IS 733

Data Mining

Project Topic

Forecast Covid 19

Group 9

Aishwarya Anil Kadam

Ria Vasa

Sonal Ingle

Surbhi Singhal

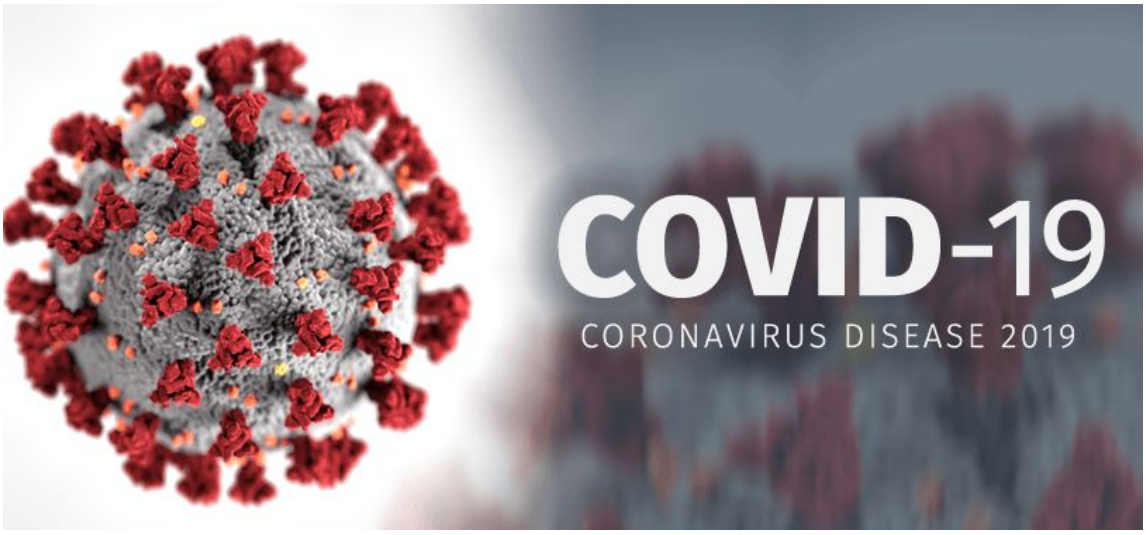


Table of Content

1. Abstract:

The world is in a crisis, currently dealing with a global pandemic in the form of COVID 19. The spread of COVID 19 in the whole world has put humanity at risk. The resources of some largest economies are stressed out by the large infectivity and transmissibility of this disease. The growing number of cases has caused stress on administration and health professionals, so some prediction methods would be required to predict the numbers of cases and factors that cause an increase in cases.

So we aim to use a predictive model to analyze the historical data and also predict the future case. For prediction, we use regression and time series models based on time series data. The increase in cases is due to the temperature change. Temperature changes with seasons. So COVID 19 is going up or down depending on different factors. This model has done an exploratory analysis of data and predicts the number of confirmed cases. In this project, we analyzed various methods for the prediction of COVID cases and compared the results so the best fit model with higher accuracy is used for prediction.[1]

So this project presents the prediction of the number of cases and factors that are suitable for an increase in COVID 19 cases so that it will help in providing future safety to the people all over the world and control the spread of the pandemic.

2. Introduction:

The 2019 Coronavirus pandemic, as known as COVID-19, is a current event which requires gathering, compilation, modeling, and forecasting to understand effects of this pandemic. At the beginning of the pandemic, there were many organizations that were gathering the appropriate COVID-19 data. These organizations, which included hospitals, state and national health agencies, universities, and other research groups, continue to collect the data from the onset of the pandemic and provide this data to other organizations on an ongoing basis. The COVID-19 pandemic creates a large problem to governments, people, and health care industries as there are many variables, both known and unknown, which are affecting the spread of the diseases.

As more and more data are available, creation and use of modeling tools can help identify key data and use it to forecast or affirm theories that can help health and government organizations plan to mitigate the risks of this pandemic. The project involves multiple variables, such as location, temperature, etc., which can be gathered over time. Therefore, we will have multivariable time series data for analysis. Additionally, as we receive more data from a diverse number of sources, we will cleanse, standardize, synchronize, and compile the data in a homogenous format to create a working model. Finally, the goal of this project is to model the data into providing real analysis on an on-going basis from an accurate model.

3. Background and Related Work:

It has become much more complicated and critical in the midst of the fallout from COVID-19, as cases are increasingly rising. It is really important to consider what the future looks like and what steps can be taken by all. In order to do so by predicting the possible effects of the COVID-19 pandemic in coming weeks, forecasting cases can help guide public health decision-making. And understanding the factors causing the rise in COVID-19 can help people understand the triggers in a better way. And they can avoid areas where the possibility of transmission occurs.

There is a lot of research going on in the field of COVID-19 prediction. In our project, we strive to achieve the forecast of cases and factors influencing the rise in cases in the coming future.

4. Methodology:

4.1 Dataset Description:

Datasets used and their selected features: This project utilizes four distinct data sets focused on COVID-19 confirmed cases, deaths, recovered cases, global temperatures.

covid_19_clean_complete: Province, Country, Lat , LONG, Confirmed, Deaths, Recovered

Time_series_covid_19_confirmed_US:Province,Country, No_of_Confirmed_cases_by_date

Time_series_covid_19_Deaths_US: Province,Country,No_of_Deaths_cases_by_date

Time_series_covid_19_Recovered: Province,Country,No_of_Recoevred_cases_by_date

GlobalTemprature: LandAverageTemperature, LandAverageTemperatureUncertainty

4.2 Data Cleaning:

We used multiple steps to clean our various data sets. Some minor initial changes included standardizing, capitalization and replacing spaces with underscores to make our data more compatible with various types of data mining tools. Much more cleaning work was done on the dataset – 'covid_19_clean_complete' data which is the largest dataset. We have segregated the temperature dataset into multiple datasets according to city, state, and country using excel VLOOKUP function, so that it will help us to develop better visualizations in future using matplotlib.

This dataset had missing values under various attributes. To handle missing values, we had faced challenges to recover the missing values. Hence we have treated missing attributes (latitude and longitude) in the dataset with the help of 'WHO Region' using the built-in R function, geopy(), which is important data towards our project goal. For other attributes, we

decided to ignore the rows with missing values as the dataset has 11,000+ rows which is huge data already. We uploaded the data into MySQL and used queries/functions to retrieve all the rows which are non-empty. As a part of cleaning, we dropped the columns which were not necessary and will be unused data. We used the drop() function in python to drop the columns from all the dataset – 'time_series_covid_19'.

4.3 Data Preparation:

1 Reading and understanding the data.

2: Preparing the data.

The below code is used generate the latitude and longitude for 'covid_19_clean_complete':

```
from geopy.geocoders import Nominatim

geolocator = Nominatim(user_agent="my_user_agent")

city ="Belgium"

country ="Europe"

loc = geolocator.geocode(city+','+ country)

print("latitude is :-" ,loc.latitude,"\nlongtitude is:-" ,loc.longitude)
```

The below code is used to drop the columns in our dataset – 'time_series_covid_19'. We have eliminated the columns 'iso2', 'iso3' ,'code3', and 'FIPS' which we are not planning to represent or use in our visualization. Here we have created a data frame and stored the dataset as data frame to drop the columns which are not needed:

```
import pandas as pd
import numpy as np
df = pd.read_csv('time_series_covid_19_confirmed_US.csv')
df.head()
to_drop = ['iso2',
...        'iso3',
...        'code3',
...        'FIPS']
df.drop(to_drop, inplace=True, axis=1)
```

```
In [15]: df.head()
```

Out[15]:

	UID	Admin2	Province_State	Country_Region	Lat	Long_	Combined_Key	1/22/20	1/23/20	1/24/20	...	9/14/20	9/15/20	9/16/20	9/17/20
0	84001001	Autauga	Alabama	US	32.539527	-86.644082	Autauga, Alabama, US	0	0	0	...	1447	1463	1619	1624
1	84001003	Baldwin	Alabama	US	30.727750	-87.722071	Baldwin, Alabama, US	0	0	0	...	4800	4812	5003	5021
2	84001005	Barbour	Alabama	US	31.868263	-85.387129	Barbour, Alabama, US	0	0	0	...	626	629	809	809
3	84001007	Bibb	Alabama	US	32.996421	-87.125115	Bibb, Alabama, US	0	0	0	...	581	580	612	617
4	84001009	Blount	Alabama	US	33.982109	-86.567906	Blount, Alabama, US	0	0	0	...	1128	1139	1487	1504

5 rows × 253 columns

3. Creating and consolidating table, which gives country wise total defined cases and showing top 10 countries.

```
In [9]: temp_f = full_latest_grouped.sort_values(by='Confirmed', ascending=False)
        temp_f = temp_f.reset_index(drop=True)
        temp_f.style.background_gradient(cmap='BuPu')
```

Out[9]:

	Country/Region	Confirmed	Deaths	Recovered	Active
0	US	4290259	148011	1325804	2816444
1	Brazil	2442375	87618	1846641	508116
2	India	1480073	33408	951166	495499
3	Russia	816680	13334	602249	201097
4	South Africa	452529	7067	274925	170537
5	Mexico	395489	44022	303810	47657
6	Peru	389717	18418	272547	98752
7	Chile	347923	9187	319954	18782
8	United Kingdom	301708	45844	1437	254427
9	Iran	293606	15912	255144	22550
10	Pakistan	274289	5842	241026	27421

Fig1: This plot shows the heatmap of the confirmed, recovered, deaths and active cases for different countries. The highest cases in all categories are observed for the US country. Hence our major focus will be on the US and we will predict cases for this country along the environmental factors affecting it.

4.4 Model Planning:

We have considered 4 autoregression methods to predict future values of confirmed COVID-19 cases

1. SVM (Support Vector Method Regression Method)
2. Polynomial Regression Method
3. Linear Regression Method
4. Bayesian Ridge Regression Method

Support Vector Machine: It is known for the discriminative power of classification. This training algorithm develops a model by finding a hyperplane and classifies data correctly by maximizing the data between data clusters.

Bayesian Ridge Regression: We formulate linear regression from the Bayesian perspective using probability distributions rather than point estimates. The answer, y, is not calculated as a single value, but is presumed to be derived from a distribution of probability. The goal of the Bayesian Linear Regression is not to find the single "best value" of the parameters of the model, but rather to determine the lateral distribution of the parameters of the model.

Linear Regression: This deals with a linear approach to modelling the relationship between a scalar response and one or more explanatory variables.

Polynomial Regression: This type of regression analysis is performed by considering the independent variable x and dependent variable y modelled as 4th degree polynomial in x.

A special case of linear regression where we fit a polynomial equation on the data with a curvilinear relationship between the target variable and the independent variables is polynomial regression.

The value of the target variable varies in a non-uniform fashion with respect to the indicator in a curvilinear relationship (s).

In Linear Regression, we have the following equation for a single predictor:

$$Y = \theta_0 + \theta_1 x$$

Where,
Y is the target,
The predictor is x ,
$\theta 0$ is the bias,
and $\theta 1$ is the weight in the regression equation

It is possible to use this linear equation to describe a linear relationship. But we have a polynomial equation of degree n in polynomial regression, represented as Equation of Polynomial Regression

$$Y = \theta_0+\theta_1 x+\theta_2 x^2+\theta_3 x^3+...+\theta_n x^n$$

Here:
$\theta 0$ is the bias,
$\theta 1$, $\theta 2$, …, θn are the weights in the equation of the polynomial regression, and n is the degree of the polynomial. The number of terms of the higher order increases with the increase in value of n, and hence the equation becomes more complicated.

5. Implementation:

Imports required for the implementation:

```
#SVM, Polynomial regression, bayesian regression and linear regression Methods

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import pandas as pd
import random
import math
import time
from sklearn.linear_model import LinearRegression, BayesianRidge
from sklearn.model_selection import RandomizedSearchCV, train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error
import datetime
import operator
plt.style.use('fivethirtyeight')
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

Data Transformation:

```
#creating array to store range of days
days_since_11_30 = np.array([i for i in range(len(dates))]).reshape(-1, 1)
world_cases = np.array(world_cases).reshape(-1, 1)

#predicting for 10 days
days_in_future = 10
future_forcast = np.array([i for i in range(len(dates)+days_in_future)]).reshape(-1, 1)
adjusted_dates = future_forcast[:-10]

start = '11/30/2020'
start_date = datetime.datetime.strptime(start, '%m/%d/%Y')
future_forcast_dates = []
for i in range(len(future_forcast)):
    future_forcast_dates.append((start_date + datetime.timedelta(days=i)).strftime('%m/%d/%Y'))

# Fit the data to the model
X_train_confirmed, X_test_confirmed, y_train_confirmed, y_test_confirmed = train_test_split(days_since_11_30[50:],
                                                                                            world_cases[50:], test_size=0.05, shuffle=False)
```

Methods:

This section is about the implementation of the 4 autoregression methods which we have considered to model and test our data[4]. We have calculated the parameters MAE, MSE and R2 to compare these methods and which will give us the best results. We have plotted the values for MAE and MSE for better visualization of the values from which we can deduce the method having minimum error value:

1. **SVM (Support Vector Method Regression Method):**

```
# svm_method
svm_confirmed = SVR(shrinking=True, kernel='poly',gamma=0.01, epsilon=1,degree=3, C=0.1)
svm_confirmed.fit(X_train_confirmed, y_train_confirmed)
svm_pred = svm_confirmed.predict(future_forcast)
# check against testing data
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))
print('MSE:',mean_squared_error(svm_test_pred, y_test_confirmed))
```

```
MAE: 2562802.516721352
MSE: 6688361097681.3955
```


```
from sklearn.metrics import r2_score
r_squared = r2_score(y_test_confirmed, svm_test_pred)
print(r_squared)
```

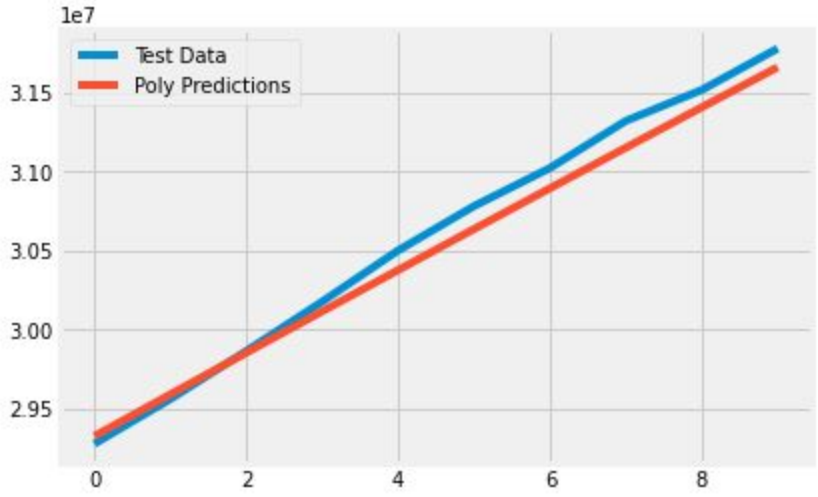
```
-9.232084179811778
```

2. Polynomial Regression Method:

```
# transform our data for polynomial regression
poly = PolynomialFeatures(degree=4)
poly_X_train_confirmed = poly.fit_transform(X_train_confirmed)
poly_X_test_confirmed = poly.fit_transform(X_test_confirmed)
poly_future_forcast = poly.fit_transform(future_forcast)
# 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_forcast)

plt.plot(y_test_confirmed)
plt.plot(test_linear_pred)
plt.legend(['Test Data', 'Poly Predictions'])
print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
print('MSE:',mean_squared_error(test_linear_pred, y_test_confirmed))
```

```
MAE: 94657.12531040757
MSE: 11516541771.083717
```




```
from sklearn.metrics import r2_score
r_squared_p = r2_score(y_test_confirmed, test_linear_pred)
print(r_squared_p)
```

```
0.982381599447001
```

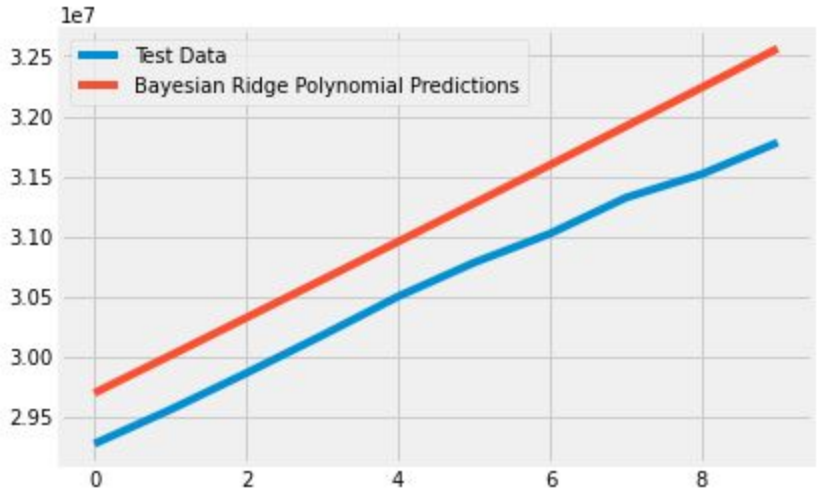
3. Bayesian Ridge Regression Method:

```
bayesian_search.best_params_
```

```
{'tol': 1e-06,
 'normalize': True,
 'lambda_2': 0.001,
 'lambda_1': 1e-07,
 'alpha_2': 0.0001,
 'alpha_1': 0.0001}
```

```
bayesian_confirmed = bayesian_search.best_estimator_
test_bayesian_pred = bayesian_confirmed.predict(bayesian_poly_X_test_confirmed)
bayesian_pred = bayesian_confirmed.predict(bayesian_poly_future_forcast)
plt.plot(y_test_confirmed)
plt.plot(test_bayesian_pred)
plt.legend(['Test Data', 'Bayesian Ridge Polynomial Predictions'])
print('MAE:', mean_absolute_error(test_bayesian_pred, y_test_confirmed))
print('MSE:',mean_squared_error(test_bayesian_pred, y_test_confirmed))
```

```
MAE: 541086.6332854275
MSE: 306622478448.3949
```




```
from sklearn.metrics import r2_score
r_squared_b = r2_score(y_test_confirmed, test_bayesian_pred)
print(r_squared_b)
```

```
0.5309184984653296
```

4. Linear Regression Method:

```
#Linear regression
lreg_confirmed = LinearRegression()
lreg_confirmed.fit(X_train_confirmed, y_train_confirmed)
lreg_pred = lreg_confirmed.predict(future_forcast)
```

```
lreg_test_pred = lreg_confirmed.predict(X_test_confirmed)
plt.plot(y_test_confirmed)
plt.plot(lreg_test_pred)
plt.legend(['Test Data', 'Multivariate LR Predictions'])
print('MAE:', mean_absolute_error(lreg_test_pred, y_test_confirmed))
print('MSE:',mean_squared_error(lreg_test_pred, y_test_confirmed))
```

```
MAE: 5096733.912250668
MSE: 26110494871468.33
```


```
from sklearn.metrics import r2_score
r_squared = r2_score(y_test_confirmed, lreg_test_pred)
print(r_squared)
```

```
-38.94473049519167
```

6. Experimental Results:

Comparing the parameters MAE, MSE and R2 for all the four methods, we observe that the polynomial regression is best fit for our dataset. The reason is that the mean absolute error is less and coefficient value is higher than other methods[5].

MODELS	MAE	MSE	R^2
Support Vector Machine	2562802.52	6.68836E+12	-9.23
Polynomial regression	94657.13	11516541771.08	0.98
Bayesian Ridge Regression	541086.58	3E+11	0.53
Linear Regression	5096733.91	2.61105E+13	-38.94

Table1: This represents the comparison of the various parameters of methods we used to train the model on our dataset.

The temperature plays a major role in transmission of the covid which can be observed with the below choropleth globe visualization[3]:

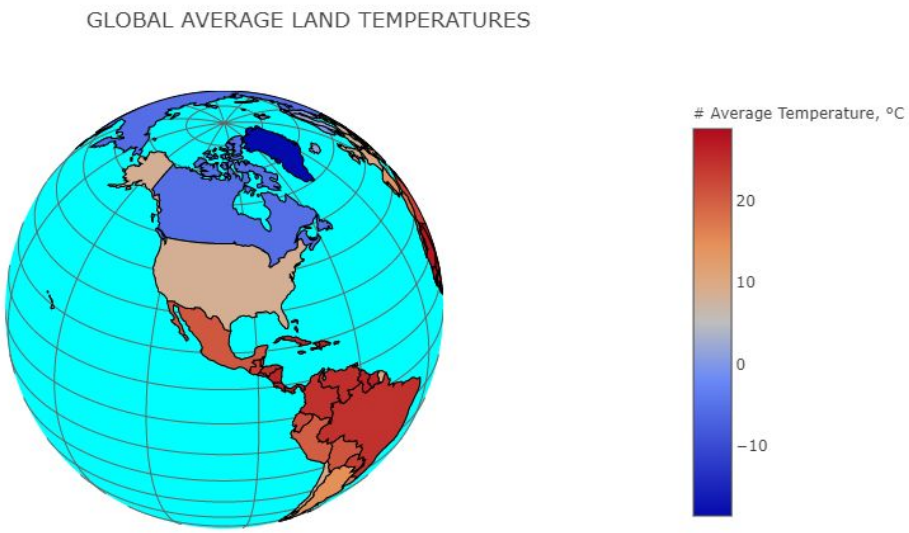


Figure2: The visualization represents the land average temperature for different countries for which we will plot the number of cases to deduce the temperature variations affecting on covid cases.

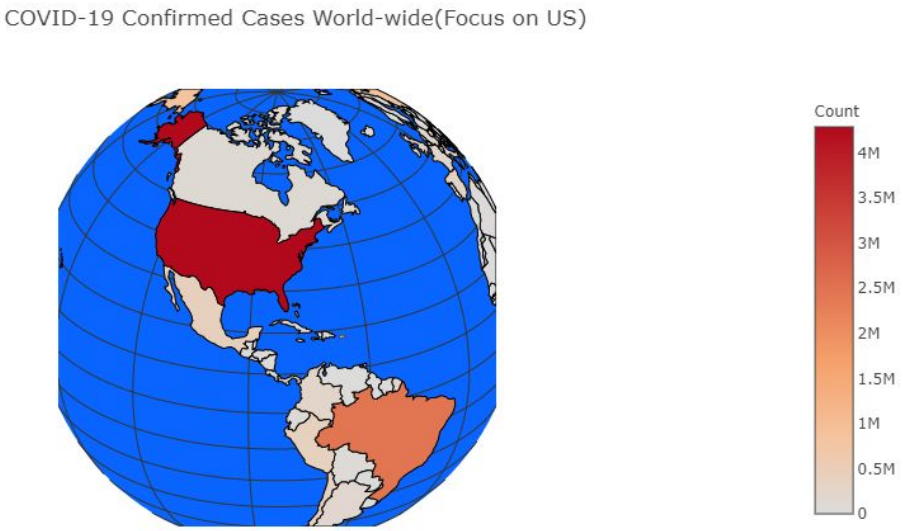


Figure3: This visualization represents the number of confirmed cases worldwide (focussed on US).

Forecast of number of cases using autoregression(polynomial regression) model:

```
prediction = model_fit.predict(start=len(df['Cases'][:-1]), end=len(df['Cases'][:-1])+10)
prediction.apply(np.int64)
```

```
310    9359034
311    9408387
312    9457872
313    9507535
314    9557404
315    9607495
316    9657771
317    9708206
318    9758770
319    9809465
320    9860315
dtype: int64
```

The table consists of the number of cases for 5 consecutive days. The visualization below compares the recovered and confirmed cases till july 2020 and it is a range slider graph whose functionality can be viewed in python.

Date	12/8/2020	12/9/2020	12/10/2020	12/11/2020	12/12/2020

No. of Confirmed Cases	9657771	9708206	9758770	9809465	9860315

Table2: Number of confirmed cases from 8th december 2020 to 12th december 2020

Figure4: The confirmed and recovered cases are plotted against the date in this graph. This is a range slider type of graph which compares the number of confirmed and recovered cases over time.

7. Future Scope:

The future scope of this project is to compose and implement error-correcting methods to increase predictability and have accurate utilization of models. While we analyze the current and new data with different models, such as Bayesian Ridge Regression model, Linear Regression model, Polynomial Regression model, etc., we can identify and gather a new set of errors within the models. If the error terms are correlated between each iteration of the model, then we can estimate this error and remove it to create a model that is more accurate.

This future state of the model with built in error-correcting methods will be an on-going process as we receive new data, and the model is updated. Our current best model uses MAE, MSE, and R^2 approach to identify the best model, but an updated model with error-correcting methods would boost the goodness-of-fit while reducing the errors in the model. Therefore, our

future scope is to compose and implement error-correcting methods to increase predictability and accuracy of the models.

8. Conclusion:

Information and communication technology help in the decision-making process based on past data with data analytics and data mining perspectives The size of data available is huge and gathering information and getting predictions out of the cumulated data is a challenging task. So with the existing COVID dataset, various methods are carried on that data so that we can find the most accurate method by using different parameters MSE, MAE, and R2.

In various methods Polynomial Regression is best suited for the prediction of cases because its mean absolute error(MAE) is less than other methods and R square(coefficient of Determination) is close to 1. So choose Polynomial Regression as the best fit model for the prediction of cases.

The factors temperature and humidity are identified to affect the transmission. The COVID 19 is spreading more in the temperature zone of the planet compared to the tropical zones. So to control the pandemic situation people should avoid traveling to places where the temperature is suitable for COVID 19.[2] The complete progression of the covid cases is summarized in the below visualization whose functionality can be viewed in python:

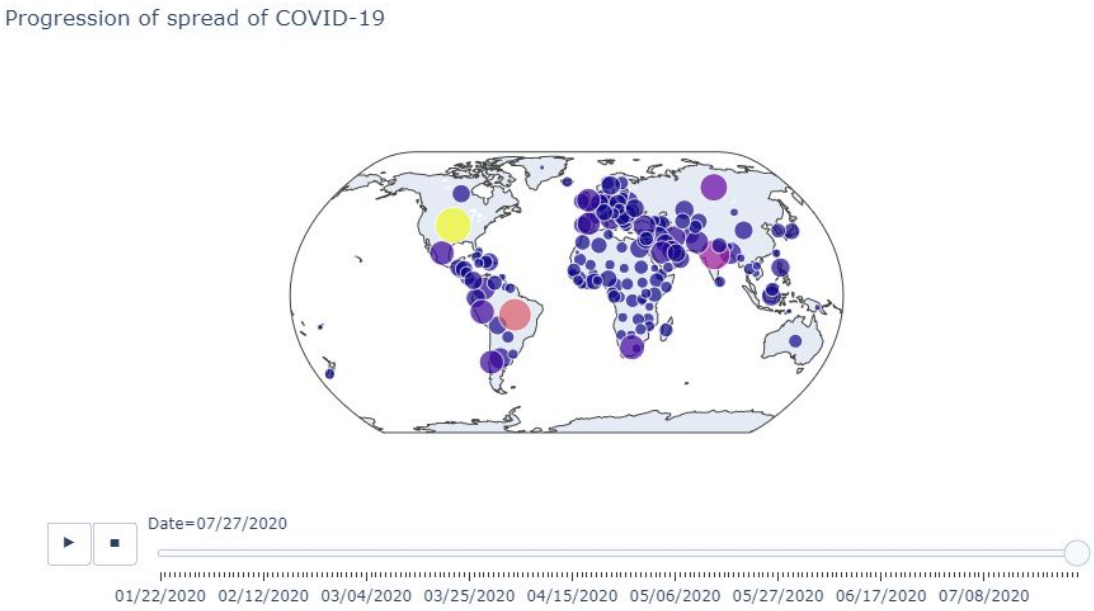


Figure5: Summary-This is a progression of COVID 19 cases spread graph over time. The radius of the circle represents the number of cases and the colors indicate intensity of cases.

9. References:

1. S. Singh, P. Raj, R. Kumar and R. Chaujar, "Prediction and forecast for COVID-19 Outbreak in India based on Enhanced Epidemiological Models," *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, 2020, pp. 93-97, doi: 10.1109/ICIRCA48905.2020.9183126. Retrieved from

   https://ieeexplore.ieee.org/document/9183126

2. https://coronavirus.jhu.edu/data
3. https://plotly.com/python/choropleth-maps/
4. https://towardsdatascience.com/introduction-to-bayesian-linear-regression-e66e60791ea7
5. https://www.analyticsvidhya.com/blog/2020/03/polynomial-regression-python/
6. F. Rustam et al., "COVID-19 Future Forecasting Using Supervised Machine Learning Models," in IEEE Access, vol. 8, pp. 101489-101499, 2020, doi: 10.1109/ACCESS.2020.2997311.

   Retrieved from:
   https://ieeexplore-ieee-org.proxy-bc.researchport.umd.edu/document/9099302

7. E. Gambhir, R. Jain, A. Gupta and U. Tomer, "Regression Analysis of COVID-19 using Machine Learning Algorithms," 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2020, pp. 65-71, doi: 10.1109/ICOSEC49089.2020.9215356.

   Retrieved from:
   https://ieeexplore-ieee-org.proxy-bc.researchport.umd.edu/document/9215356