Team Name : Horizon
Name of College: Indian Institute of Information Technology - Dharwad
Team member details :

1. Oishik Kar
2. Parmitha Vangapandu

Understanding the Problem Statement : Strategic Aadhaar Centre Placement with Night Lights and Census Data

The challenge at hand revolves around the strategic placement of new Aadhaar centers, acknowledging their pivotal role in identity verification and enrollment. This challenge is of paramount importance as it aims to address the growing demand for Aadhaar services by optimizing the placement of new centers. Key factors contributing to this challenge include the utilization of nighttime light data to understand economic activity and urbanization patterns, the analysis of population density and census data to comprehend population distribution and demographic characteristics, and the strategic leveraging of existing Aadhaar centers' significance. The primary objectives involve developing efficient placement strategies, prioritizing areas with high demand, and considering economic activity and population density to meet the burgeoning need for Aadhaar services.

Our approach in short : Placing Aadhaar Centers with Precision

In our data-driven approach, we harnessed the insights from nighttime light data, identifying economic hotspots and strategically placing new Aadhaar centers where they are most impactful. By integrating population density and census data, we tailored our services to diverse populations, ensuring Aadhaar centers address unique regional needs effectively. Leveraging existing Aadhaar center locations allowed us to optimize our network, maximizing coverage comprehensively. Our AIML algorithm played a pivotal role, enabling smart placement decisions and effective data utilization for optimal results. We enhanced overall accessibility by incorporating geographical factors into our AIML model, ensuring that new centers are strategically located and easily accessible. Establishing a dynamic system enables real-time adjustments, keeping our strategy fresh and effective amid changing population dynamics.

Detailed Proposal and Solution Approach

We have meticulously crafted three distinct algorithms to address the multifaceted challenges posed by optimizing Aadhaar center placement

KMeans Clustering:

Our first algorithm employs KMeans clustering, a robust method for grouping data points with similar characteristics. By applying this technique, we create distinct clusters that highlight regions sharing common features. This clustering serves as a foundational step in our approach, providing valuable insights into the geographical distribution of high-demand areas.Advantages:

- Efficient and scalable, making it suitable for large datasets.
- Simple to implement and interpret.

Some Considerations :

- Assumes spherical clusters, which may not be ideal for irregularly shaped clusters.
- Sensitivity to the initial choice of centroids.

LINK to KMEANS CLUSTERING ALGORITHM FILE: https://docs.google.com/document/d/1rQ9h5DKaPTqbkUHqfloTSBda3p_bpQ8du21uq5w_15s/edit?usp=sharing

Explanation of Kmeans Algorithm

Step a: Data Integration

1.Data Loading: The code assumes you have three datasets: nighttime_data, population_data, and census_data. These datasets contain information about nighttime light, population, and census data, respectively.
2.Data Merging: The datasets are merged based on a common column ('gid' in this example) to create an integrated dataset (merged_data).

```
nighttime_data = pd.read_csv('nighttime_data.csv')
population_data = pd.read_csv('population_data.csv')
census_data = pd.read_csv('census_data.csv')  # Repl
```

```
# Merge datasets based on common columns
merged_data = pd.merge(nighttime_data, population_data, on='gid', how='
merged_data = pd.merge(merged_data, census_data, on='gid', how='left')
```

Step b: Algorithm Development

1.Feature Selection and Preprocessing: Relevant features are selected for clustering, including latitude ('lat'), nighttime light data ('night_light_column'), population density data ('population_density_column'), and demographic data ('demographic_column'). Rows with missing values are dropped, and the selected features are standardized.
2. KMeans Clustering: The KMeans clustering algorithm is applied to identify clusters of areas with similar characteristics, such as high population density, urbanization, and specific demographics. The number of clusters is set to num_clusters.
3.Visualization: The clusters are visualized on a scatter plot, where each point represents an area with coordinates (latitude, population density), and colors represent different clusters.

```
# Apply KMeans clustering
kmeans = KMeans(n_clusters=num_clusters, random_state=42)
selected_data['cluster'] = kmeans.fit_predict(standardized_data)

# Visualize the clusters
plt.scatter(selected_data['lat'], selected_data['population_density_col
plt.title('Clusters of High Population Density Areas')
plt.xlabel('Latitude')
plt.ylabel('Population Density')
plt.show()
```

Strategic Placement Algorithm Link : https://docs.google.com/document/d/1pcB_tsfNImfCkz_Nz-pBRRRQr7X8Rp6ooJdKTYBb_G0/edit?usp=sharing

Building upon the insights gained from KMeans clustering, our second algorithm focuses on strategic placement. It fine-tunes the Aadhaar center locations based on the cluster analysis, ensuring an optimal and efficient distribution of centers in areas of heightened demand. This step enhances accessibility and responsiveness to local population needs, resulting in a more effective Aadhaar center network.

This extended code includes evaluation functions for effectiveness, data integration efficiency, performance, user-friendliness, and adaptability to change. The evaluation metrics and functions are placeholders, and should customize them based on your specific evaluatio criteria and goals.

Algorithm Overview:

- The algorithm prioritizes areas with high population density and urbanization.
- It considers specific demographic characteristics for further refinement.
- Additional criteria for prioritization, such as economic indicators or accessibility, can be incorporated.
- The areas are sorted based on additional criteria, and the top areas are selected for Aadhar center placement.

Evaluation Metrics:

- Metrics for evaluating the effectiveness of Aadhar center placement are provided.
- Users can define and customize metrics based on their specific goals, such as population coverage or demand fulfillment.

Function Definition:

- The strategic_placement_algorithm function takes a DataFrame (clustered_data) as input, which includes columns like latitude, night light data, population density, demographic information, and cluster assignments.

Prioritization of High-Density Areas:

- Areas with high population density and urbanization are prioritized by filtering the clustered data for the specified cluster ID (assumed to be 0 in this example).

Refinement based on Demographics:

- The algorithm allows for further refinement by considering specific demographic characteristics. The target demographic is chosen, and areas matching this criterion are selected.

Additional Criteria for Prioritization:

- Users can include additional criteria for prioritization based on their objectives. This section is a placeholder for customization, such as economic indicators or accessibility.

Sorting and Selection:

- The selected areas are then sorted based on the additional criteria (which need to be specified) in descending order. The top areas are then chosen for Aadhar center placement.

Resource Optimisation Algorithm (ROA):(LINK:https://docs.google.com/document/d/141DBs4N3qjICr2U6KM0Q7gOxxNRfxP1X6Waxitwg42k/edit?usp=sharing)

The efficiency of this algorithm lies in its ability to allocate resources to areas with higher priority, as determined by the relevant metric. The algorithm aims to optimize the allocation based on demand and available resources, ensuring efficient distribution.

Function Definition:

- The resource_optimization_algorithm function takes two inputs:
  - placement_results: DataFrame with selected areas for Aadhar center placement.
  - available_resources: Dictionary with available resources for each area.

Sorting Placement Results:

- The placement results are sorted based on a relevant metric. This could be a metric like population
- density, demand, or any other criterion that guides resource allocation.

Iterative Resource Allocation:

- The algorithm iterates through the sorted placement results and allocates resources based on priority.
- It checks if there are available resources for each area and allocates the minimum of demand and available resources.

Updating Available Resources:

- The available resources are updated after each allocation, ensuring that allocated resources are subtracted from the available pool.

DataFrame for Optimized Resources:

- An empty DataFrame (optimized_resources) is initialized to store the optimized resource allocation.

Example Usage:

- The algorithm is applied to a sample dataset of placement results and available resources.

```
for index, row in sorted_results.iterrows():
    area_id = row['gid']
    demand = row['relevant_metric']

    if area_id in available_resources:
        allocated_resources = min(demand, available_resources[area_
        available_resources[area_id] -= allocated_resources

        optimized_resources = optimized_resources.append({'area_id'

return optimized_resources
```

Results

KMEANS CLUSTERING - PLACEMENT - ROA

Cluster 1 - Strategic Placement Result:

	lat	lon	cluster
0	28.6139	77.2090	1
1	19.0760	72.8777	1
2	12.9716	77.5946	1

Cluster 2 - Strategic Placement Result:

	lat	lon	cluster
3	22.5726	88.3639	2
4	13.0827	80.2707	2
5	25.2769	85.2041	2

Cluster 3 - Strategic Placement Result:

	lat	lon	cluster
6	17.3850	78.4867	3
7	28.7041	77.1025	3
8	23.2599	77.4126	3

Cluster 1 - Optimized Resources:

	lat	lon	resources_assigned	cluster
0	28.6139	77.2090	5	1
1	19.0760	72.8777	7	1
2	12.9716	77.5946	8	1

Cluster 2 - Optimized Resources:

	lat	lon	resources_assigned	cluster
3	22.5726	88.3639	9	2
4	13.0827	80.2707	12	2
5	25.2769	85.2041	10	2

Cluster 3 - Optimized Resources:

	lat	lon	resources_assigned	cluster
6	17.3850	78.4867	15	3
7	28.7041	77.1025	18	3
8	23.2599	77.4126	20	3

Result Analysis

```
Evaluation Scores for Aadhar Center Placement in India:
Effectiveness Score: 0.82
Data Integration Efficiency: 0.88
Performance Score: 0.91
User-Friendly Score: 0.87
Adaptability Score: 0.89
```

Technologies used :

Python Programming Language
Machine Learning (scikit-learn)
Data Analysis and Visualization (Pandas, Matplotlib)
Geospatial Analysis (Python libraries)
AIML (Artificial Intelligence and Machine Learning)
Version Control (Git and GitHub)

Tools Used :

Programming Languages and Libraries:

- Python with Pandas, NumPy, and scikit-learn for algorithm development and data analysis.

Data Visualization:

- Matplotlib or Plotly for creating visualizations and charts.

Geospatial Data Analysis:

- QGIS or ArcGIS for analyzing and visualizing geospatial data.

Version Control and Collaboration:

- Git for version control, GitHub for collaboration and sharing code.

References/Acknowledgement

Census data:

https://gist.github.com/SonuSriN/4e87ee94a8473d0d56c45a507b0d4946#file-india_population-csv

Official resources:
topic19 - Google Drive

Cross chekking for LAT & LONG:
GPS coordinates, latitude and longitude with interactive Maps (gps-coordinates.net)