Exploratory Data Analysis of Aadhaar Enrolment Data

Problem Statement

Aadhaar enrolment data captures Enrolment patterns across different age groups over time. Understanding these patterns is crucial for identifying enrolment trends, enrolment participation, and potential accessibility gaps.

This study performs a structured Exploratory Data Analysis (EDA) on Aadhaar enrolment data to:

- Understand enrolment distribution across age groups
- Identify temporal trends
- Explore relationships between Enrolment groups
- Derive insights with potential administrative and social impact

Importing api_data_aadhar_enrolment_0_500000.csv

In [953…

```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df1 = pd.read_csv('/Users/karansingh/Desktop/Hackathon/DAtaHackathon/api_dat
```

Dataset Overview

Understanding dataset structure, size, and column information.

In [954…

```
df1.head()
```

Out[954…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
0	02-03-2025	Meghalaya	East Khasi Hills	793121	11	61	37
1	09-03-2025	Karnataka	Bengaluru Urban	560043	14	33	39
2	09-03-2025	Uttar Pradesh	Kanpur Nagar	208001	29	82	12
3	09-03-2025	Uttar Pradesh	Aligarh	202133	62	29	15
4	09-03-2025	Karnataka	Bengaluru Urban	560016	14	16	21

In [955…
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500000 entries, 0 to 499999
Data columns (total 7 columns):
 #   Column          Non-Null Count   Dtype 
---  ------          --------------   ----- 
 0   date            500000 non-null  object
 1   state           500000 non-null  object
 2   district        500000 non-null  object
 3   pincode         500000 non-null  int64 
 4   age_0_5         500000 non-null  int64 
 5   age_5_17        500000 non-null  int64 
 6   age_18_greater  500000 non-null  int64 
dtypes: int64(4), object(3)
memory usage: 26.7+ MB
```

In [956…
```
df1.shape
```

Out[956…
```
(500000, 7)
```

In [957…
```
df1.columns
```

Out[957…
```
Index(['date', 'state', 'district', 'pincode', 'age_0_5', 'age_5_17',
       'age_18_greater'],
      dtype='object')
```

In [958…
```
df1.dtypes
```

Out[958…
```
date              object
state             object
district          object
pincode            int64
age_0_5            int64
age_5_17           int64
age_18_greater     int64
dtype: object
```

In [959…
```
df1.describe()
```

Out[959…

	pincode	age_0_5	age_5_17	age_18_greater
count	500000.000000	500000.000000	500000.000000	500000.000000
mean	519204.051054	4.040812	2.315682	0.245558
std	206793.322085	24.417921	20.191196	4.438557
min	100000.000000	0.000000	0.000000	0.000000
25%	362229.000000	1.000000	0.000000	0.000000
50%	517131.000000	2.000000	0.000000	0.000000
75%	712139.000000	3.000000	1.000000	0.000000
max	855456.000000	2688.000000	1812.000000	855.000000

In [960…
```
## check null value
df1['state'].isnull().sum()
```

Out[960…
```
np.int64(0)
```

In [961…
```
## check state column
df1['state'].nunique()
```

Out[961…
```
54
```

Importing api_data_aadhar_enrolment_500000_10000

In [962…
```
df2 = pd.read_csv('/Users/karansingh/Desktop/Hackathon/DAtaHackathon/api_dat
```

Dataset Overview

Understanding dataset structure, size, and column information.

In [963…
```
df2.head()
```

Out[963…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
0	26-10-2025	Andhra Pradesh	Nalgonda	508004	0	1	0
1	26-10-2025	Andhra Pradesh	Nalgonda	508238	1	0	0
2	26-10-2025	Andhra Pradesh	Nalgonda	508278	1	0	0
3	26-10-2025	Andhra Pradesh	Nandyal	518432	0	1	0
4	26-10-2025	Andhra Pradesh	Nandyal	518543	1	0	0

In [964…
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500000 entries, 0 to 499999
Data columns (total 7 columns):
 #   Column          Non-Null Count   Dtype 
---  ------          --------------   ----- 
 0   date            500000 non-null  object
 1   state           500000 non-null  object
 2   district        500000 non-null  object
 3   pincode         500000 non-null  int64 
 4   age_0_5         500000 non-null  int64 
 5   age_5_17        500000 non-null  int64 
 6   age_18_greater  500000 non-null  int64 
dtypes: int64(4), object(3)
memory usage: 26.7+ MB
```

In [965…
```
# check shape
df2.shape
```

Out[965… `(500000, 7)`

In [966…
```
# check columns
df2.columns
```

Out[966…
```
Index(['date', 'state', 'district', 'pincode', 'age_0_5', 'age_5_17',
       'age_18_greater'],
      dtype='object')
```

In [967…
```
df2.describe()
```

Out[967…

	pincode	age_0_5	age_5_17	age_18_greater
count	500000.000000	500000.000000	500000.000000	500000.000000
mean	518077.362504	3.009628	1.082962	0.089984
std	204615.861812	4.658289	2.537596	1.073745
min	100000.000000	0.000000	0.000000	0.000000
25%	365541.000000	1.000000	0.000000	0.000000
50%	517583.000000	2.000000	0.000000	0.000000
75%	695023.000000	3.000000	1.000000	0.000000
max	855456.000000	210.000000	97.000000	318.000000

Importing api_data_aadhar_enrolment_1000000_1006(

In [968…

```
df3 = pd.read_csv('/Users/karansingh/Desktop/Hackathon/DAtaHackathon/api_dat
```

Dataset Overview

Understanding dataset structure, size, and column information.

In [969…

```
df3.head()
```

Out[969…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
0	31-12-2025	Karnataka	Bidar	585330	2	3	0
1	31-12-2025	Karnataka	Bidar	585402	6	0	0
2	31-12-2025	Karnataka	Bidar	585413	1	0	0
3	31-12-2025	Karnataka	Bidar	585418	1	2	0
4	31-12-2025	Karnataka	Bidar	585421	4	3	0

In [970…

```
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6029 entries, 0 to 6028
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   date            6029 non-null   object
 1   state           6029 non-null   object
 2   district        6029 non-null   object
 3   pincode         6029 non-null   int64 
 4   age_0_5         6029 non-null   int64 
 5   age_5_17        6029 non-null   int64 
 6   age_18_greater  6029 non-null   int64 
dtypes: int64(4), object(3)
memory usage: 329.8+ KB
```

In [971…
```
df3.shape
```

Out[971…
```
(6029, 7)
```

In [972…
```
df3.describe()
```

Out[972…

	pincode	age_0_5	age_5_17	age_18_greater
count	6029.000000	6029.000000	6029.000000	6029.000000
mean	518765.547023	3.606734	3.493448	0.096533
std	193308.752319	6.055847	6.694502	0.479475
min	110003.000000	0.000000	0.000000	0.000000
25%	380022.000000	1.000000	0.000000	0.000000
50%	518005.000000	2.000000	1.000000	0.000000
75%	685595.000000	4.000000	3.000000	0.000000
max	855116.000000	102.000000	89.000000	9.000000

Merging all three Enrolment datasets

In [973…
```
df = pd.concat([df1,df2,df3],ignore_index=True)
```

In [974…
```
#Datatype of column and number of values in each column are
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1006029 entries, 0 to 1006028
Data columns (total 7 columns):
 #   Column          Non-Null Count    Dtype
---  ------          --------------    -----
 0   date            1006029 non-null  object
 1   state           1006029 non-null  object
 2   district        1006029 non-null  object
 3   pincode         1006029 non-null  int64
 4   age_0_5         1006029 non-null  int64
 5   age_5_17        1006029 non-null  int64
 6   age_18_greater  1006029 non-null  int64
dtypes: int64(4), object(3)
memory usage: 53.7+ MB
```

In [975…

```
# Top rows of dataset are :
df.head()
```

Out[975…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
0	02-03-2025	Meghalaya	East Khasi Hills	793121	11	61	37
1	09-03-2025	Karnataka	Bengaluru Urban	560043	14	33	39
2	09-03-2025	Uttar Pradesh	Kanpur Nagar	208001	29	82	12
3	09-03-2025	Uttar Pradesh	Aligarh	202133	62	29	15
4	09-03-2025	Karnataka	Bengaluru Urban	560016	14	16	21

In [976…

```
# Bottom rows of dataset are :
df.tail()
```

Out[976…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
1006024	31-12-2025	West Bengal	West Midnapore	721149	2	0	0
1006025	31-12-2025	West Bengal	West Midnapore	721150	2	2	0
1006026	31-12-2025	West Bengal	West Midnapore	721305	0	1	0
1006027	31-12-2025	West Bengal	West Midnapore	721504	1	0	0
1006028	31-12-2025	West Bengal	West Midnapore	721517	2	1	0

In [977…
```
#Total Columns Present in Dataset
df.columns
```

Out[977…
```
Index(['date', 'state', 'district', 'pincode', 'age_0_5', 'age_5_17',
       'age_18_greater'],
      dtype='object')
```

In [978…
```
# check shape
print("Shape of df is :",df.shape)
```

```
Shape of df is : (1006029, 7)
```

In [979…
```
#Describing total count , mean, std deviation, min value , max value,25%ile,
df.describe()
```

Out[979…

	pincode	age_0_5	age_5_17	age_18_greater
count	1.006029e+06	1.006029e+06	1.006029e+06	1.006029e+06
mean	5.186415e+05	3.525709e+00	1.710074e+00	1.673441e-01
std	2.056360e+05	1.753851e+01	1.436963e+01	3.220525e+00
min	1.000000e+05	0.000000e+00	0.000000e+00	0.000000e+00
25%	3.636410e+05	1.000000e+00	0.000000e+00	0.000000e+00
50%	5.174170e+05	2.000000e+00	0.000000e+00	0.000000e+00
75%	7.001040e+05	3.000000e+00	1.000000e+00	0.000000e+00
max	8.554560e+05	2.688000e+03	1.812000e+03	8.550000e+02

Data Cleaning & Preprocessing

This step ensures consistency, correctness, and readiness for analysis.

In [980…
```
# Convert date column
df['date'] = pd.to_datetime(df['date'], dayfirst=True)

# Extract temporal features
df['year'] = df['date'].dt.year
df['month'] = df['date'].dt.month
df['month_name'] = df['date'].dt.strftime('%b')
```

In [981…
```
#Checking Count of Unique value present in sate
print("There are ",df['state'].nunique() ,"Values present in state column.")
```

```
There are  55 Values present in state column.
```

In [982…
```
#Checking for unique value present in state
df['state'].unique()
```

Out[982…
```
array(['Meghalaya', 'Karnataka', 'Uttar Pradesh', 'Bihar', 'Maharashtra',
       'Haryana', 'Rajasthan', 'Punjab', 'Delhi', 'Madhya Pradesh',
       'West Bengal', 'Assam', 'Uttarakhand', 'Gujarat', 'Andhra Pradesh',
       'Tamil Nadu', 'Chhattisgarh', 'Jharkhand', 'Nagaland', 'Manipur',
       'Telangana', 'Tripura', 'Mizoram', 'Jammu and Kashmir',
       'Chandigarh', 'Sikkim', 'Odisha', 'Kerala',
       'The Dadra And Nagar Haveli And Daman And Diu',
       'Arunachal Pradesh', 'Himachal Pradesh', 'Goa',
       'Jammu And Kashmir', 'Dadra and Nagar Haveli and Daman and Diu',
       'Ladakh', 'Andaman and Nicobar Islands', 'Orissa', 'Pondicherry',
       'Puducherry', 'Lakshadweep', 'Andaman & Nicobar Islands',
       'Dadra & Nagar Haveli', 'Dadra and Nagar Haveli', 'Daman and Diu',
       'WEST BENGAL', 'Jammu & Kashmir', 'West  Bengal', '100000',
       'Daman & Diu', 'West Bangal', 'Westbengal', 'West bengal',
       'andhra pradesh', 'ODISHA', 'WESTBENGAL'], dtype=object)
```

There are some state with different spelling We are going to map them to a single state name .

In [983…
```
import pandas as pd
import re

def clean_state_name(x):
    if pd.isna(x):
        return x
    x = str(x).lower()
    x = re.sub(r'[^a-z]', '', x) # remove symbols like &,
    x = re.sub(r'\s+',' ', x).strip() # remove extra spaces
    return x
```

In [984…
```
state_mapping = {
    # Andhra Pradesh
    "andhrapradesh": "Andhra Pradesh",

    # Arunachal Pradesh
    "arunachalpradesh": "Arunachal Pradesh",
```

```
    # Assam
    "assam": "Assam",

    # Bihar
    "bihar": "Bihar",

    # Chhattisgarh
    "chhattisgarh": "Chhattisgarh",

    # Delhi
    "delhi": "Delhi",

    # Goa
    "goa": "Goa",

    # Gujarat
    "gujarat": "Gujarat",

    # Haryana
    "haryana": "Haryana",

    # Himachal Pradesh
    "himachalpradesh": "Himachal Pradesh",

    # Jammu & Kashmir / Ladakh
    "jammuandkashmir": "Jammu and Kashmir",
    "jammukashmir": "Jammu and Kashmir",

    "ladakh": "Ladakh",

    # Jharkhand
    "jharkhand": "Jharkhand",

    # Karnataka
    "karnataka": "Karnataka",

    # Kerala
    "kerala": "Kerala",

    # Madhya Pradesh
    "madhyapradesh": "Madhya Pradesh",

    # Maharashtra
    "maharashtra": "Maharashtra",

    # Manipur
    "manipur": "Manipur",

    # Meghalaya
    "meghalaya": "Meghalaya",

    # Mizoram
    "mizoram": "Mizoram",

    # Nagaland
    "nagaland": "Nagaland",
```

```
    # Odisha (Orissa old name)
    "odisha": "Odisha",
    "orissa": "Odisha",

    # Punjab
    "punjab": "Punjab",

    # Rajasthan
    "rajasthan": "Rajasthan",

    # Sikkim
    "sikkim": "Sikkim",

    # Tamil Nadu
    "tamilnadu": "Tamil Nadu",

    # Telangana
    "telangana": "Telangana",

    # Tripura
    "tripura": "Tripura",

    # Uttar Pradesh
    "uttarpradesh": "Uttar Pradesh",

    # Uttarakhand
    "uttarakhand": "Uttarakhand",

    # West Bengal (ALL variations including typo "Bangal")
    "westbengal": "West Bengal",
    "westbangal": "West Bengal",

    # Andaman & Nicobar Islands
    "andamannicobarislands": "Andaman and Nicobar Islands",
    "andamanandnicobarislands": "Andaman and Nicobar Islands",

    # Chandigarh
    "chandigarh": "Chandigarh",

    # Dadra & Nagar Haveli / Daman & Diu (merged UT)
    "dadraandnagarhaveli": "Dadra and Nagar Haveli and Daman and Diu",
    "damananddiu": "Dadra and Nagar Haveli and Daman and Diu",
    "dadranagarhaveli": "Dadra and Nagar Haveli and Daman and Diu",
    "damandiu": "Dadra and Nagar Haveli and Daman and Diu",
    "dadraandnagarhavelianddamananddiu": "Dadra and Nagar Haveli and Daman a
    "thedadraandnagarhavelianddamananddiu": "Dadra and Nagar Haveli and Dama

    # Lakshadweep
    "lakshadweep": "Lakshadweep",

    # Puducherry
    "pondicherry": "Puducherry",
    "puducherry": "Puducherry",
}
```

In [985…
```
df['state_clean'] = (
    df['state']
    .apply(clean_state_name)
    .map(state_mapping)
)
```

In [986…
```
# Drop invalid entries
df = df[~df['state'].astype(str).str.isnumeric()]

# check unmapped states
unmapped_states = df[df['state_clean'].isnull()]['state'].unique()
print("Unmapped States:", unmapped_states)
```

```
Unmapped States: []
```

In [987…
```
# Counting state unique values after mapping
print("There are ",df['state_clean'].nunique(),"Unique value present in stat
```

```
There are  36 Unique value present in state.
```

In [988…
```
df['state_clean'].value_counts()
```

Out[988…
```
state_clean
Uttar Pradesh                                    110369
Tamil Nadu                                        92552
Maharashtra                                       77191
West Bengal                                       76561
Karnataka                                         70198
Andhra Pradesh                                    65663
Bihar                                             60567
Rajasthan                                         56159
Madhya Pradesh                                    50225
Odisha                                            47011
Gujarat                                           46624
Telangana                                         42774
Kerala                                            39145
Assam                                             31827
Jharkhand                                         23218
Punjab                                            20439
Chhattisgarh                                      18550
Haryana                                           15997
Jammu and Kashmir                                 11455
Himachal Pradesh                                  10346
Uttarakhand                                       10007
Delhi                                              6804
Meghalaya                                          3771
Tripura                                            3729
Manipur                                            3218
Nagaland                                           1999
Puducherry                                         1859
Arunachal Pradesh                                  1601
Goa                                                1527
Mizoram                                            1481
Sikkim                                             1010
Chandigarh                                          859
Dadra and Nagar Haveli and Daman and Diu            416
Andaman and Nicobar Islands                         392
Ladakh                                              304
Lakshadweep                                         159
Name: count, dtype: int64
```

In [989…
```
# checking null values in each column
df.isnull().sum()
```

Out[989…
```
date              0
state             0
district          0
pincode           0
age_0_5           0
age_5_17          0
age_18_greater    0
year              0
month             0
month_name        0
state_clean       0
dtype: int64
```

In [990…
```
# Checking for duplicate rows
df.duplicated().sum()
```

Out[990… np.int64(22956)

In [991…
```
# dropping duplicates rows
df = df.drop_duplicates()
```

In [992…
```
## total enrolment by state
state_summary = df.groupby('state_clean')[[
    'age_0_5', 'age_5_17', 'age_18_greater'
]].sum()

state_summary['total_enrollment'] = (
    state_summary['age_0_5'] +
    state_summary['age_5_17'] +
    state_summary['age_18_greater']
)

top10_states = state_summary.sort_values(
    by='total_enrollment', ascending=False
).head(10)
```

In [993…
```
state_summary
```

Out [993...

state_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Andaman and Nicobar Islands	469	32	0	501
Andhra Pradesh	109394	13414	1465	124273
Arunachal Pradesh	1914	2176	150	4240
Assam	137970	64834	22555	225359
Bihar	254911	327043	11799	593753
Chandigarh	2377	210	33	2620
Chhattisgarh	79653	18158	1962	99773
Dadra and Nagar Haveli and Daman and Diu	1484	248	50	1782
Delhi	67844	21971	3023	92838
Goa	1871	253	156	2280
Gujarat	188709	70270	16063	275042
Haryana	85112	8897	1076	95085
Himachal Pradesh	16081	650	178	16909
Jammu and Kashmir	39314	7802	522	47638
Jharkhand	96048	56152	1412	153612
Karnataka	176178	33402	10038	219618
Kerala	52950	18360	2640	73950
Ladakh	466	133	18	617
Lakshadweep	188	10	1	199
Madhya Pradesh	363244	115172	9476	487892
Maharashtra	274274	81069	8103	363446
Manipur	5044	7895	260	13199
Meghalaya	21072	53089	35078	109239
Mizoram	4044	1259	471	5774
Nagaland	4453	9856	1120	15429
Odisha	97500	22228	726	120454
Puducherry	2746	193	44	2983
Punjab	60481	12175	3117	75773
Rajasthan	224977	110131	5483	340591

state_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Sikkim	1040	1030	105	2175
Tamil Nadu	178294	36214	1202	215710
Telangana	103768	24035	1145	128948
Tripura	7165	3597	246	11008
Uttar Pradesh	511727	473205	17699	1002631
Uttarakhand	31208	5410	338	36956
West Bengal	270419	90335	8495	369249

Data Quality & Consistency Checks

Logical consistency

Output Interpretation

True ✅ → No negative values in that column

False ❌ → At least one negative value exists (data error)

In [994…
```
(df['age_0_5'] >= 0).all()
(df['age_5_17'] >= 0).all()
(df['age_18_greater'] >= 0).all()
```

Out[994… `np.True_`

In [995…
```
(df['age_0_5'] + df['age_5_17'] + df['age_18_greater'] > 0).all()
```

Out[995… `np.True_`

In [996…
```
df['total_enrollment'] = df['age_0_5'] + df['age_5_17'] + df["age_18_greater
```

Distribution Shape Analysis

Skewness measures the asymmetry of a data distribution, while kurtosis measures the tailedness (presence of outliers) in the distribution.

- Skewness = 0 → Distribution is perfectly symmetric
- Skewness > 0 → Distribution is right-skewed (long right tail)
- Skewness < 0 → Distribution is left-skewed (long left tail)
- Kurtosis = 0 → Distribution is normal (mesokurtic)

- Kurtosis > 0 → Distribution has heavy tails (more extreme outliers)
- Kurtosis < 0 → Distribution has light tails (fewer outliers)

In [997…
```
from scipy.stats import skew, kurtosis

skew(df['total_enrollment']), kurtosis(df['total_enrollment'])
```

Out[997…
```
(np.float64(38.72179961381051), np.float64(2273.9610853382146))
```

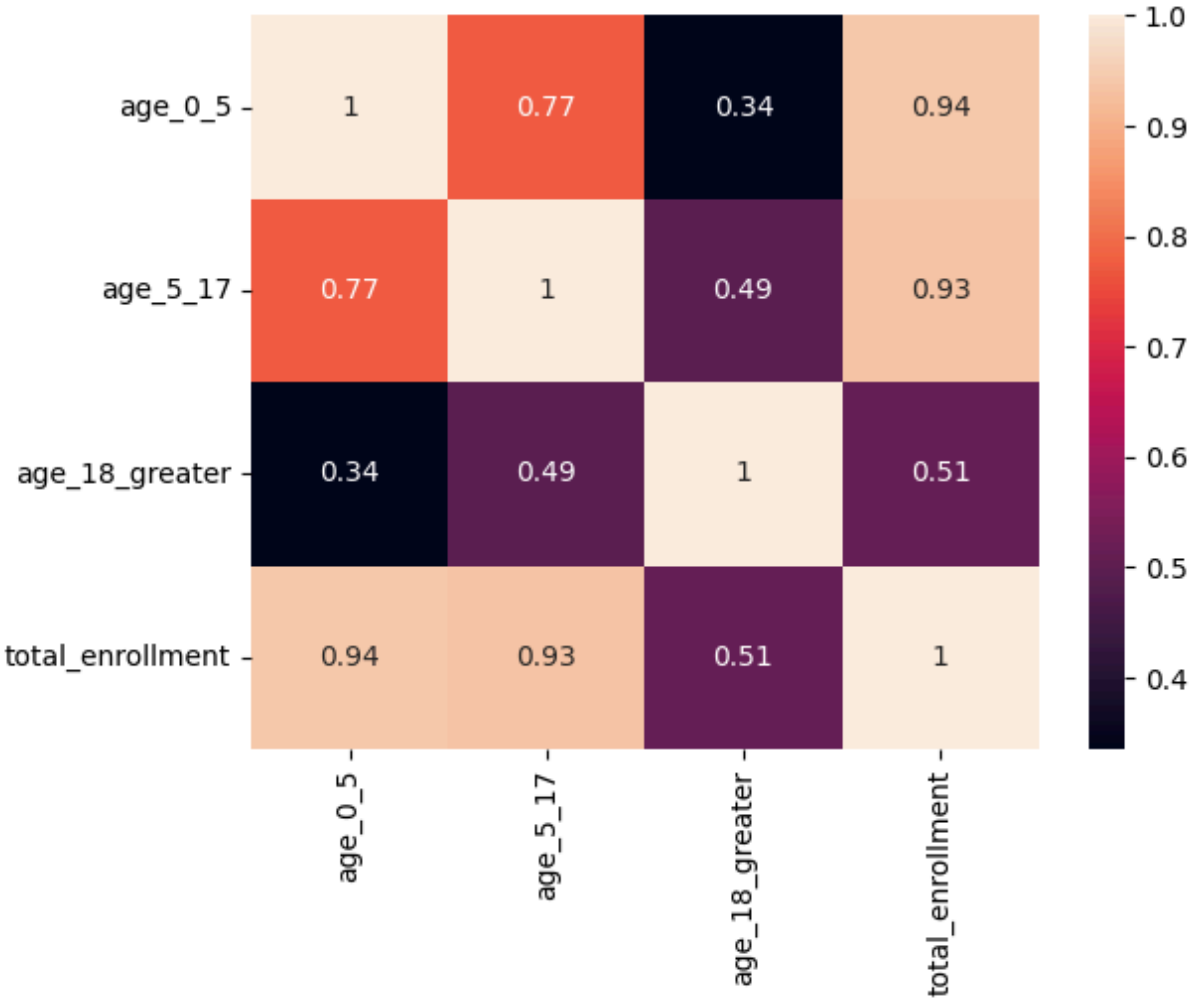
Our data is highly non-normal, with most values clustered at the lower end and a few extreme high values dominating the distribution.

Correlation Analysis

Total enrollment shows a strong positive correlation with all age-group variables, indicating that higher population in any age group contributes directly to higher total enrollment.

In [998…
```
import seaborn as sns
corr = df[['age_0_5','age_5_17','age_18_greater','total_enrollment']].corr()

sns.heatmap(corr, annot=True)
```

Out[998…
```
<Axes: >
```



Normalized Comparisons

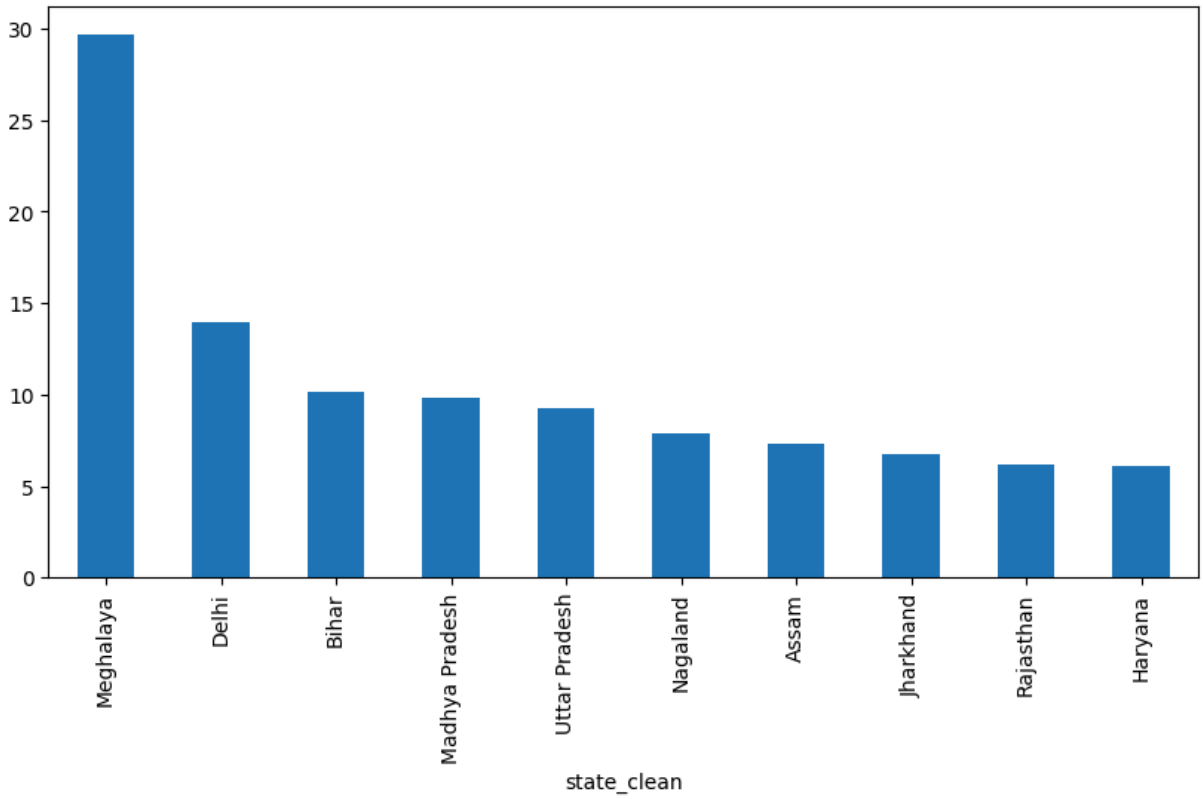
The top-ranked states show higher average enrollment per institution, suggesting the presence of larger schools or denser student populations.

In [999…

```
# Per-capita style normalization (proxy)
state_avg = df.groupby('state_clean')['total_enrollment'].mean()

state_avg.sort_values(ascending=False).head(10).plot(
    kind='bar', figsize=(10,5)
)
```

Out[999…

```
<Axes: xlabel='state_clean'>
```

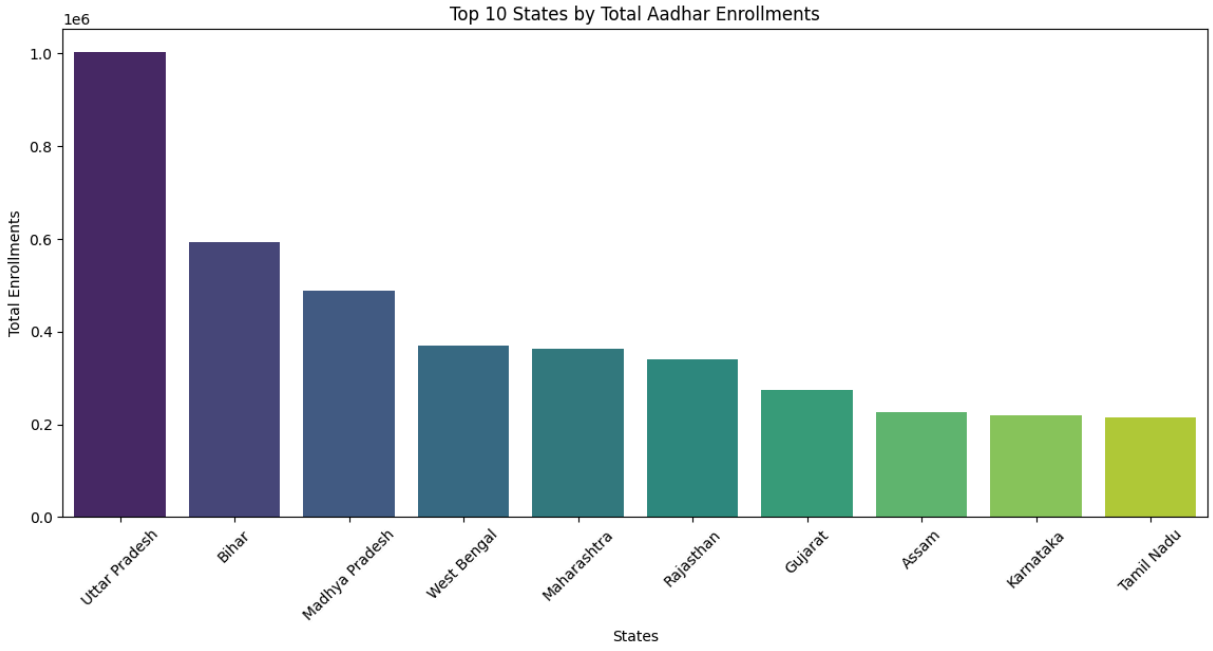


Bar chart → Top 10 states by total enrollment

Top 10 states account for a significant share of total enrollments, indicating population concentration and higher enrollment activity

In [100…

```
## bar plot for top 10 states
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(14,6))
sns.barplot(
    x=top10_states.index,
    y=top10_states['total_enrollment'],
    palette='viridis'
)
plt.title('Top 10 States by Total Aadhar Enrollments')
plt.xlabel('States')
plt.ylabel('Total Enrollments')
plt.xticks(rotation=45)
plt.show()
```



Rank Analysis

In [100…
```
state_rank = state_summary['total_enrollment'].rank(ascending=False)

state_rank.sort_values().head(10)
```

Out[100…
```
state_clean
Uttar Pradesh      1.0
Bihar              2.0
Madhya Pradesh     3.0
West Bengal        4.0
Maharashtra        5.0
Rajasthan          6.0
Gujarat            7.0
Assam              8.0
Karnataka          9.0
Tamil Nadu        10.0
Name: total_enrollment, dtype: float64
```

Variability Analysis

🧠 Meaning:

- High std → inconsistent enrollment
- Low std → stable system

In [100…
```
state_std = df.groupby('state_clean')['total_enrollment'].std()

state_std.sort_values(ascending=False).head(10)
```

Out[100…
```
state_clean
Meghalaya         135.104214
Delhi              77.593888
Uttar Pradesh      56.025312
Bihar              45.115614
Madhya Pradesh     42.377277
Nagaland           41.984775
Haryana            36.049266
Gujarat            34.775802
Assam              33.639722
Manipur            33.200808
Name: total_enrollment, dtype: float64
```

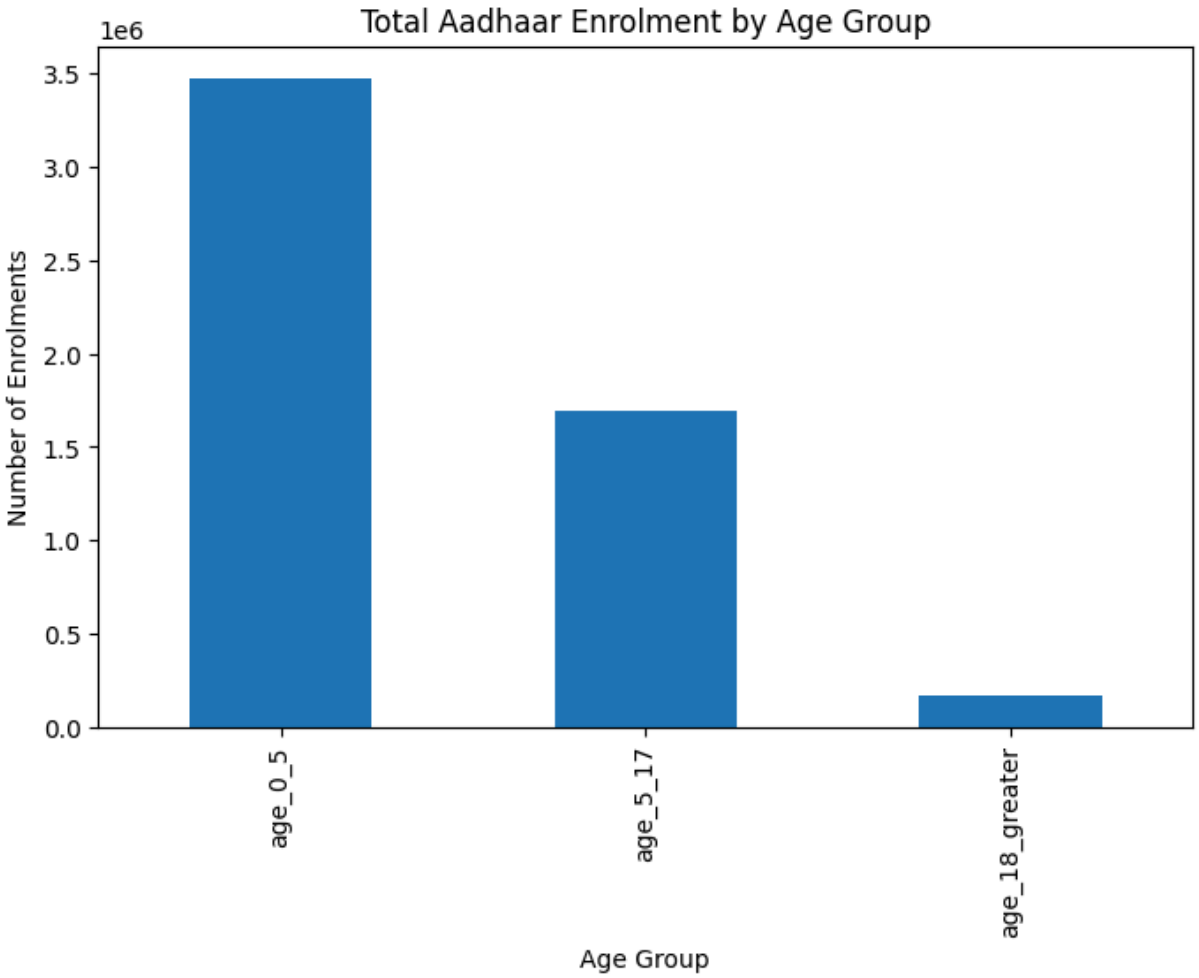
Age Group Distribution

This analysis examines enrolment contribution of individual age groups.

In [100…
```
# Identify age group columns dynamically
age_cols = [c for c in df.columns if c.startswith('age_')]
age_cols
```

Out[100…
```
['age_0_5', 'age_5_17', 'age_18_greater']
```

In [100…
```
age_totals = df[age_cols].sum().sort_values(ascending=False)

plt.figure(figsize=(8,5))
age_totals.plot(kind='bar')
plt.title("Total Aadhaar Enrolment by Age Group")
plt.xlabel("Age Group")
plt.ylabel("Number of Enrolments")
plt.show()
```



Children vs Adult Enrollment (Top 10 States)

In [100…

```
top10_states['children_enrollment'] = (
    top10_states['age_0_5'] +
    top10_states['age_5_17']
)
```

📊 Age Group Enrollment Comparison (Top 10 States)

The bar chart compares enrollment counts between **children** and **adults (18+ age group)** across the top 10 states.

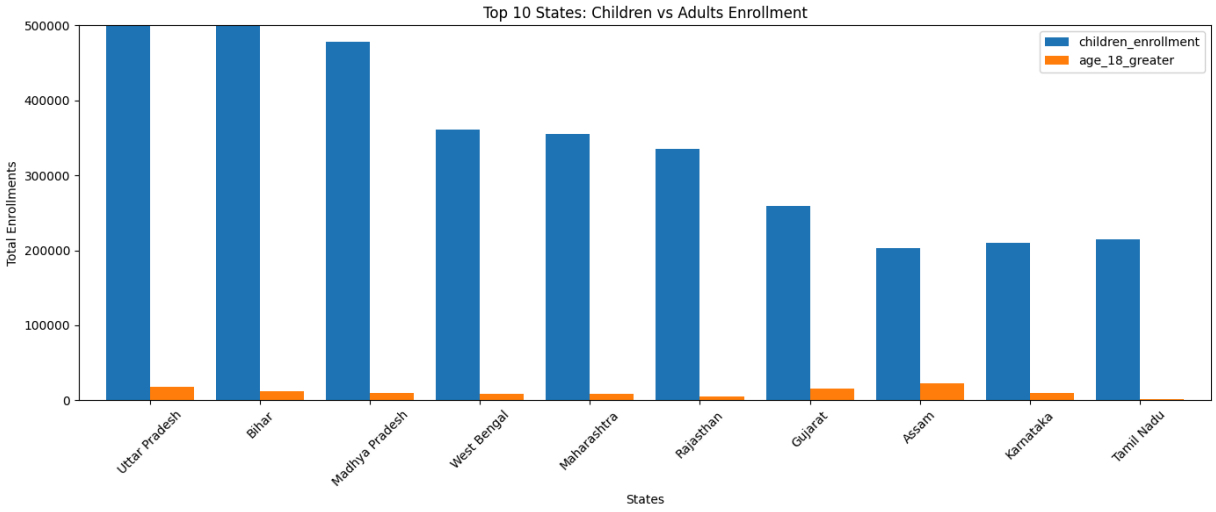
Key Observations:

- Adult (18+) enrollment is consistently higher than children enrollment in most states.
- The gap between adult and children enrollment varies significantly by state.
- States with higher total enrollments show a stronger dominance of the adult age group.

In [100…

```
import matplotlib.pyplot as plt
```

```
ax = top10_states[['children_enrollment', 'age_18_greater']].plot(
    kind='bar',
    figsize=(14,6),
    width=0.8
)

ax.set_title('Top 10 States: Children vs Adults Enrollment')
ax.set_xlabel('States')
ax.set_ylabel('Total Enrollments')

ax.set_ylim(0, 500000)   # 👈 adjust this value as needed

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Age Composition Percentage

States with higher child enrollment percentages may indicate stronger newborn and school-level enrollment outreach.

In [100…

```
top10_states['children_percent'] = (
    top10_states['children_enrollment'] /
    top10_states['total_enrollment']
) * 100

top10_states['adult_percent'] = (
    top10_states['age_18_greater'] /
    top10_states['total_enrollment']
) * 100
```

In [100…

```
top10_states[['children_percent', 'adult_percent']]
```

Out[100…

state_clean	children_percent	adult_percent
Uttar Pradesh	98.234744	1.765256
Bihar	98.012810	1.987190
Madhya Pradesh	98.057767	1.942233
West Bengal	97.699384	2.300616
Maharashtra	97.770508	2.229492
Rajasthan	98.390151	1.609849
Gujarat	94.159801	5.840199
Assam	89.991525	10.008475
Karnataka	95.429336	4.570664
Tamil Nadu	99.442770	0.557230

Plotting Monthly Enrollment trend by age group

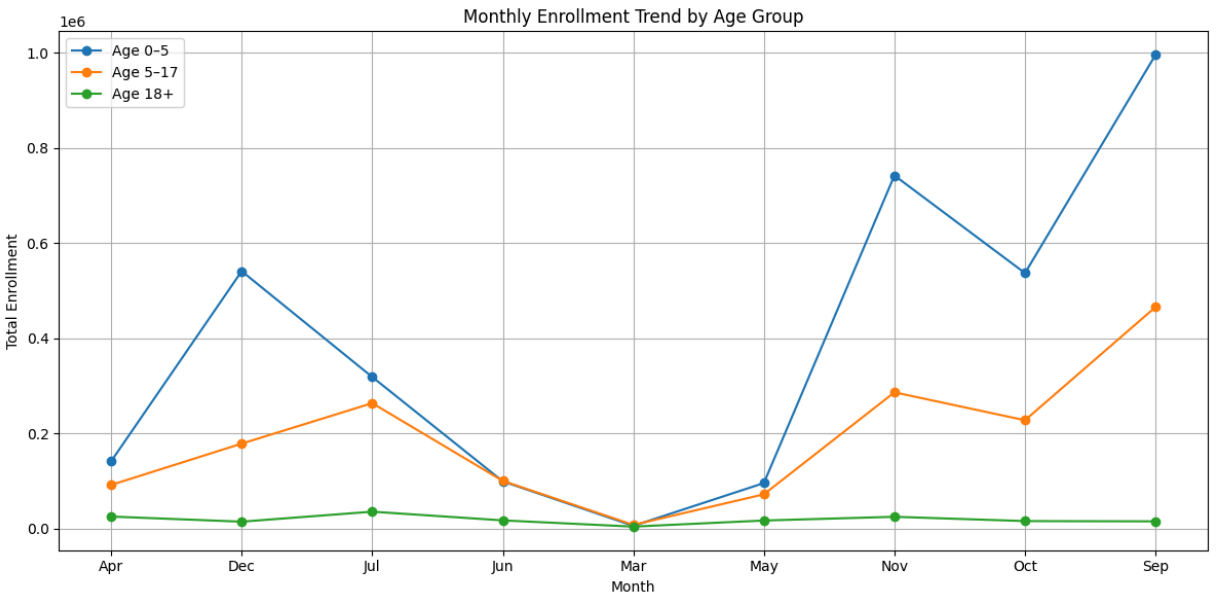
In [100…

```
# # Monthly enrollment trend
monthly_enrollment = df.groupby('month_name')[[
    'age_0_5', 'age_5_17', 'age_18_greater'
]].sum()
```

In [101…

```
plt.figure(figsize=(12,6))

plt.plot(monthly_enrollment.index,
         monthly_enrollment['age_0_5'],
         marker='o',
         label='Age 0–5')

plt.plot(monthly_enrollment.index,
         monthly_enrollment['age_5_17'],
         marker='o',
         label='Age 5–17')

plt.plot(monthly_enrollment.index,
         monthly_enrollment['age_18_greater'],
         marker='o',
         label='Age 18+')

plt.title('Monthly Enrollment Trend by Age Group')
plt.xlabel('Month')
plt.ylabel('Total Enrollment')
plt.legend()
plt.grid(True)
```

```
plt.tight_layout()
plt.show()
```



In [101…
```
df.head()
```

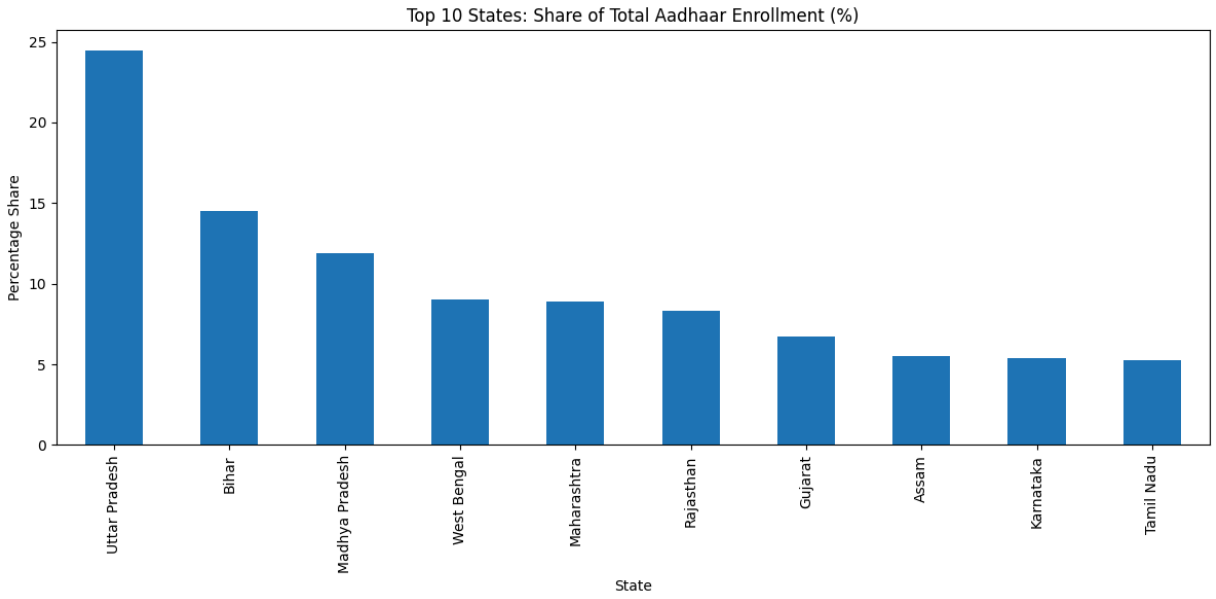
Out[101…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	year
0	2025-03-02	Meghalaya	East Khasi Hills	793121	11	61	37	2025
1	2025-03-09	Karnataka	Bengaluru Urban	560043	14	33	39	2025
2	2025-03-09	Uttar Pradesh	Kanpur Nagar	208001	29	82	12	2025
3	2025-03-09	Uttar Pradesh	Aligarh	202133	62	29	15	2025
4	2025-03-09	Karnataka	Bengaluru Urban	560016	14	16	21	2025

State-wise Enrollment Share (%)

In [101…
```
state_total = df.groupby('state_clean')[['age_0_5','age_5_17','age_18_greate
state_total['total'] = state_total.sum(axis=1)

top10 = state_total.sort_values('total', ascending=False).head(10)

(top10['total'] / top10['total'].sum() * 100).plot(
    kind='bar',
```

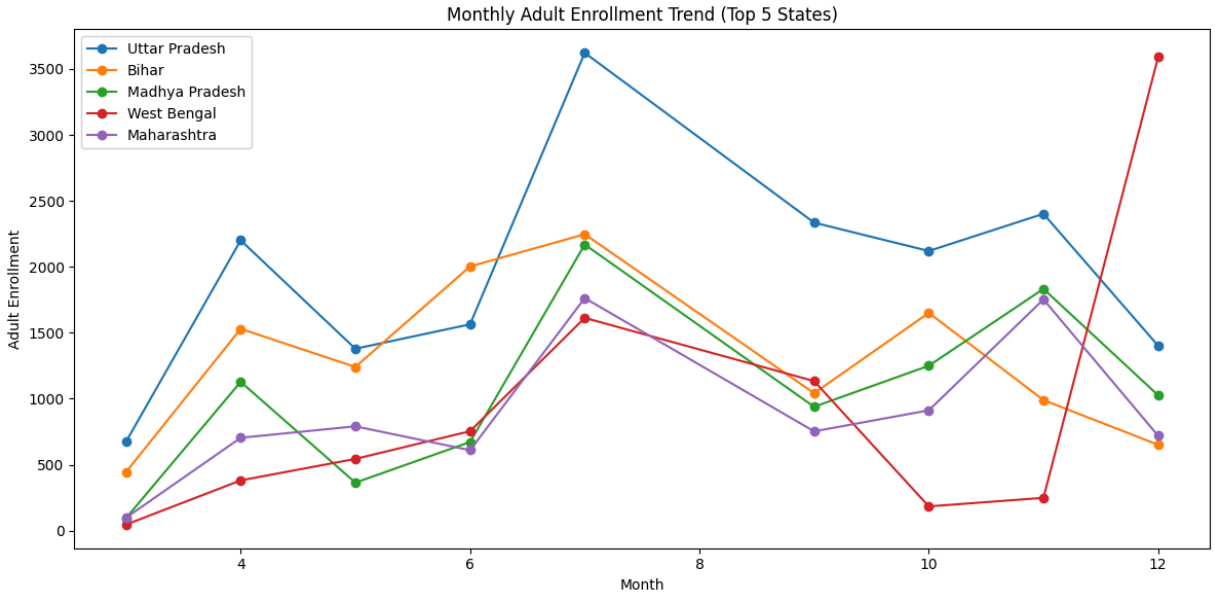
```
    figsize=(12,6)
)

plt.title('Top 10 States: Share of Total Aadhaar Enrollment (%)')
plt.xlabel('State')
plt.ylabel('Percentage Share')
plt.tight_layout()
plt.show()
```



State-wise Monthly Trend (Top 5 States)

In [101…

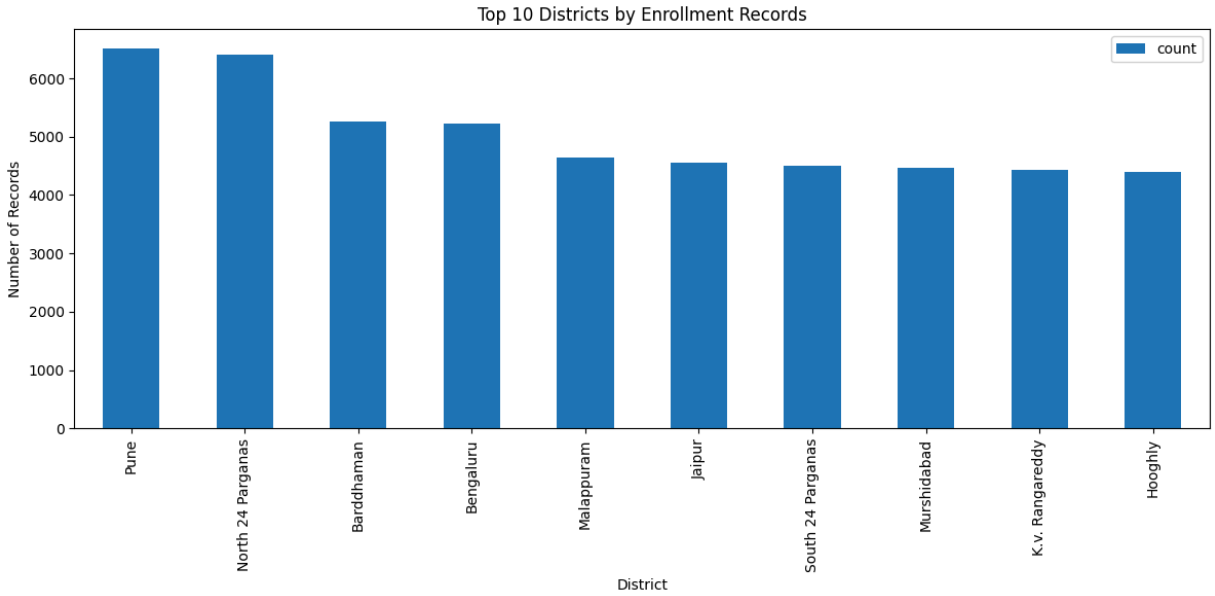
```
top5_states = top10.index[:5]

monthly_state = df[df['state_clean'].isin(top5_states)].groupby(
    ['state_clean','month']
)['age_18_greater'].sum().reset_index()

plt.figure(figsize=(12,6))

for state in top5_states:
    data = monthly_state[monthly_state['state_clean'] == state]
    plt.plot(data['month'], data['age_18_greater'], marker='o', label=state)

plt.title('Monthly Adult Enrollment Trend (Top 5 States)')
plt.xlabel('Month')
plt.ylabel('Adult Enrollment')
plt.legend()
plt.tight_layout()
plt.show()
```



District Concentration (Top Districts in Each State)

In [101…

```
district_count = df.groupby(['state_clean','district']).size().reset_index(r

top_districts = district_count.sort_values('count', ascending=False).head(10

top_districts.plot(
    x='district',
    y='count',
    kind='bar',
    figsize=(12,6)
)

plt.title('Top 10 Districts by Enrollment Records')
plt.xlabel('District')
plt.ylabel('Number of Records')
plt.tight_layout()
plt.show()
```



Performing Eda on Top Five States with heights Enrolment

Bihar

In [101…

```
# Extracting rows where state is Bihar
df_bihar= df[df['state_clean']=='Bihar']
df_bihar
```

Out[101…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	ye
5	2025-03-09	Bihar	Sitamarhi	843331	20	49	12	202
6	2025-03-09	Bihar	Sitamarhi	843330	23	24	42	202
9	2025-03-09	Bihar	Purbi Champaran	845418	30	48	10	202
11	2025-03-09	Bihar	Sitamarhi	843317	35	94	16	202
13	2025-03-09	Bihar	Sitamarhi	843324	49	186	34	202
...	...	...	...	...	...	...	...	
1002992	2025-12-31	Bihar	Vaishali	844134	2	0	0	202
1002993	2025-12-31	Bihar	Vaishali	844504	15	26	1	202
1002994	2025-12-31	Bihar	Vaishali	844509	1	2	0	202
1002995	2025-12-31	Bihar	West Champaran	845404	13	17	1	202
1002996	2025-12-31	Bihar	West Champaran	845449	9	45	0	202

58542 rows × 12 columns

In [101…
```
# Total unique districts in Bihar
df_bihar['district'].nunique()
```

Out[101… 48

In [101…
```
# Same District have more than one spelling so we have to map them
df_bihar['district'].unique()
```

Out[101…
```
array(['Sitamarhi', 'Purbi Champaran', 'Madhubani', 'Bhagalpur', 'Patna',
       'Pashchim Champaran', 'Muzaffarpur', 'Munger', 'Gaya',
       'Kaimur (Bhabua)', 'West Champaran', 'Purnia', 'Saran',
       'East Champaran', 'Vaishali', 'Jehanabad', 'Jamui', 'Gopalganj',
       'Saharsa', 'Arwal', 'Katihar', 'Siwan', 'Lakhisarai', 'Banka',
       'Nalanda', 'Araria', 'Darbhanga', 'Nawada', 'Samastipur',
       'Begusarai', 'Bhojpur', 'Aurangabad', 'Buxar', 'Khagaria',
       'Kishanganj', 'Madhepura', 'Rohtas', 'Sheohar', 'Supaul',
       'Aurangabad(bh)', 'Purba Champaran', 'Purnea', 'Sheikhpura',
       'Sheikpura', 'Bhabua', 'Monghyr', 'Samstipur', 'Aurangabad(BH)'],
      dtype=object)
```

Mapping District

In [101…
```python
import pandas as pd
import re

def clean_name(x):
    if pd.isna(x):
        return x
    x = str(x).lower()
    x = re.sub(r'[^a-z]', '', x)
    x = re.sub(r'\s+',' ', x).strip()
    return x
```

In [101…
```python
## District mapping bihar
bihar_district_mapping = {

    # Arwal
    "arwal": "Arwal",

    # Aurangabad
    "aurangabad": "Aurangabad",
    "aurangabadbh": "Aurangabad",

    # Araria
    "araria": "Araria",

    # Banka
    "banka": "Banka",

    # Begusarai
    "begusarai": "Begusarai",

    # Bhagalpur
    "bhagalpur": "Bhagalpur",

    # Bhojpur
    "bhojpur": "Bhojpur",

    # Buxar
    "buxar": "Buxar",
```

```
    # Darbhanga
    "darbhanga": "Darbhanga",

    # East Champaran
    "eastchamparan": "East Champaran",
    "purbachamparan": "East Champaran",

    # West Champaran
    "westchamparan": "West Champaran",
    "pashchimchamparan": "West Champaran",

    # Gaya
    "gaya": "Gaya",

    # Gopalganj
    "gopalganj": "Gopalganj",

    # Jamui
    "jamui": "Jamui",

    # Jehanabad
    "jehanabad": "Jehanabad",

    # Kaimur
    "kaimurbhabua": "Kaimur",
    "bhabua": "Kaimur",

    # Katihar
    "katihar": "Katihar",

    # Khagaria
    "khagaria": "Khagaria",

    # Kishanganj
    "kishanganj": "Kishanganj",

    # Lakhisarai
    "lakhisarai": "Lakhisarai",

    # Madhepura
    "madhepura": "Madhepura",

    # Madhubani
    "madhubani": "Madhubani",

    # Munger
    "munger": "Munger",
    "monghyr": "Munger",

    # Muzaffarpur
    "muzaffarpur": "Muzaffarpur",

    # Nalanda
    "nalanda": "Nalanda",

    # Nawada
```

```
    "nawada": "Nawada",

    # Patna
    "patna": "Patna",

    # Purnia
    "purnia": "Purnia",
    "purnea": "Purnia",

    # Rohtas
    "rohtas": "Rohtas",

    # Saharsa
    "saharsa": "Saharsa",

    # Samastipur
    "samastipur": "Samastipur",
    "samstipur": "Samastipur",

    # Saran
    "saran": "Saran",

    # Sheikhpura
    "sheikhpura": "Sheikhpura",
    "sheikpura": "Sheikhpura",

    # Sheohar
    "sheohar": "Sheohar",

    # Sitamarhi
    "sitamarhi": "Sitamarhi",

    # Siwan
    "siwan": "Siwan",

    # Supaul
    "supaul": "Supaul",

    # Vaishali
    "vaishali": "Vaishali",
}
```

In [102…

```
df['district_clean'] = (
    df['district']
    .apply(clean_name)
    .map(bihar_district_mapping)
    .fillna(df_bihar['district'])

)
```

In [102…

```
## Remaining unmapped
df[df['district_clean'].isna()]['district'].unique()
# count check
df['district_clean'].nunique()
```

Out[102… 39

In [102…
```
df_bihar = df[df['state_clean']=='Bihar']
df_bihar
```

Out[102…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	yea
5	2025-03-09	Bihar	Sitamarhi	843331	20	49	12	202
6	2025-03-09	Bihar	Sitamarhi	843330	23	24	42	202
9	2025-03-09	Bihar	Purbi Champaran	845418	30	48	10	202
11	2025-03-09	Bihar	Sitamarhi	843317	35	94	16	202
13	2025-03-09	Bihar	Sitamarhi	843324	49	186	34	202
...	...	...	...	...	...	...	...	
1002992	2025-12-31	Bihar	Vaishali	844134	2	0	0	202
1002993	2025-12-31	Bihar	Vaishali	844504	15	26	1	202
1002994	2025-12-31	Bihar	Vaishali	844509	1	2	0	202
1002995	2025-12-31	Bihar	West Champaran	845404	13	17	1	202
1002996	2025-12-31	Bihar	West Champaran	845449	9	45	0	202

58542 rows × 13 columns

In [102…
```
# Check Bihar-specific unmapped districts
df_bihar_unmapped = df_bihar[df_bihar['district_clean'].isna()]
print(f"Unmapped Bihar districts count: {len(df_bihar_unmapped)}")
df_bihar_unmapped['district'].unique()
```

```
Unmapped Bihar districts count: 0
```

Out[102… `array([], dtype=object)`

In [102…
```
# Final unique districts in Bihar after cleaning
df_bihar['district_clean'].unique()
```

Out[102…
```
array(['Sitamarhi', 'Purbi Champaran', 'Madhubani', 'Bhagalpur', 'Patna',
       'West Champaran', 'Muzaffarpur', 'Munger', 'Gaya', 'Kaimur',
       'Purnia', 'Saran', 'East Champaran', 'Vaishali', 'Jehanabad',
       'Jamui', 'Gopalganj', 'Saharsa', 'Arwal', 'Katihar', 'Siwan',
       'Lakhisarai', 'Banka', 'Nalanda', 'Araria', 'Darbhanga', 'Nawada',
       'Samastipur', 'Begusarai', 'Bhojpur', 'Aurangabad', 'Buxar',
       'Khagaria', 'Kishanganj', 'Madhepura', 'Rohtas', 'Sheohar',
       'Supaul', 'Sheikhpura'], dtype=object)
```

In [102…
```
# Total unique districts in Bihar after cleaning
df_bihar['district_clean'].nunique()
```

Out[102…
```
39
```

In [102…
```
# unique pincodes in bihar
df_bihar['pincode'].nunique()
```

Out[102…
```
906
```

In [102…
```
# Check unique pincodes per district in Bihar
pincode_check = df_bihar.groupby('district_clean')['pincode'].nunique().rese
pincode_check
```

Out[102…

	district_clean	unique_pincodes
0	Araria	19
1	Arwal	19
2	Aurangabad	29
3	Banka	32
4	Begusarai	33
5	Bhagalpur	34
6	Bhojpur	41
7	Buxar	27
8	Darbhanga	46
9	East Champaran	39
10	Gaya	39
11	Gopalganj	23
12	Jamui	14
13	Jehanabad	21
14	Kaimur	12
15	Katihar	23
16	Khagaria	15
17	Kishanganj	9
18	Lakhisarai	13
19	Madhepura	21
20	Madhubani	44
21	Munger	12
22	Muzaffarpur	53
23	Nalanda	31
24	Nawada	24
25	Patna	69
26	Purbi Champaran	12
27	Purnia	30
28	Rohtas	33
29	Saharsa	18
30	Samastipur	42
31	Saran	51

	district_clean	unique_pincodes
32	Sheikhpura	8
33	Sheohar	7
34	Sitamarhi	25
35	Siwan	46
36	Supaul	23
37	Vaishali	38
38	West Champaran	19

In [102…
```
# Pincode associated with multiple districts in Bihar
pin_district_count = (
    df_bihar.groupby('pincode')['district_clean']
    .nunique()
    .reset_index(name='district_count')
)
```

In [102…
```
pin_district_count
```

Out[102…

	pincode	district_count
0	800001	1
1	800002	1
2	800003	1
3	800004	1
4	800005	1
…	…	…
901	855114	1
902	855115	2
903	855116	1
904	855117	1
905	855456	1

906 rows × 2 columns

In [103…
```
# Extract problematic pincodes (associated with >1 district)
problem_pins = pin_district_count[
    pin_district_count['district_count'] > 1
]
```

In [103…
```
problem_pins
```

Out[103…

	pincode	district_count
40	801304	2
41	801305	2
53	802112	2
73	802134	2
83	802160	2
…	…	…
890	854337	2
894	855101	3
896	855105	2
898	855107	2
902	855115	2

177 rows × 2 columns

In [103…

```
# Get all records with problematic pincodes(flagged records)
df_flagged = df_bihar.merge(
    problem_pins[['pincode']],
    on='pincode',
    how='inner'
)
```

In [103…

```
df_flagged
```

Out [103…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	year
0	2025-03-09	Bihar	Purbi Champaran	845418	30	48	10	2025
1	2025-03-09	Bihar	Purbi Champaran	845304	18	72	12	2025
2	2025-03-15	Bihar	Purbi Champaran	845303	12	121	13	2025
3	2025-04-01	Bihar	Sitamarhi	843315	102	125	18	2025
4	2025-04-01	Bihar	Munger	811213	191	278	22	2025
...	...	...	...	...	...	...	...	...
16363	2025-12-31	Bihar	Sheohar	843325	4	2	0	2025
16364	2025-12-31	Bihar	Sitamarhi	843325	11	6	0	2025
16365	2025-12-31	Bihar	Siwan	841243	2	11	0	2025
16366	2025-12-31	Bihar	Supaul	852108	0	9	0	2025
16367	2025-12-31	Bihar	Supaul	852131	15	19	0	2025

16368 rows × 13 columns

In [103…

```
# Summary of flagged records by district and pincode(for review)
flagged_pincode=df_flagged.groupby(['district_clean','pincode'])[['age_0_5',
# flagged_pincode.to_excel('flagged_pincode_domain.xlsx')
```

In [103…

```
# Add total enrollment column to flagged_pincode
flagged_pincode['total_enrollment']=flagged_pincode['age_0_5']+flagged_pincc
flagged_pincode
```

Out[103…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
0	Araria	854102	127	84	1	212
1	Araria	854201	28	30	0	58
2	Araria	854202	57	73	0	130
3	Araria	854304	410	412	14	836
4	Araria	854312	831	540	6	1377
...	...	...	...	...	...	...
360	Vaishali	843104	60	34	0	94
361	Vaishali	843105	7	2	0	9
362	Vaishali	844111	143	183	0	326
363	Vaishali	844112	177	185	1	363
364	Vaishali	844120	22	30	0	52

365 rows × 6 columns

In [103…

```
idx = flagged_pincode.groupby('pincode')['total_enrollment'].idxmax()
df_filtered = flagged_pincode.loc[idx]
df_filtered
```

Out[103…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
223	Nalanda	801304	57	114	0	171
224	Nalanda	801305	38	62	1	101
63	Buxar	802112	114	212	1	327
64	Buxar	802134	117	357	1	475
59	Bhojpur	802160	66	268	2	336
...	...	...	...	...	...	...
281	Purnia	854337	323	127	0	450
165	Kishanganj	855101	1745	334	5	2084
158	Katihar	855105	361	120	6	487
166	Kishanganj	855107	1576	406	1	1983
167	Kishanganj	855115	735	198	4	937

177 rows × 6 columns

In [103…

```
# Creating a flag for pincodes associated with multiple districts
df_bihar['pin_multi_district_flag']=(
    df_bihar.groupby('pincode')['district_clean']
```

```
    .transform('nunique')>1
)
```

In [103…
```
# Creating a flag for pincodes associated with multiple districts
pin_district_map= (
    df_bihar[df_bihar['pin_multi_district_flag']]
    .groupby('pincode')['district_clean']  # noqa: SC100
    .unique()
    .reset_index()
)
```

In [103…
```
## monthly enrolment check
df_bihar['month'] = df_bihar['date'].dt.month.astype(str).str.zfill(2)
df_bihar
```

Out[103…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	ye
5	2025-03-09	Bihar	Sitamarhi	843331	20	49	12	202
6	2025-03-09	Bihar	Sitamarhi	843330	23	24	42	202
9	2025-03-09	Bihar	Purbi Champaran	845418	30	48	10	202
11	2025-03-09	Bihar	Sitamarhi	843317	35	94	16	202
13	2025-03-09	Bihar	Sitamarhi	843324	49	186	34	202
...	...	...	...	...	...	...	...	
1002992	2025-12-31	Bihar	Vaishali	844134	2	0	0	202
1002993	2025-12-31	Bihar	Vaishali	844504	15	26	1	202
1002994	2025-12-31	Bihar	Vaishali	844509	1	2	0	202
1002995	2025-12-31	Bihar	West Champaran	845404	13	17	1	202
1002996	2025-12-31	Bihar	West Champaran	845449	9	45	0	202

58542 rows × 14 columns

In [104…
```
# Dropping Date District and state as we have District clean State clean
df_bihar_cleaned=df_bihar.drop(columns=['date','district','state'], axis=1)
```

```
df_bihar_cleaned
```

Out[104…

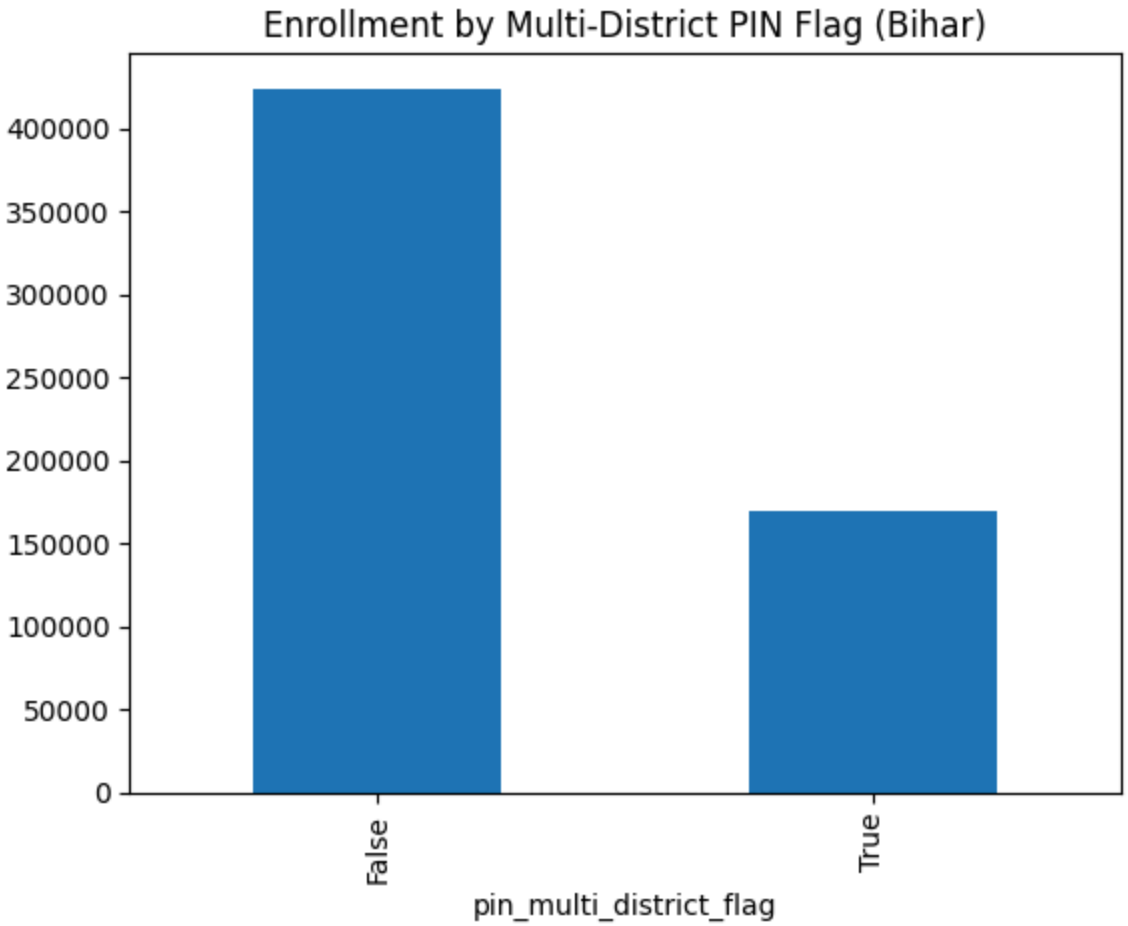
	pincode	age_0_5	age_5_17	age_18_greater	year	month	month_name	s
5	843331	20	49	12	2025	03	Mar	
6	843330	23	24	42	2025	03	Mar	
9	845418	30	48	10	2025	03	Mar	
11	843317	35	94	16	2025	03	Mar	
13	843324	49	186	34	2025	03	Mar	
…	…	…	…	…	…	…	…	…
1002992	844134	2	0	0	2025	12	Dec	
1002993	844504	15	26	1	2025	12	Dec	
1002994	844509	1	2	0	2025	12	Dec	
1002995	845404	13	17	1	2025	12	Dec	
1002996	845449	9	45	0	2025	12	Dec	

58542 rows × 11 columns

Flagged vs unflagged pincodes

In [104…

```
df_bihar.groupby('pin_multi_district_flag')['total_enrollment'].sum().plot(
    kind='bar'
)
plt.title('Enrollment by Multi-District PIN Flag (Bihar)')
plt.show()
```



In [104…
```
# Aggregate Bihar enrollment data at the district level by summing enrollmen
# across different age groups (0–5, 5–17, and 18+).
df_bihar_dist_level = df_bihar_cleaned.groupby('district_clean')[
    ['age_0_5', 'age_5_17', 'age_18_greater']
].sum()

# Compute total enrollment for each district by adding all age-group enrollm
df_bihar_dist_level['total_enrollment'] = (
    df_bihar_dist_level['age_0_5'] +
    df_bihar_dist_level['age_5_17'] +
    df_bihar_dist_level['age_18_greater']
)
```

In [104…
```
df_bihar_dist_level.shape
```

Out[104… (39, 4)

In [104…
```
df_bihar_dist_level
```

Out[104...

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Araria	9646	5357	124	15127
Arwal	777	2939	33	3749
Aurangabad	2922	6927	72	9921
Banka	5945	4669	79	10693
Begusarai	7719	4851	55	12625
Bhagalpur	10196	10025	375	20596
Bhojpur	3043	9832	104	12979
Buxar	2407	4453	60	6920
Darbhanga	9633	5466	68	15167
East Champaran	10003	18105	792	28900
Gaya	6540	19786	434	26760
Gopalganj	5195	9461	192	14848
Jamui	5042	4897	159	10098
Jehanabad	888	3683	139	4710
Kaimur	2938	4631	84	7653
Katihar	10877	4757	113	15747
Khagaria	4167	3653	67	7887
Kishanganj	6527	1680	23	8230
Lakhisarai	2599	3364	68	6031
Madhepura	4371	4225	38	8634
Madhubani	12324	12275	791	25390
Munger	2397	3466	102	5965
Muzaffarpur	13787	13854	657	28298
Nalanda	3527	10452	134	14113
Nawada	2661	12099	288	15048
Patna	6559	16758	744	24061
Purbi Champaran	4000	10071	800	14871
Purnia	11978	7687	183	19848
Rohtas	2623	4974	76	7673
Saharsa	6054	6335	201	12590
Samastipur	10877	7571	131	18579

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Saran	7401	16040	217	23658
Sheikhpura	1191	1863	19	3073
Sheohar	1900	1119	11	3030
Sitamarhi	20358	18600	2694	41652
Siwan	5961	8778	135	14874
Supaul	5473	4408	155	10036
Vaishali	7972	9424	52	17448
West Champaran	16433	28508	1330	46271

In [104…

```
# Sorting District with total enrollment
df_bihar_dist_level.sort_values("total_enrollment",ascending=False).reset_ir
# df_bihar_dist_level.to_excel('bihar_district_level_enrolment.xlsx')
```

Out[104...

	district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
0	West Champaran	16433	28508	1330	46271
1	Sitamarhi	20358	18600	2694	41652
2	East Champaran	10003	18105	792	28900
3	Muzaffarpur	13787	13854	657	28298
4	Gaya	6540	19786	434	26760
5	Madhubani	12324	12275	791	25390
6	Patna	6559	16758	744	24061
7	Saran	7401	16040	217	23658
8	Bhagalpur	10196	10025	375	20596
9	Purnia	11978	7687	183	19848
10	Samastipur	10877	7571	131	18579
11	Vaishali	7972	9424	52	17448
12	Katihar	10877	4757	113	15747
13	Darbhanga	9633	5466	68	15167
14	Araria	9646	5357	124	15127
15	Nawada	2661	12099	288	15048
16	Siwan	5961	8778	135	14874
17	Purbi Champaran	4000	10071	800	14871
18	Gopalganj	5195	9461	192	14848
19	Nalanda	3527	10452	134	14113
20	Bhojpur	3043	9832	104	12979
21	Begusarai	7719	4851	55	12625
22	Saharsa	6054	6335	201	12590
23	Banka	5945	4669	79	10693
24	Jamui	5042	4897	159	10098
25	Supaul	5473	4408	155	10036
26	Aurangabad	2922	6927	72	9921
27	Madhepura	4371	4225	38	8634
28	Kishanganj	6527	1680	23	8230
29	Khagaria	4167	3653	67	7887
30	Rohtas	2623	4974	76	7673
31	Kaimur	2938	4631	84	7653

	district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
32	Buxar	2407	4453	60	6920
33	Lakhisarai	2599	3364	68	6031
34	Munger	2397	3466	102	5965
35	Jehanabad	888	3683	139	4710
36	Arwal	777	2939	33	3749
37	Sheikhpura	1191	1863	19	3073
38	Sheohar	1900	1119	11	3030

Plotting Top 10 Districts by number of Enrollment

Visualize age-wise enrollment distribution across top 10 Bihar districts using a line plot

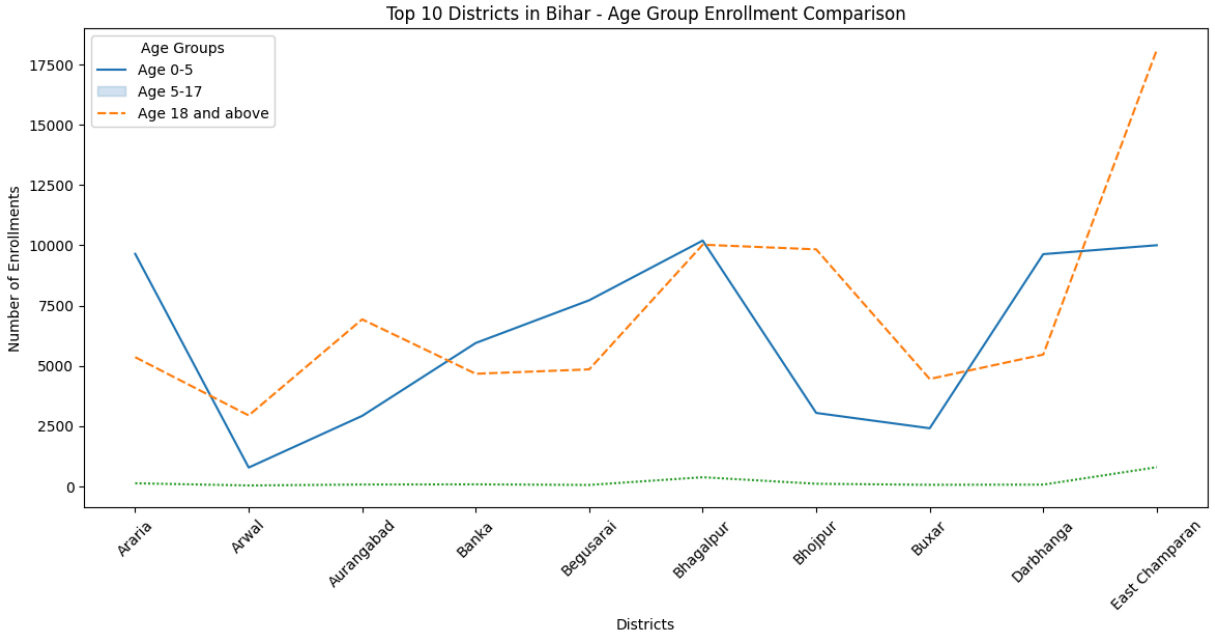
In [104…

```
df_bihar_dist_level1 = df_bihar_dist_level.head(10)
```

In [104…

```
# Import required visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Set the figure size for better readability
plt.figure(figsize=(14,6))

# Plot line chart to compare enrollment trends across age groups
# for the top 10 districts in Bihar
sns.lineplot(
    data=df_bihar_dist_level1[['age_0_5', 'age_5_17', 'age_18_greater']]
)

# Add title and axis labels
plt.title('Top 10 Districts in Bihar - Age Group Enrollment Comparison')
plt.xlabel('Districts')
plt.ylabel('Number of Enrollments')

# Customize legend to clearly represent age groups
plt.legend(
    title='Age Groups',
    labels=['Age 0-5', 'Age 5-17', 'Age 18 and above']
)

# Set district names on x-axis and rotate labels for clarity
plt.xticks(
    ticks=range(len(df_bihar_dist_level1.index)),
    labels=df_bihar_dist_level1.index,
    rotation=45
```

```
)

# Display the plot
plt.show()
```



Plotting month vs total enrollment

In [104…
```
## monthly enrolment trend in bihar
df_bihar_monthly = df_bihar_cleaned.groupby('month')[['age_0_5','age_5_17','
df_bihar_monthly['total_enrollment'] = df_bihar_monthly['age_0_5'] + df_biha
```

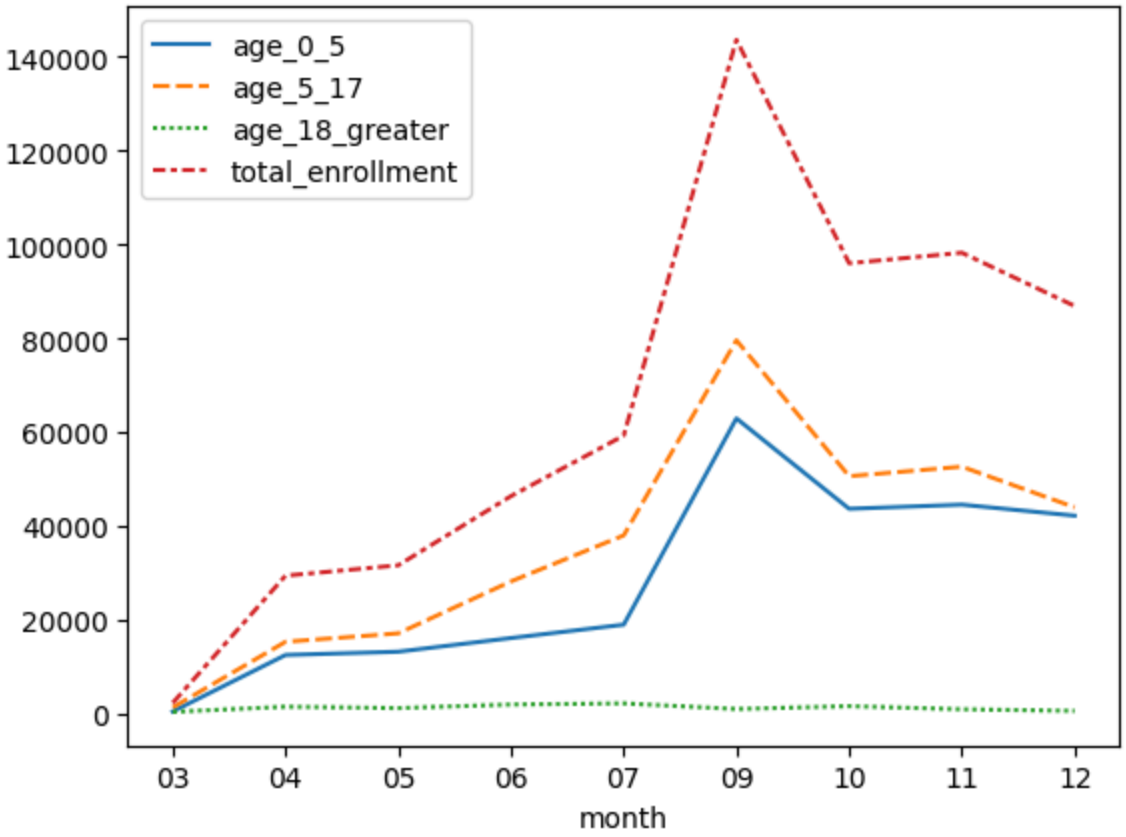
In [104…
```
df_bihar_monthly.sort_values("total_enrollment",ascending=False).reset_index
```

Out[104…

	month	age_0_5	age_5_17	age_18_greater	total_enrollment
0	09	62940	79583	1042	143565
1	11	44572	52661	990	98223
2	10	43690	50636	1651	95977
3	12	42223	44009	651	86883
4	07	19008	38019	2247	59274
5	06	16160	28223	2003	46386
6	05	13251	17159	1241	31651
7	04	12551	15361	1530	29442
8	03	516	1392	444	2352

In [105…
```
sns.lineplot(data=df_bihar_monthly)
```

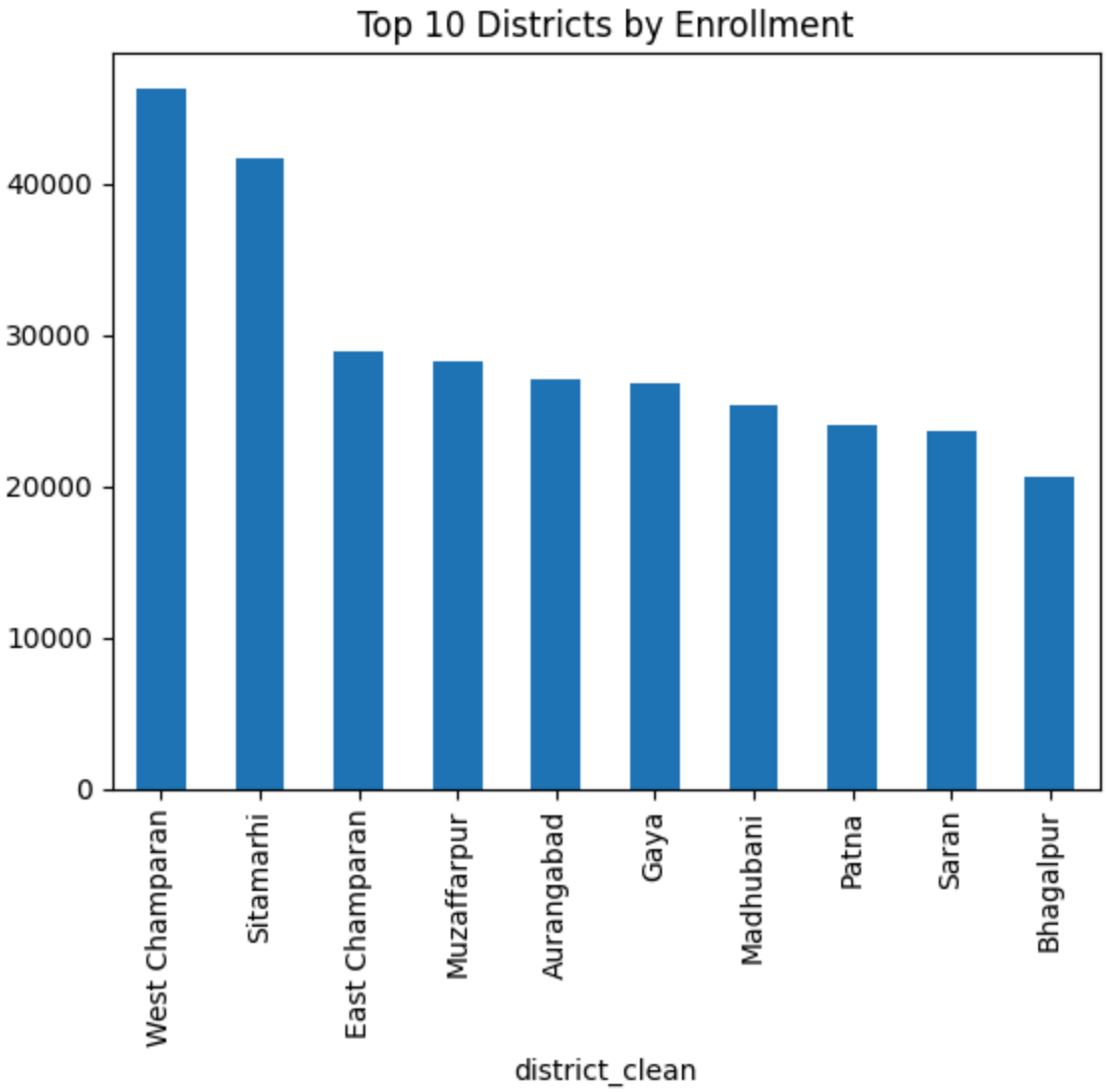
Out[105… <Axes: xlabel='month'>



Top 10 Districts by Enrollment

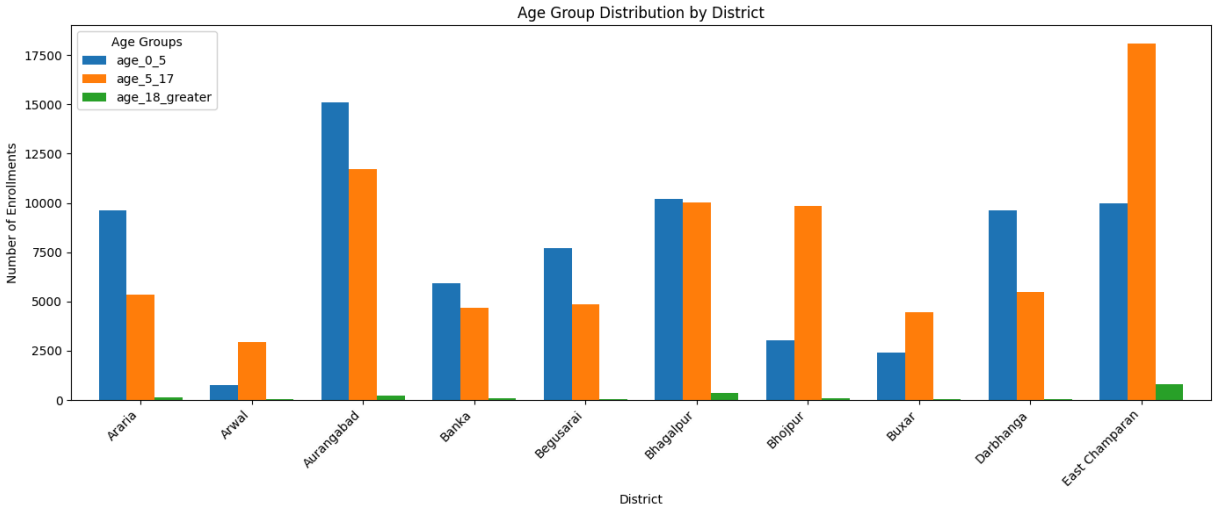
In [105…

```
top_districts = df.groupby('district_clean')['total_enrollment'] \
                  .sum().sort_values(ascending=False).head(10)

top_districts.plot(kind='bar')
plt.title('Top 10 Districts by Enrollment')
plt.show()
```



In [105…

```python
import matplotlib.pyplot as plt

district_age = df.groupby('district_clean')[
    ['age_0_5','age_5_17','age_18_greater']
].sum().head(10)

ax = district_age.plot(
    kind='bar',
    stacked=False,      # important → no overlap
    figsize=(14,6),
    width=0.75
)

plt.title('Age Group Distribution by District')
plt.xlabel('District')
plt.ylabel('Number of Enrollments')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Age Groups')
plt.tight_layout()
plt.show()
```

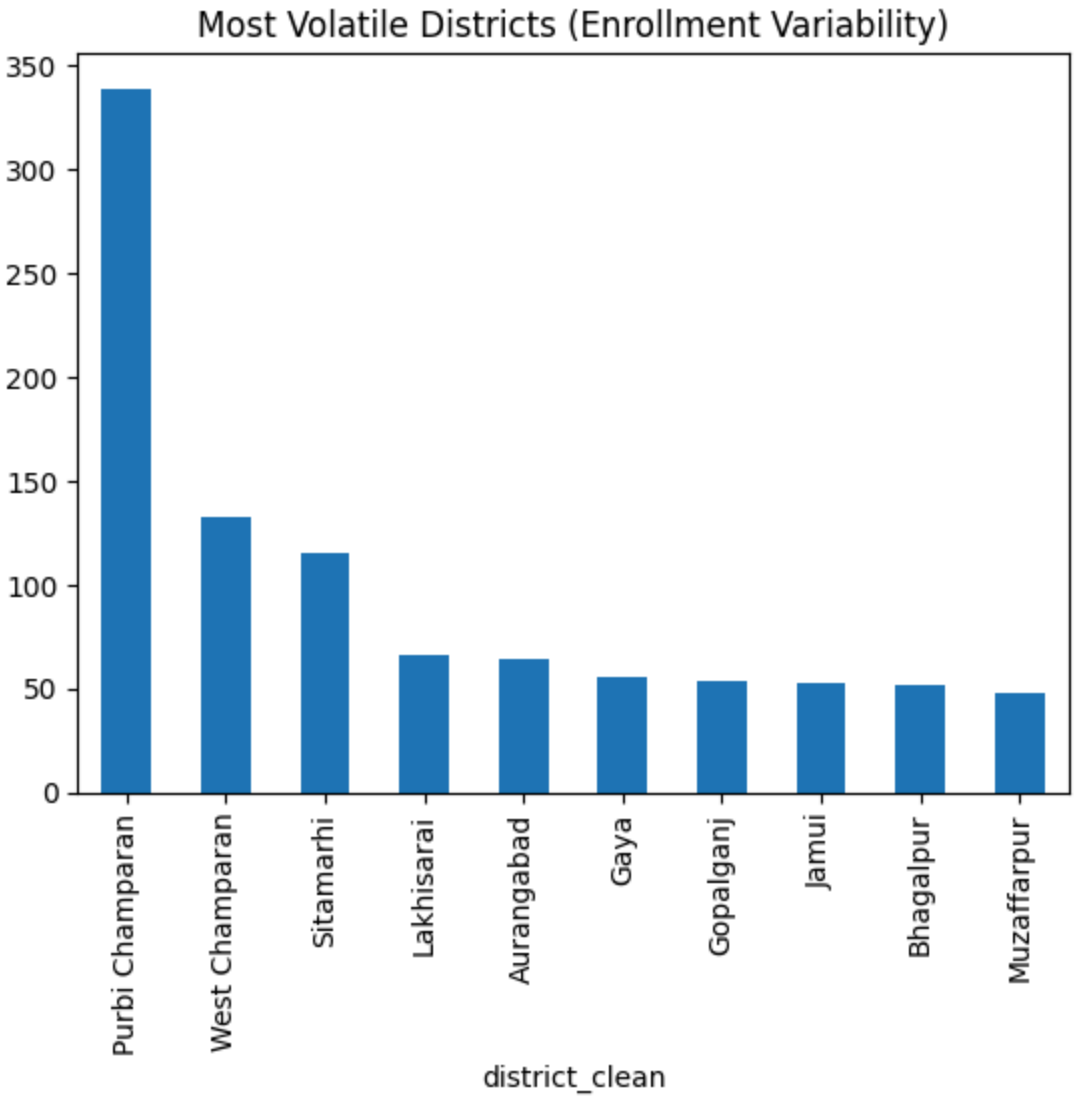


📊 Enrollment Variability Across Districts

This bar chart highlights the top 10 districts with the highest standard deviation in total enrollment, indicating significant variability over time. High volatility suggests inconsistent enrollment patterns, possibly due to seasonal drives, migration, or administrative factors.

In [105…

```
district_std = df.groupby('district_clean')['total_enrollment'].std()

district_std.sort_values(ascending=False).head(10).plot(kind='bar')
plt.title('Most Volatile Districts (Enrollment Variability)')
plt.show()
```



📌 District-wise Enrollment Ranking Analysis

This analysis ranks districts based on their **total Aadhaar enrollment** over the entire study period.

In [105…

```
district_rank = df.groupby('district_clean')['total_enrollment'].sum() \
                  .rank(ascending=False)

district_rank.sort_values().head(10)
```

Out[105…

```
district_clean
West Champaran     1.0
Sitamarhi          2.0
East Champaran     3.0
Muzaffarpur        4.0
Aurangabad         5.0
Gaya               6.0
Madhubani          7.0
Patna              8.0
Saran              9.0
Bhagalpur         10.0
Name: total_enrollment, dtype: float64
```

We performed univariate, bivariate, and multivariate analysis to study enrollment distribution across states, districts, and age groups. Advanced analysis included district-level volatility, age composition, time-based trends, and flag-based segmentation to uncover hidden patterns.

Uttar Pradesh

In [105…

```
# Extracting rows where state is Uttar Pradesh
df_uttarpradesh= df[(df['state_clean']=='Uttar Pradesh')]
df_uttarpradesh
```

Out[105…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
2	2025-03-09	Uttar Pradesh	Kanpur Nagar	208001	29	82	12
3	2025-03-09	Uttar Pradesh	Aligarh	202133	62	29	15
7	2025-03-09	Uttar Pradesh	Bahraich	271865	26	60	14
8	2025-03-09	Uttar Pradesh	Firozabad	283204	28	26	10
10	2025-03-09	Uttar Pradesh	Maharajganj	273164	31	70	13
...	...	...	...	...	...	...	...
1005743	2025-12-31	Uttar Pradesh	Varanasi	221002	10	18	0
1005744	2025-12-31	Uttar Pradesh	Varanasi	221104	4	11	0
1005745	2025-12-31	Uttar Pradesh	Varanasi	221107	1	15	0
1005746	2025-12-31	Uttar Pradesh	Varanasi	221207	1	9	0
1005747	2025-12-31	Uttar Pradesh	Varanasi	221313	1	0	0

108066 rows × 13 columns

In [105…
```
# Total unique districts in Uttar Pradesh
df_uttarpradesh['district'].nunique()
```

Out[105… 89

In [105…
```
# # Same District have more than one spelling so we have to map them
df_uttarpradesh['district'].unique()
```

Out[105…
```
array(['Kanpur Nagar', 'Aligarh', 'Bahraich', 'Firozabad', 'Maharajganj',
       'Ghaziabad', 'Gautam Buddha Nagar', 'Lucknow', 'Agra', 'Unnao',
       'Saharanpur', 'Jaunpur', 'Gorakhpur', 'Bulandshahr', 'Mathura',
       'Banda', 'Kheri', 'Budaun', 'Kanpur Dehat', 'Varanasi', 'Baghpat',
       'Fatehpur', 'Etawah', 'Shamli', 'Balrampur', 'Bara Banki',
       'Shahjahanpur', 'Gonda', 'Bareilly', 'Sitapur', 'Sultanpur',
       'Shrawasti', 'Chandauli', 'Mainpuri', 'Muzaffarnagar',
       'Siddharthnagar', 'Ambedkar Nagar', 'Pilibhit', 'Kaushambi',
       'Jalaun', 'Etah', 'Meerut', 'Basti', 'Azamgarh', 'Rampur',
       'Moradabad', 'Amroha', 'Bijnor', 'Deoria', 'Prayagraj', 'Lalitpur',
       'Hapur', 'Hathras', 'Kushinagar', 'Shravasti', 'Farrukhabad',
       'Hardoi', 'Ayodhya', 'Siddharth Nagar', 'Barabanki', 'Sambhal',
       'Jhansi', 'Kasganj', 'Kannauj', 'Kushi Nagar', 'Allahabad',
       'Amethi', 'Auraiya', 'Ballia', 'Bhadohi', 'Chitrakoot', 'Faizabad',
       'Ghazipur', 'Mahoba', 'Mau', 'Mirzapur', 'Pratapgarh',
       'Rae Bareli', 'Sant Kabir Nagar', 'Sant Ravidas Nagar',
       'Sonbhadra', 'Bulandshahar', 'Hamirpur', 'Jyotiba Phule Nagar',
       'Sant Ravidas Nagar Bhadohi', 'Raebareli', 'Mahrajganj',
       'Kushinagar *', 'Bagpat'], dtype=object)
```

Mapping District

In [105…
```
df_uttarpradesh['district_clean'] = (
    df_uttarpradesh['district']
    .str.lower()
    .str.strip()
    .str.replace(r'\*', '', regex=True)
    .str.replace(r'\(.*?\)', '', regex=True)
    .str.replace(r'[^a-z\s]', '', regex=True)
)
```

District names contained spelling variations and legacy names. A canonical district mapping was applied to standardize district identities before analysis, preventing double counting.

In [105…
```
## District mapping Uttar Pradesh
district_standard_map = {
    "bara banki": "Barabanki",
    "barabanki": "Barabanki",

    "bulandshahar": "Bulandshahr",
    "bulandshahr": "Bulandshahr",

    "siddharth nagar": "Siddharthnagar",
    "siddharthnagar": "Siddharthnagar",

    "kushi nagar": "Kushinagar",
    "kushinagar *": "Kushinagar",
    "kushinagar": "Kushinagar",

    "shrawasti": "Shravasti",
    "shravasti": "Shravasti",
```

```
    "mahrajganj": "Maharajganj",
    "maharajganj": "Maharajganj",

    "sant ravidas nagar bhadohi": "Bhadohi",
    "sant ravidas nagar": "Bhadohi",
    "bhadohi": "Bhadohi",

    "jyotiba phule nagar": "Amroha",
    "amroha": "Amroha",

    "allahabad": "Prayagraj",
    "prayagraj": "Prayagraj",

    "faizabad": "Ayodhya",
    "ayodhya": "Ayodhya",

    "rae bareli": "Raebareli",
    "raebareli": "Raebareli",

    "bagpat": "Baghpat",
    "baghpat": "Baghpat"
}
```

In [106…
```
df['district_clean'] = (
    df['district']
    .apply(clean_name)
    .map(district_standard_map)
    .fillna(df_uttarpradesh['district'])

)
```

In [106…
```
## Remaining unmapped
df_uttarpradesh[df_uttarpradesh['district_clean'].isna()]['district'].unique
# count check
df_uttarpradesh['district_clean'].nunique()
```

Out[106… 89

In [106…
```
df_uttarpradesh = df[df['state_clean']=='Uttar Pradesh']
df_uttarpradesh
```

Out[106…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
2	2025-03-09	Uttar Pradesh	Kanpur Nagar	208001	29	82	12
3	2025-03-09	Uttar Pradesh	Aligarh	202133	62	29	15
7	2025-03-09	Uttar Pradesh	Bahraich	271865	26	60	14
8	2025-03-09	Uttar Pradesh	Firozabad	283204	28	26	10
10	2025-03-09	Uttar Pradesh	Maharajganj	273164	31	70	13
...	...	...	...	...	...	...	...
1005743	2025-12-31	Uttar Pradesh	Varanasi	221002	10	18	0
1005744	2025-12-31	Uttar Pradesh	Varanasi	221104	4	11	0
1005745	2025-12-31	Uttar Pradesh	Varanasi	221107	1	15	0
1005746	2025-12-31	Uttar Pradesh	Varanasi	221207	1	9	0
1005747	2025-12-31	Uttar Pradesh	Varanasi	221313	1	0	0

108066 rows × 13 columns

In [106…

```
# Check Bihar-specific unmapped districts
df_uttarpradesh_unmapped = df_uttarpradesh[df_uttarpradesh['district_clean']
print(f"Unmapped Uttar Pradesh districts count: {len(df_uttarpradesh_unmappe
df_uttarpradesh_unmapped['district'].unique()
```

```
Unmapped Uttar Pradesh districts count: 0
```

Out[106…

```
array([], dtype=object)
```

In [106…

```
# Final unique districts in Uttar Pradesh after cleaning
df_uttarpradesh['district_clean'].unique()
```

Out[106…
```
array(['Kanpur Nagar', 'Aligarh', 'Bahraich', 'Firozabad', 'Maharajganj',
       'Ghaziabad', 'Gautam Buddha Nagar', 'Lucknow', 'Agra', 'Unnao',
       'Saharanpur', 'Jaunpur', 'Gorakhpur', 'Bulandshahr', 'Mathura',
       'Banda', 'Kheri', 'Budaun', 'Kanpur Dehat', 'Varanasi', 'Baghpat',
       'Fatehpur', 'Etawah', 'Shamli', 'Balrampur', 'Barabanki',
       'Shahjahanpur', 'Gonda', 'Bareilly', 'Sitapur', 'Sultanpur',
       'Shravasti', 'Chandauli', 'Mainpuri', 'Muzaffarnagar',
       'Siddharthnagar', 'Ambedkar Nagar', 'Pilibhit', 'Kaushambi',
       'Jalaun', 'Etah', 'Meerut', 'Basti', 'Azamgarh', 'Rampur',
       'Moradabad', 'Amroha', 'Bijnor', 'Deoria', 'Prayagraj', 'Lalitpur',
       'Hapur', 'Hathras', 'Kushinagar', 'Farrukhabad', 'Hardoi',
       'Ayodhya', 'Sambhal', 'Jhansi', 'Kasganj', 'Kannauj', 'Amethi',
       'Auraiya', 'Ballia', 'Bhadohi', 'Chitrakoot', 'Ghazipur', 'Mahoba',
       'Mau', 'Mirzapur', 'Pratapgarh', 'Raebareli', 'Sant Kabir Nagar',
       'Sant Ravidas Nagar', 'Sonbhadra', 'Hamirpur',
       'Jyotiba Phule Nagar', 'Sant Ravidas Nagar Bhadohi'], dtype=object)
```

In [106…
```
# Total unique districts in Uttar Pradesh after cleaning
df_uttarpradesh['district_clean'].nunique()
```

Out[106… 78

In [106…
```
# Check unique pincodes per district in Uttar Pradesh
pincode_check_UP = df_uttarpradesh.groupby('district_clean')['pincode'].nuni
pincode_check_UP
```

Out[106…

	district_clean	unique_pincodes
0	Agra	29
1	Aligarh	39
2	Ambedkar Nagar	34
3	Amethi	39
4	Amroha	14
…	…	…
73	Sitapur	25
74	Sonbhadra	20
75	Sultanpur	35
76	Unnao	31
77	Varanasi	42

78 rows × 2 columns

In [106…
```
# Pincode associated with multiple districts in Uttar Pradesh
pin_district_count_UP = (
    df_uttarpradesh.groupby('pincode')['district_clean']
    .nunique()
```

```
    .reset_index(name='district_count')
)
```

In [106…
```
pin_district_count_UP
```

Out[106…

	pincode	district_count
0	121705	1
1	201001	2
2	201002	2
3	201003	1
4	201004	1
...	...	...
1732	285203	1
1733	285204	1
1734	285205	2
1735	285206	1
1736	285223	2

1737 rows × 2 columns

In [106…
```
# Extract problematic pincodes (associated with >1 district)
problem_pins_UP = pin_district_count_UP[
    pin_district_count_UP['district_count'] > 1
]
```

In [107…
```
problem_pins_UP
```

Out[107…

	pincode	district_count
1	201001	2
2	201002	2
5	201005	2
6	201006	2
7	201007	2
…	…	…
1713	284403	2
1724	285125	2
1725	285126	2
1734	285205	2
1736	285223	2

276 rows × 2 columns

In [107…

```
# Get all records with problematic pincodes(flagged records)
df_flagged_UP = df_uttarpradesh.merge(
    problem_pins_UP[['pincode']],
    on='pincode',
    how='inner'
)
```

In [107…

```
df_flagged_UP
```

Out [107…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
0	2025-03-09	Uttar Pradesh	Kanpur Nagar	208001	29	82	12
1	2025-03-09	Uttar Pradesh	Bahraich	271865	26	60	14
2	2025-03-09	Uttar Pradesh	Ghaziabad	201102	50	113	11
3	2025-03-15	Uttar Pradesh	Ghaziabad	201001	33	125	16
4	2025-03-15	Uttar Pradesh	Ghaziabad	201102	19	146	30
...	...	...	...	...	...	...	...
23227	2025-12-31	Uttar Pradesh	Siddharthnagar	272152	3	31	1
23228	2025-12-31	Uttar Pradesh	Sultanpur	227806	3	2	0
23229	2025-12-31	Uttar Pradesh	Sultanpur	227808	9	11	0
23230	2025-12-31	Uttar Pradesh	Unnao	209801	25	65	0
23231	2025-12-31	Uttar Pradesh	Unnao	241504	26	32	0

23232 rows × 13 columns

In [107…

```
# Summary of flagged records by district and pincode(for review)
flagged_pincode_UP=df_flagged_UP.groupby(['district_clean','pincode'])[['age
# flagged_pincode_UP.to_excel('flagged_pincode_UP_domain.xlsx')
```

In [107…

```
# Add total enrollment column to flagged_pincode
flagged_pincode_UP['total_enrollment']=flagged_pincode_UP['age_0_5']+flagged
flagged_pincode_UP
```

Out [107…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
0	Aligarh	202001	6212	5201	265	11414.0
1	Aligarh	202002	801	750	22	1551.0
2	Aligarh	202121	188	136	1	324.0
3	Aligarh	202139	0	2	0	16.0
4	Aligarh	202150	66	29	0	101.0
...	...	...	...	...	...	...
571	Varanasi	221011	363	520	72	NaN
572	Varanasi	221101	532	1020	19	NaN
573	Varanasi	221116	46	58	0	NaN
574	Varanasi	232104	1	1	0	NaN
575	Varanasi	232105	0	1	0	NaN

576 rows × 6 columns

In [107…

```
flagged_pincode_UP['total_enrollment'] = (
        flagged_pincode_UP['age_0_5'] +
        flagged_pincode_UP['age_5_17'] +
        flagged_pincode_UP['age_18_greater']
)

idx = (
        flagged_pincode_UP
        .dropna(subset=['total_enrollment'])
        .groupby('pincode')['total_enrollment']
        .idxmax()
)

df_filtered_UP = flagged_pincode_UP.loc[idx]
df_filtered_UP
```

Out[107…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
213	Ghaziabad	201001	2479	1991	73	4543
214	Ghaziabad	201002	553	546	40	1139
215	Ghaziabad	201005	732	553	31	1316
216	Ghaziabad	201006	111	36	4	151
217	Ghaziabad	201007	153	129	3	285
…	…	…	…	…	…	…
366	Lalitpur	284403	1011	251	18	1280
283	Jalaun	285125	130	124	3	257
284	Jalaun	285126	99	72	2	173
285	Jalaun	285205	417	229	31	677
286	Jalaun	285223	34	34	0	68

276 rows × 6 columns

In [107…

```
# Creating a flag for pincodes associated with multiple districts
df_uttarpradesh['pin_multi_district_flag_UP']=(
    df_uttarpradesh.groupby('pincode')['district_clean']
    .transform('nunique')>1
)
```

In [107…

```
# Creating a flag for pincodes associated with multiple districts
pin_district_map = (
    df_uttarpradesh[df_uttarpradesh['pin_multi_district_flag_UP']]
    .groupby('pincode')['district_clean']  # noqa: SC100
    .unique()
    .reset_index()
)
```

In [107…

```
## monthly enrolment check
df_uttarpradesh['month'] = df_uttarpradesh['date'].dt.month.astype(str).str.
df_uttarpradesh
```

Out[107…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
2	2025-03-09	Uttar Pradesh	Kanpur Nagar	208001	29	82	12
3	2025-03-09	Uttar Pradesh	Aligarh	202133	62	29	15
7	2025-03-09	Uttar Pradesh	Bahraich	271865	26	60	14
8	2025-03-09	Uttar Pradesh	Firozabad	283204	28	26	10
10	2025-03-09	Uttar Pradesh	Maharajganj	273164	31	70	13
...	...	...	...	...	...	...	...
1005743	2025-12-31	Uttar Pradesh	Varanasi	221002	10	18	0
1005744	2025-12-31	Uttar Pradesh	Varanasi	221104	4	11	0
1005745	2025-12-31	Uttar Pradesh	Varanasi	221107	1	15	0
1005746	2025-12-31	Uttar Pradesh	Varanasi	221207	1	9	0
1005747	2025-12-31	Uttar Pradesh	Varanasi	221313	1	0	0

108066 rows × 14 columns

In [107…

```
# Dropping Date District and state as we have District clean State clean
df_uttarpradesh_cleaned=df_uttarpradesh.drop(columns=['date','district','sta
df_uttarpradesh_cleaned
```

Out[107…

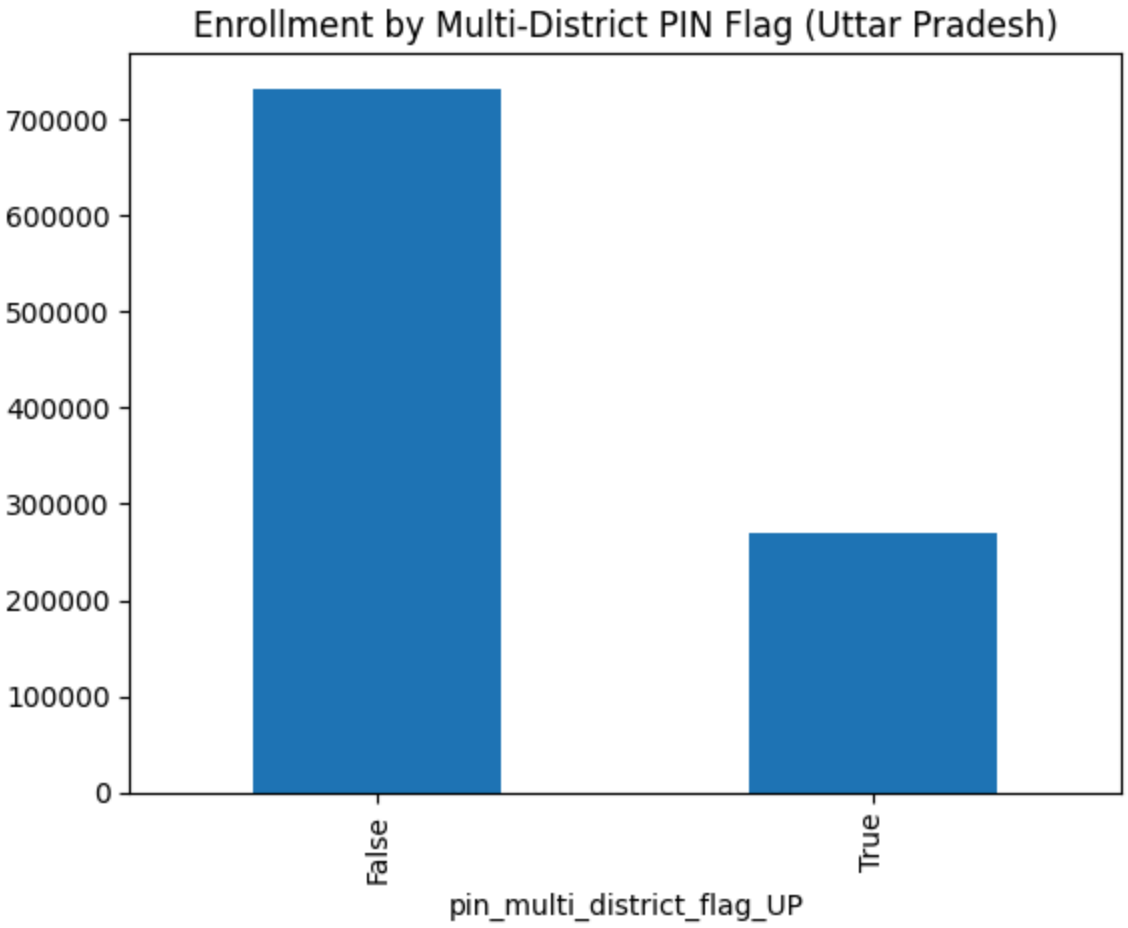
	pincode	age_0_5	age_5_17	age_18_greater	year	month	month_name	s
2	208001	29	82	12	2025	03	Mar	
3	202133	62	29	15	2025	03	Mar	
7	271865	26	60	14	2025	03	Mar	
8	283204	28	26	10	2025	03	Mar	
10	273164	31	70	13	2025	03	Mar	
...	...	...	...	...	...	...	...	
1005743	221002	10	18	0	2025	12	Dec	
1005744	221104	4	11	0	2025	12	Dec	
1005745	221107	1	15	0	2025	12	Dec	
1005746	221207	1	9	0	2025	12	Dec	
1005747	221313	1	0	0	2025	12	Dec	

108066 rows × 11 columns

Flagged vs unflagged pincodes

In [108…

```
df_uttarpradesh.groupby('pin_multi_district_flag_UP')['total_enrollment'].su
    kind='bar'
)
plt.title('Enrollment by Multi-District PIN Flag (Uttar Pradesh)')
plt.show()
```



In [108…
```
# Aggregate Uttar Pradesh enrollment data at the district level by summing e
# across different age groups (0–5, 5–17, and 18+).
df_uttarpradesh_dist_level = df_uttarpradesh_cleaned.groupby('district_clear
    ['age_0_5', 'age_5_17', 'age_18_greater']
].sum()

# Compute total enrollment for each district by adding all age-group enrollm
df_uttarpradesh_dist_level['total_enrollment'] = (
    df_uttarpradesh_dist_level['age_0_5'] +
    df_uttarpradesh_dist_level['age_5_17'] +
    df_uttarpradesh_dist_level['age_18_greater']
)
```

In [108…
```
df_uttarpradesh_dist_level.shape
```

Out[108… (78, 4)

In [108…
```
df_uttarpradesh_dist_level
```

Out[108…

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Agra	15938	12506	901	29345
Aligarh	13639	11631	581	25851
Ambedkar Nagar	3777	4021	49	7847
Amethi	3829	3184	53	7066
Amroha	5502	2448	60	8010
...	...	...	...	...
Sitapur	16003	13732	740	30475
Sonbhadra	2943	1964	13	4920
Sultanpur	5343	5723	132	11198
Unnao	9735	8860	365	18960
Varanasi	9370	13318	451	23139

78 rows × 4 columns

In [108…

```
# Sorting District with total enrollment
df_uttarpradesh_dist_level.sort_values("total_enrollment",ascending=False).r
# df_uttarpradesh_dist_level.to_excel('uttarpradesh_district_level_enrolment
```

Out[108…

	district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
0	Bahraich	14491	22132	2274	38897
1	Sitapur	16003	13732	740	30475
2	Agra	15938	12506	901	29345
3	Bareilly	16880	9892	585	27357
4	Aligarh	13639	11631	581	25851
...	...	...	...	...	...
73	Hamirpur	1985	1362	25	3372
74	Mahoba	1921	1006	11	2938
75	Bhadohi	259	270	4	533
76	Sant Ravidas Nagar Bhadohi	36	43	0	79
77	Jyotiba Phule Nagar	57	22	0	79

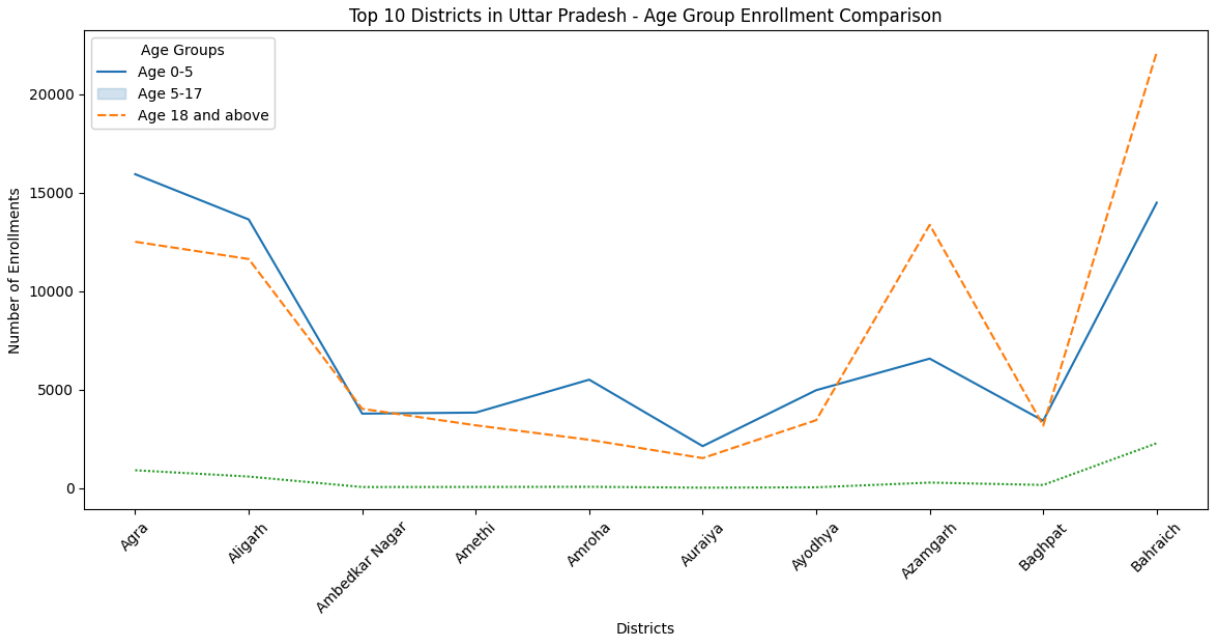
78 rows × 5 columns

Plotting Top 10 Districts by number of Enrollment

Visualize age-wise enrollment distribution across top 10 Uttar Pradesh districts using a line plot

In [108…
```
df_uttarpradesh_dist_level1 = df_uttarpradesh_dist_level.head(10)
```

In [108…
```
# Import required visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Set the figure size for better readability
plt.figure(figsize=(14,6))

# Plot line chart to compare enrollment trends across age groups
# for the top 10 districts in Bihar
sns.lineplot(
    data=df_uttarpradesh_dist_level1[['age_0_5', 'age_5_17', 'age_18_greater
)

# Add title and axis labels
plt.title('Top 10 Districts in Uttar Pradesh - Age Group Enrollment Comparis
plt.xlabel('Districts')
plt.ylabel('Number of Enrollments')

# Customize legend to clearly represent age groups
plt.legend(
    title='Age Groups',
    labels=['Age 0-5', 'Age 5-17', 'Age 18 and above']
)

# Set district names on x-axis and rotate labels for clarity
plt.xticks(
    ticks=range(len(df_uttarpradesh_dist_level1.index)),
    labels=df_uttarpradesh_dist_level1.index,
    rotation=45
)

# Display the plot
plt.show()
```



Plotting month vs total enrollment

In [108…
```
## monthly enrolment trend in bihar
df_uttarpradesh_monthly = df_uttarpradesh_cleaned.groupby('month')[['age_0_5
df_uttarpradesh_monthly['total_enrollment'] = df_bihar_monthly['age_0_5'] +
```

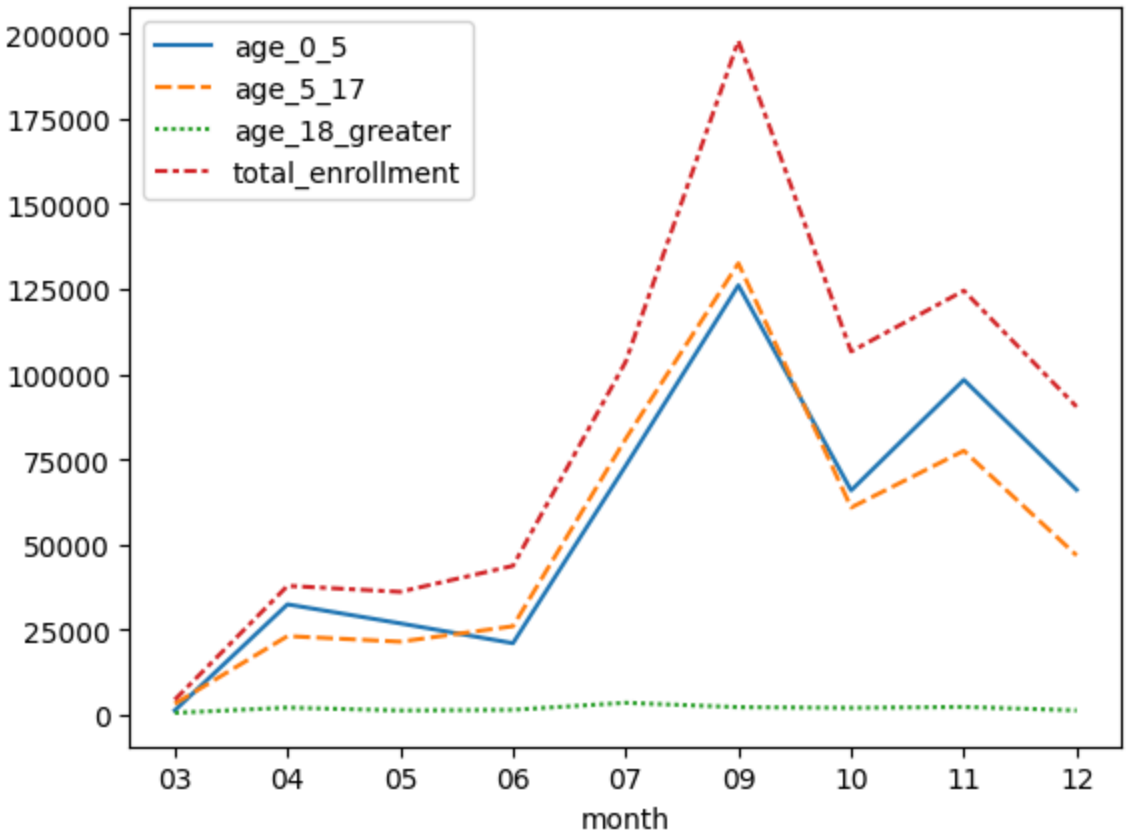
In [108…
```
df_uttarpradesh_monthly.sort_values("total_enrollment",ascending=False).rese
```

Out[108…

	month	age_0_5	age_5_17	age_18_greater	total_enrollment
0	09	126150	132594	2335	197869
1	11	98391	77584	2402	124558
2	10	65951	60924	2120	106734
3	07	73166	81078	3622	103708
4	12	66126	46876	1402	90501
5	06	21109	26080	1564	43804
6	04	32513	23189	2201	37941
7	05	26928	21554	1379	36184
8	03	1393	3326	674	4516

In [108…
```
sns.lineplot(data=df_uttarpradesh_monthly)
```

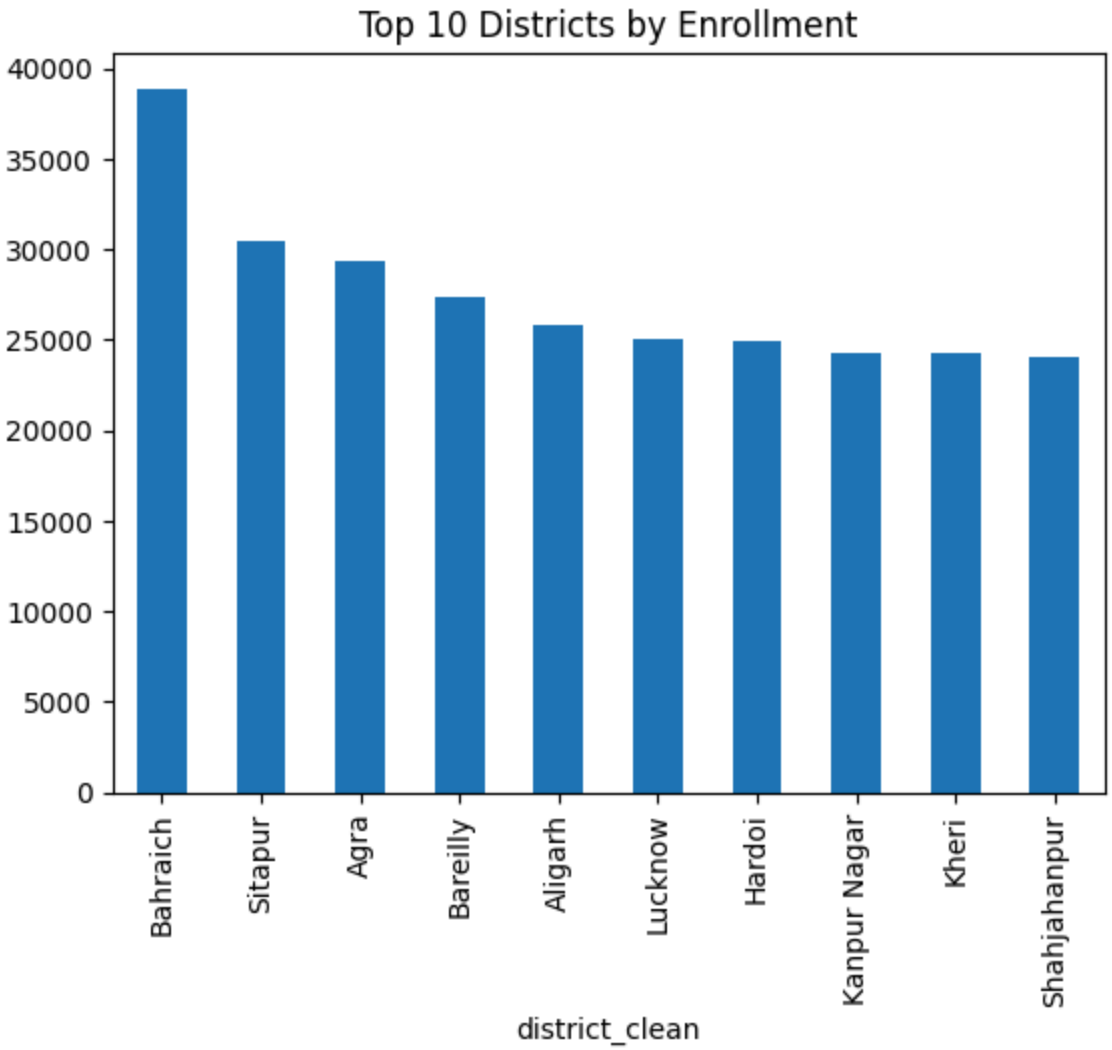
Out[108…
```
<Axes: xlabel='month'>
```



Top 10 Districts by Enrollment

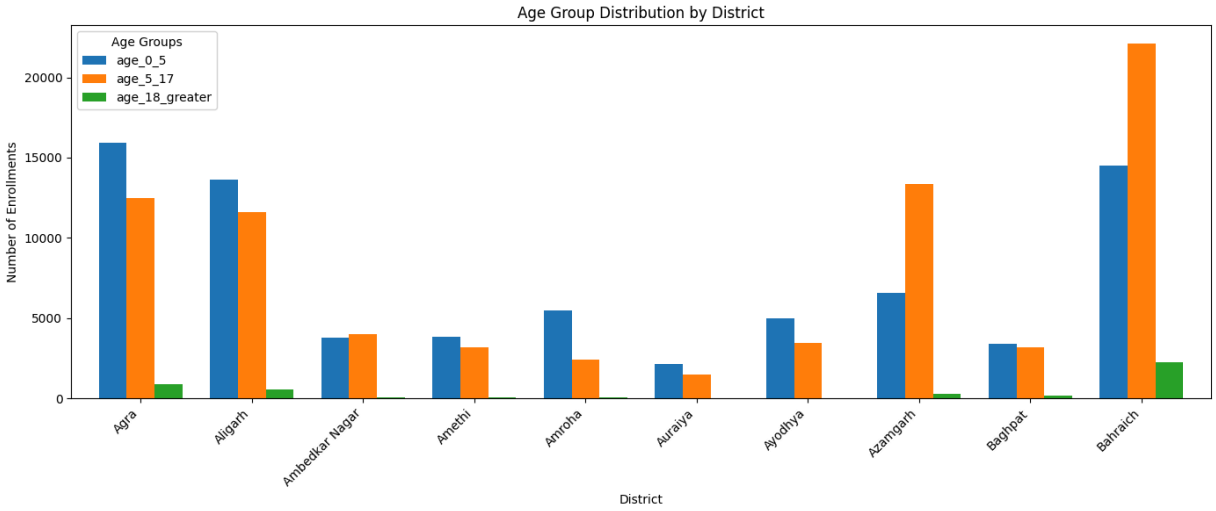
In [109…

```
top_districts_UP = df.groupby('district_clean')['total_enrollment'] \
                   .sum().sort_values(ascending=False).head(10)

top_districts_UP.plot(kind='bar')
plt.title('Top 10 Districts by Enrollment')
plt.show()
```



In [109…

```python
import matplotlib.pyplot as plt

district_age_UP = df.groupby('district_clean')[
    ['age_0_5','age_5_17','age_18_greater']
].sum().head(10)

ax = district_age_UP.plot(
    kind='bar',
    stacked=False,      # important → no overlap
    figsize=(14,6),
    width=0.75
)

plt.title('Age Group Distribution by District')
plt.xlabel('District')
plt.ylabel('Number of Enrollments')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Age Groups')
plt.tight_layout()
plt.show()
```

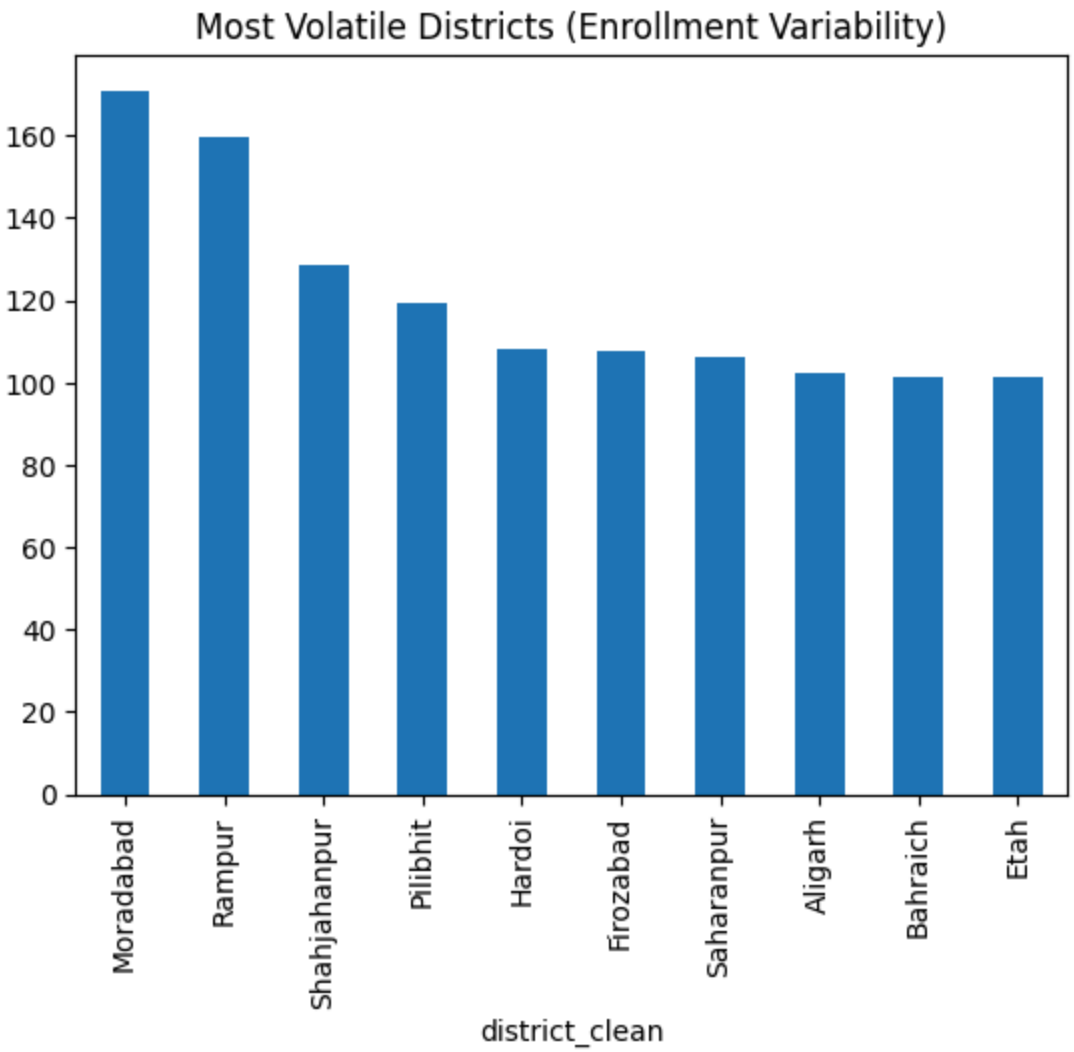


📊 Enrollment Variability Across Districts

This bar chart highlights the top 10 districts with the highest standard deviation in total enrollment, indicating significant variability over time. High volatility suggests
inconsistent enrollment patterns, possibly due to seasonal drives, migration, or administrative factors.

In [109…

```
district_std_UP = df.groupby('district_clean')['total_enrollment'].std()

district_std_UP.sort_values(ascending=False).head(10).plot(kind='bar')
plt.title('Most Volatile Districts (Enrollment Variability)')
plt.show()
```



📌 District-wise Enrollment Ranking Analysis

This analysis ranks districts based on their total Aadhaar enrollment over the entire study period.

In [109…

```
district_rank_UP = df.groupby('district_clean')['total_enrollment'].sum() \
                   .rank(ascending=False)

district_rank_UP.sort_values().head(10)
```

Out[109…
```
district_clean
Bahraich         1.0
Sitapur          2.0
Agra             3.0
Bareilly         4.0
Aligarh          5.0
Lucknow          6.0
Hardoi           7.0
Kanpur Nagar     8.0
Kheri            9.0
Shahjahanpur    10.0
Name: total_enrollment, dtype: float64
```

We performed univariate,
bivariate, and multivariate analysis to study
enrollment distribution across states, districts, and age groups.
Advanced analysis included district-level volatility, age composition,
time-based trends, and flag-based segmentation to uncover hidden patterns.

Madhya Pradesh

In [109…
```
# Extracting rows where state is Madhya Pradesh
df_madhyapradesh= df[df['state_clean']=='Madhya Pradesh']
```

In [109…
```
# Total unique districts in Madhya Pradesh
df_madhyapradesh['district'].nunique()
```

Out[109… 61

In [109…
```
# Same District have more than one spelling so we have to map them
df_madhyapradesh['district'].unique()
```

Out[109…
```
array(['Bhind', 'Gwalior', 'Katni', 'Ashok Nagar', 'Betul', 'Guna',
       'Khargone', 'Chhatarpur', 'Morena', 'Dhar', 'Jabalpur', 'Barwani',
       'Shivpuri', 'Raisen', 'Jhabua', 'Indore', 'Bhopal', 'Datia',
       'Sagar', 'Sheopur', 'Vidisha', 'Burhanpur', 'Panna', 'Khandwa',
       'Satna', 'Alirajpur', 'Dewas', 'Ashoknagar', 'Balaghat',
       'East Nimar', 'Sehore', 'Narsinghpur', 'Hoshangabad', 'Agar Malwa',
       'Anuppur', 'Chhindwara', 'Damoh', 'Dindori', 'Harda', 'Maihar',
       'Mandla', 'Mandsaur', 'Narmadapuram', 'Narsimhapur', 'Neemuch',
       'Niwari', 'Rajgarh', 'Ratlam', 'Rewa', 'Seoni', 'Shahdol',
       'Shajapur', 'Sidhi', 'Singrauli', 'Tikamgarh', 'Ujjain',
       'West Nimar', 'Harda *', 'Mauganj', 'Umaria', 'Pandhurna'],
      dtype=object)
```

Mapping District

In [109…
```
import pandas as pd
import re
```

```
def clean_name(x):
    if pd.isna(x):
        return x
    x = str(x).lower()
    x = re.sub(r'[^a-z]', '', x)
    x = re.sub(r'\s+',' ', x).strip()
    return x
```

In [109…
```
# District mapping Madhya Pradesh
mp_district_standard_map = {
    "ashok nagar": "Ashoknagar",
    "ashoknagar": "Ashoknagar",

    "east nimar": "Khandwa",
    "west nimar": "Khargone",

    "hoshangabad": "Narmadapuram",
    "narmadapuram": "Narmadapuram",

    "narsimhapur": "Narsinghpur",
    "narsinghpur": "Narsinghpur",

    "harda *": "Harda",
    "harda": "Harda",

    "mauganj": "Mauganj",
    "pandhurna": "Pandhurna"
}
```

In [109…
```
df['district_clean'] = (
    df['district']
    .apply(clean_name)
    .map(mp_district_standard_map)
    .fillna(df_madhyapradesh['district'])

)
```

In [110…
```
## Remaining unmapped
df_madhyapradesh[df_madhyapradesh['district_clean'].isna()]['district'].unic
# count check
df_madhyapradesh['district_clean'].nunique()
```

Out[110… 0

In [110…
```
df_madhyapradesh = df[df['state_clean']=='Madhya Pradesh']
df_madhyapradesh
```

Out[110…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	year
25	2025-03-09	Madhya Pradesh	Bhind	477116	37	18	11	2025
26	2025-03-09	Madhya Pradesh	Gwalior	475110	42	14	14	2025
27	2025-03-09	Madhya Pradesh	Katni	483501	53	42	11	2025
76	2025-03-20	Madhya Pradesh	Gwalior	474001	73	25	18	2025
125	2025-03-23	Madhya Pradesh	Gwalior	474001	10	21	16	2025
...	...	...	...	...	...	...	...	..
1004141	2025-12-31	Madhya Pradesh	Umaria	484661	66	23	0	2025
1004142	2025-12-31	Madhya Pradesh	Umaria	486661	2	0	0	2025
1004143	2025-12-31	Madhya Pradesh	Vidisha	464001	32	6	0	2025
1004144	2025-12-31	Madhya Pradesh	Vidisha	464220	7	3	0	2025
1004145	2025-12-31	Madhya Pradesh	West Nimar	451111	2	1	0	2025

49562 rows × 13 columns

In [110…

```
# Check Madhya Pradesh-specific unmapped districts
df_madhyapradesh_unmapped = df_madhyapradesh[df_madhyapradesh['district_clea
print(f"Unmapped Madhya Pradesh districts count: {len(df_madhyapradesh_unmap
df_madhyapradesh_unmapped['district'].unique()
```

```
Unmapped Madhya Pradesh districts count: 0
```

Out[110…

```
array([], dtype=object)
```

In [110…

```
# Final unique districts in Madhya Pradesh after cleaning
df_madhyapradesh['district_clean'].unique()
```

Out[110… array(['Bhind', 'Gwalior', 'Katni', 'Ashoknagar', 'Betul', 'Guna',
'Khargone', 'Chhatarpur', 'Morena', 'Dhar', 'Jabalpur', 'Barwani',
'Shivpuri', 'Raisen', 'Jhabua', 'Indore', 'Bhopal', 'Datia',
'Sagar', 'Sheopur', 'Vidisha', 'Burhanpur', 'Panna', 'Khandwa',
'Satna', 'Alirajpur', 'Dewas', 'Balaghat', 'East Nimar', 'Sehore',
'Narsinghpur', 'Narmadapuram', 'Agar Malwa', 'Anuppur',
'Chhindwara', 'Damoh', 'Dindori', 'Harda', 'Maihar', 'Mandla',
'Mandsaur', 'Neemuch', 'Niwari', 'Rajgarh', 'Ratlam', 'Rewa',
'Seoni', 'Shahdol', 'Shajapur', 'Sidhi', 'Singrauli', 'Tikamgarh',
'Ujjain', 'West Nimar', 'Mauganj', 'Umaria', 'Pandhurna'],
dtype=object)

In [110…
```
# Total unique districts in Madhya Pradesh after cleaning
df_madhyapradesh['district_clean'].nunique()
```

Out[110… 57

In [110…
```
# unique pincodes in Madhya Pradesh
df_madhyapradesh['pincode'].nunique()
```

Out[110… 787

In [110…
```
# Check unique pincodes per district in Madhya Pradesh
pincode_check = df_madhyapradesh.groupby('district_clean')['pincode'].nuniqu
pincode_check
```

Out[110…

	district_clean	unique_pincodes
0	Agar Malwa	6
1	Alirajpur	10
2	Anuppur	20
3	Ashoknagar	10
4	Balaghat	24
5	Barwani	11
6	Betul	21
7	Bhind	20
8	Bhopal	38
9	Burhanpur	5
10	Chhatarpur	20
11	Chhindwara	27
12	Damoh	14
13	Datia	9
14	Dewas	18
15	Dhar	23
16	Dindori	9
17	East Nimar	20
18	Guna	20
19	Gwalior	21
20	Harda	6
21	Indore	31
22	Jabalpur	27
23	Jhabua	14
24	Katni	17
25	Khandwa	16
26	Khargone	20
27	Maihar	8
28	Mandla	17
29	Mandsaur	20
30	Mauganj	9
31	Morena	12

	district_clean	unique_pincodes
32	Narmadapuram	20
33	Narsinghpur	16
34	Neemuch	15
35	Niwari	9
36	Pandhurna	8
37	Panna	15
38	Raisen	20
39	Rajgarh	15
40	Ratlam	14
41	Rewa	30
42	Sagar	28
43	Satna	26
44	Sehore	18
45	Seoni	17
46	Shahdol	23
47	Shajapur	23
48	Sheopur	5
49	Shivpuri	21
50	Sidhi	12
51	Singrauli	12
52	Tikamgarh	18
53	Ujjain	23
54	Umaria	10
55	Vidisha	13
56	West Nimar	20

In [110…

```
# Pincode associated with multiple districts in Madhya Pradesh
pin_district_count = (
    df_madhyapradesh.groupby('pincode')['district_clean']
    .nunique()
    .reset_index(name='district_count')
)
```

In [110…

```
pin_district_count
```

Out[110…

	pincode	district_count
0	450001	2
1	450051	2
2	450110	2
3	450112	2
4	450114	2
…	…	…
782	488441	1
783	488442	1
784	488443	1
785	488446	1
786	488448	1

787 rows × 2 columns

In [110…

```
# Extract problematic pincodes (associated with >1 district)
problem_pins = pin_district_count[
    pin_district_count['district_count'] > 1
]
```

In [111…

```
problem_pins
```

Out[111…

	pincode	district_count
0	450001	2
1	450051	2
2	450110	2
3	450112	2
4	450114	2
…	…	…
748	486882	2
749	486884	2
751	486886	4
754	486889	2
755	486890	2

163 rows × 2 columns

In [111…
```
# Get all records with problematic pincodes(flagged records)
df_flagged = df_madhyapradesh.merge(
    problem_pins[['pincode']],
    on='pincode',
    how='inner'
)
```

In [111…
```
df_flagged
```

Out[111…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	yea
0	2025-03-09	Madhya Pradesh	Gwalior	475110	42	14	14	2025
1	2025-03-27	Madhya Pradesh	Gwalior	475110	54	26	14	2025
2	2025-04-01	Madhya Pradesh	Ashok Nagar	473335	125	29	22	2025
3	2025-04-01	Madhya Pradesh	Ashok Nagar	473443	141	57	15	2025
4	2025-04-01	Madhya Pradesh	Guna	473101	110	32	11	2025
...	...	...	...	...	...	...	...	..
13596	2025-12-31	Madhya Pradesh	Tikamgarh	472246	6	5	0	2025
13597	2025-12-31	Madhya Pradesh	Umaria	484661	66	23	0	2025
13598	2025-12-31	Madhya Pradesh	Umaria	486661	2	0	0	2025
13599	2025-12-31	Madhya Pradesh	Vidisha	464001	32	6	0	2025
13600	2025-12-31	Madhya Pradesh	West Nimar	451111	2	1	0	2025

13601 rows × 13 columns

In [111…
```
# Summary of flagged records by district and pincode(for review)
flagged_pincode=df_flagged.groupby(['district_clean','pincode'])[['age_0_5',
# flagged_pincode.to_excel('flagged_pincode_domain_mp.xlsx')
```

In [111…
```
# Add total enrollment column to flagged_pincode
flagged_pincode['total_enrollment']=flagged_pincode['age_0_5']+flagged_pincc
flagged_pincode
```

Out[111…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
0	Agar Malwa	465230	254	54	3	311
1	Agar Malwa	465441	618	136	16	770
2	Agar Malwa	465445	369	65	0	434
3	Agar Malwa	465447	389	81	2	472
4	Agar Malwa	465550	205	47	7	259
...	...	...	...	...	...	...
345	West Nimar	451335	17	8	0	25
346	West Nimar	451440	45	6	0	51
347	West Nimar	451441	76	33	0	109
348	West Nimar	451442	27	9	1	37
349	West Nimar	451660	13	1	0	14

350 rows × 6 columns

In [111…

```
idx = flagged_pincode.groupby('pincode')['total_enrollment'].idxmax()
df_filtered = flagged_pincode.loc[idx]
df_filtered
```

Out[111…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
141	Khandwa	450001	1082	301	17	1400
142	Khandwa	450051	45	17	1	63
143	Khandwa	450110	52	23	0	75
144	Khandwa	450112	107	61	5	173
84	East Nimar	450114	143	103	2	248
...	...	...	...	...	...	...
303	Singrauli	486882	988	130	4	1122
297	Sidhi	486884	47	6	0	53
305	Singrauli	486886	2124	337	3	2464
306	Singrauli	486889	220	65	6	291
300	Sidhi	486890	67	21	0	88

163 rows × 6 columns

In [111…

```
# Creating a flag for pincodes associated with multiple districts
df_madhyapradesh['pin_multi_district_flag']=(
    df_madhyapradesh.groupby('pincode')['district_clean']
```

```
    .transform('nunique')>1
)
```

In [111…
```
# Creating a flag for pincodes associated with multiple districts
pin_district_map= (
    df_madhyapradesh[df_madhyapradesh['pin_multi_district_flag']]
    .groupby('pincode')['district_clean']  # noqa: SC100
    .unique()
    .reset_index()
)
```

In [111…
```
## monthly enrolment check
df_madhyapradesh['month'] = df_madhyapradesh['date'].dt.month.astype(str).st
df_madhyapradesh
```

Out[111…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	year
25	2025-03-09	Madhya Pradesh	Bhind	477116	37	18	11	2025
26	2025-03-09	Madhya Pradesh	Gwalior	475110	42	14	14	2025
27	2025-03-09	Madhya Pradesh	Katni	483501	53	42	11	2025
76	2025-03-20	Madhya Pradesh	Gwalior	474001	73	25	18	2025
125	2025-03-23	Madhya Pradesh	Gwalior	474001	10	21	16	2025
...	...	...	...	...	...	...	...	...
1004141	2025-12-31	Madhya Pradesh	Umaria	484661	66	23	0	2025
1004142	2025-12-31	Madhya Pradesh	Umaria	486661	2	0	0	2025
1004143	2025-12-31	Madhya Pradesh	Vidisha	464001	32	6	0	2025
1004144	2025-12-31	Madhya Pradesh	Vidisha	464220	7	3	0	2025
1004145	2025-12-31	Madhya Pradesh	West Nimar	451111	2	1	0	2025

49562 rows × 14 columns

In [111…
```
# Dropping Date District and state as we have District clean State clean
df_madhyapradesh_cleaned=df_madhyapradesh.drop(columns=['date','district','s
```

```
df_madhyapradesh_cleaned
```

Out[111…

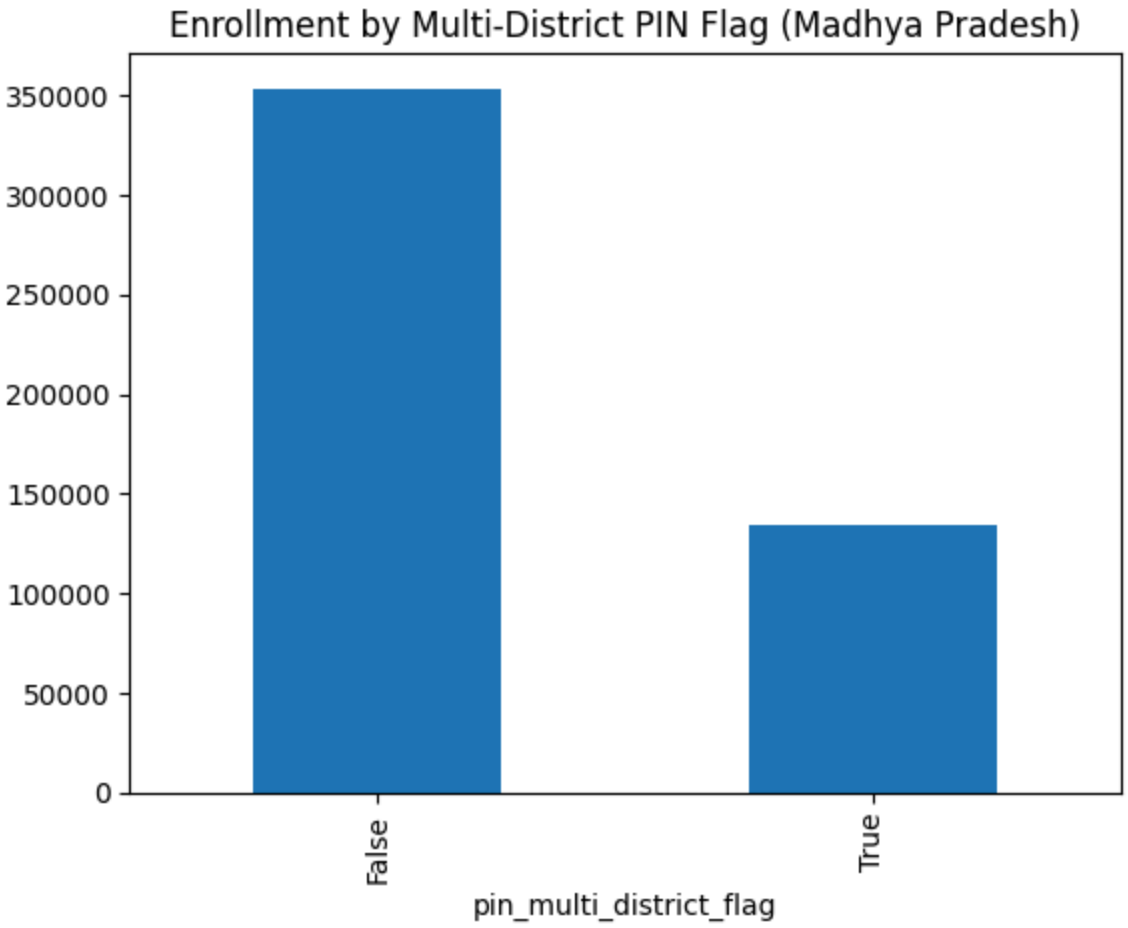
	pincode	age_0_5	age_5_17	age_18_greater	year	month	month_name	s
25	477116	37	18	11	2025	03	Mar	
26	475110	42	14	14	2025	03	Mar	
27	483501	53	42	11	2025	03	Mar	
76	474001	73	25	18	2025	03	Mar	
125	474001	10	21	16	2025	03	Mar	
…	…	…	…	…	…	…	…	…
1004141	484661	66	23	0	2025	12	Dec	
1004142	486661	2	0	0	2025	12	Dec	
1004143	464001	32	6	0	2025	12	Dec	
1004144	464220	7	3	0	2025	12	Dec	
1004145	451111	2	1	0	2025	12	Dec	

49562 rows × 11 columns

Flagged vs unflagged pincodes

In [112…

```
df_madhyapradesh.groupby('pin_multi_district_flag')['total_enrollment'].sum(
    kind='bar'
)
plt.title('Enrollment by Multi-District PIN Flag (Madhya Pradesh)')
plt.show()
```



In [112…
```
# Aggregate Madhya Pradesh enrollment data at the district level by summing
# across different age groups (0–5, 5–17, and 18+).
df_madhyapradesh_dist_level = df_madhyapradesh_cleaned.groupby('district_cle
    ['age_0_5', 'age_5_17', 'age_18_greater']
].sum()

# Compute total enrollment for each district by adding all age-group enrollm
df_madhyapradesh_dist_level['total_enrollment'] = (
    df_madhyapradesh_dist_level['age_0_5'] +
    df_madhyapradesh_dist_level['age_5_17'] +
    df_madhyapradesh_dist_level['age_18_greater']
)
```

In [112…
```
df_madhyapradesh_dist_level.shape
```

Out[112… (57, 4)

In [112…
```
df_madhyapradesh_dist_level
```

Out [112...

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Agar Malwa	2079	418	29	2526
Alirajpur	3184	3359	369	6912
Anuppur	2824	344	13	3181
Ashoknagar	5300	2600	261	8161
Balaghat	6104	534	28	6666
Barwani	12794	8352	667	21813
Betul	6754	1202	126	8082
Bhind	8421	4821	635	13877
Bhopal	11063	1430	154	12647
Burhanpur	6542	2759	388	9689
Chhatarpur	11372	5925	367	17664
Chhindwara	8201	618	47	8866
Damoh	4957	923	46	5926
Datia	3972	1794	43	5809
Dewas	8333	1943	91	10367
Dhar	8607	3517	425	12549
Dindori	2370	247	1	2618
East Nimar	2293	1482	21	3796
Guna	7410	2224	65	9699
Gwalior	11981	5566	823	18370
Harda	2198	487	18	2703
Indore	16493	4963	602	22058
Jabalpur	10578	1704	101	12383
Jhabua	6552	6556	328	13436
Katni	8305	2365	116	10786
Khandwa	3166	1213	82	4461
Khargone	9781	4637	504	14922
Maihar	362	19	19	400
Mandla	4595	259	5	4859
Mandsaur	7321	803	24	8148
Mauganj	365	57	1	423

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Morena	14005	7605	658	22268
Narmadapuram	5696	903	41	6640
Narsinghpur	4779	1440	91	6310
Neemuch	4134	758	17	4909
Niwari	114	15	9	138
Pandhurna	66	4	28	98
Panna	6775	2397	128	9300
Raisen	6129	1219	122	7470
Rajgarh	6381	1535	34	7950
Ratlam	6376	1433	75	7884
Rewa	9199	1442	29	10670
Sagar	11373	1829	378	13580
Satna	11068	1908	128	13104
Sehore	5069	1292	54	6415
Seoni	5481	510	8	5999
Shahdol	4322	817	59	5198
Shajapur	5792	1030	18	6840
Sheopur	3740	3264	139	7143
Shivpuri	6979	4728	720	12427
Sidhi	5544	655	13	6212
Singrauli	5427	819	19	6265
Tikamgarh	5581	1521	28	7130
Ujjain	12249	1726	82	14057
Umaria	4225	890	24	5139
Vidisha	7855	2178	169	10202
West Nimar	608	133	6	747

In [112…

```
# Sorting District with total enrollment
df_madhyapradesh_dist_level.sort_values("total_enrollment",ascending=False).
# df_madhyapradesh_dist_level.to_excel('madhyapradesh_district_level_enrolme
```

Out [112...

	district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
0	Morena	14005	7605	658	22268
1	Indore	16493	4963	602	22058
2	Barwani	12794	8352	667	21813
3	Gwalior	11981	5566	823	18370
4	Chhatarpur	11372	5925	367	17664
5	Khargone	9781	4637	504	14922
6	Ujjain	12249	1726	82	14057
7	Bhind	8421	4821	635	13877
8	Sagar	11373	1829	378	13580
9	Jhabua	6552	6556	328	13436
10	Satna	11068	1908	128	13104
11	Bhopal	11063	1430	154	12647
12	Dhar	8607	3517	425	12549
13	Shivpuri	6979	4728	720	12427
14	Jabalpur	10578	1704	101	12383
15	Katni	8305	2365	116	10786
16	Rewa	9199	1442	29	10670
17	Dewas	8333	1943	91	10367
18	Vidisha	7855	2178	169	10202
19	Guna	7410	2224	65	9699
20	Burhanpur	6542	2759	388	9689
21	Panna	6775	2397	128	9300
22	Chhindwara	8201	618	47	8866
23	Ashoknagar	5300	2600	261	8161
24	Mandsaur	7321	803	24	8148
25	Betul	6754	1202	126	8082
26	Rajgarh	6381	1535	34	7950
27	Ratlam	6376	1433	75	7884
28	Raisen	6129	1219	122	7470
29	Sheopur	3740	3264	139	7143
30	Tikamgarh	5581	1521	28	7130
31	Alirajpur	3184	3359	369	6912

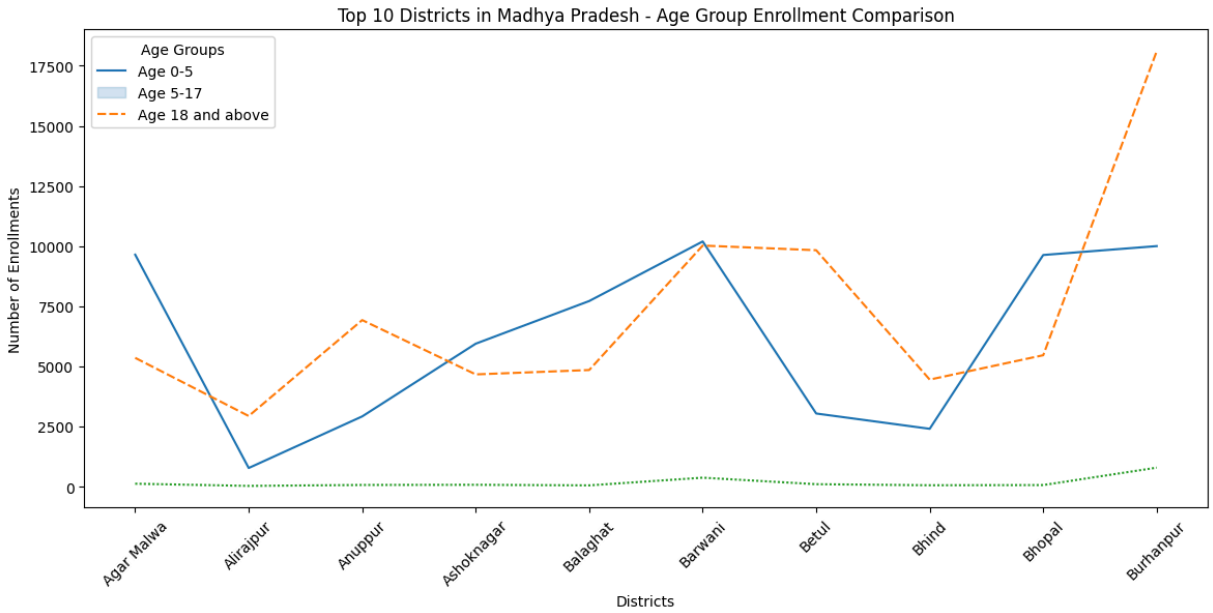
	district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
32	Shajapur	5792	1030	18	6840
33	Balaghat	6104	534	28	6666
34	Narmadapuram	5696	903	41	6640
35	Sehore	5069	1292	54	6415
36	Narsinghpur	4779	1440	91	6310
37	Singrauli	5427	819	19	6265
38	Sidhi	5544	655	13	6212
39	Seoni	5481	510	8	5999
40	Damoh	4957	923	46	5926
41	Datia	3972	1794	43	5809
42	Shahdol	4322	817	59	5198
43	Umaria	4225	890	24	5139
44	Neemuch	4134	758	17	4909
45	Mandla	4595	259	5	4859
46	Khandwa	3166	1213	82	4461
47	East Nimar	2293	1482	21	3796
48	Anuppur	2824	344	13	3181
49	Harda	2198	487	18	2703
50	Dindori	2370	247	1	2618
51	Agar Malwa	2079	418	29	2526
52	West Nimar	608	133	6	747
53	Mauganj	365	57	1	423
54	Maihar	362	19	19	400
55	Niwari	114	15	9	138
56	Pandhurna	66	4	28	98

Plotting Top 10 Districts by number of Enrollment

Visualize age-wise enrollment distribution across top 10 Madhya Pradesh districts using a line plot

In [112…
```
df_madhyapradesh_dist_level1 = df_madhyapradesh_dist_level.head(10)
```

In [112…
```
# Import required visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Set the figure size for better readability
plt.figure(figsize=(14,6))

# Plot line chart to compare enrollment trends across age groups
# for the top 10 districts in Madhya Pradesh
sns.lineplot(
    data=df_bihar_dist_level1[['age_0_5', 'age_5_17', 'age_18_greater']]
)

# Add title and axis labels
plt.title('Top 10 Districts in Madhya Pradesh - Age Group Enrollment Compari
plt.xlabel('Districts')
plt.ylabel('Number of Enrollments')

# Customize legend to clearly represent age groups
plt.legend(
    title='Age Groups',
    labels=['Age 0-5', 'Age 5-17', 'Age 18 and above']
)

# Set district names on x-axis and rotate labels for clarity
plt.xticks(
    ticks=range(len(df_bihar_dist_level1.index)),
    labels=df_madhyapradesh_dist_level1.index,
    rotation=45
)

# Display the plot
plt.show()
```



Plotting month vs total enrollment

In [112…
```
## monthly enrolment trend in Madhya Pradesh
df_madhyapradesh_monthly = df_madhyapradesh_cleaned.groupby('month')[['age_0
df_madhyapradesh_monthly['total_enrollment'] = df_madhyapradesh_monthly['age
```

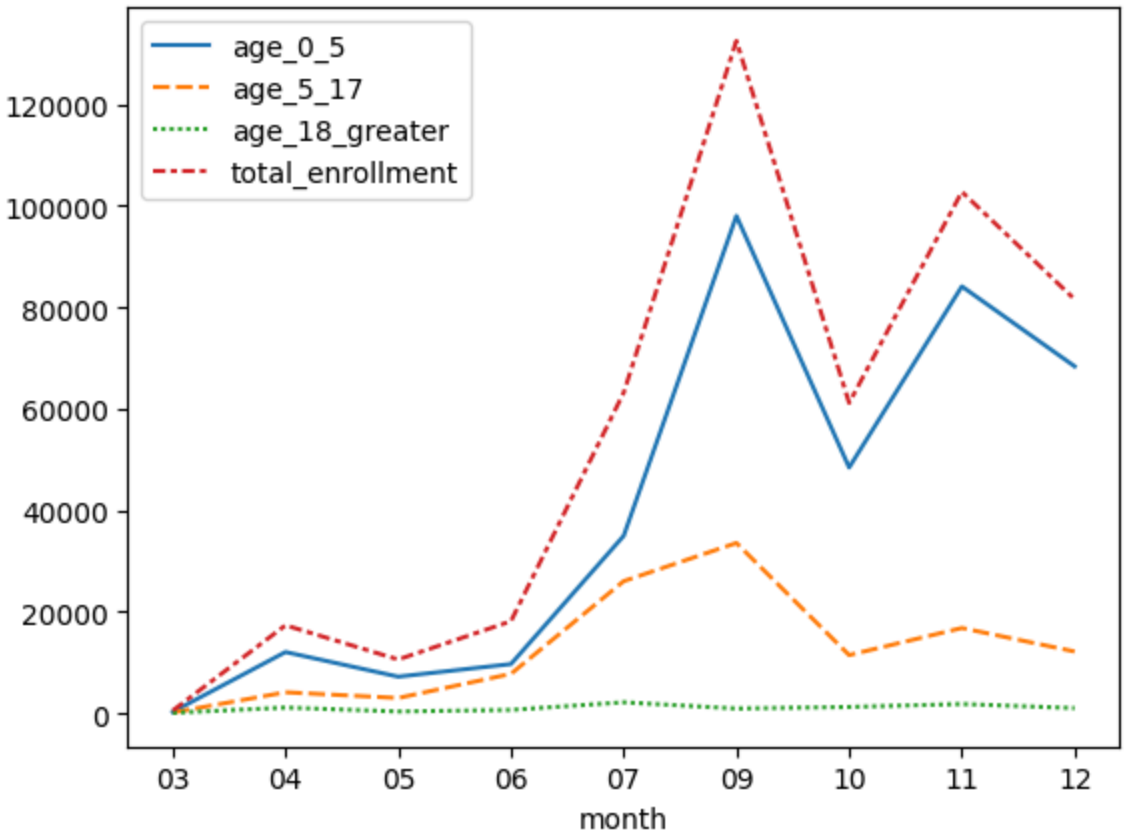
In [112…
```
df_madhyapradesh_monthly.sort_values("total_enrollment",ascending=False).res
```

Out[112…

	month	age_0_5	age_5_17	age_18_greater	total_enrollment
0	09	98016	33562	939	132517
1	11	84129	16814	1833	102776
2	12	68381	12180	1026	81587
3	07	34979	26040	2169	63188
4	10	48447	11477	1250	61174
5	06	9708	7774	672	18154
6	04	12077	4146	1128	17351
7	05	7221	3018	364	10603
8	03	286	161	95	542

In [112…
```
sns.lineplot(data=df_madhyapradesh_monthly)
```

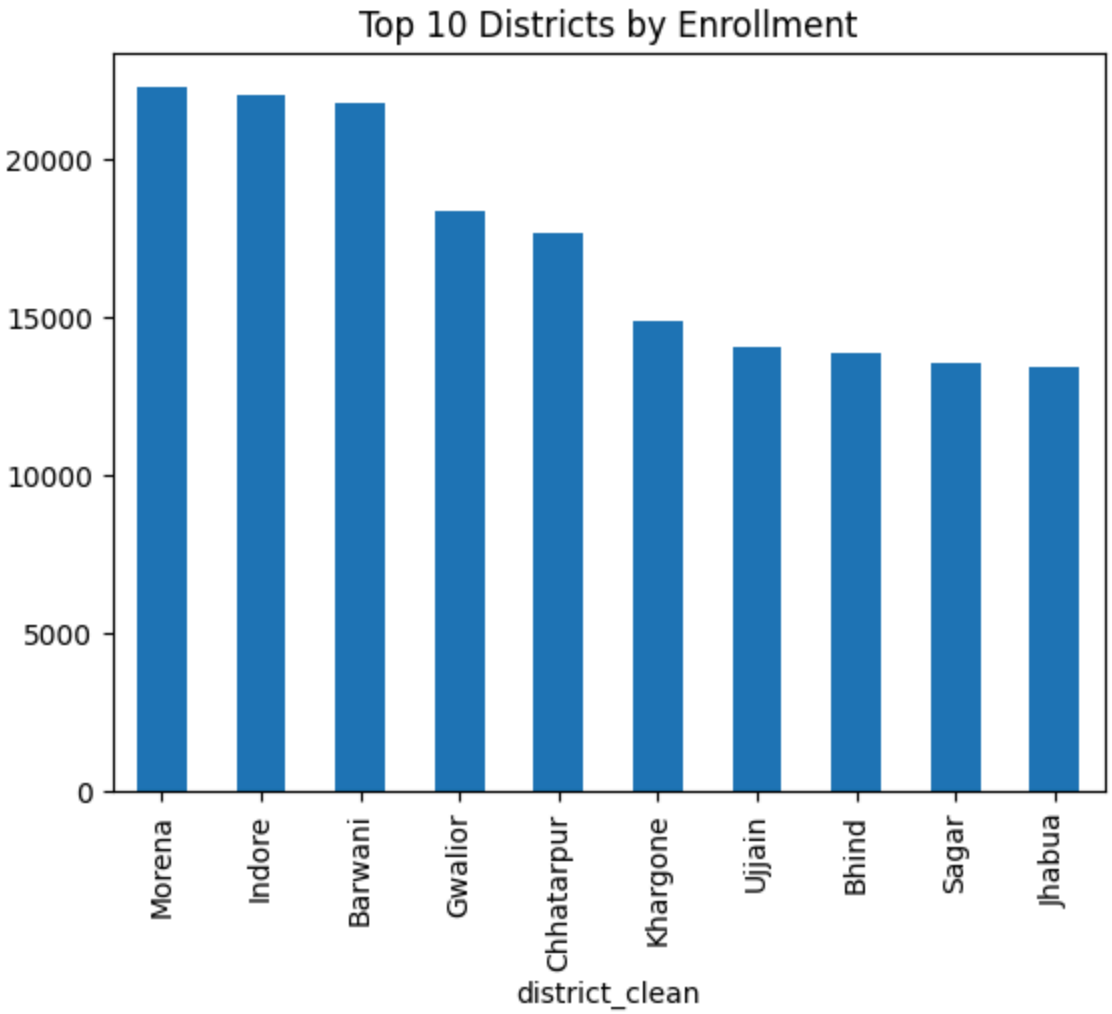
Out[112…
```
<Axes: xlabel='month'>
```



Top 10 Districts by Enrollment

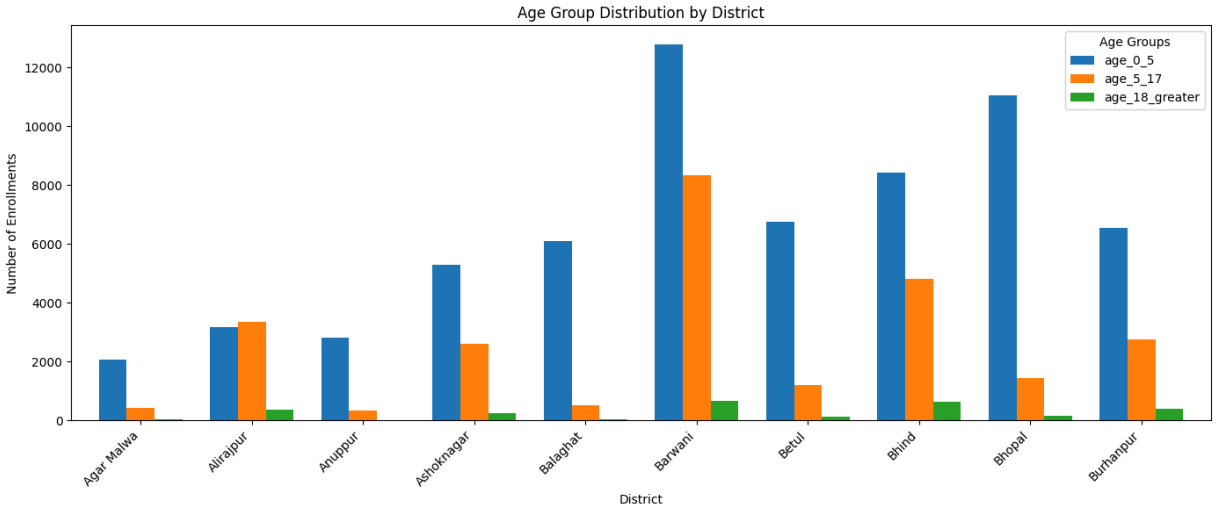
In [113…

```
top_districts = df.groupby('district_clean')['total_enrollment'] \
                  .sum().sort_values(ascending=False).head(10)

top_districts.plot(kind='bar')
plt.title('Top 10 Districts by Enrollment')
plt.show()
```



In [113…

```
import matplotlib.pyplot as plt

district_age = df.groupby('district_clean')[
    ['age_0_5','age_5_17','age_18_greater']
].sum().head(10)

ax = district_age.plot(
    kind='bar',
    stacked=False,      # important → no overlap
    figsize=(14,6),
    width=0.75
)

plt.title('Age Group Distribution by District')
plt.xlabel('District')
plt.ylabel('Number of Enrollments')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Age Groups')
plt.tight_layout()
plt.show()
```

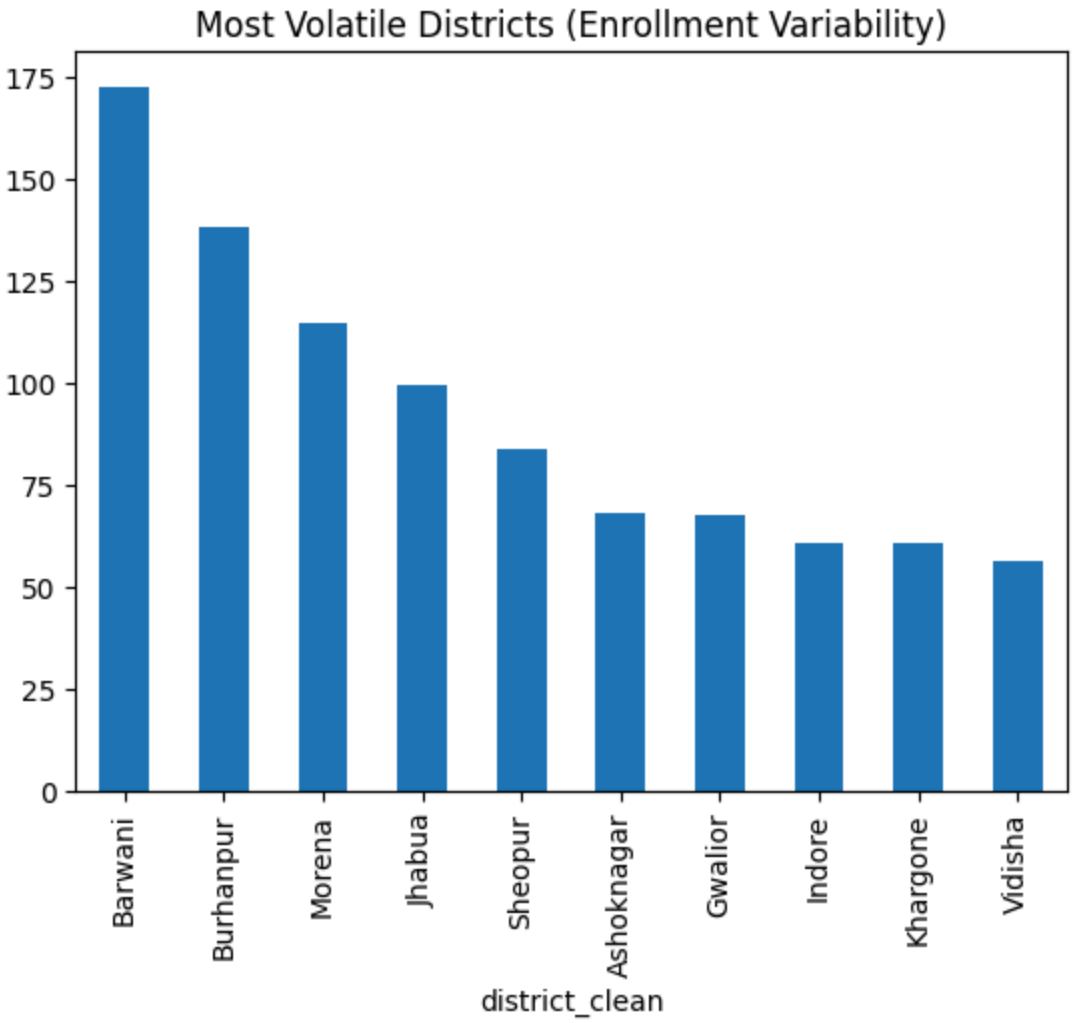


📊 Enrollment Variability Across Districts

This bar chart highlights the top 10 districts with the highest standard deviation in total enrollment, indicating significant variability over time.
High volatility suggests inconsistent enrollment patterns, possibly due to seasonal drives, migration, or administrative factors.

In [113…

```
district_std = df.groupby('district_clean')['total_enrollment'].std()

district_std.sort_values(ascending=False).head(10).plot(kind='bar')
plt.title('Most Volatile Districts (Enrollment Variability)')
plt.show()
```



📌 District-wise Enrollment Ranking Analysis

This analysis ranks districts based on their total Aadhaar enrollment over the entire study period.

In [113…

```
district_rank = df.groupby('district_clean')['total_enrollment'].sum() \
                  .rank(ascending=False)

district_rank.sort_values().head(10)
```

Out[113…
```
district_clean
Morena         1.0
Indore         2.0
Barwani        3.0
Gwalior        4.0
Chhatarpur     5.0
Khargone       6.0
Ujjain         7.0
Bhind          8.0
Sagar          9.0
Jhabua        10.0
Name: total_enrollment, dtype: float64
```

We performed univariate, bivariate, and multivariate analysis to study enrollment distribution across states, districts, and age groups. Advanced analysis included district-level volatility, age composition, time-based trends, and flag-based segmentation to uncover hidden patterns.

West Bengal

In [113…
```
# Extracting rows where state is West Bengal
df_West_Bengal= df[df['state_clean']=='West Bengal']
```

In [113…
```
# Total unique districts in West Bengal
df_West_Bengal['district'].nunique()
```

Out[113… 58

In [113…
```
# Same District have more than one spelling so we have to map them
df_West_Bengal['district'].unique()
```

Out[113…
```
array(['Coochbehar', 'Dinajpur Uttar', 'Darjeeling', 'Cooch Behar',
       'North 24 Parganas', 'Uttar Dinajpur', 'Jhargram', 'Nadia',
       'Jalpaiguri', 'Alipurduar', 'Malda', 'Kolkata', 'Dakshin Dinajpur',
       'Kalimpong', 'Birbhum', '24 Paraganas North', 'Medinipur West',
       'Purba Bardhaman', 'Hooghly', '24 Paraganas South', 'Howrah',
       'Dinajpur Dakshin', 'Bankura', 'Barddhaman', 'Bardhaman',
       'Darjiling', 'East Midnapore', 'Haora', 'Koch Bihar', 'Maldah',
       'Murshidabad', 'North Twenty Four Parganas', 'Paschim Bardhaman',
       'Paschim Medinipur', 'Purba Medinipur', 'Purulia', 'Puruliya',
       'South 24 Parganas', 'South Dinajpur',
       'South Twenty Four Parganas', 'West Midnapore', 'Hugli',
       'North Dinajpur', 'HOOGHLY', 'NADIA', 'HOWRAH', 'Hawrah', 'MALDA',
       'hooghly', 'Medinipur', 'East Midnapur', 'nadia', 'Hooghiy',
       'KOLKATA', 'West Medinipur', 'Burdwan', 'South 24 parganas',
       'South 24 Pargana'], dtype=object)
```

In [113…
```
df_West_Bengal['district_clean'] = (
    df_West_Bengal['district']
    .str.lower()
    .str.strip()
```

```
    .str.replace(r'\*', '', regex=True)
    .str.replace(r'\(.*?\)', '', regex=True)
    .str.replace(r'[^a-z\s]', '', regex=True)
)
```

Mapping District

In [113…

```
## District mapping west Bengal
wb_district_standard_map = {
    "coochbehar": "Cooch Behar",
    "cooch behar": "Cooch Behar",
    "koch bihar": "Cooch Behar",

    "dinajpur uttar": "Uttar Dinajpur",
    "uttar dinajpur": "Uttar Dinajpur",
    "north dinajpur": "Uttar Dinajpur",

    "dinajpur dakshin": "Dakshin Dinajpur",
    "dakshin dinajpur": "Dakshin Dinajpur",
    "south dinajpur": "Dakshin Dinajpur",

    "darjeeling": "Darjeeling",
    "darjiling": "Darjeeling",

    "kalimpong": "Kalimpong",

    "north 24 parganas": "North 24 Parganas",
    "24 paraganas north": "North 24 Parganas",
    "north twenty four parganas": "North 24 Parganas",

    "south 24 parganas": "South 24 Parganas",
    "south 24 pargana": "South 24 Parganas",
    "south 24 parganas": "South 24 Parganas",
    "24 paraganas south": "South 24 Parganas",
    "south twenty four parganas": "South 24 Parganas",

    "nadia": "Nadia",

    "jalpaiguri": "Jalpaiguri",
    "alipurduar": "Alipurduar",

    "malda": "Malda",
    "maldah": "Malda",

    "kolkata": "Kolkata",

    "jhargram": "Jhargram",
    "birbhum": "Birbhum",

    "medinipur west": "Paschim Medinipur",
    "west medinipur": "Paschim Medinipur",
    "west midnapore": "Paschim Medinipur",
    "paschim medinipur": "Paschim Medinipur",
    "medinipur": "Paschim Medinipur",
```

```
    "purba medinipur": "Purba Medinipur",
    "east midnapore": "Purba Medinipur",
    "east midnapur": "Purba Medinipur",

    "barddhaman": "Bardhaman",
    "bardhaman": "Bardhaman",
    "burdwan": "Bardhaman",

    "purba bardhaman": "Purba Bardhaman",
    "paschim bardhaman": "Paschim Bardhaman",

    "hooghly": "Hooghly",
    "hugli": "Hooghly",
    "hooghiy": "Hooghly",

    "howrah": "Howrah",
    "haora": "Howrah",
    "hawrah": "Howrah",

    "murshidabad": "Murshidabad",
    "bankura": "Bankura",

    "purulia": "Purulia",
    "puruliya": "Purulia"
}
```

In [113…
```
df['district_clean'] = (
df['district']
.apply(clean_name)
.map(wb_district_standard_map)
.fillna(df_West_Bengal['district'])
)
```

In [114…
```
## Remaining unmapped
df[df['district_clean'].isna()]['district'].unique()
# count check
df['district_clean'].nunique()
```

Out[114… 40

In [114…
```
df_West_Bengal = df[df['state_clean']=='West Bengal']
df_West_Bengal
```

Out[114…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	y
30	2025-03-09	West Bengal	Coochbehar	736135	19	12	19	2
32	2025-03-09	West Bengal	Dinajpur Uttar	733129	26	18	27	2
173	2025-04-01	West Bengal	Darjeeling	734010	81	49	17	2
199	2025-04-01	West Bengal	Cooch Behar	736135	243	127	20	2
208	2025-04-01	West Bengal	North 24 Parganas	700159	35	28	14	2
...	...	...	...	...	...	...	...	
1006024	2025-12-31	West Bengal	West Midnapore	721149	2	0	0	2
1006025	2025-12-31	West Bengal	West Midnapore	721150	2	2	0	2
1006026	2025-12-31	West Bengal	West Midnapore	721305	0	1	0	2
1006027	2025-12-31	West Bengal	West Midnapore	721504	1	0	0	2
1006028	2025-12-31	West Bengal	West Midnapore	721517	2	1	0	2

75371 rows × 13 columns

In [114…

```
#Check West Bengal-specific unmapped districts
df_West_Bengal_unmapped = df_West_Bengal[df_West_Bengal['district_clean'].is
print(f"Unmapped West Bengal districts count: {len(df_West_Bengal_unmapped)}
df_West_Bengal_unmapped['district'].unique()
```

```
Unmapped West Bengal districts count: 0
```

Out[114…

```
array([], dtype=object)
```

In [114…

```
# Final unique districts in West Bengal after cleaning
df_West_Bengal['district_clean'].unique()
```

Out[114…
```
array(['Cooch Behar', 'Dinajpur Uttar', 'Darjeeling', 'North 24 Parganas',
       'Uttar Dinajpur', 'Jhargram', 'Nadia', 'Jalpaiguri', 'Alipurduar',
       'Malda', 'Kolkata', 'Dakshin Dinajpur', 'Kalimpong', 'Birbhum',
       '24 Paraganas North', 'Medinipur West', 'Purba Bardhaman',
       'Hooghly', '24 Paraganas South', 'Howrah', 'Dinajpur Dakshin',
       'Bankura', 'Bardhaman', 'East Midnapore', 'Koch Bihar',
       'Murshidabad', 'North Twenty Four Parganas', 'Paschim Bardhaman',
       'Paschim Medinipur', 'Purba Medinipur', 'Purulia',
       'South 24 Parganas', 'South Dinajpur',
       'South Twenty Four Parganas', 'West Midnapore', 'North Dinajpur',
       'East Midnapur', 'West Medinipur', 'South 24 parganas',
       'South 24 Pargana'], dtype=object)
```

In [114…
```
df_West_Bengal['total_enrollment'] = (
    df_West_Bengal['age_0_5'] +
    df_West_Bengal['age_5_17'] +
    df_West_Bengal['age_18_greater']
)
```

In [114…
```
# Total unique districts in West Bengal after cleaning
df_West_Bengal['district_clean'].nunique()
```

Out[114… 40

In [114…
```
# unique pincodes in West Bengal
df_West_Bengal['pincode'].nunique()
```

Out[114… 1336

In [114…
```
# Check unique pincodes per district in West Bengal
pincode_check = df_West_Bengal.groupby('district_clean')['pincode'].nunique(
pincode_check
```

Out [114...

	district_clean	pincode
0	24 Paraganas North	23
1	24 Paraganas South	1
2	Alipurduar	25
3	Bankura	74
4	Bardhaman	167
5	Birbhum	49
6	Cooch Behar	45
7	Dakshin Dinajpur	27
8	Darjeeling	64
9	Dinajpur Dakshin	2
10	Dinajpur Uttar	6
11	East Midnapore	77
12	East Midnapur	1
13	Hooghly	106
14	Howrah	59
15	Jalpaiguri	71
16	Jhargram	46
17	Kalimpong	17
18	Koch Bihar	39
19	Kolkata	94
20	Malda	39
21	Medinipur West	2
22	Murshidabad	98
23	Nadia	79
24	North 24 Parganas	150
25	North Dinajpur	15
26	North Twenty Four Parganas	96
27	Paschim Bardhaman	83
28	Paschim Medinipur	95
29	Purba Bardhaman	91
30	Purba Medinipur	103
31	Purulia	43

	district_clean	pincode
32	South 24 Pargana	1
33	South 24 Parganas	92
34	South 24 parganas	1
35	South Dinajpur	20
36	South Twenty Four Parganas	64
37	Uttar Dinajpur	25
38	West Medinipur	1
39	West Midnapore	61

In [114…
```
# Pincode associated with multiple districts in Bihar
pin_district_count = (
    df_West_Bengal.groupby('pincode')['district_clean']
    .nunique()
    .reset_index(name='district_count')
)
```

In [114…
```
pin_district_count
```

Out[114…

	pincode	district_count
0	700001	1
1	700002	1
2	700003	1
3	700004	1
4	700005	1
…	…	…
1331	743702	1
1332	743704	2
1333	743710	2
1334	743711	3
1335	756084	1

1336 rows × 2 columns

In [115…
```
# Extract problematic pincodes (associated with >1 district)
problem_pins = pin_district_count[
pin_district_count['district_count'] > 1
]
```

In [115…
```
problem_pins
```

Out[115…

	pincode	district_count
7	700008	3
17	700018	2
23	700024	2
27	700028	2
29	700030	3
…	…	…
1328	743613	2
1330	743701	2
1332	743704	2
1333	743710	2
1334	743711	3

634 rows × 2 columns

In [115…
```
# Get all records with problematic pincodes(flagged records)
df_flagged = df_West_Bengal.merge(
problem_pins[['pincode']],
on='pincode',
how='inner'
)
```

In [115…
```
df_flagged
```

Out [115…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	yea
0	2025-03-09	West Bengal	Coochbehar	736135	19	12	19	202
1	2025-03-09	West Bengal	Dinajpur Uttar	733129	26	18	27	202
2	2025-04-01	West Bengal	Cooch Behar	736135	243	127	20	202
3	2025-04-01	West Bengal	North 24 Parganas	700159	35	28	14	202
4	2025-04-01	West Bengal	Uttar Dinajpur	733134	484	109	27	202
...	...	...	...	...	...	...	...	.
44031	2025-12-31	West Bengal	West Midnapore	721149	2	0	0	202
44032	2025-12-31	West Bengal	West Midnapore	721150	2	2	0	202
44033	2025-12-31	West Bengal	West Midnapore	721305	0	1	0	202
44034	2025-12-31	West Bengal	West Midnapore	721504	1	0	0	202
44035	2025-12-31	West Bengal	West Midnapore	721517	2	1	0	202

44036 rows × 13 columns

In [115…

```
# Summary of flagged records by district and pincode(for review)
flagged_pincode=df_flagged.groupby(['district_clean','pincode'])[['age_0_5',
# flagged_pincode.to_excel('flagged_pincode_domain.xlsx')
```

In [115…

```
# Add total enrollment column to flagged_pincode
flagged_pincode['total_enrollment']=flagged_pincode['age_0_5']+flagged_pincc
flagged_pincode
```

Out [115…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
0	24 Paraganas North	700049	55	67	12	134
1	24 Paraganas North	700056	34	53	11	98
2	24 Paraganas North	700102	76	67	14	157
3	24 Paraganas North	700110	114	127	25	266
4	24 Paraganas North	700119	116	132	13	261
...	...	...	...	...	...	...
1445	West Midnapore	721513	20	8	1	29
1446	West Midnapore	721515	1	0	0	1
1447	West Midnapore	721516	3	0	0	3
1448	West Midnapore	721517	52	11	0	63
1449	West Midnapore	721641	3	0	0	3

1450 rows × 6 columns

In [115…

```
idx = flagged_pincode.groupby('pincode')['total_enrollment'].idxmax()
df_filtered = flagged_pincode.loc[idx]
df_filtered
```

Out [115…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
1211	South 24 Parganas	700008	29	29	0	58
573	Kolkata	700018	167	258	9	434
574	Kolkata	700024	290	285	9	584
639	North 24 Parganas	700028	86	38	7	131
640	North 24 Parganas	700030	22	20	2	44
...	...	...	...	...	...	...
1286	South 24 Parganas	743613	37	13	0	50
753	North 24 Parganas	743701	100	15	2	117
754	North 24 Parganas	743704	183	40	12	235
755	North 24 Parganas	743710	191	35	14	240
756	North 24 Parganas	743711	287	53	3	343

634 rows × 6 columns

In [115…

```
# Creating a flag for pincodes associated with multiple districts
df_West_Bengal['pin_multi_district_flag']=(
df_West_Bengal.groupby('pincode')['district_clean']
.transform('nunique')>1
)
```

In [115…

```
# Creating a flag for pincodes associated with multiple districts
pin_district_map= (
    df_West_Bengal[df_West_Bengal['pin_multi_district_flag']]
    .groupby('pincode')['district_clean']  # noqa: SC100
    .unique()
    .reset_index()
)
```

In [115…

```
## monthly enrolment check
df_West_Bengal['month'] = df_West_Bengal['date'].dt.month.astype(str).str.zf
df_West_Bengal
```

Out [115…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater	
30	2025-03-09	West Bengal	Coochbehar	736135	19	12	19	2
32	2025-03-09	West Bengal	Dinajpur Uttar	733129	26	18	27	2
173	2025-04-01	West Bengal	Darjeeling	734010	81	49	17	2
199	2025-04-01	West Bengal	Cooch Behar	736135	243	127	20	2
208	2025-04-01	West Bengal	North 24 Parganas	700159	35	28	14	2
...	...	...	...	...	...	...	...	
1006024	2025-12-31	West Bengal	West Midnapore	721149	2	0	0	2
1006025	2025-12-31	West Bengal	West Midnapore	721150	2	2	0	2
1006026	2025-12-31	West Bengal	West Midnapore	721305	0	1	0	2
1006027	2025-12-31	West Bengal	West Midnapore	721504	1	0	0	2
1006028	2025-12-31	West Bengal	West Midnapore	721517	2	1	0	2

75371 rows × 14 columns

In [116…

```
# Dropping Date District and state as we have District clean State clean
df_West_Bengal_cleaned=df_West_Bengal.drop(columns=['date','district','state
df_West_Bengal_cleaned
```

Out [116…

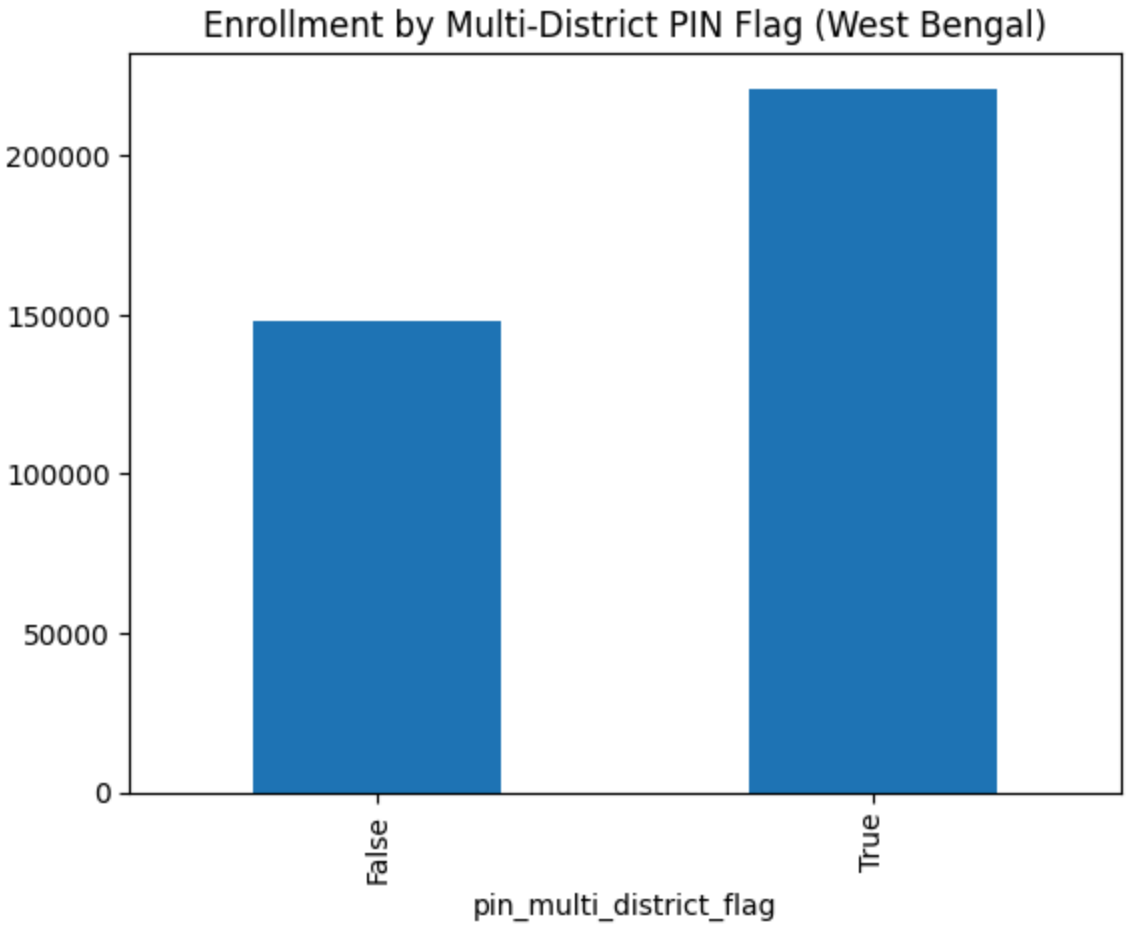
	pincode	age_0_5	age_5_17	age_18_greater	year	month	month_name	s
30	736135	19	12	19	2025	03	Mar	
32	733129	26	18	27	2025	03	Mar	
173	734010	81	49	17	2025	04	Apr	
199	736135	243	127	20	2025	04	Apr	
208	700159	35	28	14	2025	04	Apr	
...	...	...	...	...	...	...	...	
1006024	721149	2	0	0	2025	12	Dec	
1006025	721150	2	2	0	2025	12	Dec	
1006026	721305	0	1	0	2025	12	Dec	
1006027	721504	1	0	0	2025	12	Dec	
1006028	721517	2	1	0	2025	12	Dec	

75371 rows × 11 columns

Flagged vs unflagged pincodes

In [116…

```
df_West_Bengal.groupby('pin_multi_district_flag')['total_enrollment'].sum().
kind='bar'
)
plt.title('Enrollment by Multi-District PIN Flag (West Bengal)')
plt.show()
```



In [116…
```
# Aggregate West Bengal enrollment data at the district level by summing enr
# across different age groups (0–5, 5–17, and 18+).
df_West_Bengal_dist_level = df_West_Bengal_cleaned.groupby('district_clean')
['age_0_5', 'age_5_17', 'age_18_greater']
].sum()
# Compute total enrollment for each district by adding all age-group enrollm
df_West_Bengal_dist_level['total_enrollment'] = (
df_West_Bengal_dist_level['age_0_5'] +
df_West_Bengal_dist_level['age_5_17'] +
df_West_Bengal_dist_level['age_18_greater']
)
```

In [116…
```
df_West_Bengal_dist_level.shape
```

Out[116… (40, 4)

In [116…
```
df_West_Bengal_dist_level
```

Out [116...

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
24 Paraganas North	3177	2458	512	6147
24 Paraganas South	364	104	22	490
Alipurduar	1513	1706	438	3657
Bankura	8803	2073	150	11026
Bardhaman	13868	3367	17	17252
Birbhum	13115	2002	107	15224
Cooch Behar	9279	5272	429	14980
Dakshin Dinajpur	5257	1059	99	6415
Darjeeling	4439	2736	772	7947
Dinajpur Dakshin	707	256	34	997
Dinajpur Uttar	6478	4859	334	11671
East Midnapore	1750	315	4	2069
East Midnapur	1	0	0	1
Hooghly	11047	3842	342	15231
Howrah	9837	3961	146	13944
Jalpaiguri	5596	4398	537	10531
Jhargram	726	167	50	943
Kalimpong	168	183	153	504
Koch Bihar	2163	1090	10	3263
Kolkata	4927	4557	722	10206
Malda	12756	4456	84	17296
Medinipur West	325	300	20	645
Murshidabad	30593	4290	85	34968
Nadia	13096	3599	840	17535
North 24 Parganas	21205	5668	1347	28220
North Dinajpur	260	78	0	338
North Twenty Four Parganas	1202	393	2	1597
Paschim Bardhaman	856	317	93	1266
Paschim Medinipur	13984	3651	150	17785
Purba Bardhaman	2474	386	128	2988
Purba Medinipur	12760	1299	57	14116

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Purulia	10072	3620	134	13826
South 24 Pargana	1	1	0	2
South 24 Parganas	24659	8125	304	33088
South 24 parganas	2	0	0	2
South Dinajpur	366	68	0	434
South Twenty Four Parganas	2931	1113	4	4048
Uttar Dinajpur	18040	8039	366	26445
West Medinipur	1	1	0	2
West Midnapore	1621	526	3	2150

In [116…

```
# Sorting District with total enrollment
df_West_Bengal_dist_level.sort_values("total_enrollment",ascending=False).re
# df_West_Bengal_dist_level.to_excel('west_bengal_district_level_enrolment.x
```

Out[116...

	age_0_5	age_5_17	age_18_greater	total_enrollment
0	30593	4290	85	34968
1	24659	8125	304	33088
2	21205	5668	1347	28220
3	18040	8039	366	26445
4	13984	3651	150	17785
5	13096	3599	840	17535
6	12756	4456	84	17296
7	13868	3367	17	17252
8	11047	3842	342	15231
9	13115	2002	107	15224
10	9279	5272	429	14980
11	12760	1299	57	14116
12	9837	3961	146	13944
13	10072	3620	134	13826
14	6478	4859	334	11671
15	8803	2073	150	11026
16	5596	4398	537	10531
17	4927	4557	722	10206
18	4439	2736	772	7947
19	5257	1059	99	6415
20	3177	2458	512	6147
21	2931	1113	4	4048
22	1513	1706	438	3657
23	2163	1090	10	3263
24	2474	386	128	2988
25	1621	526	3	2150
26	1750	315	4	2069
27	1202	393	2	1597
28	856	317	93	1266
29	707	256	34	997
30	726	167	50	943
31	325	300	20	645

	age_0_5	age_5_17	age_18_greater	total_enrollment
32	168	183	153	504
33	364	104	22	490
34	366	68	0	434
35	260	78	0	338
36	1	1	0	2
37	2	0	0	2
38	1	1	0	2
39	1	0	0	1

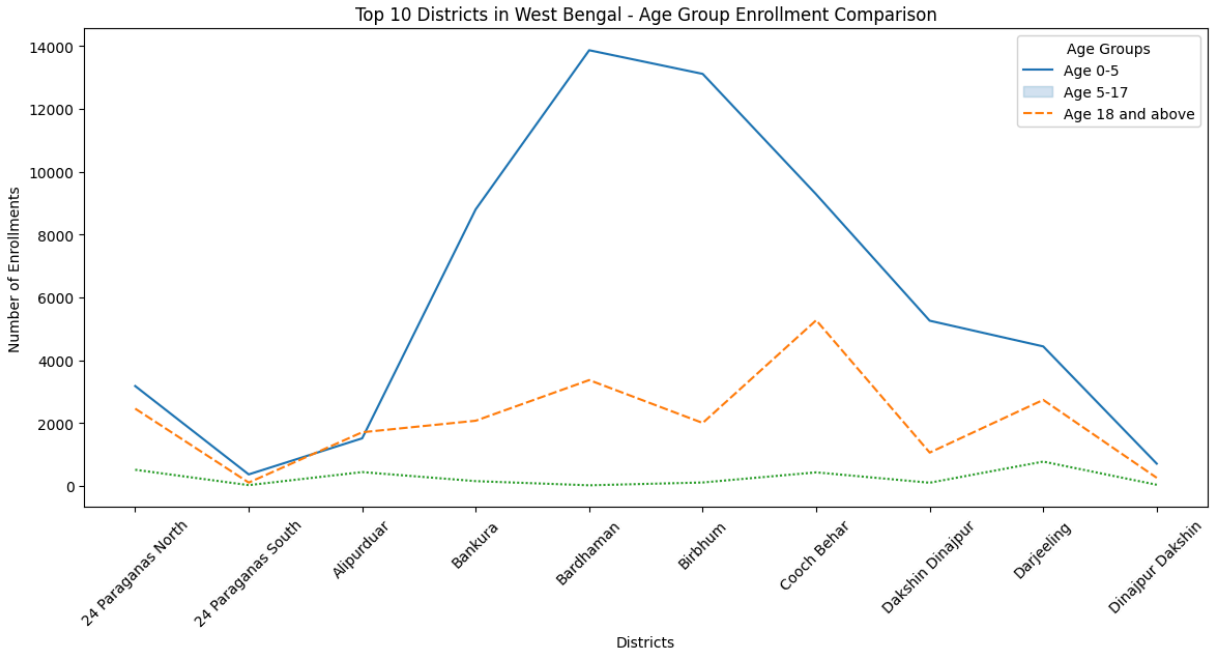
Plotting Top 10 Districts by number of Enrollment

Visualize age-wise enrollment distribution across top 10 Bihar districts using a line plot

In [116…
```
df_West_Bengal_dist_level1 = df_West_Bengal_dist_level.head(10)
```

In [116…
```
# Import required visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns
# Set the figure size for better readability
plt.figure(figsize=(14,6))
# Plot line chart to compare enrollment trends across age groups
# for the top 10 districts in Bihar
sns.lineplot(
data=df_West_Bengal_dist_level1[['age_0_5', 'age_5_17', 'age_18_greater']]
)
# Add title and axis labels
plt.title('Top 10 Districts in West Bengal - Age Group Enrollment Comparisor
plt.xlabel('Districts')
plt.ylabel('Number of Enrollments')
# Customize legend to clearly represent age groups
plt.legend(
title='Age Groups',
labels=['Age 0-5', 'Age 5-17', 'Age 18 and above']
)
# Set district names on x-axis and rotate labels for clarity
plt.xticks(
ticks=range(len(df_West_Bengal_dist_level1.index)),
labels=df_West_Bengal_dist_level1.index,
rotation=45
)
```

```
# Display the plot
plt.show()
```



Plotting month vs total enrollment

In [116…
```
## monthly enrolment trend in West Bengal
df_West_Bengal_monthly = df_West_Bengal_cleaned.groupby('month')[['age_0_5',
df_West_Bengal_monthly['total_enrollment'] = df_West_Bengal_monthly['age_0_5
```

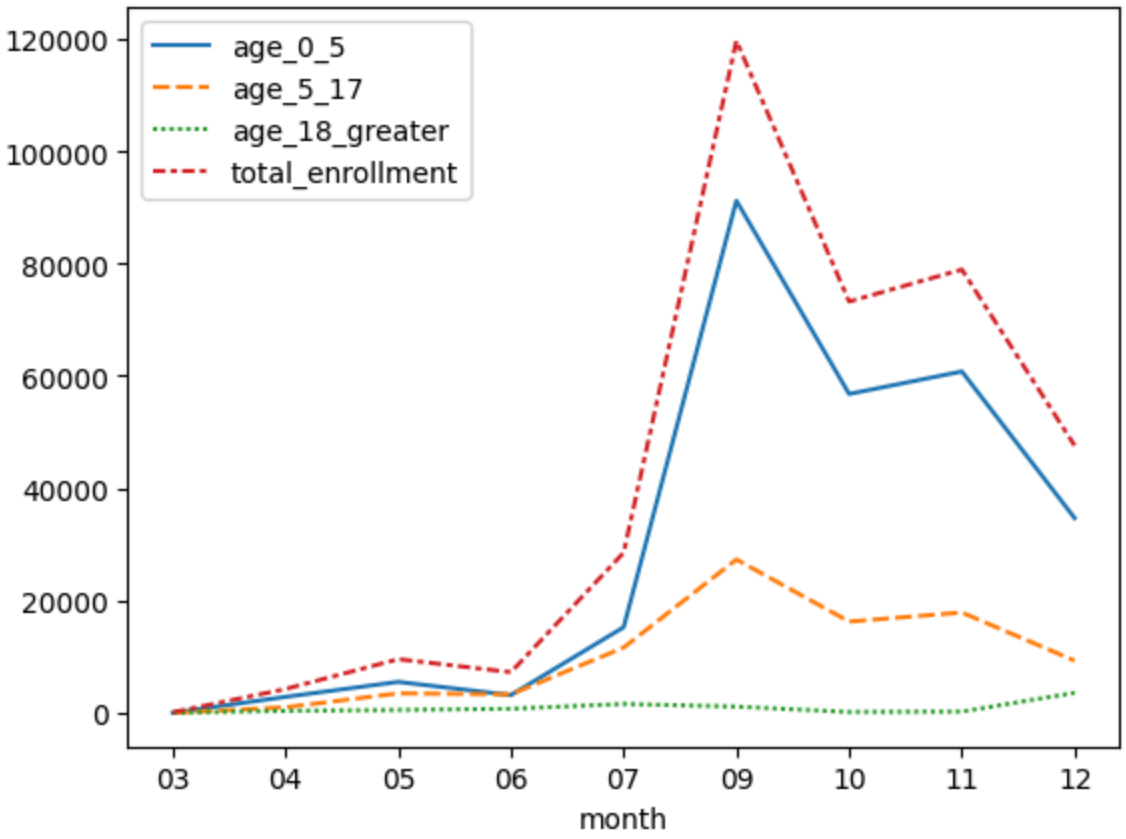
In [116…
```
df_West_Bengal_monthly.sort_values("total_enrollment",ascending=False).reset
```

Out[116…

	month	age_0_5	age_5_17	age_18_greater	total_enrollment
0	09	91195	27328	1134	119657
1	11	60799	17892	249	78940
2	10	56795	16271	184	73250
3	12	34706	9320	3590	47616
4	07	15274	11657	1613	28544
5	05	5538	3500	544	9582
6	06	3194	3323	754	7271
7	04	2873	1014	381	4268
8	03	45	30	46	121

In [117…
```
sns.lineplot(data=df_West_Bengal_monthly)
```

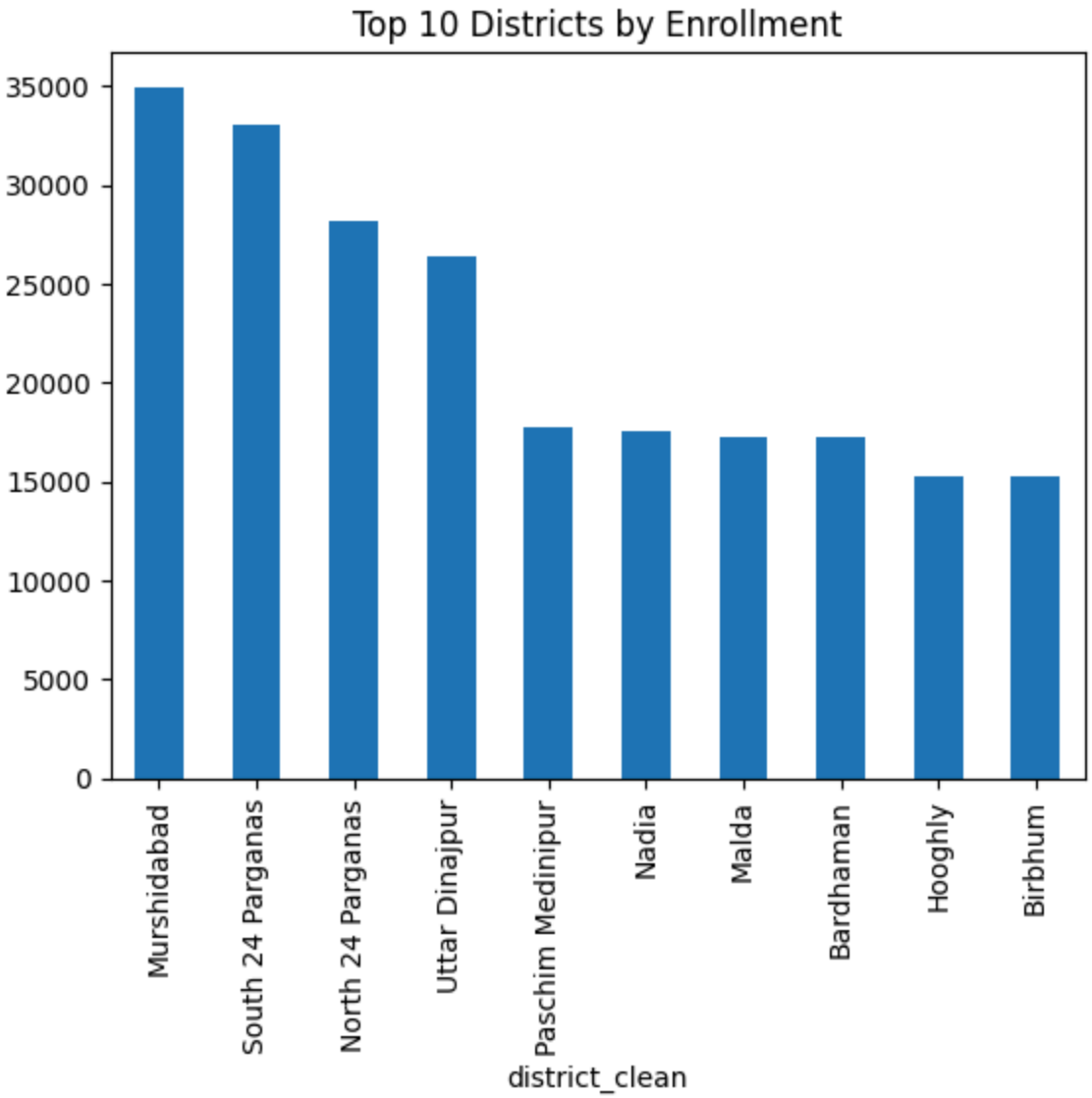
Out[117…
```
<Axes: xlabel='month'>
```



Top 10 Districts by Enrollment

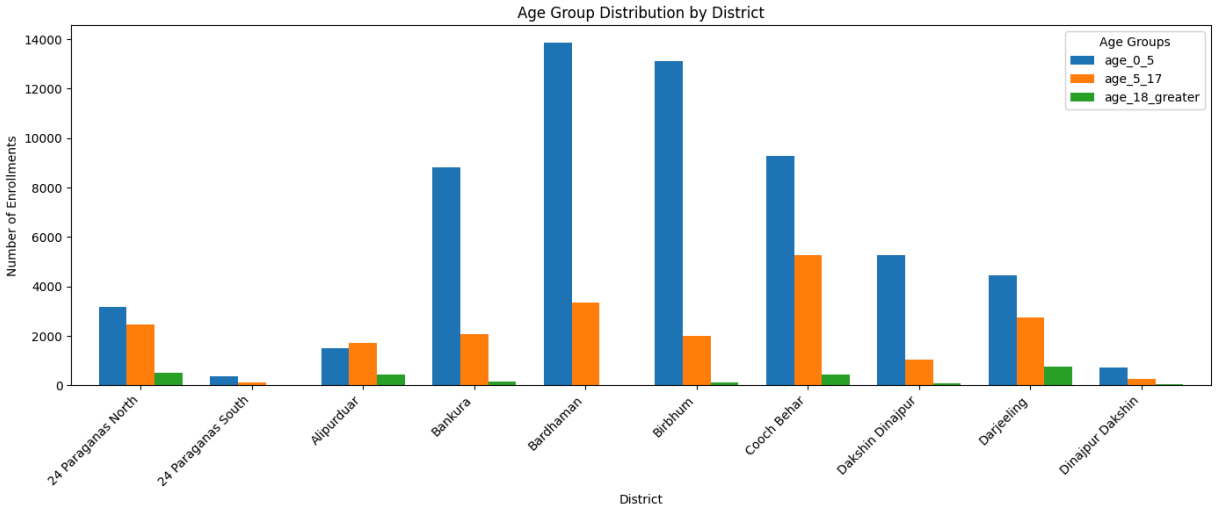
In [117…

```
top_districts = df.groupby('district_clean')['total_enrollment'] \
                  .sum().sort_values(ascending=False).head(10)

top_districts.plot(kind='bar')
plt.title('Top 10 Districts by Enrollment')
plt.show()
```



In [117…

```
import matplotlib.pyplot as plt

district_age = df.groupby('district_clean')[
    ['age_0_5','age_5_17','age_18_greater']
].sum().head(10)

ax = district_age.plot(
    kind='bar',
    stacked=False,      # important → no overlap
    figsize=(14,6),
    width=0.75
)

plt.title('Age Group Distribution by District')
plt.xlabel('District')
plt.ylabel('Number of Enrollments')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Age Groups')
plt.tight_layout()
plt.show()
```

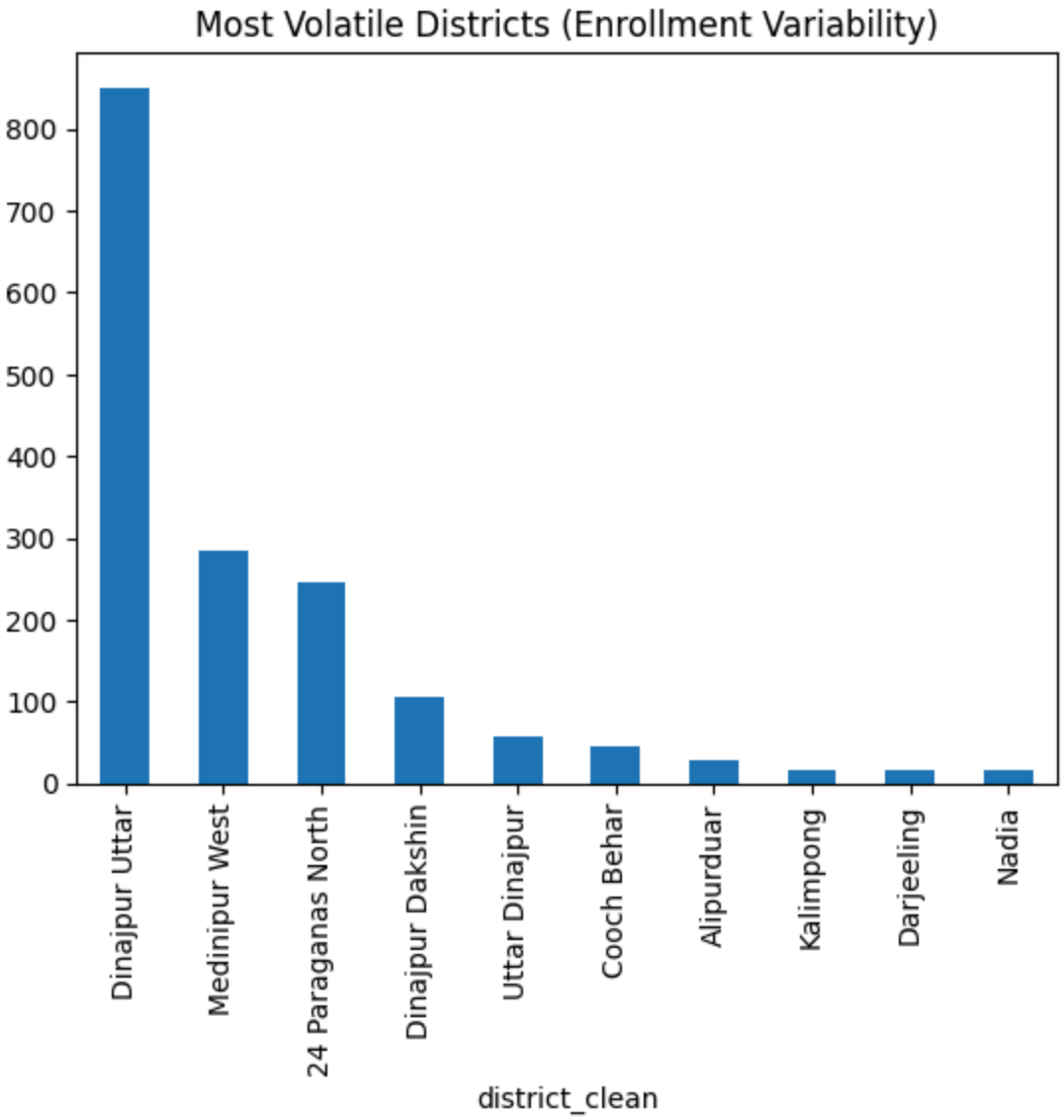


📊 Enrollment Variability Across Districts

This bar chart highlights the top 10 districts with the highest standard deviation in total enrollment, indicating significant variability over time. High volatility suggests inconsistent
enrollment patterns, possibly due to seasonal drives, migration, or administrative factors.

In [117…

```
district_std = df.groupby('district_clean')['total_enrollment'].std()

district_std.sort_values(ascending=False).head(10).plot(kind='bar')
plt.title('Most Volatile Districts (Enrollment Variability)')
plt.show()
```



📌 District-wise Enrollment Ranking Analysis

This analysis ranks districts based on their total Aadhaar enrollment over the entire study period.

In [117…

```
district_rank = df.groupby('district_clean')['total_enrollment'].sum() \
                  .rank(ascending=False)

district_rank.sort_values().head(10)
```

Out[117…
```
district_clean
Murshidabad           1.0
South 24 Parganas     2.0
North 24 Parganas     3.0
Uttar Dinajpur        4.0
Paschim Medinipur     5.0
Nadia                 6.0
Malda                 7.0
Bardhaman             8.0
Hooghly               9.0
Birbhum              10.0
Name: total_enrollment, dtype: float64
```

We performed univariate, bivariate, and multivariate analysis to study enrollment distribution across states, districts, and age groups. Advanced analysis included
district-level volatility, age composition, time-based trends, and flag-based segmentation to uncover hidden patterns.

Maharashtra

In [117…
```
# Extracting rows where state is Maharashtra
df_Maharashtra= df[df['state_clean']=='Maharashtra']
```

In [117…
```
# Total unique districts in
df_Maharashtra['district'].nunique()
```

Out[117… 53

In [117…
```
# Same District have more than one spelling so we have to map them
df_Maharashtra['district'].unique()
```

Out[117…
```
array(['Aurangabad', 'Parbhani', 'Thane', 'Nanded', 'Nagpur', 'Jalgaon',
       'Hingoli', 'Ahmadnagar', 'Palghar', 'Satara', 'Raigad',
       'Mumbai Suburban', 'Nandurbar', 'Beed', 'Chandrapur', 'Solapur',
       'Pune', 'Latur', 'Nashik', 'Yavatmal', 'Dhule', 'Washim', 'Sangli',
       'Buldhana', 'Amravati', 'Ahmednagar', 'Mumbai', 'Akola',
       'Osmanabad', 'Ahmed Nagar', 'Bhandara', 'Buldana',
       'Chhatrapati Sambhajinagar', 'Gadchiroli', 'Gondiya', 'Jalna',
       'Kolhapur', 'Mumbai City', 'Raigarh', 'Ratnagiri', 'Sindhudurg',
       'Wardha', 'Chatrapati Sambhaji Nagar', 'Mumbai( Sub Urban )',
       'Nandurbar *', 'Bid', 'Gondiya *', 'Dharashiv', 'Gondia',
       'Washim *', 'Raigarh(MH)', 'Hingoli *', 'Ahilyanagar'],
      dtype=object)
```

In [117…
```
df_Maharashtra['district_clean'] = (
    df_Maharashtra['district']
    .str.lower()
    .str.strip()
    .str.replace(r'\*', '', regex=True)
    .str.replace(r'\(.*?\)', '', regex=True)
```

```
    .str.replace(r'[^a-z\s]', '', regex=True)
)
```

Mapping District

In [117…

```
## District mapping Maharashtra
mh_district_standard_map = {
    "aurangabad": "Chhatrapati Sambhajinagar",
    "chhatrapati sambhajinagar": "Chhatrapati Sambhajinagar",
    "chatrapati sambhaji nagar": "Chhatrapati Sambhajinagar",

    "ahilyanagar": "Ahmednagar",
    "ahmadnagar": "Ahmednagar",
    "ahmednagar": "Ahmednagar",
    "ahmed nagar": "Ahmednagar",

    "bid": "Beed",
    "beed": "Beed",

    "buldana": "Buldhana",
    "buldhana": "Buldhana",

    "osmanabad": "Dharashiv",
    "dharashiv": "Dharashiv",

    "gondiya": "Gondia",
    "gondiya *": "Gondia",
    "gondia": "Gondia",

    "raigarh(mh)": "Raigad",
    "raigad": "Raigad",

    "nandurbar *": "Nandurbar",
    "nandurbar": "Nandurbar",

    "hingoli *": "Hingoli",
    "hingoli": "Hingoli",

    "washim *": "Washim",
    "washim": "Washim",

    "mumbai suburban": "Mumbai Suburban",
    "mumbai( sub urban )": "Mumbai Suburban",

    "mumbai": "Mumbai City",
    "mumbai city": "Mumbai City"
}
```

In [118…

```
df['district_clean'] = (
    df['district']
    .apply(clean_name)
    .map(mh_district_standard_map)
```

```
    .fillna(df_Maharashtra['district'])
)
```

In [118…
```
## Remaining unmapped
df[df['district_clean'].isna()]['district'].unique()
# count check
df['district_clean'].nunique()
```

Out[118… 40

In [118…
```
df_Maharashtra = df[df['state_clean']=='Maharashtra']
df_Maharashtra
```

Out[118…

	date	state	district	pincode	age_0_5	age_5_17	age_18_great
12	2025-03-09	Maharashtra	Aurangabad	431001	42	46	
43	2025-03-15	Maharashtra	Parbhani	431401	17	14	
67	2025-03-20	Maharashtra	Thane	421503	19	13	
78	2025-03-20	Maharashtra	Aurangabad	431001	134	38	
151	2025-03-27	Maharashtra	Aurangabad	431001	20	19	
...	...	...	...	...	...	...	
1004434	2025-12-31	Maharashtra	Yavatmal	445105	2	1	
1004435	2025-12-31	Maharashtra	Yavatmal	445109	6	1	
1004436	2025-12-31	Maharashtra	Yavatmal	445205	4	2	
1004437	2025-12-31	Maharashtra	Yavatmal	445211	1	0	
1004438	2025-12-31	Maharashtra	Yavatmal	445401	2	1	

75531 rows × 13 columns

In [118…
```
# Check Bihar-specific unmapped districts
df_Maharashtra_unmapped = df_Maharashtra[df_Maharashtra['district_clean'].is
print(f"Unmapped Maharashtra districts count: {len(df_Maharashtra_unmapped)}
df_Maharashtra_unmapped['district'].unique()
```

```
Unmapped Maharashtra districts count: 0
```

Out[118… array([], dtype=object)

In [118…
```
# Final unique districts in Maharashtra after cleaning
df_Maharashtra['district_clean'].unique()
```

Out[118…
```
array(['Chhatrapati Sambhajinagar', 'Parbhani', 'Thane', 'Nanded',
       'Nagpur', 'Jalgaon', 'Hingoli', 'Ahmednagar', 'Palghar', 'Satara',
       'Raigad', 'Mumbai Suburban', 'Nandurbar', 'Beed', 'Chandrapur',
       'Solapur', 'Pune', 'Latur', 'Nashik', 'Yavatmal', 'Dhule',
       'Washim', 'Sangli', 'Buldhana', 'Amravati', 'Mumbai City', 'Akola',
       'Dharashiv', 'Bhandara', 'Gadchiroli', 'Gondia', 'Jalna',
       'Kolhapur', 'Raigarh', 'Ratnagiri', 'Sindhudurg', 'Wardha',
       'Chatrapati Sambhaji Nagar', 'Mumbai( Sub Urban )', 'Raigarh(MH)'],
      dtype=object)
```

In [118…
```
# Total unique districts in Maharashtra after cleaning
df_Maharashtra['district_clean'].nunique()
```

Out[118… 40

In [118…
```
# unique pincodes in Maharashtra
df_Maharashtra['pincode'].nunique()
```

Out[118… 1580

In [118…
```
# Check unique pincodes per district in Maharashtra
pincode_check = df_Maharashtra.groupby('district_clean')['pincode'].nunique(
pincode_check
```

Out [118...

	district_clean	unique_pincodes
0	Ahmednagar	90
1	Akola	29
2	Amravati	42
3	Beed	28
4	Bhandara	21
5	Buldhana	32
6	Chandrapur	33
7	Chatrapati Sambhaji Nagar	3
8	Chhatrapati Sambhajinagar	44
9	Dharashiv	28
10	Dhule	29
11	Gadchiroli	15
12	Gondia	14
13	Hingoli	14
14	Jalgaon	62
15	Jalna	26
16	Kolhapur	78
17	Latur	24
18	Mumbai City	109
19	Mumbai Suburban	55
20	Mumbai(Sub Urban)	47
21	Nagpur	67
22	Nanded	44
23	Nandurbar	20
24	Nashik	76
25	Palghar	37
26	Parbhani	19
27	Pune	147
28	Raigad	58
29	Raigarh	62
30	Raigarh(MH)	14
31	Ratnagiri	72

	district_clean	unique_pincodes
32	Sangli	66
33	Satara	79
34	Sindhudurg	52
35	Solapur	71
36	Thane	93
37	Wardha	17
38	Washim	14
39	Yavatmal	33

In [118…
```
# Pincode associated with multiple districts in Maharashtra
pin_district_count = (
    df_Maharashtra.groupby('pincode')['district_clean']
    .nunique()
    .reset_index(name='district_count')
)
```

In [118…
```
# Extract problematic pincodes (associated with >1 district)
problem_pins = pin_district_count[
    pin_district_count['district_count'] > 1
]
```

In [119…
```
problem_pins
```

Out[119…

	pincode	district_count
0	400001	2
11	400012	2
22	400024	3
27	400029	2
31	400033	2
...	...	...
1472	444108	2
1474	444110	2
1494	444405	2
1497	444501	2
1499	444503	2

213 rows × 2 columns

In [119…
```
# Get all records with problematic pincodes(flagged records)
df_flagged = df_Maharashtra.merge(
    problem_pins[['pincode']],
    on='pincode',
    how='inner'
)
```

In [119…
```
df_flagged
```

Out[119…

	date	state	district	pincode	age_0_5	age_5_17	age_18_greater
0	2025-03-09	Maharashtra	Aurangabad	431001	42	46	12
1	2025-03-20	Maharashtra	Aurangabad	431001	134	38	18
2	2025-03-27	Maharashtra	Aurangabad	431001	20	19	16
3	2025-04-01	Maharashtra	Hingoli	431512	243	22	33
4	2025-04-01	Maharashtra	Palghar	401209	174	90	16
…	…	…	…	…	…	…	…
16261	2025-12-31	Maharashtra	Thane	401209	4	5	0
16262	2025-12-31	Maharashtra	Thane	401302	0	1	0
16263	2025-12-31	Maharashtra	Thane	401404	1	0	0
16264	2025-12-31	Maharashtra	Thane	421303	2	0	0
16265	2025-12-31	Maharashtra	Wardha	442302	3	0	0

16266 rows × 13 columns

In [119…
```
# Summary of flagged records by district and pincode(for review)
flagged_pincode=df_flagged.groupby(['district_clean','pincode'])[['age_0_5',
# flagged_pincode.to_excel('flagged_pincode_domain.xlsx')
```

In [119…
```
# Add total enrollment column to flagged_pincode
flagged_pincode['total_enrollment']=flagged_pincode['age_0_5']+flagged_pincc
flagged_pincode
```

Out[119…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
0	Ahmednagar	400037	2	2	0	4
1	Ahmednagar	412210	32	5	0	37
2	Akola	444105	3	0	0	3
3	Akola	444107	151	20	0	171
4	Akola	444108	177	32	1	210
...	...	...	...	...	...	...
492	Washim	444105	352	28	5	385
493	Washim	444107	32	3	0	35
494	Washim	444110	77	5	4	86
495	Washim	444405	3	1	0	4
496	Washim	444503	262	20	4	286

497 rows × 6 columns

In [119…

```
idx = flagged_pincode.groupby('pincode')['total_enrollment'].idxmax()
df_filtered = flagged_pincode.loc[idx]
df_filtered
```

Out[119…

	district_clean	pincode	age_0_5	age_5_17	age_18_greater	total_enrollment
77	Mumbai City	400001	43	13	6	62
78	Mumbai City	400012	245	21	9	275
154	Mumbai Suburban	400024	51	12	2	65
80	Mumbai City	400029	39	9	0	48
81	Mumbai City	400033	128	26	2	156
...	...	...	...	...	...	...
4	Akola	444108	177	32	1	210
494	Washim	444110	77	5	4	86
6	Akola	444405	55	2	0	57
7	Akola	444501	252	23	3	278
496	Washim	444503	262	20	4	286

213 rows × 6 columns

In [119…

```
# Creating a flag for pincodes associated with multiple districts
df_Maharashtra['pin_multi_district_flag']=(
    df_Maharashtra.groupby('pincode')['district_clean']
```

```
    .transform('nunique')>1
)
```

In [119…

```
# Creating a flag for pincodes associated with multiple districts
pin_district_map= (
    df_Maharashtra[df_Maharashtra['pin_multi_district_flag']]
    .groupby('pincode')['district_clean']  # noqa: SC100
    .unique()
    .reset_index()
)
```

In [119…

```
## monthly enrolment check
df_Maharashtra['month'] = df_Maharashtra['date'].dt.month.astype(str).str.zf
df_Maharashtra
```

Out[119…

	date	state	district	pincode	age_0_5	age_5_17	age_18_great
12	2025-03-09	Maharashtra	Aurangabad	431001	42	46	
43	2025-03-15	Maharashtra	Parbhani	431401	17	14	
67	2025-03-20	Maharashtra	Thane	421503	19	13	
78	2025-03-20	Maharashtra	Aurangabad	431001	134	38	
151	2025-03-27	Maharashtra	Aurangabad	431001	20	19	
...	...	...	...	...	...	...	
1004434	2025-12-31	Maharashtra	Yavatmal	445105	2	1	
1004435	2025-12-31	Maharashtra	Yavatmal	445109	6	1	
1004436	2025-12-31	Maharashtra	Yavatmal	445205	4	2	
1004437	2025-12-31	Maharashtra	Yavatmal	445211	1	0	
1004438	2025-12-31	Maharashtra	Yavatmal	445401	2	1	

75531 rows × 14 columns

In [119…

```
# Dropping Date District and state as we have District clean State clean
df_Maharashtra_cleaned=df_Maharashtra.drop(columns=['date','district','state
df_Maharashtra_cleaned
```

Out [119…

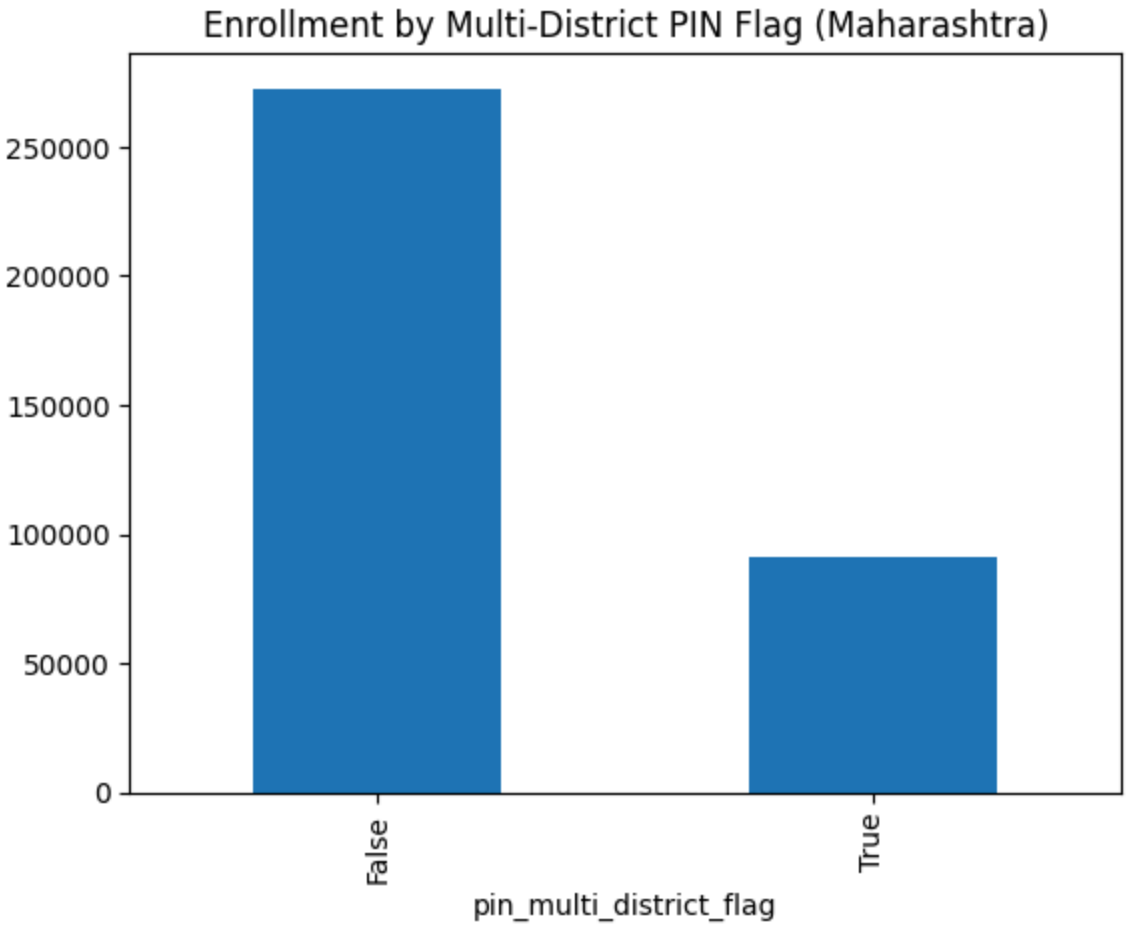
	pincode	age_0_5	age_5_17	age_18_greater	year	month	month_name	
12	431001	42	46	12	2025	03	Mar	I
43	431401	17	14	37	2025	03	Mar	I
67	421503	19	13	15	2025	03	Mar	I
78	431001	134	38	18	2025	03	Mar	I
151	431001	20	19	16	2025	03	Mar	I
…	…	…	…	…	…	…	…	…
1004434	445105	2	1	0	2025	12	Dec	I
1004435	445109	6	1	0	2025	12	Dec	I
1004436	445205	4	2	0	2025	12	Dec	I
1004437	445211	1	0	0	2025	12	Dec	I
1004438	445401	2	1	0	2025	12	Dec	I

75531 rows × 11 columns

Flagged vs unflagged pincodes

In [120…

```
df_Maharashtra.groupby('pin_multi_district_flag')['total_enrollment'].sum().
    kind='bar'
)
plt.title('Enrollment by Multi-District PIN Flag (Maharashtra)')
plt.show()
```



In [120…
```
# Aggregate Maharashtra enrollment data at the district level by summing enr
# across different age groups (0–5, 5–17, and 18+).
df_maharashtra_dist_level = df_Maharashtra_cleaned.groupby('district_clean')
    ['age_0_5', 'age_5_17', 'age_18_greater']
].sum()

# Compute total enrollment for each district by adding all age-group enrollm
df_maharashtra_dist_level['total_enrollment'] = (
    df_maharashtra_dist_level['age_0_5'] +
    df_maharashtra_dist_level['age_5_17'] +
    df_maharashtra_dist_level['age_18_greater']
)
```

In [120…
```
df_maharashtra_dist_level.shape
```

Out[120… (40, 4)

In [120…
```
df_maharashtra_dist_level
```

Out[120...

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Ahmednagar	9697	2514	153	12364
Akola	3922	691	49	4662
Amravati	5700	1009	107	6816
Beed	7823	2569	270	10662
Bhandara	1801	55	6	1862
Buldhana	6713	1189	141	8043
Chandrapur	3965	371	86	4422
Chatrapati Sambhaji Nagar	114	81	16	211
Chhatrapati Sambhajinagar	12602	5162	375	18139
Dharashiv	3270	721	37	4028
Dhule	7191	4804	516	12511
Gadchiroli	2054	265	22	2341
Gondia	2799	109	6	2914
Hingoli	6327	1449	370	8146
Jalgaon	10346	2560	223	13129
Jalna	4564	930	18	5512
Kolhapur	6324	970	69	7363
Latur	5709	1605	148	7462
Mumbai City	12705	4249	236	17190
Mumbai Suburban	12320	4882	779	17981
Mumbai(Sub Urban)	334	77	0	411
Nagpur	9943	1487	229	11659
Nanded	9010	2427	330	11767
Nandurbar	7688	2652	237	10577
Nashik	15984	5731	304	22019
Palghar	7220	3157	162	10539
Parbhani	4961	1000	198	6159
Pune	23622	6404	1122	31148
Raigad	2548	730	102	3380
Raigarh	4318	800	1	5119
Raigarh(MH)	70	15	0	85

district_clean	age_0_5	age_5_17	age_18_greater	total_enrollment
Ratnagiri	2267	213	32	2512
Sangli	5502	1781	132	7415
Satara	5005	707	153	5865
Sindhudurg	1088	55	20	1163
Solapur	9324	2454	249	12027
Thane	28692	13492	958	43142
Wardha	1800	83	5	1888
Washim	3000	299	44	3343
Yavatmal	5952	1320	198	7470

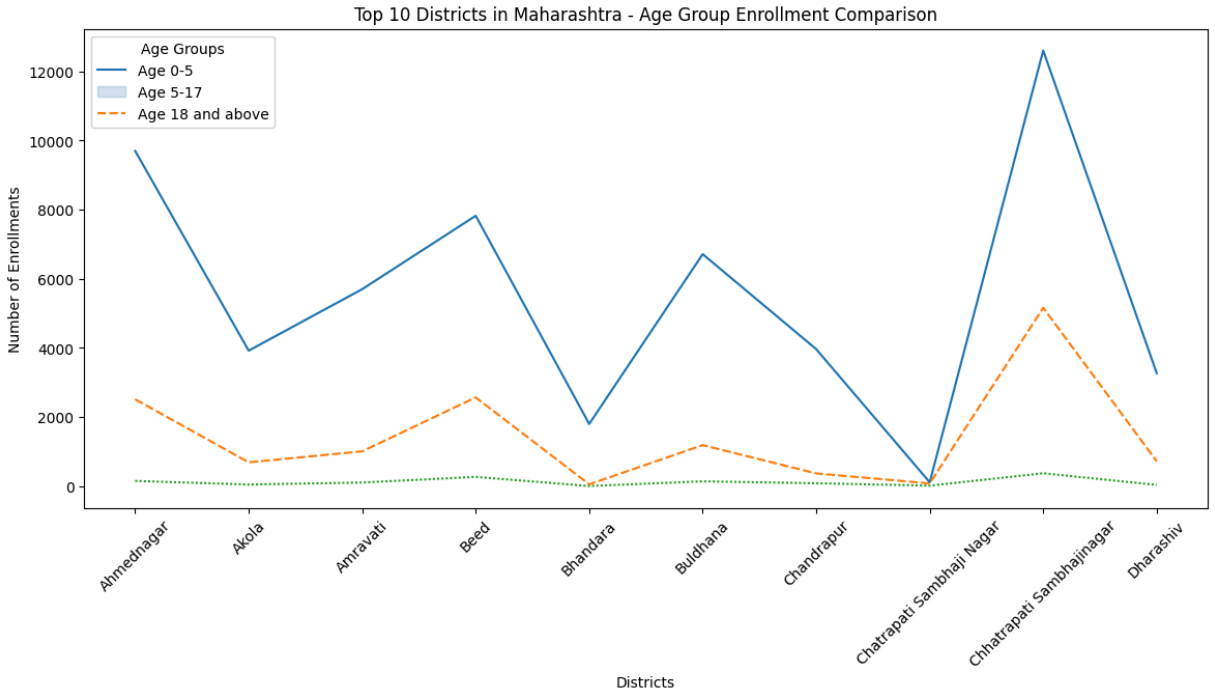
Plotting Top 10 Districts by number of Enrollment

Visualize age-wise enrollment distribution across top Maharashtra districts using a line plot

In [120…
```
df_maharashtra_dist_level1 = df_maharashtra_dist_level.head(10)
```

In [120…
```
# Import required visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Set the figure size for better readability
plt.figure(figsize=(14,6))

# Plot line chart to compare enrollment trends across age groups
# for the top 10 districts in Maharashtra
sns.lineplot(
    data=df_maharashtra_dist_level1[['age_0_5', 'age_5_17', 'age_18_greater'
)

# Add title and axis labels
plt.title('Top 10 Districts in Maharashtra - Age Group Enrollment Comparisor
plt.xlabel('Districts')
plt.ylabel('Number of Enrollments')

# Customize legend to clearly represent age groups
plt.legend(
    title='Age Groups',
    labels=['Age 0-5', 'Age 5-17', 'Age 18 and above']
)
```

```
# Set district names on x-axis and rotate labels for clarity
plt.xticks(
    ticks=range(len(df_maharashtra_dist_level1.index)),
    labels=df_maharashtra_dist_level1.index,
    rotation=45
)

# Display the plot
plt.show()
```



Plotting month vs total enrollment

In [120…
```
## monthly enrolment trend in maharashtra
df_maharashtra_monthly = df_Maharashtra_cleaned.groupby('month')[['age_0_5',
df_maharashtra_monthly['total_enrollment'] = df_maharashtra_monthly['age_0_5
```

In [120…
```
df_maharashtra_monthly.sort_values("total_enrollment",ascending=False).reset
```

Out[120…

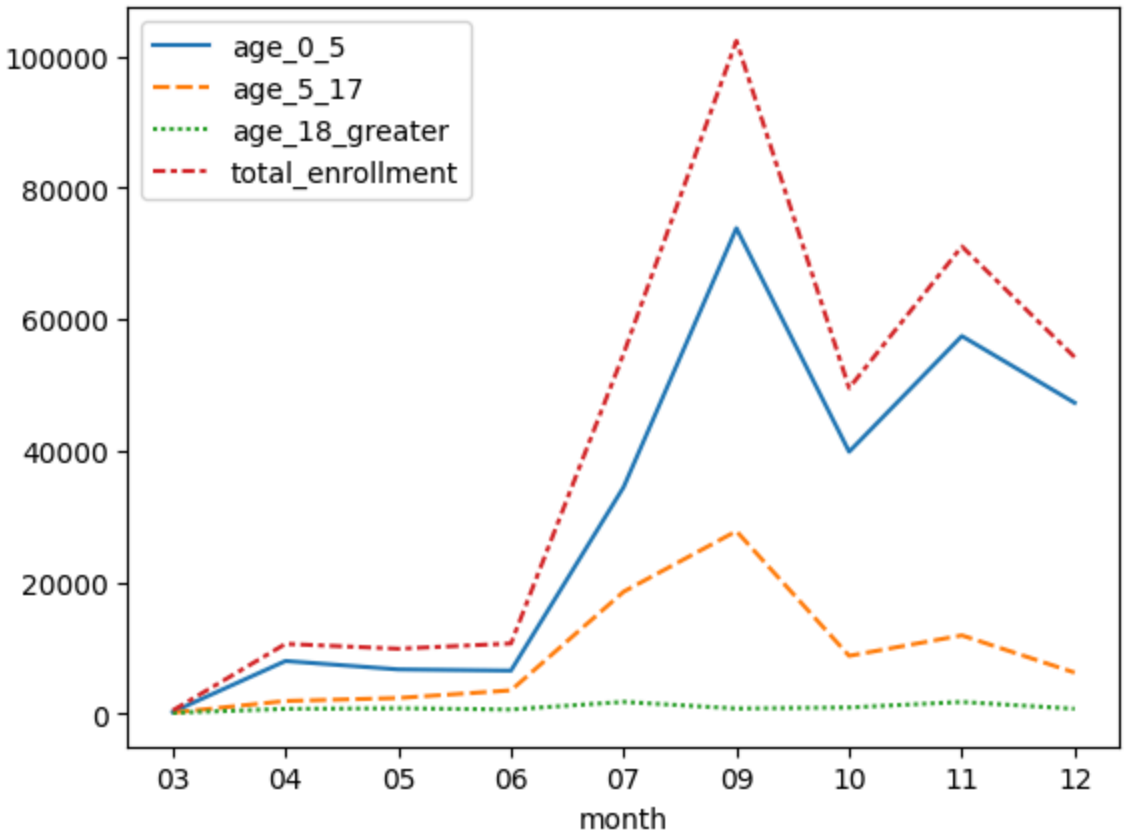
	month	age_0_5	age_5_17	age_18_greater	total_enrollment
0	09	73837	27751	753	102341
1	11	57425	11885	1753	71063
2	07	34468	18536	1763	54767
3	12	47273	6224	719	54216
4	10	39826	8772	912	49510
5	06	6521	3518	610	10649
6	04	7986	1901	704	10591
7	05	6706	2352	791	9849
8	03	232	130	98	460

In [120…

```
sns.lineplot(data=df_maharashtra_monthly)
```

Out[120…

```
<Axes: xlabel='month'>
```

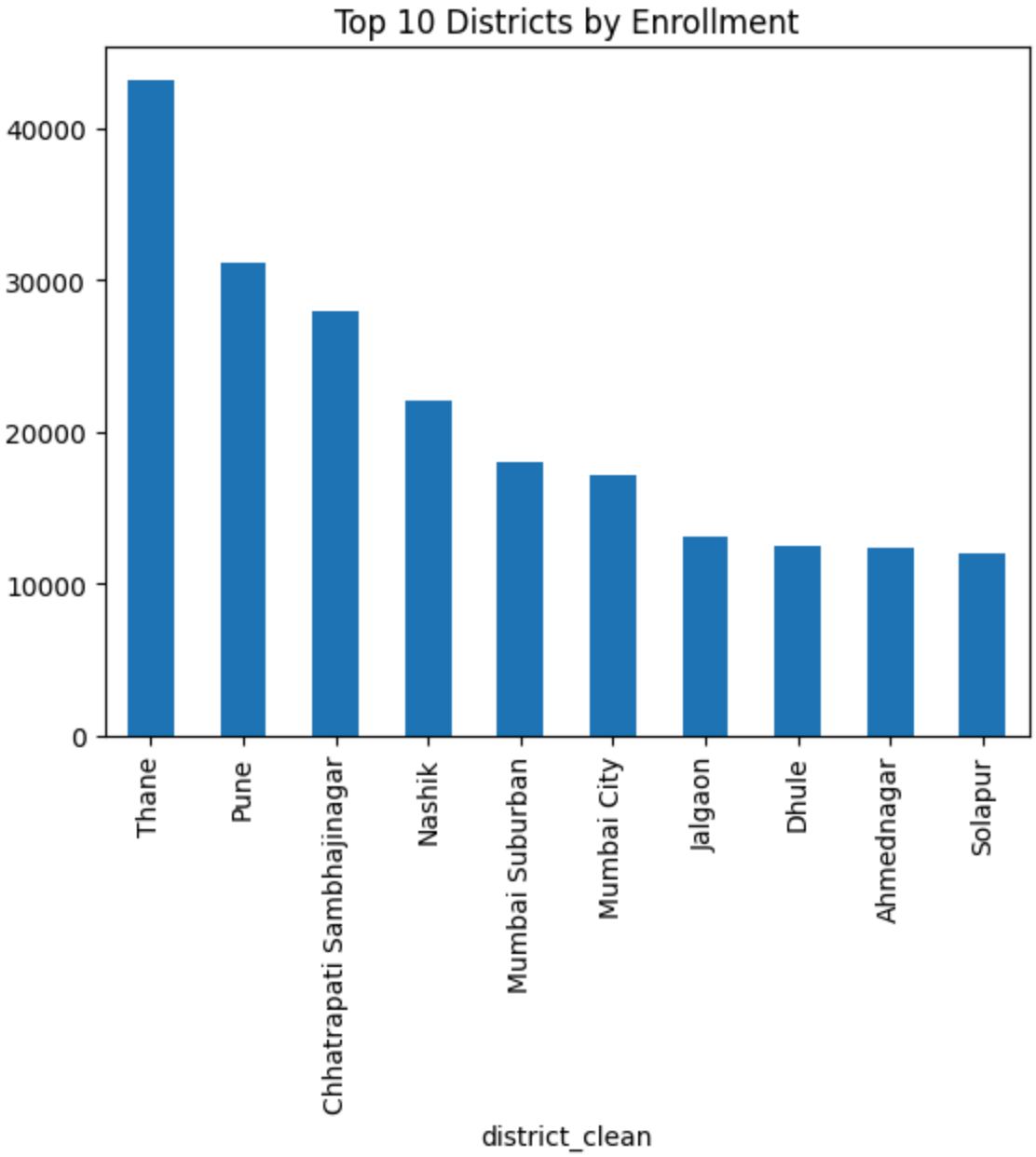


Top 10 Districts by Enrollment

In [120…

```
top_districts = df.groupby('district_clean')['total_enrollment'] \
                  .sum().sort_values(ascending=False).head(10)
```

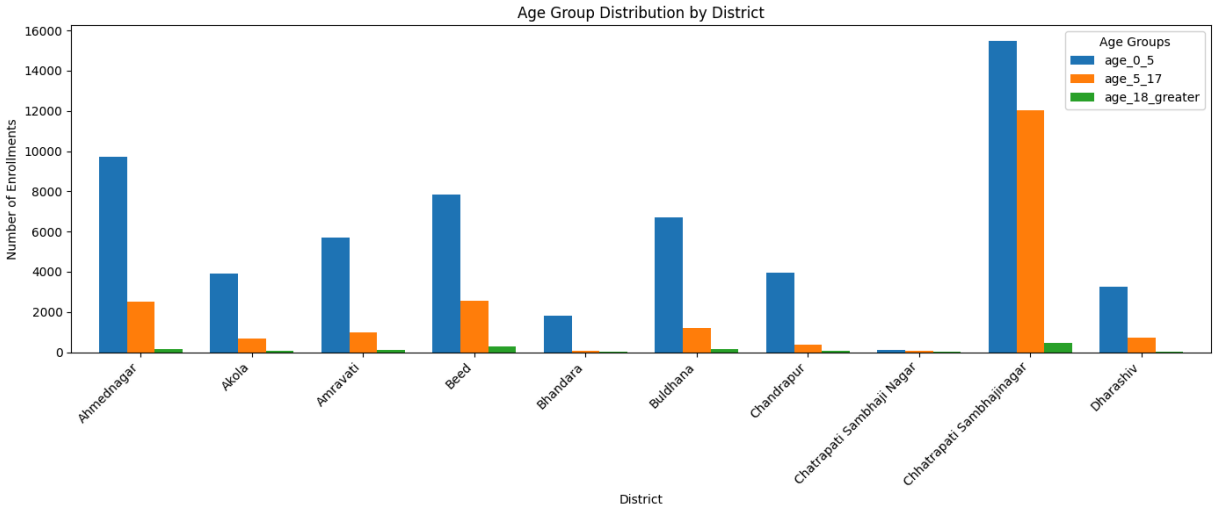
```
top_districts.plot(kind='bar')
plt.title('Top 10 Districts by Enrollment')
plt.show()
```



In [121…

```
import matplotlib.pyplot as plt

district_age = df.groupby('district_clean')[
    ['age_0_5','age_5_17','age_18_greater']
].sum().head(10)

ax = district_age.plot(
    kind='bar',
    stacked=False,      # important → no overlap
    figsize=(14,6),
    width=0.75
)

plt.title('Age Group Distribution by District')
```

```
plt.xlabel('District')
plt.ylabel('Number of Enrollments')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Age Groups')
plt.tight_layout()
plt.show()
```

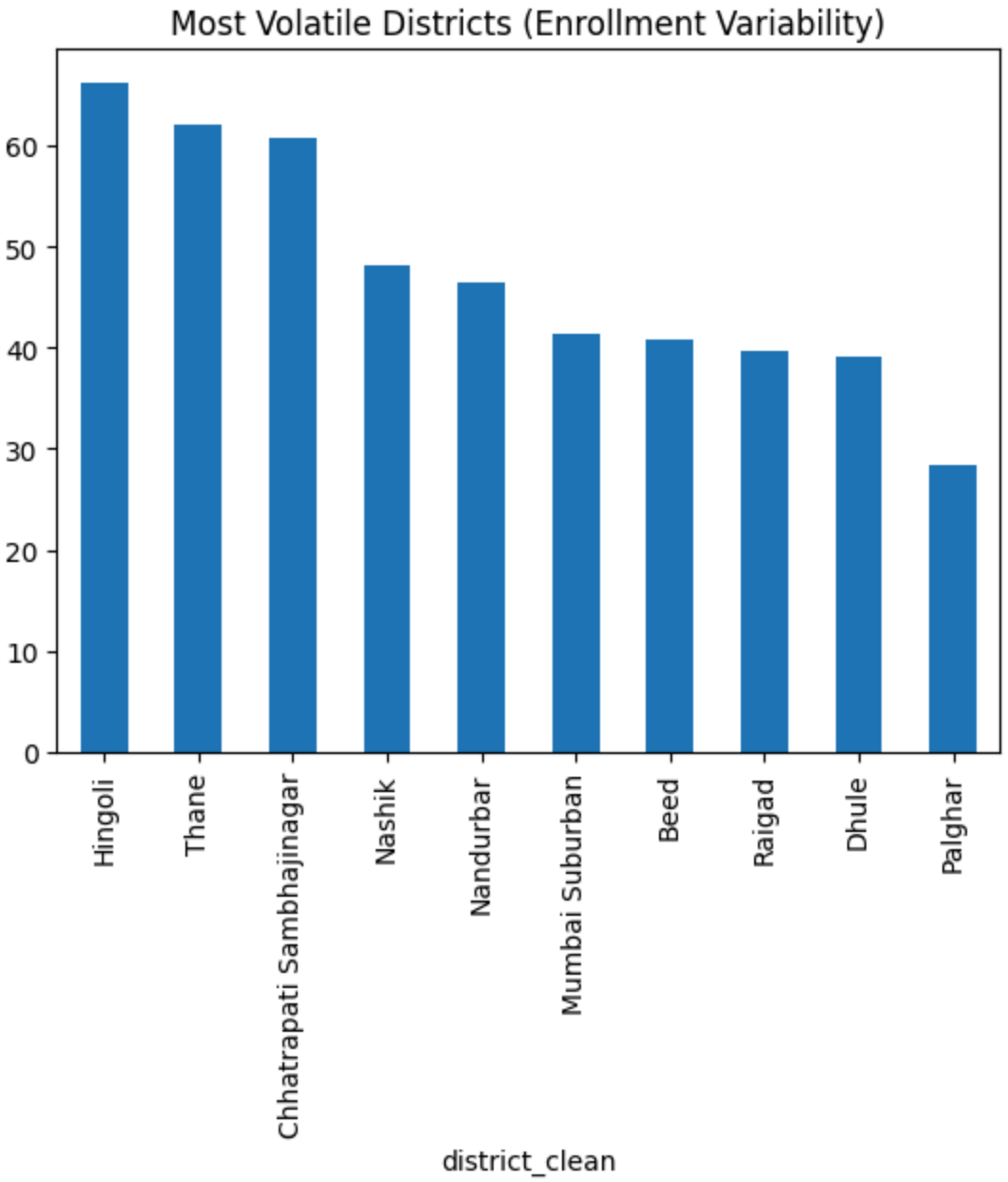


📊 Enrollment Variability Across Districts

This bar chart highlights the top 10 districts with the highest standard deviation in total enrollment, indicating significant variability over time. High volatility suggests
inconsistent enrollment patterns, possibly due to seasonal drives, migration, or administrative factors.

In [121…

```
district_std = df.groupby('district_clean')['total_enrollment'].std()

district_std.sort_values(ascending=False).head(10).plot(kind='bar')
plt.title('Most Volatile Districts (Enrollment Variability)')
plt.show()
```



📌 District-wise Enrollment Ranking Analysis

In [121…

```
district_rank = df.groupby('district_clean')['total_enrollment'].sum() \
                  .rank(ascending=False)

district_rank.sort_values().head(10)
```

Out[121…
```
district_clean
Thane                          1.0
Pune                           2.0
Chhatrapati Sambhajinagar      3.0
Nashik                         4.0
Mumbai Suburban                5.0
Mumbai City                    6.0
Jalgaon                        7.0
Dhule                          8.0
Ahmednagar                     9.0
Solapur                       10.0
Name: total_enrollment, dtype: float64
```

We performed univariate, bivariate, and multivariate analysis to study enrollment distribution across states,
districts, and age groups. Advanced analysis included district-level volatility, age composition,
time-based trends, and flag-based segmentation to uncover hidden patterns.