EAST WEST UNIVERSITY

Advanced Database Systems

Project Report

1. Introduction

At present covid-19 is the most terror in whole world. It first emerged in Chinese city of Wuhan and quickly spread around the world [1]. Every country has been victim to the covid-19 and suffered financially. Almost 162 countries throughout the world are taking precautions to prevent touch and preserve social distance [2]. Failed to take proper precautionary measure can result in rapid spread of the disease and deterioration of the condition. With the development of computer technology, now it's very easy to predict where the situation will go and can take necessary steps taken in advance to avoid catastrophic damage. Linear regression is a widely used algorithm that is used to estimate this type of upcoming situation. In this paper a few regression model such as simple, multiple and polynomial regression has been built to predict death case using Bangladesh covid-19 data set. This dataset contains the following type of data " Date, lab test, confirm case, first and second vaccine and death case".

2. Data Pre-processing

We have three data set to build the model which are covid-19_dataset, covid_first_dose and covid_second_dose. If we observe the dataset for covid-19, it can be seen that the records of the first dataset start from 4th April, 2020. But the 1st and 2nd vaccination records started from 27th January, 2021 and 8th April, 2021 respectively. If we join them with any other join method besides inner join it will generate null values. These type of dataset is not suitable for machine learning applications. To prevent null values, we have used inner join for these datasets shown in figure 1, in this way we make the dataset pliable for the model.

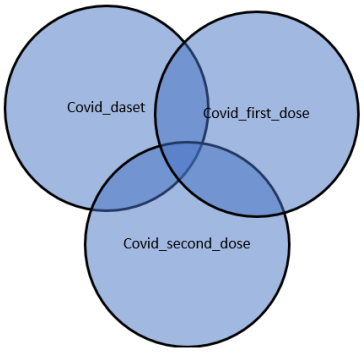


Figure1: Inner join.

We have removed all the null date containing rows. The final table contains the following columns 'day',lab_test', 'confirmed_case', 'first_dose', second_dose', and 'death_case'.

3. Dataset Characteristics and Exploratory Data Analysis (EDA)

We have merged all the given three table set and created a new data set called final_data_table. There are total 188 rows and 6 columns in our final table. We used five of the six columns as independent veriable and the last one dependent veriable. The 'X' datasets are 'day',lab_test', 'confirmed_case', 'first_dose', second_dose' and for 'y' the datasets is 'death_case'. A heat map is shown to see the relation between all the variables shown in figure 2.

It is a two dimensional data visualization tool that displays the magnitude of a phenomenon as colour. The color fluctuation shows hue and intensity, giving us a clear visual indication about how the occurrence is clustered or evolves over time.

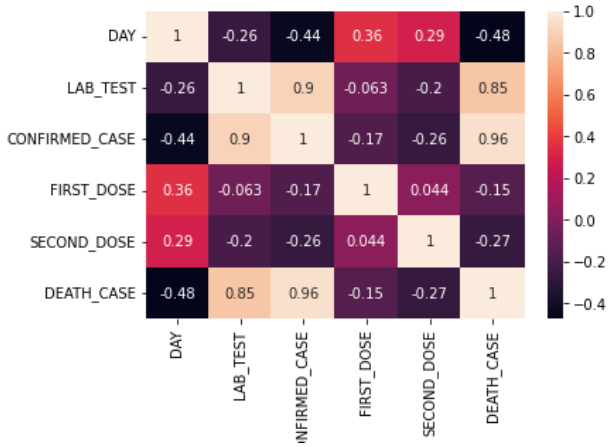


Figure. 2 Heat map.

From the heat map we have found that every data column has different relation some have negative relation and some have positive relation. The_first_dose and second_dose column have negative relation with the death_case data. The confirm_case and lab_test have shown strong positive correlation with the death_case.
So, we have taken the positively related data to build linear regression and polynomial regression model to generate result based on the positive dataset, and we also used all the positive strong correlation and the negative correlation data such as first_dose, second_dose in a multi regression model with the death_case data to see what kind of difference does it show.

For the positively correlation data such as death case and confirm case:

Figure 3. Confirm case vs. death case

Here we can see the close relation between two variables related to each other. It's a typical way of describing simple relationships without stating a cause and effect relationship. We found correlation between lab test and death case shows 0.85. The correlation between confirm case and death case is 0.96. We found that the confirm case and death case has the most positively strong relationship in the dataset.

From the diagram the can clearly see that the relation the x-axis and y-axis are very high when the confirmed case was very high and responding to these high cases the death was also very high. As the confirm cases of covid-19 affected declined bellow 900, we can see that the death case also when down below 100. It seems end of the year 2021 showed less death case.

4. Machine Learning Models

Regression is a method for comparing two theories. For starters, regression analyses are commonly employed for forecasting and prediction, and their use overlaps heavily with machine learning. Second, regression analysis can be used to discover causal relationships between independent and dependent variables in particular circumstances. Importantly, regressions alone reveal only relationships between a dependent variable and a set of fixed variables in a dataset [3].

Regression Models

There is a plethora of models in the machine learning sector. Among the wide varieties of models, regression is one of the most prominent and widely used models. According to regression models, the dependent variables are predicted by the independent variables. Due to the range of independent variable values 'x,' regression analysis calculates the value of the dependent 'y' variable. In this project, we compare different regression methods to see which one best matches the predictive model. There are different types of regression for suiting different needs:

- **Simple Linear Regression**

  Linear regression model is a supervised machine learning model that determines the best fit linear line between the independent and dependent variables. It determines the dependent and independent variables linear connection.

  A case model with a single independent variable is called Simple Linear Regression. The dependence of the variable is defined using simple linear regression. $y=\beta0+\beta1x+\varepsilon$ [3].The

influence of independent factors is distinguished from the interaction of dependent variables in simple regression.

- **Multiple Linear Regression**
  Multiple regression is a machine learning approach that uses two or more predictors to predict a dependent variable. In three issue categories, multi regression has extensive real world applications, evaluating correlation between variables, producing numerical predictions and time series forecasting.
  This model uses the same principle as Simple Linear Regression, but there's a slight variation in the method. The dependent variable here accounts for multiple independent variables in the dataset.
- **Polynomial Regression**
  Polynomial regression is a sort of regression analysis in which the relationship between independent and dependent variables is modelled using nth degree polynomial modelling. Polynomial regression is one type of MLR in which the data's polynomial equation is combined with the dependent and independent variables' curvilinear interactions $y=\beta0+\beta1x+\beta2x2+\cdots+\beta hxh+\varepsilon$ [3].

5. Description of Models and Associated Parameters

A few models of regression have been used for experimenting with the merged dataset for covid 19 in Bangladesh, to figure out the most accurate among all these algorithms. To begin the process, there are a couple of common steps used in all the models.
First of all, all the necessary libraries required for the model to function were imported such as- pandas, numpy, seaborn, mpl_toolkits, matplotlib.pyplot. Then, the dataset was loaded. Plots are shown using sns.pairplot and sns.distplot. Heatmap is represented using sns.heatmap. Then the x and y variables are defined as needed for the specific task. After the defining, the dataset is then split for training and testing by using train_test_split from sklearn.model_selection with 20% test size and random state parameters.
Specific regression is then applied for training the model by using Linear Regression and Polynomial Features from sklearn.linear_model. To get the predicted values, mul.predict (X_mul_test), regressor.predict(X_lp_test),lin2.predict(poly.fit_transform(X_lp_test)) and first_second.predict(X_first_second_test) are used.
All the predicted variables are shown side by side with the actual death value by fitting into a data frame.
A scatter plot is shown for the test and prediction values for the dependent variable. Then the actual and predicted data are fitted into a data frame from the test and predicted values of the dependent variable respectively.
Lastly, to check the viability of the model, three types of errors are checked. These are Mean Absolute Error, Mean Squared Error and Root Mean Squared Error metrics imported from sklearn libraries.

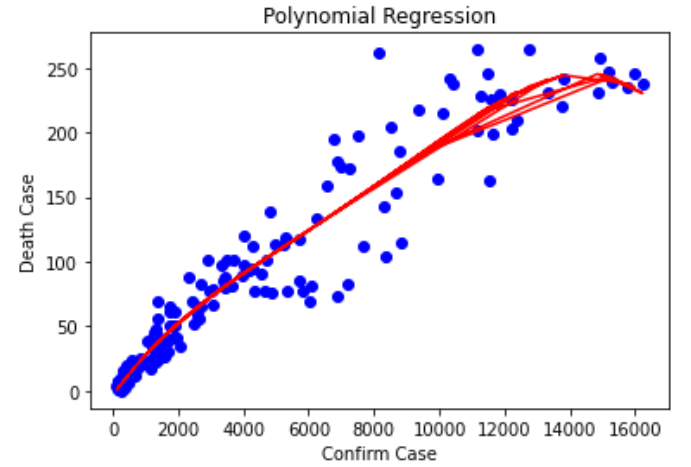
- **Multiple Linear Regression (All columns):** Here, dependent variable 'DEATH_CASE' is predicted with the independent variables 'LAB_TEST', 'CONFIRMED_CASE',

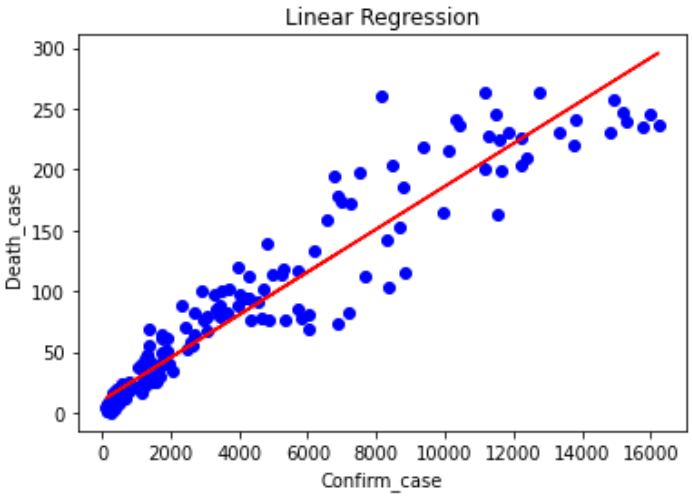
'FIRST_DOSE', 'SECOND_DOSE'. This model shows the accuracy of the predicted 'DEATH_CASE' in relation to all the other independent variables.

- **Multiple Linear Regression (FIRST_DOSE and SECOND_DOSE):** Here, dependent variable 'DEATH_CASE' is predicted with the independent variables 'FIRST_DOSE', 'SECOND_DOSE'. This model shows the accuracy of the predicted 'DEATH_CASE' in relation to those two independent variables.
- **Simple Linear Regression (CONFIRMED_CASE):** Here, dependent variable 'DEATH_CASE' is predicted only with the independent variable 'CONFIRMED_CASE'. This model shows the accuracy of the predicted 'DEATH_CASE' in relation to only one independent variable.
- **Polynomial Regression (CONFIRMED_CASE):** Here, dependent variable 'DEATH_CASE' is predicted only with the independent variable 'CONFIRMED_CASE'. This model shows the accuracy of the predicted 'DEATH_CASE' in relation to only one independent variable. In PolynomialFeatures, the degree parameter is passed as 4.

6. Performance Evaluation

It is of no surprise that different models will boast different outcomes for any dataset in any given context. In the context of our dataset for covid19, where we decide to find the death case prediction using various regression models in relation to all the other independent variables, these models have varying degrees of errors in the outcome. These errors help to compare them to figure out which one is most suitable for the task at hand. There are clearly varying degrees of differences among all the models once we observe the following graphs and error values among them.
The scatter plots of the Polynomial Regression (CONFIRMED_CASE) and Simple Linear Regression (CONFIRMED_CASE) represents the visualization of the dependent and independent variables.





The errors for all the models used are shown and compared in the table below:

Table 1: Error Comparison

Models	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error
Multiple Linear Regression (All columns)	19.391134659649765	793.9893100893663	28.17781592120593
Multiple Linear Regression (FIRST_DOSE and SECOND_DOSE)	64.08914658544859	6474.721427902278	80.46565371574556
Simple Linear Regression (CONFIRMED_CASE)	19.108431463234837	743.9103982098969	27.2747208640143
Polynomial Regression (CONFIRMED_CASE)	14.652238781359621	619.4778098295654	24.889311156188423

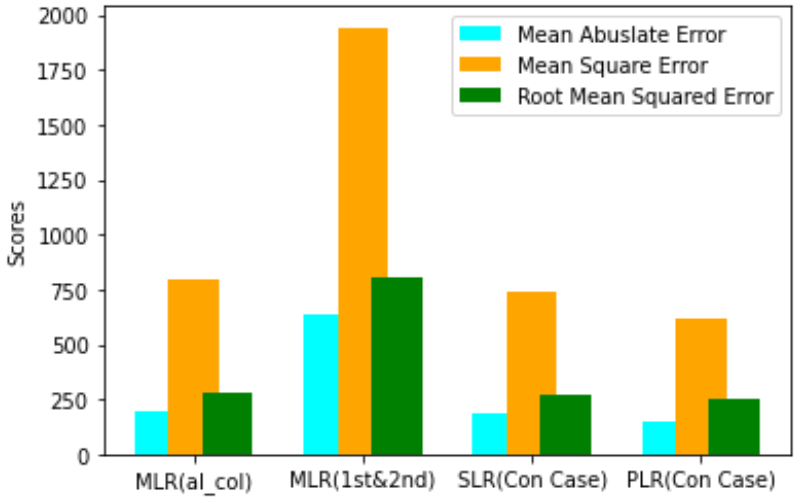


figure 6. Error Comparison

7. Discussion

From the graphs and error comparison, we can conclude that 'Polynomial Regression (CONFIRMED_CASE)' performs the best for our task at hand. We can confidently say this because this model has the least error in all three error checking methods among all the other regression models tested. There are a few reasons why this method worked better than the other ones tested.

One of the reasons stems from the level of correlation between dependent and independent variables already present in the database. We can see that 'CONFIRMED_CASE' and 'DEATH_CASE' has the most sensible amount of correlation between each other. Realistically speaking, the more the confirmed cases, the more it is likely that the death case rises. For this logic, even among the linear regression methods, the 'Simple Linear Regression (CONFIRMED_CASE)' model performs better. We can notice this correlation by observing the heat map of the dataset where we can see that 'CONFIRMED_CASE' and 'DEATH_CASE' has the closest value to 1 having 0.96. Another reason for the polynomial one to perform better is the nature of the technique. Linear regression normally goes on a straight line as its name suggests. So, it might fail to capture some of the data points while predicting, as some of the data points might reside further away from the line. But polynomial one avoids this issue by going through some of the data points by following the density among the data points, resulting in the curve of the line.

8. Conclusion

After finishing the project, we can safely conclude that we have been able to predict the necessary outcome for a major pandemic covid-19. Our goal was to predict the casualties as a result of this global crisis. This could help healthcare sectors in preparing for future outbreaks of similar scale and cause by using these predictive machine learning models. They can correlate different events with each other to mitigate the adverse effects on the lives of millions in the near future just by

taking the right course of action. For example, we can see that death cases directly correlates to the number of confirmed cases and the number of confirmed cases correlated directly with the lab tests. So, to increase the efficiency of the precaution measures, people can take steps in the necessary ways for preventing such disasters in the future.
There were a few challenges throughout the process of completing the project. One of the most prominent one was the pre-processing part in which the fragmented datasets were merged into one dataset for the ease of use and applying the machine learning models.
There were a few missed opportunities in our project for the lack of time and the limited knowledge we have. First of all, we could find out various other predictions from this dataset other than just the casualties caused by covid19. For example, we could use the model to train it with the existing variables for predicting the future outcomes for the same variables such as the future of confirmed cases, lab tests or the number of people that might take vaccines, etc.
Since the pandemic is fairly at an early stage, it is tough to gather enough resources to say anything of assurance for the coming days. Any pandemic of such a large scale requires further research to work upon a better future for the masses. This is of no exception and thus we hope to contribute what little we can for the majority of this field.

References

[1] M. A. Shereen, S. Khan, A. Kazmi, N. Bashir, and R. Siddique, "COVID-19 infection: Origin, transmission, and characteristics of human coronaviruses," *J. Adv. Res.*, vol. 24, pp. 91–98, 2020, doi: 10.1016/j.jare.2020.03.005.

[2] V. Chaurasia and S. Pal, "COVID-19 Pandemic: ARIMA and Regression Model-Based Worldwide Death Cases Predictions," *SN Comput. Sci.*, vol. 1, no. 5, pp. 1–12, 2020, doi: 10.1007/s42979-020-00298-6.

[3] D. Maulud and A. M. Abdulazeez, "A Review on Linear Regression Comprehensive in Machine Learning," *J. Appl. Sci. Technol. Trends*, vol. 1, no. 4, pp. 140–147, 2020, doi: 10.38094/jastt1457.

Appendix

```
import pandas as pd
import numpy as np
import seaborn as sns
import mpl_toolkits
%matplotlib inline
import matplotlib.pyplot as plt

covid_19 = pd.read_csv('/content/drive/MyDrive/464final/FINAL_DATA_TABLE.csv'
)

covid_19.head()

sns.pairplot(covid_19)

sns.heatmap(covid_19.corr(), annot=True)
```

```
#Multiple Regression using all column
X_mul = covid_19[['LAB_TEST', 'CONFIRMED_CASE', 'FIRST_DOSE', 'SECOND_DOSE']]

y_mul = covid_19['DEATH_CASE']

#Multiple linear regression using First and Second dose with DeathCase column
X_first_second=covid_19[['FIRST_DOSE', 'SECOND_DOSE']]
y_first_second=covid_19['DEATH_CASE']

#linear and polynomial regression using ConfirmCase with DeathCase column
X_lp = covid_19.iloc[:, 2:3].values
y_lp = covid_19.iloc[:, 5].values

#Multiple regression with all colums train test
from sklearn.model_selection import train_test_split
X_mul_train, X_mul_test, y_mul_train, y_mul_test = train_test_split(X_mul, y_
mul, test_size=0.2, random_state=0)

#linear and polynomial regression train test
from sklearn.model_selection import train_test_split
X_lp_train, X_lp_test, y_lp_train, y_lp_test = train_test_split(X_lp, y_lp, t
est_size=0.2, random_state=0)

#mul  regression train test with 1st and 2nd vaccine
from sklearn.model_selection import train_test_split
X_first_second_train, X_first_second_test, y_first_second_train, y_first_seco
nd_test = train_test_split(X_first_second, y_first_second, test_size=0.2, ran
dom_state=0)

# Fitting multiple linear Regression with all coloms to the covid_19
from sklearn.linear_model import LinearRegression
mul = LinearRegression()
mul.fit(X_mul_train, y_mul_train)

# Fitting Linear Regression to the covid_19
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_lp_train, y_lp_train)

# Fitting Polynomial Regression to the covid_19
from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree = 4)
X_poly = poly.fit_transform(X_lp)
poly.fit(X_poly, y_lp)
lin2 = LinearRegression()
lin2.fit(X_poly, y_lp)

# Fitting 1st and second vac dose Regression to the covid_19
from sklearn.linear_model import LinearRegression
first_second = LinearRegression()
first_second.fit(X_first_second_train, y_first_second_train)

# Visualising the Linear Regression results
plt.scatter(X_lp, y_lp, color = 'blue')

plt.plot(X_lp, regressor.predict(X_lp), color = 'red')
```

```
plt.title('Linear Regression')
plt.xlabel('Confirm_case')
plt.ylabel('Death_case')

# Visualising the Polynomial Regression results
plt.scatter(X_lp, y_lp, color = 'blue')

plt.plot(X_lp, lin2.predict(poly.fit_transform(X_lp)), color = 'red')
plt.title('Polynomial Regression')
plt.xlabel('Confirm Case')
plt.ylabel('Death Case')

plt.show()

y_mul_pred = mul.predict(X_mul_test)

y_lin_pred = regressor.predict(X_lp_test)

y_poly_pred = lin2.predict(poly.fit_transform(X_lp_test))

y_first_second_pred = first_second.predict(X_first_second_test)

df = pd.DataFrame({'Actual Death': y_test, 'MLR (all col)': y_mul_pred,'MLR (
1st & 2nd Dose)':y_first_second_pred,'SLR (Confirm Case)':y_lin_pred,'PR (Con
firm Case) ':y_poly_pred})
df

#multiple linear regression
from sklearn import metrics
mul_mean_abs_er= metrics.mean_absolute_error(y_test, y_mul_pred)
mul_mean_squ_er= metrics.mean_squared_error(y_test, y_mul_pred)
mul_Rot_mean_squ_er= np.sqrt(metrics.mean_squared_error(y_test, y_mul_pred))


print('Mean Absolute Error with Multiple linear Regression:', mul_mean_abs_er
)
print('Mean Squared Error with Multiple linear Regression:', mul_mean_squ_er)
print('Root Mean Squared Error with Multiple linear Regression:',mul_Rot_mean
_squ_er)

#first and second dose
from sklearn import metrics
first_second_mean_abs_er=  metrics.mean_absolute_error(y_first_second_test, y
_first_second_pred)
first_second_mean_squ_er= metrics.mean_squared_error(y_first_second_test, y_f
irst_second_pred)
first_second_Rot_mean_squ_er=  np.sqrt(metrics.mean_squared_error(y_first_sec
ond_test, y_first_second_pred))


print('First second Mean Absolute Error with Multiple linear Regression:', fi
rst_second_mean_abs_er)
print('First second Mean Squared Error with Multiple linear Regression:', fir
st_second_mean_squ_er)
print('First second Root Mean Squared Error with Multiple linear Regression:'
,first_second_Rot_mean_squ_er)
```

```
#simple linear Rigrassion using confirm case
from sklearn import metrics
linear_mean_abs_er=  metrics.mean_absolute_error(y_test, y_lin_pred)
linear_mean_squ_er= metrics.mean_squared_error(y_test, y_lin_pred)
linear_Rot_mean_squ_er=  np.sqrt(metrics.mean_squared_error(y_test, y_lin_pre
d))

print(' Mean Absolute Error with linear Rigrassion:', linear_mean_abs_er)
print(' Mean Squared Error with linear Rigrassion:', linear_mean_squ_er)
print(' Root Mean Squared Error with linear Rigrassion:',linear_Rot_mean_squ_
er)

#Polynomial Regression uisng conform case
from sklearn import metrics
pol_mean_abs_er=  metrics.mean_absolute_error(y_test, y_poly_pred)
pol_mean_squ_er= metrics.mean_squared_error(y_test, y_poly_pred)
pol_Rot_mean_squ_er= np.sqrt(metrics.mean_squared_error(y_test, y_poly_pred))

print('poly Mean Absolute Error with Polynomial Regression:', pol_mean_abs_er
)
print('poly Mean Squared Error with Polynomial Regression:', pol_mean_squ_er)
print('poly Root Mean Squared Error with Polynomial Regression:',pol_Rot_mean
_squ_er)

x = np.arange(4)
y1 = [mul_mean_abs_er*10, first_second_mean_abs_er*10, linear_mean_abs_er*10,
pol_mean_abs_er*10]
y2 = [mul_mean_squ_er, first_second_mean_squ_er*.3, linear_mean_squ_er,pol_me
an_squ_er]
y3 = [mul_Rot_mean_squ_er*10, first_second_Rot_mean_squ_er*10, linear_Rot_mea
n_squ_er*10,pol_Rot_mean_squ_er*10]
width = 0.3

# plot data in grouped manner of bar type
plt.bar(x-0.2, y1, width, color='cyan')
plt.bar(x, y2, width, color='orange')
plt.bar(x+0.2, y3, width, color='green')
plt.xticks(x, ['MLR(al_col)','MLR(1st&2nd)','SLR(Con Case)','PLR(Con Case)'])

plt.ylabel("Scores")
plt.legend(["Mean Abuslate Error", "Mean Square Error", "Root Mean Squared Er
ror"])
plt.show()
```

.