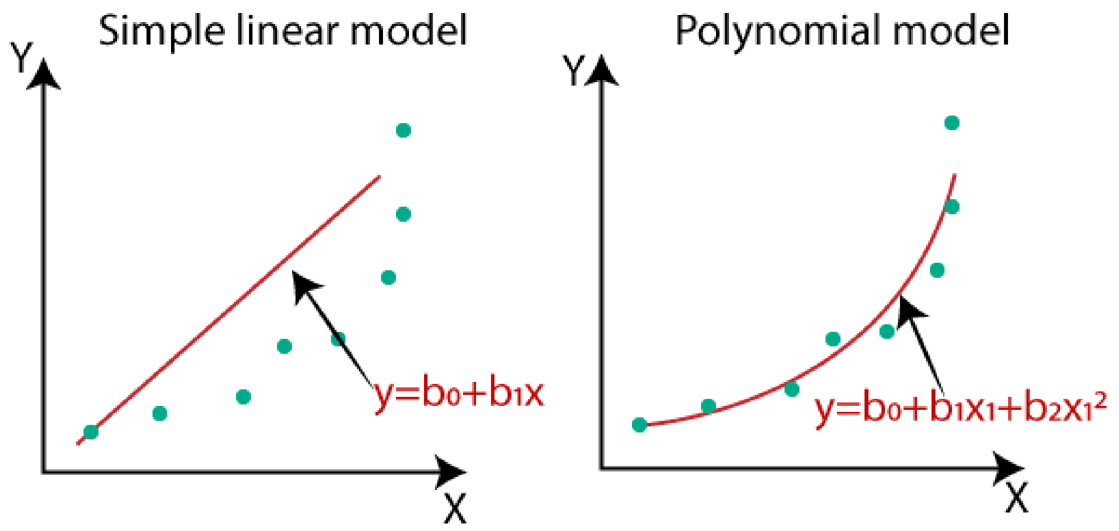
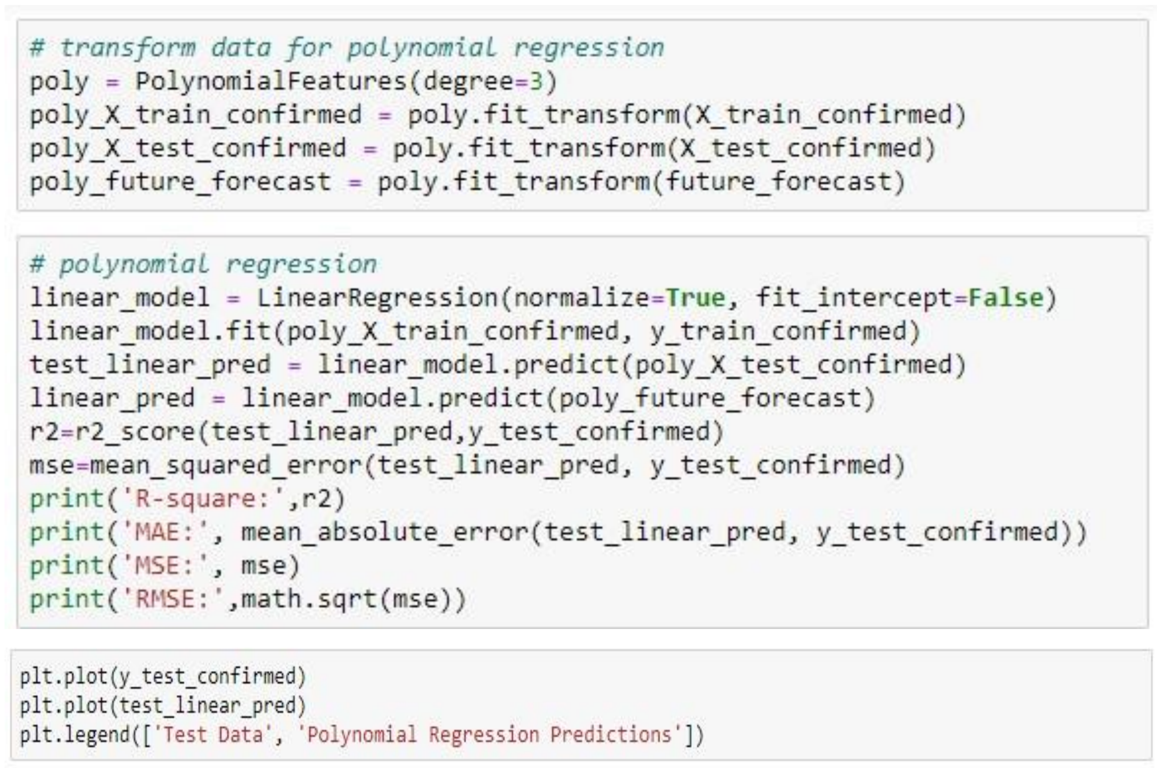


Figure 4.3.1.1 : Simple linear model vs Polynomial model

In this study, polynomial regression model has been used. A polynomial term– a quadratic (squared) or cubic (cubed) term turns a linear regression model into a curve. But because it is X that is squared or cubed, not the Beta coefficient, it still qualifies as a linear model. This makes it a nice, straightforward way to model curves without having to model complicated non-linear models. Our ultimate goal is always to build a model with maximum accuracy and minimum errors. We can implement polynomial regression that best fits our data by using curves. Some basic polynomial functions with its graphs plotted.

5.3.1.1. <u>SOURCE CODE</u>

Machine learning strategies are performed using the python library to predict the total number of confirmed, recovered, and death cases extensively. This prediction will allow undertaking specific determinations based on transmission growth, such as expanding the lockdown phase, performing the sanitation plan, and providing daily support and supplies.



```
# transform data for polynomial regression
poly = PolynomialFeatures(degree=3)
poly_X_train_confirmed = poly.fit_transform(X_train_confirmed)
poly_X_test_confirmed = poly.fit_transform(X_test_confirmed)
poly_future_forecast = poly.fit_transform(future_forecast)
```

```
# polynomial regression
linear_model = LinearRegression(normalize=True, fit_intercept=False)
linear_model.fit(poly_X_train_confirmed, y_train_confirmed)
test_linear_pred = linear_model.predict(poly_X_test_confirmed)
linear_pred = linear_model.predict(poly_future_forecast)
r2=r2_score(test_linear_pred,y_test_confirmed)
mse=mean_squared_error(test_linear_pred, y_test_confirmed)
print('R-square:',r2)
print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
print('MSE:', mse)
print('RMSE:',math.sqrt(mse))
```

```
plt.plot(y_test_confirmed)
plt.plot(test_linear_pred)
plt.legend(['Test Data', 'Polynomial Regression Predictions'])
```

Figure 5.3.1.1 : Linear Regression (Source code)

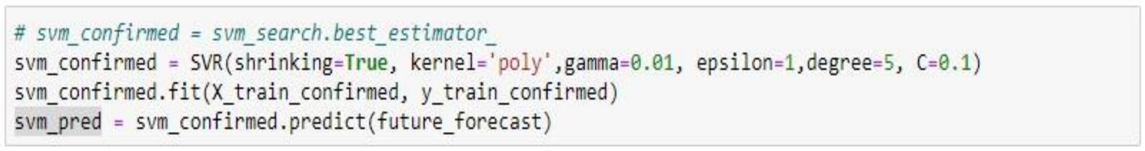
5.3.2. SUPPORT VECTOR MACHINE

A support vector machine (SVM) is a type of supervised ML algorithm used for both regression and classification. SVM regression being a non-parametric technique depends on a set of mathematical functions. The set of functions called kernel transforms the data inputs into the desired form. SVM solves the regression problems using a linear function, so while dealing with problems of non-linear regression, it maps the input vector(x) non-dimensional space called a feature space(z).This mapping is done by nonlinear mapping techniques after that linear regression is applied to space. Putting the concept in ML context with a multivariate training dataset (x_n) with N number of observations with y_n as a set of observed responses. The linear function can be depicted as:

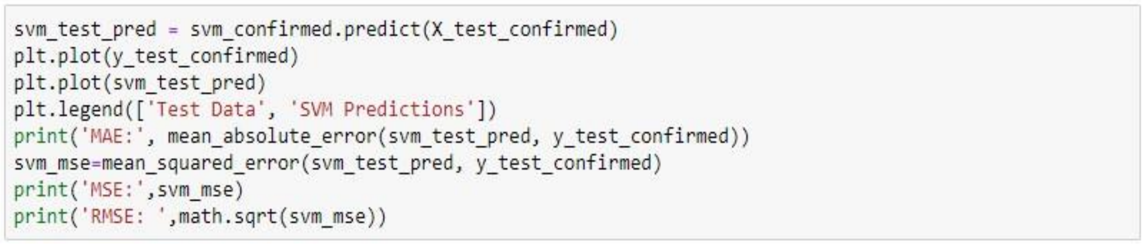
$$f(x) = x'\beta + b \quad (5)$$

5.3.2.1. SOURCE CODE

Machine learning strategies are performed using the python library to predict the total number of confirmed, recovered, and death cases extensively. This prediction will allow undertaking specific determinations based on transmission growth, such as expanding the lockdown phase, performing the sanitation plan, and providing daily support and supplies.



```
# svm_confirmed = svm_search.best_estimator_
svm_confirmed = SVR(shrinking=True, kernel='poly',gamma=0.01, epsilon=1,degree=5, C=0.1)
svm_confirmed.fit(X_train_confirmed, y_train_confirmed)
svm_pred = svm_confirmed.predict(future_forecast)
```



```
svm_test_pred = svm_confirmed.predict(X_test_confirmed)
plt.plot(y_test_confirmed)
plt.plot(svm_test_pred)
plt.legend(['Test Data', 'SVM Predictions'])
print('MAE:', mean_absolute_error(svm_test_pred, y_test_confirmed))
svm_mse=mean_squared_error(svm_test_pred, y_test_confirmed)
print('MSE:',svm_mse)
print('RMSE: ',math.sqrt(svm_mse))
```

Figure 5.3.2.1 : Support Vector Machine (Source code)

5.4. EVALUATION PARAMETERS

In this study, we evaluate the performance of each of the learning models in terms of R-squared (R2) score, mean square error (MSE), mean absolute error (MAE), and root mean square error (RMSE).

5.4.1. R-SQUARED SCORE

R-squared (R2) score is a statistical measure used to evaluate the performance of regression models. The statistic shows the dependent variable's variance percentage that collectively determines the independent variable. It measures the relationship strength between the dependent variable and regression models on a convenient 0 – 100% scale. After training the regression model, we can check the goodness of-fit of trained models by using the R2 score. R2 score finds the scatteredness of data points around the regression line which can also be referred to as the coefficient of determination. Its score always between 0 and 100%. 0% score implies the response variable has no variability around its mean explained by the model, and 100% implies that the response variable has all the variability around its mean. The high R2 score shows the goodness of the trained model. R2 is a linear model that explains the percentage of variation independent variable. It can be found as:

$$R^2 = \frac{Varianceexplainedbymodel}{Totalvariance} \quad (6)$$

5.4.2. MEAN ABSOLUTE ERROR (MAE)

The mean absolute error is the average magnitude of the errors in the set of model predictions [22], [23]. This is an average on test data between the model predictions and actual data where all individual differences have equal weight. Its matrix value range is from 0 to infinity and fewer score values show the goodness of learning models that's the reason it's also called negatively-oriented scores.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j| \quad (7)$$

This measures the absolute average distance **between** the real data and the predicted data, but it fails to punish large **errors** in prediction.

5.4.3. MEAN SQUARE ERROR (MSE)

Mean square error is another way to measure the performance of regression models [22]. MSE takes the distance of data points from the regression line and squaring them. Squaring is necessary because it removes the negative sign from the value and gives more weight to larger differences. The smaller mean squared error shows the closer you are to finding the line of best fit. MSE can be calculated as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (8)$$

This measures the squared average distance between the real data and the predicted data.

5.4.4. ROOT MEAN SQUARE ERROR (RMSE)

Root mean square error can be defined as the standard deviation of the prediction errors. Prediction errors also known as residuals is the distance from the best fit line and actual data points. RMSE is thus a measure of how concentrated the actual data points are around the best fit line. It is the error rate given by the square root of MSE given as follows.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (9)$$

It is the square root of the mean of the square of all of the error. The use of RMSE is very common, and it is considered an excellent general-purpose error metric for numerical predictions.

6. TESTING AND VALIDATION

This study attempts to develop a system for the future forecasting of the number of cases affected by COVID-19 using machine learning methods. The dataset used for the study contains information about the daily reports of the number of newly infected cases, the number of recoveries, and the number of deaths due to COVID-19 worldwide. As the death rate and confirmed cases are increasing day by day which is an alarming situation for the world. The number of people who can be affected by the COVID-19 pandemic in different countries of the world is not well known. This study is an attempt to forecast the number of people that can be affected in terms of new infected cases and deaths including the number of expected recoveries for the upcoming 10 days.

6.1. FUTURE FORECASTING OF CONFIRM CASES

The new confirmed cases of COVID-19 increase day by day, the table shows the forecasting results of the models used in this study. LR also performs well, while SVM performs poor. Graphs in figures below show the predictions of learning models.

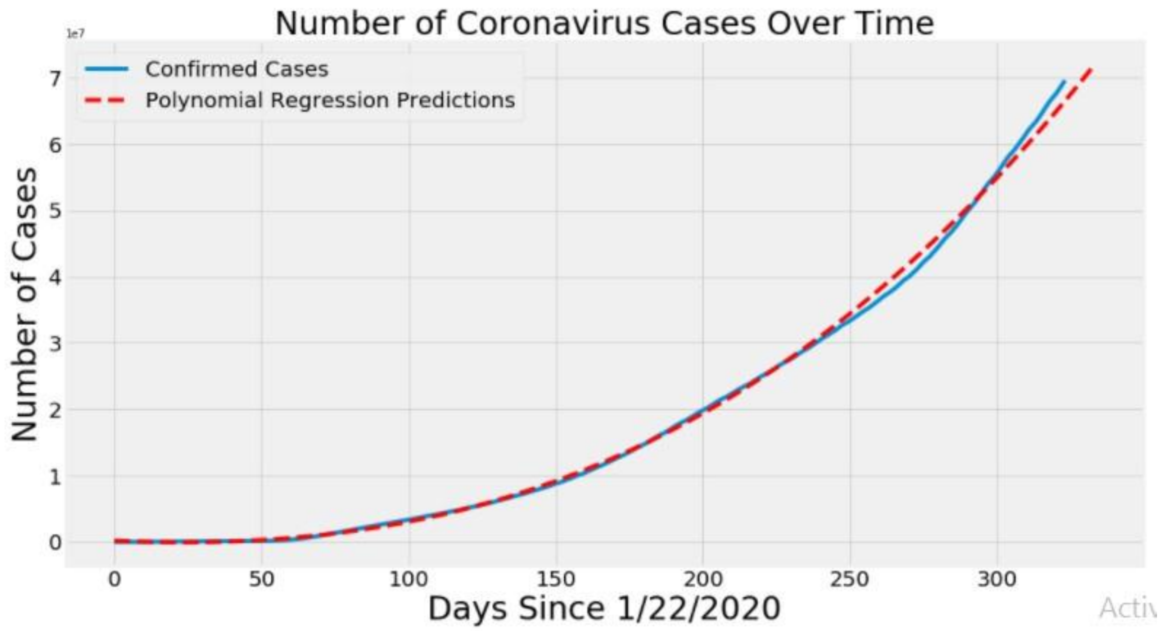


Figure 6.1.1 : Confirmed cases prediction by LR for the upcoming 10 days.

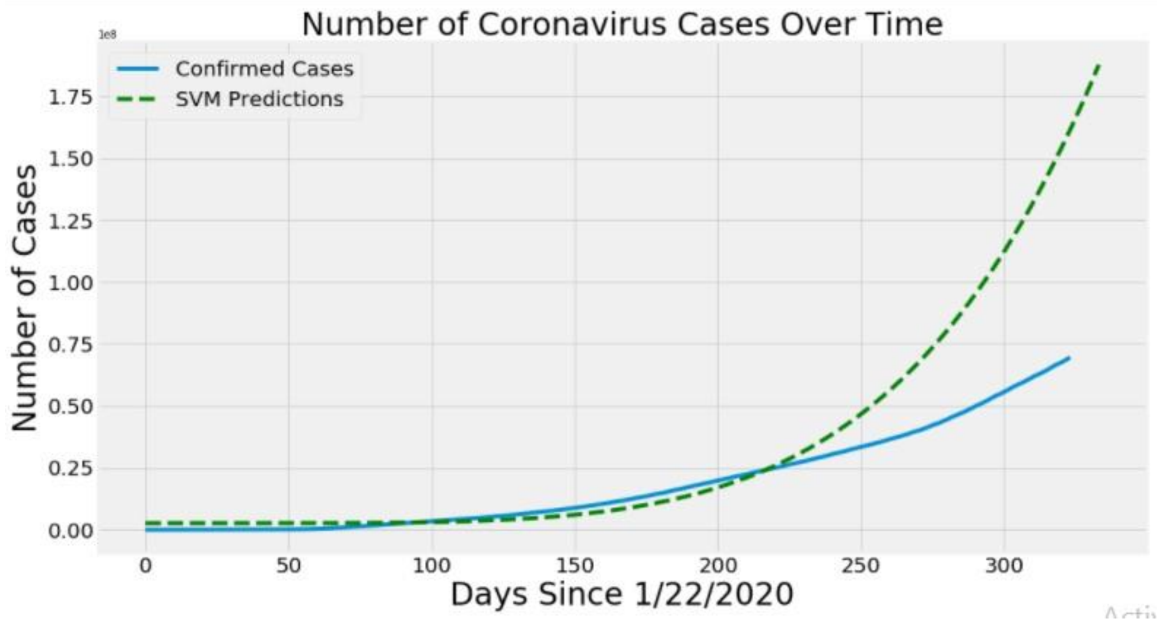


Figure 6.1.2 : Confirmed cases prediction by SVM for the upcoming 10 days.

6.2. FUTURE FORECASTING OF DEATH CASES

The death cases of COVID-19 increase day by day the table shows the forecasting results of the models used in this study. LR also performs well, while SVM performs poor. Graphs in figures below show the predictions of learning models.

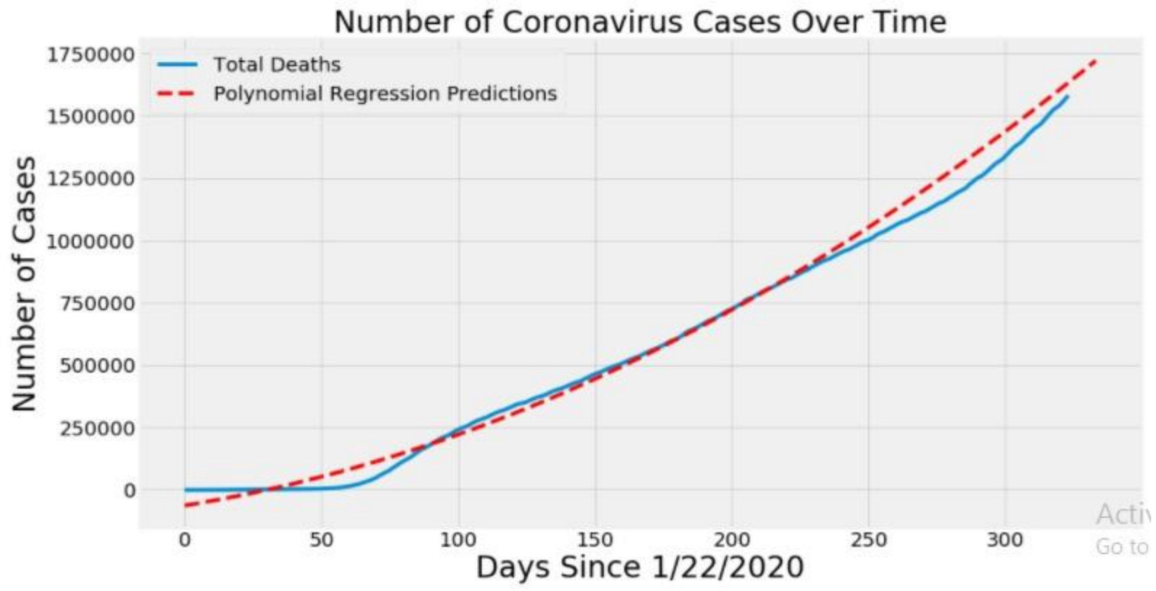


Figure 6.2.1 : Death cases prediction by LR for the upcoming 10 days.

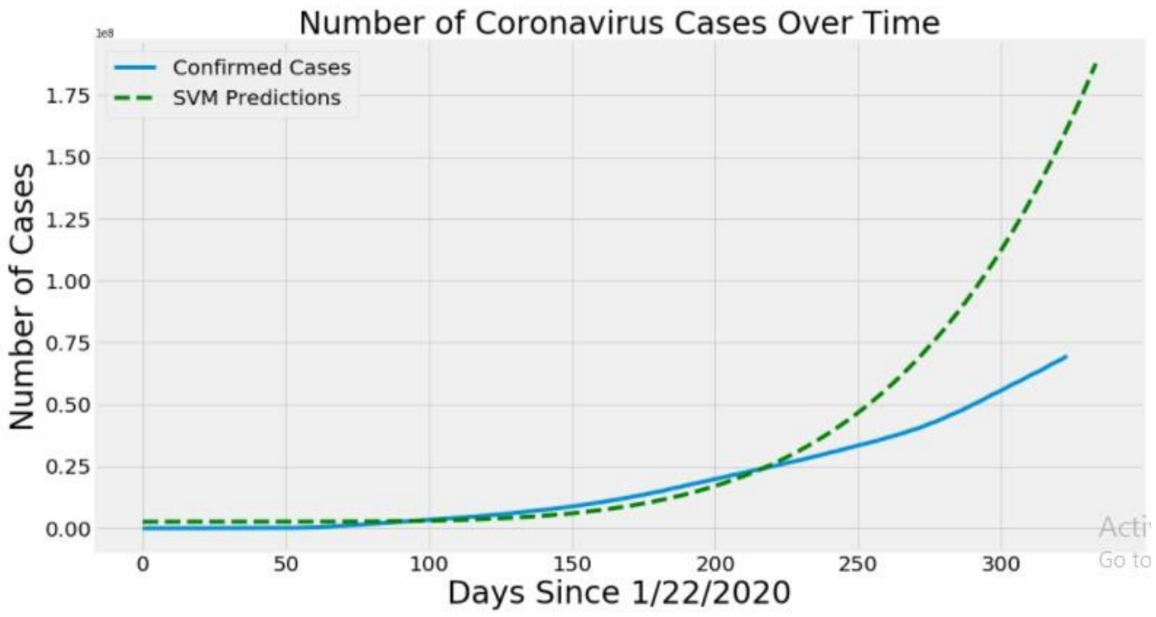


Figure 6.2.2 : Death cases prediction by SVM for the upcoming 10 days.

6.3. <u>FUTURE FORECASTING OF RECOVERED CASES</u>

The recovered cases of COVID-19 increase day by day the table shows the forecasting results of the models used in this study. LR also performs well, while SVM performs poor. Graphs in figures below show the predictions of learning models.

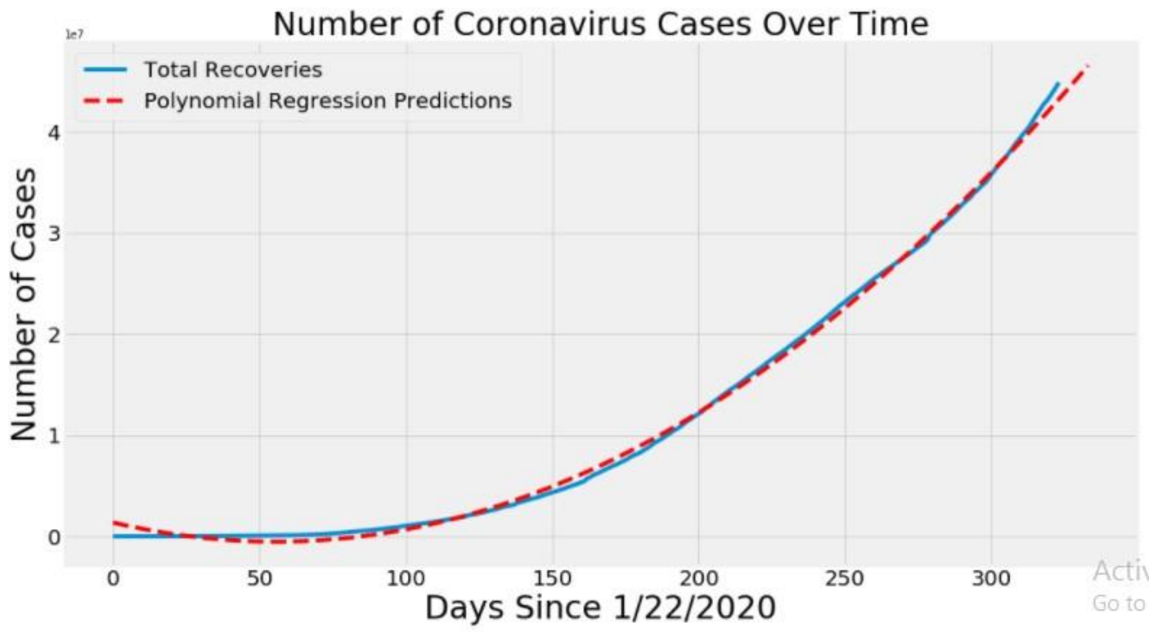


Figure 6.3.1 : Recovered cases prediction by LR for the upcoming 10 days

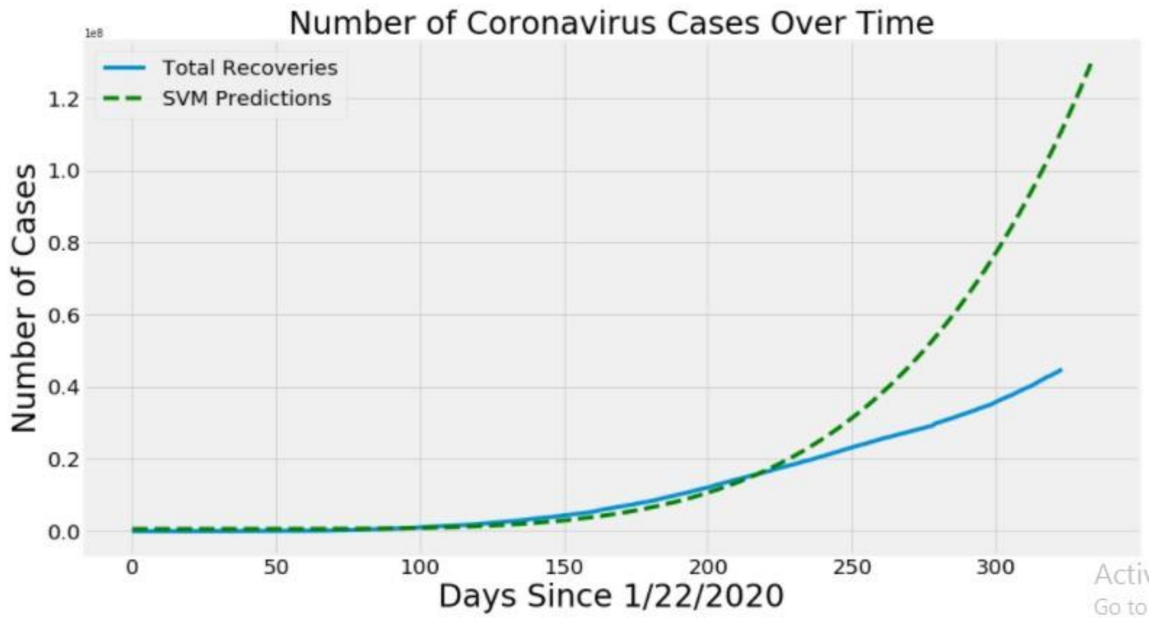
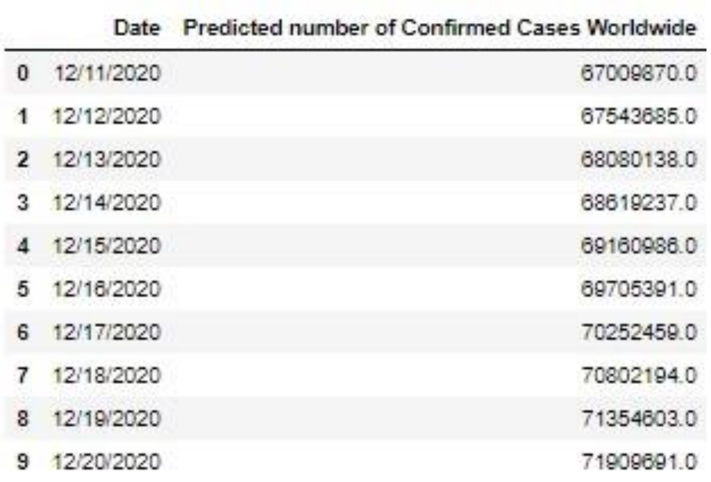


Figure 6.3.2 : Recovered cases prediction by SVM for the upcoming 10 days

7. RESULT

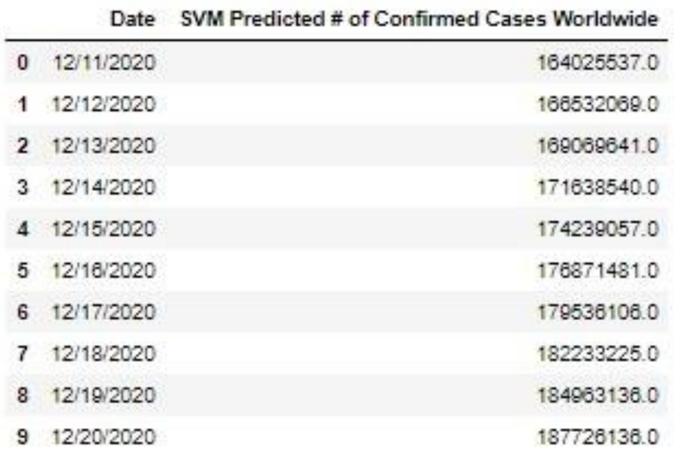
7.1. PREDICTED NUMBER OF CONFIRMED CASES

The confirmed cases of COVID-19 have been predicted for the upcoming 10 days using LR as well as SVM.



	Date	Predicted number of Confirmed Cases Worldwide
0	12/11/2020	67009870.0
1	12/12/2020	67543685.0
2	12/13/2020	68080138.0
3	12/14/2020	68619237.0
4	12/15/2020	69160986.0
5	12/16/2020	69705391.0
6	12/17/2020	70252459.0
7	12/18/2020	70802194.0
8	12/19/2020	71354603.0
9	12/20/2020	71909691.0

Figure 7.1.1 : Predicted number of Confirmed Cases using LR

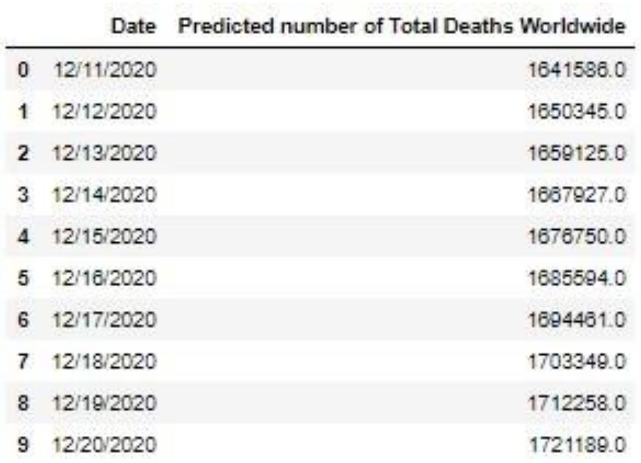


	Date	SVM Predicted # of Confirmed Cases Worldwide
0	12/11/2020	164025537.0
1	12/12/2020	166532069.0
2	12/13/2020	169069641.0
3	12/14/2020	171638540.0
4	12/15/2020	174239057.0
5	12/16/2020	176871481.0
6	12/17/2020	179536106.0
7	12/18/2020	182233225.0
8	12/19/2020	184963136.0
9	12/20/2020	187726136.0

Figure 7.1.2 : Predicted number of Confirmed Cases using SVM

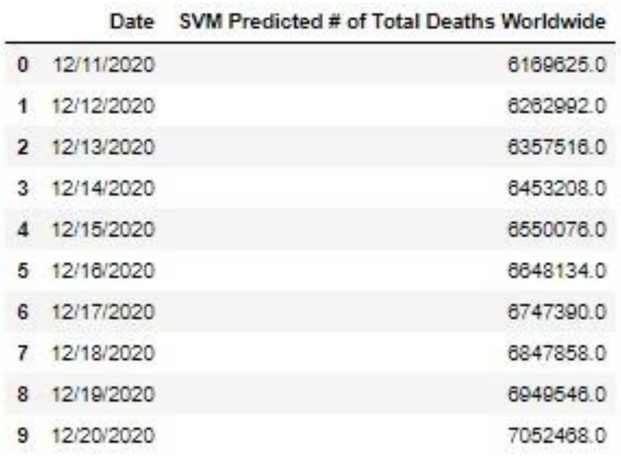
7.2. PREDICTED NUMBER OF DEATH CASES

The death cases of COVID-19 have been predicted for the upcoming 10 days using LR as well as SVM.



	Date	Predicted number of Total Deaths Worldwide
0	12/11/2020	1641586.0
1	12/12/2020	1650345.0
2	12/13/2020	1659125.0
3	12/14/2020	1667927.0
4	12/15/2020	1676750.0
5	12/16/2020	1685594.0
6	12/17/2020	1694461.0
7	12/18/2020	1703349.0
8	12/19/2020	1712258.0
9	12/20/2020	1721189.0

Figure 7.2.1 : Predicted number of Death Cases using LR

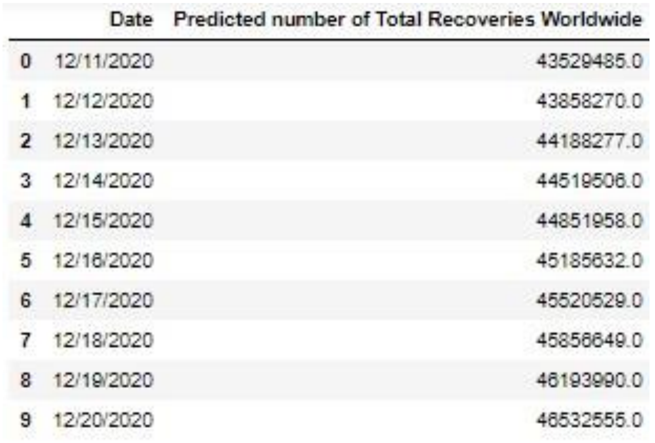


	Date	SVM Predicted # of Total Deaths Worldwide
0	12/11/2020	6169625.0
1	12/12/2020	6262992.0
2	12/13/2020	6357516.0
3	12/14/2020	6453208.0
4	12/15/2020	6550076.0
5	12/16/2020	6648134.0
6	12/17/2020	6747390.0
7	12/18/2020	6847858.0
8	12/19/2020	6949546.0
9	12/20/2020	7052468.0

Figure 7.2.2 : Predicted number of Death Cases using SVM

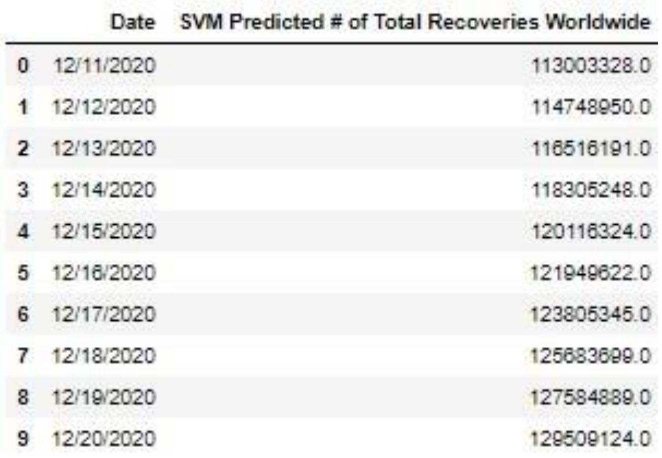
7.3. <u>PREDICTED NUMBER OF RECOVERED CASES</u>

The recovered cases of COVID-19 have been predicted for the upcoming 10 days using LR as well as SVM.



	Date	Predicted number of Total Recoveries Worldwide
0	12/11/2020	43529485.0
1	12/12/2020	43858270.0
2	12/13/2020	44188277.0
3	12/14/2020	44519506.0
4	12/15/2020	44851958.0
5	12/16/2020	45185632.0
6	12/17/2020	45520529.0
7	12/18/2020	45856649.0
8	12/19/2020	46193990.0
9	12/20/2020	46532555.0

Figure 7.3.1 : Predicted number of Recovered Cases using LR



	Date	SVM Predicted # of Total Recoveries Worldwide
0	12/11/2020	113003328.0
1	12/12/2020	114748950.0
2	12/13/2020	116516191.0
3	12/14/2020	118305248.0
4	12/15/2020	120116324.0
5	12/16/2020	121949622.0
6	12/17/2020	123805345.0
7	12/18/2020	125683699.0
8	12/19/2020	127584889.0
9	12/20/2020	129509124.0

Figure 7.3.2 : Predicted number of Recovered Cases using SVM

8. CONCLUSION

The precariousness of the COVID-19 pandemic can ignite a massive global crisis. Some researchers and government agencies throughout the world have apprehensions that the pandemic can affect a large proportion of the world population. In this study, an MLbased prediction system has been proposed for predicting the risk of COVID-19 outbreak globally. The system analyses dataset containing the day-wise actual past data and makes predictions for upcoming days using machine learning algorithms. The results of the study prove that LR performs best in the current forecasting domain given the nature and size of the dataset. LR performs well for forecasting to some extent to predict death rate and confirm cases. According to the results of this model, the death rates will increase in upcoming days, and recoveries rate will be slowed down. SVM produces poor results in all scenarios because of the ups and downs in the dataset values. It was very difficult to put an accurate hyper plane between the given values of the dataset. Overall we conclude that model predictions according to the current scenario are correct which may be helpful to understand the upcoming situation. The study forecasts thus can also be of great help for the authorities to take timely actions and make decisions to contain the COVID-19 crisis. This study will be enhanced continuously in the future course, next we plan to explore the prediction methodology using the updated dataset and use the most accurate and appropriate ML methods for forecasting. Real-time live forecasting will be one of the primary focuses in our future work.

9. REFERENCES

- https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data
- https://www.who.int/health-topics/coronavirus#tab=tab_1 http://www.stat.yale.edu/Courses/1997-98/101/linreg.htm#:~:text=Linear%20regression%20attempts%20to%20model,to%20be%20a%20dependent%20variable.
- https://en.wikipedia.org/wiki/Support_vector_machine
- https://en.wikipedia.org/wiki/Linear_regression
- https://realpython.com/linear-regression-in-python/#polynomial-regression-withscikit-learn

- https://monkeylearn.com/blog/introduction-to-support-vector-machinessvm/#:~:text=A%20support%20vector%20machine%20(SVM,on%20a%20text%20cl assification%20problem.
- https://towardsdatascience.com/a-brief-introduction-to-supervised-learning-54a3e3932590