Assessment Report

on

"COVID-19 Case Prediction"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

Intro To AI

By

Varun Prakash Srivastava (55)

Tushar Bhardwaj (45)

Shivam Pandey (14)

Vansh Karnwal (54)

Shivansh Gupta (16)

Under the supervision of

Mr. Abhishek Shukla

KIET Group of Institutions, Ghaziabad

COVID-19 Case Prediction

🧩 1. Problem Statement

The objective of this project is to develop a time-series prediction model to forecast future COVID-19 cases based on historical data. By applying regression techniques, the model aims to analyze trends, identify growth patterns, and provide visual insights to support public health planning and decision-making amid the ongoing pandemic..

📦 2. Dataset Description

The dataset used in this project is sourced from the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) and contains:

- **Daily confirmed COVID-19 cases for various countries.**
- **Format: Dates as rows, country names as columns.**
- **File used: CONVENIENT_global_confirmed_cases.csv (cleaned version of raw global case data).**

Key Features:

- **Date-wise cumulative confirmed cases.**
- **Multiple countries included; India was selected for forecasting.**
- **Data range: From January 23, 2020 to the last recorded date.**

🧠 3. Objective

The primary objective of this project is to:

Forecast future COVID-19 cases in India using regression-based time series modeling techniques.

Specific goals:

- **Understand growth trends in COVID-19 case counts.**
- **Predict the number of future cases for the next 30 days.**
- **Visualize the results for decision-making or trend analysis.**

🔍 4. Data Preprocessing

The raw data required several transformation steps:

✅ Steps:

1. **Removed metadata rows (first row was not actual data).**
2. **Converted the 'Country/Region' column to DateTime format.**
3. **Set the Date as the index of the dataframe.**
4. **Selected India's column to isolate case counts.**
5. **Handled missing values by dropping them using .dropna().**
6. **Created features for regression:**
    - **Independent variable: Day index (0, 1, 2, ..., N)**
    - **Dependent variable: Confirmed COVID-19 case**

📐 5. Model Building

We used Linear Regression, a simple supervised learning method, to model the trend of cases.

📌 Steps:

- **Model: LinearRegression() from sklearn.linear_model**
- **Training data: All days except last 30 days**
- **Future prediction: Next 30 days beyond the available data**

🛠 Tools & Libraries:

- **pandas: For data manipulation**
- **numpy: For numerical operations**
- **scikit-learn: For machine learning model**
- **matplotlib: For visualization**

🎯 6. Results

- **The model successfully learned a linear trend in India's COVID-19 case counts.**
- **Future predictions followed the established upward trajectory.**

📈 Plot:

A graph was generated showing:

- **Blue line: Actual cumulative cases**
- **Orange dashed line: Predicted trend (including 30 future days)**
- **Gray dotted line: Train/test split point**

🛠️ Approach to Solving the Customer Churn Prediction Problem

📥 Data Collection

Source: Johns Hopkins University (JHU CSSE COVID-19 dataset).

File used: CONVENIENT_global_confirmed_cases.csv

Type: Time-series data (date-wise cumulative confirmed cases per country).

Country Selected: India

🖌️ Data Preprocessing

- **Remove metadata rows and irrelevant columns.**
- **Convert date strings to datetime format.**
- **Set Date column as the index.**
- **Select country-specific data (India).**
- **Convert cumulative case numbers to numeric format.**
- **Handle missing/null values using .dropna() or imputation.**
- **Create time index as feature (e.g., day 0, 1, 2, ..., N).**

📊 Exploratory Data Analysis (EDA)

- **Plot raw case trends using Matplotlib/Seaborn.**
- **Observe growth pattern (e.g., linear, exponential).**
- **Optionally, perform log transformation if exponential growth is observed.**

🧠 Model Selection & Building:

- **Linear Regression (baseline model).**
- **Polynomial Regression (to capture non-linearity).**
- **Advanced options: ARIMA, Facebook Prophet, LSTM (if needed).**
- **For Linear Regression:**
- **X (feature): time (in days).**
- **y (target): cumulative confirmed cases.**

Model Training:

- **Train on majority of data (e.g., all but last 30 days).**
- **Predict future values using extrapolated time steps.**

💉 **Model Evaluation:**

- **MAE (Mean Absolute Error)**
- **MSE (Mean Squared Error)**
- **RMSE (Root Mean Squared Error)**
- **R^2 Score (goodness-of-fit)**
- **Evaluate predictions on both train and test splits.**

📈 **Visualization**

- **Plot actual vs predicted case counts.**
- **Use a vertical line to indicate the train-test split.**
- **Forecast 30 future days and display as a dotted/dashed line.**

📌 **Result Interpretation**

- **Assess whether the model captures the trend well.**
- **Use R^2 to check how well the model explains variance.**
- **Identify overfitting or underfitting based on metrics.**

🛠 **Recommendations**

- **If linear model underfits, try polynomial or exponential fitting.**
- **For realistic forecasting, consider epidemiological models or LSTM/Prophet.**

📦 **Future Enhancements**

- **Use daily new cases instead of cumulative counts.**
- **Incorporate external features (e.g., lockdown dates, testing rates).**
- **Try ensemble methods or neural networks for more accurate prediction.**

CODE

```python
# Step 1: Upload the dataset.zip
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen

```python
# Step 2: Extract the zip file
import zipfile
import os

zip_path = 'dataset.zip'
extracted_path = 'covid_dataset'

with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extracted_path)

os.listdir(extracted_path)
```

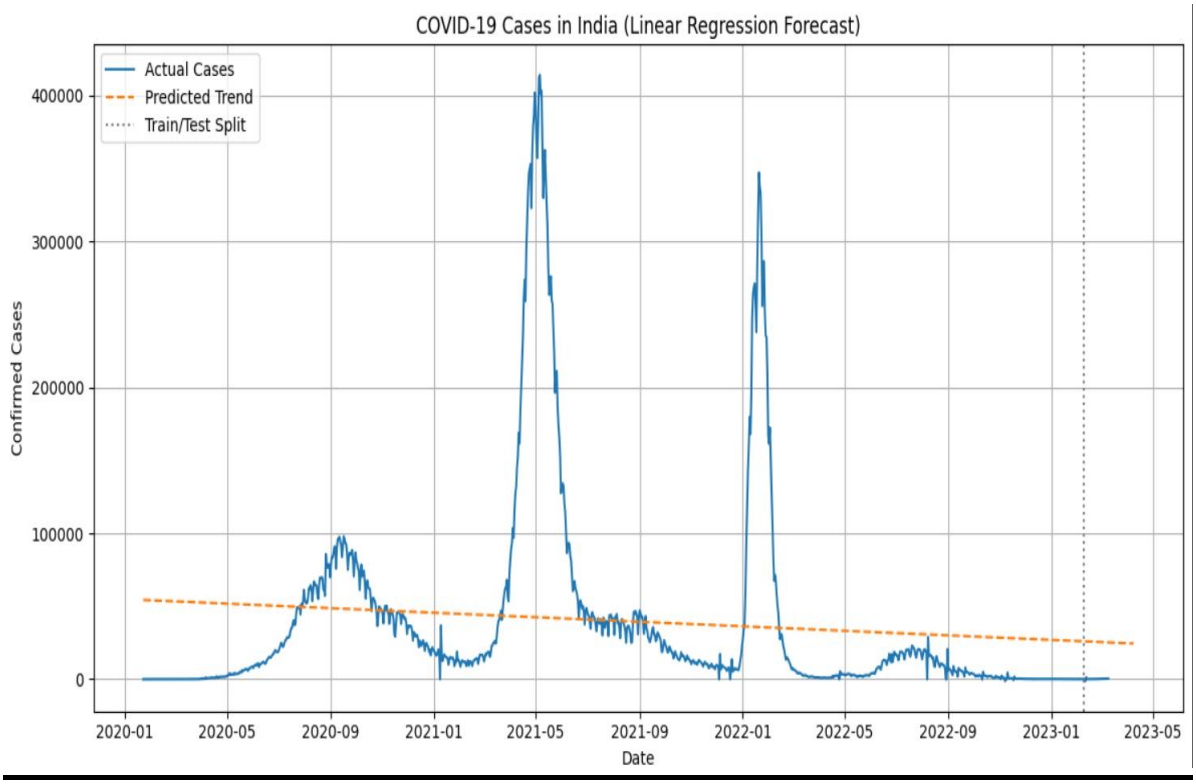
-

```python
# Step 3: Load global confirmed cases data
import pandas as pd

file_path = os.path.join(extracted_path, 'CONVENIENT_global_confirmed_cases.csv')
df = pd.read_csv(file_path)

# Drop the metadata row and fix headers
df = df.drop(index=0)
df.rename(columns={'Country/Region': 'Date'}, inplace=True)
df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
df = df.apply(pd.to_numeric, errors='coerce')

# Extract India's time series
india_cases = df['India'].dropna()
india_cases.head()
```

```python
# Step 4: Linear Regression and Prediction
from sklearn.linear_model import LinearRegression
import numpy as np
import matplotlib.pyplot as plt

# Prepare features (days since start)
X = np.arange(len(india_cases)).reshape(-1, 1)
y = india_cases.values

# Train/test split
train_size = len(X) - 30
X_train, y_train = X[:train_size], y[:train_size]

# Train linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict for historical + 30 future days
X_future = np.arange(len(india_cases) + 30).reshape(-1, 1)
y_pred = model.predict(X_future)

# Create date range for full prediction
future_dates = pd.date_range(start=india_cases.index[0], periods=len(X_future))
```

```
# Plotting
plt.figure(figsize=(12, 6))
plt.plot(india_cases.index, y, label='Actual Cases')
plt.plot(future_dates, y_pred, label='Predicted Trend', linestyle='--')
plt.axvline(india_cases.index[train_size], color='gray', linestyle=':', label='Train/Test Split')
plt.title('COVID-19 Cases in India (Linear Regression Forecast)')
plt.xlabel('Date')
plt.ylabel('Confirmed Cases')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

RESULT


COVID-19 Cases in India (Linear Regression Forecast)
Actual Cases
Predicted Trend
Train/Test Split
Confirmed Cases
400000
300000
200000
100000
0
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01
2022-05
2022-09
2023-01
2023-05
Date