Demographic Geofencing

Clustering Analysis for Targeted Resource Allocation and Strategic Urban Planning
A machine learning approach to regional demographic categorization.

Arun

Executive Summary & Objective

Building a framework for data-driven decisions via machine learning and regional categorization

Segment Geographical Regions Based On **Age Demographics** Using Unsupervised Learning.

Categorize Areas Into **Youth-Dominant**, Adult-Dominant, And Balanced Profiles.

Enable Efficient **Resource Allocation**, Housing Policy, And Infrastructure Planning.

Maintain A **Privacy-First** Approach Utilizing Aggregated Data Over Individual Identifiers.

Methodology: The Clustering Approach

Utilizing unsupervised machine learning to identify distinct regional demographic patterns

01

Data Inputs

Processing proportions of Youth (Age 5-17) and Adults (Age 17+) for all target regions.

02

K-Means Clustering

Applying the **K-Means** algorithm to group districts and pincodes based on demographic similarity.

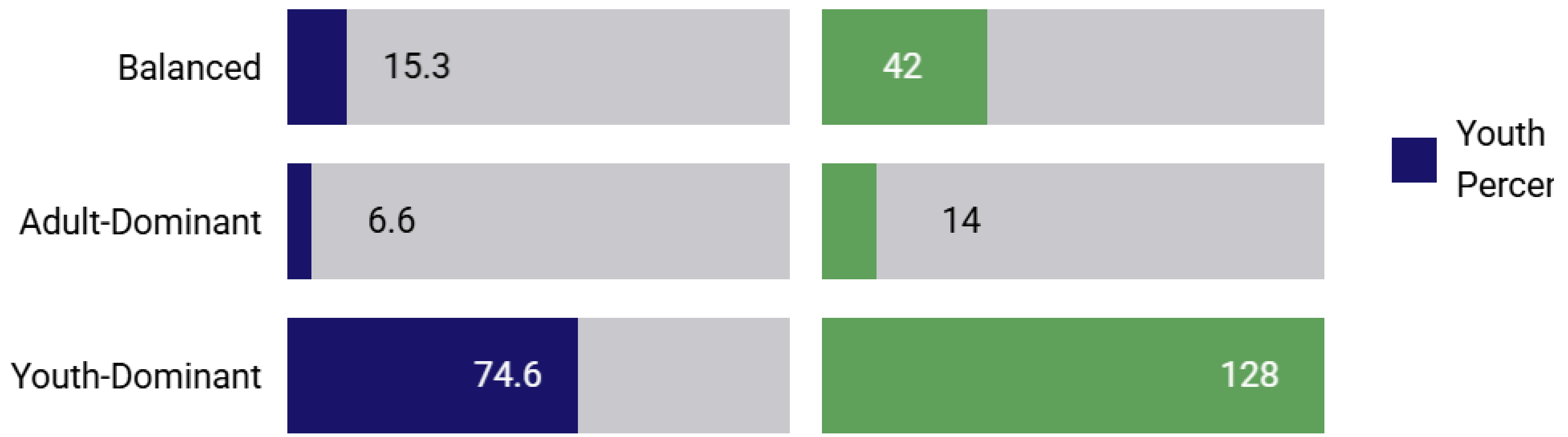
03

Granularity Levels

Conducting analysis at both Macro (Districts) and Micro (Pincodes) levels for comprehensive coverage.

District Cluster Profiles

The Macro View: Visualizing regional distribution through broad demographic characteristics



Pincode Cluster Profiles

The Micro View: Hyper-localized demographic mapping for precision resource planning

Cluster ID	Profile Type	Primary Metric	Strategic Focus
P-C1	Highly Adult-Dominant	94.1% Adults	Urbanized centers and retirement zones
P-C2	Youth-Balanced	21% Youth	Emerging residential for young families
P-C3	Highly Youth-Dominant	82.5% Youth	Educational campuses and child-care zones

Targeted Resource Allocation

Actionable Insight 1: Aligning essential services with specific regional demographic needs

Pediatric Health

Targeted deployment of healthcare services in high-concentration **Youth-Dominant** zones.

School Infrastructure

Prioritizing educational facilities in areas with growing youth populations like Cluster **P-C3**.

Vocational Training

Focusing workforce upskilling in Adult-Dominant urbanized centers and business hubs.

Geriatric Services

Allocating senior-care and specialized medical support in mature regions.

Strategic Urban Planning

Actionable Insight 2: Solving the urban development puzzle through demographic synchronization

01 Housing Strategy

Matching family-sized apartments to Youth zones vs. studio units for Adult zones.

02 Transit Routes

Optimizing 'School-run' paths for youth vs. 'Commuter-hub' routes for working adults.

03 Public Spaces

Developing child-friendly spaces in P-C2/P-C3 vs. retirement-friendly zones in P-C1.

Policy-Making & Governance

Actionable Insight 3: Shifting from national averages to targeted, privacy-preserving governance

Demographic 'Character' Is A Better Predictor Of Regional Needs Than **Raw Population Counts**.

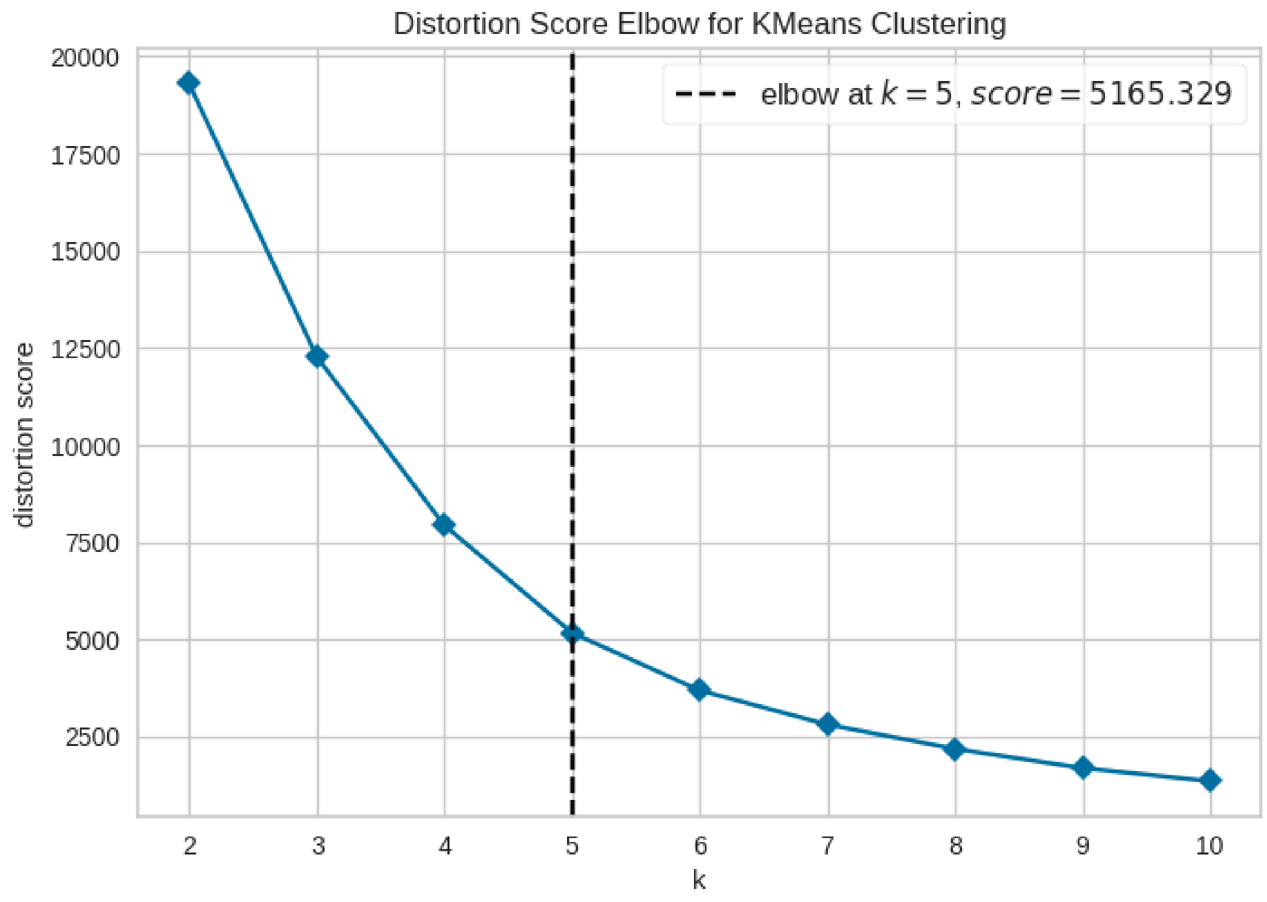
Move Away From **One-Size-Fits-All** Policies To Localized Governance.

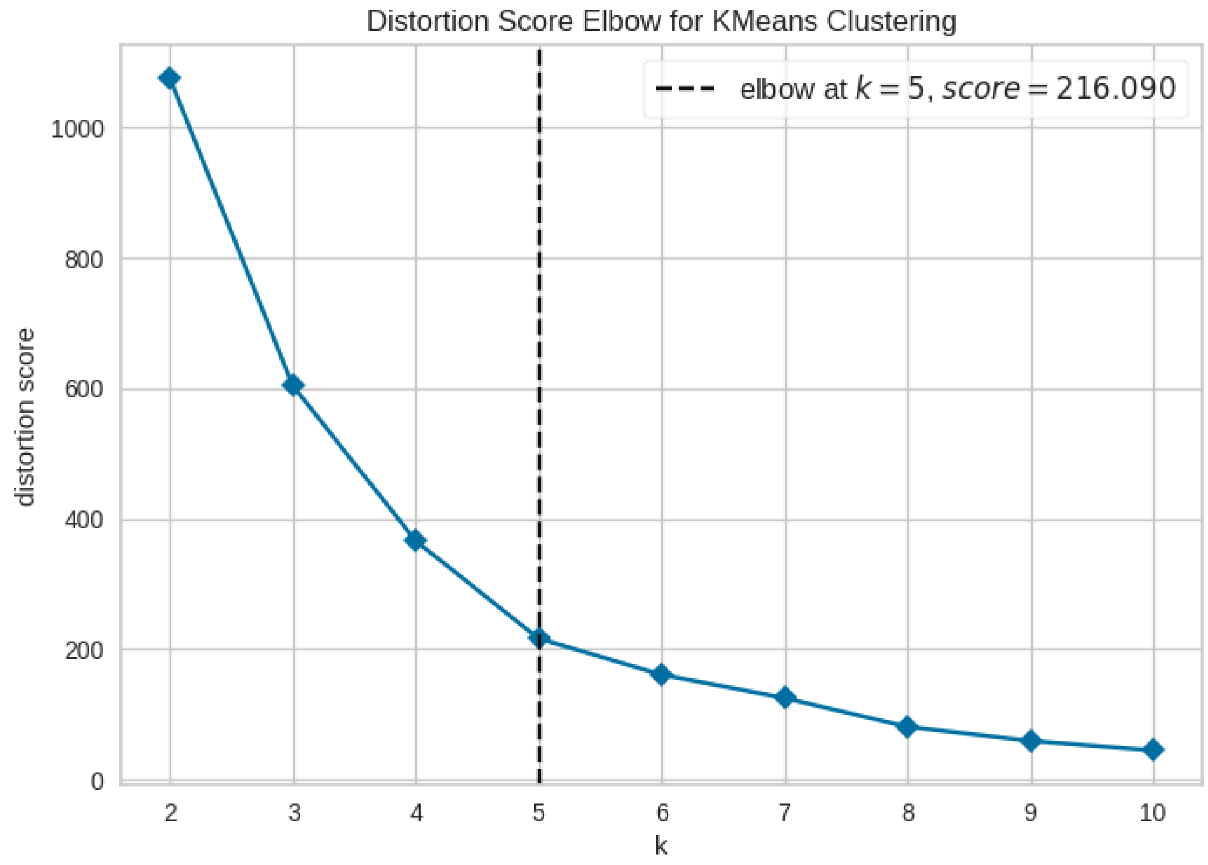
Privacy Remains Central By Using **Aggregated Data** Rather Than Individual Identifiers.

Conclusion & Next Steps

Transitioning from model analysis to a pilot resource allocation program in high-priority zones
Launch pilot in Cluster **P-C3** and begin temporal demographic shifts analysis.

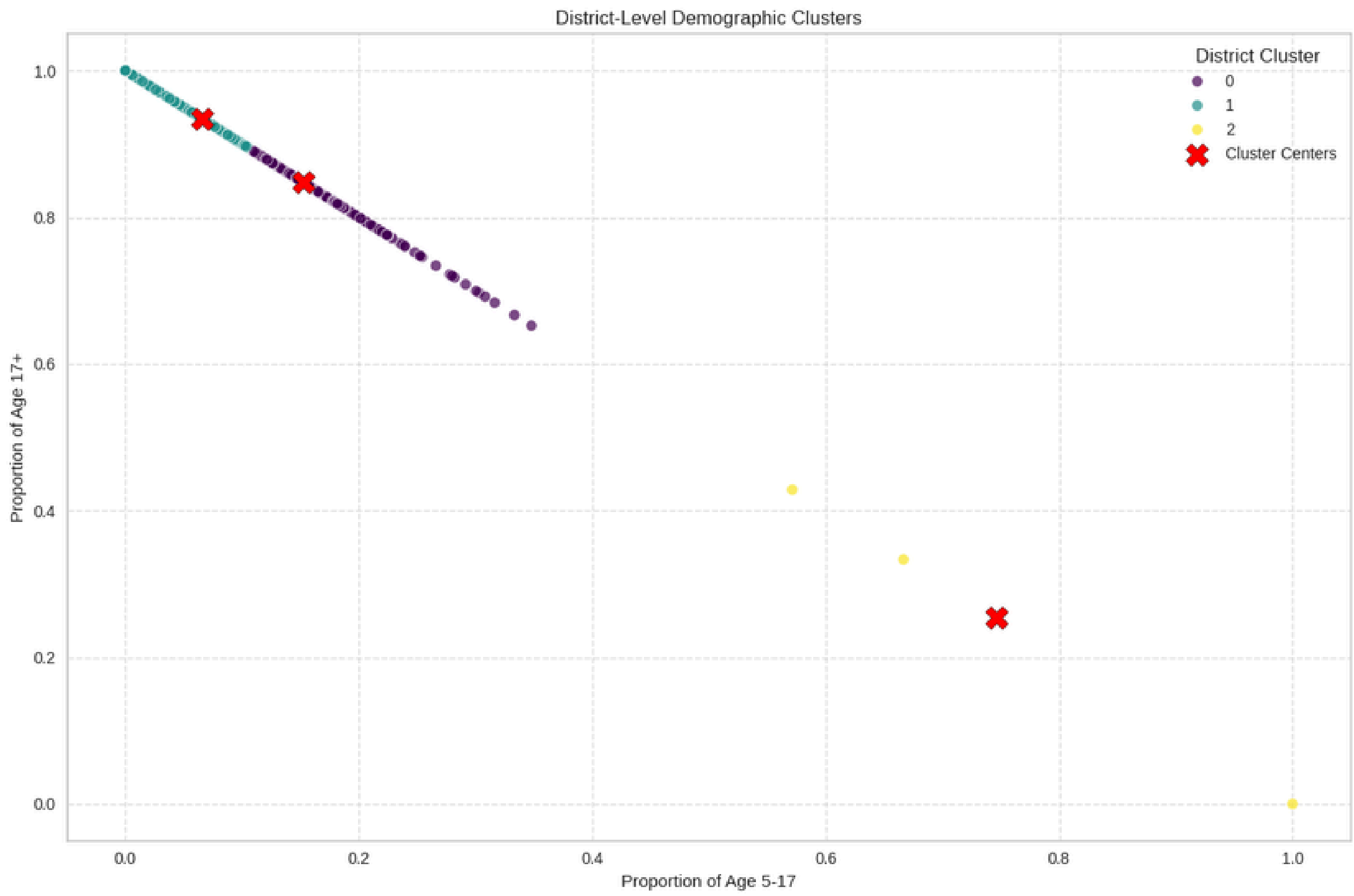
Code & Output



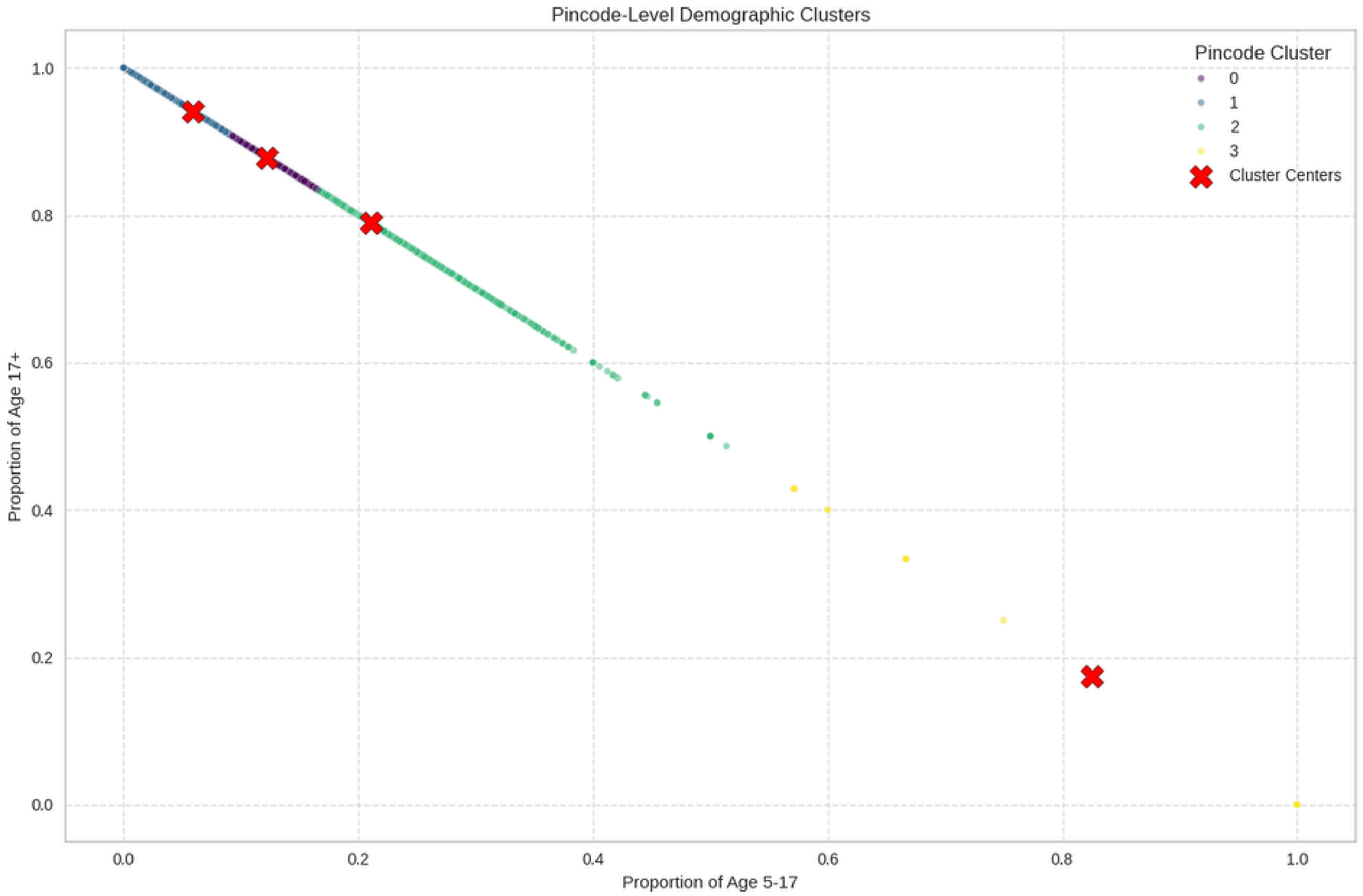



```
# Determine optimal clusters for district-level data
model_districts = KMeans(random_state=42, n_init='auto')
visualizer_districts = KElbowVisualizer(model_districts, k=(2,11), timings=False)
print("Elbow Method for District Data:")
visualizer_districts.fit(X_districts_scaled)
visualizer_districts.show()

# Determine optimal clusters for pincode-level data
model_pincodes = KMeans(random_state=42, n_init='auto')
visualizer_pincodes = KElbowVisualizer(model_pincodes, k=(2,11), timings=False)
print("\nElbow Method for Pincode Data:")
visualizer_pincodes.fit(X_pincodes_scaled)
visualizer_pincodes.show()

# Based on the elbow plots, let's assume optimal K for districts is 3 and for pincodes is 4 (this wil
# Instantiate KMeans for district data and assign labels
k_districts = 3 # Placeholder, to be adjusted after viewing plot
kmeans_districts = KMeans(n_clusters=k_districts, random_state=42, n_init='auto')
district_demographics_combined['district_cluster'] = kmeans_districts.fit_predict(X_districts_scaled)

# Instantiate KMeans for pincode data and assign labels
k_pincodes = 4 # Placeholder, to be adjusted after viewing plot
kmeans_pincodes = KMeans(n_clusters=k_pincodes, random_state=42, n_init='auto')
pincode_demographics_combined['pincode_cluster'] = kmeans_pincodes.fit_predict(X_pincodes_scaled)
```

Code & Output




```python
# 1. Visualize District-Level Clusters
plt.figure(figsize=(12, 8))
sns.scatterplot(
    x='district_prop_age_5_17',
    y='district_prop_age_17_',
    hue='district_cluster',
    data=district_demographics_combined,
    palette='viridis',
    s=50, alpha=0.7, legend='full'
)
plt.scatter(
    district_centers_unscaled[:, 0],
    district_centers_unscaled[:, 1],
    marker='X', s=200, c='red', edgecolor='black', label='Cluster Centers'
)
plt.title('District-Level Demographic Clusters')
plt.xlabel('Proportion of Age 5-17')
plt.ylabel('Proportion of Age 17+')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend(title='District Cluster')
plt.tight_layout()
plt.show()
```




```python
# 2. Visualize Pincode-Level Clusters
plt.figure(figsize=(12, 8))
sns.scatterplot(
    x='pincode_prop_age_5_17',
    y='pincode_prop_age_17_',
    hue='pincode_cluster',
    data=pincode_demographics_combined,
    palette='viridis',
    s=20, alpha=0.5, legend='full'
)
plt.scatter(
    pincode_centers_unscaled[:, 0],
    pincode_centers_unscaled[:, 1],
    marker='X', s=200, c='red', edgecolor='black', label='Cluster Centers'
)
plt.title('Pincode-Level Demographic Clusters')
plt.xlabel('Proportion of Age 5-17')
plt.ylabel('Proportion of Age 17+')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend(title='Pincode Cluster')
plt.tight_layout()
plt.show()
```

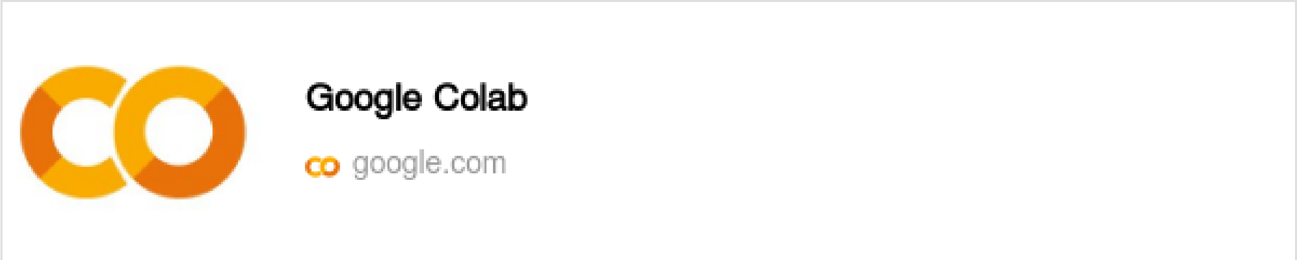
Code & Output

Machine Learning Model

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler

# Create separate scalers for districts and pincodes to correctly inverse transform cluster centers
scaler_districts = StandardScaler()
scaler_districts.fit(X_districts)

scaler_pincodes = StandardScaler()
scaler_pincodes.fit(X_pincodes)

# Inverse transform cluster centers for districts
district_centers_scaled = kmeans_districts.cluster_centers_
district_centers_unscaled = scaler_districts.inverse_transform(district_centers_scaled)

# Inverse transform cluster centers for pincodes
pincode_centers_scaled = kmeans_pincodes.cluster_centers_
pincode_centers_unscaled = scaler_pincodes.inverse_transform(pincode_centers_scaled)
```

Code Link to access

click here for colob link



Github Link

click here for the github