Project Report

Project: DBSCAN-Covid-Contact-Tracing

Dhiraj H. Gawhare
Roll Number: 22M0062
M. Tech (Aerodynamics)
IIT Bombay.

Project Report: Analyzing Location Data and Cluster-based Infection Analysis DBSCAN Covid contact tracing

Objective: The goal of this project was to analyze location data, perform clustering using DBSCAN, and identify potentially infected individuals based on their proximity in the clusters.

Technologies Used:

- Python
- NumPy
- Pandas
- Matplotlib
- Seaborn
- Scikit-learn (DBSCAN)

Project Steps:

1. **Data Loading and Conversion:**
    - The project started by importing necessary libraries, including NumPy, Pandas, Matplotlib, Seaborn, and Scikit-learn.
    - Location data was loaded from a JSON file named 'livedata.json' into a Pandas DataFrame named **df**.
    - The DataFrame was then converted into a CSV file named 'livedata.csv' for easier storage and sharing.
2. **Data Visualization:**
    - Kernel Density Estimation (KDE) plots were used to visualize the distribution of longitude, latitude, and timestamp data using Seaborn's **sns.kdeplot()**.
    - The x-axis labels of the timestamp KDE plot were rotated for better readability.
3. **Geospatial Scatter Plot:**

- A scatter plot was created using Seaborn's **sns.scatterplot()** to visualize the geospatial distribution of latitude and longitude data.
- Points were colored based on their 'id' values, and a legend was added to the plot for clarity.

4. **DBSCAN Clustering:**
    - The DBSCAN clustering algorithm from Scikit-learn was employed to cluster data points based on their geographic proximity.
    - The 'haversine' distance metric was used to consider the Earth's curvature while calculating distances.
    - The clustering results were assigned to the DataFrame **df** as a new column named 'cluster'.
5. **Cluster Analysis and Visualization:**
    - The clustered DataFrame was saved to a CSV file named 'clustered.csv' without the index column.
    - A count plot (**sns.countplot()**) was created to display the distribution of points in each cluster, excluding noise points.
6. **Visualizing Clustered Data:**
    - A scatter plot was generated using Seaborn's **sns.scatterplot()** to visualize the clustered data points.
    - Points were colored by their cluster labels, and noise points were labeled as "Noise." A legend was added for clarity.
7. **Identifying Potentially Infected Individuals:**
    - A function named **get_infected_names(input_name)** was defined to find potentially infected individuals based on clusters.
    - The function iterated through the clusters associated with the input name.
    - For each cluster, it retrieved IDs of individuals in the same cluster (excluding the input name and already identified infected names).
    - The function returned a list of potentially infected names.

Conclusion: In this project, we successfully analyzed and visualized location data, performed clustering using the DBSCAN algorithm, and identified potential infections based on cluster proximity. The visualizations provided insights into the distribution of location data and clusters, while the infection analysis could help in identifying individuals at risk within the same clusters. The success of this analysis relies on accurate data and appropriate clustering parameters. Further enhancements could involve refining the clustering approach and integrating more complex data sources for a more comprehensive analysis