Epidemic Prediction System: Algorithm

1. Overview

The system predicts epidemic outbreaks by allowing doctors or users to upload data files. The algorithm handles two cases:

1. Doctor knows the pathogen type: Pathogen type is selected.

2. Doctor does not know the pathogen type: CSV file is uploaded for automated processing.

2. Step-by-Step Workflow

File Upload Stage

User Action: The doctor selects pathogen type (optional) and uploads a CSV file.

Route: upload_and_predict function in views.py processes the file upload.

Processing Stage

Function Called: process_and_predict in prediction.py.

Input:

- CSV file (containing features like cases, deaths, dates, etc.).

- Optional parameter: Pathogen type (e.g., cholera, ebola, etc.).

Steps:

1. Read the CSV file into a DataFrame.

2. Check for missing pathogen type.

  - If provided, use the pathogen-specific prediction logic.

  - If not provided, infer the pathogen type based on data trends.

3. Clean the data (handle missing/null values).

4. Extract relevant features like cases, deaths, and population density.

5. Apply the Random Forest Classifier model for prediction.

6. Calculate the epidemic probability and thresholds.

Output Stage

Generated Outputs:

- Epidemic prediction result in percentage.

- Graphs:

- Case vs Time

- Death vs Time

- Growth Rate vs Time

Display:

Results and graphs are rendered in the result.html template.

An interactive slider is provided for daily trends.

3. Code Integration

```
views.py (upload_and_predict function):

def upload_and_predict(request):

    if request.method == 'POST':

        uploaded_file = request.FILES['csv_file']

        pathogen_type = request.POST.get('pathogen_type', None)

        prediction_result, graphs = process_and_predict(uploaded_file, pathogen_type)

           return render(request, 'result.html', {'result': prediction_result, 'graphs':

graphs})

    return render(request, 'upload.html')


prediction.py (process_and_predict function):

import pandas as pd

from model import random_forest_classifier
```

```
def process_and_predict(csv_file, pathogen_type=None):
    df = pd.read_csv(csv_file)
    df = df.dropna()
    features = df[['Cases', 'Deaths', 'Population_density']]
    result = random_forest_classifier.predict(features)
    graphs = generate_graphs(df)
    return result, graphs
```

4. End Result

The system ensures:

- Accurate epidemic prediction.

- User-friendly output visualization.

- Support for both known and unknown pathogen types.