```python
import pandas as pd

data_a2=pd.read_excel('/kaggle/input/mospi-hces/Table
A2.xlsx',header=[0,1,2,3,4])

data_a2.head(20)
```

```
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/
format.py:1458: RuntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
```

```
   Table A2:  Estimated number of households and persons by gender and
average MPCE for each fractile class of MPCE  \

Sector

Unnamed: 0_level_2

Unnamed: 0_level_3

(1)
0                                                                Rural

1                                                                  NaN

2                                                                  NaN

3                                                                  NaN

4                                                                  NaN

5                                                                  NaN

6                                                                  NaN

7                                                                  NaN

8                                                                  NaN

9                                                                  NaN

10                                                                 NaN
```

```
11                                                NaN

12                                                NaN

13                                                NaN

14                                              Urban

15                                                NaN

16                                                NaN

17                                                NaN

18                                                NaN

19                                                NaN



\
   State/UT/All-India Fractile class of MPCE Estimated no. (00)

   Unnamed: 1_level_2     Unnamed: 2_level_2         Households
Adults#
   Unnamed: 1_level_3     Unnamed: 2_level_3 Unnamed: 3_level_3
Male
                  (2)                    (3)                (4)
(5)
0      Andhra Pradesh                   0-5%             3723.0
5082.0
1                 NaN                  5-10%             3701.0
5153.0
2                 NaN                 10-20%             7785.0
11566.0
3                 NaN                 20-30%             8260.0
12320.0
4                 NaN                 30-40%             8729.0
11933.0
5                 NaN                 40-50%             8837.0
12297.0
6                 NaN                 50-60%             9532.0
12992.0
7                 NaN                 60-70%            10055.0
12769.0
8                 NaN                 70-80%            10516.0
13440.0
9                 NaN                 80-90%            11880.0
13577.0
10                NaN                 90-95%             5778.0
```

```
6872.0
11                NaN              95-100%             7019.0
6894.0
12                NaN          All classes            95813.0
124893.0
13                NaN   Sample no. of hhs.             6245.0
8153.0
14                NaN                 0-5%             1657.0
2319.0
15                NaN                5-10%             1976.0
2405.0
16                NaN               10-20%             3809.0
5157.0
17                NaN               20-30%             3819.0
5241.0
18                NaN               30-40%             3815.0
5476.0
19                NaN               40-50%             4194.0
5654.0


\
                                                    Average MPCE (Rs.)

                       Children*                   Unnamed: 10_level_2

      Female   Persons      Male   Female  Persons Unnamed: 10_level_3

         (6)       (7)       (8)      (9)     (10)                (11)

0     6093.0   11175.0    2369.0   2221.0   4590.0             1952.46

1     6331.0   11484.0    2138.0   2162.0   4300.0             2441.62

2    12527.0   24093.0    4107.0   3331.0   7438.0             2861.11

3    12376.0   24696.0    3612.0   3232.0   6844.0             3320.33

4    12732.0   24666.0    3633.0   3272.0   6905.0             3721.22

5    12700.0   24996.0    3361.0   3143.0   6503.0             4126.01

6    12752.0   25744.0    3319.0   2491.0   5810.0             4570.37

7    13119.0   25888.0    2869.0   2790.0   5659.0             5111.09

8    12895.0   26335.0    2746.0   2428.0   5174.0             5754.48

9    13594.0   27171.0    2325.0   2055.0   4380.0             6732.20

10    7079.0   13951.0     879.0    961.0   1840.0             8062.02
```

```
11    7445.0   14339.0     815.0    602.0   1417.0            12560.21
12  129643.0  254536.0   32171.0  28688.0  60860.0             4870.30
13    8578.0   16731.0    2196.0   1980.0   4176.0                 NaN
14    2605.0    4925.0    1243.0   1030.0   2274.0             2187.48
15    2840.0    5245.0    1022.0    879.0   1902.0             3017.42
16    5389.0   10547.0    2200.0   1597.0   3797.0             3675.68
17    5830.0   11071.0    1717.0   1534.0   3251.0             4297.05
18    5740.0   11216.0    1492.0   1645.0   3137.0             4858.12
19    5943.0   11598.0    1629.0   1143.0   2772.0             5450.81
```

```
     Sample households      Sample persons
   Unnamed: 11_level_2 Unnamed: 12_level_2
   Unnamed: 11_level_3 Unnamed: 12_level_3
                  (12)                (13)
0                319.0              1387.0
1                281.0              1208.0
2                542.0              2192.0
3                551.0              2098.0
4                573.0              2071.0
5                586.0              2061.0
6                623.0              2063.0
7                648.0              2020.0
8                643.0              1968.0
9                705.0              1925.0
10               364.0               973.0
11               410.0               941.0
12              6245.0             20907.0
13                 NaN                 NaN
14               135.0               485.0
15               176.0               646.0
16               338.0              1316.0
17               350.0              1342.0
18               341.0              1307.0
19               401.0              1399.0
```

```
indices=data_a2.columns

indices[10]
```

```
('Table A2:  Estimated number of households and persons by gender and
average MPCE for each fractile class of MPCE',
 'Average MPCE (Rs.)',
 'Unnamed: 10_level_2',
 'Unnamed: 10_level_3',
 '(11)')
```

```
i_states=indices[1]
i_average_MPCE=indices[10]
i_fractile_classes=indices[2]
```

```
data_a2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1038 entries, 0 to 1037
Data columns (total 13 columns):
 #   Column
Non-Null Count  Dtype
---  ------
--------------  -----
 0   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Sector, Unnamed:
0_level_2, Unnamed: 0_level_3, (1))                  76 non-null
object
 1   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, State/UT/All-India,
Unnamed: 1_level_2, Unnamed: 1_level_3, (2))      37 non-null
object
 2   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Fractile class of
MPCE, Unnamed: 2_level_2, Unnamed: 2_level_3, (3))  1036 non-null
object
 3   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Households, Unnamed: 3_level_3, (4))              1036 non-null
float64
 4   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Adults#, Male, (5))                               1036 non-null
float64
 5   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Adults#, Female, (6))                             1036 non-null
float64
 6   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Adults#, Persons, (7))                            1036 non-null
float64
 7   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Children*, Male, (8))                                    1036 non-null
float64
 8   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Children*, Female, (9))                                  1036 non-null
float64
 9   (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Estimated no. (00),
Children*, Persons, (10))                                1036 non-null
float64
 10  (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Average MPCE (Rs.),
Unnamed: 10_level_2, Unnamed: 10_level_3, (11))   962 non-null
float64
 11  (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Sample households,
Unnamed: 11_level_2, Unnamed: 11_level_3, (12))    962 non-null
float64
 12  (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, Sample persons,
Unnamed: 12_level_2, Unnamed: 12_level_3, (13))       962 non-null
float64
dtypes: float64(10), object(3)
memory usage: 105.5+ KB
```

```
data_a2[i_states]
```

```
0       Andhra Pradesh
1                  NaN
2                  NaN
3                  NaN
4                  NaN
             ...
1033               NaN
1034               NaN
1035               NaN
1036               NaN
1037               NaN
Name: (Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE, State/UT/All-India,
Unnamed: 1_level_2, Unnamed: 1_level_3, (2)), Length: 1038, dtype:
object
```

```
states_and_All_India=data_a2[i_states].value_counts(sort=False)
```

```
states=states_and_All_India.drop('All-India')
```

```
states
```

```
(Table A2:  Estimated number of households and persons by gender and
average MPCE for each fractile class of MPCE, State/UT/All-India,
Unnamed: 1_level_2, Unnamed: 1_level_3, (2))
Andhra Pradesh                             1
Arunachal Pradesh                          1
Assam                                      1
Bihar                                      1
Chhattisgarh                               1
Delhi                                      1
Goa                                        1
Gujarat                                    1
Haryana                                    1
Himachal Pradesh                           1
Jharkhand                                  1
Karnataka                                  1
Kerala                                     1
Madhya Pradesh                             1
Maharashtra                                1
Manipur                                    1
Meghalaya                                  1
Mizoram                                    1
Nagaland                                   1
Odisha                                     1
Punjab                                     1
Rajasthan                                  1
Sikkim                                     1
Tamil Nadu                                 1
Telangana                                  1
Tripura                                    1
Uttar Pradesh                              1
Uttarakhand                                1
West Bengal                                1
Andaman & Nicobar Islands                  1
Chandigarh                                 1
Dadra and Nagar Haveli & Daman and Diu     1
Jammu & Kashmir                            1
Ladakh                                     1
Lakshadweep                                1
Puducherry                                 1
Name: count, dtype: int64
```

```
average_MPCE=data_a2[

    [i_fractile_classes,
    i_average_MPCE]

]

average_MPCE_all_classes=average_MPCE.loc[average_MPCE[i_fractile_classes]=='All classes']

#average_MPCE_all_classes=average_MPCE_all_classes.iloc[:-2]
```

```
average_MPCE_all_classes

     Table A2:  Estimated number of households and persons by gender
and average MPCE for each fractile class of MPCE  \

Fractile class of MPCE

Unnamed: 2_level_2

Unnamed: 2_level_3

(3)
12                                          All classes

26                                          All classes

40                                          All classes

54                                          All classes

68                                          All classes

...                                                 ...

978                                         All classes

992                                         All classes

1006                                        All classes

1020                                        All classes

1034                                        All classes



      Average MPCE (Rs.)
     Unnamed: 10_level_2
     Unnamed: 10_level_3
                    (11)
12               4870.30
26               6781.76
40               5276.34
54               8635.53
68               3432.41
...                  ...
978              5474.78
992              6590.36
1006             7706.44
1020             3773.06
1034             6458.70
```

```
[74 rows x 2 columns]
```

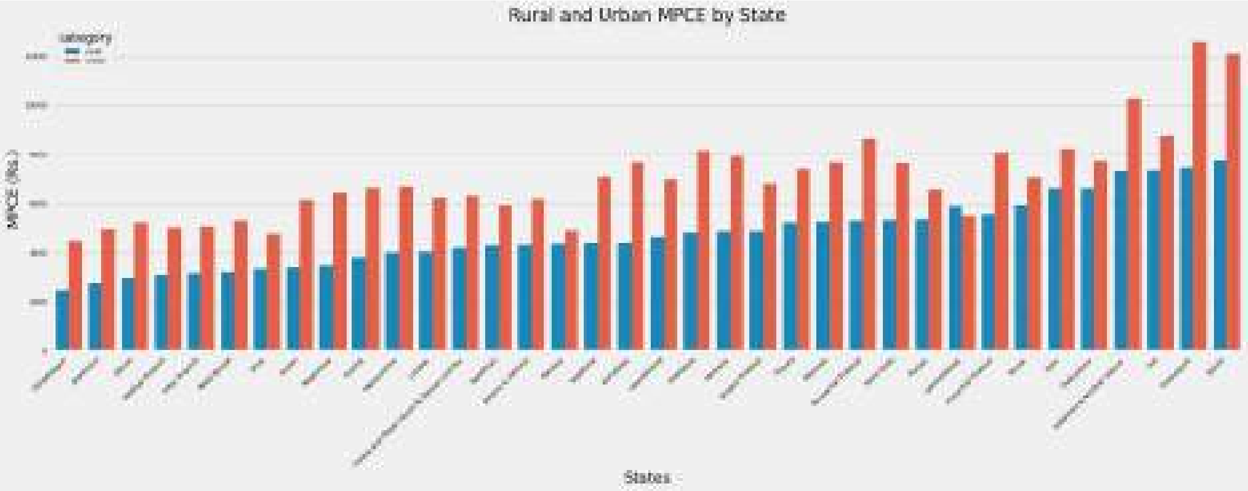
Figure 1 State wise MPCE For Rural and Urban

```
states_list=states.index.to_list()

states_list

['Andhra Pradesh',
 'Arunachal Pradesh',
 'Assam',
 'Bihar',
 'Chhattisgarh',
 'Delhi',
 'Goa',
 'Gujarat',
 'Haryana',
 'Himachal Pradesh',
 'Jharkhand',
 'Karnataka',
 'Kerala',
 'Madhya Pradesh',
 'Maharashtra',
 'Manipur',
 'Meghalaya',
 'Mizoram',
 'Nagaland',
 'Odisha',
 'Punjab',
 'Rajasthan',
 'Sikkim',
 'Tamil Nadu',
 'Telangana',
 'Tripura',
 'Uttar Pradesh',
 'Uttarakhand',
 'West Bengal',
 'Andaman & Nicobar Islands ',
 'Chandigarh',
 'Dadra and Nagar Haveli & Daman and Diu ',
 'Jammu & Kashmir',
 'Ladakh ',
 'Lakshadweep',
 'Puducherry']

states_MPCE=pd.DataFrame(states_list)

states_MPCE.rename(columns={0:'state'},inplace=True)
```

```
rural_MPCE=average_MPCE_all_classes[i_average_MPCE].iloc[::2].to_list(
)

states_MPCE['rural']=pd.DataFrame(rural_MPCE)

urban_MPCE=average_MPCE_all_classes[i_average_MPCE].iloc[1::2].to_list
()

states_MPCE['urban']=pd.DataFrame(urban_MPCE)

states_MPCE
```

```
                                      state    rural     urban
0                            Andhra Pradesh  4870.30   6781.76
1                         Arunachal Pradesh  5276.34   8635.53
2                                     Assam  3432.41   6135.51
3                                     Bihar  3384.11   4767.69
4                              Chhattisgarh  2466.16   4483.10
5                                     Delhi  6575.67   8217.49
6                                       Goa  7366.57   8733.86
7                                   Gujarat  3798.30   6620.72
8                                   Haryana  4858.70   7910.51
9                          Himachal Pradesh  5560.85   8075.28
10                                Jharkhand  2763.26   4930.99
11                                Karnataka  4397.47   7665.88
12                                   Kerala  5923.62   7078.22
13                           Madhya Pradesh  3112.63   4987.29
14                              Maharashtra  4010.45   6657.03
15                                  Manipur  4360.42   4880.47
16                                Meghalaya  3513.84   6433.36
17                                  Mizoram  5223.69   7655.03
18                                 Nagaland  4393.10   7097.75
19                                   Odisha  2949.63   5187.39
20                                   Punjab  5314.75   6543.52
21                                Rajasthan  4263.14   5913.06
22                                   Sikkim  7730.89  12105.11
23                               Tamil Nadu  5310.34   7630.05
24                                Telangana  4802.23   8158.44
25                                  Tripura  5206.25   7404.69
26                            Uttar Pradesh  3190.98   5040.41
27                              Uttarakhand  4640.93   7004.37
28                              West Bengal  3239.16   5267.20
29               Andaman & Nicobar Islands   7331.60  10268.15
30                               Chandigarh  7466.86  12575.28
31  Dadra and Nagar Haveli & Daman and Diu   4184.11   6298.49
32                          Jammu & Kashmir  4295.69   6178.51
33                                  Ladakh   4035.30   6214.52
34                              Lakshadweep  5895.46   5474.78
35                               Puducherry  6590.36   7706.44
```

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.style.use(['fivethirtyeight'])

# First, reshape the data from wide to long format
states_MPCE_long = states_MPCE.melt(
    id_vars=['state'],
    value_vars=['rural', 'urban'],
    var_name='category',
    value_name='MPCE'
)

# Create the plot
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
plot=sns.barplot(
    data=states_MPCE_long.sort_values('MPCE'),
    x='state',
    y='MPCE',
    hue='category',
    #palette=['lightblue', 'darkblue']  # You can change colors as
needed
)

# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Rural and Urban MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show the plot
plt.show()

plot.figure.savefig('plot.png', dpi=300)
```



```
print(plt.style.available)
```

```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-
gallery-nogrid', 'bmh', 'classic', 'dark_background', 'fast',
'fivethirtyeight', 'ggplot', 'grayscale', 'seaborn-v0_8', 'seaborn-
v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-v0_8-dark',
'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-
deep', 'seaborn-v0_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-
paper', 'seaborn-v0_8-pastel', 'seaborn-v0_8-poster', 'seaborn-v0_8-
talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'seaborn-v0_8-
whitegrid', 'tableau-colorblind10']
```

```
## Insights

# Sikkhim and Chandigarh have the highest MPCE
```

Figure 2 MPCE of Food, Intoxicants and Non Food Expenses in all the states

```
data_a6=pd.read_excel('/kaggle/input/mospi-hces/Table
A6.xlsx',header=[0,1,2,3])

data_a6.head()
```

```
  Table A6:  Value of consumption (Rs.) of broad groups of food and
non-food items per person for a period of 30 days for each fractile
class of MPCE   \

Sector

Unnamed: 0_level_2

(1)
0                                                  Rural
```

```
1                                                 NaN

2                                                 NaN

3                                                 NaN

4                                                 NaN



\
  State/UT/All-India         Item description Fractile classes of MPCE

  Unnamed: 1_level_2       Unnamed: 2_level_2                      0-
5%
                 (2)                      (3)
(4)
0     Andhra Pradesh                   Cereal
101.43
1                NaN                     Gram
2.32
2                NaN  Pulses & pulse products
59.11
3                NaN                    Sugar
13.25
4                NaN                     Salt
3.65


\


    5-10%  10-20%  20-30%  30-40%  40-50%  50-60%  60-70%  70-80%  80-
90%
      (5)     (6)     (7)     (8)     (9)    (10)    (11)    (12)
(13)
0  138.65  146.44  156.94  191.74  184.58  208.45  224.00  230.33
242.50
1    2.53    3.56    4.24    4.65    5.14    4.94    5.14    6.30
7.25
2   65.59   68.65   71.59   77.38   77.61   81.89   89.32   95.49
102.30
3   15.04   17.18   19.35   20.26   20.84   22.75   23.46   24.82
28.92
4    4.13    4.48    4.44    4.72    4.62    4.75    4.99    5.29
5.37



                              No. of households reporting consumption
```

```
   90-95% 95-100% All Classes                          Estimated (00)
Sample
     (14)    (15)        (16)                                    (17)
(18)
0  283.75  328.82      201.13                                 93871.0
6195.0
1    6.63    9.53        5.17                                 36046.0
2368.0
2  112.13  132.13       84.87                                 93170.0
6152.0
3   31.32   39.03       22.69                                 93876.0
6198.0
4    5.86    6.83        4.89                                 92476.0
6110.0
```

```
indices=data_a6.columns
```

```
i_states=indices[1]
i_MPCE=indices[15]
i_categories=indices[2]
```

```
indices[2]
```

```
('Table A6:  Value of consumption (Rs.) of broad groups of food and
non-food items per person for a period of 30 days for each fractile
class of MPCE',
 'Item description',
 'Unnamed: 2_level_2',
 '(3)')
```

```
MPCE=data_a6[
[
    i_states,
    i_categories,
    i_MPCE
]
]
```

```
MPCE.head()
```

```
  Table A6:  Value of consumption (Rs.) of broad groups of food and
non-food items per person for a period of 30 days for each fractile
class of MPCE  \

State/UT/All-India

Unnamed: 1_level_2

(2)
0                                      Andhra Pradesh
```

```
1                                                NaN

2                                                NaN

3                                                NaN

4                                                NaN
```

```
          Item description Fractile classes of MPCE
        Unnamed: 2_level_2               All Classes
                       (3)                      (16)
0                   Cereal                    201.13
1                     Gram                      5.17
2  Pulses & pulse products                     84.87
3                    Sugar                     22.69
4                     Salt                      4.89
```

```
MPCE[i_states]=MPCE[i_states].fillna(method='ffill')
```

```
<ipython-input-139-c6002c607d33>:1: FutureWarning: Series.fillna with
'method' is deprecated and will raise in a future version. Use
obj.ffill() or obj.bfill() instead.
  MPCE[i_states]=MPCE[i_states].fillna(method='ffill')
<ipython-input-139-c6002c607d33>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  MPCE[i_states]=MPCE[i_states].fillna(method='ffill')
```

```
MPCE.head()
```

```
  Table A6:  Value of consumption (Rs.) of broad groups of food and
non-food items per person for a period of 30 days for each fractile
class of MPCE  \

State/UT/All-India

Unnamed: 1_level_2

(2)
0                                     Andhra Pradesh

1                                     Andhra Pradesh

2                                     Andhra Pradesh
```

```
3                                     Andhra Pradesh

4                                     Andhra Pradesh
```

```
          Item description Fractile classes of MPCE
        Unnamed: 2_level_2                All Classes
                       (3)                       (16)
0                   Cereal                     201.13
1                     Gram                       5.17
2   Pulses & pulse products                     84.87
3                    Sugar                      22.69
4                     Salt                       4.89
```

```
MPCE_categories=MPCE.loc[(MPCE[i_categories]=='Food: Total') |
(MPCE[i_categories]=='Pan, Tobacco & Intoxicants') |
(MPCE[i_categories]=='Non-food Total')]

MPCE_categories.columns=['State','Category','MPCE']

MPCE_categories.head(25)
```

```
                 State                    Category     MPCE
15      Andhra Pradesh                 Food: Total  2149.18
19      Andhra Pradesh  Pan, Tobacco & Intoxicants   188.54
34      Andhra Pradesh              Non-food Total  2721.11
59      Andhra Pradesh                 Food: Total  2616.66
63      Andhra Pradesh  Pan, Tobacco & Intoxicants   148.86
78      Andhra Pradesh              Non-food Total  4165.09
104  Arunachal Pradesh                 Food: Total  2680.25
108  Arunachal Pradesh  Pan, Tobacco & Intoxicants   427.84
123  Arunachal Pradesh              Non-food Total  2596.08
149  Arunachal Pradesh                 Food: Total  3803.11
153  Arunachal Pradesh  Pan, Tobacco & Intoxicants   563.11
168  Arunachal Pradesh              Non-food Total  4832.42
194              Assam                 Food: Total  1862.36
198              Assam  Pan, Tobacco & Intoxicants   202.58
213              Assam              Non-food Total  1570.04
238              Assam                 Food: Total  2865.92
242              Assam  Pan, Tobacco & Intoxicants   325.48
257              Assam              Non-food Total  3269.58
283              Bihar                 Food: Total  1812.18
287              Bihar  Pan, Tobacco & Intoxicants    95.87
302              Bihar              Non-food Total  1571.93
328              Bihar                 Food: Total  2258.52
332              Bihar  Pan, Tobacco & Intoxicants    77.30
347              Bihar              Non-food Total  2509.18
373        Chhattisgarh                 Food: Total  1125.91
```

```
rural_food_MPCE=MPCE_categories.iloc[::6]
rural_food_MPCE.head()
```

```
                 State     Category     MPCE
15      Andhra Pradesh  Food: Total  2149.18
104  Arunachal Pradesh  Food: Total  2680.25
194              Assam  Food: Total  1862.36
283              Bihar  Food: Total  1812.18
373       Chhattisgarh  Food: Total  1125.91
```

```
rural_intoxicants_MPCE=MPCE_categories.iloc[1::6]
rural_intoxicants_MPCE.head()
```

```
                 State                    Category    MPCE
19      Andhra Pradesh  Pan, Tobacco & Intoxicants  188.54
108  Arunachal Pradesh  Pan, Tobacco & Intoxicants  427.84
198              Assam  Pan, Tobacco & Intoxicants  202.58
287              Bihar  Pan, Tobacco & Intoxicants   95.87
377       Chhattisgarh  Pan, Tobacco & Intoxicants  153.37
```

```
rural_nonfood_MPCE=MPCE_categories.iloc[2::6]
rural_nonfood_MPCE.head()
```

```
                 State        Category     MPCE
34      Andhra Pradesh  Non-food Total  2721.11
123  Arunachal Pradesh  Non-food Total  2596.08
213              Assam  Non-food Total  1570.04
302              Bihar  Non-food Total  1571.93
392       Chhattisgarh  Non-food Total  1340.24
```

```
urban_food_MPCE=MPCE_categories.iloc[3::6]
urban_food_MPCE.head()
```

```
                 State     Category     MPCE
59      Andhra Pradesh  Food: Total  2616.66
149  Arunachal Pradesh  Food: Total  3803.11
238              Assam  Food: Total  2865.92
328              Bihar  Food: Total  2258.52
418       Chhattisgarh  Food: Total  1761.19
```

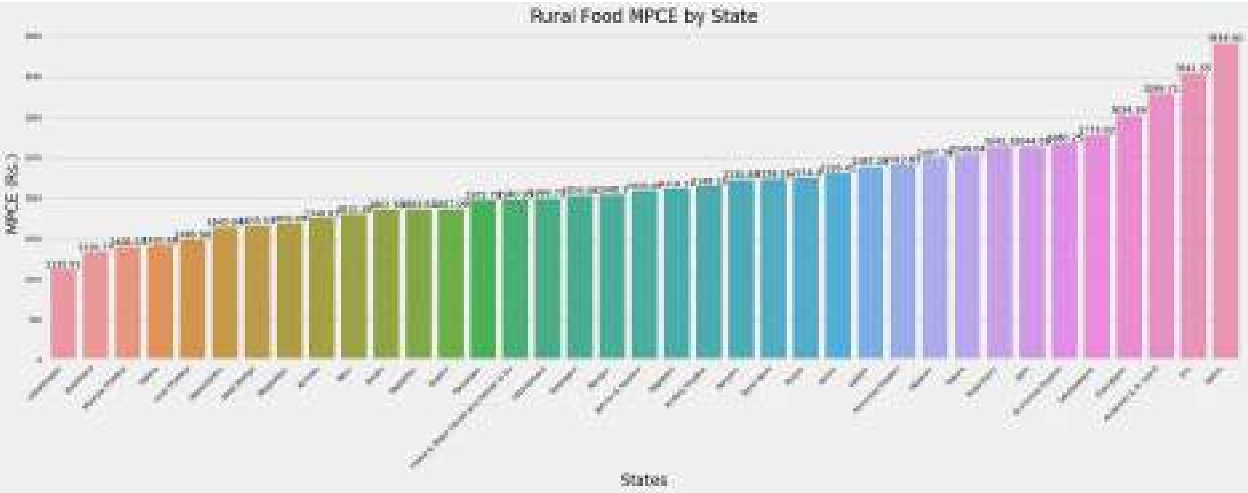
```
urban_intoxicants_MPCE=MPCE_categories.iloc[4::6]
urban_intoxicants_MPCE.head()
```

```
                 State                    Category    MPCE
63      Andhra Pradesh  Pan, Tobacco & Intoxicants  148.86
153  Arunachal Pradesh  Pan, Tobacco & Intoxicants  563.11
242              Assam  Pan, Tobacco & Intoxicants  325.48
332              Bihar  Pan, Tobacco & Intoxicants   77.30
422       Chhattisgarh  Pan, Tobacco & Intoxicants  168.29
```

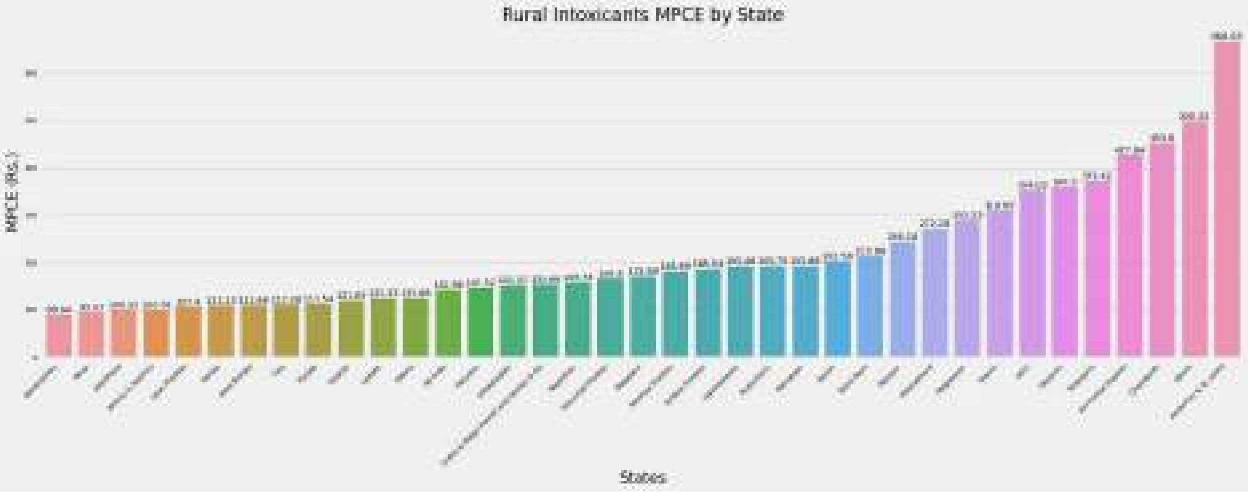
```
urban_nonfood_MPCE=MPCE_categories.iloc[5::6]
urban_nonfood_MPCE.head()
```

```
                  State        Category     MPCE
78       Andhra Pradesh  Non-food Total  4165.09
168   Arunachal Pradesh  Non-food Total  4832.42
257               Assam  Non-food Total  3269.58
347               Bihar  Non-food Total  2509.18
437        Chhattisgarh  Non-food Total  2721.92
```

```
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
ax=sns.barplot(
    data=rural_food_MPCE.sort_values('MPCE'),
    x='State',
    y='MPCE'
)
ax.bar_label(ax.containers[0], fontsize=10)
# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Rural Food MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show the plot
plt.show()

ax.figure.savefig('rural_food.png', dpi=300)
```

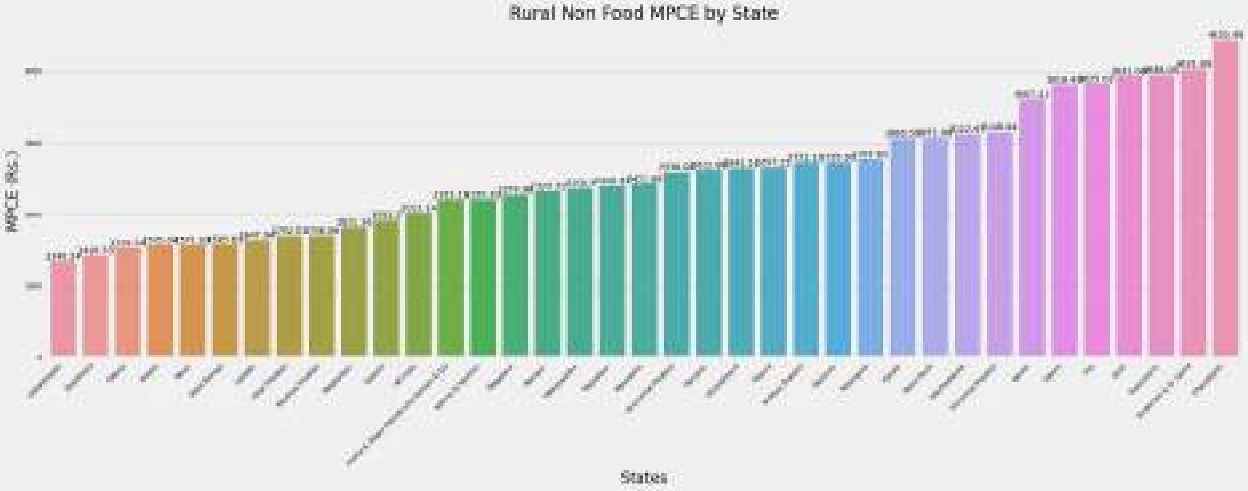


```
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
ax=sns.barplot(
    data=rural_intoxicants_MPCE.sort_values('MPCE'),
    x='State',
    y='MPCE'
)
ax.bar_label(ax.containers[0], fontsize=10)
# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Rural Intoxicants MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show the plot
plt.show()

ax.figure.savefig('rural_intoxicants.png', dpi=300)
```

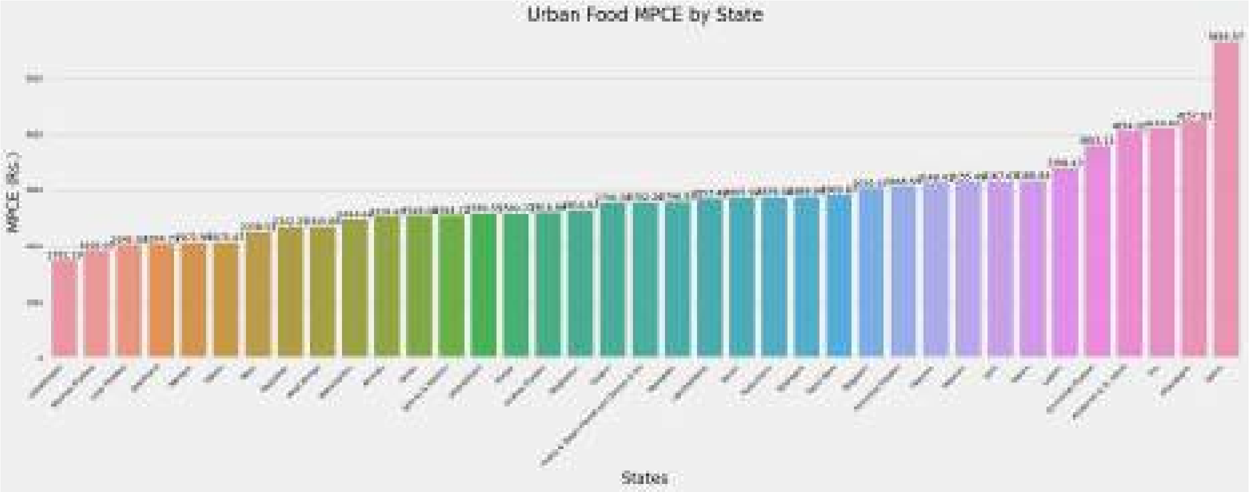


```
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
ax=sns.barplot(
    data=rural_nonfood_MPCE.sort_values('MPCE'),
    x='State',
    y='MPCE'
)
ax.bar_label(ax.containers[0], fontsize=10)
# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Rural Non Food MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()
```

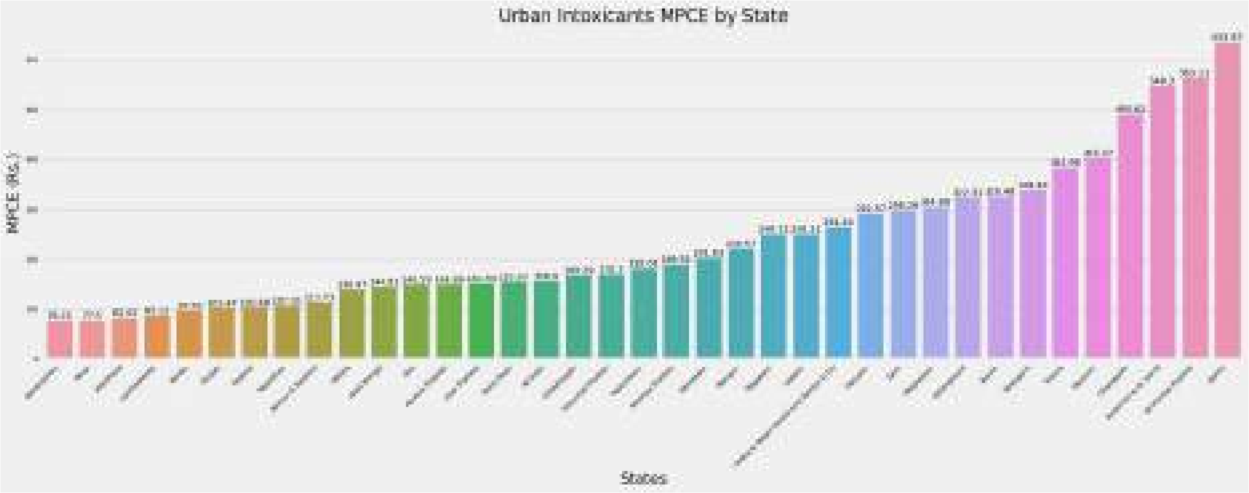
```
# Show the plot
plt.show()

ax.figure.savefig('rural_nonfood.png', dpi=300)
```



```
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
ax=sns.barplot(
    data=urban_food_MPCE.sort_values('MPCE'),
    x='State',
    y='MPCE'
)
ax.bar_label(ax.containers[0], fontsize=10)

# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Urban Food MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show the plot
plt.show()

ax.figure.savefig('urban_food.png', dpi=300)
```



```
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
ax=sns.barplot(
    data=urban_intoxicants_MPCE.sort_values('MPCE'),
    x='State',
    y='MPCE'
)
ax.bar_label(ax.containers[0], fontsize=10)
# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Urban Intoxicants MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show the plot
plt.show()
ax.figure.savefig('urban_intoxicants.png', dpi=300)
```



```
plt.figure(figsize=(20, 8))  # Adjust figure size as needed
ax=sns.barplot(
    data=rural_nonfood_MPCE.sort_values('MPCE'),
    x='State',
    y='MPCE'
)
ax.bar_label(ax.containers[0], fontsize=10)
# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Urban Non Food MPCE by State')
plt.xlabel('States')
plt.ylabel('MPCE (Rs.)')

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show the plot
plt.show()

ax.figure.savefig('urban_nonfood.png', dpi=300)
```

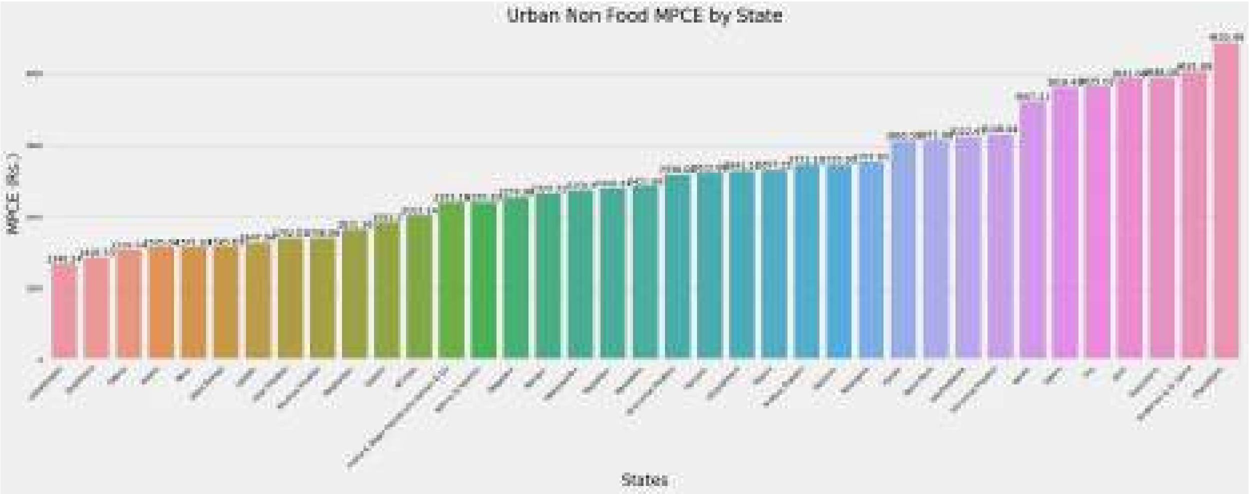


Figure 3: State-wise broad Expenditure Split Up

```
categories_MPCE=pd.DataFrame(rural_food_MPCE['State'])

categories_MPCE['urban_food_MPCE']=urban_food_MPCE['MPCE'].values

categories_MPCE['rural_food_MPCE']=rural_food_MPCE['MPCE'].values

categories_MPCE['rural_intoxicants_MPCE']=rural_intoxicants_MPCE['MPCE
'].values

categories_MPCE['urban_intoxicants_MPCE']=urban_intoxicants_MPCE['MPCE
'].values
```

```
categories_MPCE['rural_nonfood_MPCE']=rural_nonfood_MPCE['MPCE'].value
s

categories_MPCE['urban_nonfood_MPCE']=urban_nonfood_MPCE['MPCE'].value
s

categories_MPCE['food_MPCE']=categories_MPCE['rural_food_MPCE'].values
+categories_MPCE['urban_food_MPCE'].values

categories_MPCE['intoxicants_MPCE']=categories_MPCE['urban_intoxicants
_MPCE'].values+categories_MPCE['rural_intoxicants_MPCE'].values

categories_MPCE['nonfood_MPCE']=categories_MPCE['rural_nonfood_MPCE'].
values+categories_MPCE['urban_nonfood_MPCE'].values

categories_MPCE['otherexpenses_MPCE']=categories_MPCE['nonfood_MPCE'].
values-categories_MPCE['intoxicants_MPCE'].values

categories_MPCE['totalexpenses_MPCE']=categories_MPCE['food_MPCE']
+categories_MPCE['nonfood_MPCE']

categories_MPCE.head()
```

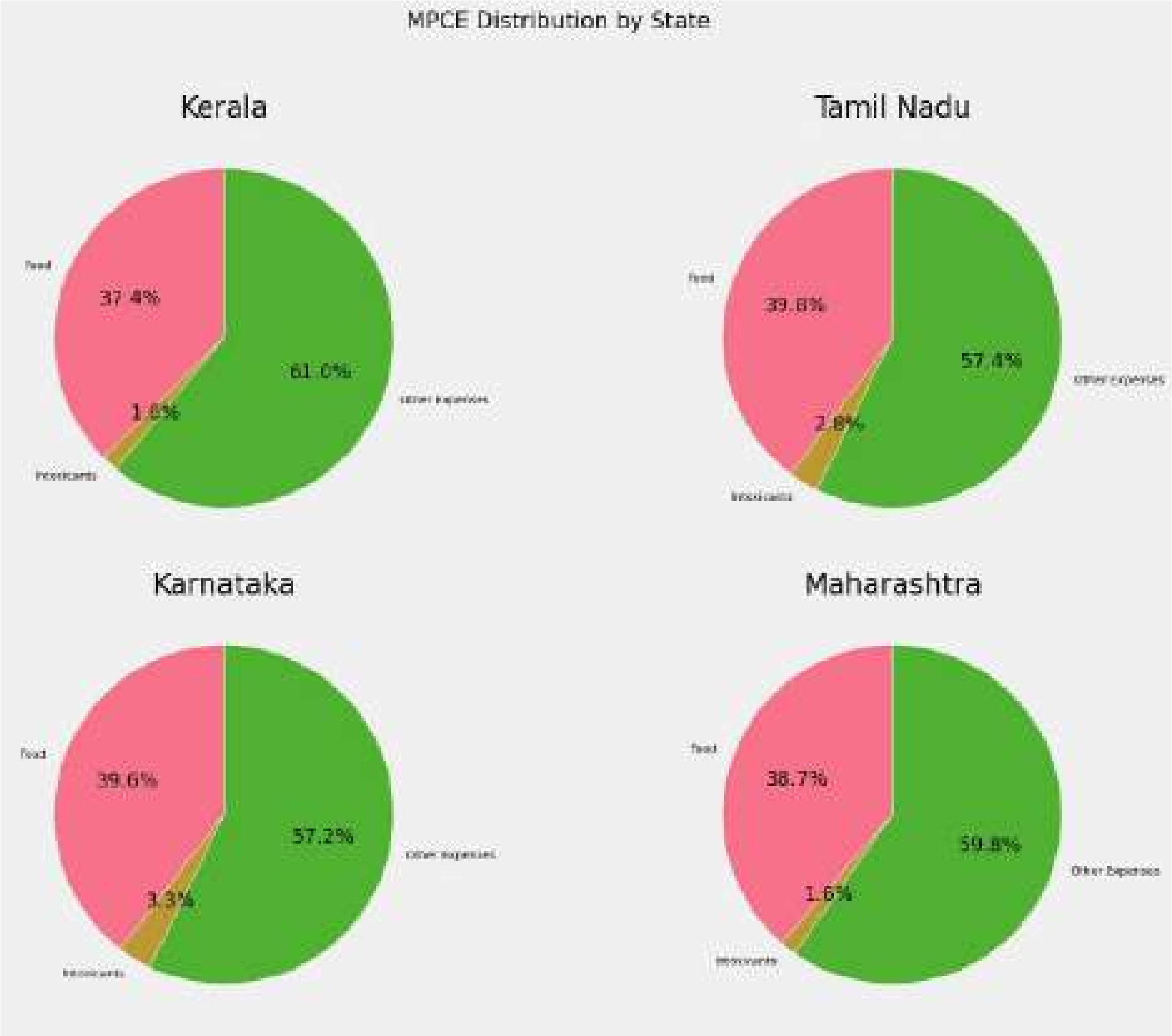
```
                 State  urban_food_MPCE  rural_food_MPCE  \
15      Andhra Pradesh          2616.66          2149.18
104  Arunachal Pradesh          3803.11          2680.25
194              Assam          2865.92          1862.36
283              Bihar          2258.52          1812.18
373       Chhattisgarh          1761.19          1125.91

     rural_intoxicants_MPCE  urban_intoxicants_MPCE
rural_nonfood_MPCE  \
15                   188.54                  148.86
2721.11
104                  427.84                  563.11
2596.08
194                  202.58                  325.48
1570.04
283                   95.87                   77.30
1571.93
373                  153.37                  168.29
1340.24

     urban_nonfood_MPCE  food_MPCE  intoxicants_MPCE  nonfood_MPCE  \
15              4165.09    4765.84            337.40       6886.20
104             4832.42    6483.36            990.95       7428.50
194             3269.58    4728.28            528.06       4839.62
283             2509.18    4070.70            173.17       4081.11
373             2721.92    2887.10            321.66       4062.16

     otherexpenses_MPCE  totalexpenses_MPCE
```

```
15               6548.80            11652.04
104              6437.55            13911.86
194              4311.56             9567.90
283              3907.94             8151.81
373              3740.50             6949.26
```

```
categories_MPCE.columns
```

```
Index(['State', 'urban_food_MPCE', 'rural_food_MPCE',
'rural_intoxicants_MPCE',
       'urban_intoxicants_MPCE', 'rural_nonfood_MPCE',
'urban_nonfood_MPCE',
       'food_MPCE', 'intoxicants_MPCE', 'nonfood_MPCE',
'otherexpenses_MPCE',
       'totalexpenses_MPCE'],
      dtype='object')
```

```
def plot_multiple_states(df, state_list):
    # Calculate number of rows and columns needed
    n_states = len(state_list)
    n_cols = 2  # You can adjust this
    n_rows = (n_states + 1) // 2

    # Create subplots
    fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, 5*n_rows))
    fig.suptitle('MPCE Distribution by State', size=16, y=1.02)

    # Flatten axes array for easier iteration
    axes = axes.flatten()

    for idx, state in enumerate(state_list):
        # Get data for the state
        state_data = df[df['State'] == state]
        values = [
            state_data['food_MPCE'].values[0],
            state_data['intoxicants_MPCE'].values[0],
            state_data['otherexpenses_MPCE'].values[0]
        ]

        # Labels
        labels = ['Food', 'Intoxicants', 'Other Expenses']

        # Create pie chart
        axes[idx].pie(values,
                      labels=labels,
                      autopct='%1.1f%%',
                      startangle=90,
                      colors=sns.color_palette('husl'),
                      wedgeprops={'edgecolor': 'white'})
        axes[idx].set_title(state)

    # Remove empty subplots if any
    for idx in range(len(state_list), len(axes)):
        fig.delaxes(axes[idx])

    plt.tight_layout()
    plt.show()

# Example usage for multiple states
states_to_plot = ['Kerala', 'Tamil Nadu', 'Karnataka', 'Maharashtra']
plot_multiple_states(categories_MPCE, states_to_plot)
```



```
def plot_multiple_states(df):
    # Get all unique states
    state_list = df['State'].unique()
    # Calculate number of rows and columns needed
    n_states = len(state_list)
    n_cols = 4  # 4 charts per row
    n_rows = (n_states + n_cols - 1) // n_cols

    # Create subplots
    fig, axes = plt.subplots(n_rows, n_cols, figsize=(20, 5*n_rows))
    #fig.suptitle('MPCE Distribution Across States', size=20, y=0.95,
fontweight='bold')

    # Flatten axes array for easier iteration
    axes = axes.flatten()

    # Custom colors and style
    colors = sns.color_palette('husl', n_colors=3)
    plt.style.use('seaborn-v0_8-paper')

    for idx, state in enumerate(state_list):
        # Get data for the state
        state_data = df[df['State'] == state]
        values = [
            state_data['food_MPCE'].values[0],
            state_data['intoxicants_MPCE'].values[0],
            state_data['otherexpenses_MPCE'].values[0]
        ]

        # Labels
        labels = ['Food', 'Intoxicants', 'Other Expenses']

        # Create pie chart
        wedges, texts, autotexts = axes[idx].pie(
            values,
            labels=labels,
            autopct='%1.1f%%',
            startangle=90,
            colors=colors,
            wedgeprops={'edgecolor': 'white', 'linewidth': 2},
            textprops={'fontsize': 10},
            pctdistance=0.85,
            explode=(0.05, 0.05, 0.05)  # Slight explosion for all
segments
        )

        # Enhance text properties
        plt.setp(autotexts, size=10, weight="bold")
        plt.setp(texts, size=10)

        axes[idx].set_title(state, pad=10, size=10, fontweight='bold')
    # Remove empty subplots if any
    for idx in range(len(state_list), len(axes)):
        fig.delaxes(axes[idx])

    # Adjust layout
    plt.tight_layout()

    # Add a common legend at the bottom
    fig.legend(
        wedges[:3],
        labels,
        title="Expenditure Categories",
        loc="center",
        bbox_to_anchor=(0.5, 0.02),
        ncol=3,
        fontsize=12
    )
    fig.savefig('multiplot.png',dpi=100)

    # Add some padding at the bottom for the legend
    plt.subplots_adjust(bottom=0.1)

    plt.show()



# Use the function
plot_multiple_states(categories_MPCE)
```

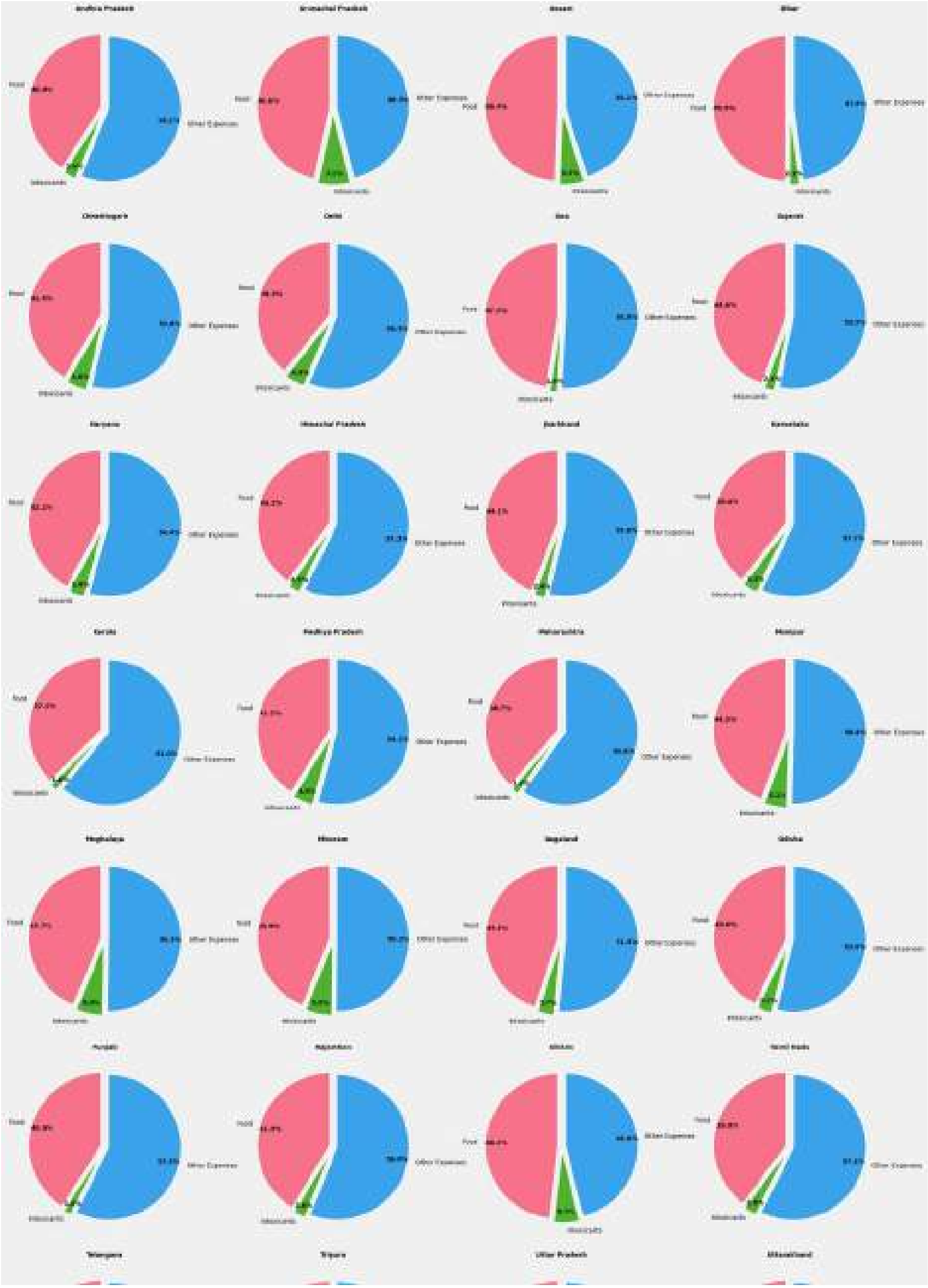

Andhra Pradesh
Food
Other Expenses
Intoxicants
Arunachal Pradesh
Food
Other Expenses
Intoxicants
Assam
Food
Other Expenses
Intoxicants
Bihar
Food
Other Expenses
Intoxicants
Chhattisgarh
Food
Other Expenses
Intoxicants
Delhi
Food
Other Expenses
Intoxicants
Goa
Food
Other Expenses
Intoxicants
Gujarat
Food
Other Expenses
Intoxicants
Haryana
Food
Other Expenses
Intoxicants
Himachal Pradesh
Food
Other Expenses
Intoxicants
Jharkhand
Food
Other Expenses
Intoxicants
Karnataka
Food
Other Expenses
Intoxicants
Kerala
Food
Other Expenses
Intoxicants
Madhya Pradesh
Food
Other Expenses
Intoxicants
Maharashtra
Food
Other Expenses
Intoxicants
Manipur
Food
Other Expenses
Intoxicants
Meghalaya
Food
Other Expenses
Intoxicants
Mizoram
Food
Other Expenses
Intoxicants
Nagaland
Food
Other Expenses
Intoxicants
Odisha
Food
Other Expenses
Intoxicants
Punjab
Food
Other Expenses
Intoxicants
Rajasthan
Food
Other Expenses
Intoxicants
Sikkim
Food
Other Expenses
Intoxicants
Tamil Nadu
Food
Other Expenses
Intoxicants
Telangana
Tripura
Uttar Pradesh
Uttarakhand

Figure 4: Statewise household category splitup

```
data_a2=pd.read_excel('/kaggle/input/mospi-hces/Table
A2.xlsx',header=[0,1,2,3,4])
data_a2.drop(data_a2.tail(2).index,inplace=True)
data_a2.columns=['col0','col1','col2','col3','col4','col5','col6','col
7','col8','col9','col10','col11','col12']
data_a2['col0']=data_a2['col0'].ffill()
data_a2['col1']=data_a2['col1'].ffill()
data_a2.head()
```

```
    col0            col1    col2    col3     col4     col5     col6
col7  \
0  Rural  Andhra Pradesh    0-5%  3723.0   5082.0   6093.0  11175.0
2369.0
1  Rural  Andhra Pradesh   5-10%  3701.0   5153.0   6331.0  11484.0
2138.0
2  Rural  Andhra Pradesh  10-20%  7785.0  11566.0  12527.0  24093.0
4107.0
3  Rural  Andhra Pradesh  20-30%  8260.0  12320.0  12376.0  24696.0
3612.0
4  Rural  Andhra Pradesh  30-40%  8729.0  11933.0  12732.0  24666.0
3633.0

     col8    col9    col10  col11   col12
0  2221.0  4590.0  1952.46  319.0  1387.0
1  2162.0  4300.0  2441.62  281.0  1208.0
2  3331.0  7438.0  2861.11  542.0  2192.0
3  3232.0  6844.0  3320.33  551.0  2098.0
4  3272.0  6905.0  3721.22  573.0  2071.0
```

```
data_a2.tail(5)
```

```
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/
format.py:1458: RuntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()

       col0       col1                    col2      col3       col4
col5  \
1031  Urban  All-India                  80-90%  110746.0   151463.0
140174.0
1032  Urban  All-India                  90-95%   64548.0    77651.0
70456.0
```

```
1033  Urban  All-India             95-100%   82376.0    84477.0
70480.0
1034  Urban  All-India         All classes  895030.0  1391106.0
1340397.0
1035  Urban  All-India  Sample no. of hhs.  106732.0   170996.0
165611.0

           col6      col7      col8      col9     col10     col11
col12
1031   291637.0   28714.0   23773.0   52487.0   9582.39   13088.0
41848.0
1032   148106.0   12615.0   11287.0   23901.0  12399.19    7266.0
20190.0
1033   154957.0    9374.0    7698.0   17073.0  20823.69    8907.0
18962.0
1034  2731503.0  372558.0  336574.0  709131.0   6458.70  106732.0
422895.0
1035   336607.0   45339.0   40949.0   86288.0       NaN       NaN
NaN
```

```
len(data_a2)
```

```
1036
```

```
list_rural_MPCE_0_30=[]
list_rural_MPCE_30_70=[]
list_rural_MPCE_70_100=[]
list_rural_average_MPCE=[]

list_urban_MPCE_0_30=[]
list_urban_MPCE_30_70=[]
list_urban_MPCE_70_100=[]
list_urban_average_MPCE=[]

list_state=[]

for i in range(0,len(data_a2),28):
    state=data_a2.iloc[i].loc['col1']
    print('-------------------------------------------------------')
    print(state)
    rural_mpce_0_30=(data_a2.iloc[i].loc['col10']
+data_a2.iloc[i+1].loc['col10']+data_a2.iloc[i+2].loc['col10']
+data_a2.iloc[i+3].loc['col10'])/4
    rural_mpce_30_70=(data_a2.iloc[i+4].loc['col10']
+data_a2.iloc[i+5].loc['col10']+data_a2.iloc[i+6].loc['col10']
+data_a2.iloc[i+7].loc['col10'])/4
    rural_mpce_70_100=(data_a2.iloc[i+8].loc['col10']
+data_a2.iloc[i+9].loc['col10']+data_a2.iloc[i+10].loc['col10']
+data_a2.iloc[i+11].loc['col10'])/4
    rural_average_mpce=data_a2.iloc[i+12].loc['col10']
    print('rural')
    print('*********************')
    print(rural_mpce_0_30)
    print(rural_mpce_30_70)
    print(rural_mpce_70_100)
    print(rural_average_mpce)



    urban_mpce_0_30=(data_a2.iloc[i+14].loc['col10']
+data_a2.iloc[i+15].loc['col10']+data_a2.iloc[i+16].loc['col10']
+data_a2.iloc[i+17].loc['col10'])/4
    urban_mpce_30_70=(data_a2.iloc[i+18].loc['col10']
+data_a2.iloc[i+19].loc['col10']+data_a2.iloc[i+20].loc['col10']
+data_a2.iloc[i+21].loc['col10'])/4
    urban_mpce_70_100=(data_a2.iloc[i+22].loc['col10']
+data_a2.iloc[i+23].loc['col10']+data_a2.iloc[i+24].loc['col10']
+data_a2.iloc[i+25].loc['col10'])/4
    urban_average_mpce=data_a2.iloc[i+26].loc['col10']

    print('urban')
    print('**********************')
    print(urban_mpce_0_30)
    print(urban_mpce_30_70)
    print(urban_mpce_70_100)
    print(urban_average_mpce)

    list_state.append(state)

    list_rural_MPCE_0_30.append(rural_mpce_0_30)
    list_rural_MPCE_30_70.append(rural_mpce_30_70)
    list_rural_MPCE_70_100.append(rural_mpce_70_100)
    list_rural_average_MPCE.append(rural_average_mpce)

    list_urban_MPCE_0_30.append(urban_mpce_0_30)
    list_urban_MPCE_30_70.append(urban_mpce_30_70)
    list_urban_MPCE_70_100.append(urban_mpce_70_100)
    list_urban_average_MPCE.append(urban_average_mpce)
```

```
-------------------------------------------------------
Andhra Pradesh
rural
*********************
2643.88
4382.1725
```

```
8277.2275
4870.3
urban
**********************
3294.4075000000003
5838.8325
12520.677500000002
6781.76
------------------------------------------------------
Arunachal Pradesh
rural
*********************
2395.6475
4769.05
9387.93
5276.34
urban
**********************
4185.9349999999995
7585.9925
15652.515
8635.53
------------------------------------------------------
Assam
rural
*********************
2034.5750000000003
3197.7949999999996
5412.1875
3432.41
urban
**********************
2975.9825
5253.7699999999995
11366.035
6135.51
------------------------------------------------------
Bihar
rural
*********************
1917.355
3132.615
5449.8925
3384.11
urban
**********************
2379.0474999999997
4091.4700000000003
8712.23
```

```
4767.69
------------------------------------------------------
Chhattisgarh
rural
*********************
1241.375
2178.4275000000002
4365.5625
2466.16
urban
**********************
1997.6350000000002
3738.2375
8766.3325
4483.1
------------------------------------------------------
Delhi
rural
*********************
3889.2475000000004
6081.345
10411.4875
6575.67
urban
**********************
3625.3149999999996
6355.639999999999
17119.44
8217.49
------------------------------------------------------
Goa
rural
*********************
4463.6900000000005
6659.74
11981.845000000001
7366.57
urban
**********************
4930.2474999999995
7466.1025
15534.160000000002
8733.86
------------------------------------------------------
Gujarat
rural
*********************
2216.7675
3400.9674999999997
```

```
6314.202499999999
3798.3
urban
**********************
3284.59
5685.8675
12221.445
6620.72
------------------------------------------------------
Haryana
rural
*********************
2629.8199999999997
4460.41
8045.54
4858.7
urban
**********************
3371.05
6381.74
15982.405
7910.51
------------------------------------------------------
Himachal Pradesh
rural
*********************
2842.035
4598.97
10596.2375
5560.85
urban
**********************
3593.1425
6824.1725
15652.335
8075.28
------------------------------------------------------
Jharkhand
rural
*********************
1443.3175
2448.6974999999998
4784.887500000001
2763.26
urban
**********************
2282.8149999999996
4222.5025
9234.9925
```

```
4930.99
------------------------------------------------------
Karnataka
rural
*********************
2533.8525
3973.5425
7240.389999999999
4397.47
urban
**********************
3452.8525
6463.2125
14710.177499999998
7665.88
------------------------------------------------------
Kerala
rural
*********************
2957.815
4992.4325
11131.994999999999
5923.62
urban
**********************
3109.985
5583.2825
14648.369999999999
7078.22
------------------------------------------------------
Madhya Pradesh
rural
*********************
1744.86
2827.4725
5163.67
3112.63
urban
**********************
2461.27
4174.775
9414.224999999999
4987.29
------------------------------------------------------
Maharashtra
rural
*********************
1952.3725
3391.1974999999998
```

```
7570.719999999999
4010.45
urban
**********************
2983.4125
5548.6224999999995
13027.314999999999
6657.03
------------------------------------------------------
Manipur
rural
*********************
2504.0625
4022.835
7000.547500000001
4360.42
urban
**********************
2784.885
4479.66
7879.9375
4880.47
------------------------------------------------------
Meghalaya
rural
*********************
1986.1524999999997
3234.99
5718.0
3513.84
urban
**********************
3256.6775000000002
5591.110000000001
11475.544999999998
6433.36
------------------------------------------------------
Mizoram
rural
*********************
2886.1025
4633.855
8925.0125
5223.69
urban
**********************
4382.535
7095.12
12279.857499999998
```

```
7655.03
------------------------------------------------------
Nagaland
rural
*********************
2376.73
3927.2
7459.3925
4393.1
urban
**********************
4020.5950000000003
6401.7025
11883.057499999999
7097.75
------------------------------------------------------
Odisha
rural
*********************
1668.8000000000002
2655.4925000000003
4927.935
2949.63
urban
**********************
2203.76
4212.98
10407.369999999999
5187.39
------------------------------------------------------
Punjab
rural
*********************
3093.08
4828.6675000000005
8679.6675
5314.75
urban
**********************
3383.6950000000006
5679.825
11761.815
6543.52
------------------------------------------------------
Rajasthan
rural
*********************
2115.1125
3668.8599999999997
```

```
7945.467500000001
4263.14
urban
**********************
2865.6749999999997
4962.025
11199.2425
5913.06
------------------------------------------------------
Sikkim
rural
*********************
4483.9575
7178.272500000001
12404.349999999999
7730.89
urban
**********************
6250.1725
10271.39
22167.655
12105.11
------------------------------------------------------
Tamil Nadu
rural
*********************
2880.8375
4721.467500000001
9097.0125
5310.34
urban
**********************
3757.7475
6589.900000000001
13981.072500000002
7630.05
------------------------------------------------------
Telangana
rural
*********************
2837.7475000000004
4457.6225
7573.1525
4802.23
urban
**********************
4107.3125
6977.397500000001
14977.619999999999
```

```
8158.44
------------------------------------------------------
Tripura
rural
*********************
3130.6025
4836.6625
8129.98
5206.25
urban
**********************
4272.28
6611.6375
12443.91
7404.69
------------------------------------------------------
Uttar Pradesh
rural
*********************
1812.0625
2872.86
5338.4775
3190.98
urban
**********************
2472.89
4197.5725
9636.0075
5040.41
------------------------------------------------------
Uttarakhand
rural
*********************
2505.6124999999997
4186.29
7806.102500000001
4640.93
urban
**********************
3169.7525
6001.465
13168.695
7004.37
------------------------------------------------------
West Bengal
rural
*********************
1858.185
2916.5950000000003
```

```
5374.9275
3239.16
urban
**********************
2446.7925
4385.2125
10148.885
5267.2
------------------------------------------------------
Andaman & Nicobar Islands
rural
*********************
3893.8175
6382.902499999999
13211.1175
7331.6
urban
**********************
5509.2699999999995
9079.455
17903.98
10268.15
------------------------------------------------------
Chandigarh
rural
*********************
4585.66
6699.294999999999
12126.7425
7466.86
urban
**********************
5796.7125
11282.599999999999
22327.04
12575.28
------------------------------------------------------
Dadra and Nagar Haveli & Daman and Diu
rural
*********************
1932.5075000000002
3641.74
7648.6625
4184.11
urban
**********************
3829.225
5696.6925
10043.1525
```

```
6298.49
------------------------------------------------------
Jammu & Kashmir
rural
*********************
2304.6150000000002
3728.3424999999997
7538.6075
4295.69
urban
**********************
3299.8099999999995
5603.67
10439.7275
6178.51
------------------------------------------------------
Ladakh
rural
*********************
1811.585
3705.88
7211.85
4035.3
urban
**********************
2893.44
5517.6375
11075.0275
6214.52
------------------------------------------------------
Lakshadweep
rural
*********************
3375.8775
4934.5175
10489.985
5895.46
urban
**********************
3347.4875
4855.3125
9031.7425
5474.78
------------------------------------------------------
Puducherry
rural
*********************
3683.0024999999996
5554.3875
12249.965
```

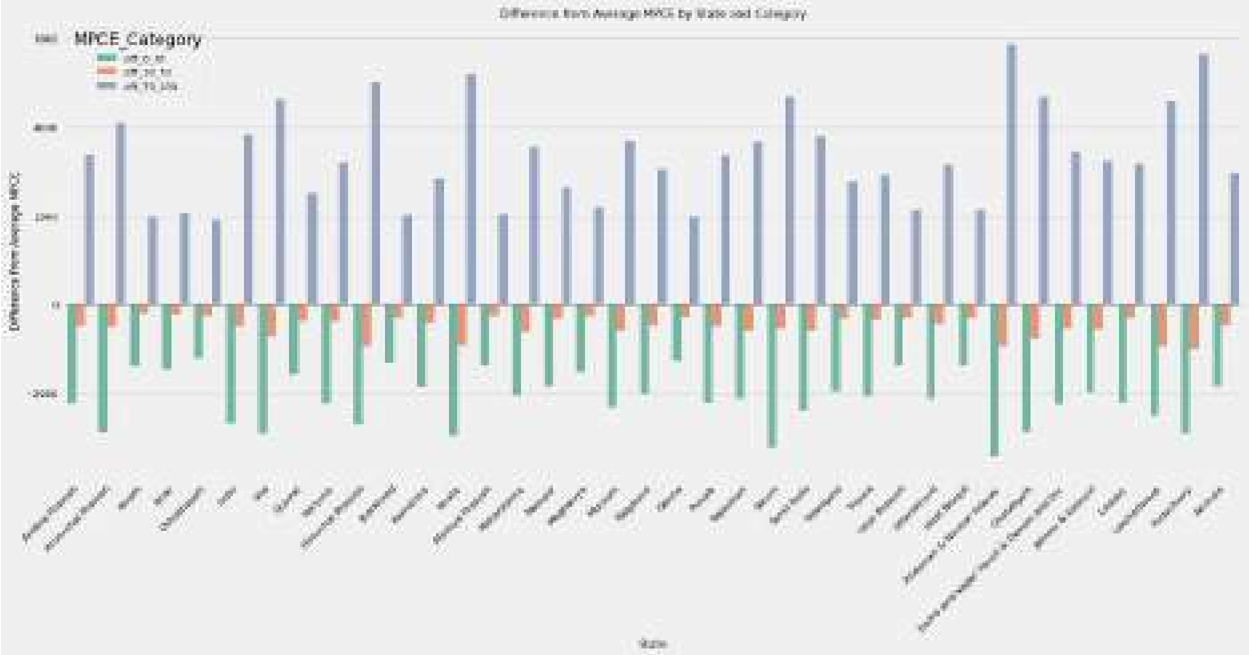
```
6590.36
urban
**********************
3972.49
7096.3175
12969.182499999999
7706.44
------------------------------------------------------
All-India
rural
*********************
1929.95
3301.0150000000003
6738.2
3773.06
urban
**********************
2881.7275
5374.1675
12619.6025
6458.7
```

```python
data_rural_MPCE=pd.DataFrame(
    {
        'state': list_state,
        'MPCE_0_30': list_rural_MPCE_0_30,
        'MPCE_30_70': list_rural_MPCE_30_70,
        'MPCE_70_100': list_rural_MPCE_70_100,
        'average_MPCE':list_rural_average_MPCE
    }
)

data_rural_MPCE.head()
```

```
               state  MPCE_0_30  MPCE_30_70  MPCE_70_100  average_MPCE
0     Andhra Pradesh  2643.8800   4382.1725    8277.2275       4870.30
1  Arunachal Pradesh  2395.6475   4769.0500    9387.9300       5276.34
2              Assam  2034.5750   3197.7950    5412.1875       3432.41
3              Bihar  1917.3550   3132.6150    5449.8925       3384.11
4       Chhattisgarh  1241.3750   2178.4275    4365.5625       2466.16
```

```python
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Calculate differences from average
data_rural_MPCE['diff_0_30'] = data_rural_MPCE['MPCE_0_30'] -
data_rural_MPCE['average_MPCE']
data_rural_MPCE['diff_30_70'] = data_rural_MPCE['MPCE_30_70'] -
data_rural_MPCE['average_MPCE']
```

```
data_rural_MPCE['diff_70_100'] = data_rural_MPCE['MPCE_70_100'] -
data_rural_MPCE['average_MPCE']

# Reshape data for plotting
plot_data = pd.melt(data_rural_MPCE,
                    id_vars=['state'],
                    value_vars=['diff_0_30', 'diff_30_70',
'diff_70_100'],
                    var_name='MPCE_Category',
                    value_name='Difference')

# Create the plot
plt.figure(figsize=(15, 8))
sns.barplot(data=plot_data,
            x='state',
            y='Difference',
            hue='MPCE_Category',
            palette='Set2')

# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Difference from Average MPCE by State and Category')
plt.xlabel('State')
plt.ylabel('Difference from Average MPCE')

# Add horizontal line at y=0
plt.axhline(y=0, color='black', linestyle='-', alpha=0.3)

# Adjust layout to prevent label cutoff
plt.tight_layout()

# Show plot
plt.show()

# Alternative: Create separate plots for each state
fig, axes = plt.subplots(nrows=(len(data_rural_MPCE) + 2) // 3,
                        ncols=3,
                        figsize=(20, 4 * ((len(data_rural_MPCE) + 2)
// 3)))
axes = axes.flatten()

for idx, (state, data) in enumerate(data_rural_MPCE.iterrows()):
    differences = [data['diff_0_30'], data['diff_30_70'],
data['diff_70_100']]
    categories = ['0-30', '30-70', '70-100']

    sns.barplot(x=categories,
                y=differences,
                ax=axes[idx],
                palette='Set2')
    axes[idx].set_title(data['state'])
    axes[idx].axhline(y=0, color='black', linestyle='-', alpha=0.3)
    axes[idx].set_ylabel('Difference from Average')

# Remove empty subplots if any
for idx in range(len(data_rural_MPCE), len(axes)):
    fig.delaxes(axes[idx])

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

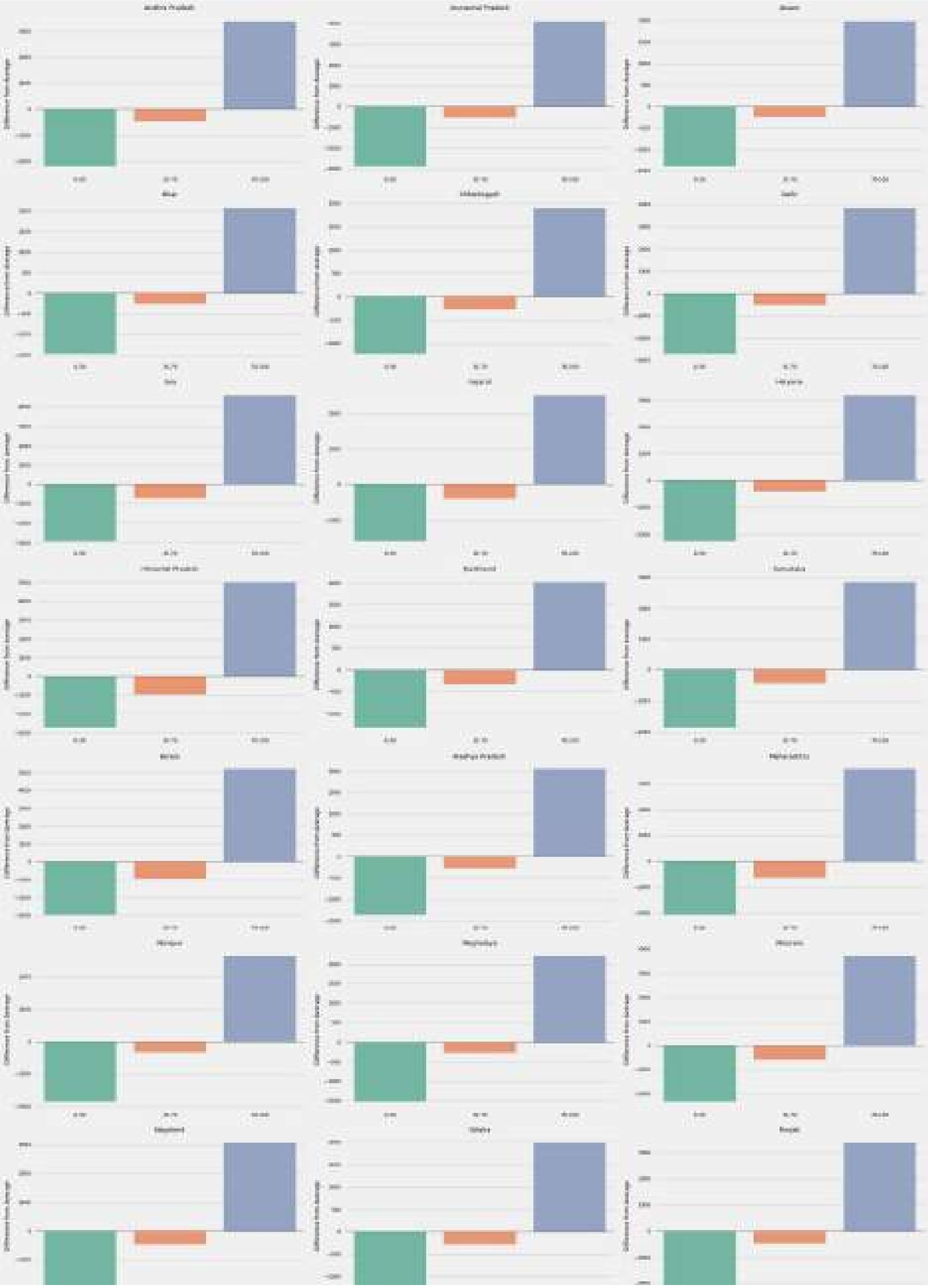


```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
```

```
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

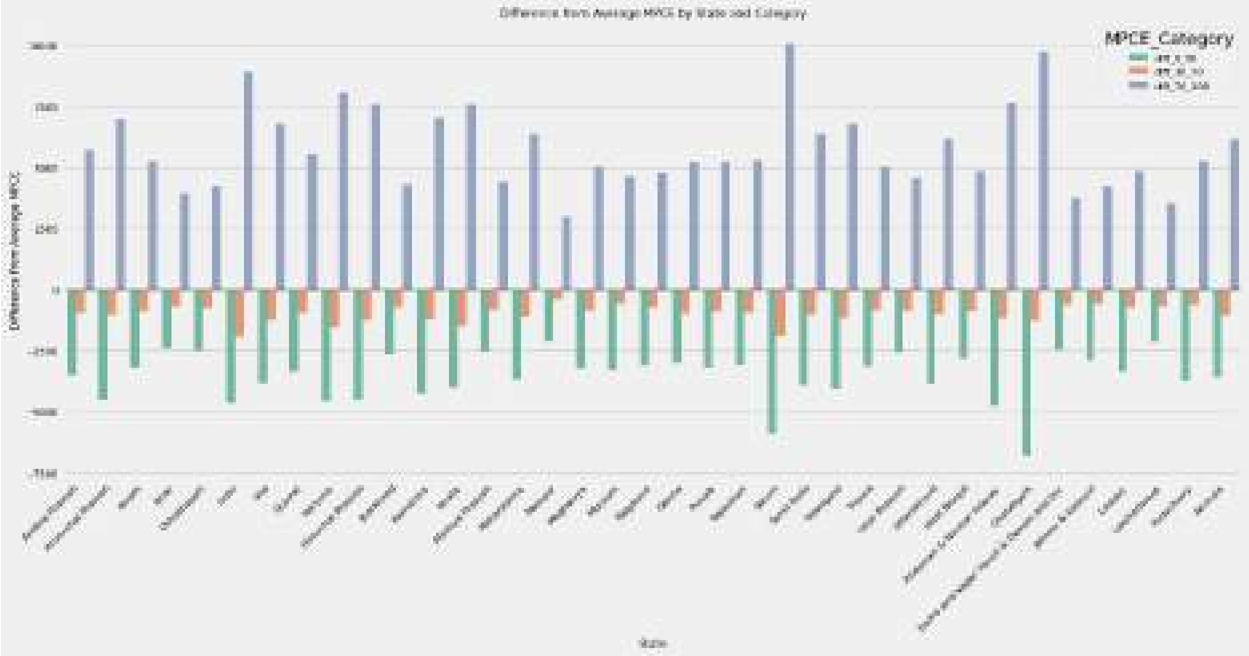
```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
```

```
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```


Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
Delhi
Goa
Gujarat
Haryana
Himachal Pradesh
Jharkhand
Karnataka
Kerala
Madhya Pradesh
Maharashtra
Manipur
Meghalaya
Mizoram
Nagaland
Odisha
Punjab

```
data_urban_MPCE=pd.DataFrame(
    {
        'state': list_state,
        'MPCE_0_30': list_urban_MPCE_0_30,
        'MPCE_30_70': list_urban_MPCE_30_70,
        'MPCE_70_100': list_urban_MPCE_70_100,
        'average_MPCE':list_urban_average_MPCE
    }
)

import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Calculate differences from average
data_urban_MPCE['diff_0_30'] = data_urban_MPCE['MPCE_0_30'] -
data_urban_MPCE['average_MPCE']
data_urban_MPCE['diff_30_70'] = data_urban_MPCE['MPCE_30_70'] -
data_urban_MPCE['average_MPCE']
data_urban_MPCE['diff_70_100'] = data_urban_MPCE['MPCE_70_100'] -
data_urban_MPCE['average_MPCE']

# Reshape data for plotting
plot_data = pd.melt(data_urban_MPCE,
                    id_vars=['state'],
                    value_vars=['diff_0_30', 'diff_30_70',
'diff_70_100'],
                    var_name='MPCE_Category',
                    value_name='Difference')

# Create the plot
plt.figure(figsize=(15, 8))
sns.barplot(data=plot_data,
            x='state',
            y='Difference',
            hue='MPCE_Category',
            palette='Set2')

# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Difference from Average MPCE by State and Category')
plt.xlabel('State')
plt.ylabel('Difference from Average MPCE')

# Add horizontal line at y=0
plt.axhline(y=0, color='black', linestyle='-', alpha=0.3)

# Adjust layout to prevent label cutoff
plt.tight_layout()
```

```
# Show plot
plt.show()

# Alternative: Create separate plots for each state
fig, axes = plt.subplots(nrows=(len(data_urban_MPCE) + 2) // 3,
                        ncols=3,
                        figsize=(20, 4 * ((len(data_urban_MPCE) + 2)
// 3)))
axes = axes.flatten()

for idx, (state, data) in enumerate(data_urban_MPCE.iterrows()):
    differences = [data['diff_0_30'], data['diff_30_70'],
data['diff_70_100']]
    categories = ['0-30', '30-70', '70-100']

    sns.barplot(x=categories,
                y=differences,
                ax=axes[idx],
                palette='Set2')

    axes[idx].set_title(data['state'])
    axes[idx].axhline(y=0, color='black', linestyle='-', alpha=0.3)
    axes[idx].set_ylabel('Difference from Average')

# Remove empty subplots if any
for idx in range(len(data_urban_MPCE), len(axes)):
    fig.delaxes(axes[idx])

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



```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

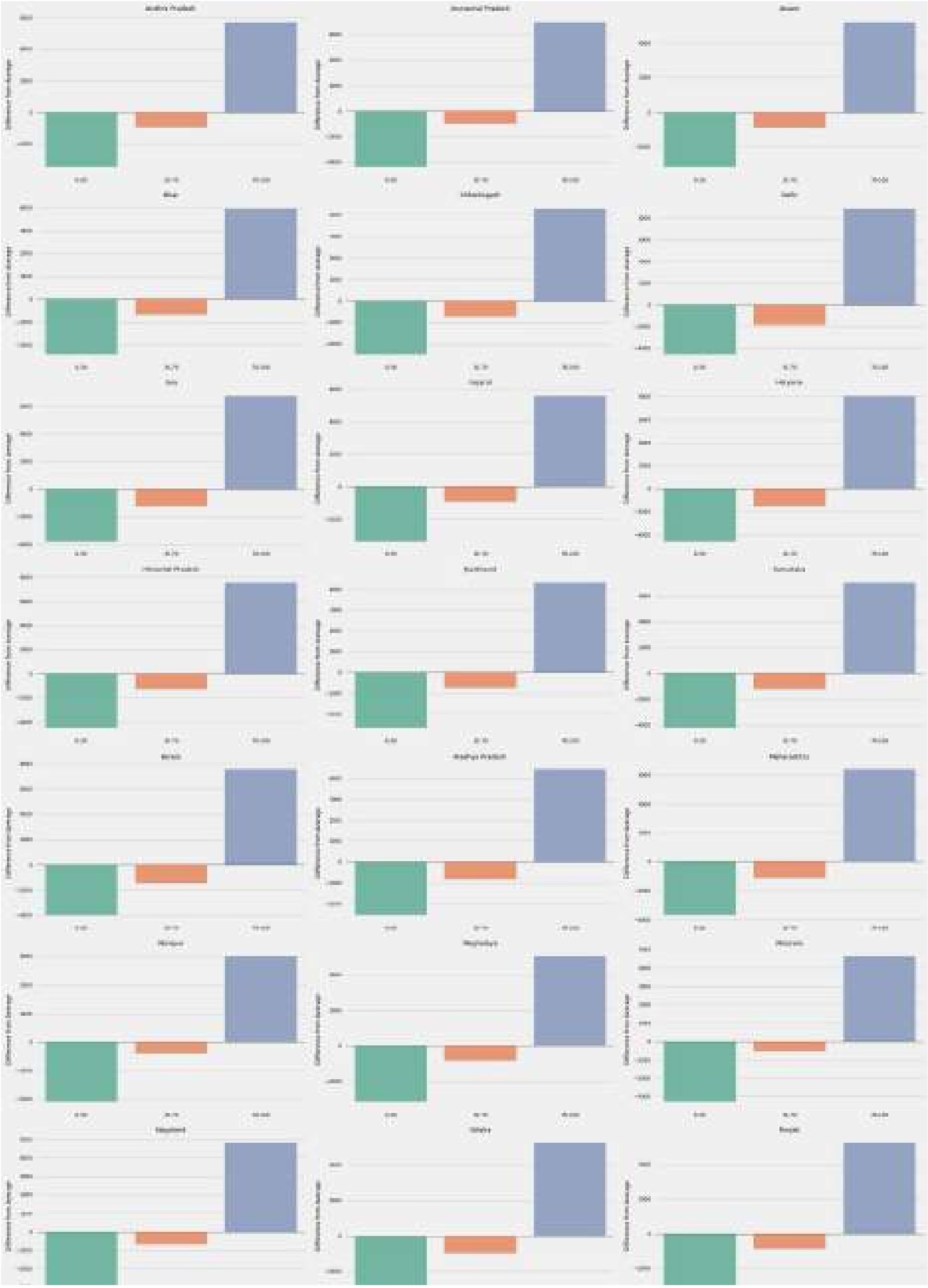

Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
Delhi
Goa
Gujarat
Haryana
Himachal Pradesh
Jharkhand
Karnataka
Kerala
Madhya Pradesh
Maharashtra
Manipur
Meghalaya
Mizoram
Nagaland
Odisha
Punjab

Figure 4: Percentage of households in each MPCE categories (broad)

```
data_a2=pd.read_excel('/kaggle/input/mospi-hces/Table
A2.xlsx',header=[0,1,2,3,4])
data_a2.drop(data_a2.tail(2).index,inplace=True)
data_a2.columns=['col0','col1','col2','col3','col4','col5','col6','col
7','col8','col9','col10','col11','col12']
data_a2['col0']=data_a2['col0'].ffill()
data_a2['col1']=data_a2['col1'].ffill()
data_a2.head()

    col0            col1    col2    col3     col4     col5     col6
col7  \
0  Rural  Andhra Pradesh    0-5%  3723.0   5082.0   6093.0  11175.0
2369.0
1  Rural  Andhra Pradesh   5-10%  3701.0   5153.0   6331.0  11484.0
2138.0
2  Rural  Andhra Pradesh  10-20%  7785.0  11566.0  12527.0  24093.0
4107.0
3  Rural  Andhra Pradesh  20-30%  8260.0  12320.0  12376.0  24696.0
3612.0
4  Rural  Andhra Pradesh  30-40%  8729.0  11933.0  12732.0  24666.0
3633.0

     col8    col9    col10  col11   col12
0  2221.0  4590.0  1952.46  319.0  1387.0
1  2162.0  4300.0  2441.62  281.0  1208.0
2  3331.0  7438.0  2861.11  542.0  2192.0
3  3232.0  6844.0  3320.33  551.0  2098.0
4  3272.0  6905.0  3721.22  573.0  2071.0

list_rural_MPCE_0_30=[]
list_rural_MPCE_30_70=[]
list_rural_MPCE_70_100=[]
list_rural_average_MPCE=[]

list_urban_MPCE_0_30=[]
list_urban_MPCE_30_70=[]
list_urban_MPCE_70_100=[]
list_urban_average_MPCE=[]

list_state=[]

for i in range(0,len(data_a2),28):
    state=data_a2.iloc[i].loc['col1']
    print('-------------------------------------------------------')
    print(state)
    rural_mpce_0_30=(data_a2.iloc[i].loc['col11']
+data_a2.iloc[i+1].loc['col11']+data_a2.iloc[i+2].loc['col11']
+data_a2.iloc[i+3].loc['col11'])
    rural_mpce_30_70=(data_a2.iloc[i+4].loc['col11']
+data_a2.iloc[i+5].loc['col11']+data_a2.iloc[i+6].loc['col11']
+data_a2.iloc[i+7].loc['col11'])
    rural_mpce_70_100=(data_a2.iloc[i+8].loc['col11']
+data_a2.iloc[i+9].loc['col11']+data_a2.iloc[i+10].loc['col11']
+data_a2.iloc[i+11].loc['col11'])
    rural_average_mpce=data_a2.iloc[i+12].loc['col11']



    print('rural')
    print('*********************')
    print(rural_mpce_0_30)
    print(rural_mpce_30_70)
    print(rural_mpce_70_100)
    print(rural_average_mpce)



    urban_mpce_0_30=(data_a2.iloc[i+14].loc['col11']
+data_a2.iloc[i+15].loc['col11']+data_a2.iloc[i+16].loc['col11']
+data_a2.iloc[i+17].loc['col11'])
    urban_mpce_30_70=(data_a2.iloc[i+18].loc['col11']
+data_a2.iloc[i+19].loc['col11']+data_a2.iloc[i+20].loc['col11']
+data_a2.iloc[i+21].loc['col11'])
    urban_mpce_70_100=(data_a2.iloc[i+22].loc['col11']
+data_a2.iloc[i+23].loc['col11']+data_a2.iloc[i+24].loc['col11']
+data_a2.iloc[i+25].loc['col11'])
    urban_average_mpce=data_a2.iloc[i+26].loc['col11']

    print('urban')
    print('**********************')
    print(urban_mpce_0_30)
    print(urban_mpce_30_70)
    print(urban_mpce_70_100)
    print(urban_average_mpce)

    list_state.append(state)

    list_rural_MPCE_0_30.append(rural_mpce_0_30)
    list_rural_MPCE_30_70.append(rural_mpce_30_70)
    list_rural_MPCE_70_100.append(rural_mpce_70_100)
    list_rural_average_MPCE.append(rural_average_mpce)

    list_urban_MPCE_0_30.append(urban_mpce_0_30)
    list_urban_MPCE_30_70.append(urban_mpce_30_70)
    list_urban_MPCE_70_100.append(urban_mpce_70_100)
    list_urban_average_MPCE.append(urban_average_mpce)
```

```
------------------------------------------------------
Andhra Pradesh
rural
*********************
1693.0
2430.0
2122.0
6245.0
urban
**********************
999.0
1554.0
1472.0
4025.0
------------------------------------------------------
Arunachal Pradesh
rural
*********************
604.0
943.0
1034.0
2581.0
urban
**********************
292.0
493.0
655.0
1440.0
------------------------------------------------------
Assam
rural
*********************
1428.0
2414.0
2203.0
6045.0
urban
**********************
646.0
1015.0
856.0
2517.0
------------------------------------------------------
Bihar
rural
*********************
3400.0
```

```
5190.0
5012.0
13602.0
urban
**********************
814.0
1421.0
1329.0
3564.0
------------------------------------------------------
Chhattisgarh
rural
*********************
732.0
1112.0
1023.0
2867.0
urban
**********************
578.0
766.0
797.0
2141.0
------------------------------------------------------
Delhi
rural
*********************
75.0
121.0
109.0
305.0
urban
**********************
574.0
1133.0
1224.0
2931.0
------------------------------------------------------
Goa
rural
*********************
96.0
129.0
135.0
360.0
urban
**********************
74.0
117.0
132.0
```

```
323.0
------------------------------------------------------
Gujarat
rural
*********************
1411.0
2207.0
2108.0
5726.0
urban
**********************
1310.0
2170.0
2080.0
5560.0
------------------------------------------------------
Haryana
rural
*********************
720.0
1099.0
977.0
2796.0
urban
**********************
488.0
846.0
1138.0
2472.0
------------------------------------------------------
Himachal Pradesh
rural
*********************
347.0
556.0
503.0
1406.0
urban
**********************
189.0
367.0
480.0
1036.0
------------------------------------------------------
Jharkhand
rural
*********************
983.0
1555.0
1389.0
```

```
3927.0
urban
**********************
568.0
951.0
939.0
2458.0
------------------------------------------------------
Karnataka
rural
*********************
1510.0
2651.0
2527.0
6688.0
urban
**********************
1343.0
2260.0
2098.0
5701.0
------------------------------------------------------
Kerala
rural
*********************
960.0
1509.0
1401.0
3870.0
urban
**********************
837.0
1418.0
1252.0
3507.0
------------------------------------------------------
Madhya Pradesh
rural
*********************
2097.0
3385.0
3069.0
8551.0
urban
**********************
1115.0
2066.0
2463.0
5644.0
------------------------------------------------------
```

```
Maharashtra
rural
*********************
3100.0
4819.0
3677.0
11596.0
urban
**********************
2545.0
4248.0
4370.0
11163.0
------------------------------------------------------
Manipur
rural
*********************
673.0
1013.0
886.0
2572.0
urban
**********************
587.0
854.0
820.0
2261.0
------------------------------------------------------
Meghalaya
rural
*********************
505.0
801.0
826.0
2132.0
urban
**********************
243.0
379.0
457.0
1079.0
------------------------------------------------------
Mizoram
rural
*********************
339.0
532.0
568.0
1439.0
urban
```

```
**********************
477.0
800.0
880.0
2157.0
------------------------------------------------------
Nagaland
rural
*********************
443.0
725.0
828.0
1996.0
urban
**********************
248.0
402.0
429.0
1079.0
------------------------------------------------------
Odisha
rural
*********************
1704.0
2585.0
2443.0
6732.0
urban
**********************
573.0
835.0
1045.0
2453.0
------------------------------------------------------
Punjab
rural
*********************
783.0
1190.0
1103.0
3076.0
urban
**********************
661.0
1015.0
1078.0
2754.0
------------------------------------------------------
Rajasthan
rural
```

```
*********************
2235.0
3352.0
3137.0
8724.0
urban
**********************
1114.0
1728.0
1596.0
4438.0
------------------------------------------------------
Sikkim
rural
*********************
332.0
525.0
554.0
1411.0
urban
**********************
144.0
237.0
339.0
720.0
------------------------------------------------------
Tamil Nadu
rural
*********************
2063.0
2967.0
2417.0
7447.0
urban
**********************
1785.0
2699.0
2433.0
6917.0
------------------------------------------------------
Telangana
rural
*********************
911.0
1401.0
1241.0
3553.0
urban
**********************
782.0
```

```
1243.0
1208.0
3233.0
------------------------------------------------------
Tripura
rural
*********************
763.0
1300.0
1159.0
3222.0
urban
**********************
420.0
709.0
671.0
1800.0
------------------------------------------------------
Uttar Pradesh
rural
*********************
4926.0
7715.0
6970.0
19611.0
urban
**********************
2481.0
3903.0
4243.0
10627.0
------------------------------------------------------
Uttarakhand
rural
*********************
474.0
689.0
537.0
1700.0
urban
**********************
190.0
390.0
493.0
1073.0
------------------------------------------------------
West Bengal
rural
*********************
2813.0
```

```
4263.0
3639.0
10715.0
urban
**********************
1700.0
2740.0
2981.0
7421.0
------------------------------------------------------
Andaman & Nicobar Islands
rural
*********************
165.0
247.0
232.0
644.0
urban
**********************
82.0
129.0
145.0
356.0
------------------------------------------------------
Chandigarh
rural
*********************
80.0
125.0
155.0
360.0
urban
**********************
76.0
139.0
145.0
360.0
------------------------------------------------------
Dadra and Nagar Haveli & Daman and Diu
rural
*********************
82.0
115.0
153.0
350.0
urban
**********************
56.0
114.0
154.0
```

```
324.0
------------------------------------------------------
Jammu & Kashmir
rural
*********************
449.0
712.0
600.0
1761.0
urban
**********************
503.0
665.0
604.0
1772.0
------------------------------------------------------
Ladakh
rural
*********************
94.0
144.0
121.0
359.0
urban
**********************
77.0
142.0
141.0
360.0
------------------------------------------------------
Lakshadweep
rural
*********************
39.0
84.0
129.0
252.0
urban
**********************
71.0
129.0
155.0
355.0
------------------------------------------------------
Puducherry
rural
*********************
98.0
141.0
120.0
```

```
359.0
urban
**********************
154.0
263.0
294.0
711.0
------------------------------------------------------
All-India
rural
*********************
35381.0
57989.0
61644.0
155014.0
urban
**********************
24982.0
40591.0
41159.0
106732.0
```

```
data_rural_households=pd.DataFrame(
    {
        'state': list_state,
        'households_0_30': list_rural_MPCE_0_30,
        'households_30_70': list_rural_MPCE_30_70,
        'households_70_100': list_rural_MPCE_70_100,
        'total_households':list_rural_average_MPCE
    }
)
```

```
data_rural_households.head()
```

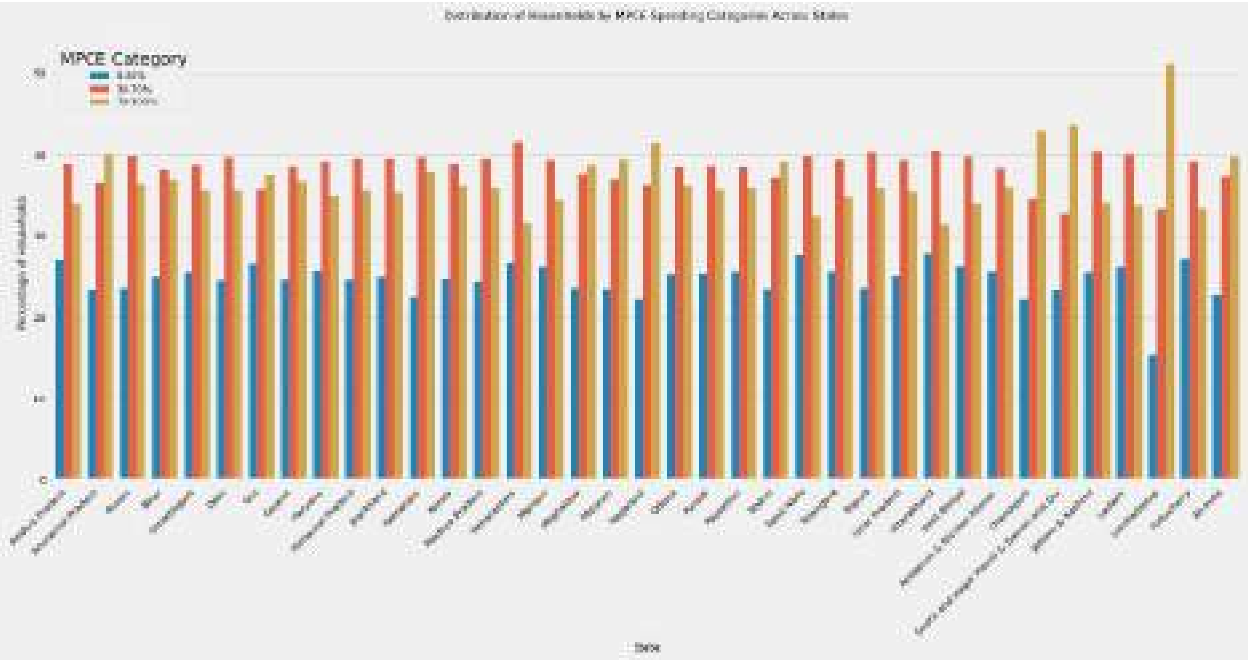
```
               state  households_0_30  households_30_70
households_70_100  \
0     Andhra Pradesh           1693.0            2430.0
2122.0
1  Arunachal Pradesh            604.0             943.0
1034.0
2              Assam           1428.0            2414.0
2203.0
3              Bihar           3400.0            5190.0
5012.0
4       Chhattisgarh            732.0            1112.0
1023.0

   total_households
0            6245.0
1            2581.0
```

```
2            6045.0
3           13602.0
4            2867.0
```

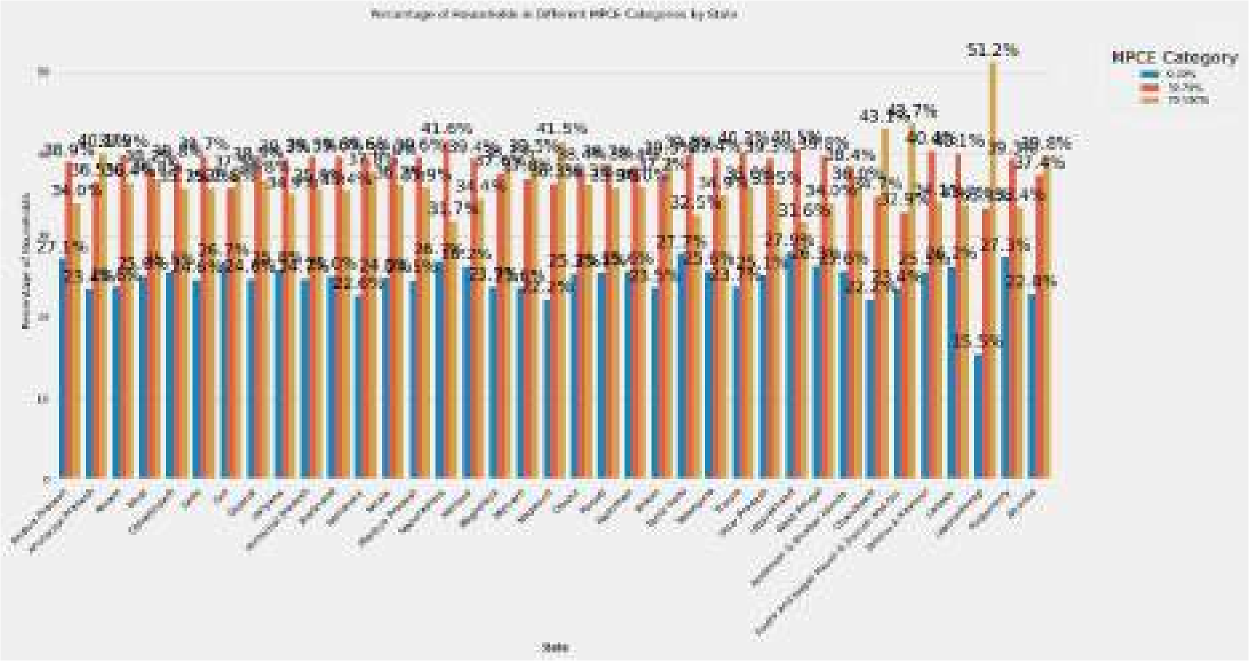
```
data_urban_households=pd.DataFrame(
    {
        'state': list_state,
        'households_0_30': list_urban_MPCE_0_30,
        'households_30_70': list_urban_MPCE_30_70,
        'households_70_100': list_urban_MPCE_70_100,
        'total_households':list_urban_average_MPCE
    }
)

data_urban_households.head()
```

```
               state  households_0_30  households_30_70  
households_70_100  \
0     Andhra Pradesh            999.0            1554.0
1472.0
1  Arunachal Pradesh            292.0             493.0
655.0
2              Assam            646.0            1015.0
856.0
3              Bihar            814.0            1421.0
1329.0
4       Chhattisgarh            578.0             766.0
797.0

   total_households
0            4025.0
1            1440.0
2            2517.0
3            3564.0
4            2141.0
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate percentages for each category
data_percentages = pd.DataFrame()
data_percentages['state'] = data_rural_households['state']
data_percentages['0-30%'] = (data_rural_households['households_0_30']
/ data_rural_households['total_households']) * 100
data_percentages['30-70%'] =
(data_rural_households['households_30_70'] /
data_rural_households['total_households']) * 100
data_percentages['70-100%'] =
(data_rural_households['households_70_100'] /
data_rural_households['total_households']) * 100

# Reshape the data for plotting
plot_data = data_percentages.melt(id_vars=['state'],
                                var_name='Category',
                                value_name='Percentage')

# Create the plot
plt.figure(figsize=(15, 8))
sns.barplot(x='state',
            y='Percentage',
            hue='Category',
            data=plot_data)

# Customize the plot
plt.title('Distribution of Households by MPCE Spending Categories
Across States', pad=20)
plt.xlabel('State')
plt.ylabel('Percentage of Households')
plt.xticks(rotation=45, ha='right')
plt.legend(title='MPCE Category')

# Adjust layout to prevent label cutoff
plt.tight_layout()

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



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate percentages for each category
data_percentages = pd.DataFrame()
data_percentages['state'] = data_rural_households['state']
data_percentages['0-30%'] = (data_rural_households['households_0_30']
/ data_rural_households['total_households']) * 100
data_percentages['30-70%'] =
(data_rural_households['households_30_70'] /
data_rural_households['total_households']) * 100
data_percentages['70-100%'] =
(data_rural_households['households_70_100'] /
data_rural_households['total_households']) * 100

# Reshape the data for plotting
plot_data = data_percentages.melt(id_vars=['state'],
                                  var_name='Category',
                                  value_name='Percentage')

# Create the plot
plt.figure(figsize=(15, 8))
ax = sns.barplot(x='state',
                 y='Percentage',
                 hue='Category',
                 data=plot_data)

# Customize the plot
plt.title('Percentage of Households in Different MPCE Categories by
State', pad=20)
plt.xlabel('State')
plt.ylabel('Percentage of Households')
plt.xticks(rotation=45, ha='right')
plt.legend(title='MPCE Category', bbox_to_anchor=(1.05, 1), loc='upper
left')

# Add percentage values on top of each bar
for container in ax.containers:
    ax.bar_label(container, fmt='%.1f%%', padding=3)

# Adjust layout to prevent label cutoff
plt.tight_layout()

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



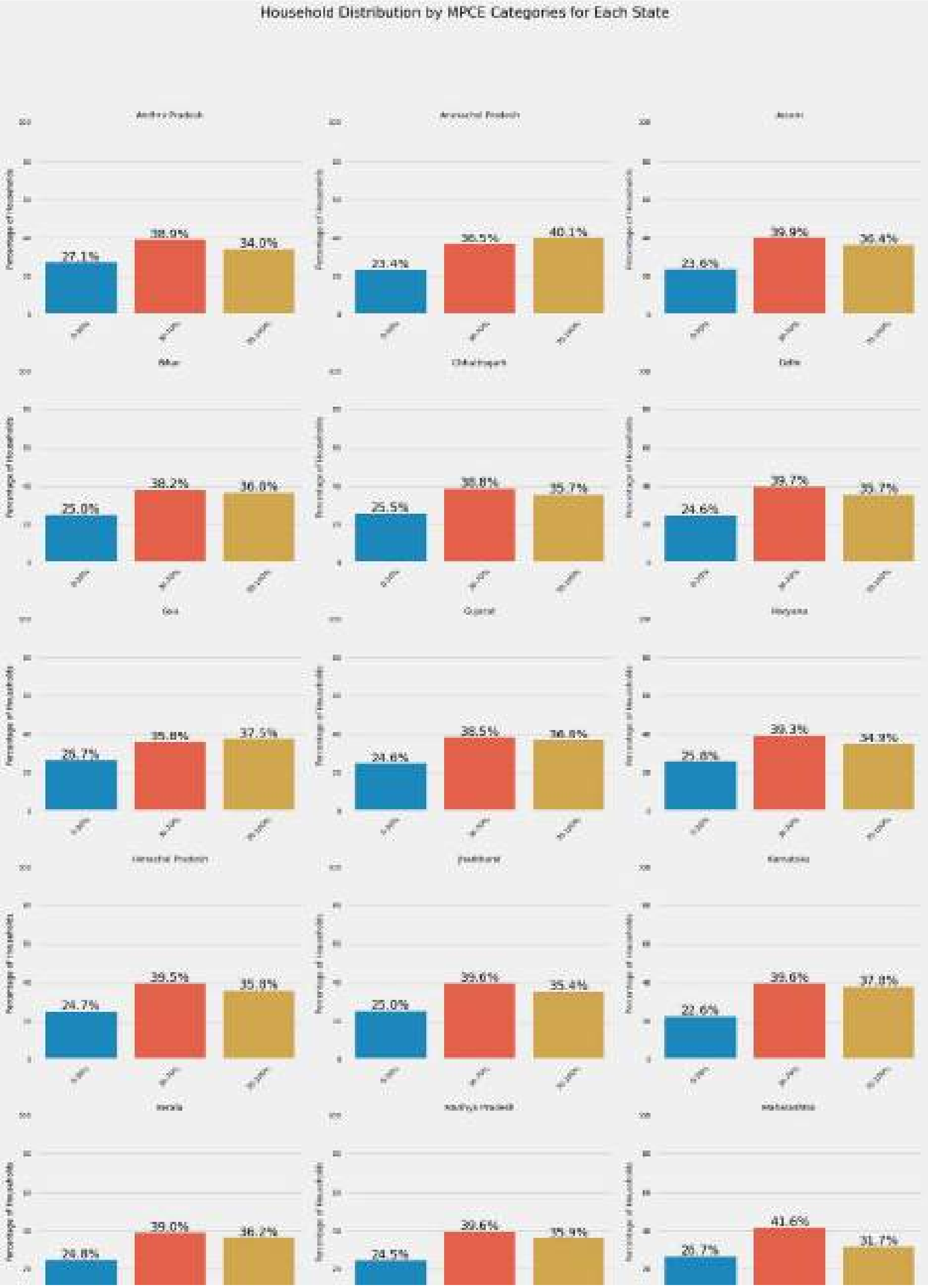
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate percentages for each category
data_percentages = pd.DataFrame()
data_percentages['state'] = data_rural_households['state']
data_percentages['0-30%'] = (data_rural_households['households_0_30']
/ data_rural_households['total_households']) * 100
data_percentages['30-70%'] =
(data_rural_households['households_30_70'] /
data_rural_households['total_households']) * 100
data_percentages['70-100%'] =
(data_rural_households['households_70_100'] /
data_rural_households['total_households']) * 100

# Calculate number of rows and columns for subplots
num_states = len(data_percentages)
num_cols = 3  # You can adjust this
num_rows = (num_states + num_cols - 1) // num_cols

# Create figure and subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(15, 4*num_rows))
axes = axes.flatten()  # Flatten the axes array for easier indexing

# Create a bar plot for each state
for idx, (state, data) in enumerate(data_percentages.iterrows()):
    # Get percentages for current state
    percentages = [data['0-30%'], data['30-70%'], data['70-100%']]
    categories = ['0-30%', '30-70%', '70-100%']
    # Create bar plot
    sns.barplot(x=categories, y=percentages, ax=axes[idx])

    # Customize subplot
    axes[idx].set_title(data['state'])
    axes[idx].set_ylabel('Percentage of Households')
    axes[idx].set_ylim(0, 100)  # Set y-axis from 0 to 100%

    # Add percentage labels on top of bars
    for i, v in enumerate(percentages):
        axes[idx].text(i, v + 1, f'{v:.1f}%', ha='center')

    # Rotate x-axis labels for better readability
    axes[idx].tick_params(axis='x', rotation=45)

# Remove empty subplots if any
for idx in range(len(data_percentages), len(axes)):
    fig.delaxes(axes[idx])

# Add a main title
fig.suptitle('Household Distribution by MPCE Categories for Each
State',
             fontsize=16, y=1.02)

# Adjust layout
plt.tight_layout()

# Show plot
plt.show()

fig.savefig('broad_household.png',dpi=100)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
```

```
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
```

```
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```


Household Distribution by MPCE Categories for Each State

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate percentages for each category
data_percentages = pd.DataFrame()
data_percentages['state'] = data_urban_households['state']
data_percentages['0-30%'] = (data_urban_households['households_0_30']
/ data_urban_households['total_households']) * 100
data_percentages['30-70%'] =
(data_urban_households['households_30_70'] /
data_urban_households['total_households']) * 100
data_percentages['70-100%'] =
(data_urban_households['households_70_100'] /
data_urban_households['total_households']) * 100

# Calculate number of rows and columns for subplots
num_states = len(data_percentages)
num_cols = 3  # You can adjust this
num_rows = (num_states + num_cols - 1) // num_cols

# Create figure and subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(15, 4*num_rows))
axes = axes.flatten()  # Flatten the axes array for easier indexing

# Create a bar plot for each state
for idx, (state, data) in enumerate(data_percentages.iterrows()):
    # Get percentages for current state
    percentages = [data['0-30%'], data['30-70%'], data['70-100%']]
    categories = ['0-30%', '30-70%', '70-100%']

    # Create bar plot
    sns.barplot(x=categories, y=percentages, ax=axes[idx])

    # Customize subplot
    axes[idx].set_title(data['state'])
    axes[idx].set_ylabel('Percentage of Households')
    axes[idx].set_ylim(0, 100)  # Set y-axis from 0 to 100%

    # Add percentage labels on top of bars
    for i, v in enumerate(percentages):
        axes[idx].text(i, v + 1, f'{v:.1f}%', ha='center')

    # Rotate x-axis labels for better readability
    axes[idx].tick_params(axis='x', rotation=45)

# Remove empty subplots if any
for idx in range(len(data_percentages), len(axes)):
    fig.delaxes(axes[idx])
```

```
# Add a main title
fig.suptitle('Household Distribution by MPCE Categories for Each
State',
             fontsize=16, y=1.02)

fig.savefig('broad_household_urban.png',dpi=100)
# Adjust layout
plt.tight_layout()

# Show plot
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
```

```
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
```

```
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

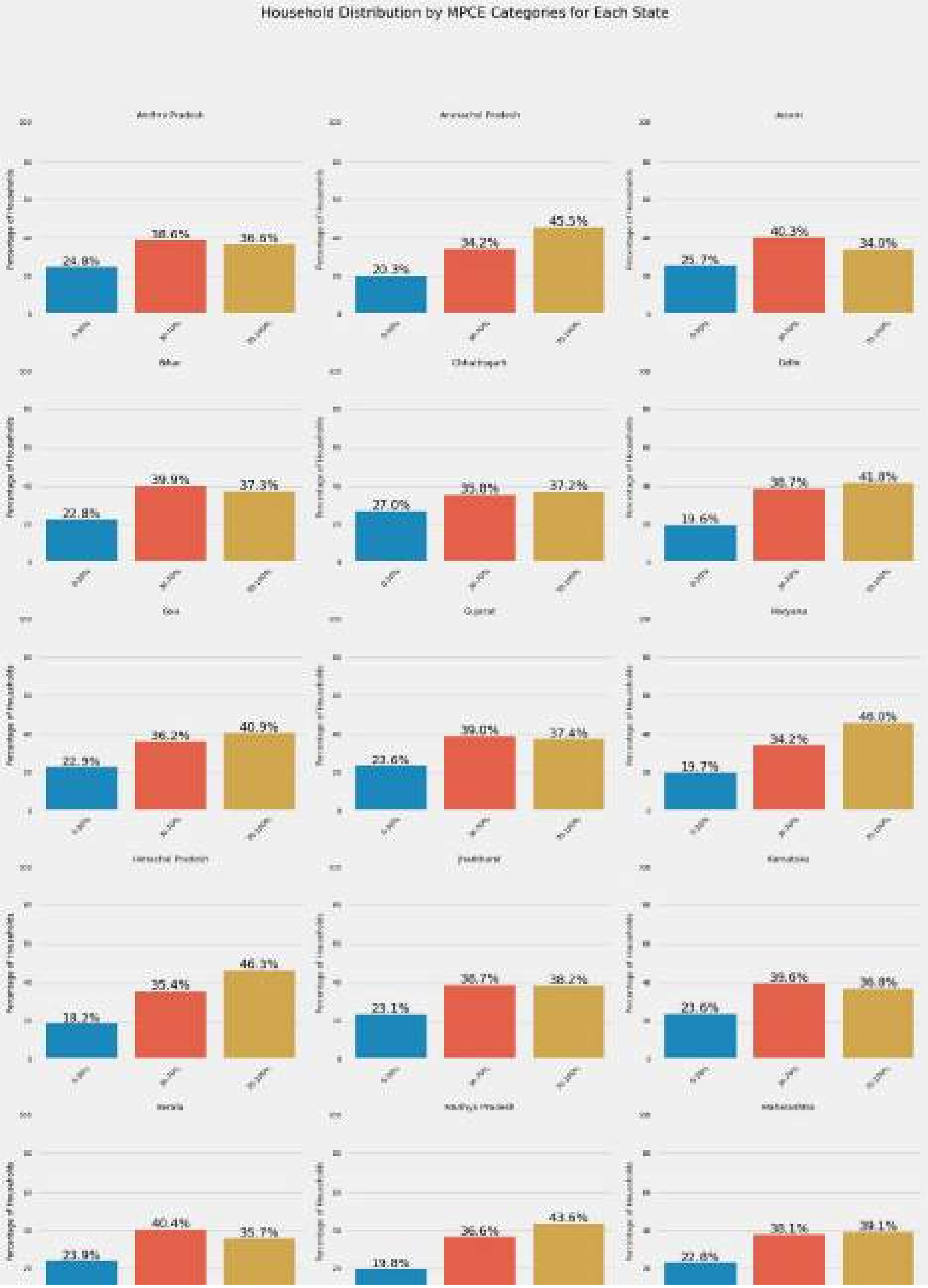

Household Distribution by MPCE Categories for Each State
Andhra Pradesh
24.8%
38.6%
36.6%
Arunachal Pradesh
20.3%
34.2%
45.5%
Assam
25.7%
40.3%
34.0%
Bihar
22.8%
39.9%
37.3%
Chhattisgarh
27.0%
35.8%
37.2%
Delhi
19.6%
38.7%
41.8%
Goa
22.9%
36.2%
40.9%
Gujarat
23.6%
39.0%
37.4%
Haryana
19.7%
34.2%
46.0%
Himachal Pradesh
18.2%
35.4%
46.3%
Jharkhand
23.1%
38.7%
38.2%
Karnataka
23.6%
39.6%
36.8%
Kerala
23.9%
40.4%
35.7%
Madhya Pradesh
19.8%
36.6%
43.6%
Maharashtra
22.8%
38.1%
39.1%
Percentage of Households

Figure 5: Percentage of households in each MPCE categories (Fine)

```
data_a7=pd.read_excel('/kaggle/input/mospi-hces/Table
A7.xlsx',header=[0,1,2,3,4])
data_a7.columns=['col0','col1','0-5%','5-10%','10-20%','20-30%','30-
40%','40-50%','50-60%','60-70%','70-80%','80-90%','90-95%','95-
100%','col14','col15','col16']
data_a7['col0']=data_a7['col0'].ffill()
data_a7.head(74)
```

```
     col0                  col1  0-5%  5-10%  10-20%  20-30%  30-40%  40-
50%  \
0   Rural        Andhra Pradesh   3.9    3.9     8.1     8.6     9.1
9.2
1   Rural     Arunachal Pradesh   3.2    3.7     7.7     8.8     8.5
8.9
2   Rural                 Assam   3.2    3.7     7.7     8.4     8.9
9.5
3   Rural                 Bihar   3.4    3.8     8.0     8.5     8.9
9.1
4   Rural          Chhattisgarh   3.7    4.0     8.2     9.0     9.3
9.2
..    ...                   ...   ...    ...     ...     ...     ...
...
69  Urban       Jammu & Kashmir   3.5    3.7     8.5     8.1     8.5
9.3
70  Urban                Ladakh   3.4    3.3     7.6     8.8     9.5
8.3
71  Urban           Lakshadweep   2.5    2.5     6.6     6.6     6.6
8.6
72  Urban            Puducherry   3.6    3.9     8.3     8.1     9.1
8.8
73  Urban             All-India   3.2    3.5     7.6     8.2     8.5
9.0

    50-60%  60-70%  70-80%  80-90%  90-95%  95-100%  col14   col15
col16
0      9.9    10.5    11.0    12.4     6.0      7.3    100   95813
6245
1      9.2     9.6    10.8    12.0     7.3     10.3    100    1953
2581
2     10.0    10.8    11.1    12.3     6.8      7.7    100   63174
6045
3      9.7    10.5    11.2    12.3     6.8      7.9    100  198464
13602
4      9.7    10.3    11.1    11.5     6.0      8.0    100   47120
```

```
2867
..      ...     ...     ...     ...     ...      ...    ...     ...
...
69      9.0     9.8    10.9    12.6     7.6      8.3    100    6347
1772
70      9.6    10.3     8.8    14.1     8.1      8.4    100      79
360
71      9.1     9.3    11.1    16.0     7.7     13.2    100      80
355
72     10.4    10.0    11.5    11.9     6.4      8.2    100    2437
711
73      9.7    10.3    11.1    12.4     7.2      9.2    100  895030
106732

[74 rows x 17 columns]
```

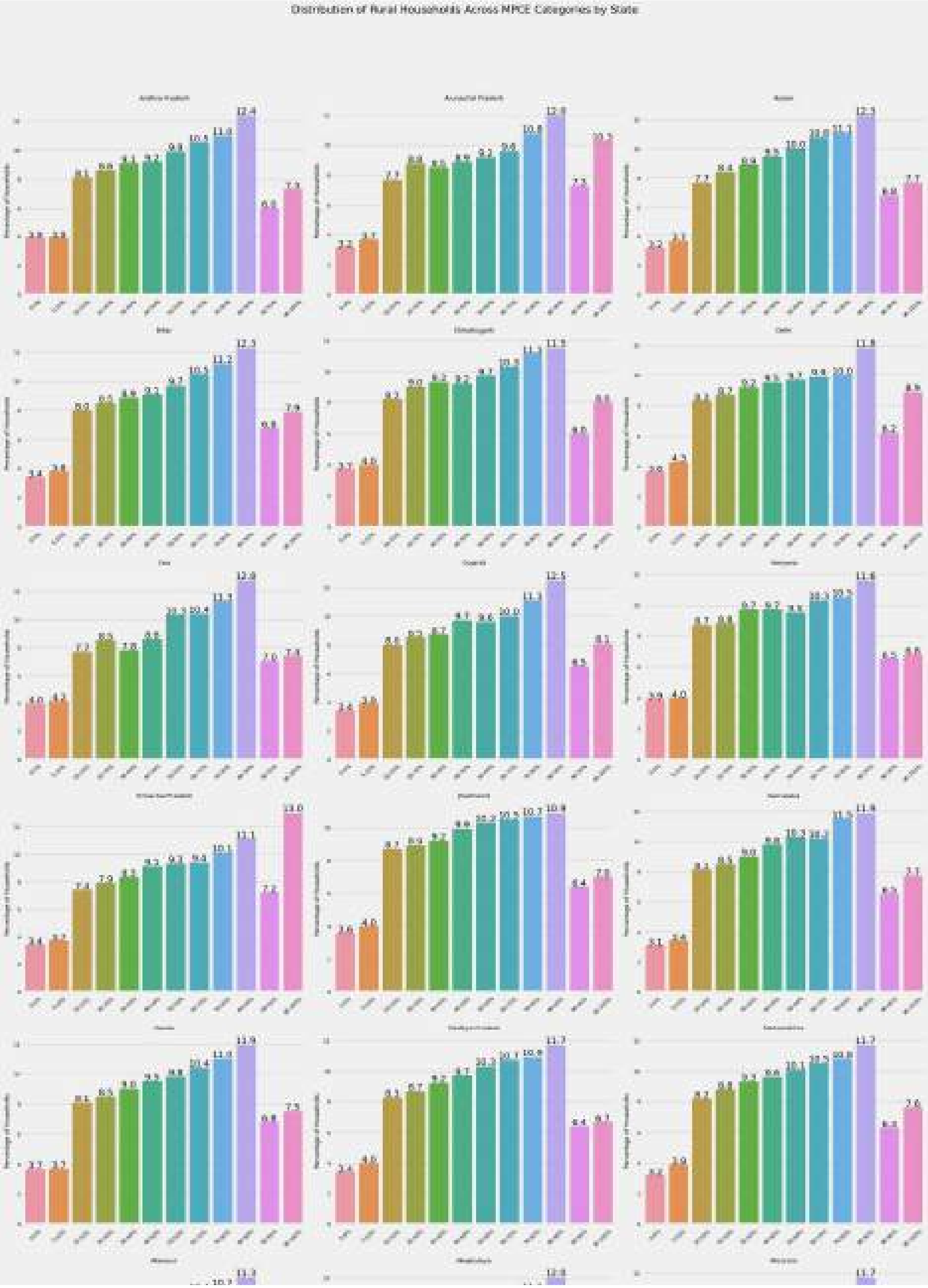
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Filter for rural data
rural_data = data_a7[data_a7['col0'] == 'Rural']

# List of percentage columns
pct_columns = ['0-5%', '5-10%', '10-20%', '20-30%', '30-40%', '40-
50%',
               '50-60%', '60-70%', '70-80%', '80-90%', '90-95%', '95-
100%']

# Calculate number of rows and columns for subplots
num_states = len(rural_data)
num_cols = 3  # You can adjust this
num_rows = (num_states + num_cols - 1) // num_cols

# Create figure and subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(20, 5*num_rows))
axes = axes.flatten()  # Flatten the axes array for easier indexing

# Create a bar plot for each state
for idx, (_, state_data) in enumerate(rural_data.iterrows()):
    # Get values for current state
    values = state_data[pct_columns]

    # Create bar plot
    sns.barplot(x=pct_columns, y=values, ax=axes[idx])

    # Customize subplot
    axes[idx].set_title(f"{state_data['col1']}", pad=10)
    axes[idx].set_ylabel('Percentage of Households')
    # Rotate x-axis labels for better readability
    axes[idx].tick_params(axis='x', rotation=45)

    # Add value labels on top of bars
    for i, v in enumerate(values):
        axes[idx].text(i, v + v*0.01, f'{v:.1f}', ha='center')

# Remove empty subplots if any
for idx in range(len(rural_data), len(axes)):
    fig.delaxes(axes[idx])

# Add a main title
fig.suptitle('Distribution of Rural Households Across MPCE Categories
by State',
             fontsize=16, y=1.02)
fig.savefig('fine_hh_rural.png',dpi=100)
# Adjust layout
plt.tight_layout()

# Show plot
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
```

```
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
```

```
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```


Distribution of Rural Households Across MPCE Categories by State

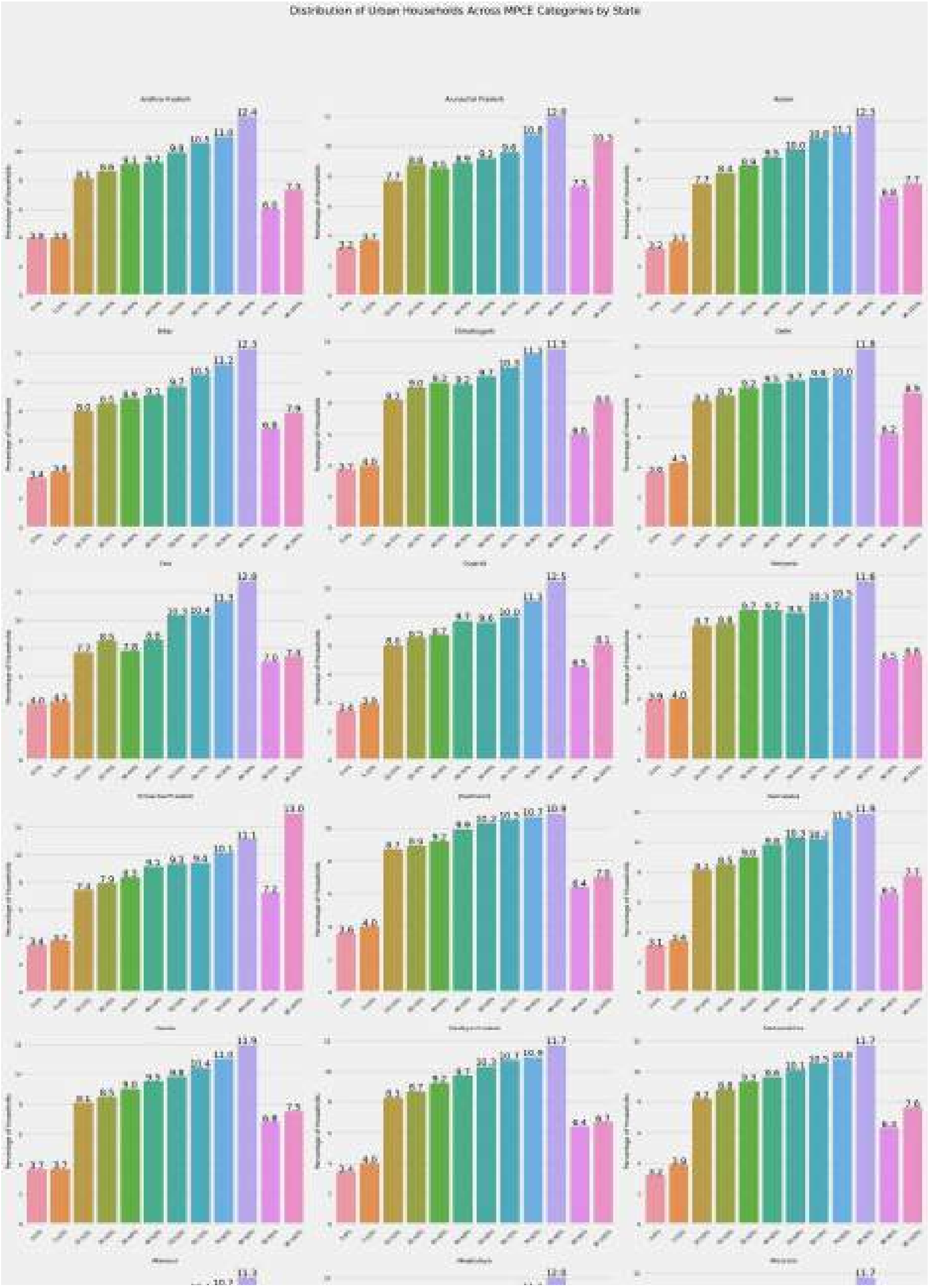
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Filter for rural data
urban_data = data_a7[data_a7['col0'] == 'Urban']

# List of percentage columns
pct_columns = ['0-5%', '5-10%', '10-20%', '20-30%', '30-40%', '40-
50%',
               '50-60%', '60-70%', '70-80%', '80-90%', '90-95%', '95-
100%']

# Calculate number of rows and columns for subplots
num_states = len(urban_data)
num_cols = 3  # You can adjust this
num_rows = (num_states + num_cols - 1) // num_cols

# Create figure and subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(20, 5*num_rows))
axes = axes.flatten()  # Flatten the axes array for easier indexing

# Create a bar plot for each state
for idx, (_, state_data) in enumerate(rural_data.iterrows()):
    # Get values for current state
    values = state_data[pct_columns]

    # Create bar plot
    sns.barplot(x=pct_columns, y=values, ax=axes[idx])

    # Customize subplot
    axes[idx].set_title(f"{state_data['col1']}", pad=10)
    axes[idx].set_ylabel('Percentage of Households')

    # Rotate x-axis labels for better readability
    axes[idx].tick_params(axis='x', rotation=45)

    # Add value labels on top of bars
    for i, v in enumerate(values):
        axes[idx].text(i, v + v*0.01, f'{v:.1f}', ha='center')

# Remove empty subplots if any
for idx in range(len(rural_data), len(axes)):
    fig.delaxes(axes[idx])

# Add a main title
fig.suptitle('Distribution of Urban Households Across MPCE Categories
by State',
             fontsize=16, y=1.02)
```

```python
# Adjust layout
plt.tight_layout()

# Show plot
plt.show()
fig.savefig('fine_hh_urban.png',dpi=100)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
```

```
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
```

```
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```



Distribution of Urban Households Across MPCE Categories by State

Figure 6: Percentage of households in each MPCE categories (Fine)

```
data_a8=pd.read_excel('/kaggle/input/mospi-hces/Table
A8R.xlsx',header=[0,1,2,3])
data_a8.columns=['col0','col1','self-employed in agriculture','self-
employed in non-agriculture','regular wage earning in
agriculture','regular wage earning in non-agriculture','casual labour
in agriculture','casual labour in non-agriculture','other
means','col9','col10','col11']

data_a8['col0']=data_a8['col0'].ffill()
data_a8['regular wage earning in agriculture']=data_a8['regular wage
earning in agriculture'].replace('-',0.0)
data_a8.head(20)
```

```
<ipython-input-194-d71afaca258a>:5: FutureWarning: Downcasting
behavior in `replace` is deprecated and will be removed in a future
version. To retain the old behavior, explicitly call
`result.infer_objects(copy=False)`. To opt-in to the future behavior,
set `pd.set_option('future.no_silent_downcasting', True)`
  data_a8['regular wage earning in agriculture']=data_a8['regular wage
earning in agriculture'].replace('-',0.0)
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
58: RuntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
```

```
                 col0                        col1  \
0      Andhra Pradesh                        0-5%
1      Andhra Pradesh                       5-10%
2      Andhra Pradesh                      10-20%
3      Andhra Pradesh                      20-30%
4      Andhra Pradesh                      30-40%
5      Andhra Pradesh                      40-50%
6      Andhra Pradesh                      50-60%
7      Andhra Pradesh                      60-70%
8      Andhra Pradesh                      70-80%
9      Andhra Pradesh                      80-90%
10     Andhra Pradesh                      90-95%
11     Andhra Pradesh                     95-100%
12     Andhra Pradesh                 All classes
```

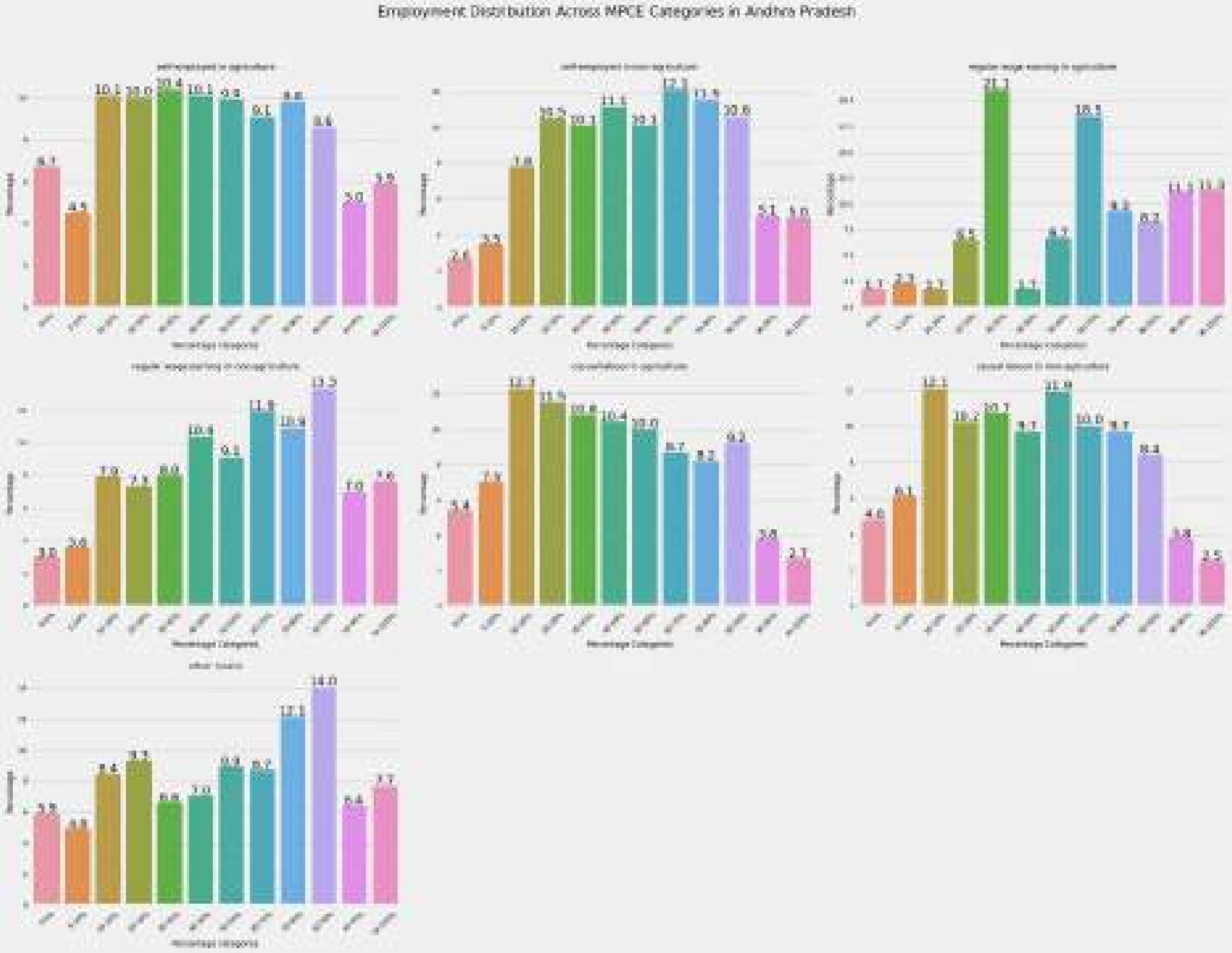
```
13     Andhra Pradesh              Avg. MPCE (Rs.)
14     Andhra Pradesh  Estd. no. of households(00)
15     Andhra Pradesh     No. of sample households
16     Andhra Pradesh     Estd. no. of persons(00)
17     Andhra Pradesh        No. of sample persons
18  Arunachal Pradesh                         0-5%
19  Arunachal Pradesh                        5-10%

    self-employed in agriculture  self-employed in non-agriculture  \
0                           6.70                              2.60   
1                           4.50                              3.50   
2                          10.10                              7.80   
3                          10.00                             10.50   
4                          10.40                             10.10   
5                          10.10                             11.10   
6                           9.90                             10.10   
7                           9.10                             12.10   
8                           9.80                             11.50   
9                           8.60                             10.60   
10                          5.00                              5.10   
11                          5.90                              5.00   
12                        100.00                            100.00   
13                       4905.71                           4999.95   
14                      24369.00                          13377.00   
15                       1586.00                            878.00   
16                      92713.00                          48099.00   
17                       6057.00                           3187.00   
18                          6.90                              0.00   
19                          6.60                              1.40   

    regular wage earning in agriculture  \
0                                  1.70   
1                                  2.30   
2                                  1.70   
3                                  6.50   
4                                 21.10   
5                                  1.70   
6                                  6.70   
7                                 18.50   
8                                  9.30   
9                                  8.20   
10                                11.10   
11                                11.30   
12                               100.00   
13                              5905.55   
14                               886.00   
15                                53.00   
16                              3151.00   
17                               186.00   
```

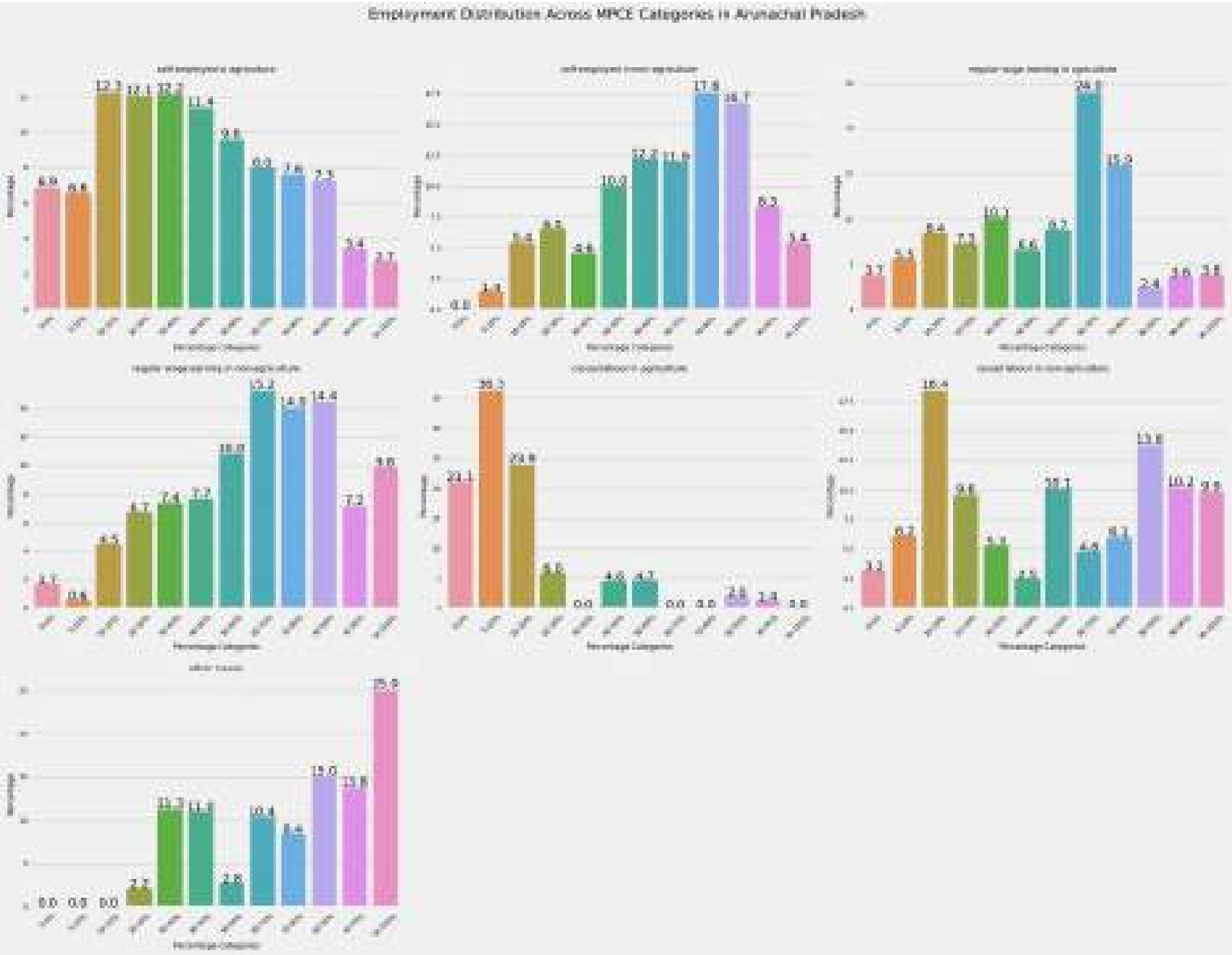
```
18                                 3.70
19                                 5.50

    regular wage earning in non-agriculture  casual labour in 
agriculture  \
0                                      3.00                     
5.40   
1                                      3.60                     
7.00   
2                                      7.90                    
12.30   
3                                      7.30                    
11.50   
4                                      8.00                    
10.80   
5                                     10.40                    
10.40   
6                                      9.10                    
10.00   
7                                     11.90                     
8.70   
8                                     10.90                     
8.20   
9                                     13.30                     
9.20   
10                                     7.00                     
3.80   
11                                     7.60                     
2.70   
12                                   100.00                   
100.00   
13                                  5334.53                  
4488.59   
14                                 10539.00                 
21066.00   
15                                   663.00                  
1446.00   
16                                 38679.00                 
69271.00   
17                                  2467.00                  
4832.00   
18                                     1.70                    
21.10   
19                                     0.60                    
36.30   

    casual labour in non-agriculture  other means      col9    col10 
col11  
0                               4.80         5.90       5.0   3723.0
```

```
319.0
1                               6.10         4.90       5.0   3701.0
281.0
2                              12.10         8.40      10.0   7785.0
542.0
3                              10.20         9.30      10.0   8260.0
551.0
4                              10.70         6.60      10.0   8729.0
573.0
5                               9.70         7.00      10.0   8837.0
586.0
6                              11.90         8.90      10.0   9532.0
623.0
7                              10.00         8.70      10.0  10055.0
648.0
8                               9.70        12.10      10.0  10516.0
643.0
9                               8.40        14.00      10.0  11880.0
705.0
10                              3.80         6.40       5.0   5778.0
364.0
11                              2.50         7.70       5.0   7019.0
410.0
12                            100.00       100.00     100.0  95813.0
6245.0
13                           4524.13      5291.51    4870.3      NaN
NaN
14                          10882.00     14694.00   95813.0      NaN
NaN
15                            732.00       887.00    6245.0      NaN
NaN
16                          40445.00     23038.00  315396.0      NaN
NaN
17                           2749.00      1429.00   20907.0      NaN
NaN
18                              3.20         0.00       5.0     62.0
92.0
19                              6.20         0.00       5.0     73.0
93.0
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# List of percentage categories to include
pct_categories = ['0-5%', '5-10%', '10-20%', '20-30%', '30-40%', '40-
50%',
                 '50-60%', '60-70%', '70-80%', '80-90%', '90-95%',
'95-100%']
```

```python
# List of employment columns to plot
employment_columns = ['self-employed in agriculture',
                      'self-employed in non-agriculture',
                      'regular wage earning in agriculture',
                      'regular wage earning in non-agriculture',
                      'casual labour in agriculture',
                      'casual labour in non-agriculture',
                      'other means']

# Get unique states
states = data_a8['col0'].unique()

# Create plots for each state
for state in states:
    # Filter data for current state
    state_data = data_a8[data_a8['col0'] == state]

    # Filter rows for percentage categories we want
    state_data = state_data[state_data['col1'].isin(pct_categories)]

    # Create subplots for each employment type
    fig, axes = plt.subplots(3, 3, figsize=(20, 15))
    axes = axes.flatten()

    # Create a bar plot for each employment type
    for idx, column in enumerate(employment_columns):
        if idx < len(axes):  # Ensure we don't exceed number of
subplots
            # Create bar plot
            sns.barplot(data=state_data, x='col1', y=column,
ax=axes[idx])

            # Customize subplot
            axes[idx].set_title(column, pad=10)
            axes[idx].set_xlabel('Percentage Categories')
            axes[idx].set_ylabel('Percentage')

            # Rotate x-axis labels
            axes[idx].tick_params(axis='x', rotation=45)

            # Add value labels on bars
            for i, v in enumerate(state_data[column]):
                axes[idx].text(i, v + v*0.01, f'{v:.1f}', ha='center')

    # Remove extra subplots if any
    for idx in range(len(employment_columns), len(axes)):
        fig.delaxes(axes[idx])

    # Add main title for the state
    fig.suptitle(f'Employment Distribution Across MPCE Categories in
{state}',
                 fontsize=16, y=1.02)

    # Adjust layout
    plt.tight_layout()

    # Show plot
    plt.show()
    fig.savefig('emp_dist.png',dpi=100)
```





Employment Distribution Across MPCE Categories in Arunachal Pradesh

self-employed in agriculture

self-employed in non-agriculture

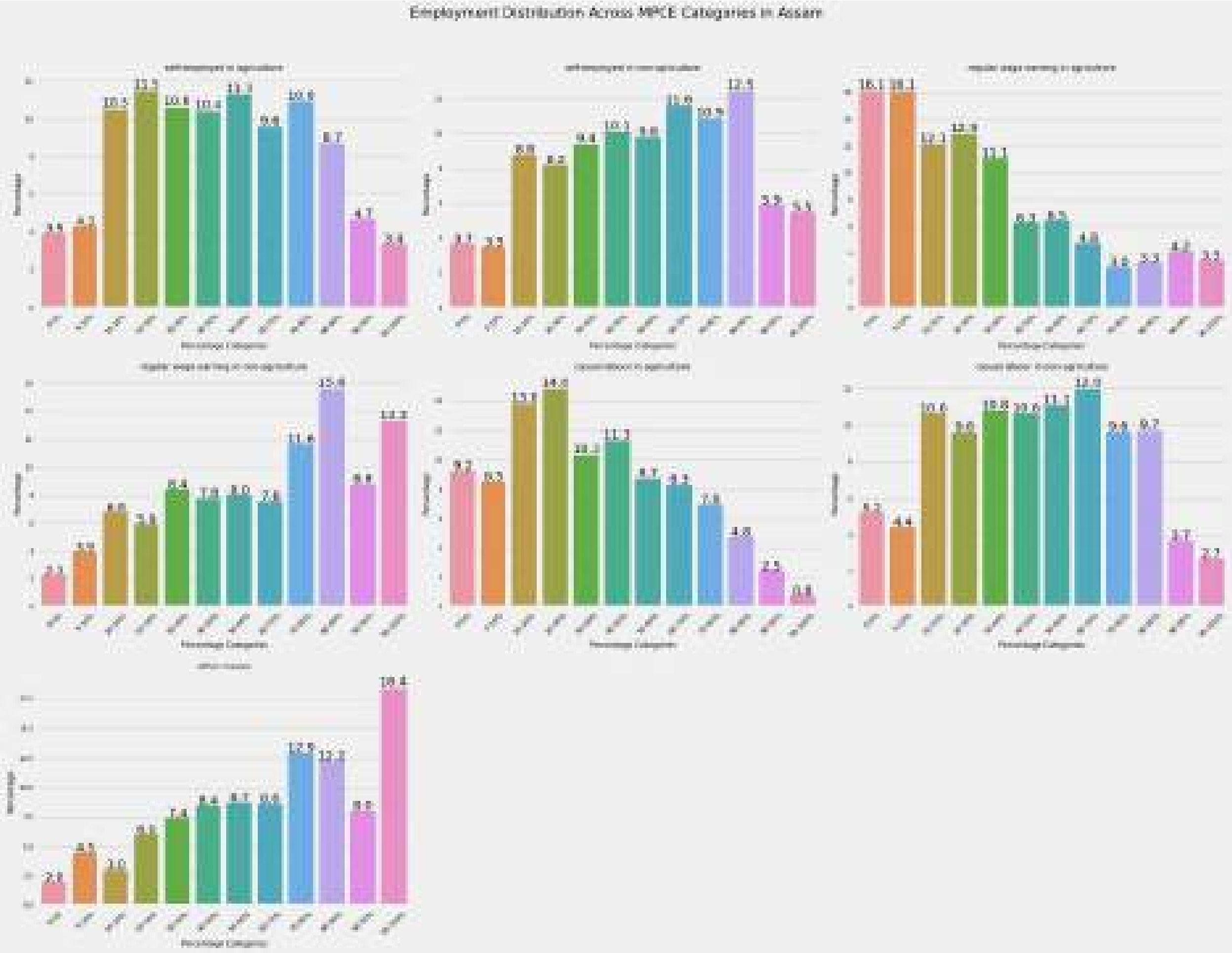
regular wage/salary earning in agriculture

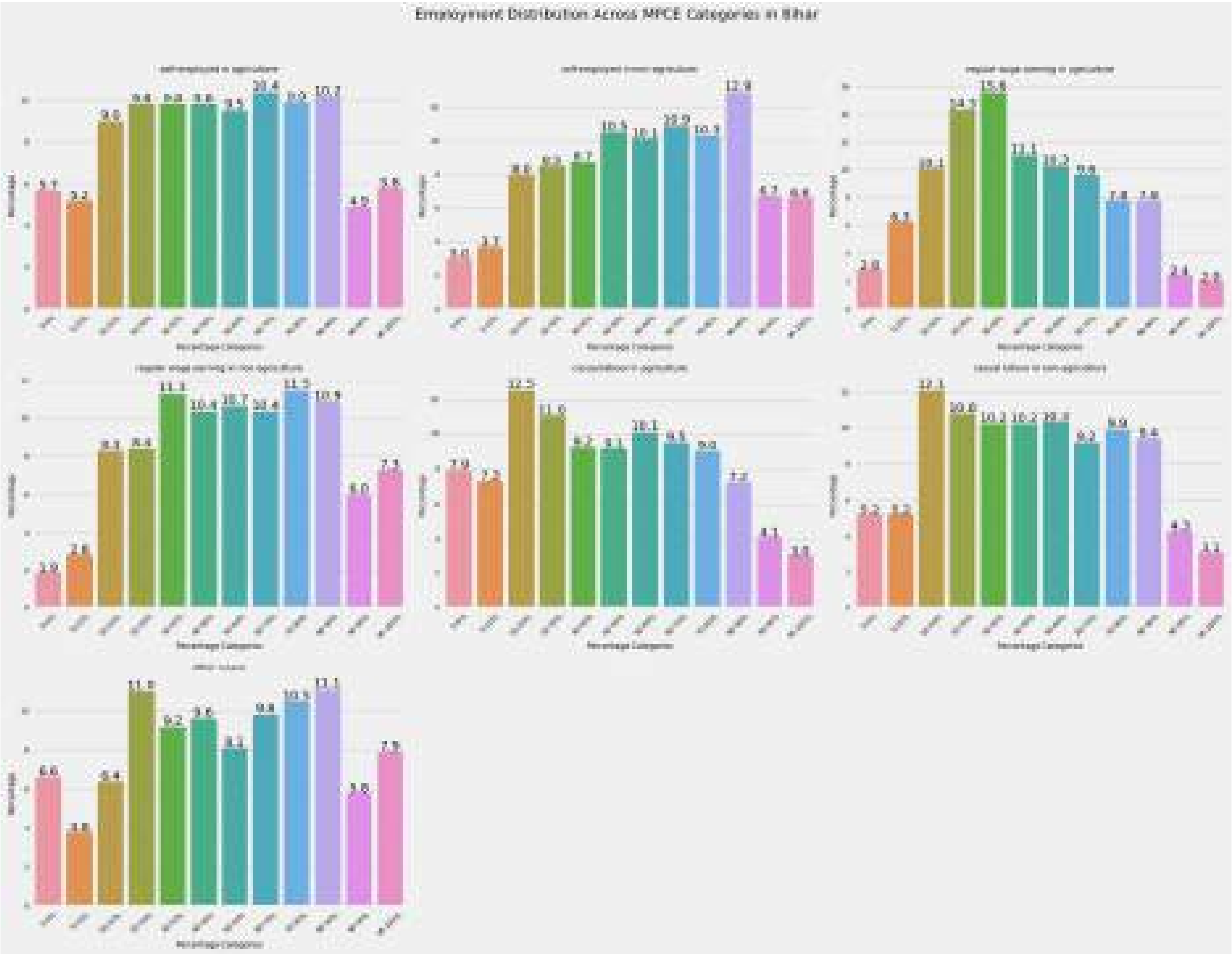
regular wage/salary earning in non-agriculture

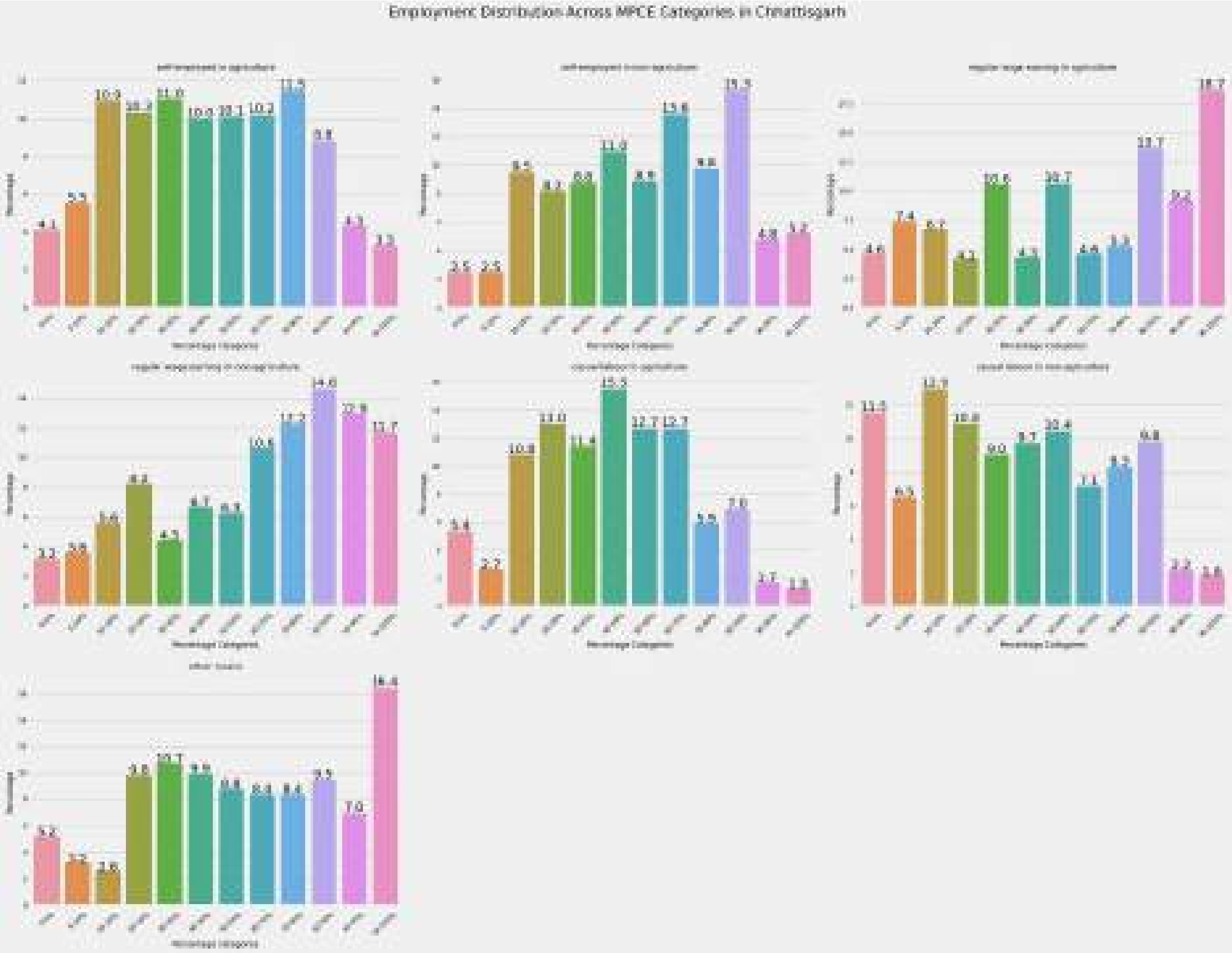
casual labour in agriculture

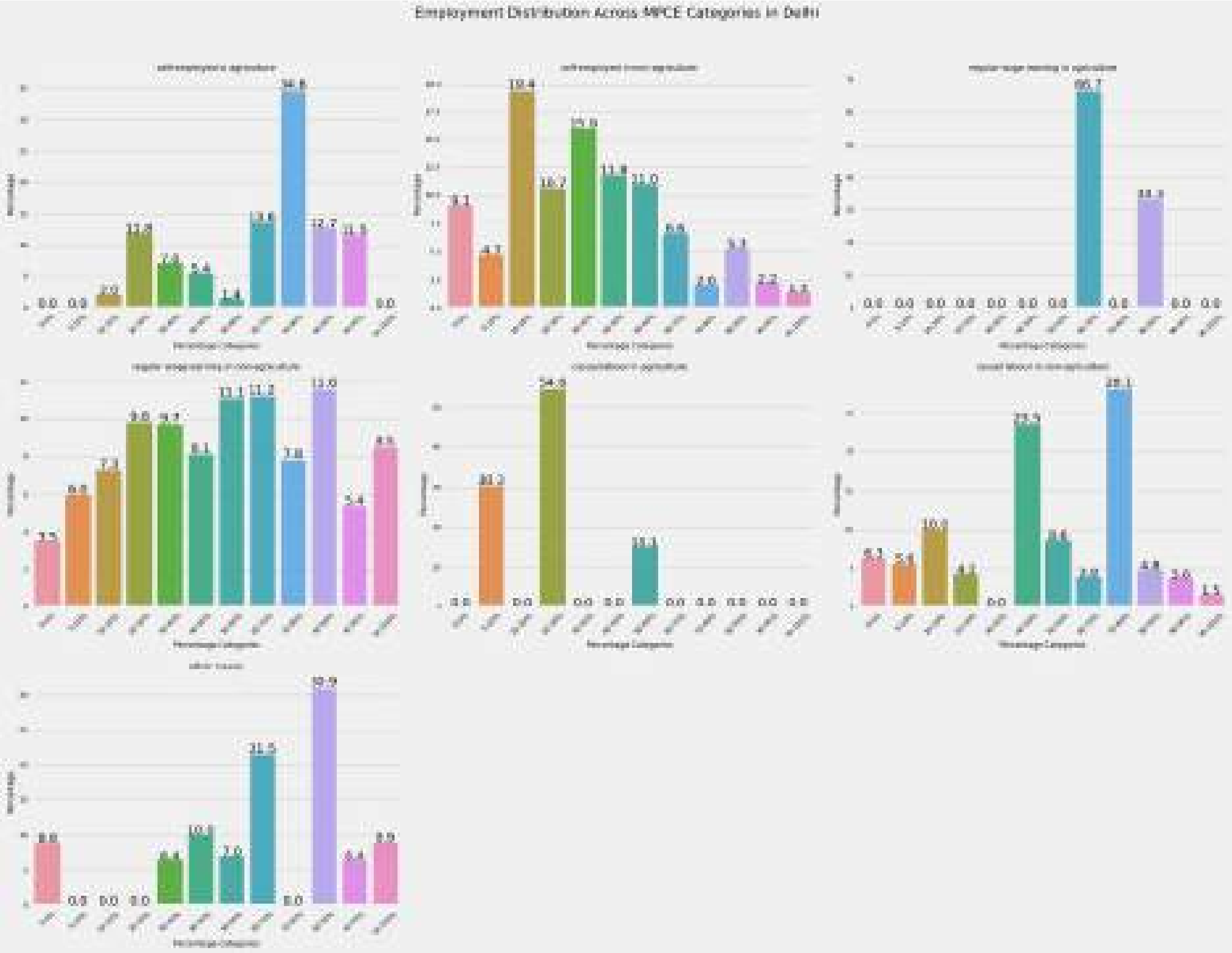
casual labour in non-agriculture

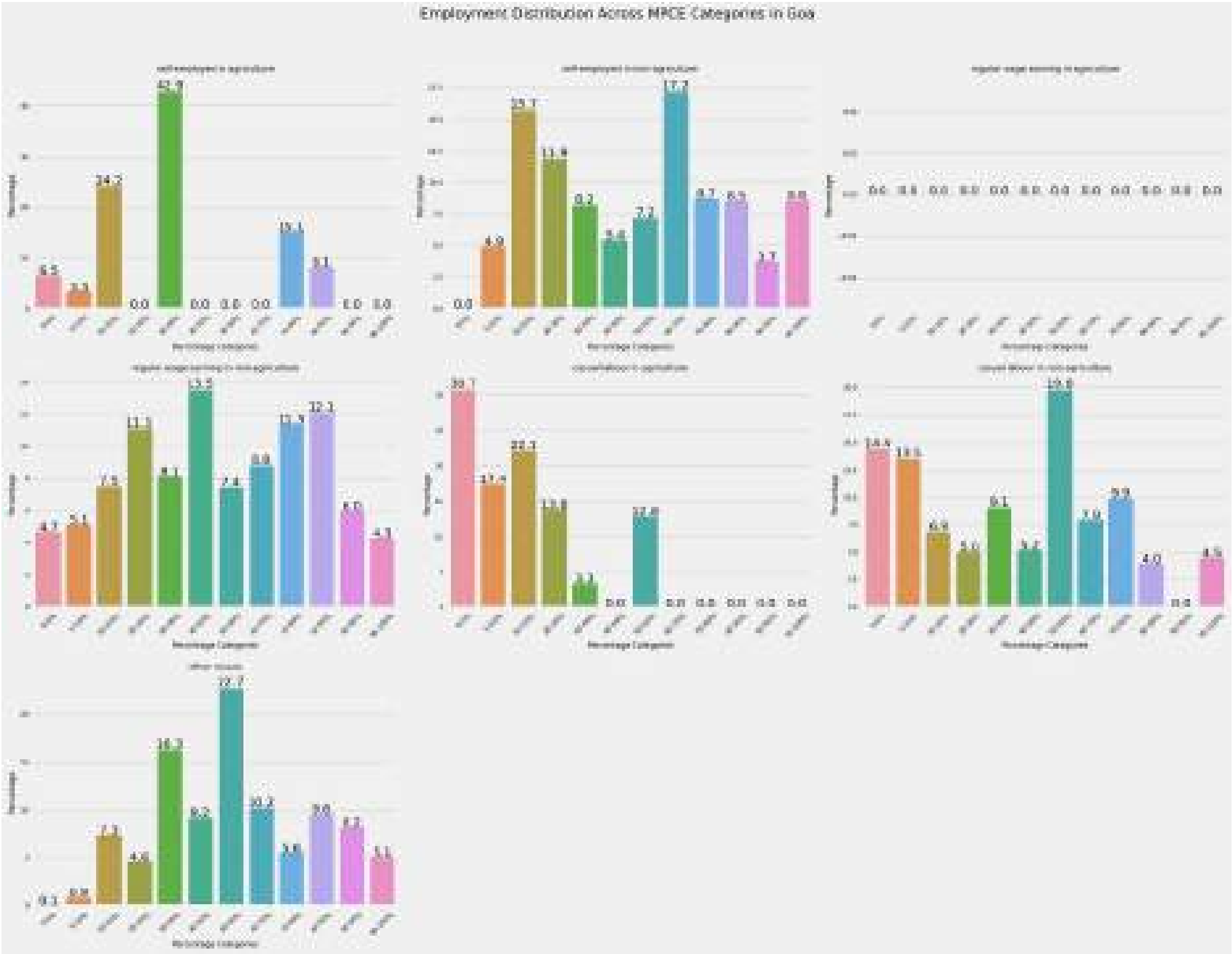
other sources

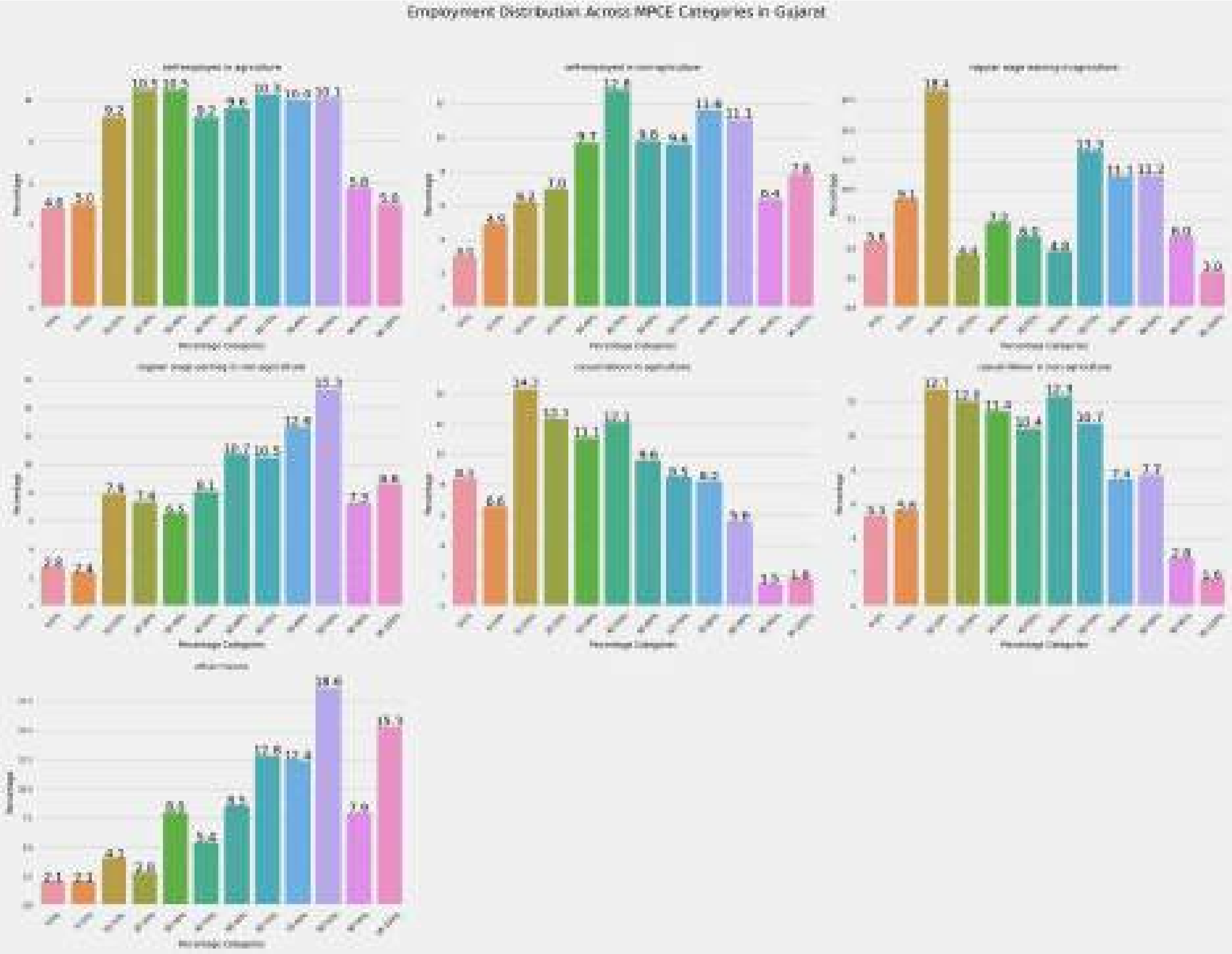

Employment Distribution Across MPCE Categories in Assam
self-employed in agriculture
3.9
4.2
10.5
11.3
10.6
10.4
11.1
9.6
10.6
8.7
4.7
3.4
self-employed in non-agriculture
3.3
3.2
8.8
8.2
9.4
10.1
9.8
11.6
10.9
12.5
5.9
5.5
regular wage earning in agriculture
16.1
16.1
12.1
12.9
11.1
6.3
6.5
4.8
3.0
3.3
4.2
3.5
regular wage earning in non-agriculture
2.3
3.9
6.6
5.8
8.4
7.8
8.0
7.6
11.6
15.8
8.8
13.2
casual labour in agriculture
9.2
8.5
13.9
14.0
10.1
11.3
8.7
8.3
7.0
4.6
2.5
0.8
casual labour in non-agriculture
5.2
4.4
10.6
9.6
10.8
10.6
11.1
12.0
9.6
9.7
3.7
2.7
others
2.8
4.5
3.0
6.0
7.4
8.6
8.7
8.6
12.5
12.2
8.0
18.4
Percentage
Percentage Categories

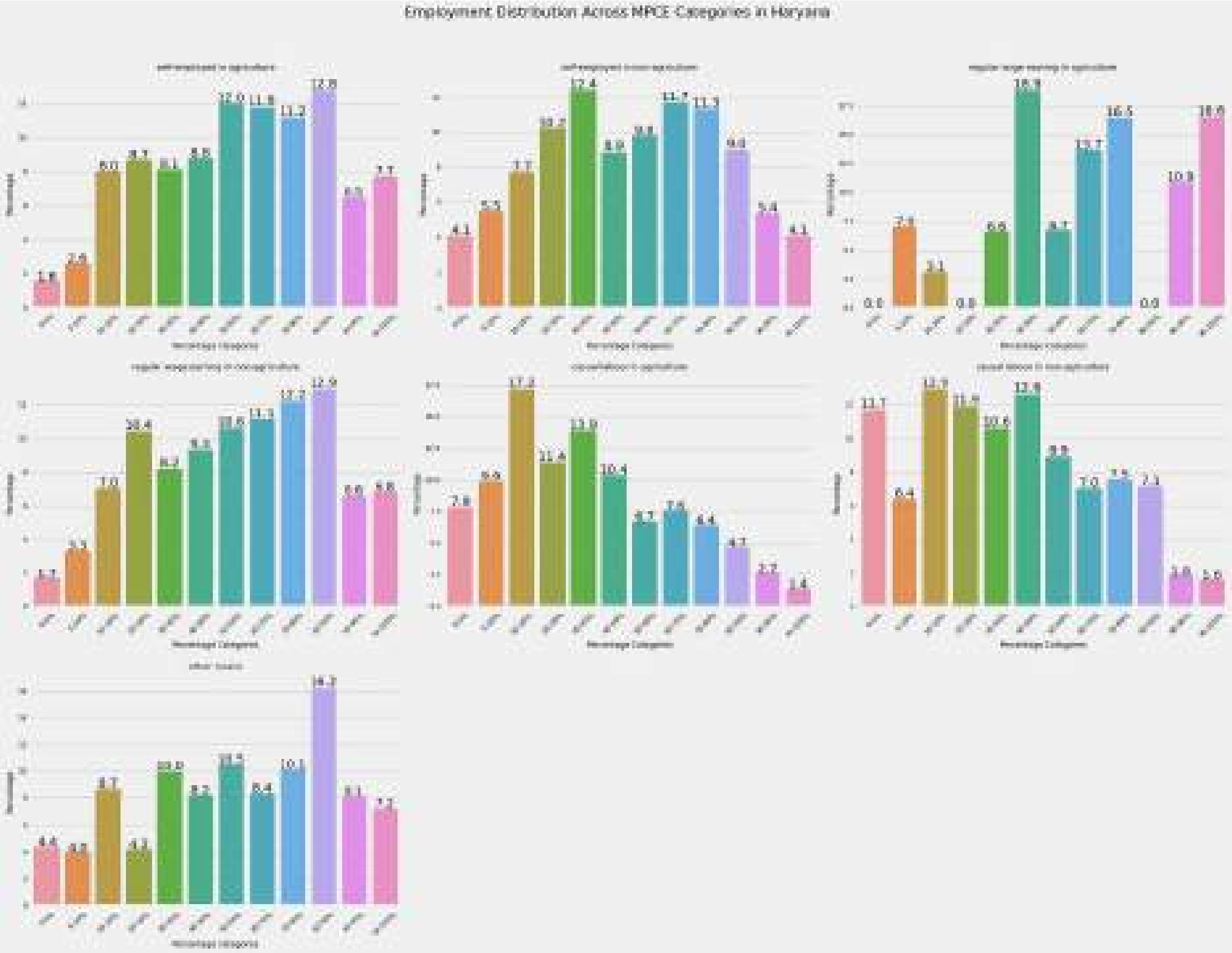

Employment Distribution Across MPCE Categories in Bihar
self-employed in agriculture
5.7
5.2
9.0
9.8
9.8
9.8
9.5
10.4
9.9
10.7
4.9
5.8
self-employed in non-agriculture
3.0
3.7
8.6
9.2
8.7
10.5
10.1
10.9
10.3
12.9
6.7
6.6
regular wage earning in agriculture
2.8
6.3
10.1
14.3
15.8
11.1
10.2
9.6
7.8
7.8
2.4
2.1
regular wage earning in non-agriculture
3.9
2.6
8.1
8.4
11.1
10.4
10.7
10.4
11.5
10.9
6.0
7.3
casual labour in agriculture
7.9
7.3
12.5
11.0
8.2
8.1
10.1
9.5
9.0
7.2
4.1
3.0
casual labour in non-agriculture
5.2
5.2
12.1
10.6
10.2
10.2
10.2
9.2
9.9
9.4
4.3
3.1
other income
6.6
3.8
6.4
11.0
9.2
9.6
8.1
9.8
10.5
11.1
3.8
7.9
Percentage
Percentage Categories


Employment Distribution Across MPCE Categories in Chhattisgarh
self-employed in agriculture
Percentage
4.1
5.5
10.9
10.2
11.0
10.0
10.1
10.2
11.9
8.8
4.3
3.3
Percentage Categories
self-employed in non-agriculture
Percentage
2.5
2.5
8.5
8.2
8.8
11.0
8.8
15.6
9.6
19.5
4.8
5.2
Percentage Categories
regular wage earning in agriculture
Percentage
4.6
7.4
6.7
4.1
10.6
4.3
10.7
4.6
5.3
13.7
9.2
18.7
Percentage Categories
regular wage/earning in non-agriculture
Percentage
3.2
3.8
5.6
8.2
4.5
6.7
6.3
10.6
12.2
14.6
12.9
11.7
Percentage Categories
casual labour in agriculture
Percentage
5.4
2.7
10.8
13.0
11.4
15.5
12.7
12.7
5.5
7.0
1.7
1.3
Percentage Categories
casual labour in non-agriculture
Percentage
11.5
6.5
12.9
10.8
9.0
9.7
10.4
7.1
8.5
9.8
2.2
1.8
Percentage Categories
other Source
Percentage
5.2
3.2
2.6
9.8
10.7
9.9
8.8
8.4
8.4
9.5
7.0
16.4
Percentage Categories

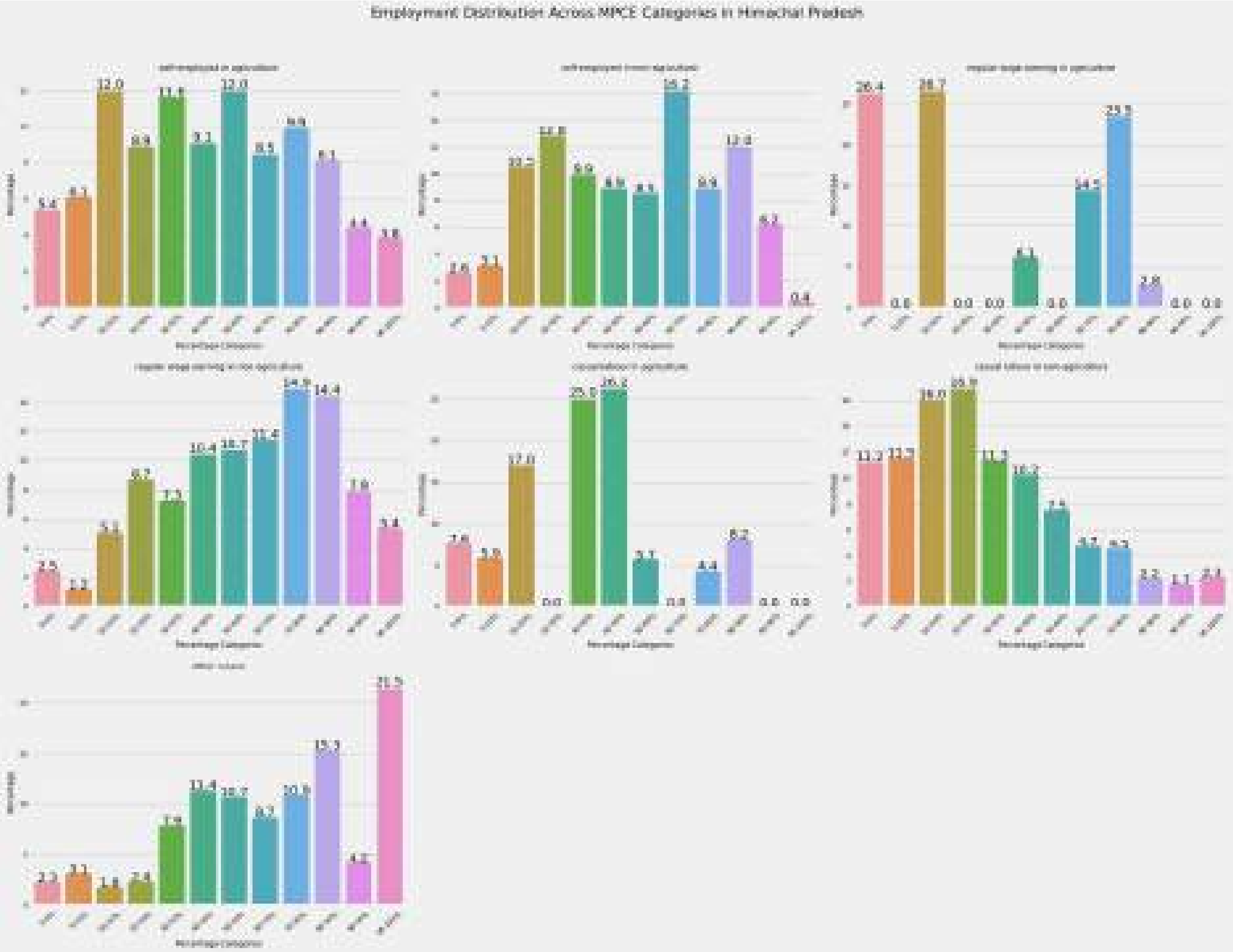

Employment Distribution Across MPCE Categories in Delhi
0.0
0.0
2.0
12.8
7.0
5.4
1.4
13.6
34.8
12.7
11.5
0.0
9.1
4.1
19.4
10.7
15.0
11.8
11.0
6.6
2.0
5.1
2.2
1.3
0.0
0.0
0.0
0.0
0.0
0.0
0.0
66.7
0.0
33.3
0.0
0.0
3.5
6.6
7.3
9.8
9.7
8.1
11.1
11.2
7.6
11.0
5.4
8.5
0.0
30.2
0.0
54.8
0.0
0.0
15.1
0.0
0.0
0.0
0.0
0.0
6.1
5.4
10.3
4.1
0.0
23.9
9.6
3.8
28.1
4.8
3.6
1.5
8.6
0.0
0.0
0.0
6.4
10.0
7.0
21.5
0.0
32.9
6.4
8.9

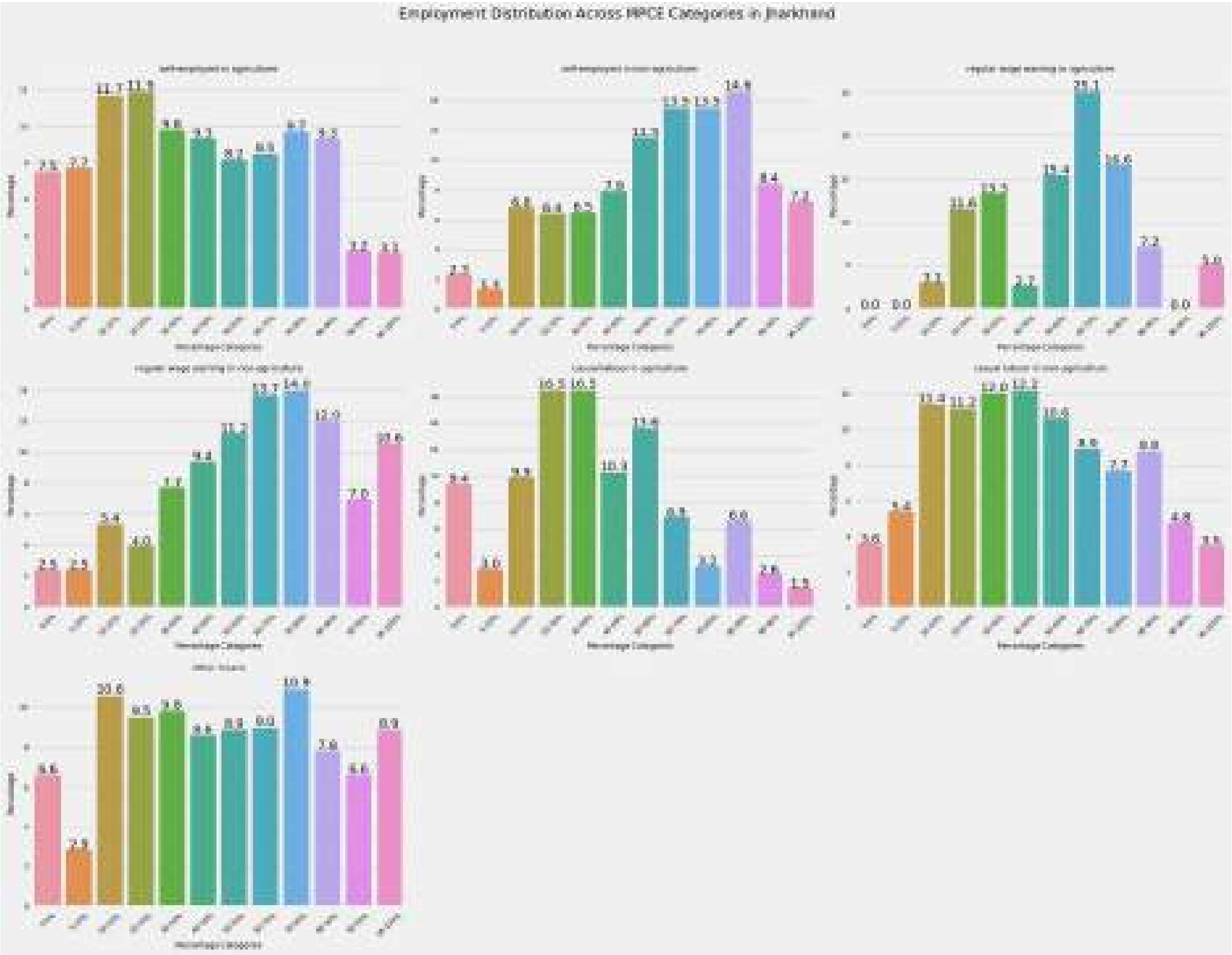

Employment Distribution Across MPCE Categories in Goa
6.5
3.3
24.2
0.0
42.9
0.0
0.0
0.0
15.1
8.1
0.0
0.0
0.0
4.9
15.7
11.8
8.2
5.6
7.2
17.7
8.7
8.5
3.7
8.6
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
4.7
5.1
7.5
11.1
8.1
13.5
7.4
8.8
11.5
12.1
6.0
4.3
38.7
17.4
22.1
13.8
3.1
0.0
17.4
0.0
0.0
0.0
0.0
0.0
14.4
13.5
6.8
5.0
9.1
5.2
19.8
7.9
9.9
4.0
0.0
4.5
0.1
0.8
7.3
4.6
16.3
9.2
22.7
10.2
5.6
9.6
8.2
5.1

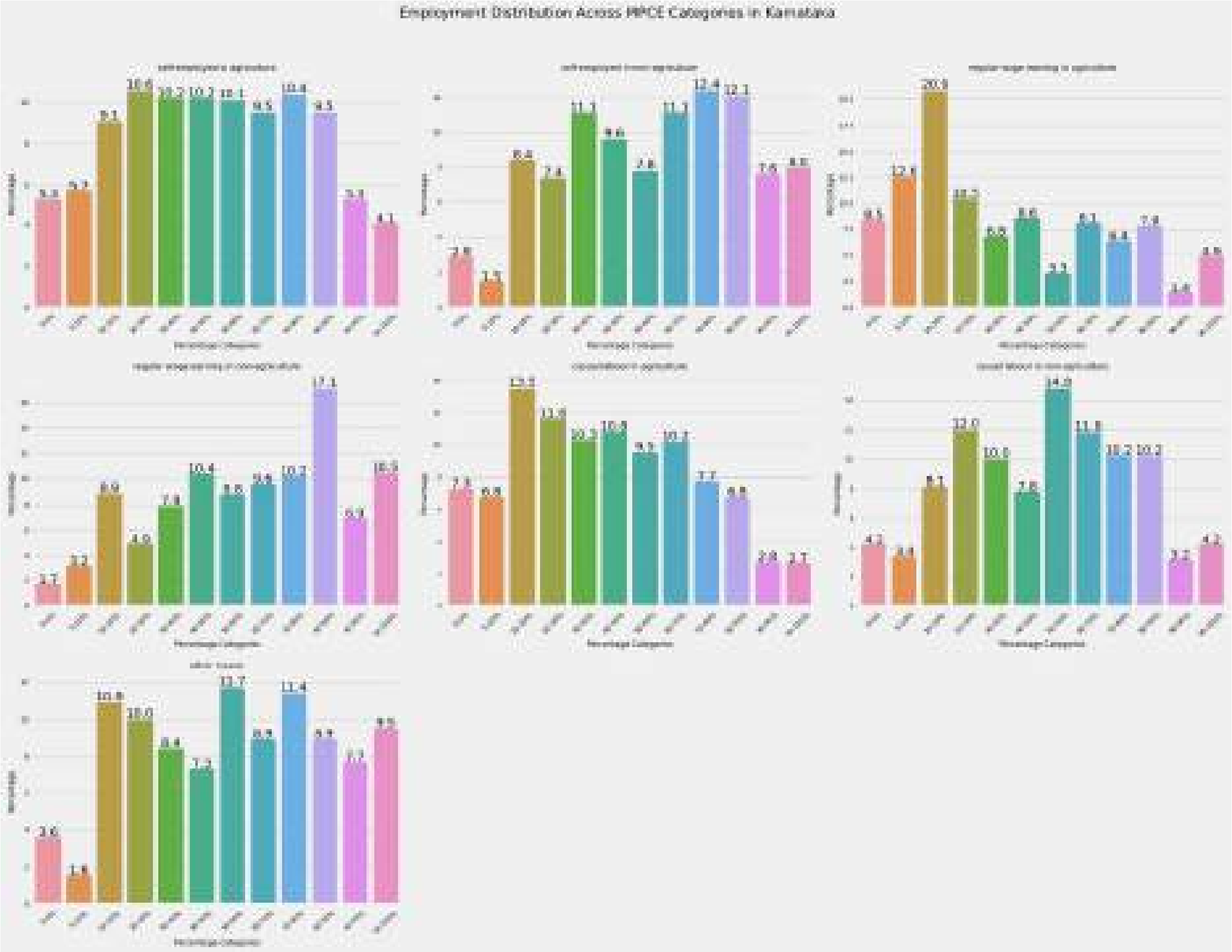

Employment Distribution Across MPCE Categories in Gujarat
self-employed in agriculture
4.8
5.0
9.2
10.5
10.5
8.2
8.6
10.3
10.0
10.1
5.8
5.0
self-employed in non-agriculture
2.5
4.5
6.1
7.0
9.7
12.8
9.8
9.6
11.6
11.1
6.4
7.8
regular wage earning in agriculture
5.8
9.1
18.4
4.4
7.3
6.5
4.8
13.7
11.1
11.2
6.0
3.0
regular wage earning in non-agriculture
2.6
2.4
7.8
7.4
6.5
8.1
10.7
10.5
12.8
15.3
7.3
8.6
casual labour in agriculture
8.3
6.6
14.2
12.3
11.1
12.1
9.6
8.5
8.2
5.6
1.5
1.8
casual labour in non-agriculture
5.3
5.6
12.1
12.0
11.6
10.4
12.9
10.7
7.4
7.7
2.8
1.6
others
2.1
2.1
4.2
2.6
9.1
5.4
8.5
12.8
12.4
18.6
7.9
15.3
Percentage
Percentage Categories

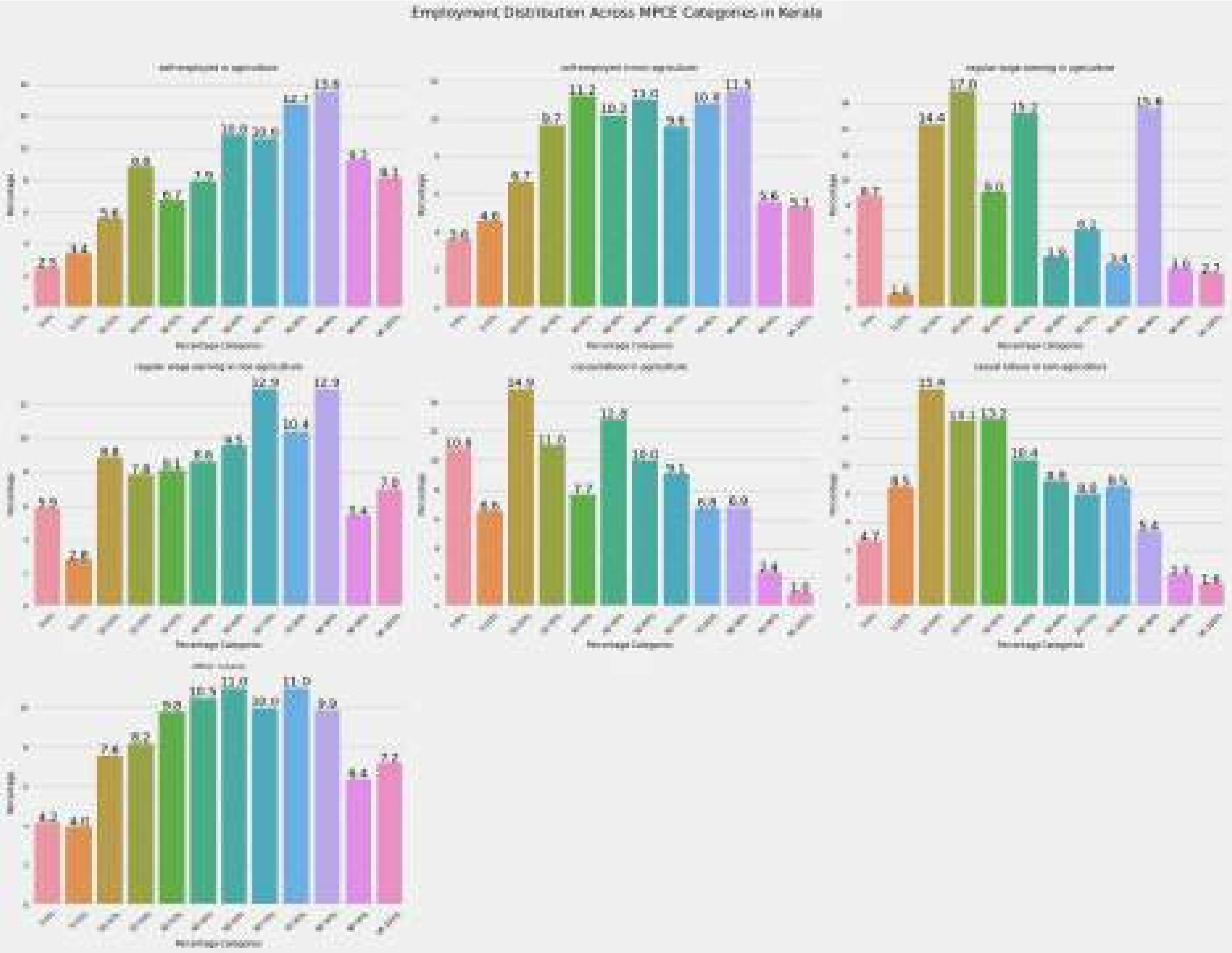


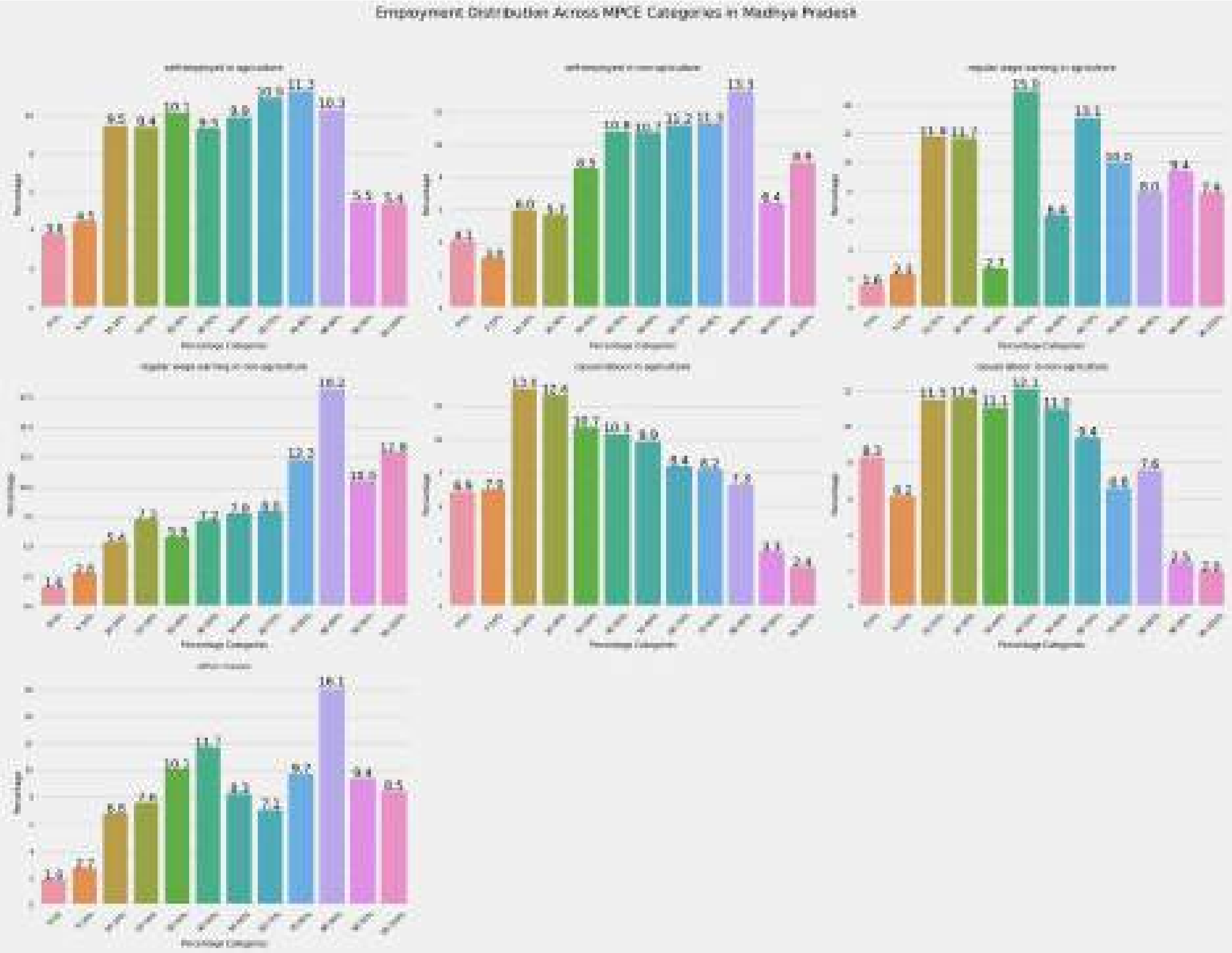
Employment Distribution Across MPCE Categories in Haryana

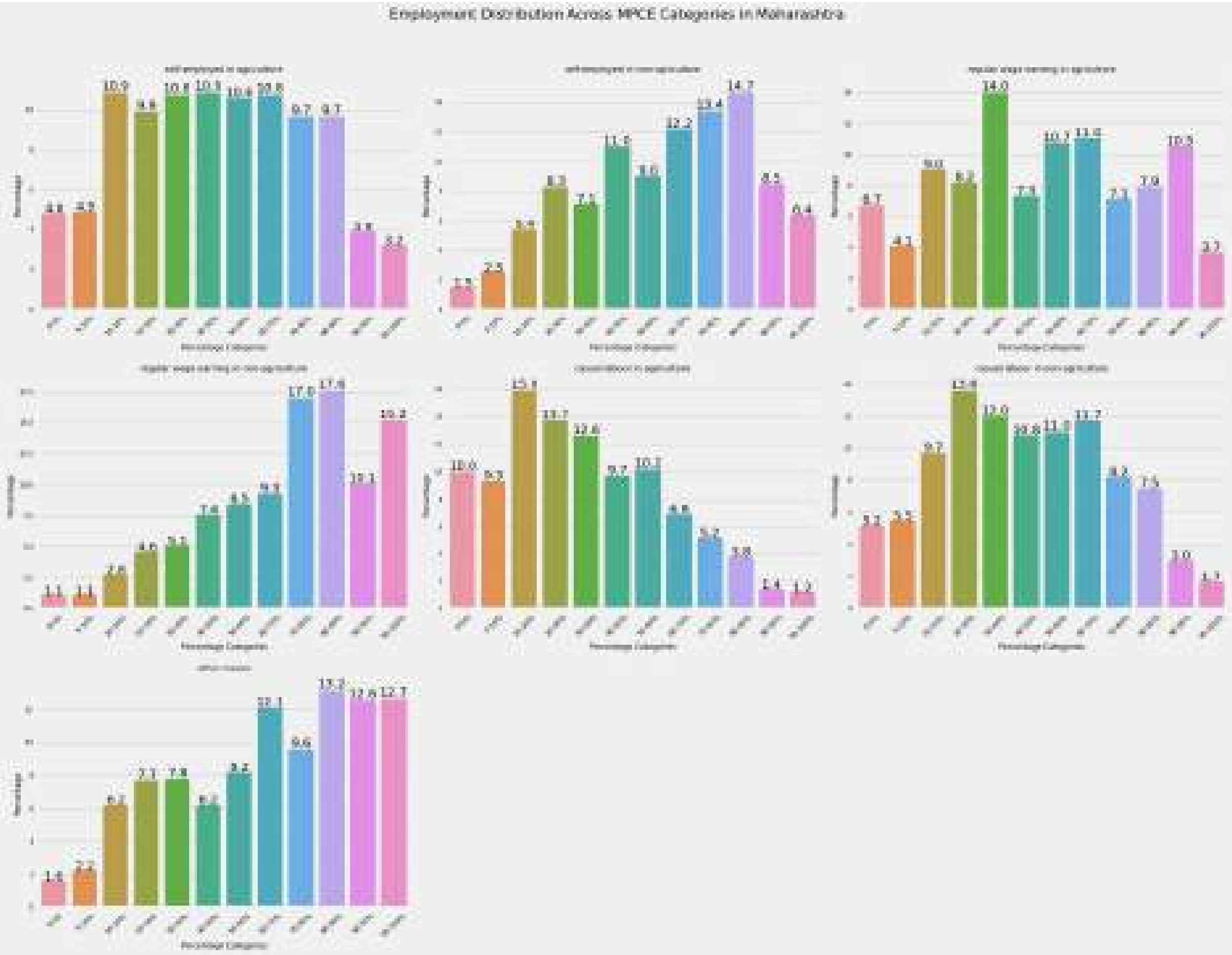
Employment Distribution Across MPCE Categories in Himachal Pradesh
self-employed in agriculture
Percentage
5.4
6.1
12.0
8.9
11.6
9.1
12.0
8.5
9.8
8.1
4.4
3.8
Percentage Categories
self-employed in non-agriculture
Percentage
2.6
3.1
10.5
12.8
9.9
8.9
8.5
16.2
8.9
12.0
6.2
0.4
Percentage Categories
regular wage earning in agriculture
Percentage
26.4
0.0
26.7
0.0
0.0
6.1
0.0
14.5
23.5
2.8
0.0
0.0
Percentage Categories
regular wage earning in non-agriculture
Percentage
2.5
1.1
5.1
8.7
7.3
10.4
10.7
11.4
14.9
14.4
7.8
5.4
Percentage Categories
casual labour in agriculture
Percentage
7.8
5.9
17.0
0.0
25.0
26.2
5.7
0.0
4.4
8.2
0.0
0.0
Percentage Categories
casual labour in non-agriculture
Percentage
11.2
11.5
16.0
16.9
11.2
10.2
7.5
4.7
4.5
2.2
1.7
2.1
Percentage Categories
Percentage
2.3
3.1
1.6
2.4
7.9
11.4
10.7
8.7
10.8
15.3
4.2
21.5
Percentage Categories

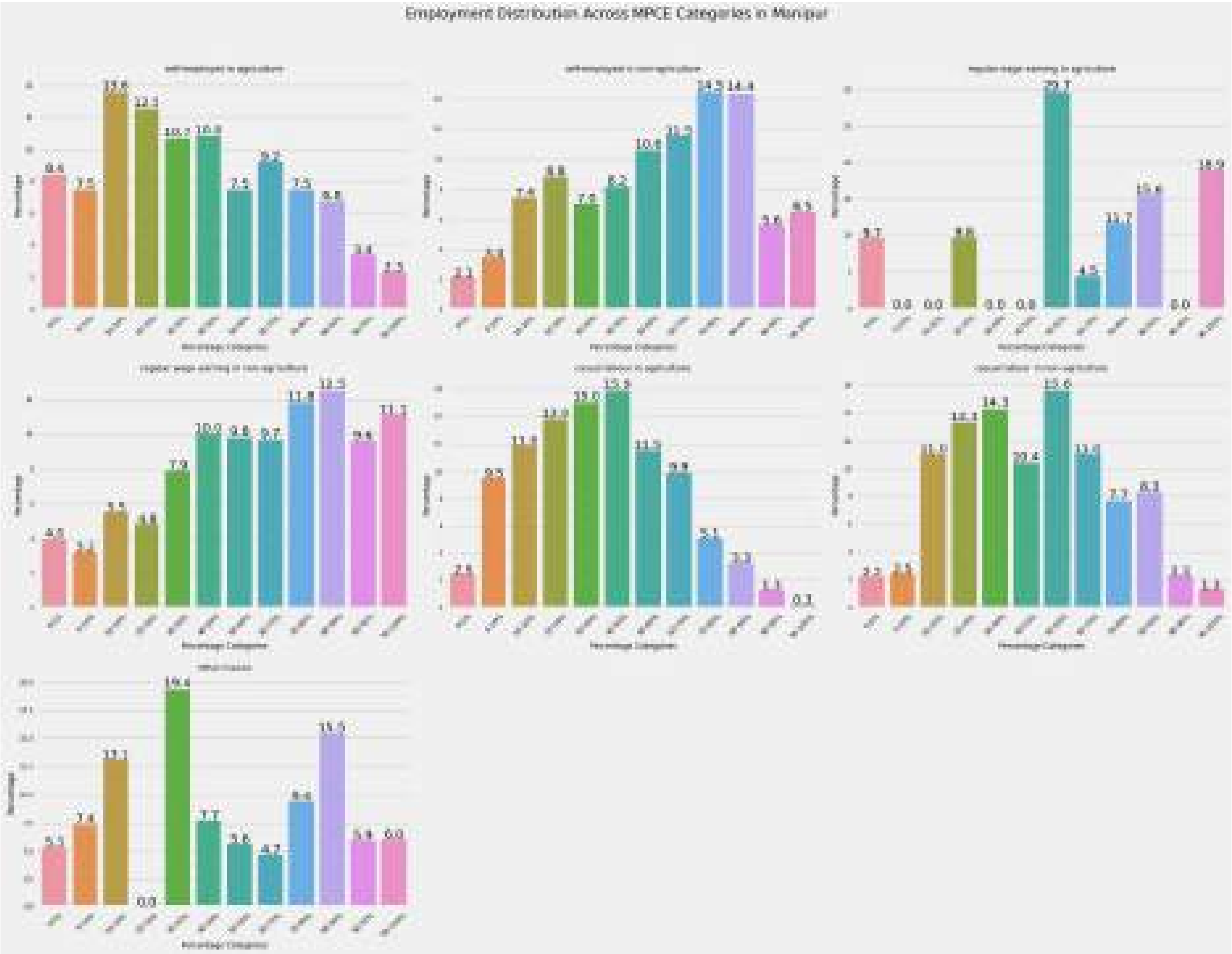

Employment Distribution Across MPCE Categories in Jharkhand
self-employed in agriculture
7.5
7.7
11.7
11.9
9.6
9.1
8.1
8.5
9.7
9.1
3.2
3.1
self-employed in non-agriculture
2.3
1.4
6.8
6.4
6.5
7.9
11.3
13.5
13.5
14.6
8.4
7.2
regular wage earning in agriculture
0.0
0.0
3.1
11.6
13.3
2.2
15.4
25.1
16.6
7.2
0.0
5.0
regular wage earning in non-agriculture
2.5
2.5
5.4
4.0
7.7
9.4
11.2
13.7
14.0
12.2
7.0
10.6
casual labour in agriculture
9.4
3.0
9.9
16.5
16.5
10.3
13.6
6.9
3.2
6.6
2.6
1.5
casual labour in non-agriculture
3.6
5.4
11.8
11.2
12.0
12.2
10.6
8.6
7.7
8.8
4.8
3.6
others
6.6
2.8
10.6
9.5
9.8
8.6
8.8
9.0
10.9
7.6
6.6
8.9
Percentage
Percentage Categories

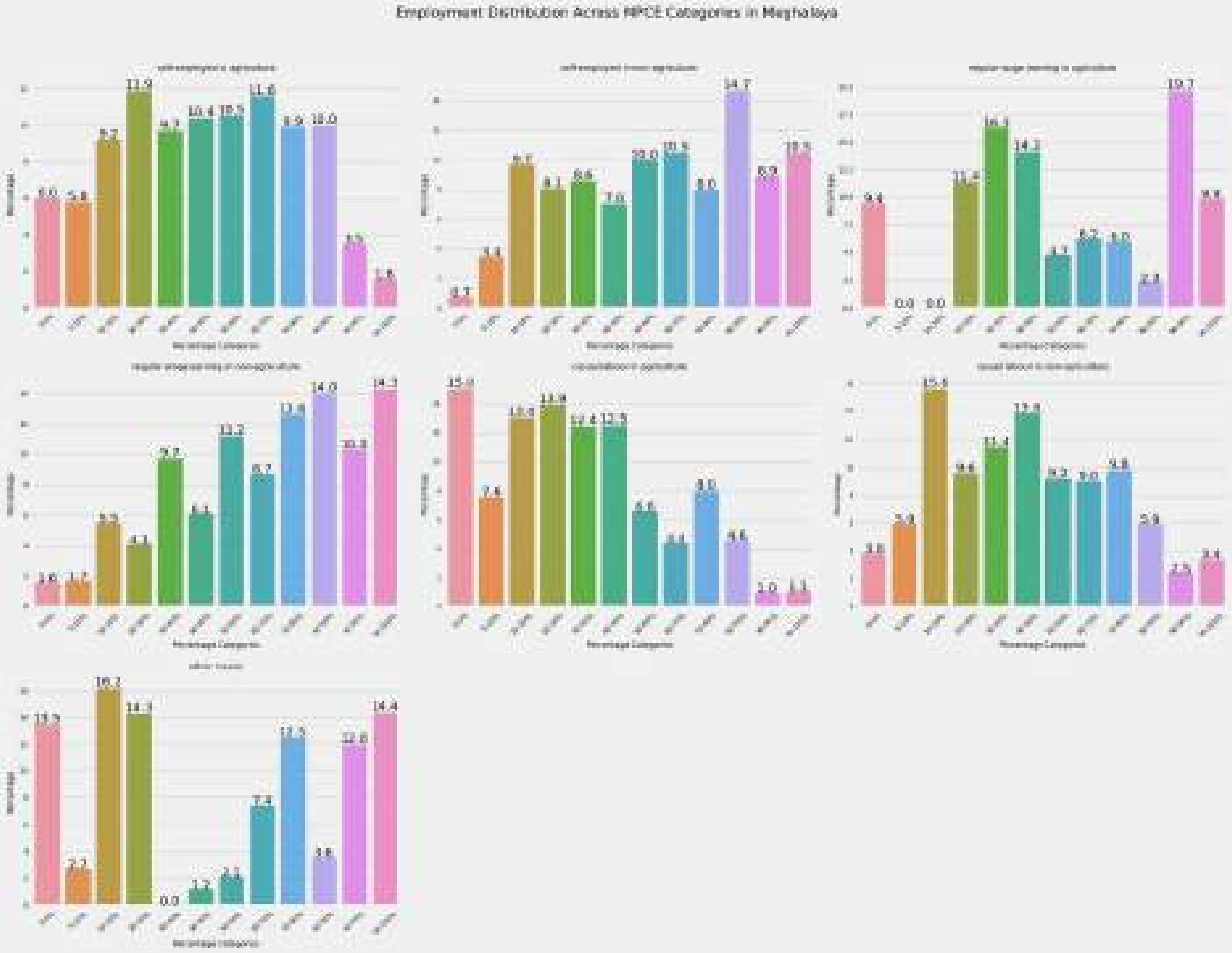

Employment Distribution Across MPCE Categories in Karnataka
self-employed in agriculture
5.5
5.7
9.1
10.6
10.2
10.2
10.1
9.5
10.4
9.5
5.3
4.1
self-employed in non-agriculture
2.8
1.5
8.4
7.4
11.1
9.6
7.8
11.1
12.4
12.1
7.6
8.0
regular wage earning in agriculture
8.5
12.4
20.9
10.2
6.8
8.6
3.1
8.1
6.4
7.8
1.6
4.9
regular wage earning in non-agriculture
1.7
3.2
8.9
4.0
7.8
10.4
8.6
9.6
10.2
17.1
6.9
10.5
casual labour in agriculture
7.1
6.8
13.3
11.6
10.3
10.8
9.5
10.7
7.7
6.8
2.8
2.7
casual labour in non-agriculture
4.2
3.4
8.1
12.0
10.0
7.8
14.0
11.8
10.2
10.2
3.2
4.2
others
3.6
1.4
10.8
10.0
8.4
7.2
12.7
8.9
11.4
8.9
7.7
9.5
Percentage
Percentage Categories

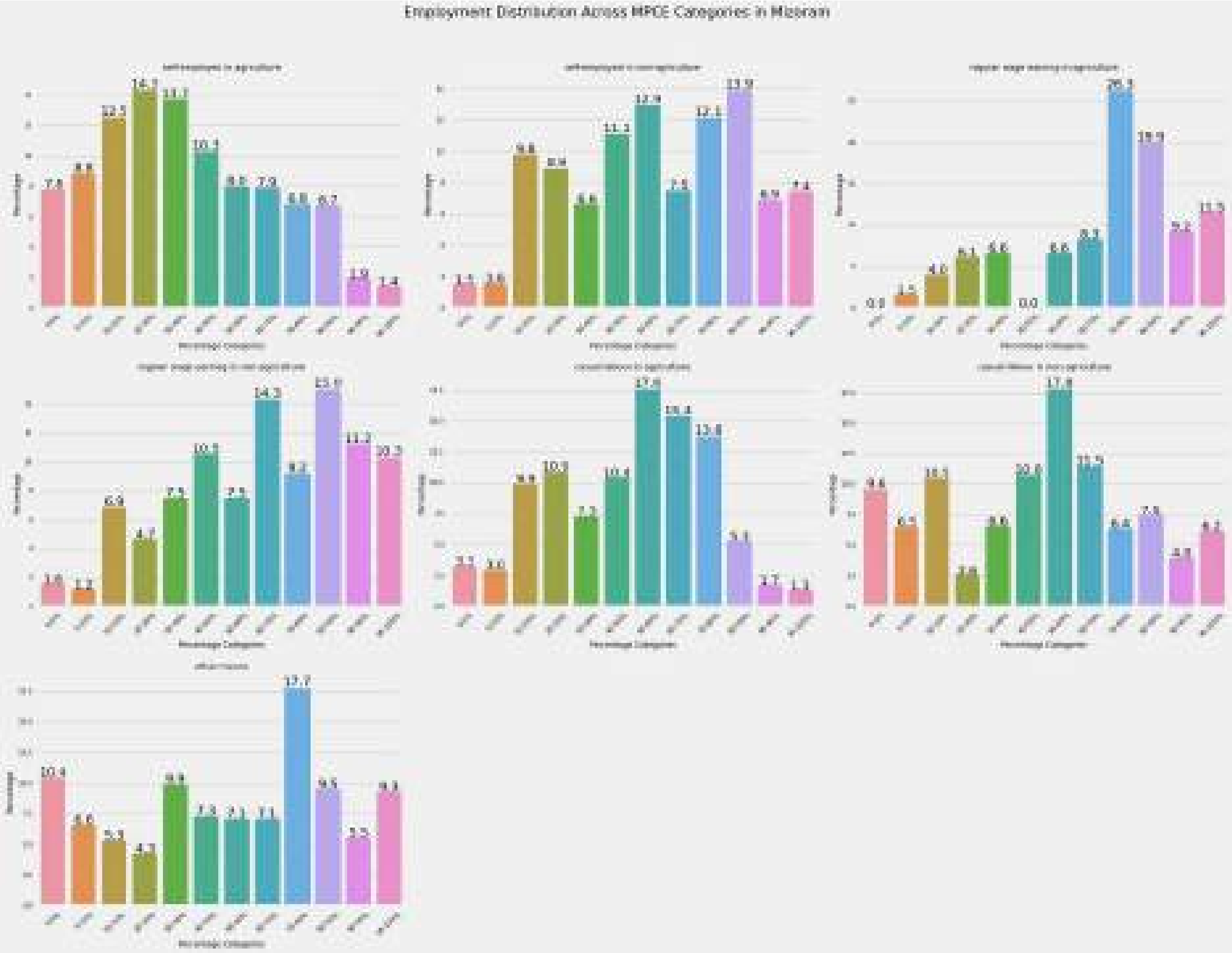

Employment Distribution Across MPCE Categories in Kerala
Percentage
Percentage Categories
2.5
3.4
5.6
8.8
6.7
7.9
10.8
10.6
12.7
13.8
9.3
8.1
3.6
4.6
6.7
9.7
11.2
10.2
11.0
9.6
10.8
11.5
5.6
5.3
8.7
1.6
14.4
17.0
9.0
15.2
3.9
6.3
3.4
15.8
3.0
2.7
5.9
2.8
8.8
7.8
8.1
8.6
9.5
12.9
10.4
12.9
5.4
7.0
10.8
6.6
14.9
11.0
7.7
11.8
10.0
9.1
6.8
6.9
2.4
1.0
4.7
8.5
15.6
13.1
13.2
10.4
8.8
8.0
8.5
5.4
2.3
1.6
4.2
4.0
7.6
8.2
9.9
10.5
11.0
10.0
11.0
9.9
6.4
7.7

Employment Distribution Across MPCE Categories in Madhya Pradesh
3.8
4.5
9.5
9.4
10.1
9.5
9.9
10.9
11.2
10.3
5.5
5.4
4.1
3.3
6.0
5.7
8.5
10.8
10.7
11.2
11.3
13.1
6.4
8.8
1.6
2.1
11.6
11.7
2.7
15.1
6.4
13.1
10.0
8.0
9.4
7.6
1.6
2.6
5.4
7.1
5.8
7.2
7.8
8.0
12.3
18.2
10.3
12.8
6.6
7.0
13.1
12.4
10.7
10.1
9.9
8.4
8.2
7.3
3.3
2.4
8.3
6.2
11.5
11.6
11.1
12.1
11.0
9.4
6.6
7.6
2.5
2.0
1.9
2.7
6.8
7.6
10.1
11.7
8.3
7.1
8.7
16.1
9.4
8.5
Percentage
Percentage Categories

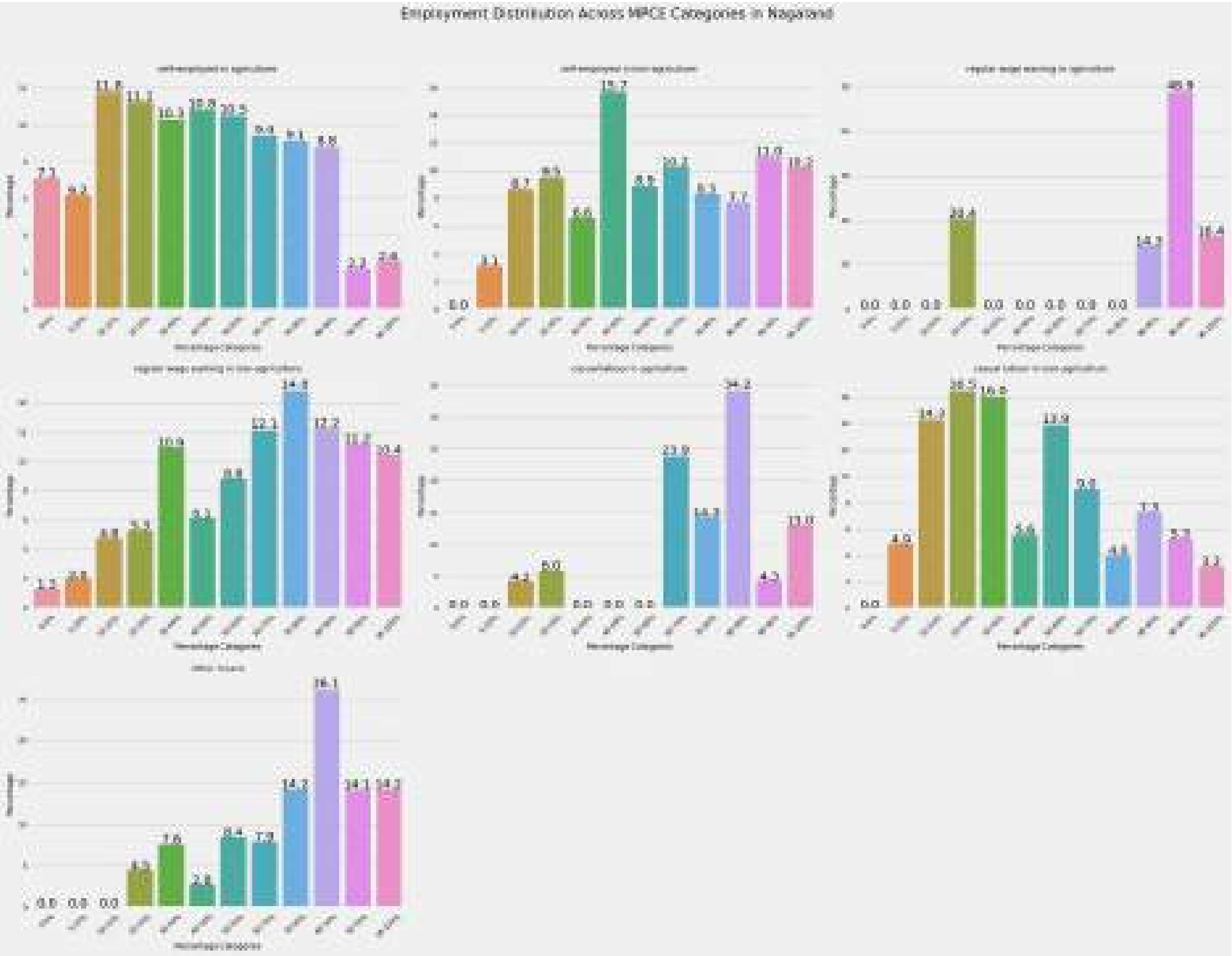

Employment Distribution Across MPCE Categories in Maharashtra
self employed in agriculture
4.8
4.9
10.9
9.8
10.8
10.9
10.6
10.8
9.7
9.7
3.8
3.2
Percentage
Percentage Categories
self employed in non agriculture
1.5
2.5
5.9
8.3
7.1
11.6
9.6
12.2
13.4
14.7
8.5
6.4
Percentage
Percentage Categories
regular wage earning in agriculture
8.7
4.1
9.0
8.2
14.0
7.3
10.7
11.0
7.1
7.9
10.5
3.7
Percentage
Percentage Categories
regular wage earning in non agriculture
1.1
1.1
2.6
4.6
5.1
7.6
8.5
9.3
17.0
17.6
10.1
15.2
Percentage
Percentage Categories
casual labour in agriculture
10.0
9.3
15.1
13.7
12.6
9.7
10.2
6.8
5.2
3.8
1.4
1.2
Percentage
Percentage Categories
casual labour in non agriculture
5.2
5.5
9.7
12.8
12.0
10.8
11.0
11.7
8.2
7.5
3.0
1.7
Percentage
Percentage Categories
other reason
1.6
2.2
6.2
7.7
7.8
6.2
8.2
12.1
9.6
13.2
12.6
12.7
Percentage
Percentage Categories

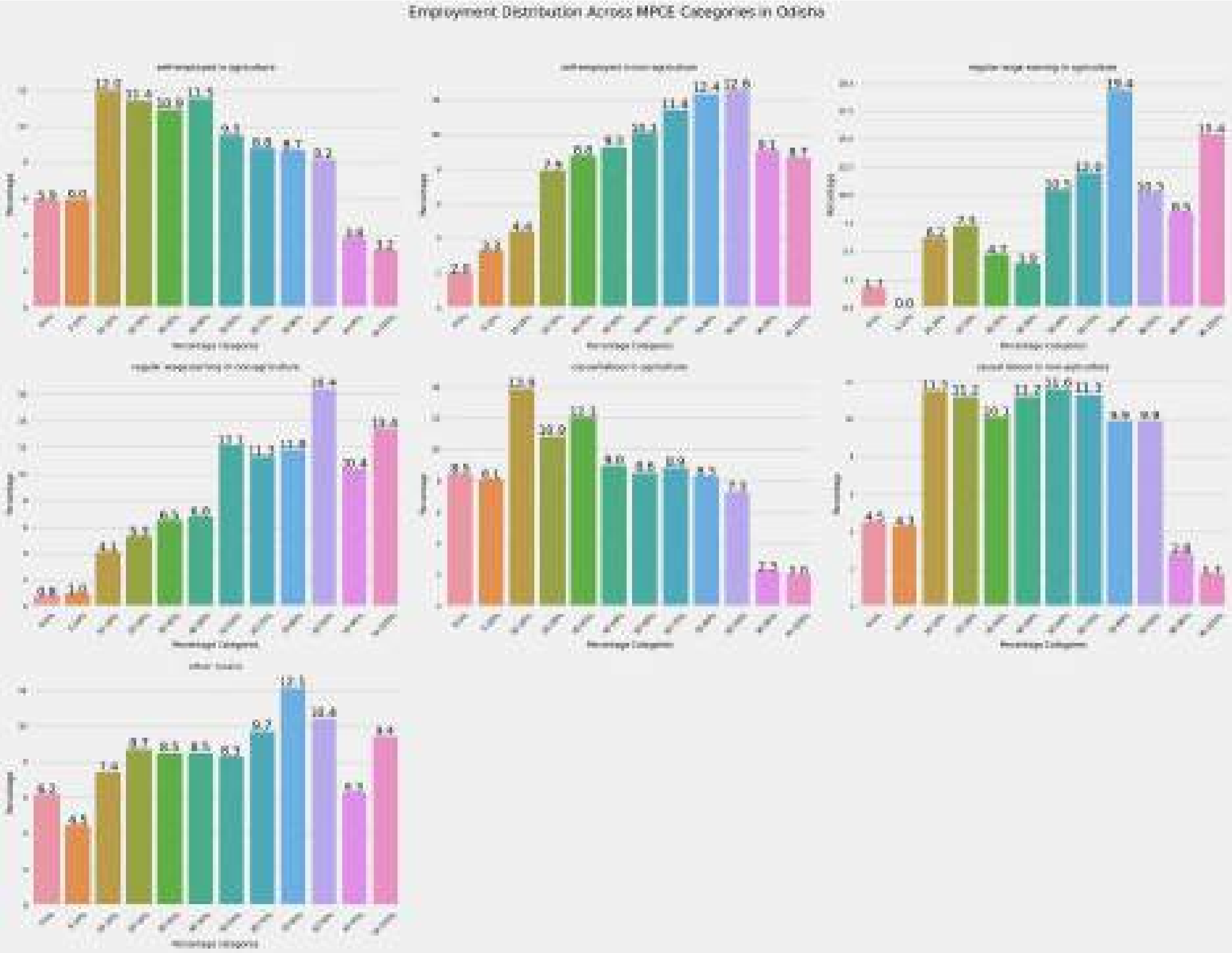

Employment Distribution Across MPCE Categories in Manipur
self-employed in agriculture
8.4
7.5
13.6
12.3
10.7
10.8
7.4
9.2
7.5
6.8
3.4
2.3
self-employed in non-agriculture
2.1
3.3
7.4
8.8
7.0
8.2
10.8
11.7
14.5
14.4
5.6
6.5
regular wage earning in agriculture
8.7
0.0
0.0
8.6
0.0
0.0
29.7
4.5
11.7
15.6
0.0
18.9
regular wage earning in non-agriculture
4.0
3.3
5.5
4.8
7.9
10.0
9.8
9.7
11.8
12.5
9.6
11.2
casual labour in agriculture
2.6
9.5
11.8
13.0
15.0
15.5
11.3
9.8
5.1
3.3
1.3
0.1
casual labour in non-agriculture
1.2
1.4
11.0
13.1
14.3
10.4
15.6
11.0
7.7
8.3
1.3
1.1
others
5.1
7.4
13.1
0.0
19.4
7.7
5.6
4.7
9.6
15.5
5.9
6.0
Percentage
Percentage Categories

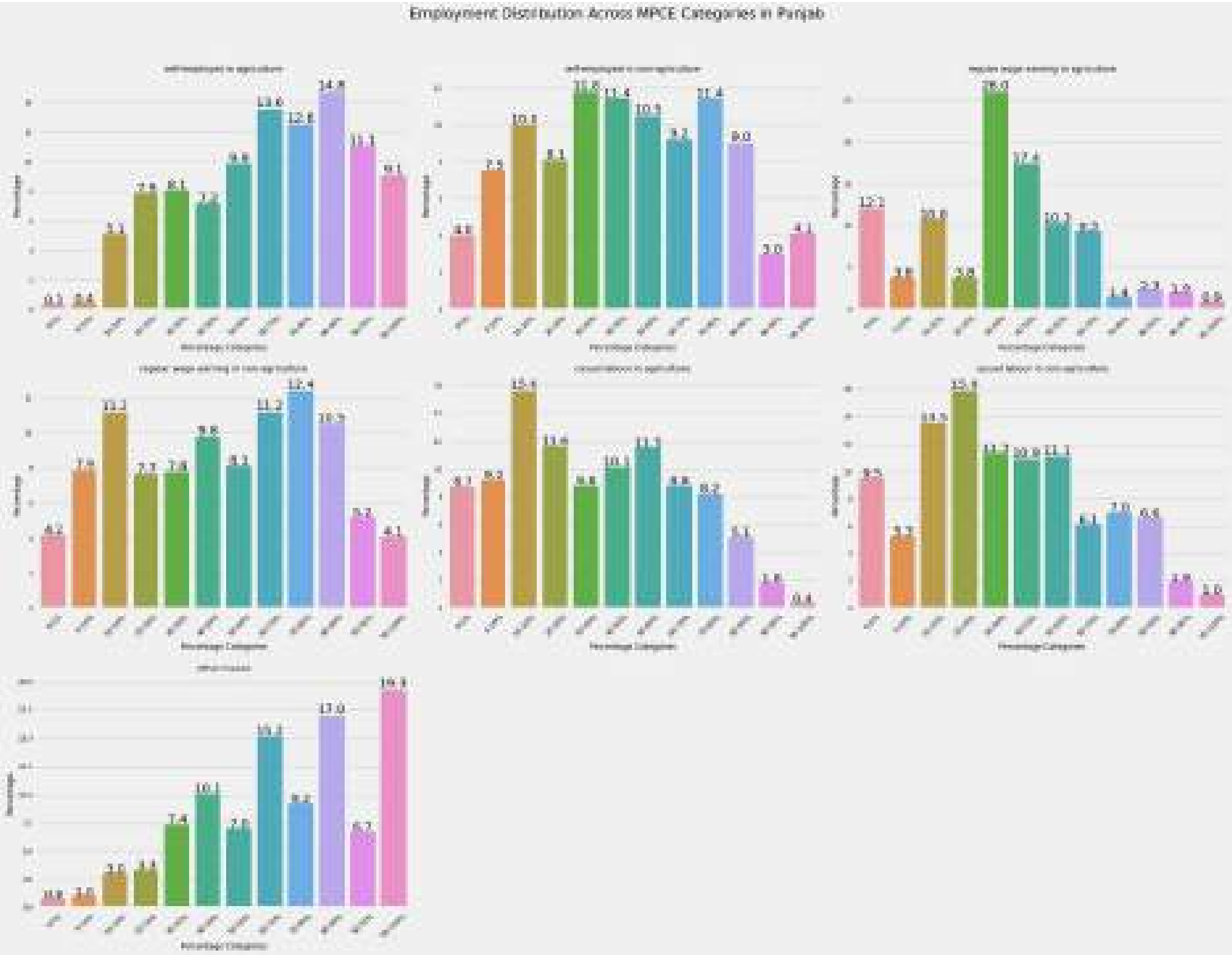

Employment Distribution Across MPCE Categories in Meghalaya
self-employed in agriculture
6.0
5.8
9.2
11.9
9.3
10.4
10.5
11.6
8.9
10.0
3.5
1.8
Percentage
Percentage Categories
self-employed in non-agriculture
0.7
3.4
9.7
8.1
8.6
7.0
10.0
10.5
8.0
14.7
8.9
10.5
Percentage
Percentage Categories
regular wage earning in agriculture
9.4
0.0
0.0
11.4
16.3
14.2
4.7
6.2
6.0
2.3
19.7
9.8
Percentage
Percentage Categories
regular wage earning in non-agriculture
1.6
1.7
5.5
4.1
9.7
6.1
11.2
8.7
12.6
14.0
10.3
14.3
Percentage
Percentage Categories
casual labour in agriculture
15.0
7.6
11.0
11.9
12.4
12.5
6.6
4.4
8.0
4.6
1.0
1.1
Percentage
Percentage Categories
casual labour in non-agriculture
3.8
5.9
15.6
9.6
11.4
13.8
9.2
9.0
9.8
5.8
2.5
3.4
Percentage
Percentage Categories
other households
13.5
2.2
16.2
14.3
0.0
1.2
2.1
7.4
12.5
3.8
12.0
14.4
Percentage
Percentage Categories

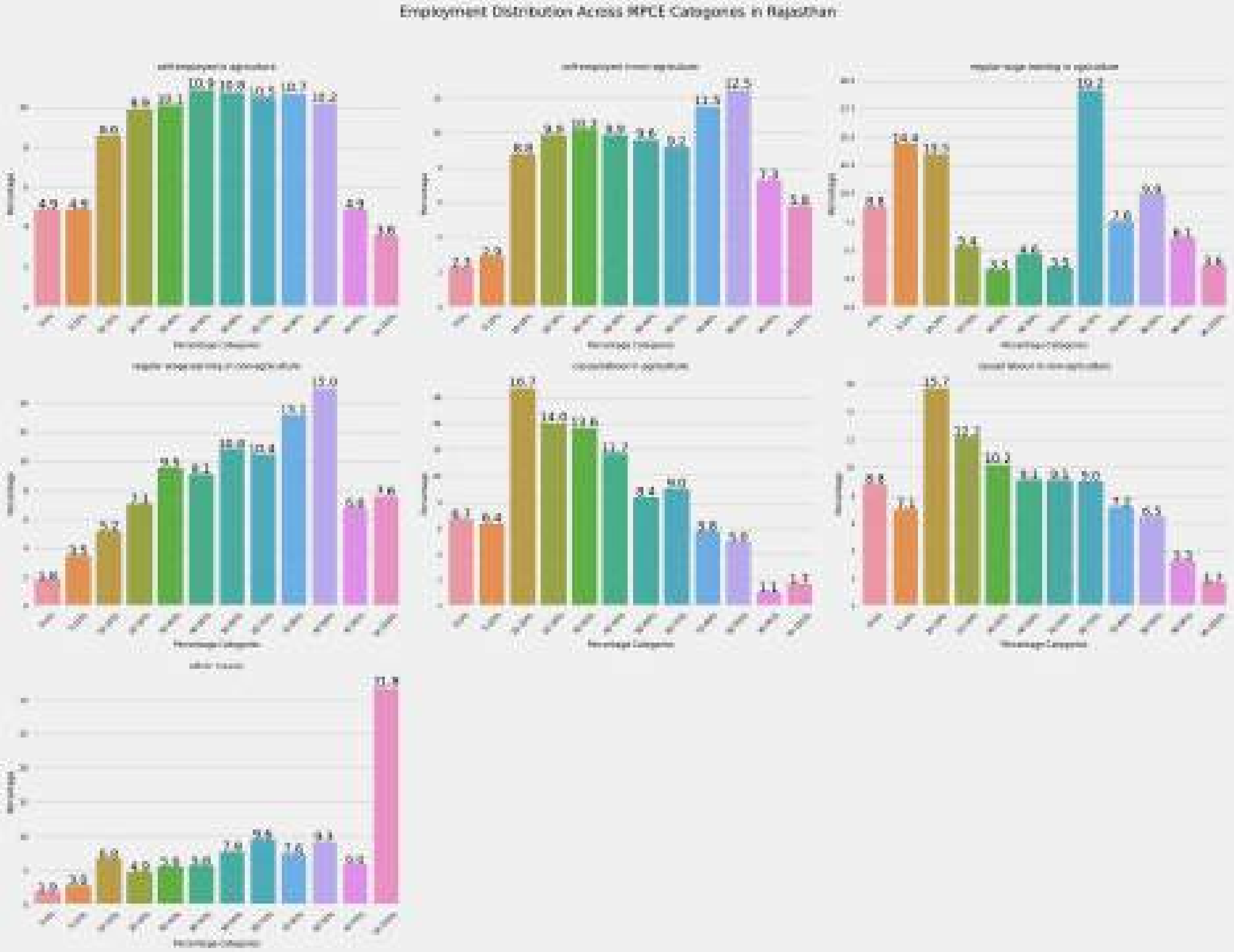


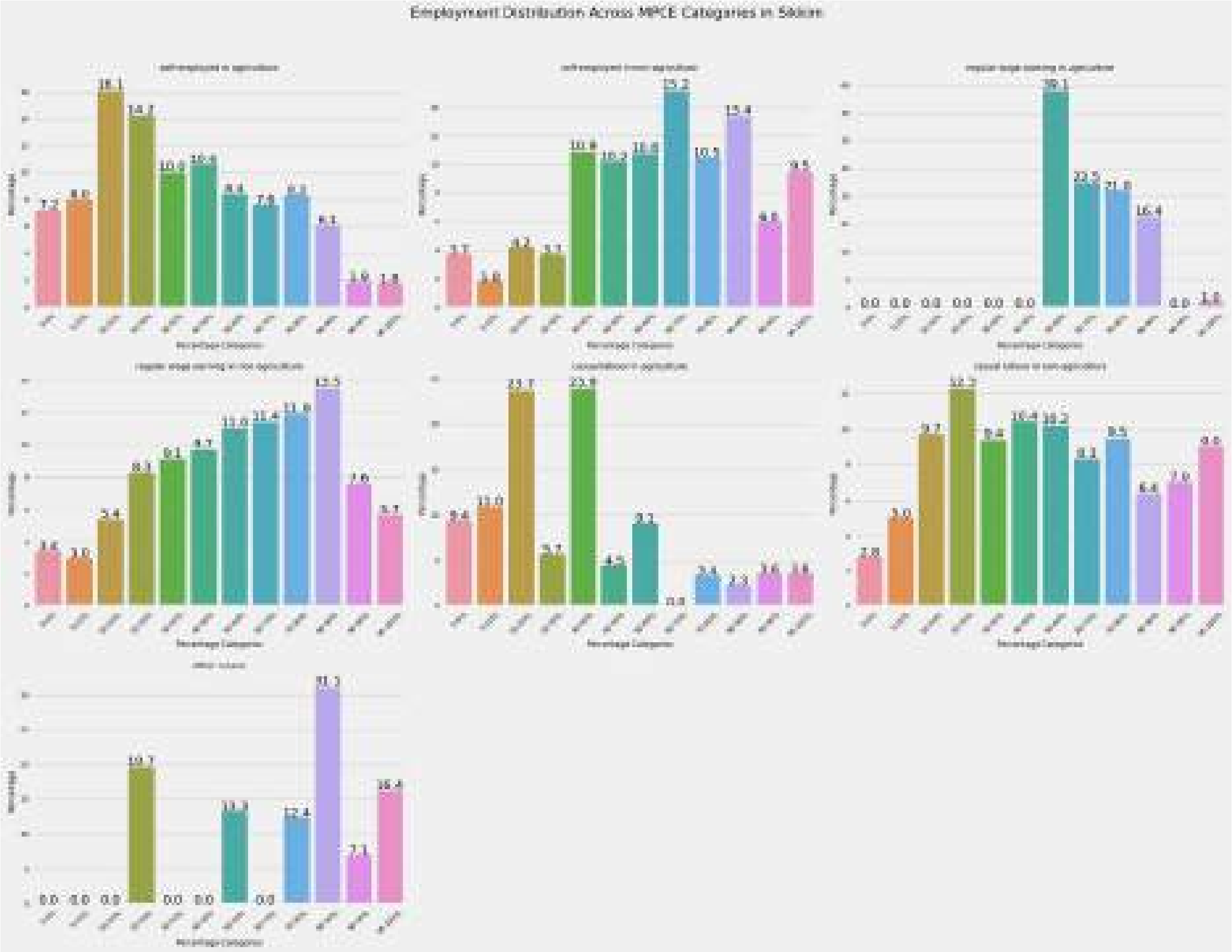
Employment Distribution Across MPCE Categories in Mizoram


Employment Distribution Across MPCE Categories in Nagaland
7.1
6.2
11.8
11.1
10.3
10.8
10.5
9.4
9.1
8.8
2.2
2.6
0.0
3.1
8.7
9.5
6.6
15.7
8.5
10.2
8.1
7.7
11.0
10.2
0.0
0.0
0.0
20.4
0.0
0.0
0.0
0.0
0.0
14.3
49.9
16.4
1.3
2.0
4.8
5.3
10.9
5.1
8.8
12.1
14.5
12.2
11.2
10.4
0.0
0.0
4.2
6.0
0.0
0.0
0.0
23.9
14.3
34.2
4.3
13.0
0.0
4.9
14.2
20.5
16.6
5.6
13.9
9.0
4.0
7.3
5.3
3.2
0.0
0.0
0.0
4.3
7.6
2.6
8.4
7.9
14.2
26.1
14.1
14.2

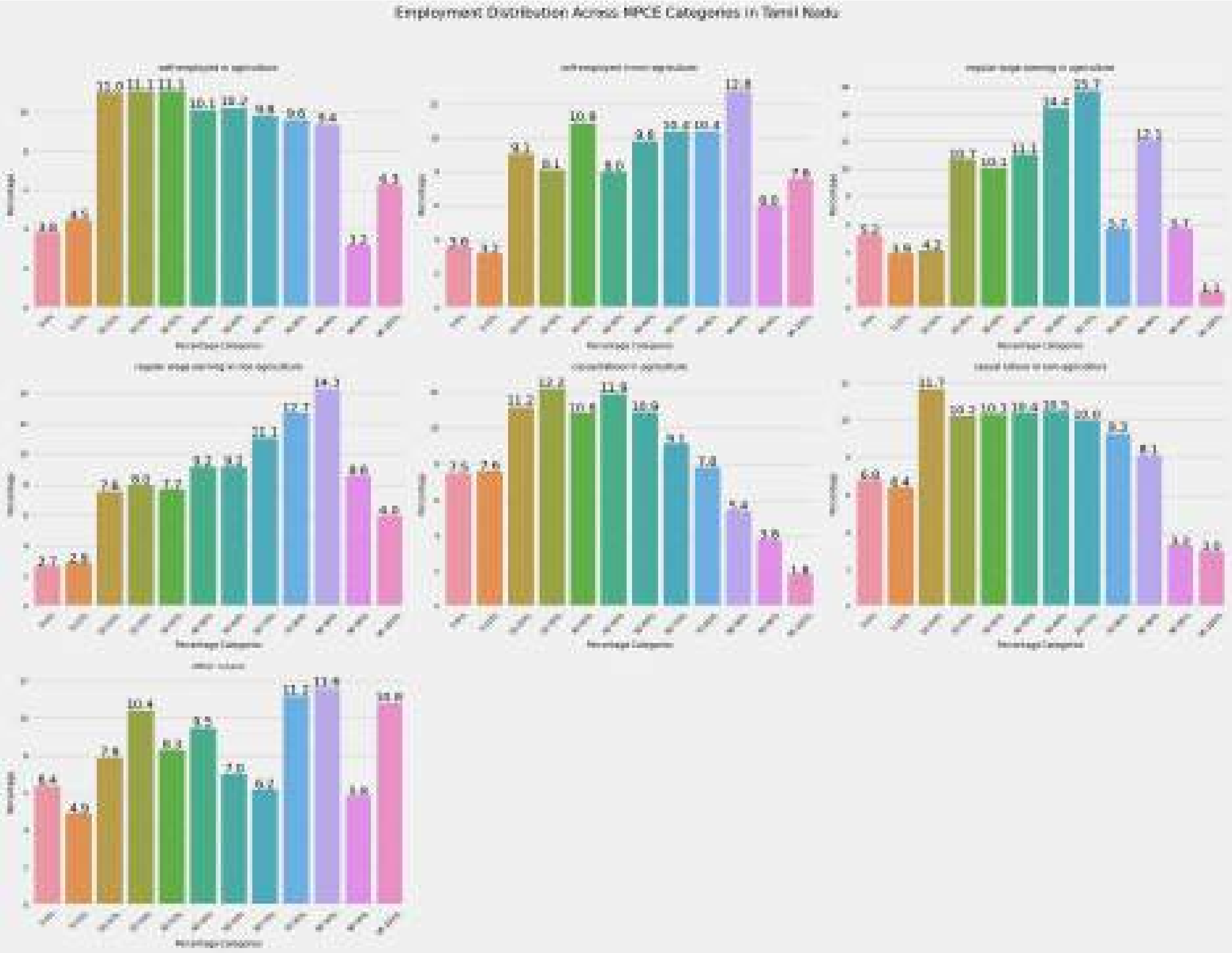

Employment Distribution Across MPCE Categories in Odisha
self-employed in agriculture
5.9
6.0
12.0
11.4
10.9
11.5
9.3
8.8
8.7
8.2
3.8
3.2
Percentage
Percentage Categories
self-employed in non-agriculture
2.0
3.3
4.4
7.9
8.8
9.3
10.3
11.6
12.4
12.8
8.1
8.7
regular wage earning in agriculture
1.1
0.0
6.2
7.1
4.7
3.8
10.7
12.0
19.4
10.3
8.5
15.4
regular wage/salary in non-agriculture
0.8
1.0
4.1
5.3
6.5
8.0
12.1
11.3
11.8
19.4
10.4
13.6
casual labour in agriculture
8.5
8.1
13.9
10.9
12.1
9.0
8.6
8.9
8.5
7.3
2.3
2.0
casual labour in non-agriculture
4.5
4.3
11.5
11.2
10.1
11.2
11.6
11.3
9.9
9.8
2.8
1.7
other sources
8.2
4.4
7.4
8.7
8.5
8.5
8.3
9.7
12.1
10.4
6.3
9.4

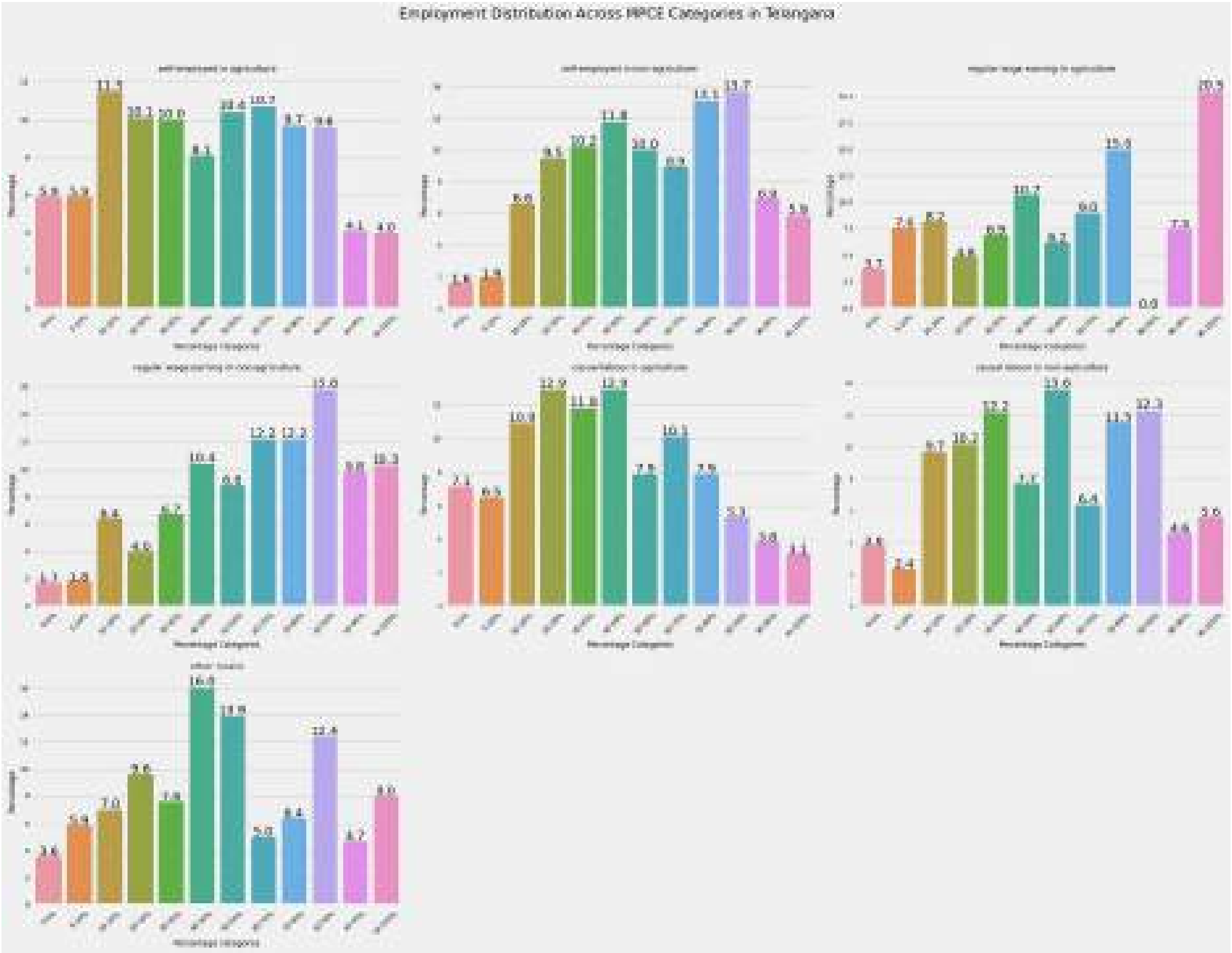

Employment Distribution Across MPCE Categories in Punjab
self-employed in agriculture
0.1
0.4
5.1
7.8
8.1
7.2
9.8
13.6
12.8
14.8
11.1
9.1
self-employed in non-agriculture
4.8
7.5
10.1
8.1
11.8
11.4
10.3
9.2
11.4
9.0
3.0
4.1
regular wage earning in agriculture
12.1
3.8
10.6
3.8
28.0
17.4
10.3
9.5
1.4
2.3
1.9
0.8
regular wage earning in non-agriculture
4.2
7.9
11.2
7.7
7.8
9.8
8.1
11.2
12.4
10.5
5.2
4.1
casual labour in agriculture
8.7
8.2
15.1
11.6
8.6
10.1
11.3
8.8
8.2
5.1
1.6
0.4
casual labour in non-agriculture
9.5
3.7
13.5
15.9
11.7
10.9
11.1
6.1
7.0
6.6
1.9
1.0
others
0.8
1.0
3.0
3.3
7.4
10.1
7.0
15.2
8.2
17.0
6.7
19.3
Percentage
Percentage Categories

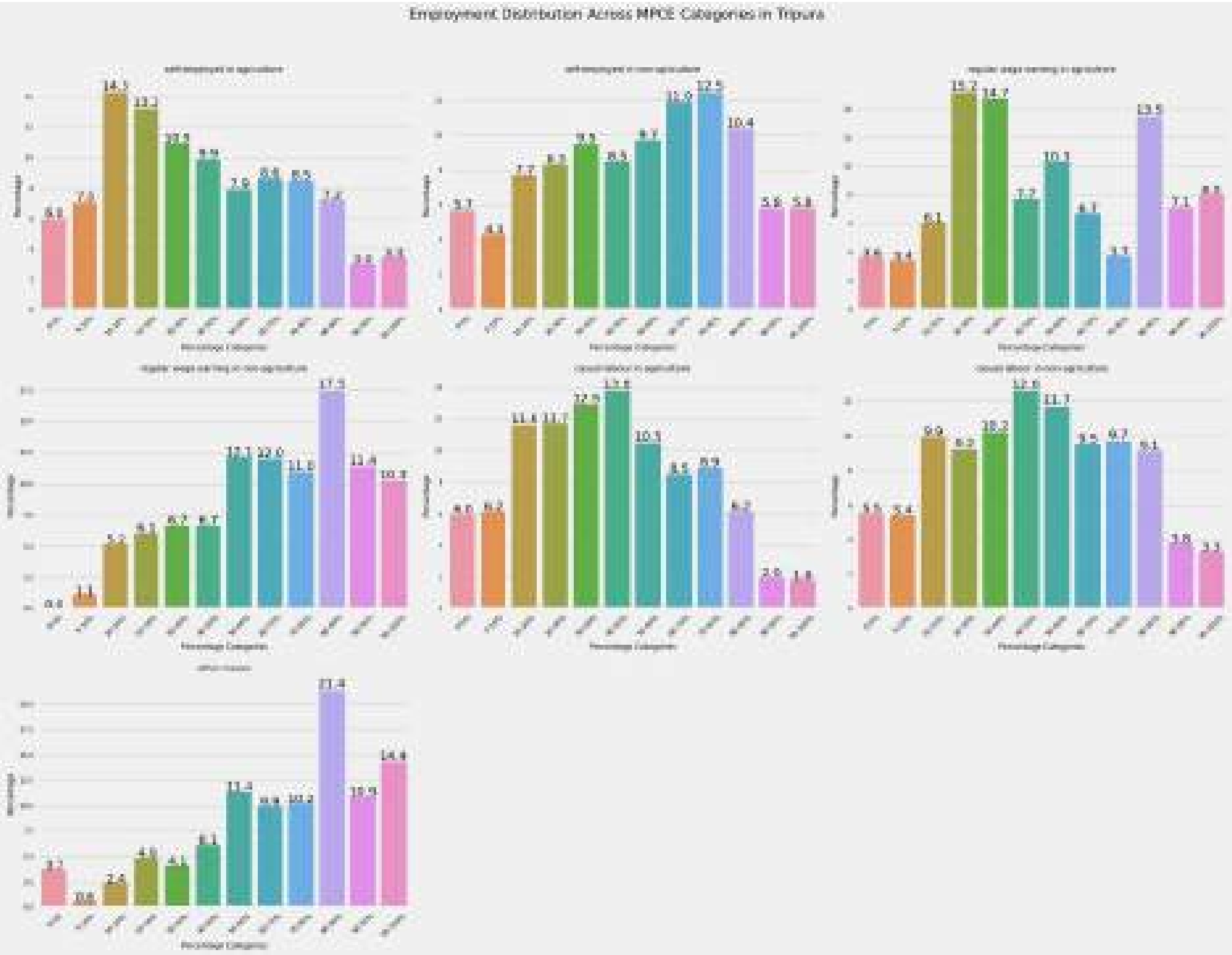

Employment Distribution Across MPCE Categories in Rajasthan
self-employed in agriculture
Percentage
4.9
4.9
8.6
9.9
10.1
10.9
10.9
10.5
10.7
10.2
4.9
3.6
Percentage Categories
self-employed in non-agriculture
Percentage
2.2
2.9
8.8
9.8
10.2
9.9
9.6
9.2
11.5
12.5
7.3
5.8
Percentage Categories
regular wage/salary earning in agriculture
Percentage
8.8
14.4
13.5
5.4
3.3
4.6
3.5
19.2
7.6
9.9
6.1
3.6
Percentage Categories
regular wage/salary earning in non-agriculture
Percentage
1.6
3.5
5.2
7.1
9.5
9.1
10.8
10.4
13.1
15.0
6.6
7.6
Percentage Categories
casual labour in agriculture
Percentage
6.1
6.4
16.7
14.0
13.6
11.2
8.4
9.0
5.6
5.0
1.1
1.7
Percentage Categories
casual labour in non-agriculture
Percentage
8.8
7.1
15.7
12.3
10.2
9.1
9.1
9.0
7.2
6.5
3.3
1.7
Percentage Categories
other sources
Percentage
2.0
3.0
6.0
4.0
5.0
5.0
7.6
9.6
7.6
9.1
6.0
31.8
Percentage Categories

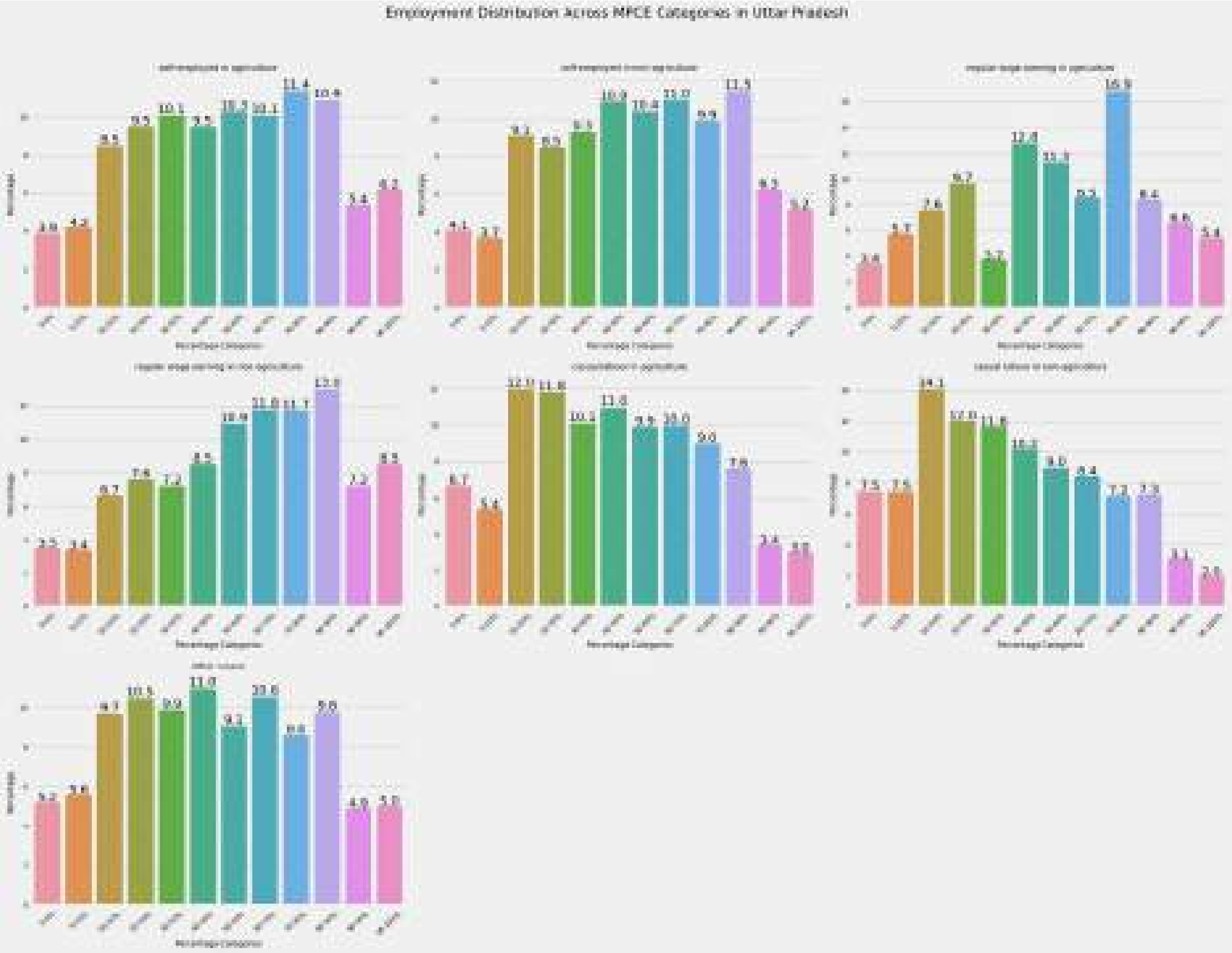


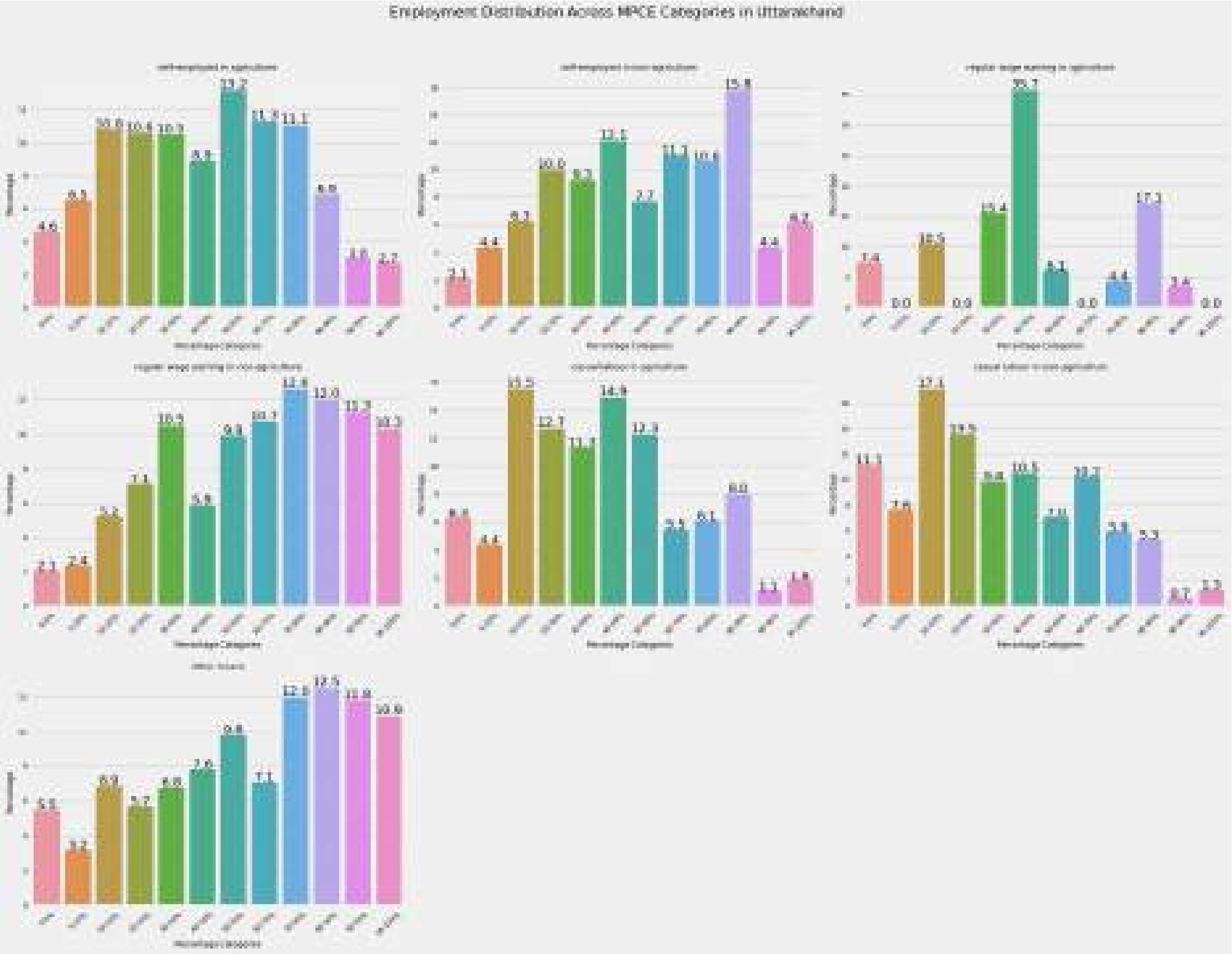
Employment Distribution Across MPCE Categories in Sikkim

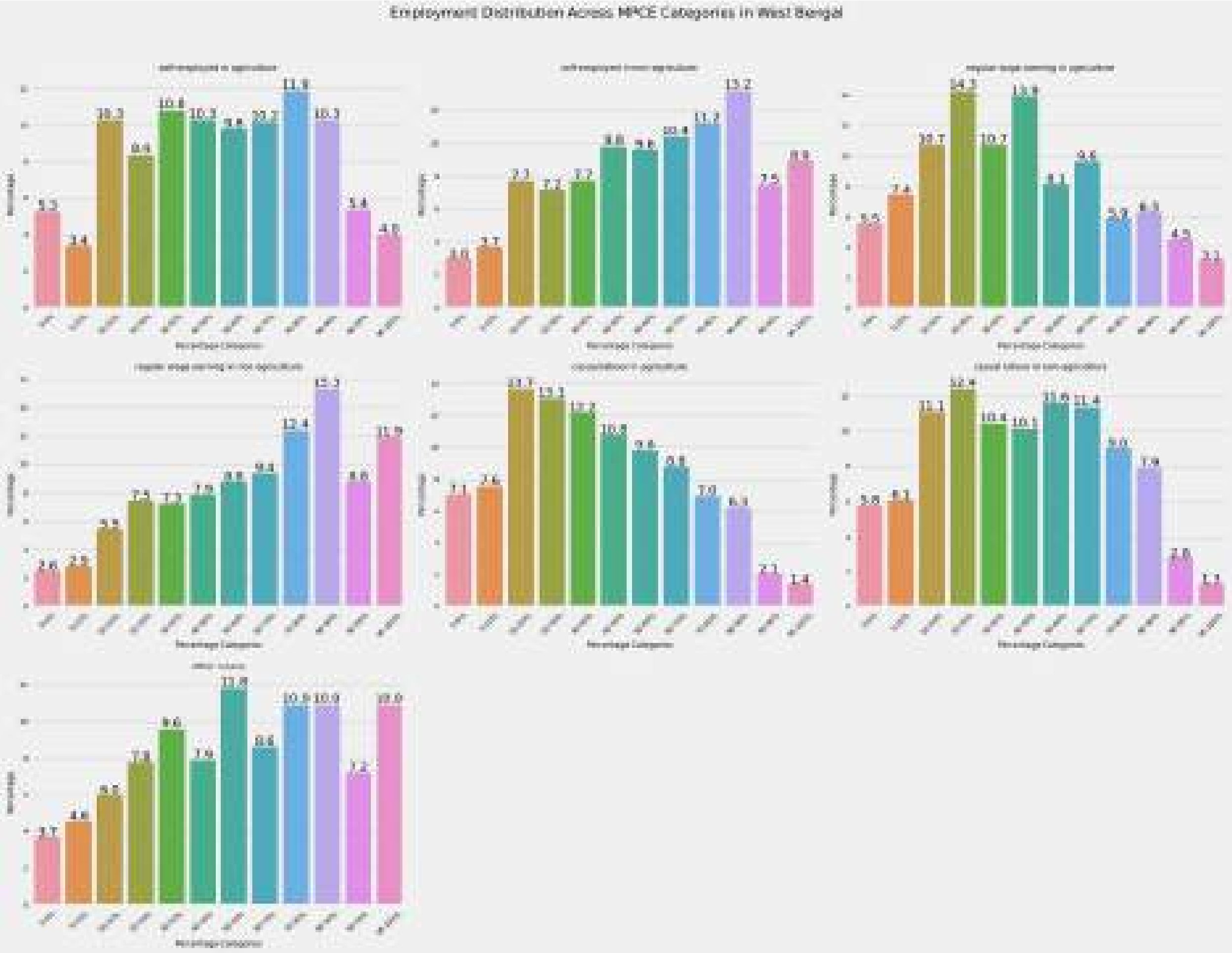

Employment Distribution Across MPCE Categories in Tamil Nadu
self-employed in agriculture
3.8
4.5
11.0
11.1
11.1
10.1
10.2
9.8
9.6
9.4
3.2
6.3
self-employed in non-agriculture
3.6
3.1
9.1
8.1
10.8
8.0
9.6
10.4
10.4
12.8
6.0
7.6
regular wage earning in agriculture
5.2
3.9
4.2
10.7
10.1
11.1
14.4
15.7
5.7
12.1
5.7
1.1
regular wage earning in non-agriculture
2.7
2.8
7.6
8.0
7.7
9.2
9.1
11.1
12.7
14.3
8.6
6.0
casual labour in agriculture
7.5
7.6
11.2
12.2
10.6
11.8
10.9
9.1
7.8
5.4
3.8
1.8
casual labour in non-agriculture
6.9
6.4
11.7
10.2
10.3
10.4
10.5
10.0
9.2
8.1
3.2
3.0
other income
6.4
4.9
7.8
10.4
8.2
9.5
7.0
6.2
11.2
11.6
5.8
10.8
Percentage
Percentage Categories

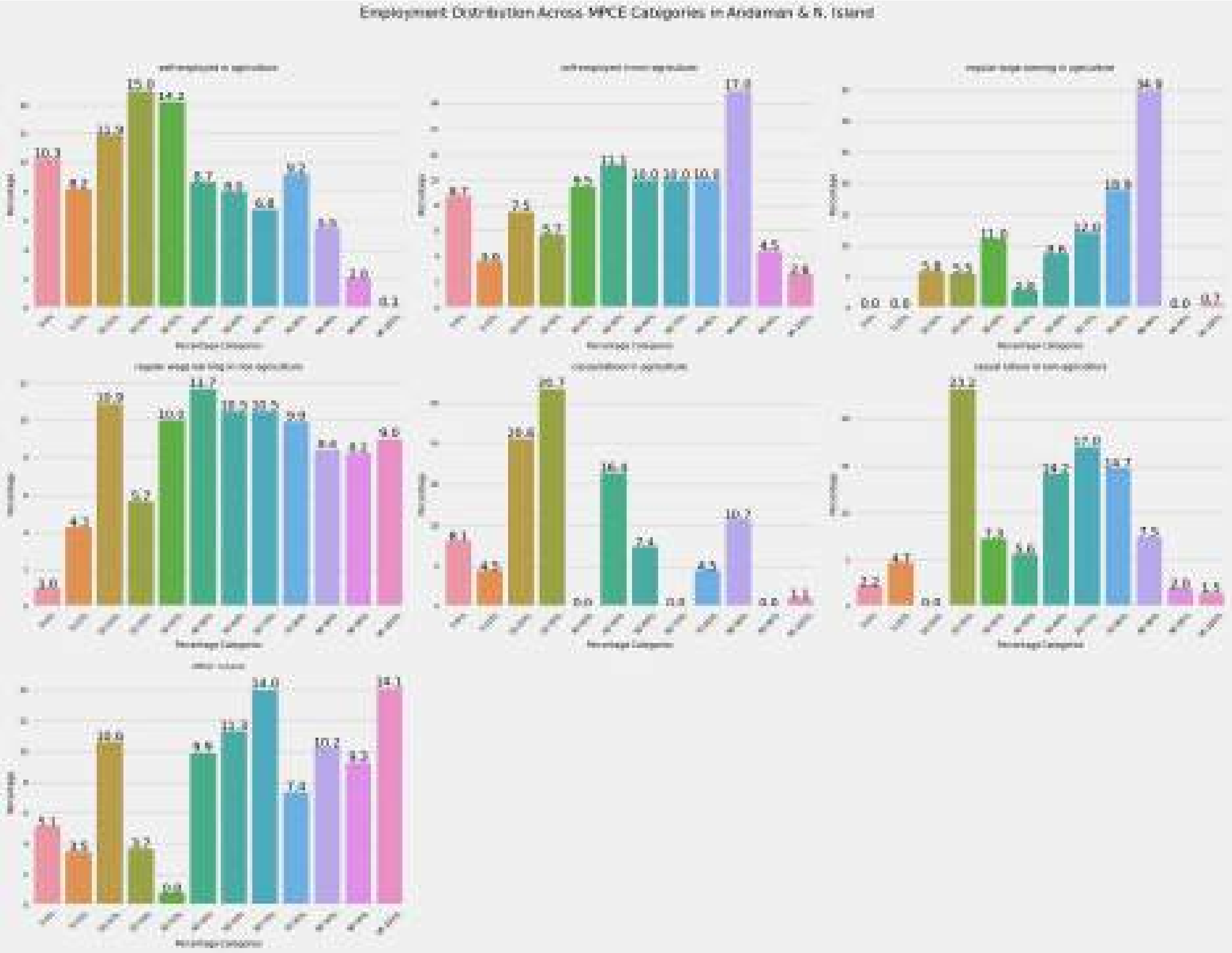

Employment Distribution Across MPCE Categories in Telangana
self-employed in agriculture
5.8
5.8
11.5
10.1
10.0
8.1
10.6
10.7
9.7
9.6
4.1
4.0
self-employed in non-agriculture
1.6
1.9
6.6
9.5
10.2
11.8
10.0
8.9
13.1
13.7
6.9
5.9
regular wage earning in agriculture
3.7
7.4
8.2
4.8
6.9
10.7
6.2
9.0
15.0
0.0
7.3
20.5
regular wage/salary earning in non-agriculture
1.7
1.8
6.4
4.0
6.7
10.4
8.8
12.2
12.2
15.8
9.8
10.3
casual labour in agriculture
7.1
6.5
10.8
12.9
11.6
12.9
7.8
10.1
7.8
5.3
3.8
3.1
casual labour in non-agriculture
3.6
2.4
8.7
10.1
13.2
7.7
15.6
6.4
11.5
12.3
4.6
5.6
other source
3.6
5.8
7.0
9.6
7.8
16.0
13.9
5.0
6.4
12.4
4.7
8.0
Percentage
Percentage Categories

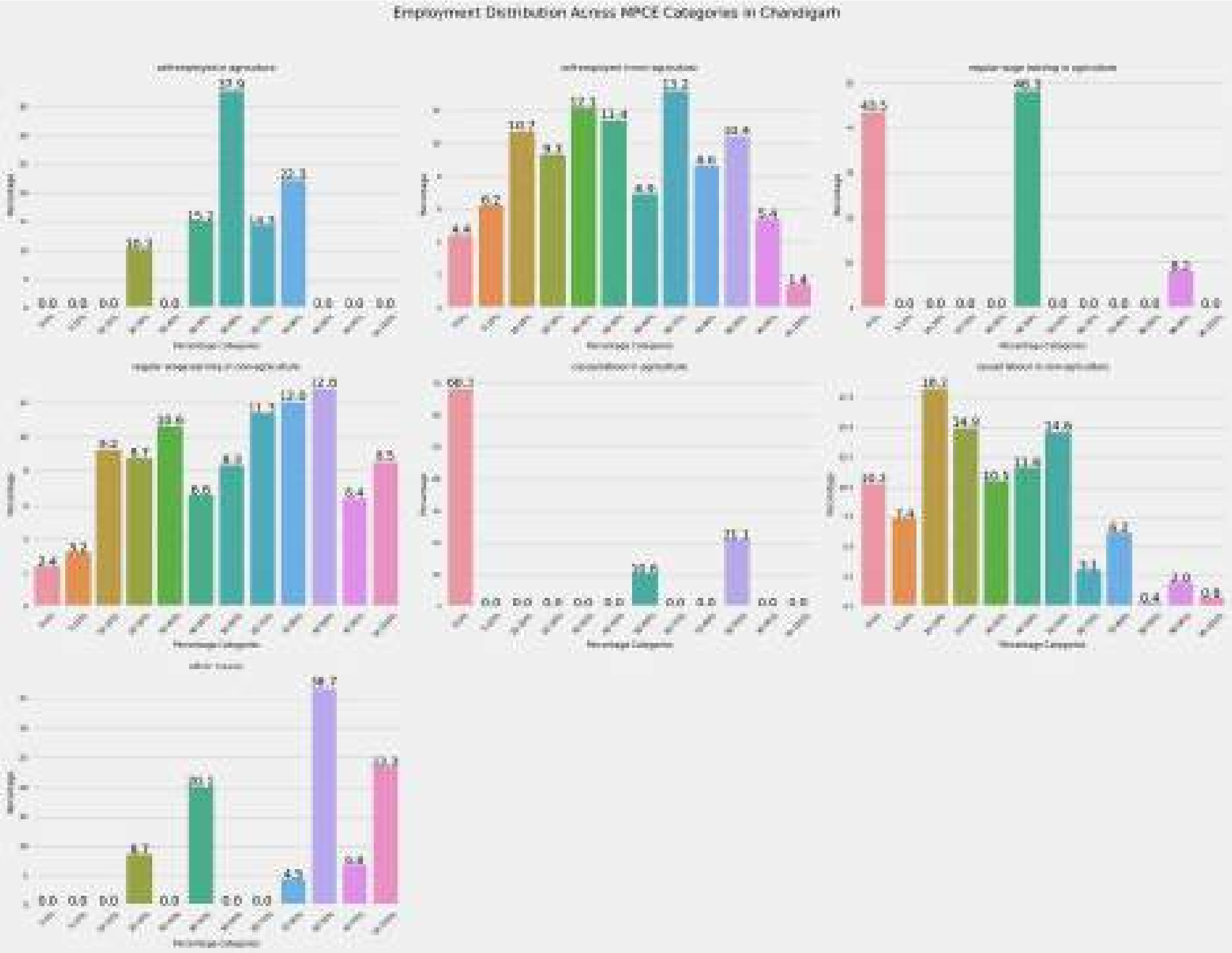

Employment Distribution Across MPCE Categories in Tripura
self-employed in agriculture
6.0
7.0
14.1
13.1
10.9
9.9
7.9
8.6
8.5
7.2
3.0
3.5
Percentage
Percentage Categories
self-employed in non-agriculture
5.7
4.3
7.7
8.3
9.5
8.5
9.7
11.9
12.5
10.4
5.8
5.8
Percentage
Percentage Categories
regular wage earning in agriculture
3.6
3.4
6.1
15.7
14.7
7.7
10.3
6.7
3.7
13.5
7.1
8.0
Percentage
Percentage Categories
regular wage earning in non-agriculture
0.0
1.1
5.2
6.1
6.7
6.7
12.1
12.0
11.0
17.5
11.4
10.3
Percentage
Percentage Categories
casual labour in agriculture
6.0
6.2
11.4
11.7
12.9
13.8
10.5
8.5
8.9
6.2
2.0
1.8
Percentage
Percentage Categories
casual labour in non-agriculture
5.5
5.4
9.9
9.2
10.2
12.6
11.7
9.5
9.7
9.1
3.8
3.3
Percentage
Percentage Categories
others
3.1
0.6
2.4
4.6
4.1
6.1
11.4
9.8
10.2
21.4
10.9
14.4
Percentage
Percentage Categories

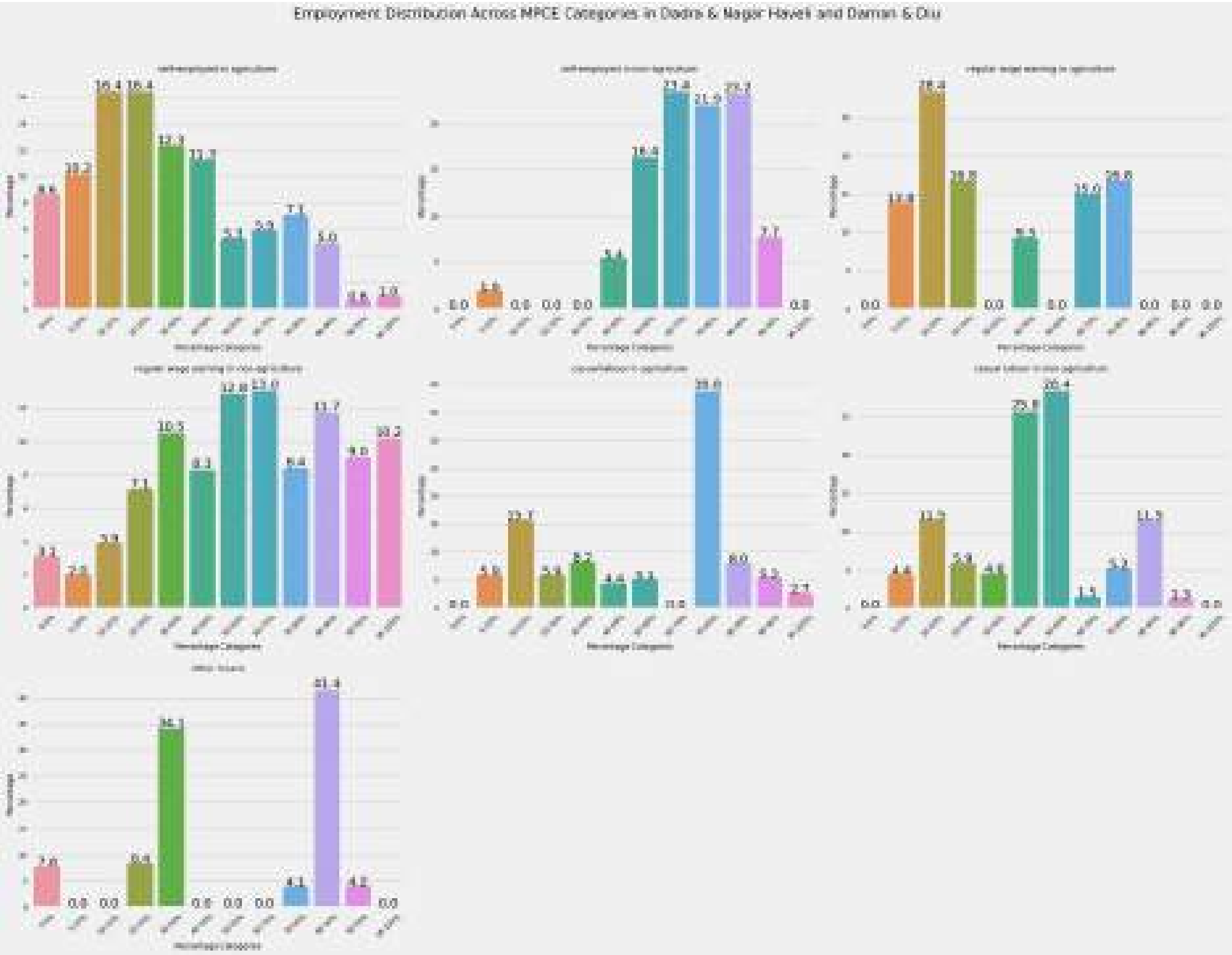

Employment Distribution Across MPCE Categories in Uttar Pradesh
self-employed in agriculture
3.8
4.2
8.5
9.5
10.1
9.5
10.2
10.1
11.4
10.9
5.4
6.2
self-employed in non-agriculture
4.1
3.7
9.1
8.5
9.1
10.9
10.4
11.0
9.9
11.5
6.1
5.2
regular wage earning in agriculture
2.8
5.7
7.6
9.7
3.7
12.8
11.3
9.2
16.9
8.4
6.6
5.4
regular wage earning in non-agriculture
3.5
3.4
6.7
7.6
7.2
8.5
10.9
11.8
11.7
13.0
7.2
8.5
casual labour in agriculture
6.7
5.4
12.1
11.8
10.1
11.0
9.9
10.0
9.0
7.8
3.4
3.0
casual labour in non-agriculture
7.5
7.5
14.1
12.0
11.6
10.2
9.0
8.4
7.3
7.3
3.1
2.0
5.2
5.6
9.7
10.5
9.9
11.0
9.1
10.6
8.6
9.8
4.6
5.0
Percentage
Percentage Categories

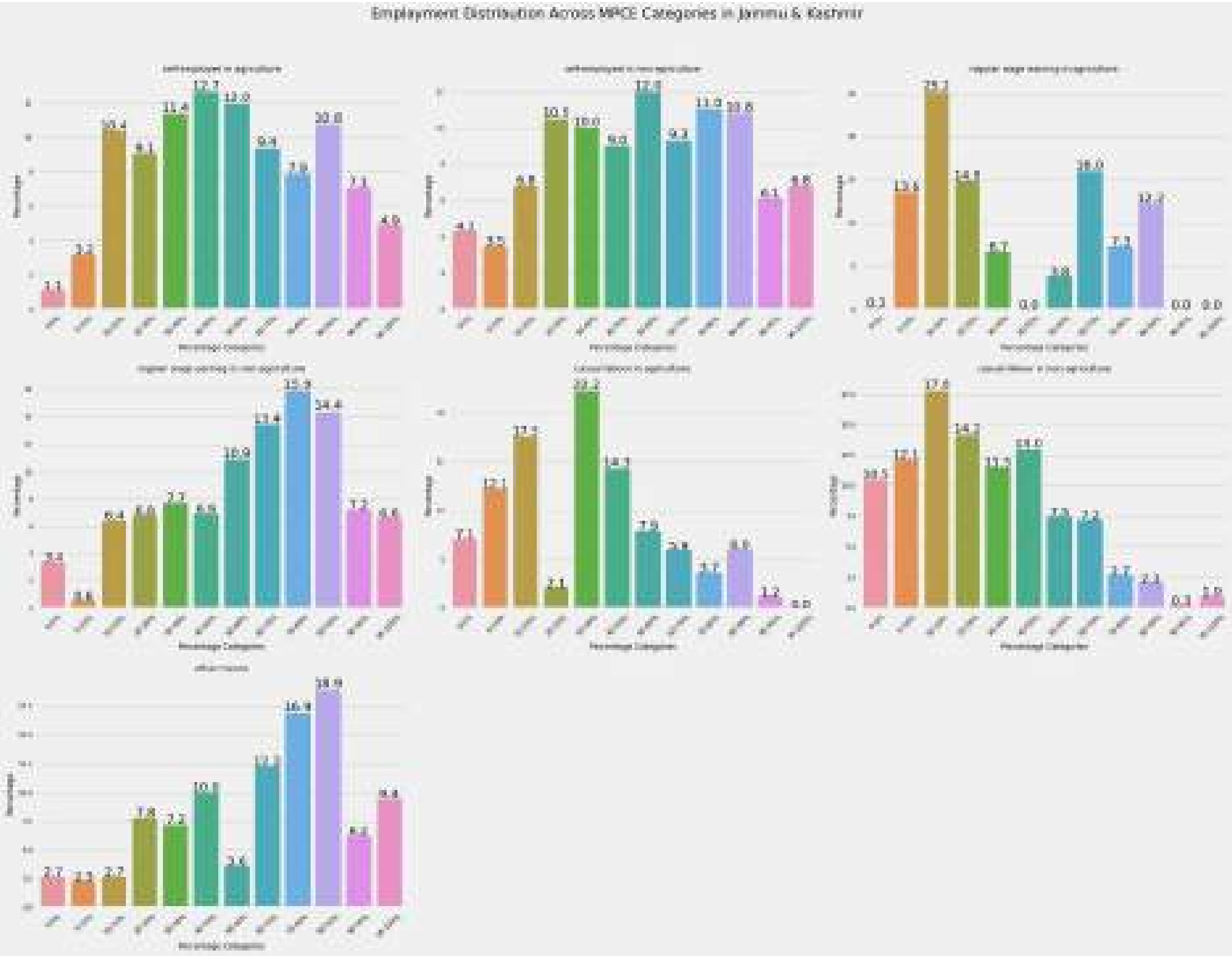

Employment Distribution Across MPCE Categories in Uttarakhand
self-employed in agriculture
Percentage
4.6
6.5
10.6
10.4
10.3
8.8
13.2
11.3
11.1
6.6
3.0
2.7
Percentage Categories
self-employed in non-agriculture
Percentage
2.1
4.4
6.2
10.0
9.2
12.1
7.7
11.3
10.6
15.8
4.4
6.2
Percentage Categories
regular wage earning in agriculture
Percentage
7.4
0.0
10.5
0.0
15.4
35.7
6.1
0.0
4.4
17.3
3.6
0.0
Percentage Categories
regular wage earning in non-agriculture
Percentage
2.1
2.4
5.2
7.1
10.5
5.8
9.8
10.7
12.6
12.0
11.3
10.3
Percentage Categories
casual labour in agriculture
Percentage
6.4
4.4
15.5
12.7
11.3
14.9
12.3
5.5
6.1
8.0
1.1
1.8
Percentage Categories
casual labour in non-agriculture
Percentage
11.1
7.6
17.1
13.5
9.4
10.5
7.0
10.2
5.9
5.3
0.7
1.2
Percentage Categories
Percentage
5.5
3.2
6.9
5.7
6.6
7.6
9.8
7.1
12.1
12.5
11.8
10.8
Percentage Categories

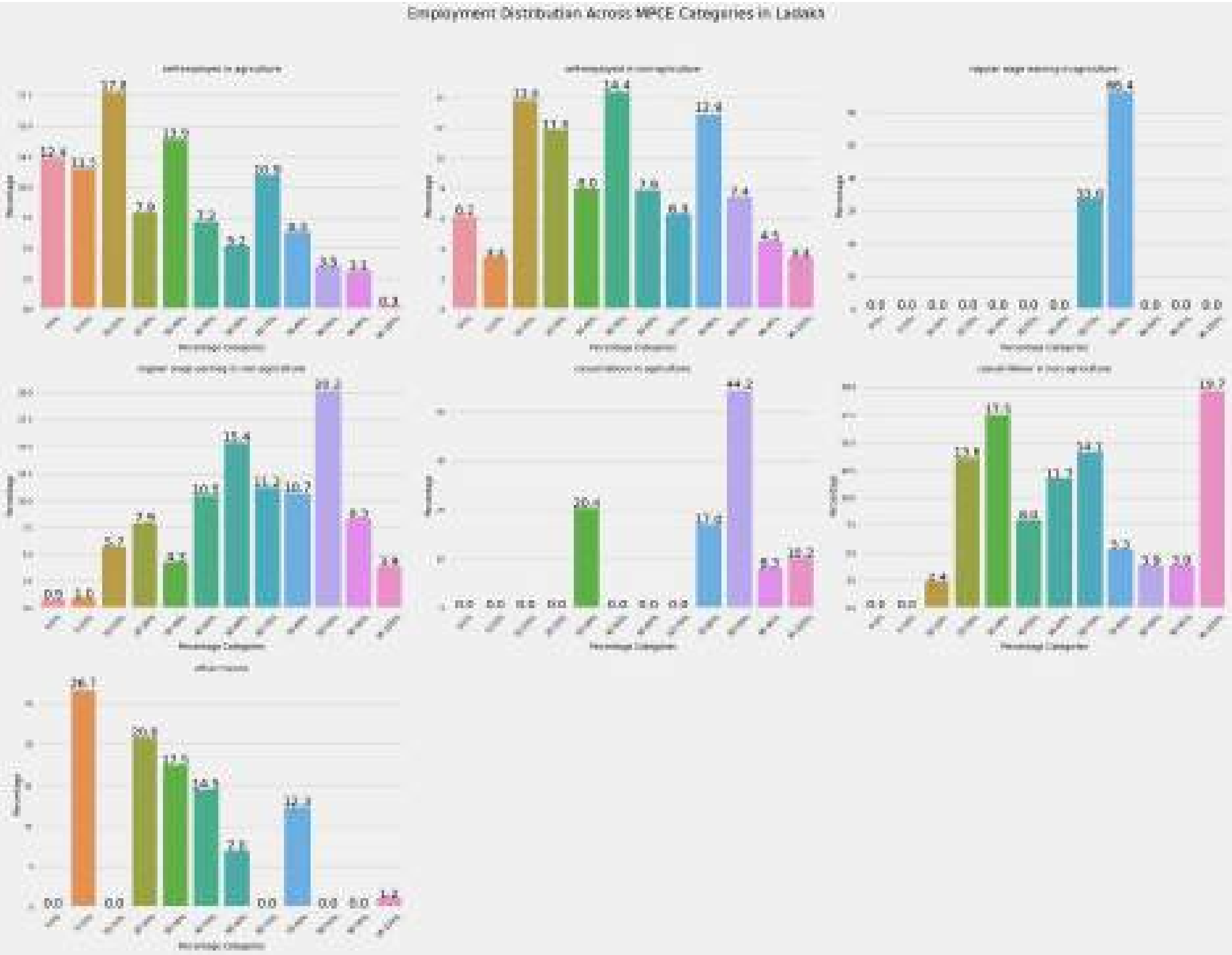

Employment Distribution Across MPCE Categories in West Bengal
self-employed in agriculture
Percentage
5.3
3.4
10.3
8.4
10.8
10.3
9.8
10.2
11.9
10.3
5.4
4.0
Percentage Categories
self-employed in non-agriculture
Percentage
4.0
3.7
7.7
7.2
7.7
8.8
8.6
10.4
11.2
13.2
7.5
8.9
Percentage Categories
regular wage earning in agriculture
Percentage
5.5
7.4
10.7
14.3
10.7
13.9
8.1
9.6
5.9
6.3
4.5
3.1
Percentage Categories
regular wage earning in non agriculture
Percentage
2.6
2.5
5.5
7.5
7.3
7.9
8.8
9.4
12.4
15.3
8.6
11.9
Percentage Categories
casual labour in agriculture
Percentage
7.1
7.6
13.7
13.1
12.2
10.8
9.6
8.8
7.0
6.1
2.1
1.4
Percentage Categories
casual labour in non-agriculture
Percentage
5.8
6.1
11.1
12.4
10.6
10.1
11.6
11.4
9.0
7.9
2.8
1.3
Percentage Categories
Percentage
3.7
4.6
5.9
7.8
9.6
7.9
11.8
8.6
10.9
10.9
7.2
10.9
Percentage Categories

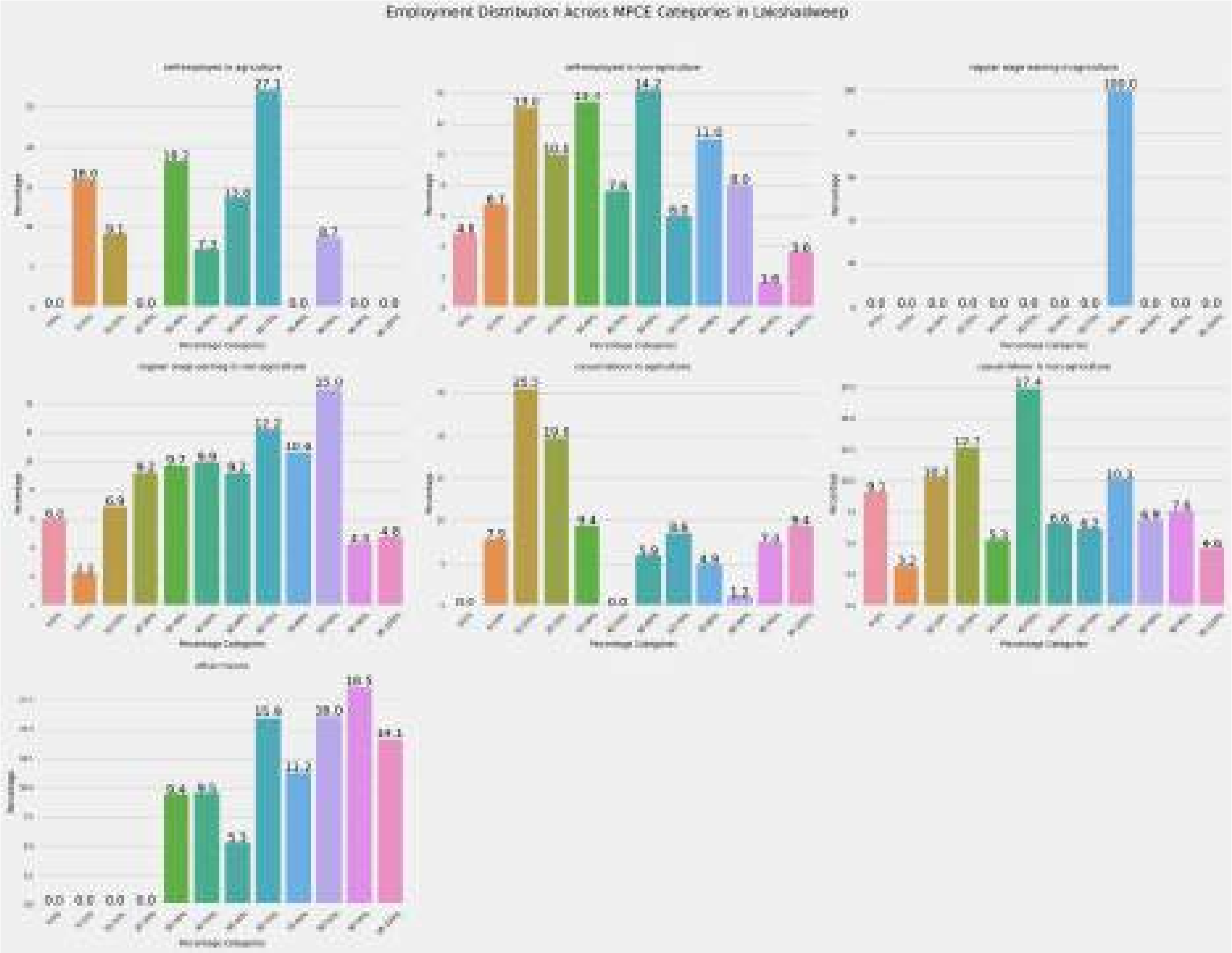

Employment Distribution Across MPCE Categories in Andaman & N. Island
self-employed in agriculture
Percentage
10.3
8.2
11.9
15.0
14.2
8.7
8.0
6.8
9.2
5.5
2.0
0.3
Percentage Categories
self-employed in non-agriculture
Percentage
8.7
3.6
7.5
5.7
9.5
11.1
10.0
10.0
10.0
17.0
4.5
2.6
Percentage Categories
regular wage/salary earning in agriculture
Percentage
0.0
0.0
5.8
5.5
11.0
2.8
8.6
12.0
18.8
34.9
0.0
0.7
Percentage Categories
regular wage/salary earning in non-agriculture
Percentage
1.0
4.1
10.9
5.7
10.0
11.7
10.5
10.5
9.9
8.6
8.1
9.6
casual labour in agriculture
Percentage
8.1
4.5
19.6
25.7
0.0
16.4
7.4
0.0
4.5
10.7
0.0
1.1
Percentage Categories
casual labour in non-agriculture
Percentage
2.2
4.7
0.0
23.2
7.3
5.6
14.2
17.0
14.7
7.5
2.0
1.5
Percentage Categories
others
Percentage
5.1
3.5
10.6
3.7
0.8
9.6
11.3
14.0
7.4
10.2
9.2
14.1
Percentage Categories

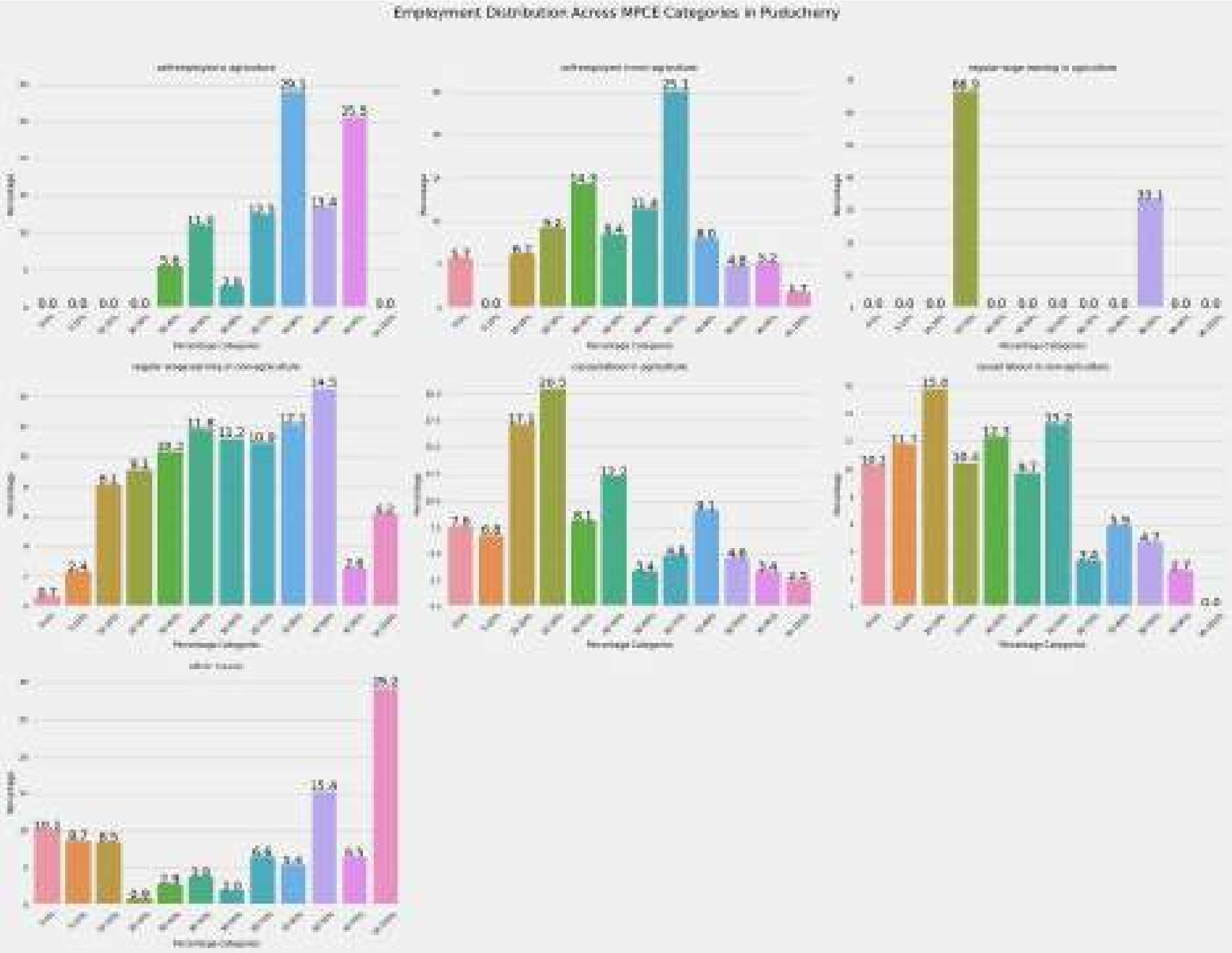

Employment Distribution Across MPCE Categories in Chandigarh
self-employed in agriculture
0.0
0.0
0.0
10.3
0.0
15.2
37.9
14.3
22.1
0.0
0.0
0.0
self-employed in non-agriculture
4.4
6.2
10.7
9.3
12.1
11.4
6.6
13.2
8.6
10.4
5.4
1.4
regular wage/salary earning in agriculture
43.5
0.0
0.0
0.0
0.0
48.3
0.0
0.0
0.0
0.0
8.2
0.0
regular wage/salary earning in non-agriculture
2.4
3.2
9.2
8.7
10.6
6.6
8.3
11.7
12.0
12.0
6.4
8.5
casual labour in agriculture
68.1
0.0
0.0
0.0
0.0
0.0
10.6
0.0
0.0
21.3
0.0
0.0
casual labour in non-agriculture
10.3
7.4
18.2
14.9
10.5
11.6
14.6
3.1
6.2
0.4
2.0
0.8
others
0.0
0.0
0.0
8.7
0.0
20.1
0.0
0.0
4.3
58.7
6.6
13.3
Percentage
Percentage Categories

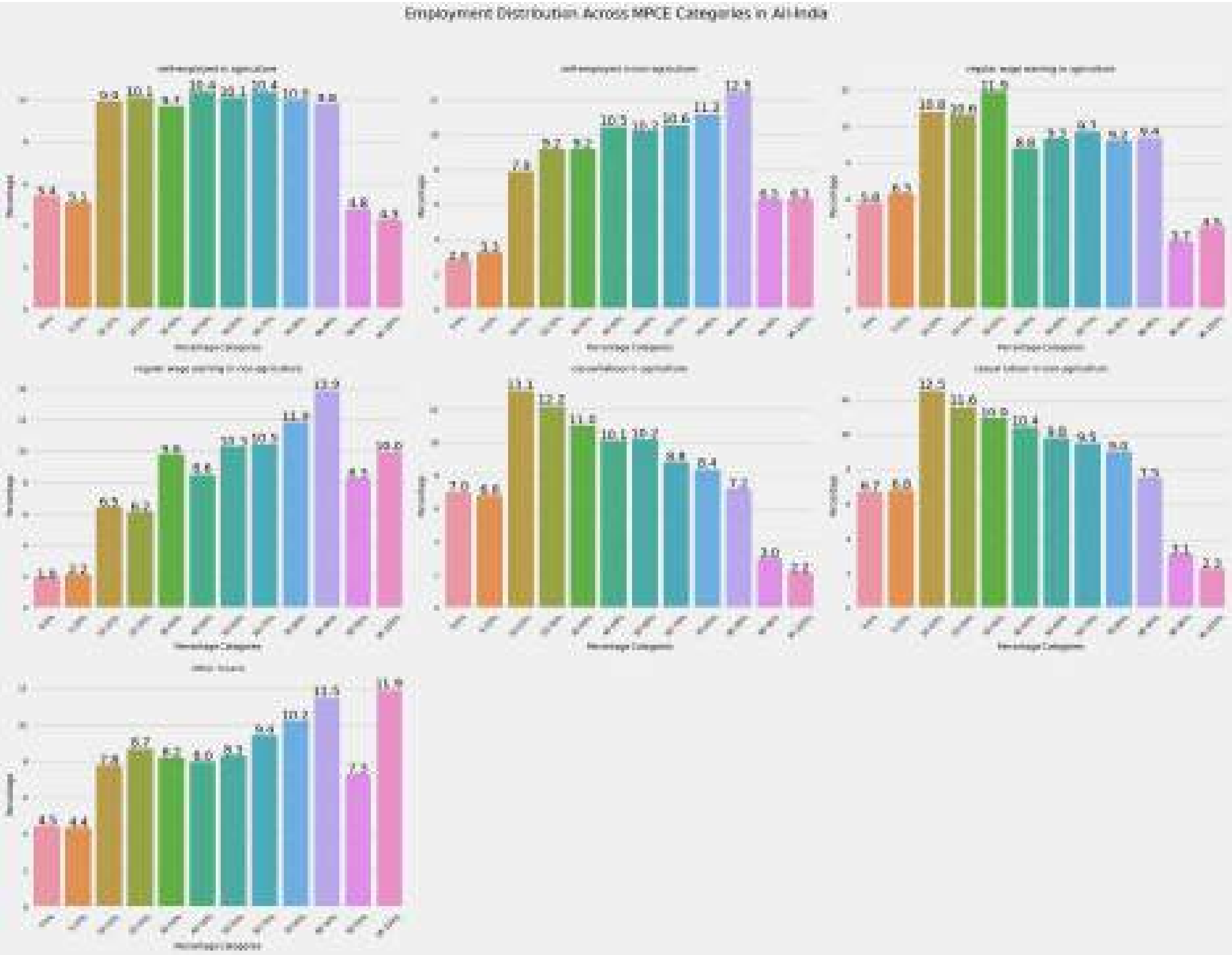

Employment Distribution Across MPCE Categories in Dadra & Nagar Haveli and Daman & Diu
self-employed in agriculture
8.6
10.2
16.4
16.4
12.3
11.3
5.1
5.9
7.1
5.0
0.6
1.0
self-employed in non-agriculture
0.0
1.6
0.0
0.0
0.0
5.4
16.4
23.4
21.9
23.3
7.7
0.0
regular wage earning in agriculture
0.0
13.8
28.4
19.8
0.0
9.3
0.0
15.0
16.8
0.0
0.0
0.0
regular wage earning in non-agriculture
3.1
2.0
3.9
7.1
10.5
8.1
12.8
13.0
8.4
11.7
9.0
10.2
casual labour in agriculture
0.0
5.9
15.7
5.8
8.2
4.4
5.1
0.0
39.0
8.0
5.2
2.7
casual labour in non-agriculture
0.0
4.4
11.5
5.9
4.6
25.8
28.4
1.1
5.2
11.5
1.1
0.0
others
7.6
0.0
0.0
8.4
36.1
0.0
0.0
0.0
4.1
41.3
4.2
0.0
Percentage
Percentage Categories


Employment Distribution Across MPCE Categories in Jammu & Kashmir
Self-employed in agriculture
1.1
3.2
10.4
8.1
11.4
12.7
12.0
9.4
7.8
10.8
7.1
4.9
Self-employed in non-agriculture
4.1
3.5
6.8
10.7
10.0
9.0
12.0
9.3
11.0
10.8
6.1
6.8
Regular wage earning in agriculture
0.3
13.6
25.2
14.3
6.7
0.0
3.8
16.0
7.1
12.7
0.0
0.0
Regular wage earning in non-agriculture
3.2
0.6
6.4
6.8
7.3
6.5
10.9
13.4
15.9
14.4
7.2
6.6
Casual labour in agriculture
7.1
12.1
17.5
2.1
22.2
14.3
7.9
5.9
3.7
6.0
1.2
0.0
Casual labour in non-agriculture
10.5
12.1
17.0
14.2
11.5
13.0
7.5
7.2
2.7
2.1
0.3
1.0
Others
2.7
2.3
2.7
7.8
7.2
10.0
3.6
12.3
16.9
18.9
6.2
9.4
Percentage
Percentage Categories


Employment Distribution Across MPCE Categories in Ladakh
Self-employed in agriculture
12.4
11.5
17.8
7.9
13.9
7.2
5.2
10.9
6.6
3.5
3.1
0.3
Self-employed in non-agriculture
6.7
3.3
13.8
11.8
8.0
14.4
7.9
6.3
12.6
7.4
4.5
3.4
Regular wage earning in agriculture
0.0
0.0
0.0
0.0
0.0
0.0
0.0
33.6
66.4
0.0
0.0
0.0
Regular wage earning in non-agriculture
0.0
1.0
5.7
7.9
3.3
10.7
15.4
11.2
10.7
20.2
8.3
3.8
Casual labour in agriculture
0.0
0.0
0.0
0.0
20.4
0.0
0.0
0.0
17.4
44.2
8.3
10.2
Casual labour in non-agriculture
0.0
0.0
2.4
13.8
17.1
8.0
11.7
14.1
5.3
3.9
3.9
19.7
Others
0.0
26.1
0.0
20.8
17.5
14.5
7.1
0.0
12.3
0.0
0.0
1.2
Percentage
Percentage Categories


Employment Distribution Across MPCE Categories in Lakshadweep
self-employed in agriculture
Percentage
0.0
18.0
9.1
0.0
18.2
7.3
12.8
27.1
0.0
8.7
0.0
0.0
Percentage Categories
self-employed in non-agriculture
Percentage
4.8
6.7
13.6
10.1
13.7
7.8
14.7
6.0
11.0
8.0
1.6
3.6
Percentage Categories
regular wage earning in agriculture
Percentage
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
100.0
0.0
0.0
0.0
Percentage Categories
regular wage earning in non-agriculture
Percentage
6.0
2.2
6.9
9.2
9.7
9.9
9.2
12.2
10.8
15.0
4.3
4.8
Percentage Categories
casual labour in agriculture
Percentage
0.0
7.5
25.2
19.6
9.4
0.0
5.9
8.6
4.9
1.2
7.3
9.4
Percentage Categories
casual labour in non-agriculture
Percentage
9.1
3.2
10.1
12.7
5.3
17.4
6.8
6.2
10.3
6.8
7.6
4.6
Percentage Categories
others
Percentage
0.0
0.0
0.0
0.0
9.4
9.5
5.1
15.6
11.7
15.6
16.5
14.1
Percentage Categories


Employment Distribution Across MPCE Categories in Puducherry
self-employed in agriculture
Percentage
0.0 0.0 0.0 0.0 5.8 11.6 2.9 12.9 29.1 13.4 25.5 0.0
Percentage Categories
self-employed in non-agriculture
Percentage
5.7 0.0 6.7 9.2 14.9 8.4 11.4 25.1 8.0 4.8 5.2 1.7
Percentage Categories
regular wage earning in agriculture
Percentage
0.0 0.0 0.0 66.9 0.0 0.0 0.0 0.0 0.0 33.1 0.0 0.0
Percentage Categories
regular wage earning in non-agriculture
Percentage
0.7 2.4 8.1 9.1 10.2 11.8 11.2 10.9 12.1 14.5 2.6 6.2
Percentage Categories
casual labour in agriculture
Percentage
7.6 6.8 17.1 20.5 8.1 12.2 3.4 4.8 9.1 4.8 3.4 2.5
Percentage Categories
casual labour in non-agriculture
Percentage
10.1 11.7 15.6 10.4 12.3 9.3 13.2 3.5 5.9 4.7 2.7 0.0
Percentage Categories
other sources
Percentage
10.0 8.7 8.5 0.6 2.9 3.9 2.0 6.6 5.4 15.4 6.5 39.2
Percentage Categories



```
data_a8=pd.read_excel('/kaggle/input/mospi-hces/Table A8R.xlsx',header=[0,1,2,3])
data_a8.columns=['col0','col1','self-employed in agriculture','self-employed in non-agriculture','regular wage earning in agriculture','regular wage earning in non-agriculture','casual labour in agriculture','casual labour in non-agriculture','other means','All','col10','col11']

data_a8['col0']=data_a8['col0'].ffill()
data_a8['regular wage earning in agriculture']=data_a8['regular wage earning in agriculture'].replace('-',0.0)
data_a8.head(20)
```

```
<ipython-input-196-e2f08b077866>:5: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in a future version. To retain the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  data_a8['regular wage earning in agriculture']=data_a8['regular wage earning in agriculture'].replace('-',0.0)
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
```

```
58: RuntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()

                 col0                         col1  \
0      Andhra Pradesh                         0-5%
1      Andhra Pradesh                        5-10%
2      Andhra Pradesh                       10-20%
3      Andhra Pradesh                       20-30%
4      Andhra Pradesh                       30-40%
5      Andhra Pradesh                       40-50%
6      Andhra Pradesh                       50-60%
7      Andhra Pradesh                       60-70%
8      Andhra Pradesh                       70-80%
9      Andhra Pradesh                       80-90%
10     Andhra Pradesh                       90-95%
11     Andhra Pradesh                      95-100%
12     Andhra Pradesh                  All classes
13     Andhra Pradesh              Avg. MPCE (Rs.)
14     Andhra Pradesh  Estd. no. of households(00)
15     Andhra Pradesh     No. of sample households
16     Andhra Pradesh     Estd. no. of persons(00)
17     Andhra Pradesh        No. of sample persons
18  Arunachal Pradesh                         0-5%
19  Arunachal Pradesh                        5-10%

    self-employed in agriculture  self-employed in non-agriculture  \
0                           6.70                              2.60
1                           4.50                              3.50
2                          10.10                              7.80
3                          10.00                             10.50
4                          10.40                             10.10
5                          10.10                             11.10
6                           9.90                             10.10
7                           9.10                             12.10
8                           9.80                             11.50
9                           8.60                             10.60
10                          5.00                              5.10
11                          5.90                              5.00
12                        100.00                            100.00
13                       4905.71                           4999.95
14                      24369.00                          13377.00
15                       1586.00                            878.00
```

```
16                      92713.00                          48099.00
17                       6057.00                           3187.00
18                          6.90                              0.00
19                          6.60                              1.40

    regular wage earning in agriculture  \
0                                  1.70   
1                                  2.30   
2                                  1.70   
3                                  6.50   
4                                 21.10   
5                                  1.70   
6                                  6.70   
7                                 18.50   
8                                  9.30   
9                                  8.20   
10                                11.10   
11                                11.30   
12                               100.00   
13                              5905.55   
14                               886.00   
15                                53.00   
16                              3151.00   
17                               186.00   
18                                 3.70   
19                                 5.50   

    regular wage earning in non-agriculture  casual labour in 
agriculture  \
0                                      3.00                          
5.40   
1                                      3.60                          
7.00   
2                                      7.90                          
12.30   
3                                      7.30                          
11.50   
4                                      8.00                          
10.80   
5                                     10.40                          
10.40   
6                                      9.10                          
10.00   
7                                     11.90                          
8.70   
8                                     10.90                          
8.20   
9                                     13.30                          
9.20   
```

```
10                                     7.00
3.80
11                                     7.60
2.70
12                                   100.00
100.00
13                                  5334.53
4488.59
14                                 10539.00
21066.00
15                                   663.00
1446.00
16                                 38679.00
69271.00
17                                  2467.00
4832.00
18                                     1.70
21.10
19                                     0.60
36.30

    casual labour in non-agriculture  other means        All    col10
col11
0                               4.80         5.90        5.0   3723.0
319.0
1                               6.10         4.90        5.0   3701.0
281.0
2                              12.10         8.40       10.0   7785.0
542.0
3                              10.20         9.30       10.0   8260.0
551.0
4                              10.70         6.60       10.0   8729.0
573.0
5                               9.70         7.00       10.0   8837.0
586.0
6                              11.90         8.90       10.0   9532.0
623.0
7                              10.00         8.70       10.0  10055.0
648.0
8                               9.70        12.10       10.0  10516.0
643.0
9                               8.40        14.00       10.0  11880.0
705.0
10                              3.80         6.40        5.0   5778.0
364.0
11                              2.50         7.70        5.0   7019.0
410.0
12                            100.00       100.00      100.0  95813.0
6245.0
```

```
13                           4524.13      5291.51    4870.3      NaN
NaN
14                          10882.00     14694.00   95813.0      NaN
NaN
15                            732.00       887.00    6245.0      NaN
NaN
16                          40445.00     23038.00  315396.0      NaN
NaN
17                           2749.00      1429.00   20907.0      NaN
NaN
18                              3.20         0.00       5.0     62.0
92.0
19                              6.20         0.00       5.0     73.0
93.0
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# List of columns to plot
employment_columns = ['self-employed in agriculture',
                     'self-employed in non-agriculture',
                     'regular wage earning in agriculture',
                     'regular wage earning in non-agriculture',
                     'casual labour in agriculture',
                     'casual labour in non-agriculture',
                     'other means',
                     'All']

# Get unique states
states = data_a8['col0'].unique()

# Calculate number of rows and columns for subplots
num_states = len(states)
num_cols = 3  # You can adjust this
num_rows = (num_states + num_cols - 1) // num_cols

# Create figure and subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(20, 5*num_rows))
axes = axes.flatten()  # Flatten the axes array for easier indexing

# Create a bar plot for each state
for idx, state in enumerate(states):
    # Filter data for current state and 'Avg. MPCE' row
    state_data = data_a8[(data_a8['col0'] == state) &
                        (data_a8['col1'] == 'Avg. MPCE (Rs.)')]

    if not state_data.empty:
        # Get values for plotting
        values = state_data[employment_columns].values[0]
        # Create bar plot
        bars = sns.barplot(x=employment_columns, y=values,
ax=axes[idx])

        # Customize subplot
        axes[idx].set_title(f"{state}", pad=10)
        axes[idx].set_ylabel('Average MPCE (Rs.)')

        # Rotate x-axis labels
        axes[idx].set_xticklabels(employment_columns, rotation=45,
horizontalalignment='right')

        # Add value labels on top of bars
        for i, v in enumerate(values):
            axes[idx].text(i, v + v*0.01, f'{v:.0f}', ha='center')

# Remove empty subplots if any
for idx in range(len(states), len(axes)):
    fig.delaxes(axes[idx])

# Add a main title
fig.suptitle('Average MPCE (Rs.) by Employment Category Across
States',
             fontsize=16, y=1.02)

# Adjust layout to prevent label overlap
plt.tight_layout()

# Show plot
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
```

```
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
```

```
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py:1765:
FutureWarning: unique with argument that is not not a Series, Index,
ExtensionArray, or np.ndarray is deprecated and will raise in a future
version.
  order = pd.unique(vector)
```

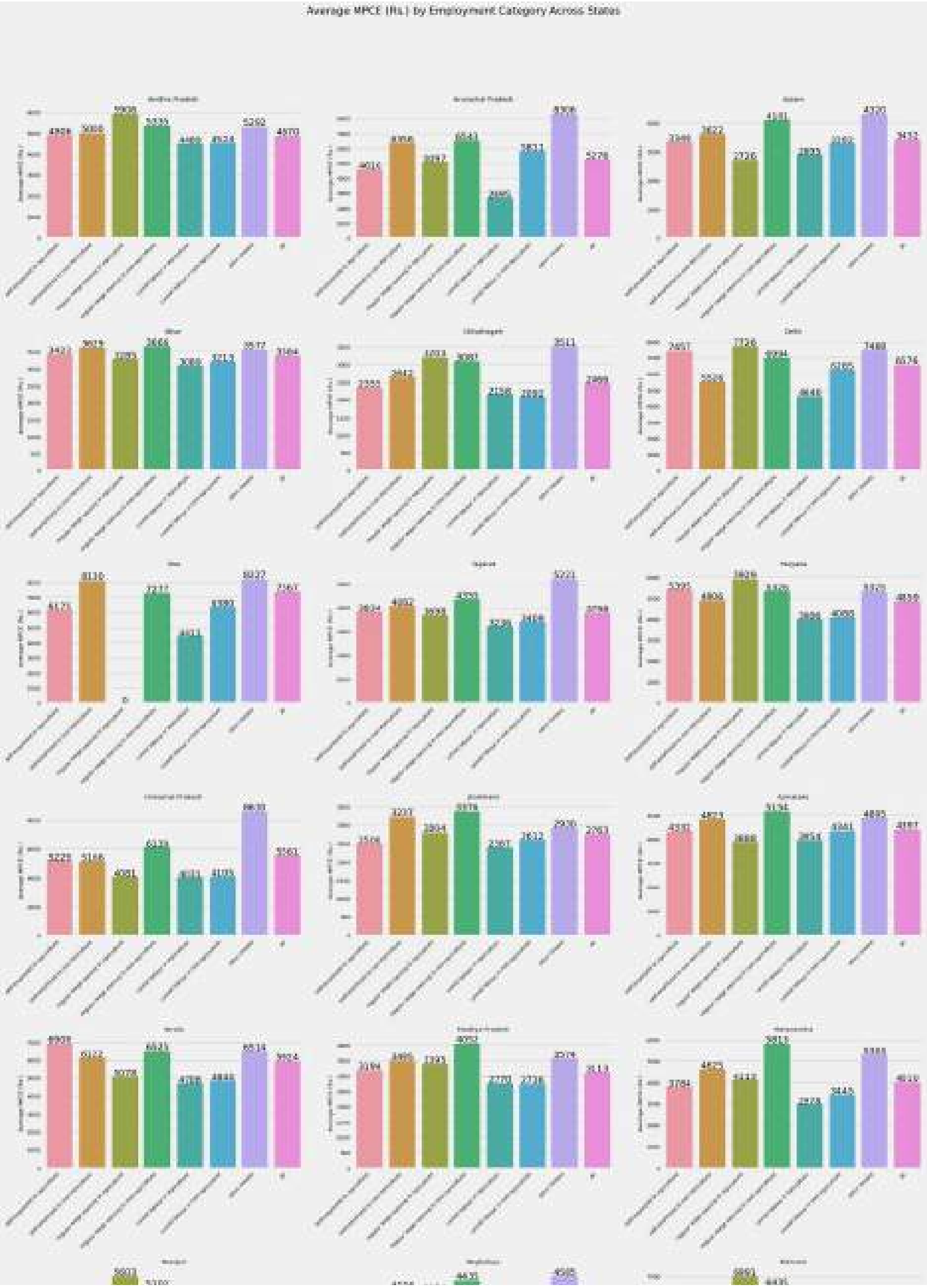

Average MPCE (Rs.) by Employment Category Across States

Figure 7 Categories of people in each MPCE

```
data_a9=pd.read_excel('/kaggle/input/mospi-hces/Table
A9.xlsx',header=[0,1,2,3])
data_a9.columns=['col0','state','col2','ST','SC','OBC','Others','All',
'col8','col9']
data_a9['col0']=data_a9['col0'].ffill()
data_a9['state']=data_a9['state'].ffill()
data_a9['Others']=data_a9['Others'].replace('-',0.0)
data_a9['ST']=data_a9['ST'].replace('-',0.0)
data_a9['SC']=data_a9['SC'].replace('-',0.0)
data_a9.head(20)
```

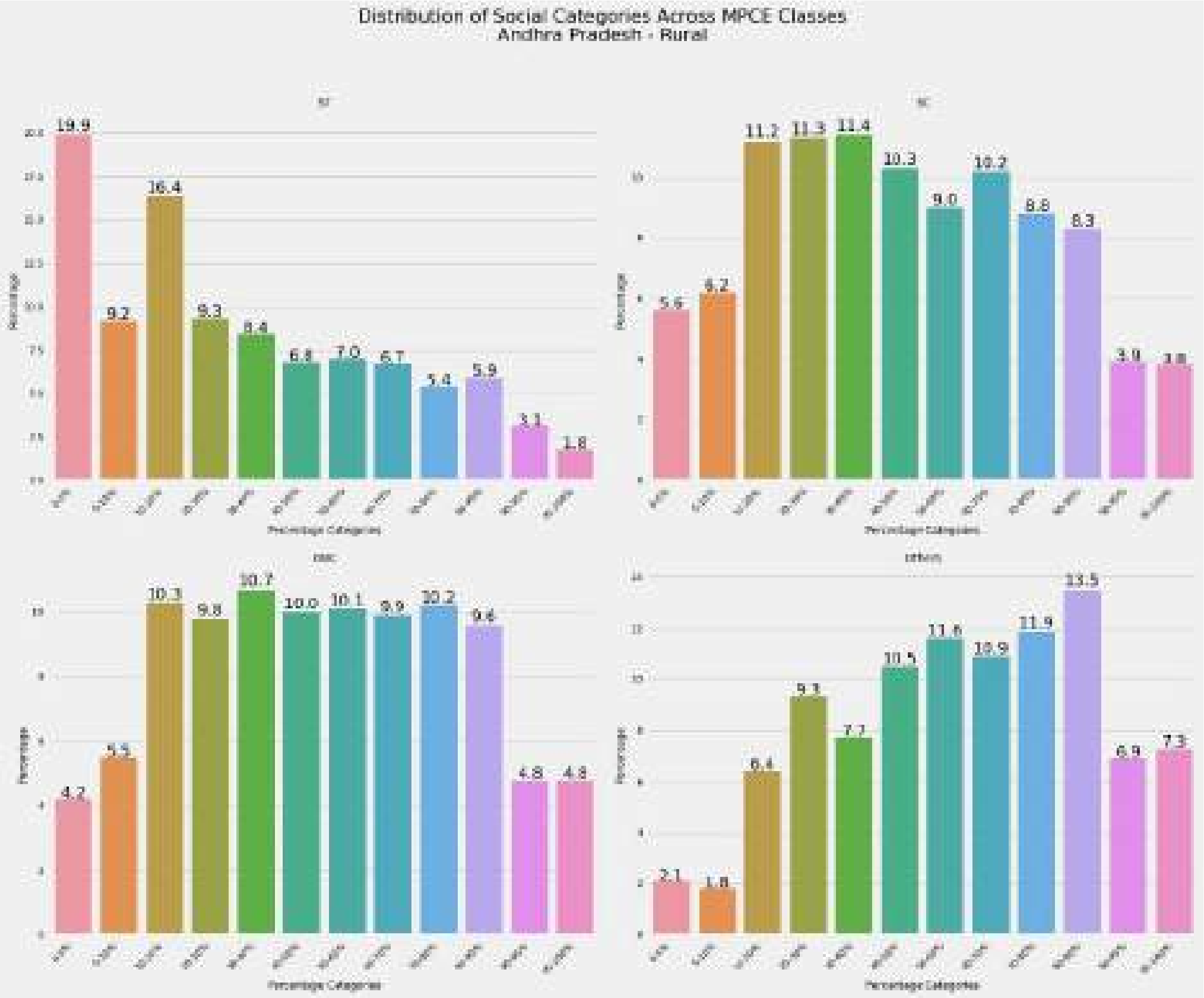
```
<ipython-input-198-296a4deb2f6f>:5: FutureWarning: Downcasting
behavior in `replace` is deprecated and will be removed in a future
version. To retain the old behavior, explicitly call
`result.infer_objects(copy=False)`. To opt-in to the future behavior,
set `pd.set_option('future.no_silent_downcasting', True)`
  data_a9['Others']=data_a9['Others'].replace('-',0.0)
<ipython-input-198-296a4deb2f6f>:6: FutureWarning: Downcasting
behavior in `replace` is deprecated and will be removed in a future
version. To retain the old behavior, explicitly call
`result.infer_objects(copy=False)`. To opt-in to the future behavior,
set `pd.set_option('future.no_silent_downcasting', True)`
  data_a9['ST']=data_a9['ST'].replace('-',0.0)
<ipython-input-198-296a4deb2f6f>:7: FutureWarning: Downcasting
behavior in `replace` is deprecated and will be removed in a future
version. To retain the old behavior, explicitly call
`result.infer_objects(copy=False)`. To opt-in to the future behavior,
set `pd.set_option('future.no_silent_downcasting', True)`
  data_a9['SC']=data_a9['SC'].replace('-',0.0)
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
58: RuntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:14
59: RuntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
```

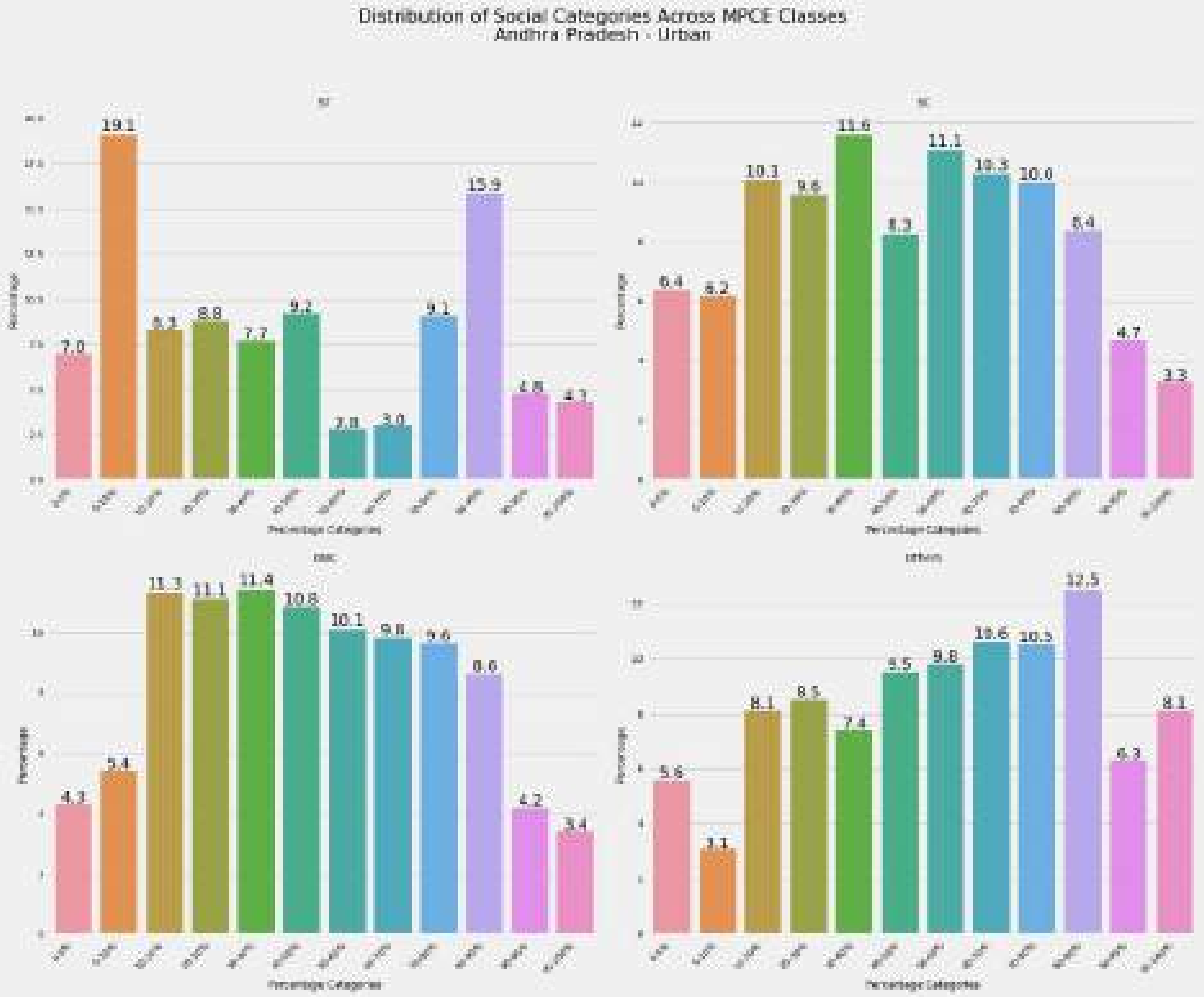
```
     col0            state                              col2        ST
SC  \
0   Rural  Andhra Pradesh                               0-5%     19.90
5.60
1   Rural  Andhra Pradesh                              5-10%      9.20
6.20
2   Rural  Andhra Pradesh                             10-20%     16.40
```

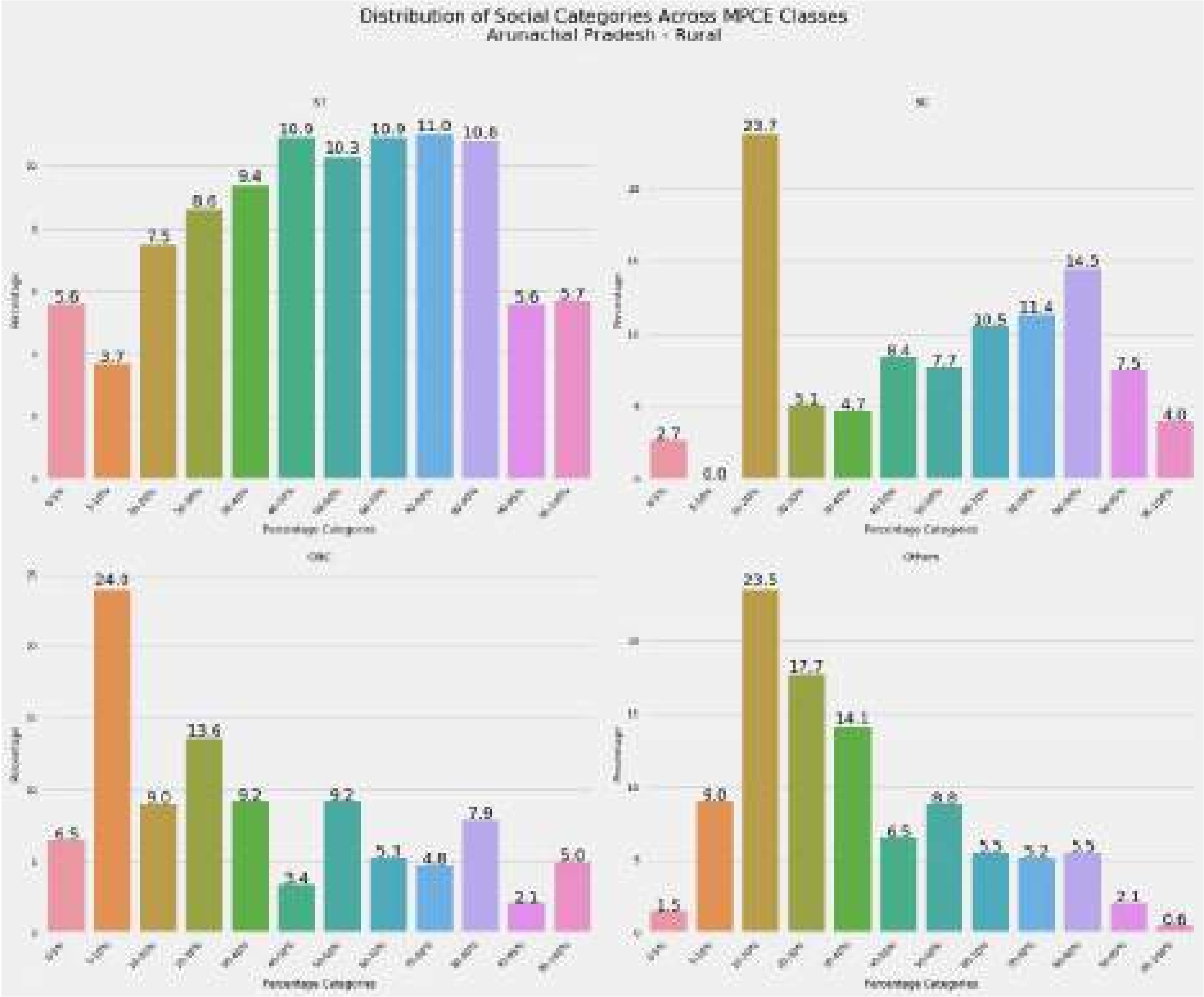
```
11.20
3   Rural  Andhra Pradesh                        20-30%      9.30
11.30
4   Rural  Andhra Pradesh                        30-40%      8.40
11.40
5   Rural  Andhra Pradesh                        40-50%      6.80
10.30
6   Rural  Andhra Pradesh                        50-60%      7.00
9.00
7   Rural  Andhra Pradesh                        60-70%      6.70
10.20
8   Rural  Andhra Pradesh                        70-80%      5.40
8.80
9   Rural  Andhra Pradesh                        80-90%      5.90
8.30
10  Rural  Andhra Pradesh                        90-95%      3.10
3.90
11  Rural  Andhra Pradesh                       95-100%      1.80
3.80
12  Rural  Andhra Pradesh                   All classes    100.00
100.00
13  Rural  Andhra Pradesh               Avg. MPCE (Rs.)   3772.44
4564.72
14  Rural  Andhra Pradesh  Estd. no. of households (00)   5762.00
22540.00
15  Rural  Andhra Pradesh      No. of sample households    532.00
1404.00
16  Rural  Andhra Pradesh     Estd. no. of persons (00)  19144.00
74085.00
17  Rural  Andhra Pradesh         No. of sample persons   1814.00
4827.00
18  Urban  Andhra Pradesh                          0-5%      7.00
6.40
19  Urban  Andhra Pradesh                         5-10%     19.10
6.20

          OBC   Others       All     col8    col9
0        4.20      2.1       5.0   3723.0   319.0
1        5.50      1.8       5.0   3701.0   281.0
2       10.30      6.4      10.0   7785.0   542.0
3        9.80      9.3      10.0   8260.0   551.0
4       10.70      7.7      10.0   8729.0   573.0
5       10.00     10.5      10.0   8837.0   586.0
6       10.10     11.6      10.0   9532.0   623.0
7        9.90     10.9      10.0  10055.0   648.0
8       10.20     11.9      10.0  10516.0   643.0
9        9.60     13.5      10.0  11880.0   705.0
10       4.80      6.9       5.0   5778.0   364.0
11       4.80      7.3       5.0   7019.0   410.0
```

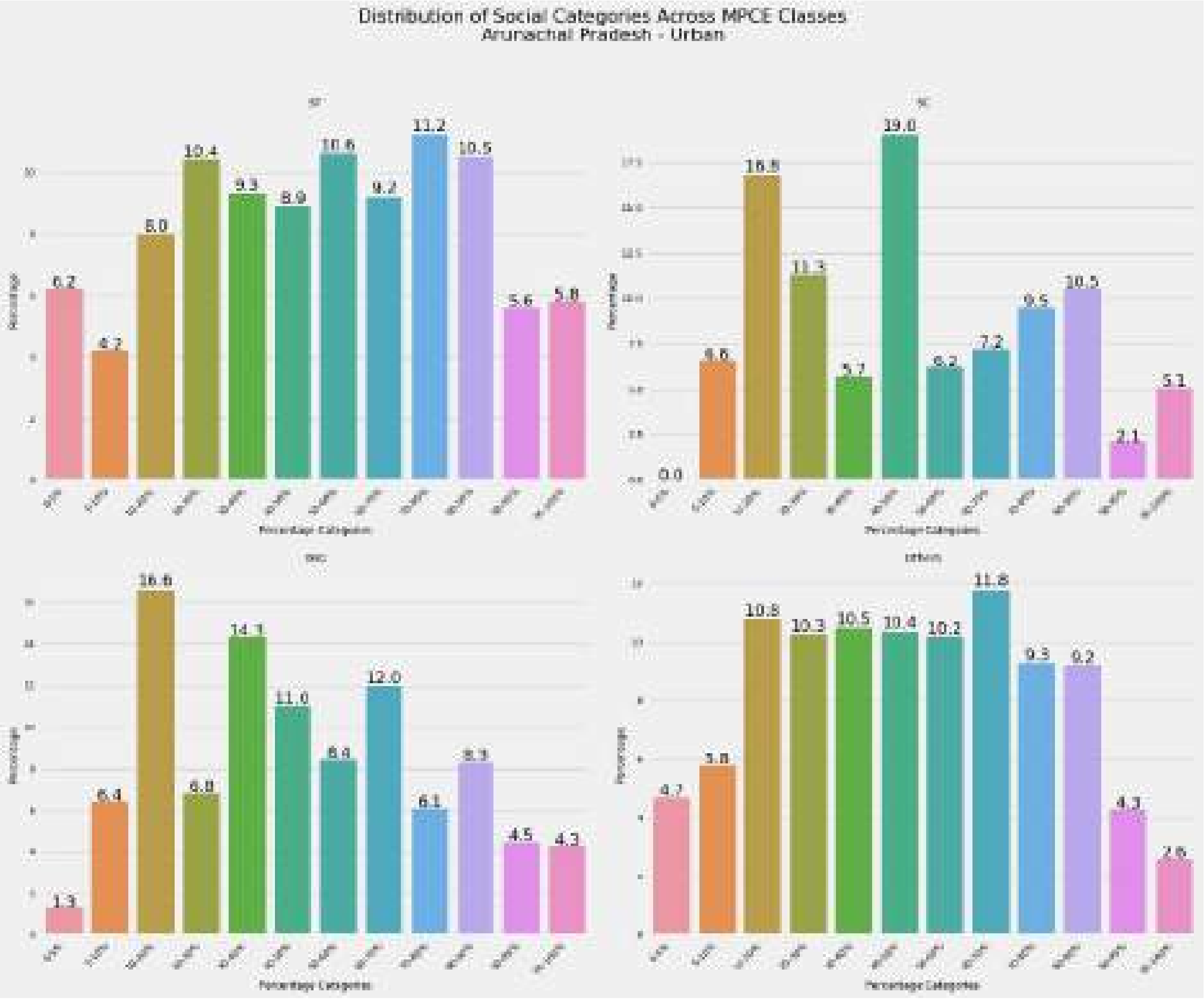
```
12     100.00    100.0     100.0  95813.0  6245.0
13    4823.68   5549.8    4870.3      NaN     NaN
14   43905.00  23606.0   95813.0      NaN     NaN
15    2870.00   1439.0    6245.0      NaN     NaN
16  147781.00  74385.0  315396.0      NaN     NaN
17    9750.00   4516.0   20907.0      NaN     NaN
18       4.30      5.6       5.0   1657.0   135.0
19       5.40      3.1       5.0   1976.0   176.0
```

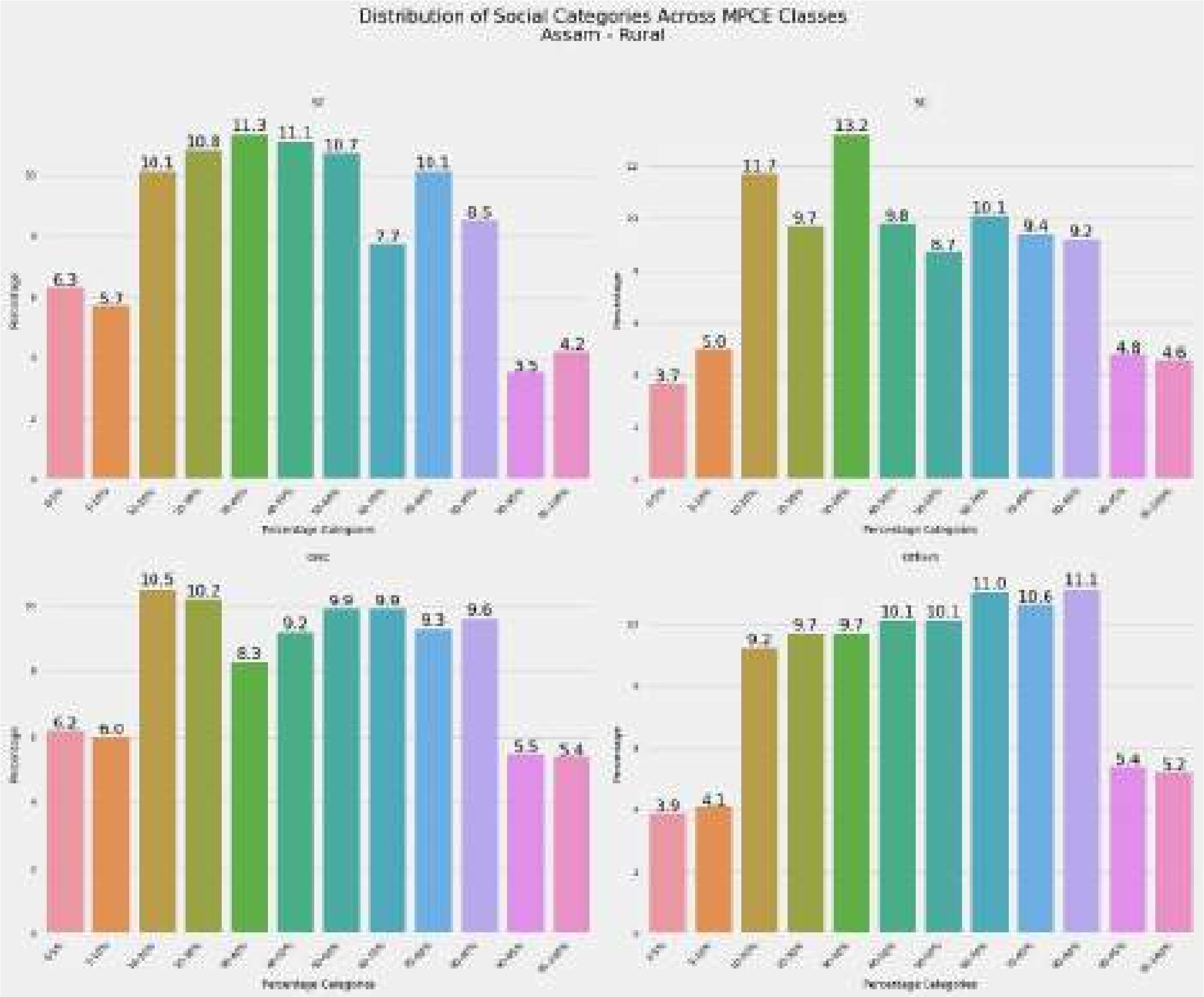
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# List of percentage categories to include
pct_categories = ['0-5%', '5-10%', '10-20%', '20-30%', '30-40%', '40-
50%',
                  '50-60%', '60-70%', '70-80%', '80-90%', '90-95%',
'95-100%']

# List of social category columns to plot
social_columns = ['ST', 'SC', 'OBC', 'Others']

# Get unique states
states = data_a9['state'].unique()

# Process state by state
for state in states:
    # Process Rural and Urban for each state
    for area in ['Rural', 'Urban']:
        # Filter data for current state and area
        current_data = data_a9[(data_a9['state'] == state) &
                               (data_a9['col0'] == area)]

        # Filter rows for percentage categories we want
        current_data =
current_data[current_data['col2'].isin(pct_categories)]

        if not current_data.empty:
            # Create subplots for each social category
            fig, axes = plt.subplots(2, 2, figsize=(15, 12))
            axes = axes.flatten()

            # Create a bar plot for each social category
            for idx, column in enumerate(social_columns):
                # Create bar plot
                sns.barplot(data=current_data, x='col2', y=column,
ax=axes[idx])

                # Customize subplot
                axes[idx].set_title(f'{column}', pad=10)
                axes[idx].set_xlabel('Percentage Categories')
                axes[idx].set_ylabel('Percentage')

                # Rotate x-axis labels
                axes[idx].set_xticklabels(pct_categories, rotation=45,

                                        horizontalalignment='right')

                # Add value labels on bars
                for i, v in enumerate(current_data[column]):
                    if pd.notna(v):  # Check if value is not NaN
                        axes[idx].text(i, v + v*0.01, f'{v:.1f}',
ha='center')

            # Add main title for the state and area
            fig.suptitle(f'Distribution of Social Categories Across
MPCE Classes\n{state} - {area}',
                        fontsize=16, y=1.02)

            # Adjust layout
            plt.tight_layout()

            # Show plot
            plt.show()

            # Add a small pause between plots (optional)
            plt.pause(0.5)
```

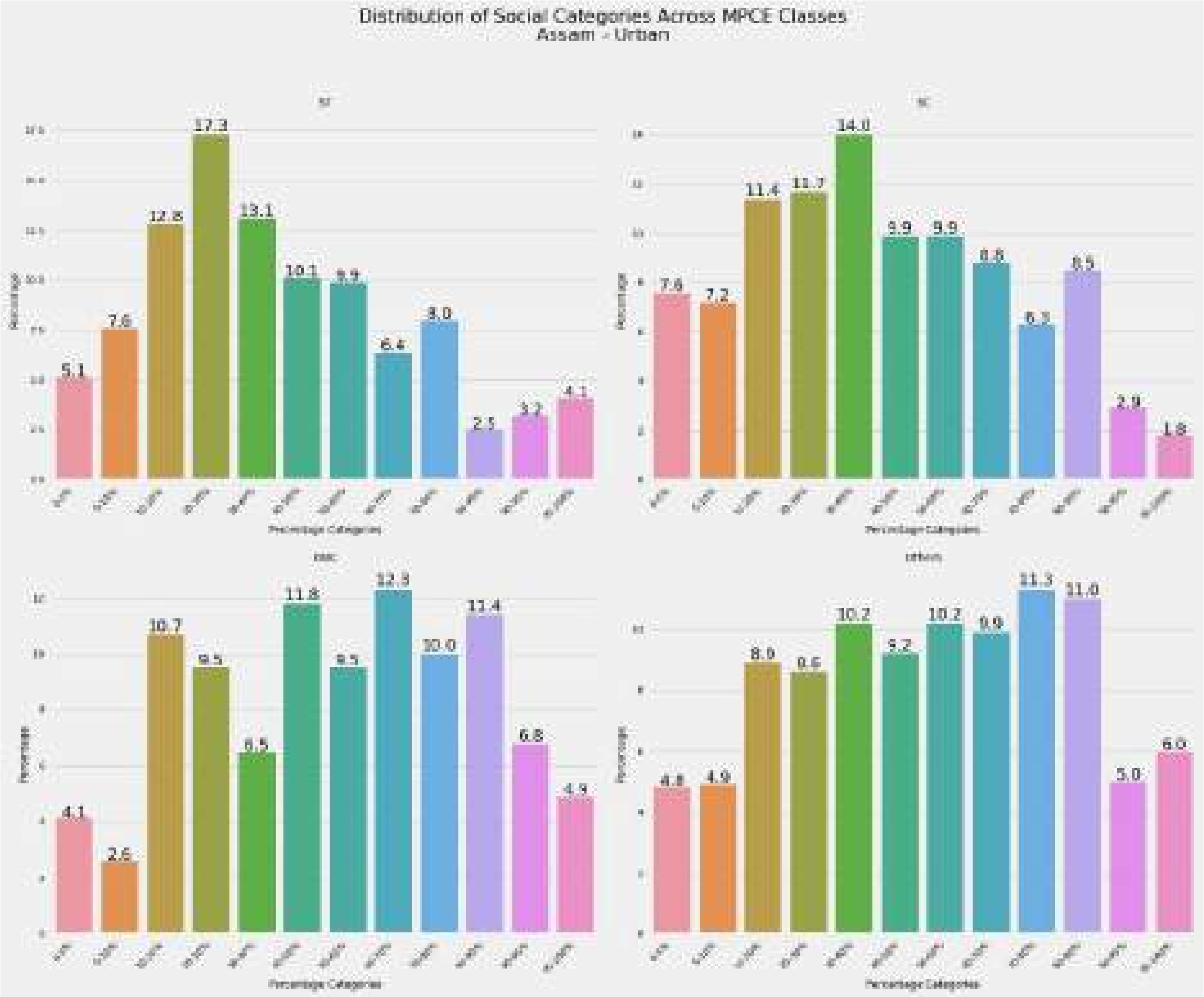

Distribution of Social Categories Across MPCE Classes
Andhra Pradesh - Rural
ST
19.9
9.2
16.4
9.3
8.4
6.8
7.0
6.7
5.4
5.9
3.1
1.8
SC
5.6
6.2
11.2
11.3
11.4
10.3
9.0
10.2
8.8
8.3
3.9
3.8
OBC
4.2
5.5
10.3
9.8
10.7
10.0
10.1
9.9
10.2
9.6
4.8
4.8
Others
2.1
1.8
6.4
9.3
7.7
10.5
11.6
10.9
11.9
13.5
6.9
7.3
Percentage
Percentage Categories

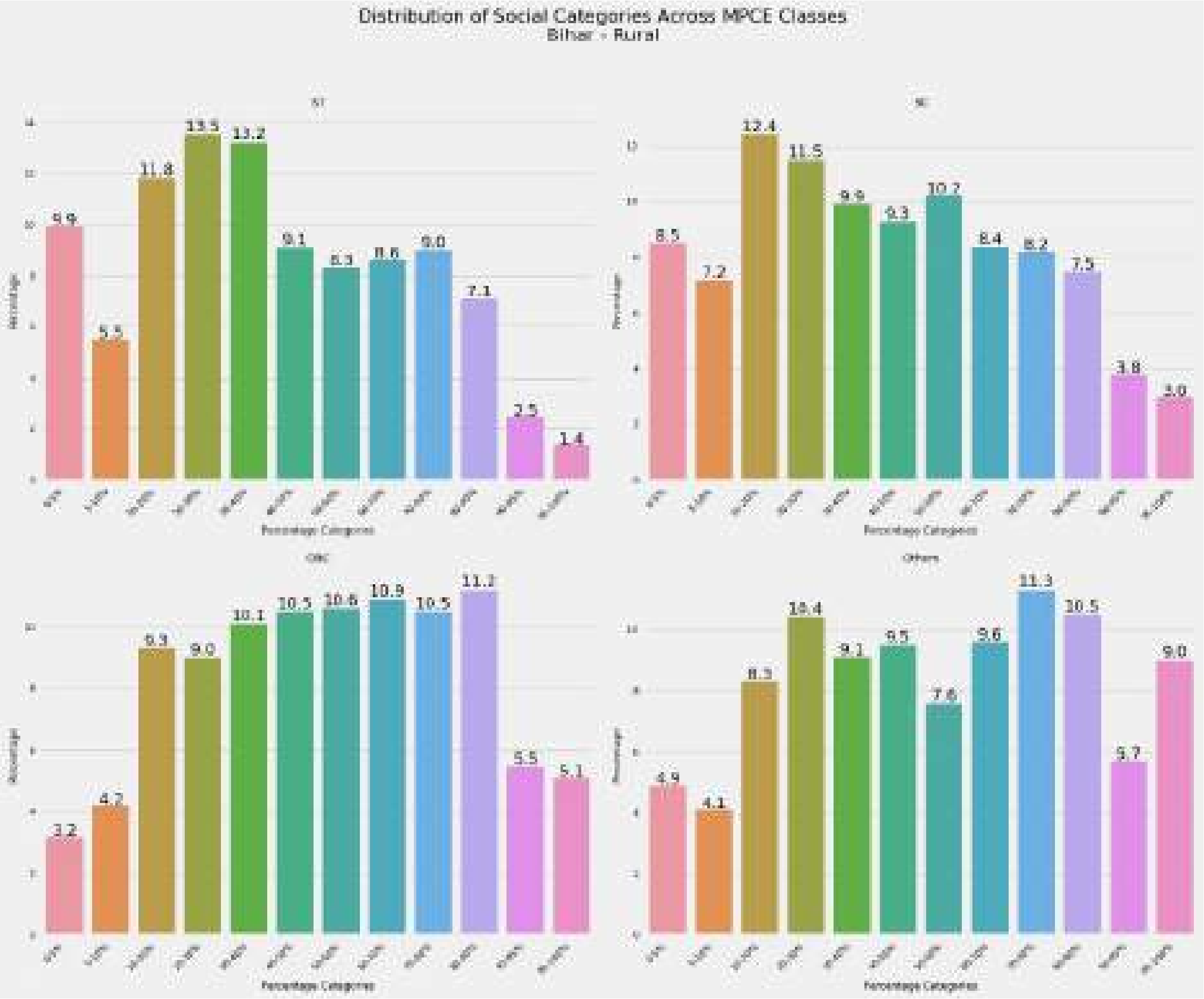

Distribution of Social Categories Across MPCE Classes
Andhra Pradesh - Urban
ST
SC
OBC
Others
Percentage
Percentage Categories
7.0
19.1
8.3
8.8
7.7
9.2
2.8
3.0
9.1
15.9
4.8
4.3
6.4
6.2
10.1
9.6
11.6
8.3
11.1
10.3
10.0
8.4
4.7
3.3
4.3
5.4
11.3
11.1
11.4
10.8
10.1
9.8
9.6
8.6
4.2
3.4
5.6
3.1
8.1
8.5
7.4
9.5
9.8
10.6
10.5
12.5
6.3
8.1

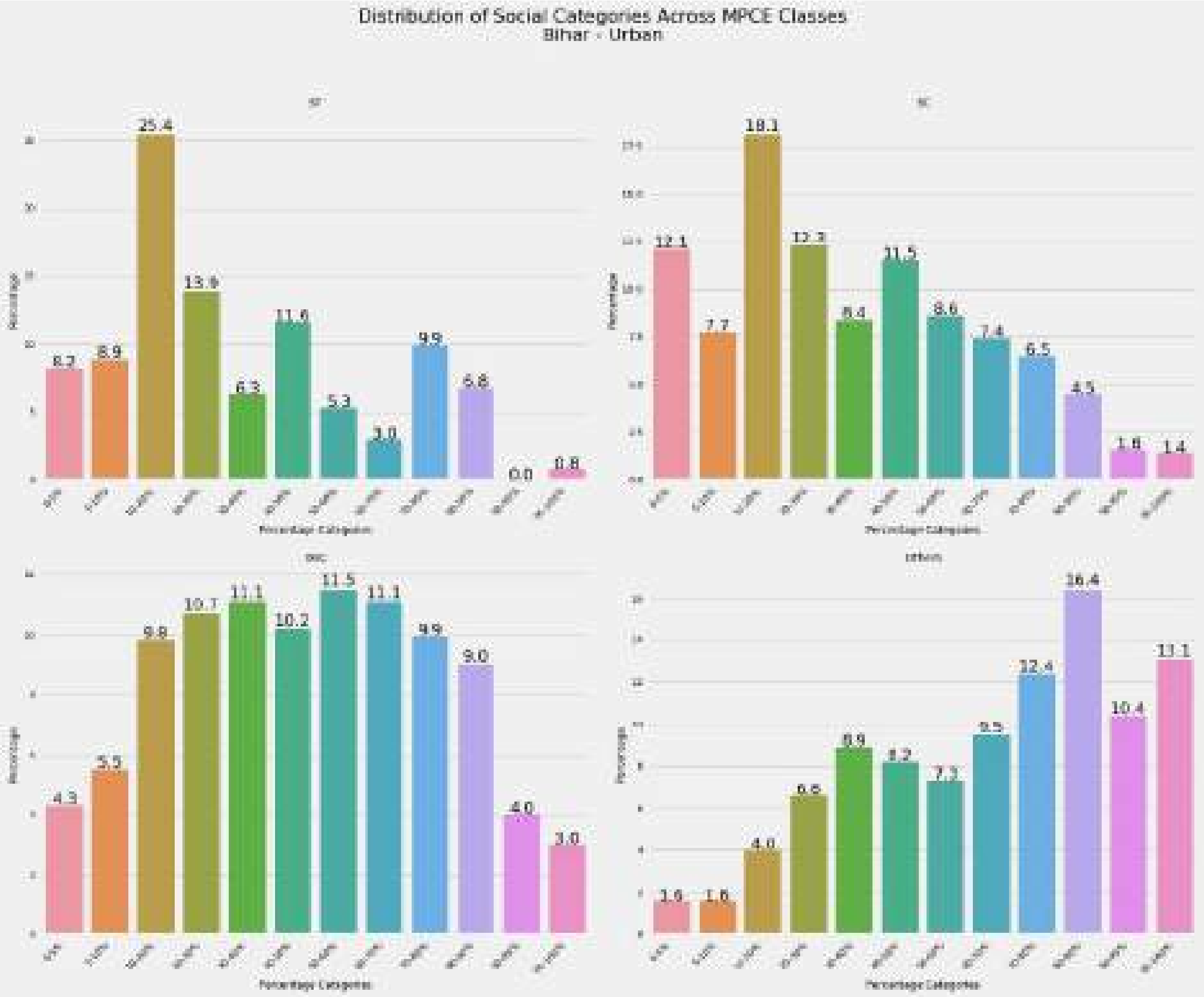

Distribution of Social Categories Across MPCE Classes
Arunachal Pradesh - Rural
ST
Percentage
5.6
3.7
7.5
8.6
9.4
10.9
10.3
10.9
11.0
10.8
5.6
5.7
Percentage Categories
SC
Percentage
2.7
0.0
23.7
5.1
4.7
8.4
7.7
10.5
11.4
14.5
7.5
4.0
Percentage Categories
OBC
Percentage
6.5
24.3
9.0
13.6
9.2
3.4
9.2
5.3
4.8
7.9
2.1
5.0
Percentage Categories
Others
Percentage
1.5
9.0
23.5
17.7
14.1
6.5
8.8
5.5
5.2
5.5
2.1
0.6
Percentage Categories

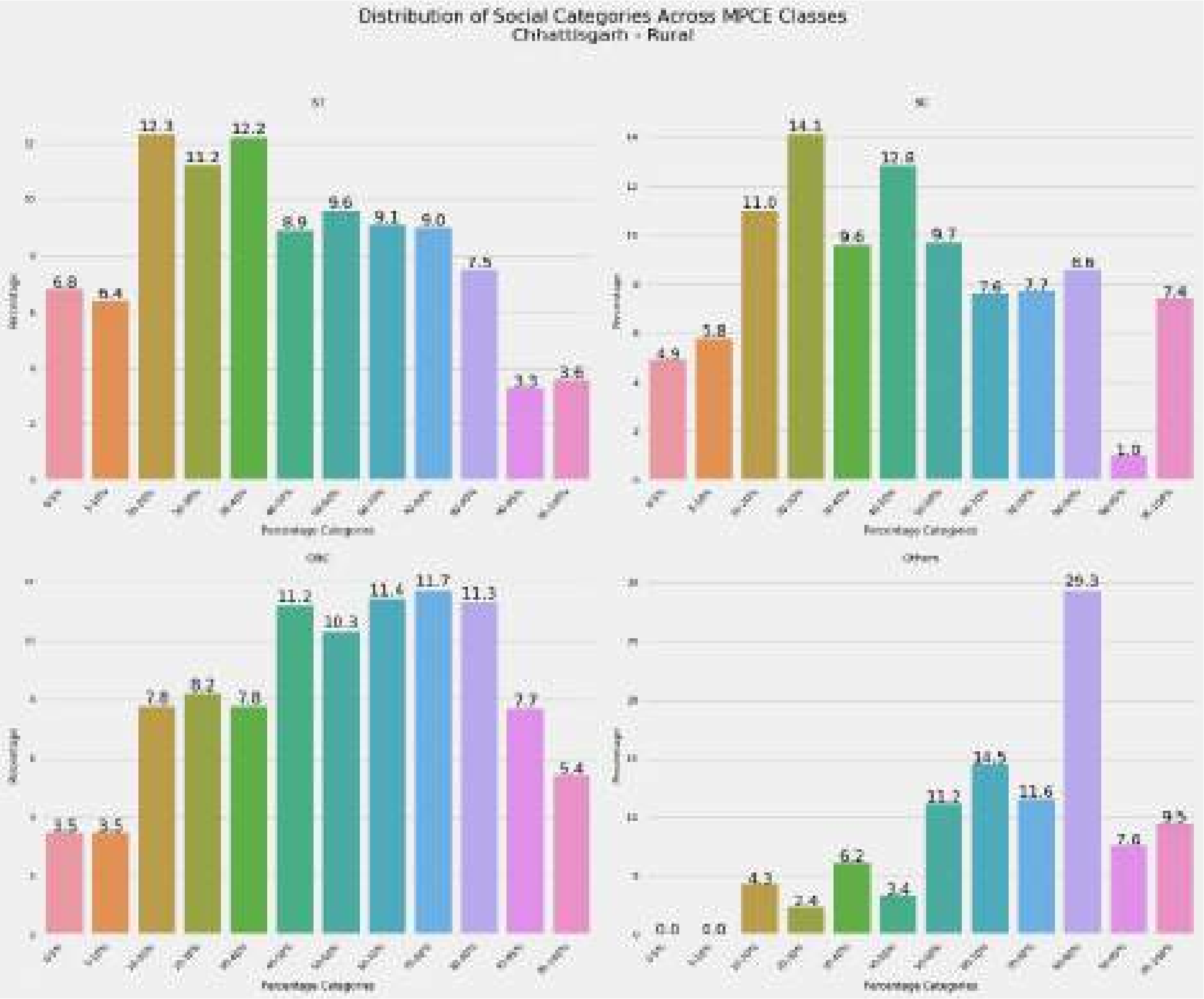

Distribution of Social Categories Across MPCE Classes
Arunachal Pradesh - Urban
ST
Percentage
Percentage Categories
6.2
4.2
8.0
10.4
9.3
8.9
10.6
9.2
11.2
10.5
5.6
5.8
SC
Percentage
Percentage Categories
0.0
6.6
16.8
11.3
5.7
19.0
6.2
7.2
9.5
10.5
2.1
5.1
OBC
Percentage
Percentage Categories
1.3
6.4
16.6
6.8
14.3
11.0
8.4
12.0
6.1
8.3
4.5
4.3
Others
Percentage
Percentage Categories
4.7
5.8
10.8
10.3
10.5
10.4
10.2
11.8
9.3
9.2
4.3
2.6

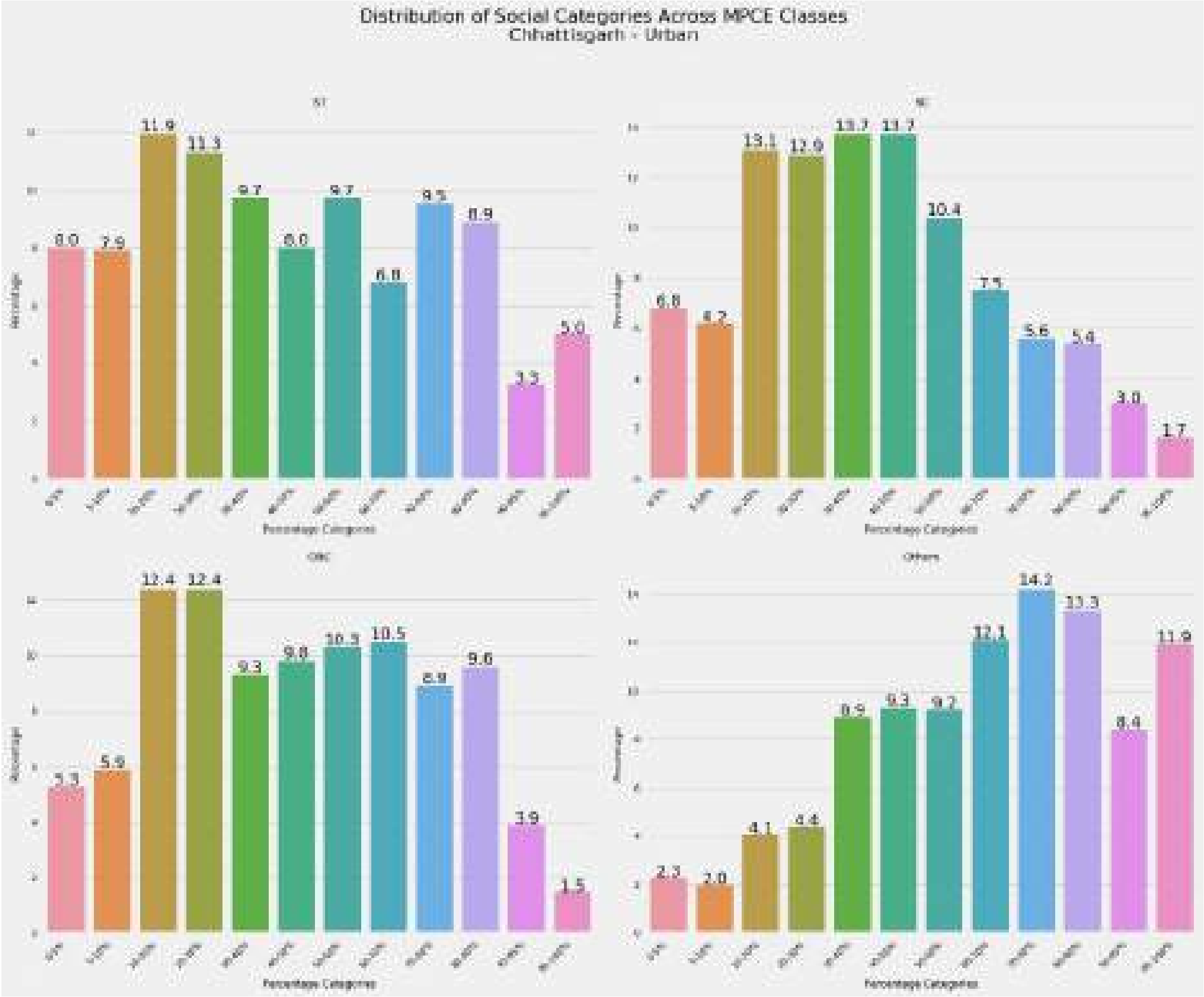

Distribution of Social Categories Across MPCE Classes
Assam - Rural
ST
Percentage
Percentage Categories
6.3
5.7
10.1
10.8
11.3
11.1
10.7
7.7
10.1
8.5
3.5
4.2
SC
Percentage
Percentage Categories
3.7
5.0
11.7
9.7
13.2
9.8
8.7
10.1
9.4
9.2
4.8
4.6
OBC
Percentage
Percentage Categories
6.2
6.0
10.5
10.2
8.3
9.2
9.9
9.9
9.3
9.6
5.5
5.4
Others
Percentage
Percentage Categories
3.9
4.1
9.2
9.7
9.7
10.1
10.1
11.0
10.6
11.1
5.4
5.2

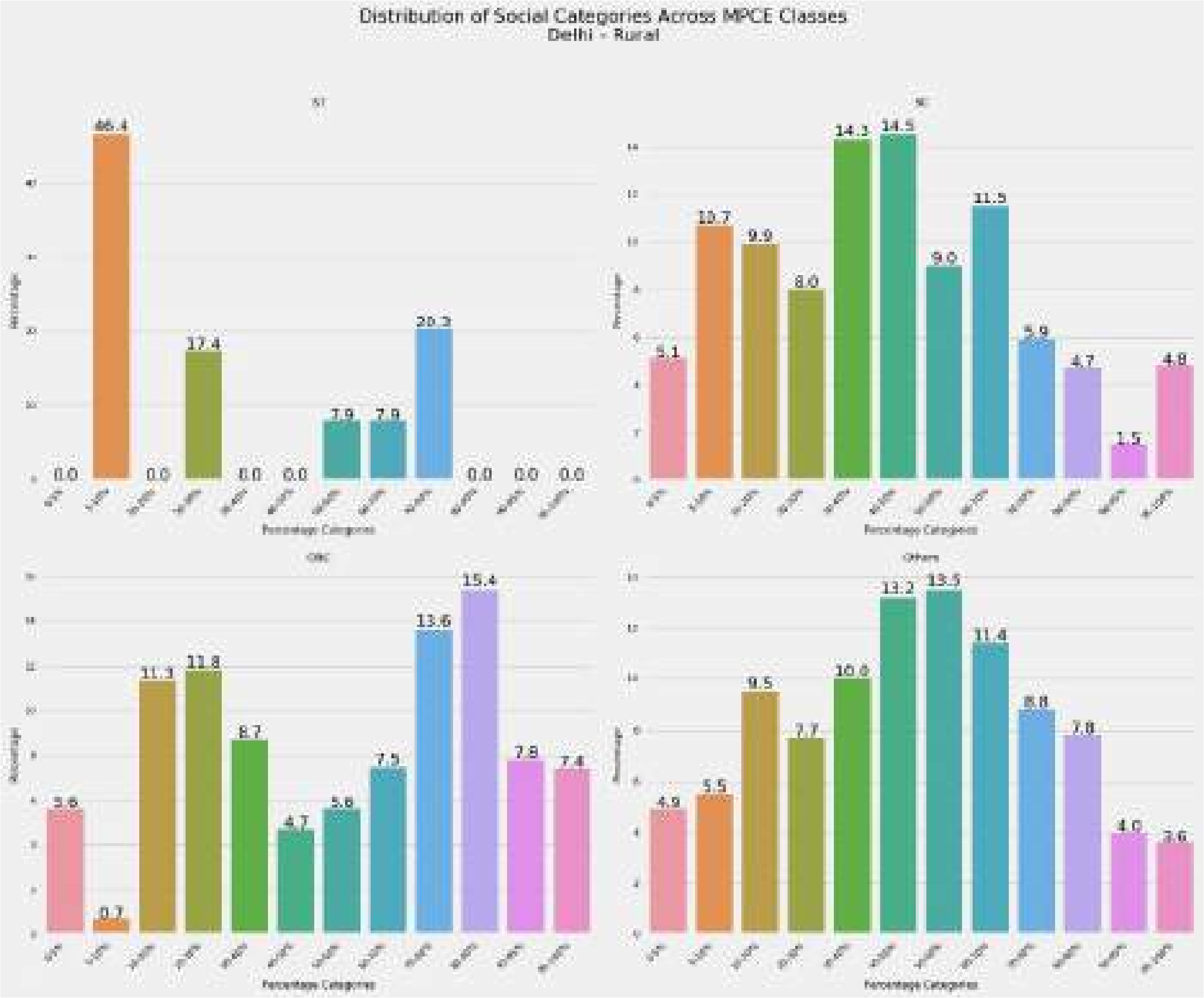
Distribution of Social Categories Across MPCE Classes
Assam - Urban
ST
Percentage
Percentage Categories
5.1
7.6
12.8
17.3
13.1
10.1
9.9
6.4
8.0
2.5
3.2
4.1
SC
Percentage
Percentage Categories
7.6
7.2
11.4
11.7
14.0
9.9
9.9
8.8
6.3
8.5
2.9
1.8
OBC
Percentage
Percentage Categories
4.1
2.6
10.7
9.5
6.5
11.8
9.5
12.3
10.0
11.4
6.8
4.9
Others
Percentage
Percentage Categories
4.8
4.9
8.9
8.6
10.2
9.2
10.2
9.9
11.3
11.0
5.0
6.0

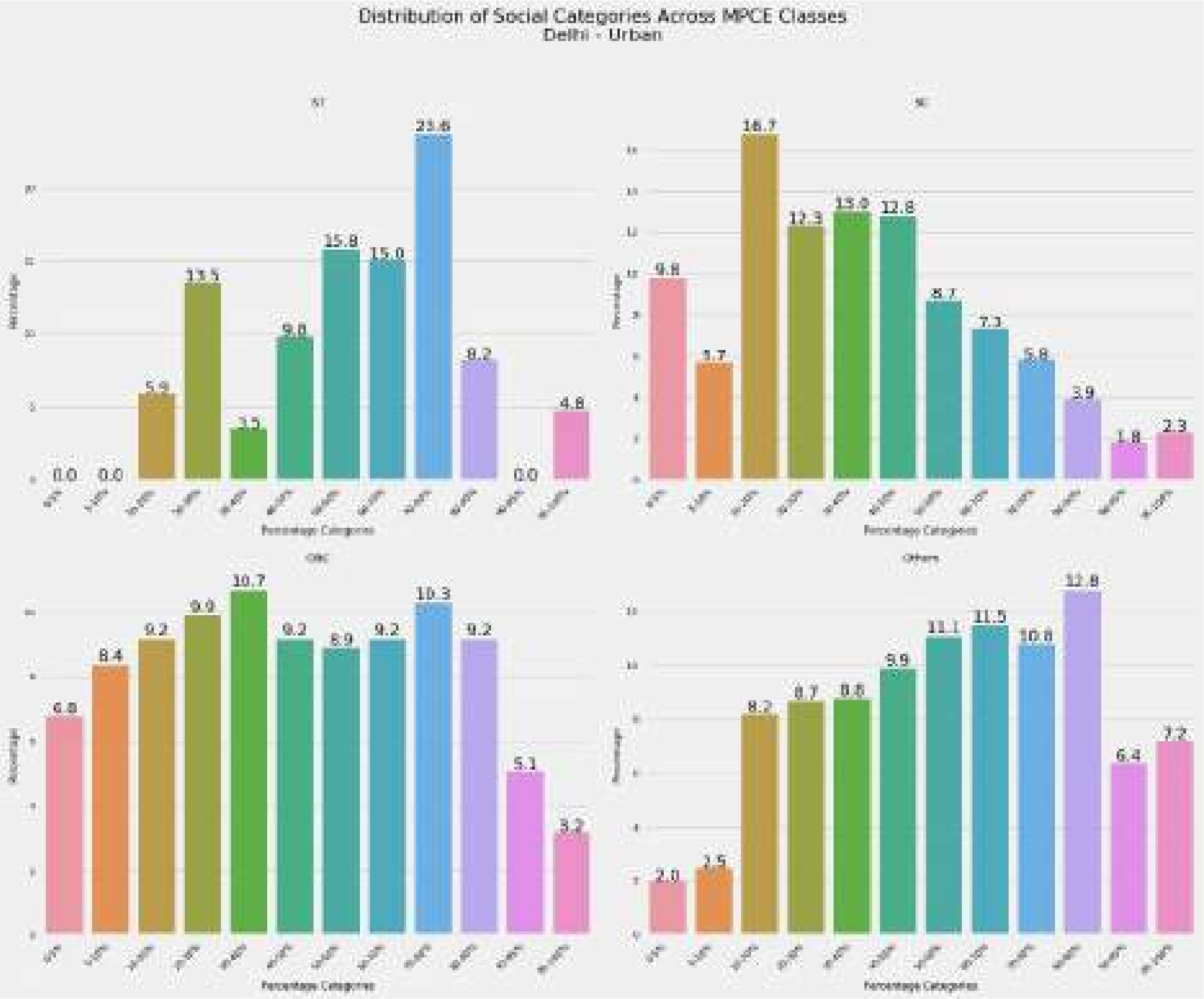

Distribution of Social Categories Across MPCE Classes
Bihar - Rural
ST
Percentage
9.9
5.5
11.8
13.5
13.2
9.1
8.3
8.6
9.0
7.1
2.5
1.4
Percentage Categories
SC
Percentage
8.5
7.2
12.4
11.5
9.9
9.3
10.2
8.4
8.2
7.5
3.8
3.0
Percentage Categories
OBC
Percentage
3.2
4.2
9.3
9.0
10.1
10.5
10.6
10.9
10.5
11.2
5.5
5.1
Percentage Categories
Others
Percentage
4.9
4.1
8.3
10.4
9.1
9.5
7.6
9.6
11.3
10.5
5.7
9.0
Percentage Categories

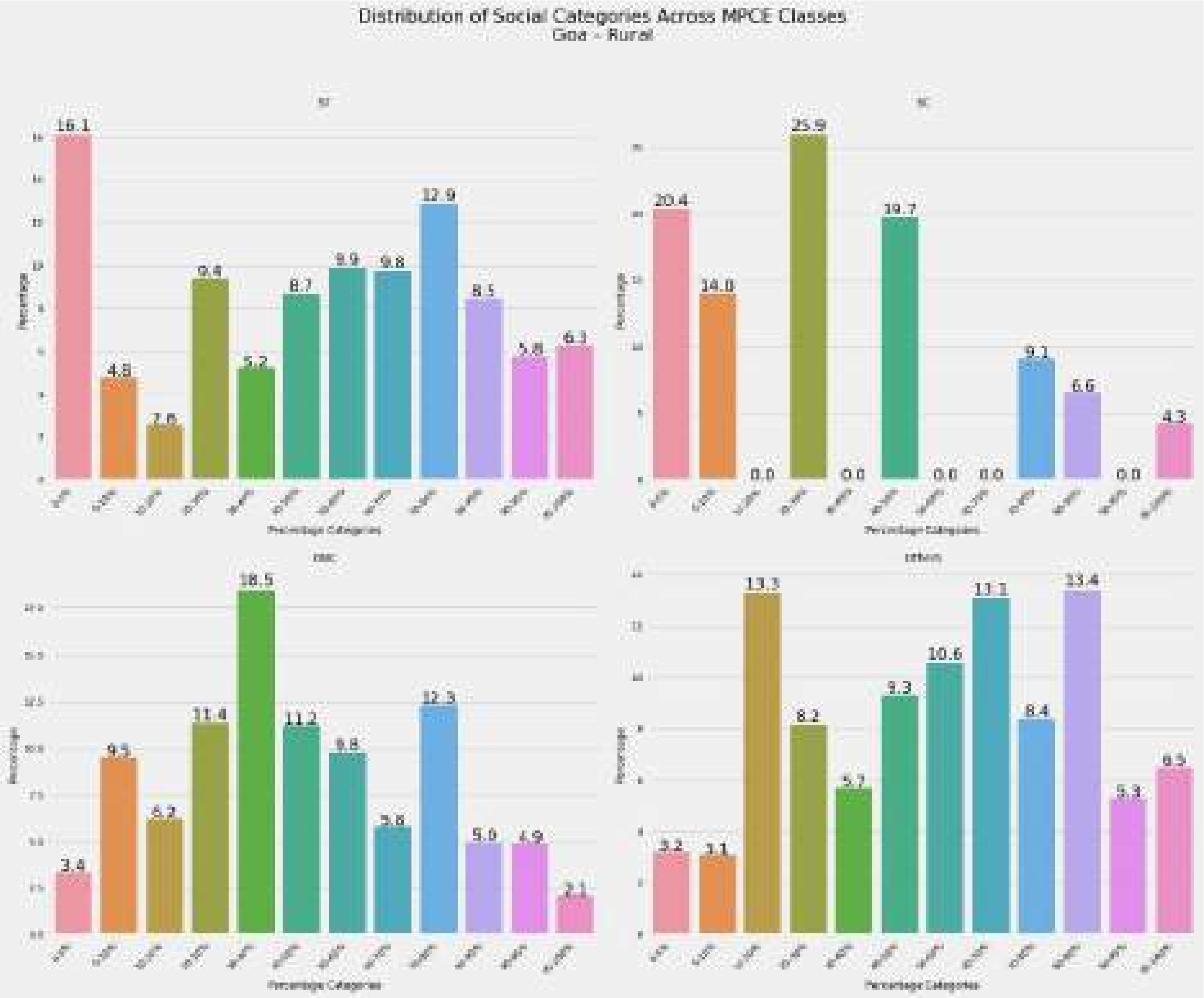

Distribution of Social Categories Across MPCE Classes
Bihar - Urban
ST
Percentage
8.2
8.9
25.4
13.9
6.3
11.6
5.3
3.0
9.9
6.8
0.0
0.8
Percentage Categories
SC
Percentage
12.1
7.7
18.1
12.3
8.4
11.5
8.6
7.4
6.5
4.5
1.6
1.4
Percentage Categories
OBC
Percentage
4.3
5.5
9.8
10.7
11.1
10.2
11.5
11.1
9.9
9.0
4.0
3.0
Percentage Categories
Others
Percentage
1.6
1.6
4.0
6.6
8.9
8.2
7.3
9.5
12.4
16.4
10.4
13.1
Percentage Categories

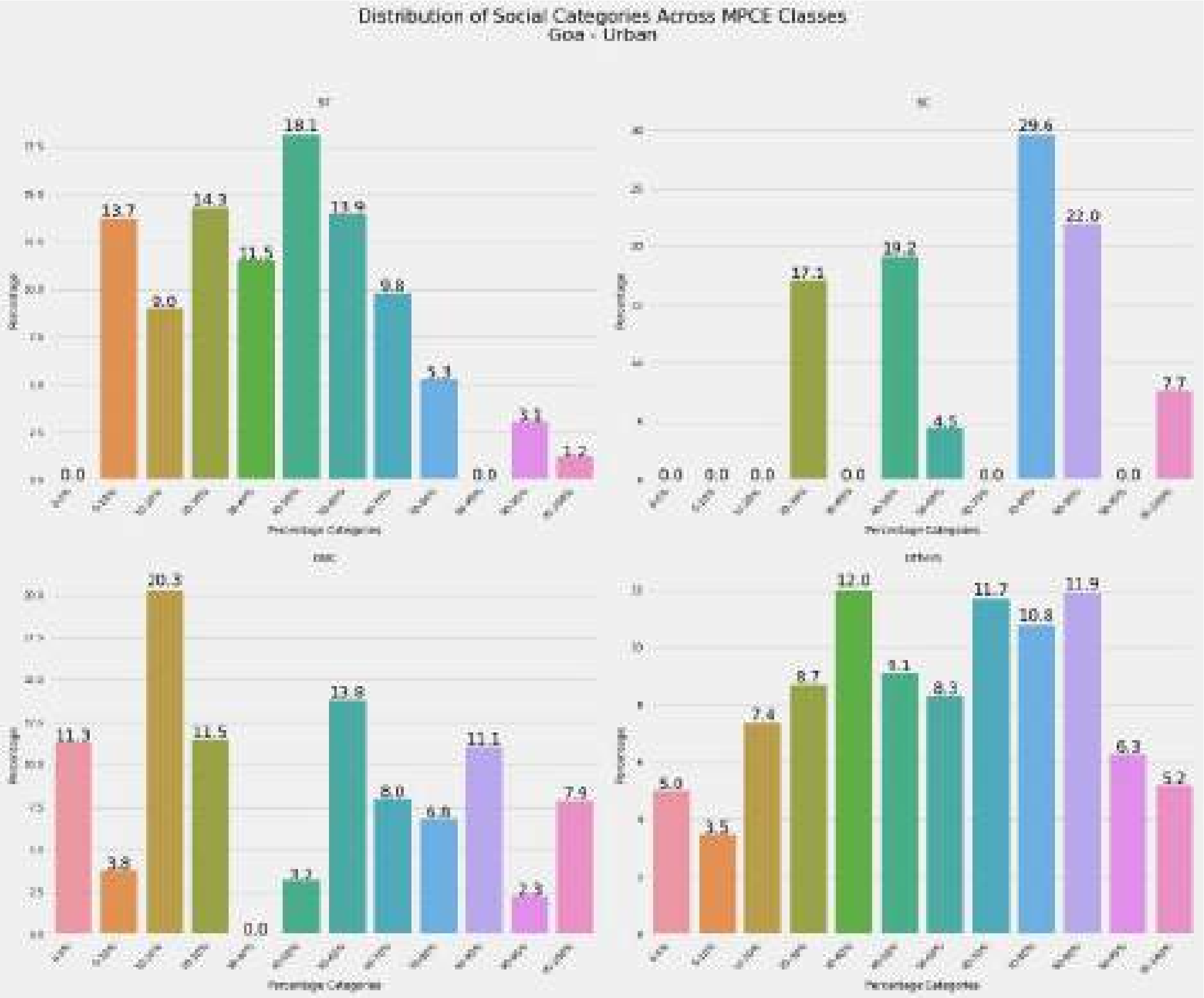

Distribution of Social Categories Across MPCE Classes
Chhattisgarh - Rural
ST
6.8
6.4
12.3
11.2
12.2
8.9
9.6
9.1
9.0
7.5
3.3
3.6
Percentage
Percentage Categories
SC
4.9
5.8
11.0
14.1
9.6
12.9
9.7
7.6
7.7
8.6
1.0
7.4
Percentage
Percentage Categories
OBC
3.5
3.5
7.8
8.2
7.8
11.2
10.3
11.4
11.7
11.3
7.7
5.4
Percentage
Percentage Categories
Others
0.0
0.0
4.3
2.4
6.2
3.4
11.2
14.5
11.6
29.3
7.6
9.5
Percentage
Percentage Categories

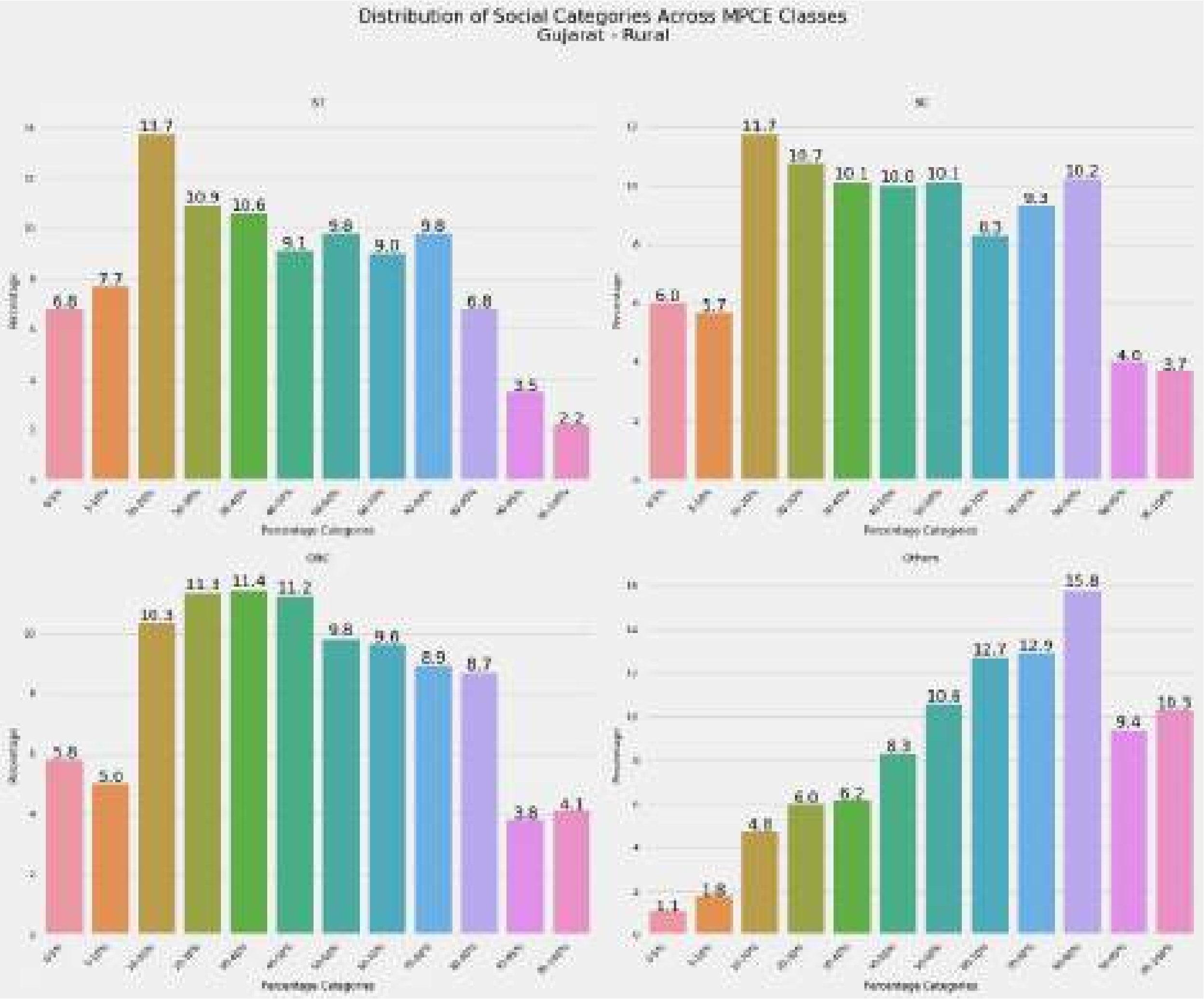
Distribution of Social Categories Across MPCE Classes
Chhattisgarh - Urban
ST
Percentage
8.0
7.9
11.9
11.3
9.7
8.0
9.7
6.8
9.5
8.9
3.3
5.0
Percentage Categories
SC
Percentage
6.8
6.2
13.1
12.9
13.7
13.7
10.4
7.5
5.6
5.4
3.0
1.7
Percentage Categories
OBC
Percentage
5.3
5.9
12.4
12.4
9.3
9.8
10.3
10.5
8.9
9.6
3.9
1.5
Percentage Categories
Others
Percentage
2.3
2.0
4.1
4.4
8.9
9.3
9.2
12.1
14.2
13.3
8.4
11.9
Percentage Categories

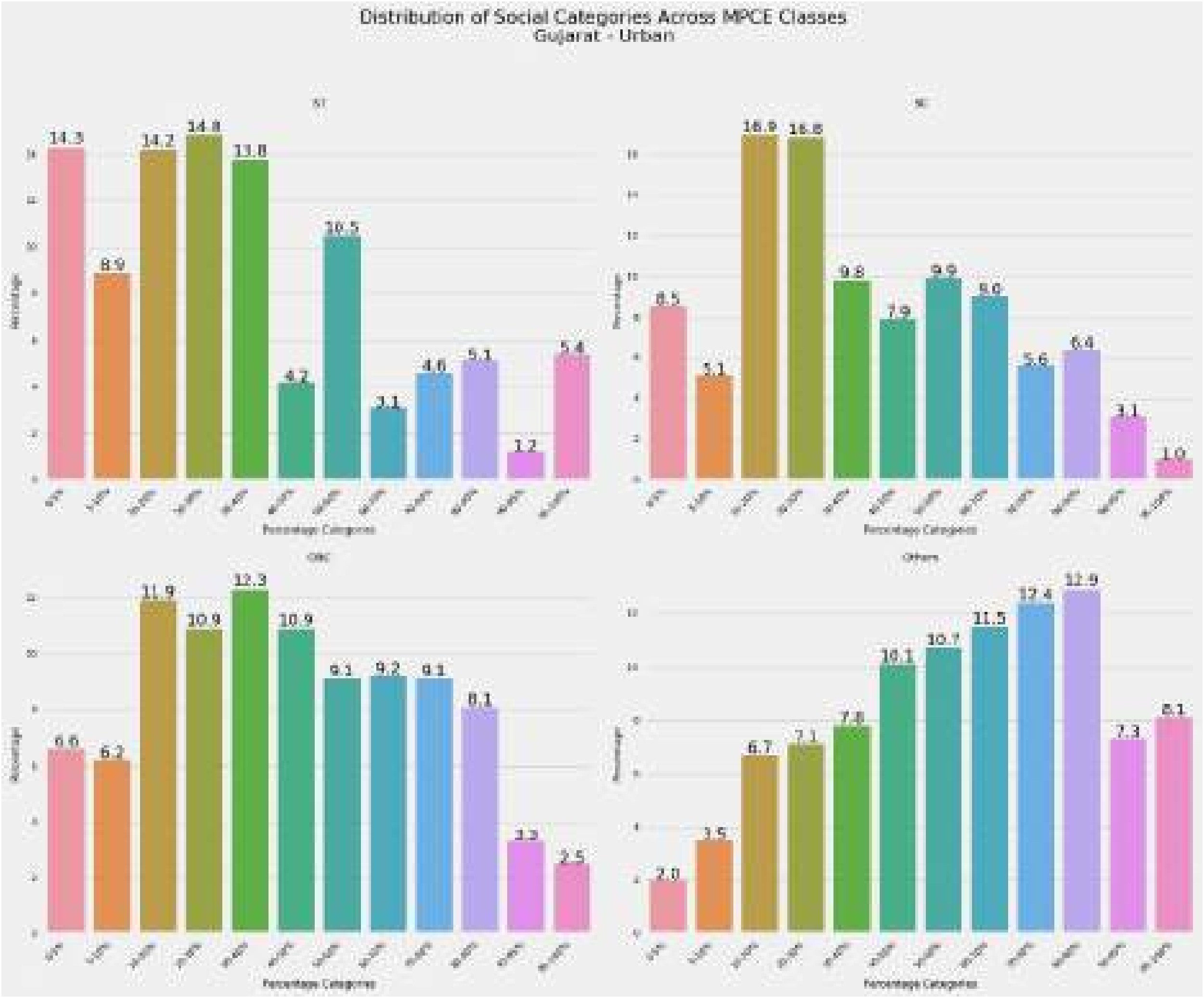

Distribution of Social Categories Across MPCE Classes
Delhi - Rural
ST
0.0
46.4
0.0
17.4
0.0
0.0
7.9
7.9
20.3
0.0
0.0
0.0
Percentage
Percentage Categories
SC
5.1
10.7
9.9
8.0
14.3
14.5
9.0
11.5
5.9
4.7
1.5
4.8
Percentage
Percentage Categories
OBC
3.6
0.7
11.3
11.8
8.7
4.7
5.8
7.5
13.6
15.4
7.8
7.4
Percentage
Percentage Categories
Others
4.9
5.5
9.5
7.7
10.0
13.2
13.5
11.4
8.8
7.8
4.0
3.6
Percentage
Percentage Categories

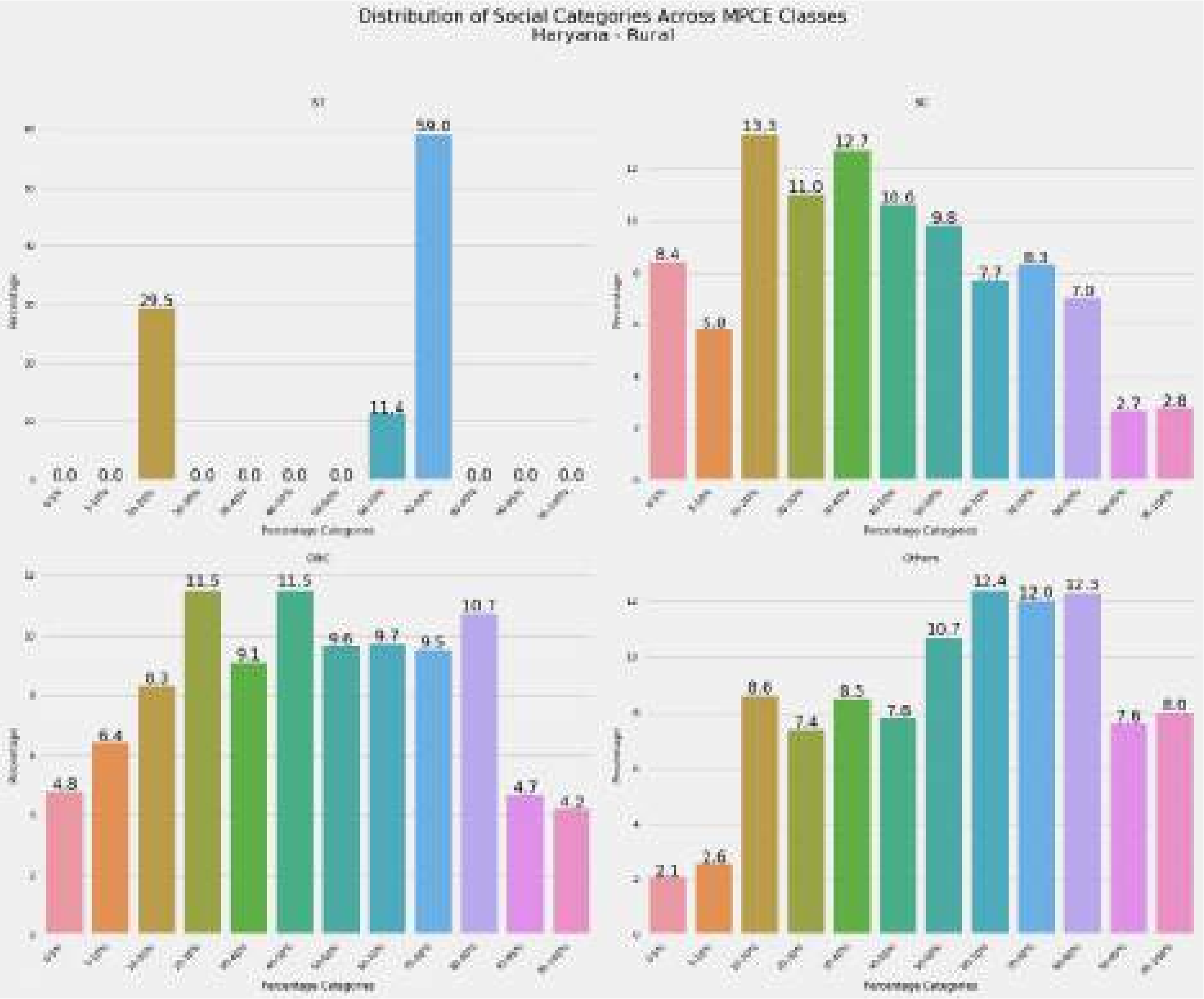

Distribution of Social Categories Across MPCE Classes
Delhi - Urban
ST
0.0
0.0
5.9
13.5
3.5
9.8
15.8
15.0
23.6
8.2
0.0
4.8
Percentage
Percentage Categories
SC
9.8
5.7
16.7
12.3
13.0
12.8
8.7
7.3
5.8
3.9
1.8
2.3
Percentage
Percentage Categories
OBC
6.8
8.4
9.2
9.9
10.7
9.2
8.9
9.2
10.3
9.2
5.1
3.2
Percentage
Percentage Categories
Others
2.0
2.5
8.2
8.7
8.8
9.9
11.1
11.5
10.8
12.8
6.4
7.2
Percentage
Percentage Categories

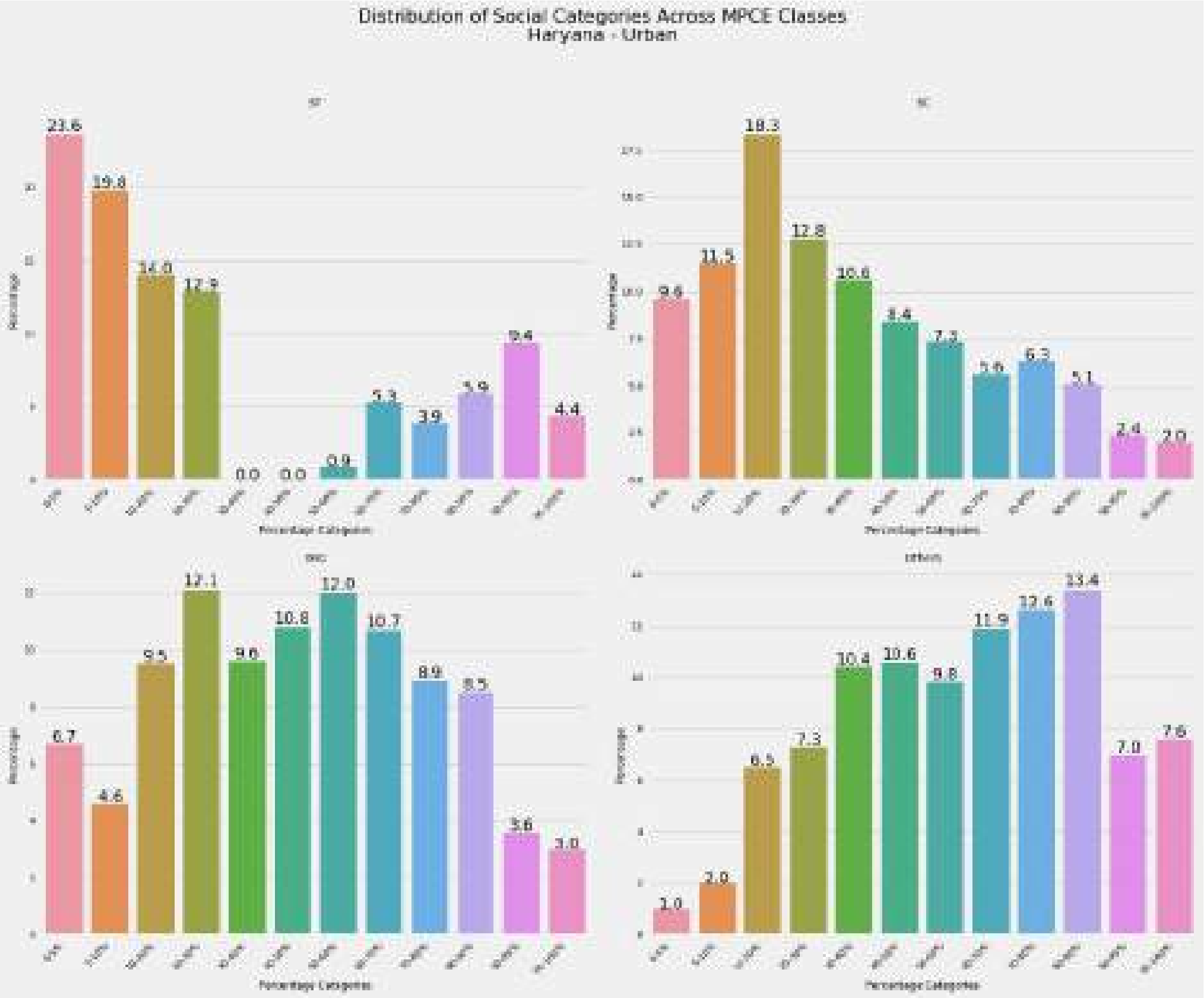

Distribution of Social Categories Across MPCE Classes
Goa - Rural
ST
16.1
4.8
2.6
9.4
5.2
8.7
9.9
9.8
12.9
8.5
5.8
6.3
Percentage
Percentage Categories
SC
20.4
14.0
0.0
25.9
0.0
19.7
0.0
0.0
9.1
6.6
0.0
4.3
Percentage
Percentage Categories
OBC
3.4
9.5
6.2
11.4
18.5
11.2
9.8
5.8
12.3
5.0
4.9
2.1
Percentage
Percentage Categories
Others
3.2
3.1
13.3
8.2
5.7
9.3
10.6
13.1
8.4
13.4
5.3
6.5
Percentage
Percentage Categories

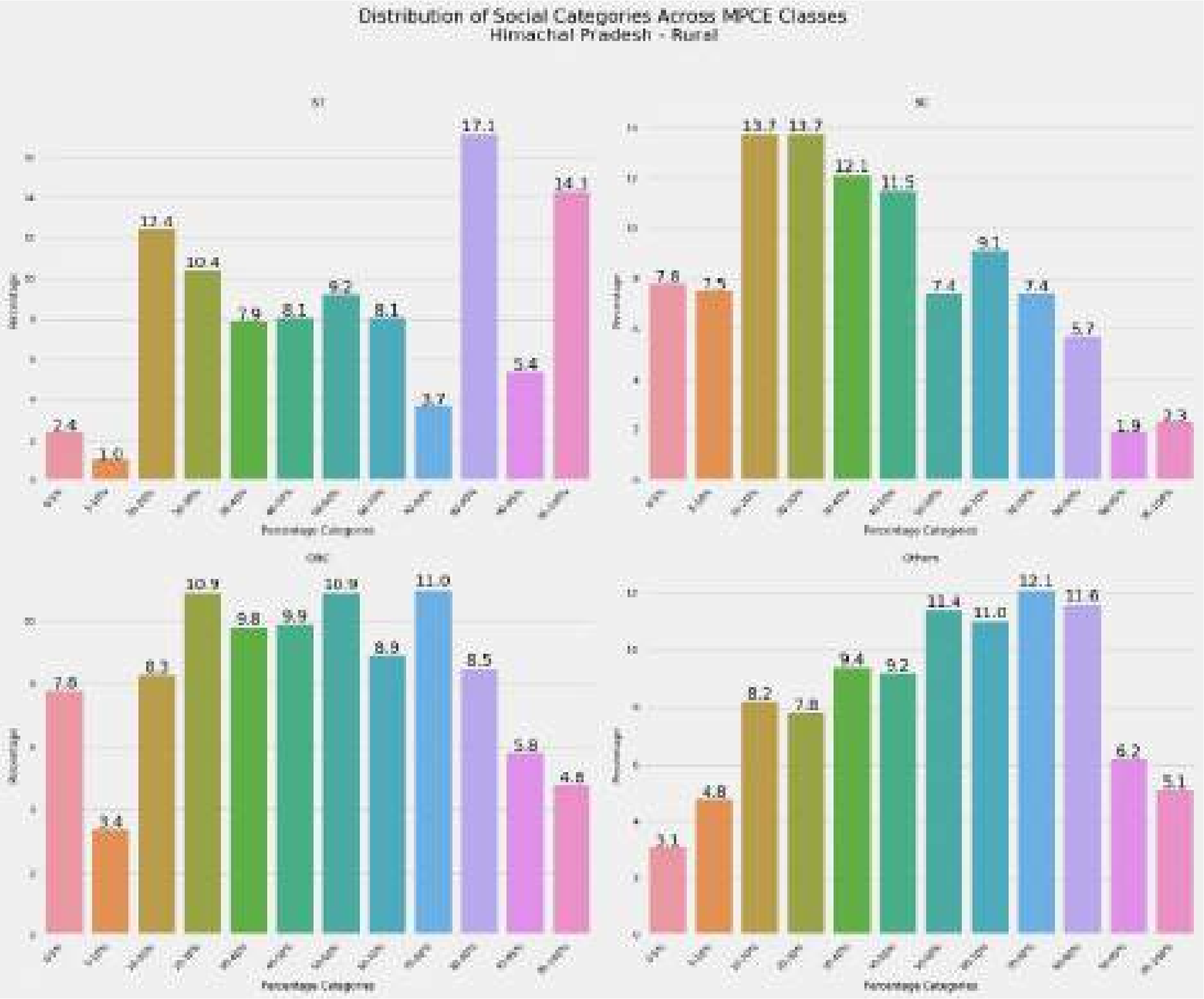

Distribution of Social Categories Across MPCE Classes
Goa - Urban
ST
Percentage
0.0
13.7
8.0
14.3
11.5
18.1
13.9
9.8
5.3
0.0
3.1
1.2
Percentage Categories
SC
Percentage
0.0
0.0
0.0
17.1
0.0
19.2
4.5
0.0
29.6
22.0
0.0
7.7
Percentage Categories
OBC
Percentage
11.3
3.8
20.3
11.5
0.0
3.2
13.8
8.0
6.8
11.1
2.3
7.9
Percentage Categories
Others
Percentage
5.0
3.5
7.4
8.7
12.0
9.1
8.3
11.7
10.8
11.9
6.3
5.2
Percentage Categories

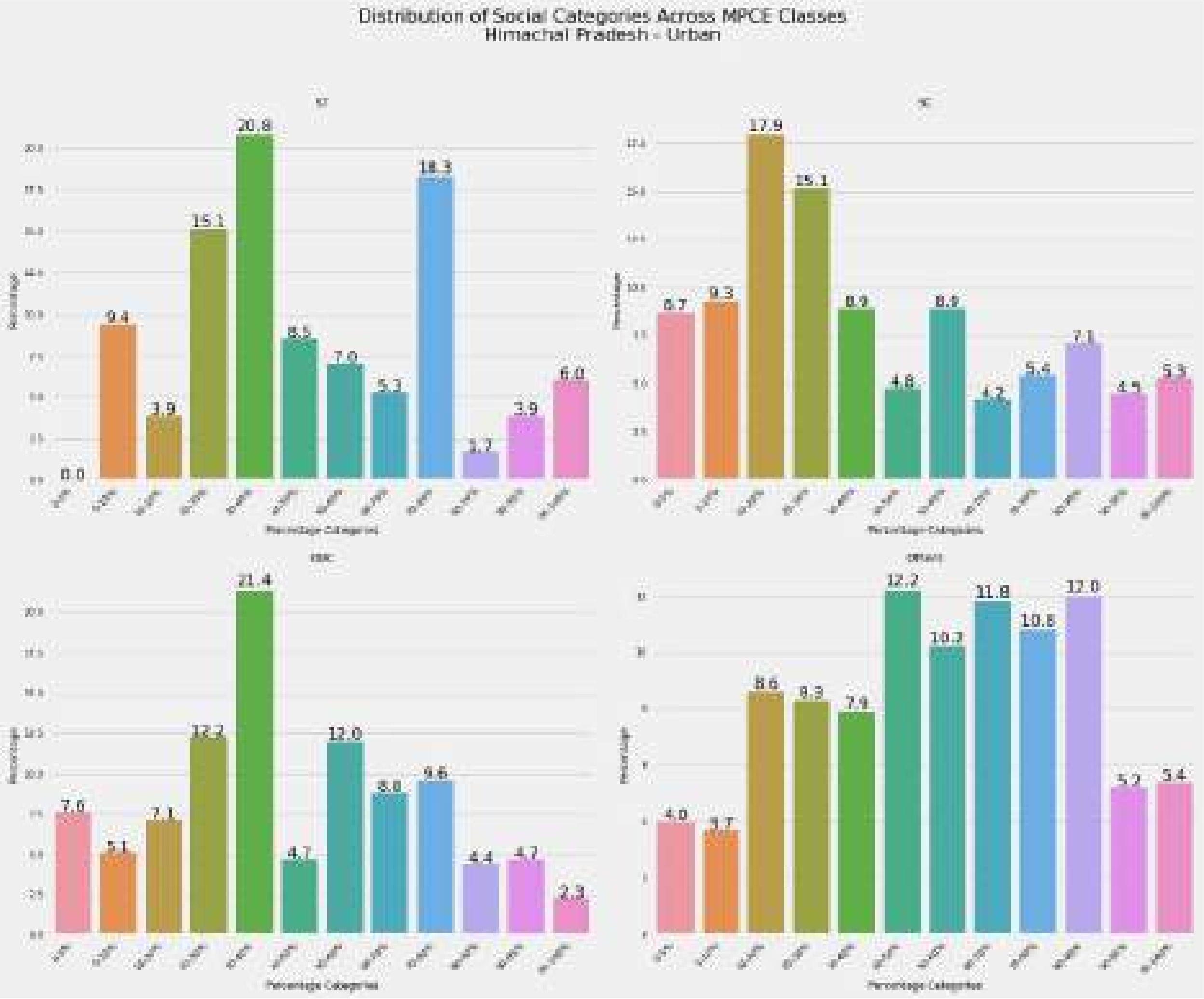

Distribution of Social Categories Across MPCE Classes
Gujarat - Rural
ST
Percentage
6.8
7.7
13.7
10.9
10.6
9.1
9.8
9.0
9.8
6.8
3.5
2.2
Percentage Categories
SC
Percentage
6.0
5.7
11.7
10.7
10.1
10.0
10.1
8.3
9.3
10.2
4.0
3.7
Percentage Categories
OBC
Percentage
5.8
5.0
10.3
11.3
11.4
11.2
9.8
9.6
8.9
8.7
3.8
4.1
Percentage Categories
Others
Percentage
1.1
1.8
4.8
6.0
6.2
8.3
10.6
12.7
12.9
15.8
9.4
10.3
Percentage Categories

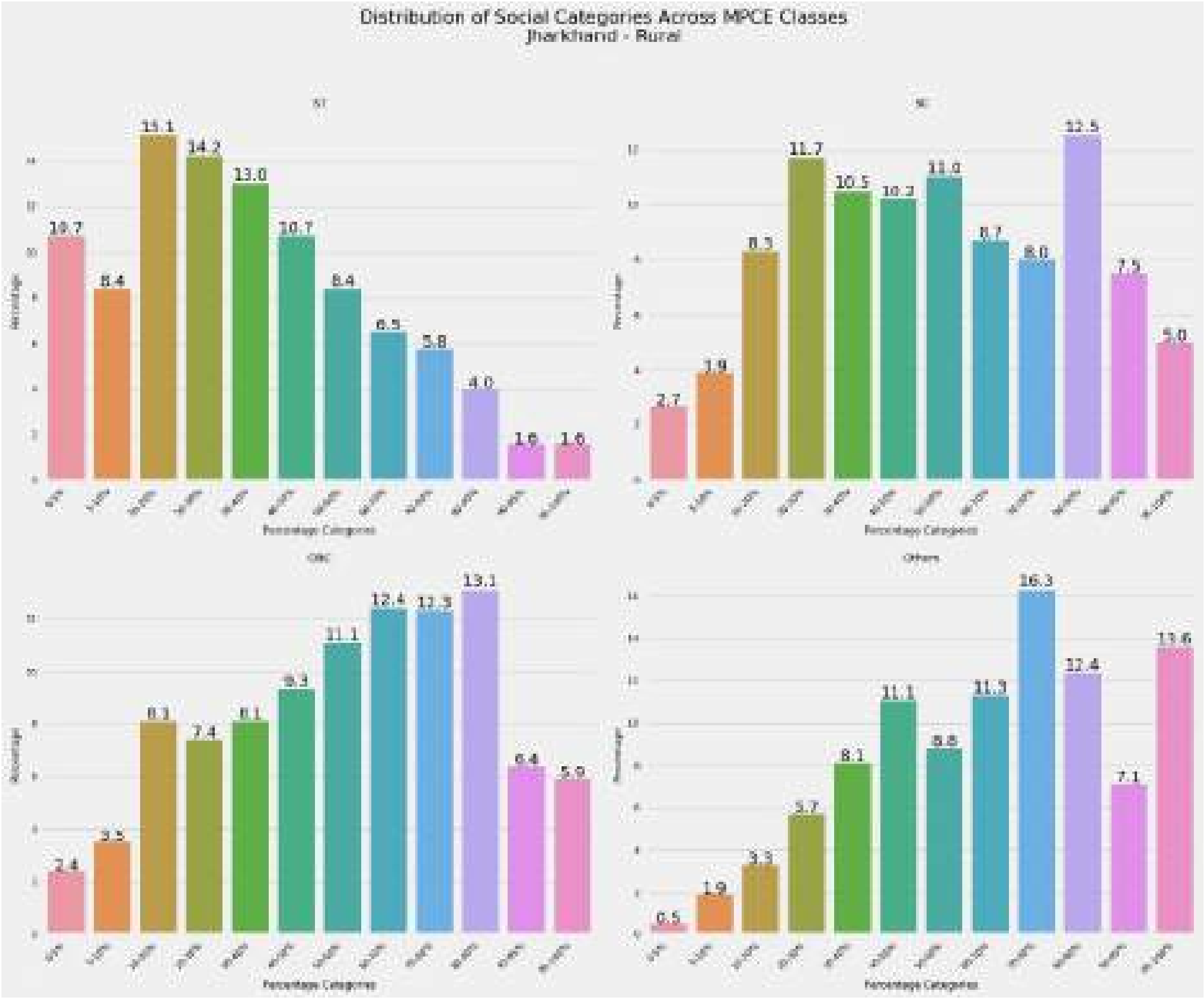

Distribution of Social Categories Across MPCE Classes
Gujarat - Urban
ST
Percentage
14.3
8.9
14.2
14.8
13.8
4.2
10.5
3.1
4.6
5.1
1.2
5.4
Percentage Categories
SC
Percentage
8.5
5.1
16.9
16.8
9.8
7.9
9.9
9.0
5.6
6.4
3.1
1.0
Percentage Categories
OBC
Percentage
6.6
6.2
11.9
10.9
12.3
10.9
9.1
9.2
9.1
8.1
3.3
2.5
Percentage Categories
Others
Percentage
2.0
3.5
6.7
7.1
7.8
10.1
10.7
11.5
12.4
12.9
7.3
8.1
Percentage Categories

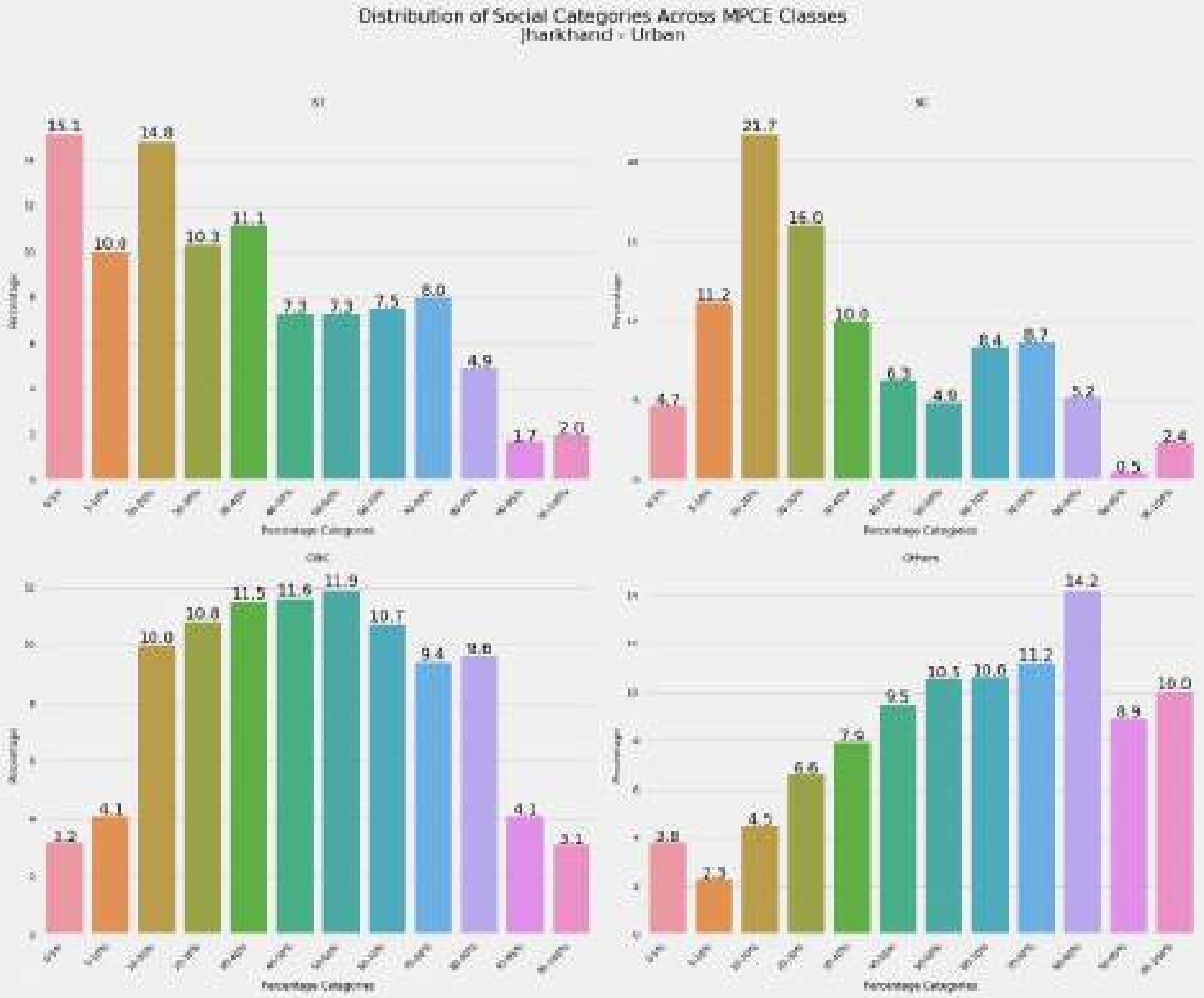

Distribution of Social Categories Across MPCE Classes
Haryana - Rural
ST
Percentage
0.0
0.0
29.5
0.0
0.0
0.0
0.0
11.4
59.0
0.0
0.0
0.0
Percentage Categories
SC
Percentage
8.4
5.8
13.3
11.0
12.7
10.6
9.8
7.7
8.3
7.0
2.7
2.8
Percentage Categories
OBC
Percentage
4.8
6.4
8.3
11.5
9.1
11.5
9.6
9.7
9.5
10.7
4.7
4.2
Percentage Categories
Others
Percentage
2.1
2.6
8.6
7.4
8.5
7.8
10.7
12.4
12.0
12.3
7.6
8.0
Percentage Categories

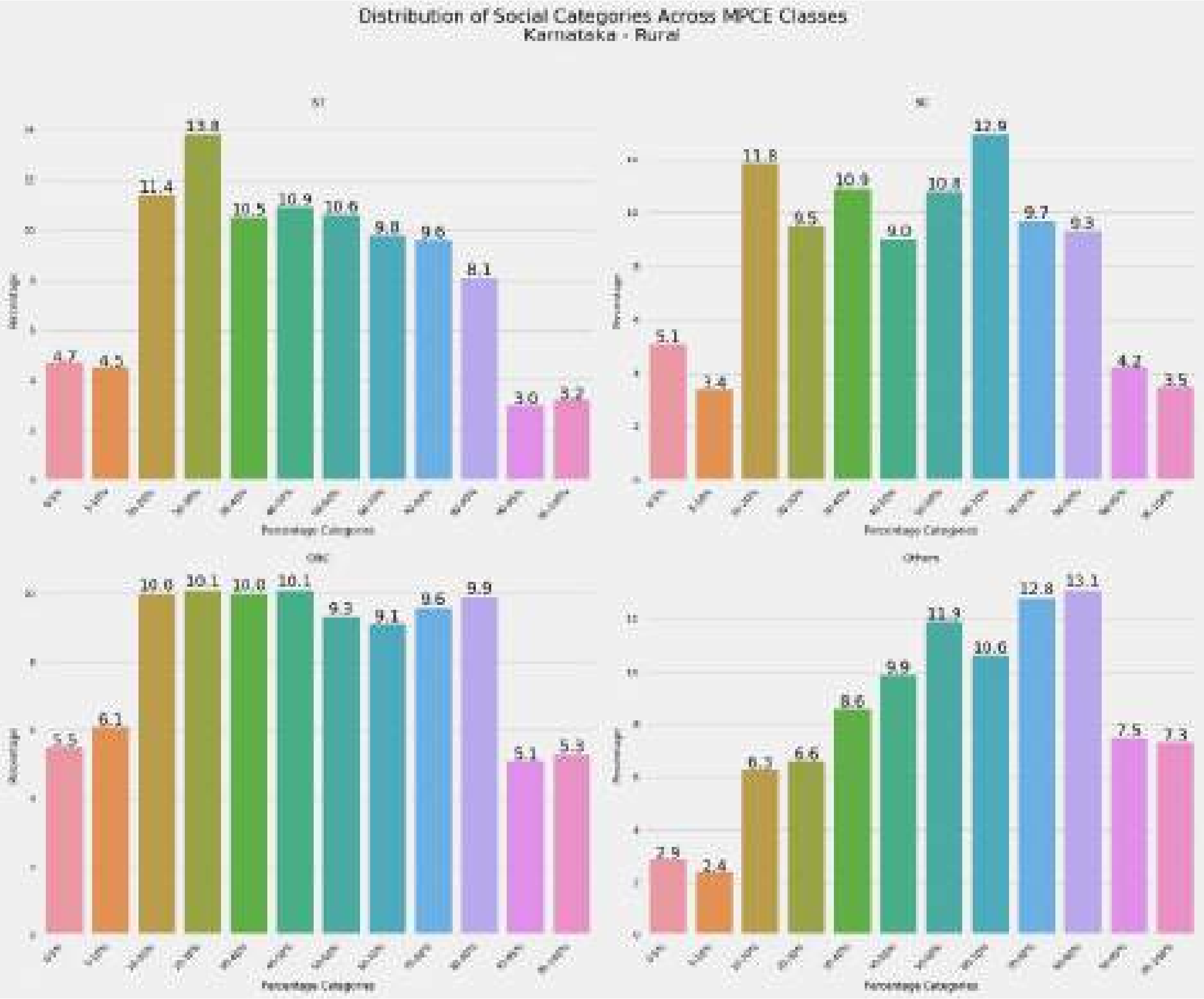

Distribution of Social Categories Across MPCE Classes
Haryana - Urban
ST
Percentage
23.6
19.8
14.0
12.9
0.0
0.0
0.9
5.3
3.9
5.9
9.4
4.4
Percentage Categories
SC
Percentage
9.6
11.5
18.3
12.8
10.6
8.4
7.2
5.6
6.3
5.1
2.4
2.0
Percentage Categories
OBC
Percentage
6.7
4.6
9.5
12.1
9.6
10.8
12.0
10.7
8.9
8.5
3.6
3.0
Percentage Categories
Others
Percentage
1.0
2.0
6.5
7.3
10.4
10.6
9.8
11.9
12.6
13.4
7.0
7.6
Percentage Categories

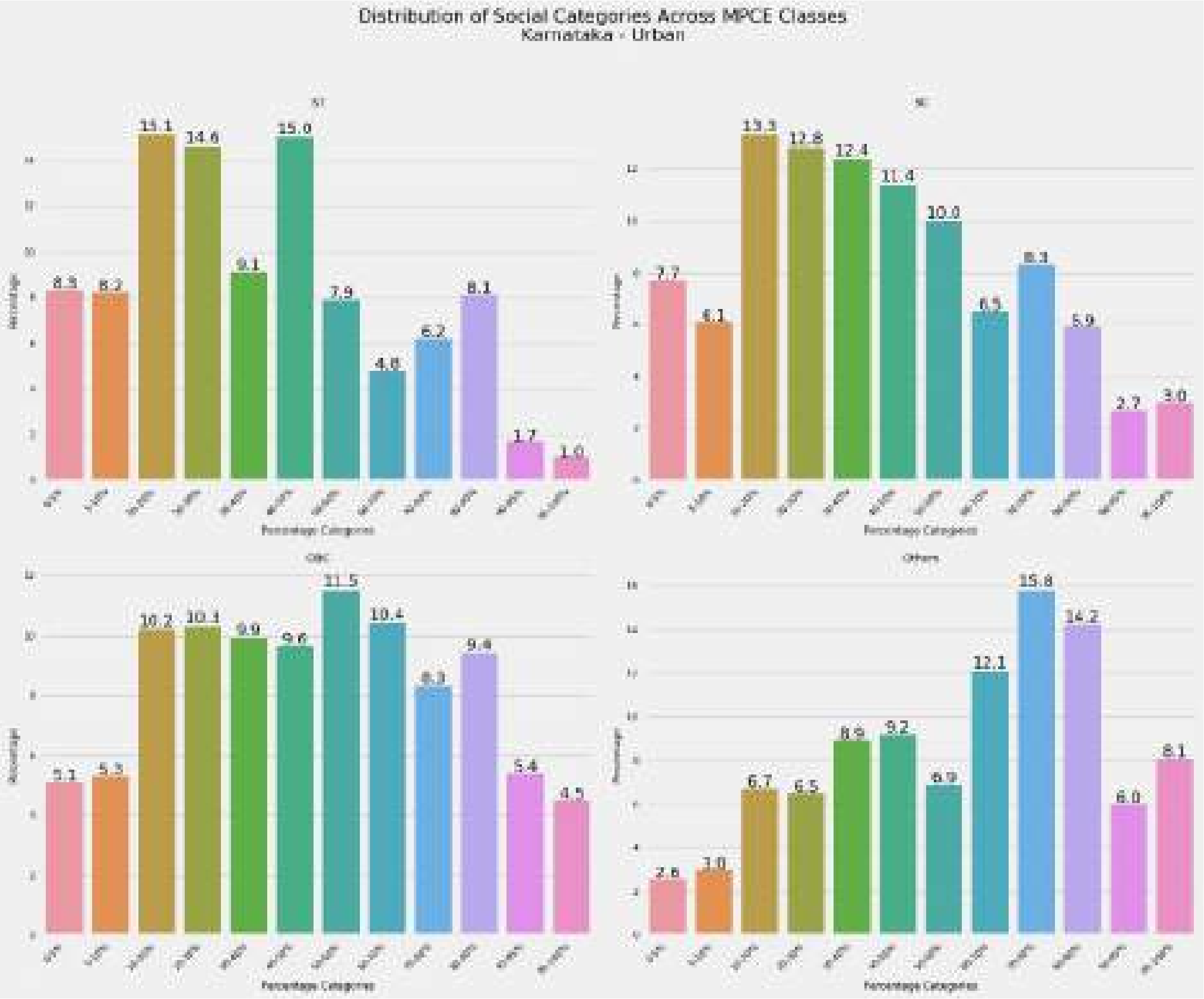

Distribution of Social Categories Across MPCE Classes
Himachal Pradesh - Rural
ST
Percentage
2.4
1.0
12.4
10.4
7.9
8.1
9.2
8.1
3.7
17.1
5.4
14.1
Percentage Categories
SC
Percentage
7.8
7.5
13.7
13.7
12.1
11.5
7.4
9.1
7.4
5.7
1.9
2.3
Percentage Categories
OBC
Percentage
7.8
3.4
8.3
10.9
9.8
9.9
10.9
8.9
11.0
8.5
5.8
4.8
Percentage Categories
Others
Percentage
3.1
4.8
8.2
7.8
9.4
9.2
11.4
11.0
12.1
11.6
6.2
5.1
Percentage Categories


Distribution of Social Categories Across MPCE Classes
Himachal Pradesh - Urban
ST
Percentage
0.0
9.4
3.9
15.1
20.8
8.5
7.0
5.3
18.3
1.7
3.9
6.0
Percentage Categories
SC
Percentage
8.7
9.3
17.9
15.1
8.9
4.8
8.9
4.2
5.4
7.1
4.5
5.3
Percentage Categories
OBC
Percentage
7.6
5.1
7.1
12.2
21.4
4.7
12.0
8.8
9.6
4.4
4.7
2.3
Percentage Categories
Others
Percentage
4.0
3.7
8.6
8.3
7.9
12.2
10.2
11.8
10.8
12.0
5.2
5.4
Percentage Categories

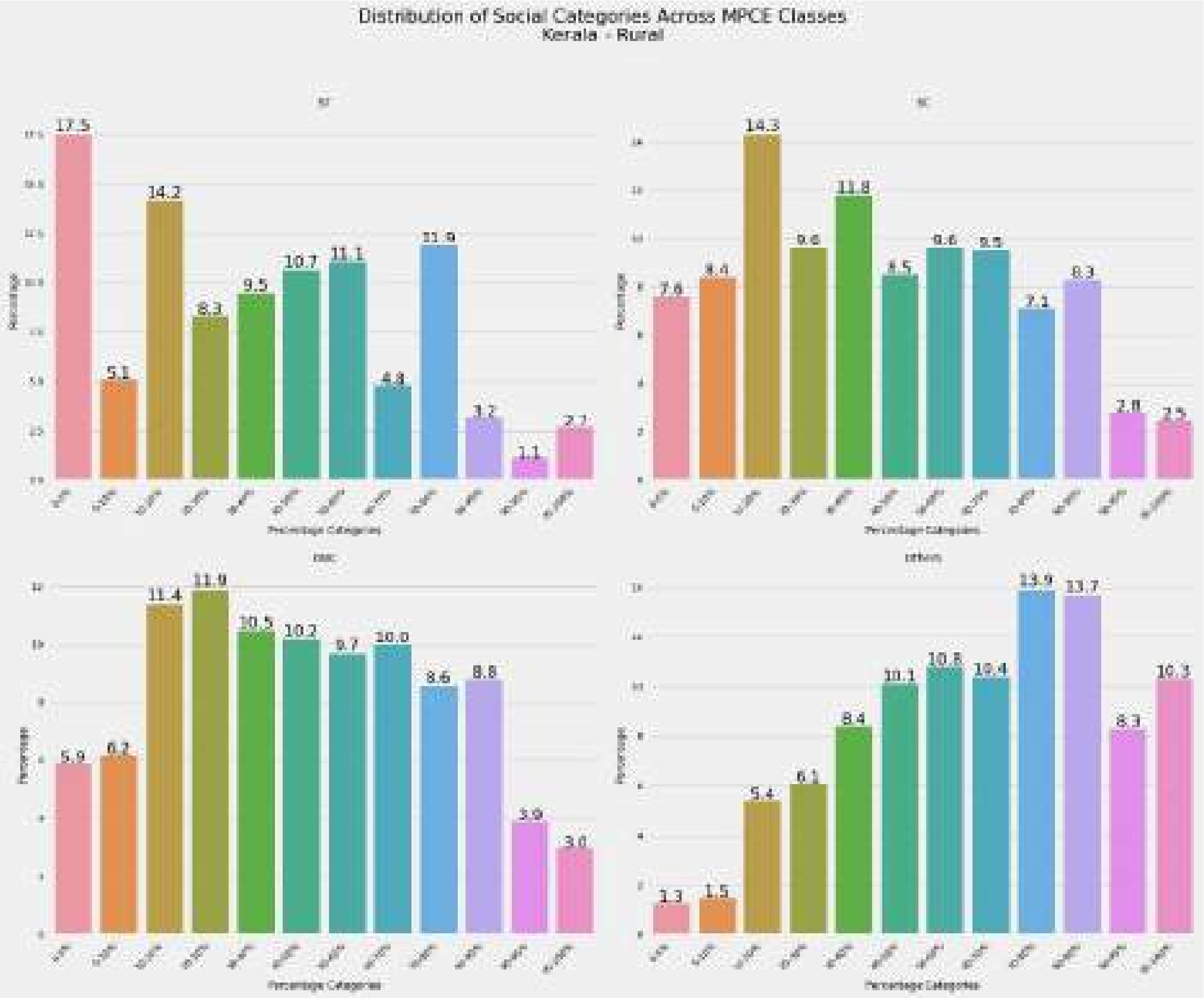

Distribution of Social Categories Across MPCE Classes
Jharkhand - Rural
ST
10.7
8.4
15.1
14.2
13.0
10.7
8.4
6.5
5.8
4.0
1.6
1.6
Percentage
Percentage Categories
SC
2.7
3.9
8.3
11.7
10.5
10.2
11.0
8.7
8.0
12.5
7.5
5.0
Percentage
Percentage Categories
OBC
2.4
3.5
8.1
7.4
8.1
9.3
11.1
12.4
12.3
13.1
6.4
5.9
Percentage
Percentage Categories
Others
0.5
1.9
3.3
5.7
8.1
11.1
8.8
11.3
16.3
12.4
7.1
13.6
Percentage
Percentage Categories

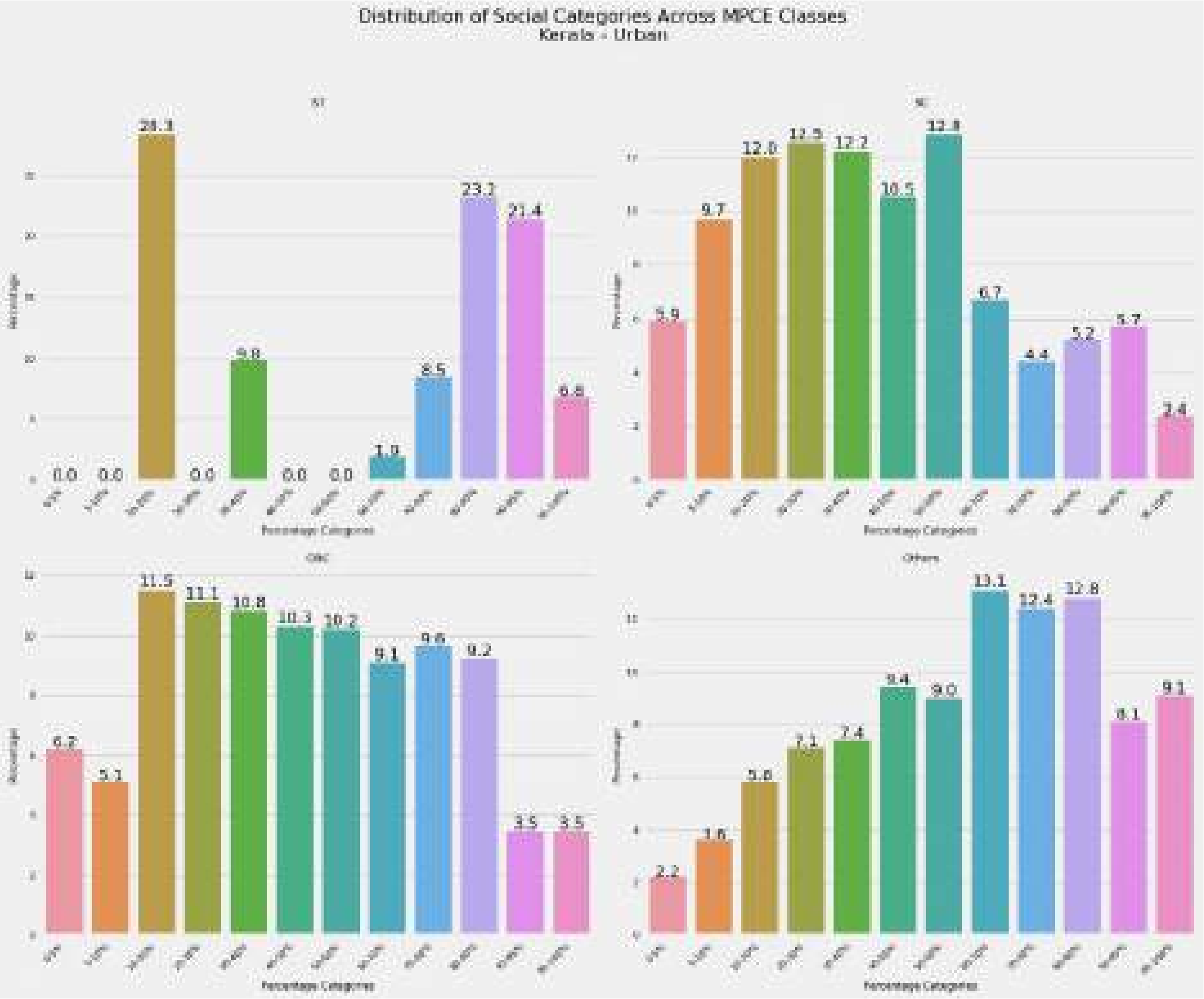

Distribution of Social Categories Across MPCE Classes
Jharkhand - Urban
ST
Percentage
15.1
10.0
14.8
10.3
11.1
7.3
7.3
7.5
8.0
4.9
1.7
2.0
Percentage Categories
SC
Percentage
4.7
11.2
21.7
16.0
10.0
6.3
4.9
8.4
8.7
5.2
0.5
2.4
Percentage Categories
OBC
Percentage
3.2
4.1
10.0
10.8
11.5
11.6
11.9
10.7
9.4
9.6
4.1
3.1
Percentage Categories
Others
Percentage
3.8
2.3
4.5
6.6
7.9
9.5
10.5
10.6
11.2
14.2
8.9
10.0
Percentage Categories

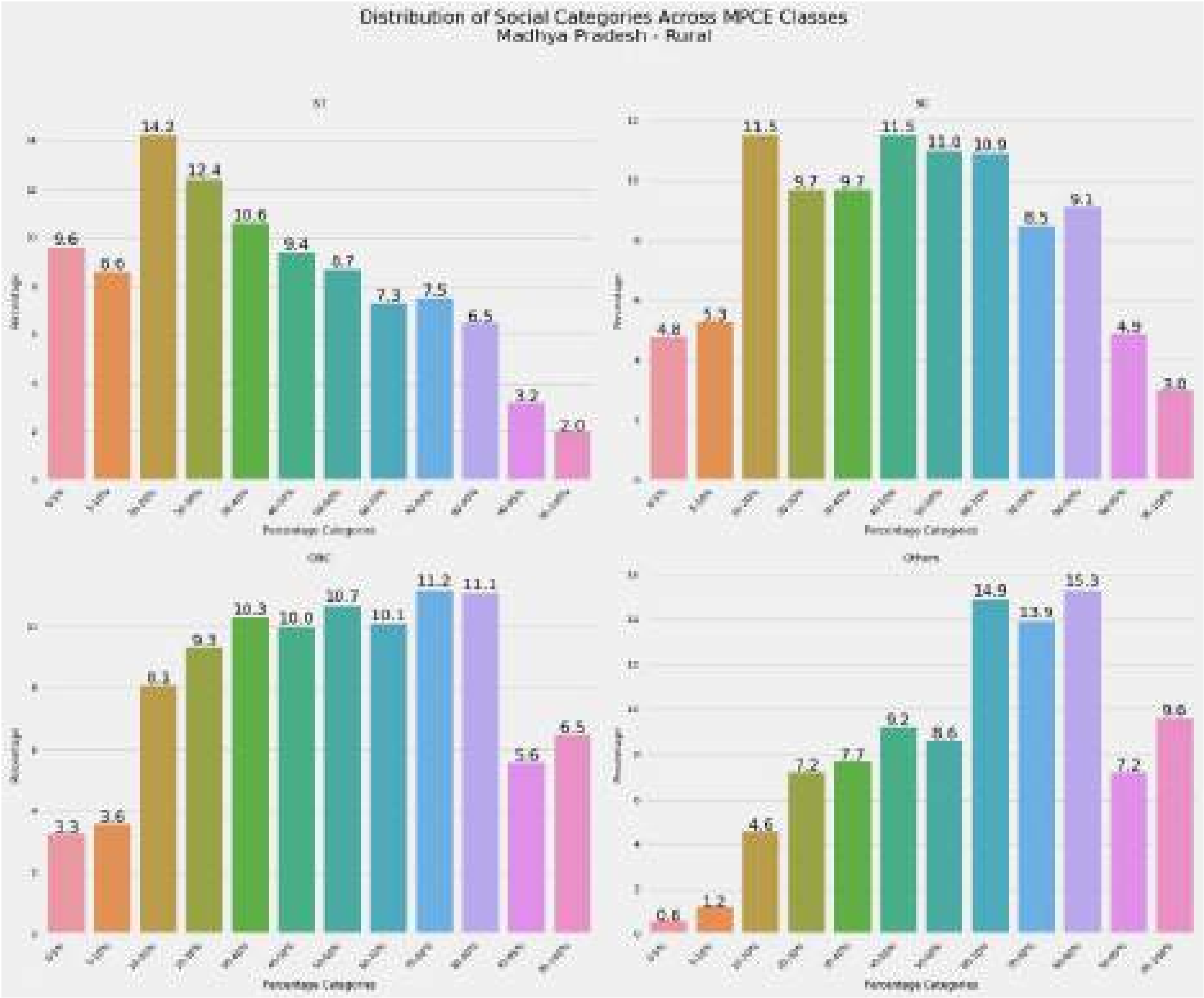

Distribution of Social Categories Across MPCE Classes
Karnataka - Rural
ST
Percentage
4.7
4.5
11.4
13.8
10.5
10.9
10.6
9.8
9.6
8.1
3.0
3.2
Percentage Categories
SC
Percentage
5.1
3.4
11.8
9.5
10.9
9.0
10.8
12.9
9.7
9.3
4.2
3.5
Percentage Categories
OBC
Percentage
5.5
6.1
10.0
10.1
10.0
10.1
9.3
9.1
9.6
9.9
5.1
5.3
Percentage Categories
Others
Percentage
2.9
2.4
6.3
6.6
8.6
9.9
11.9
10.6
12.8
13.1
7.5
7.3
Percentage Categories

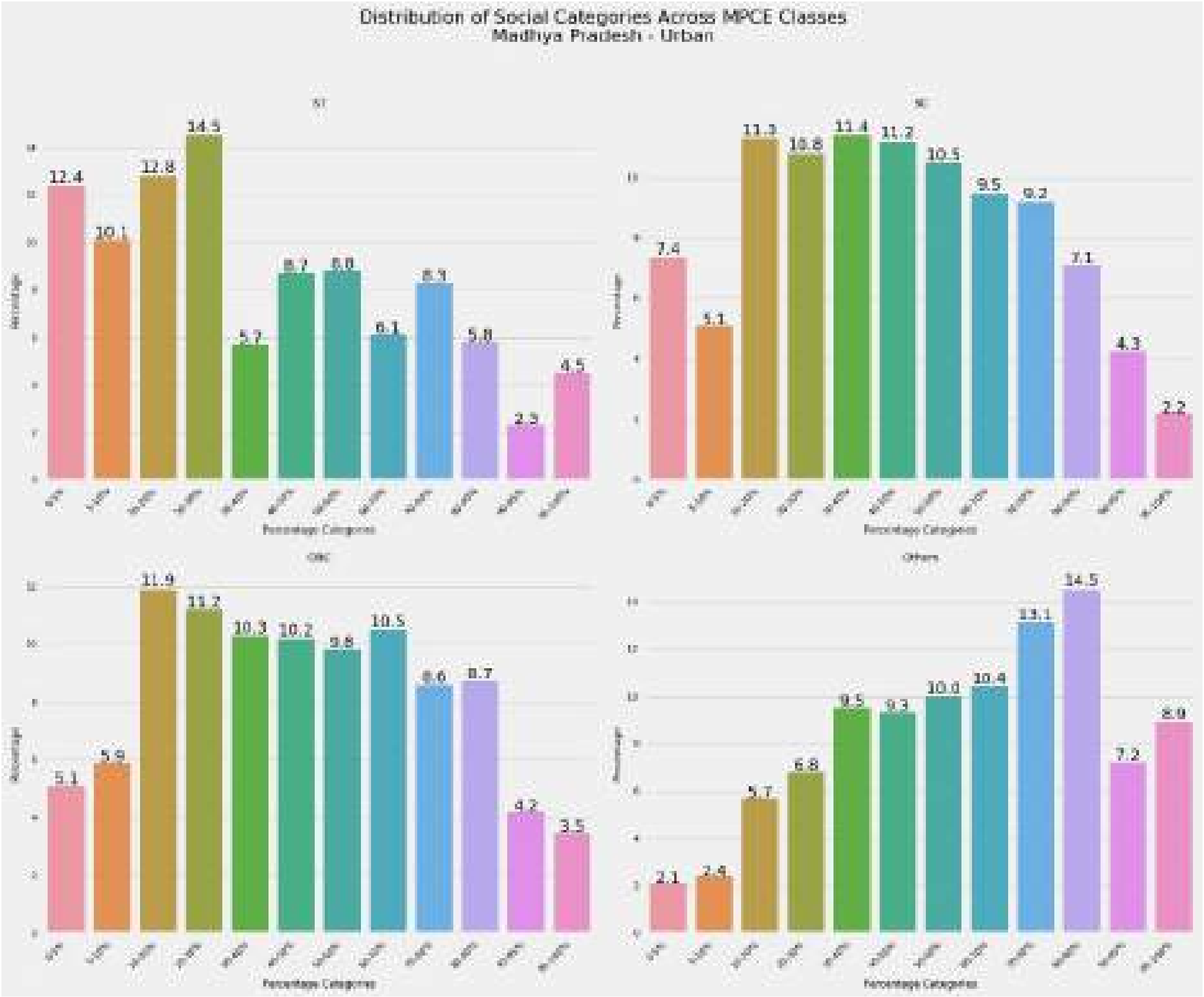


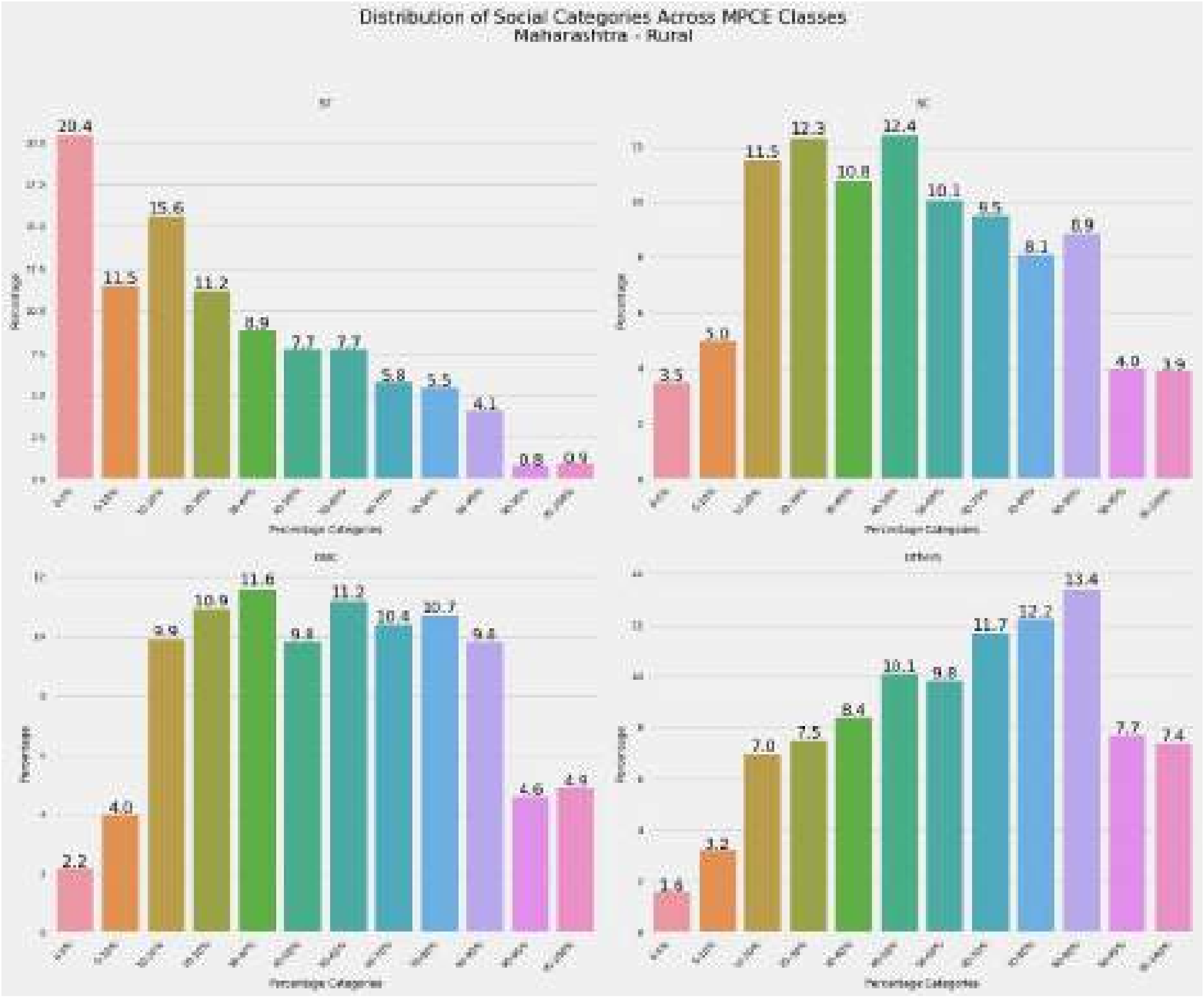
Distribution of Social Categories Across MPCE Classes
Karnataka - Urban

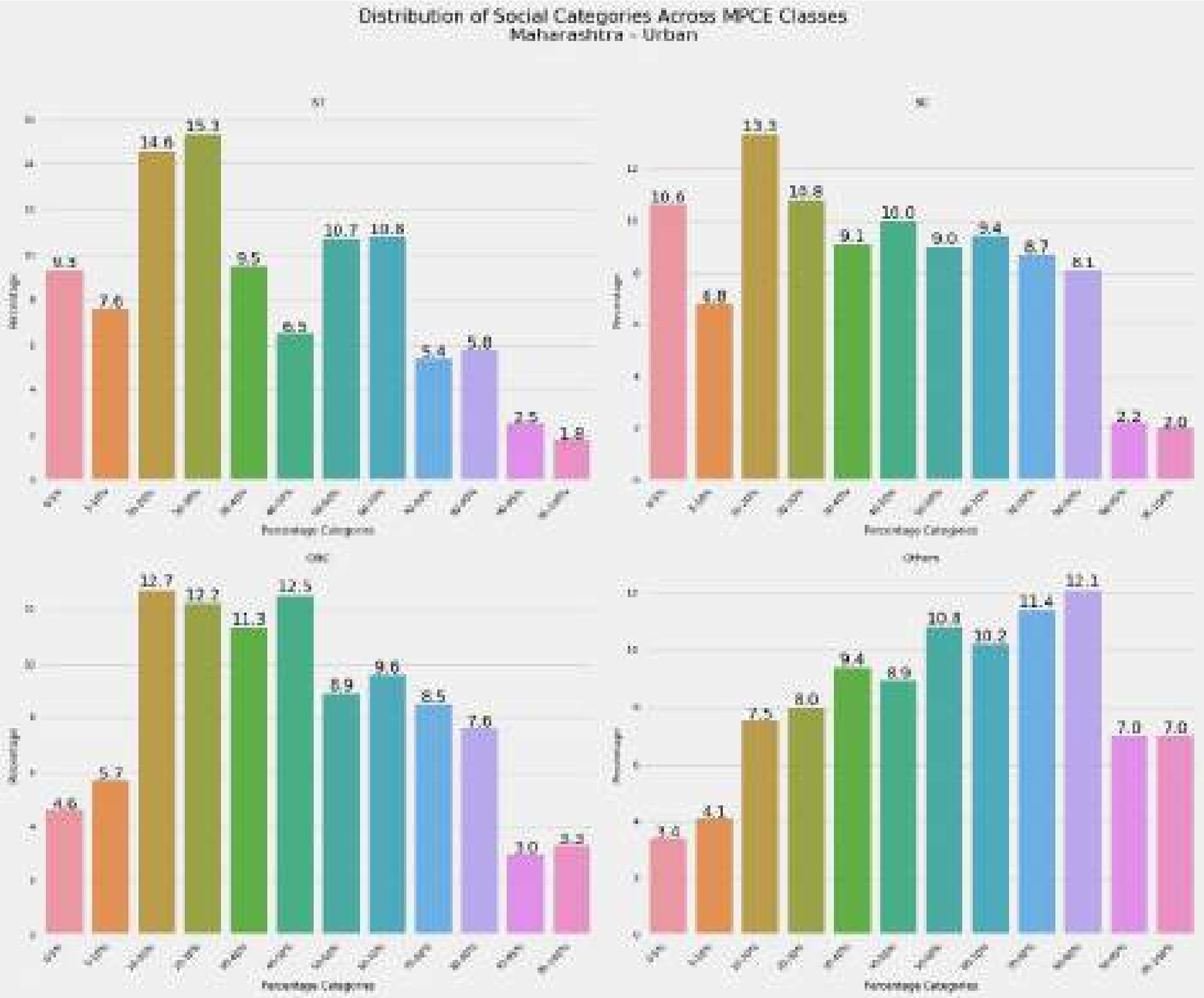
Distribution of Social Categories Across MPCE Classes
Kerala - Rural
ST
17.5
5.1
14.2
8.3
9.5
10.7
11.1
4.8
11.9
3.2
1.1
2.7
Percentage
Percentage Categories
SC
7.6
8.4
14.3
9.6
11.8
8.5
9.6
9.5
7.1
8.3
2.8
2.5
Percentage
Percentage Categories
OBC
5.9
6.2
11.4
11.9
10.5
10.2
9.7
10.0
8.6
8.8
3.9
3.0
Percentage
Percentage Categories
Others
1.3
1.5
5.4
6.1
8.4
10.1
10.8
10.4
13.9
13.7
8.3
10.3
Percentage
Percentage Categories

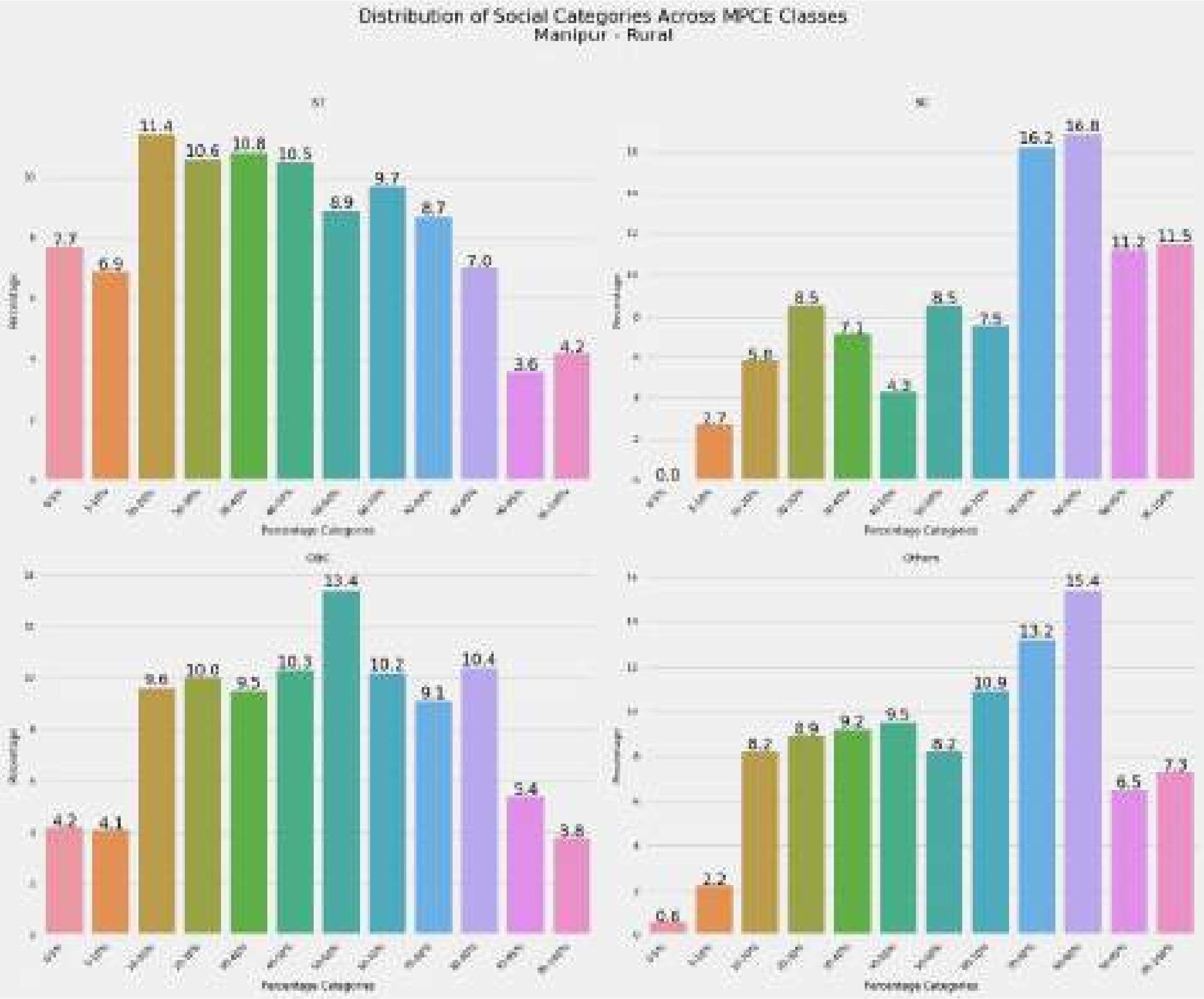

Distribution of Social Categories Across MPCE Classes
Kerala - Urban
ST
0.0
0.0
28.3
0.0
9.8
0.0
0.0
1.9
8.5
23.2
21.4
6.8
0
5
10
15
20
25
Percentage
Percentage Categories
SC
5.9
9.7
12.0
12.5
12.2
10.5
12.8
6.7
4.4
5.2
5.7
2.4
0
2
4
6
8
10
12
Percentage
Percentage Categories
OBC
6.2
5.1
11.5
11.1
10.8
10.3
10.2
9.1
9.6
9.2
3.5
3.5
0
2
4
6
8
10
12
Percentage
Percentage Categories
Others
2.2
3.6
5.8
7.1
7.4
9.4
9.0
13.1
12.4
12.8
8.1
9.1
0
2
4
6
8
10
12
Percentage
Percentage Categories

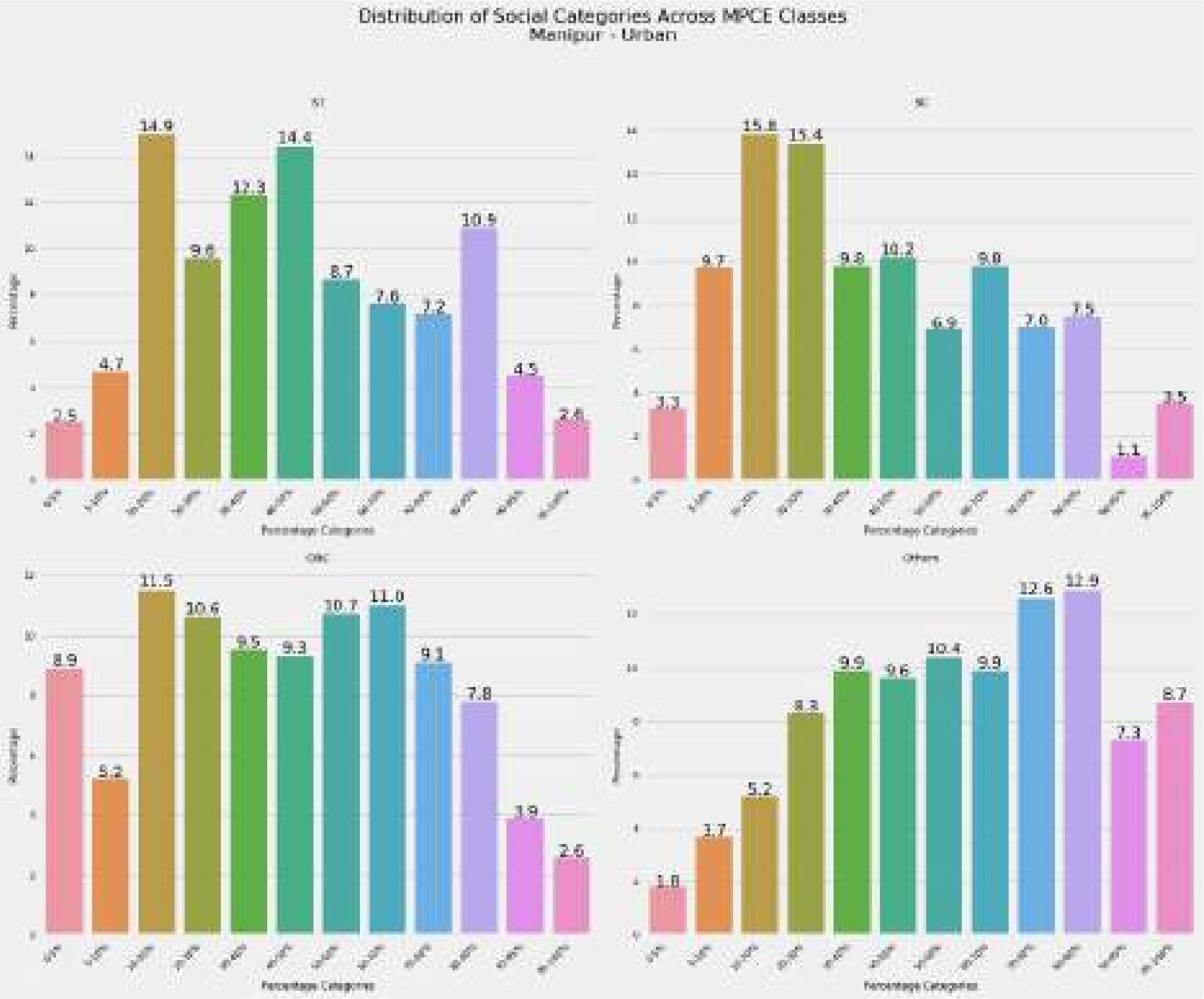

Distribution of Social Categories Across MPCE Classes
Madhya Pradesh - Rural
ST
Percentage
9.6
8.6
14.2
12.4
10.6
9.4
8.7
7.3
7.5
6.5
3.2
2.0
Percentage Categories
SC
Percentage
4.8
5.3
11.5
9.7
9.7
11.5
11.0
10.9
8.5
9.1
4.9
3.0
Percentage Categories
OBC
Percentage
3.3
3.6
8.1
9.3
10.3
10.0
10.7
10.1
11.2
11.1
5.6
6.5
Percentage Categories
Others
Percentage
0.6
1.2
4.6
7.2
7.7
9.2
8.6
14.9
13.9
15.3
7.2
9.6
Percentage Categories

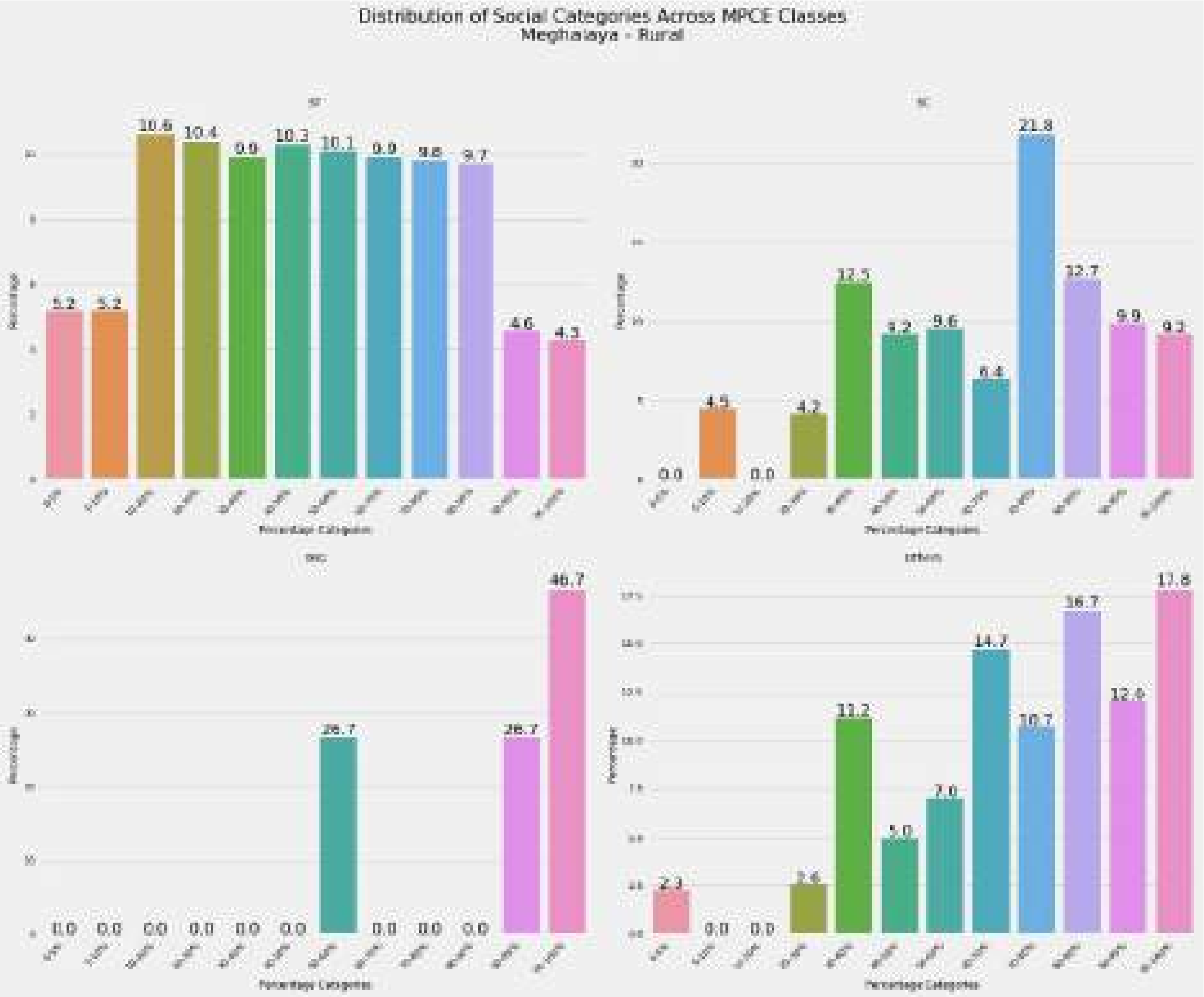

Distribution of Social Categories Across MPCE Classes
Madhya Pradesh - Urban
ST
Percentage
Percentage Categories
12.4
10.1
12.8
14.5
5.7
8.7
8.8
6.1
8.3
5.8
2.3
4.5
SC
Percentage
Percentage Categories
7.4
5.1
11.3
10.8
11.4
11.2
10.5
9.5
9.2
7.1
4.3
2.2
OBC
Percentage
Percentage Categories
5.1
5.9
11.9
11.2
10.3
10.2
9.8
10.5
8.6
8.7
4.2
3.5
Others
Percentage
Percentage Categories
2.1
2.4
5.7
6.8
9.5
9.3
10.0
10.4
13.1
14.5
7.2
8.9

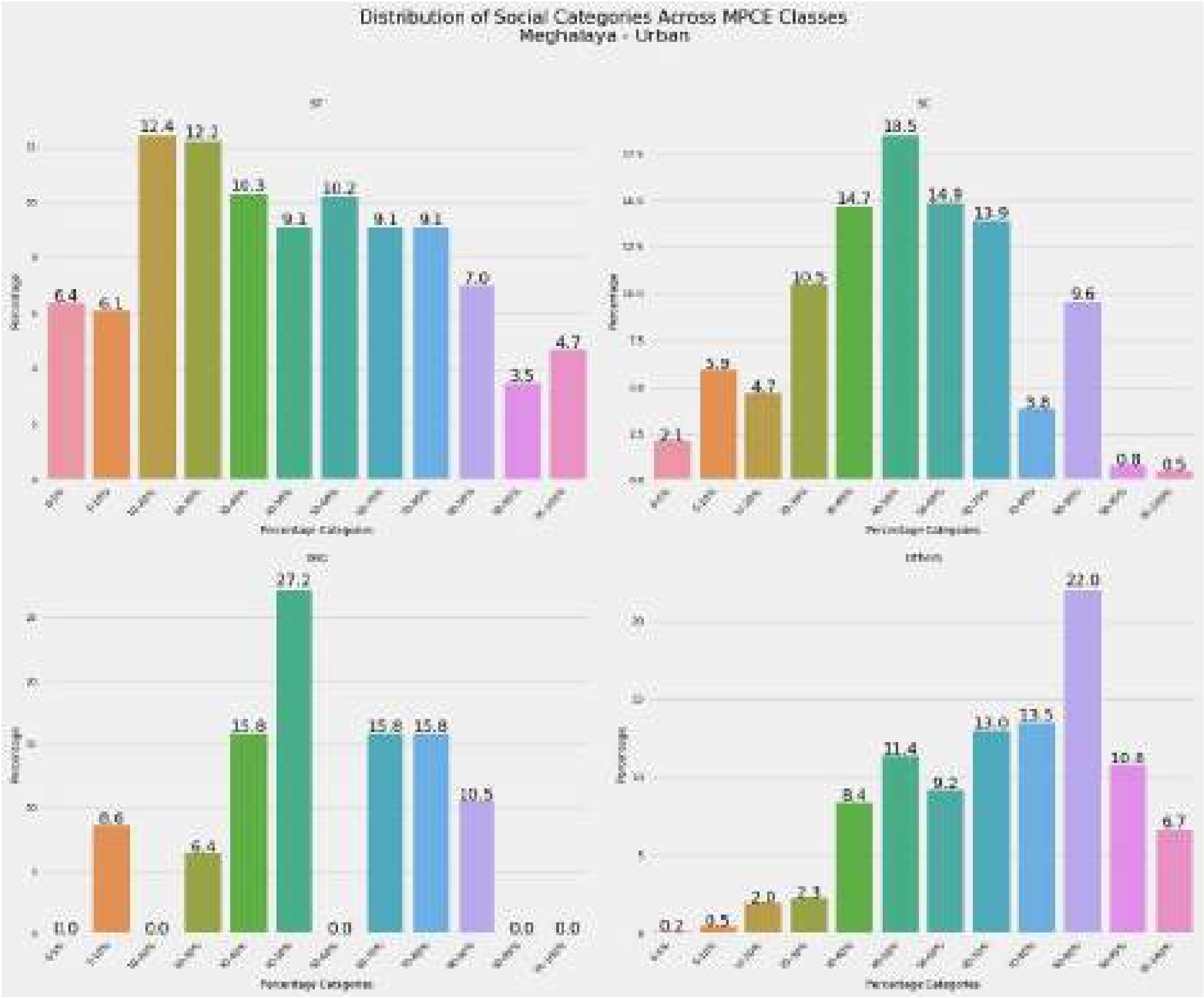

Distribution of Social Categories Across MPCE Classes
Maharashtra - Rural
ST
Percentage
Percentage Categories
20.4
11.5
15.6
11.2
8.9
7.7
7.7
5.8
5.5
4.1
0.8
0.9
SC
Percentage
Percentage Categories
3.5
5.0
11.5
12.3
10.8
12.4
10.1
9.5
8.1
8.9
4.0
3.9
OBC
Percentage
Percentage Categories
2.2
4.0
9.9
10.9
11.6
9.8
11.2
10.4
10.7
9.4
4.6
4.9
Others
Percentage
Percentage Categories
1.6
3.2
7.0
7.5
8.4
10.1
9.8
11.7
12.2
13.4
7.7
7.4

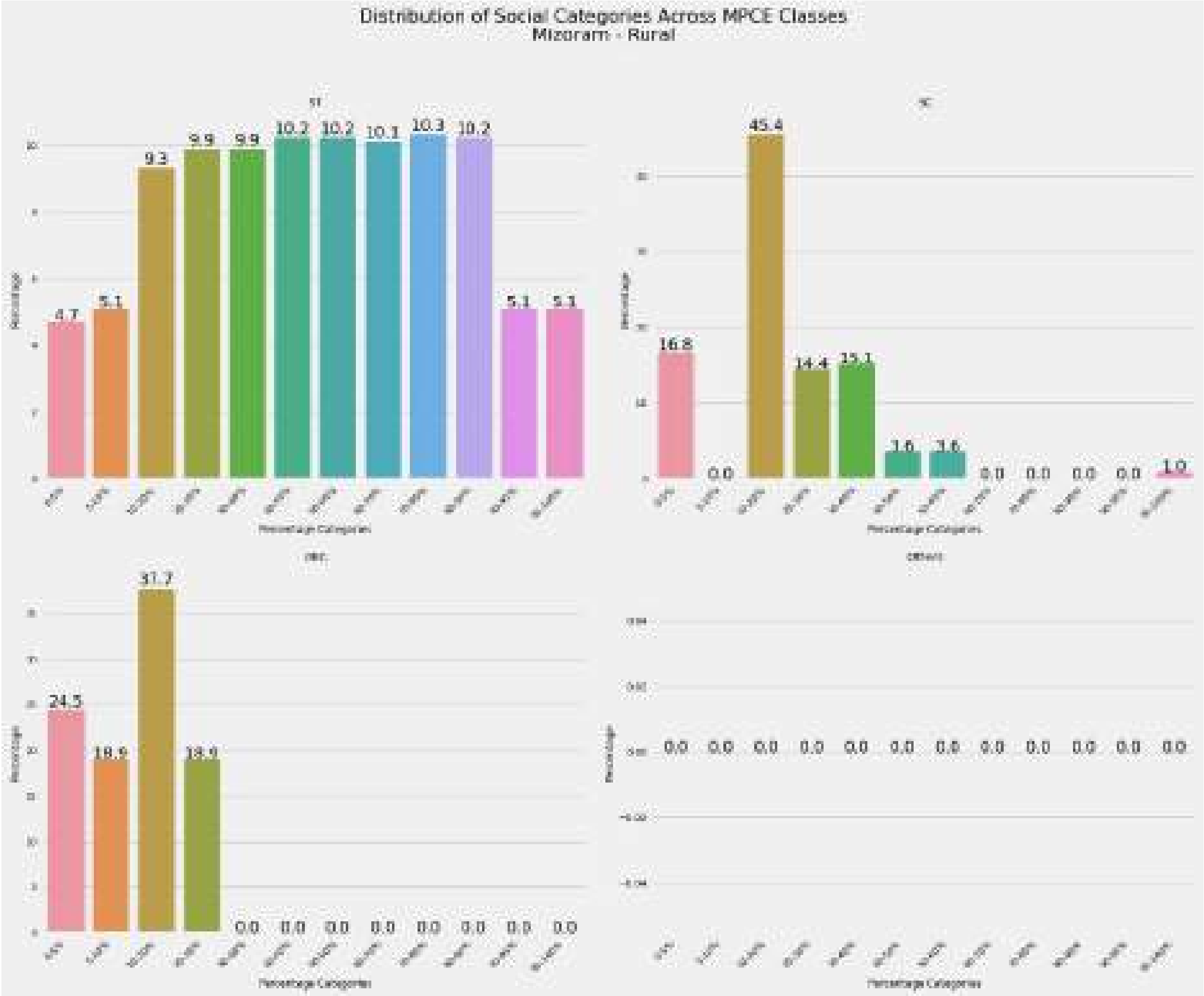

Distribution of Social Categories Across MPCE Classes
Maharashtra - Urban
ST
Percentage
9.3
7.6
14.6
15.3
9.5
6.5
10.7
10.8
5.4
5.8
2.5
1.8
Percentage Categories
SC
Percentage
10.6
6.8
13.3
10.8
9.1
10.0
9.0
9.4
8.7
8.1
2.2
2.0
Percentage Categories
OBC
Percentage
4.6
5.7
12.7
12.2
11.3
12.5
8.9
9.6
8.5
7.6
3.0
3.3
Percentage Categories
Others
Percentage
3.4
4.1
7.5
8.0
9.4
8.9
10.8
10.2
11.4
12.1
7.0
7.0
Percentage Categories

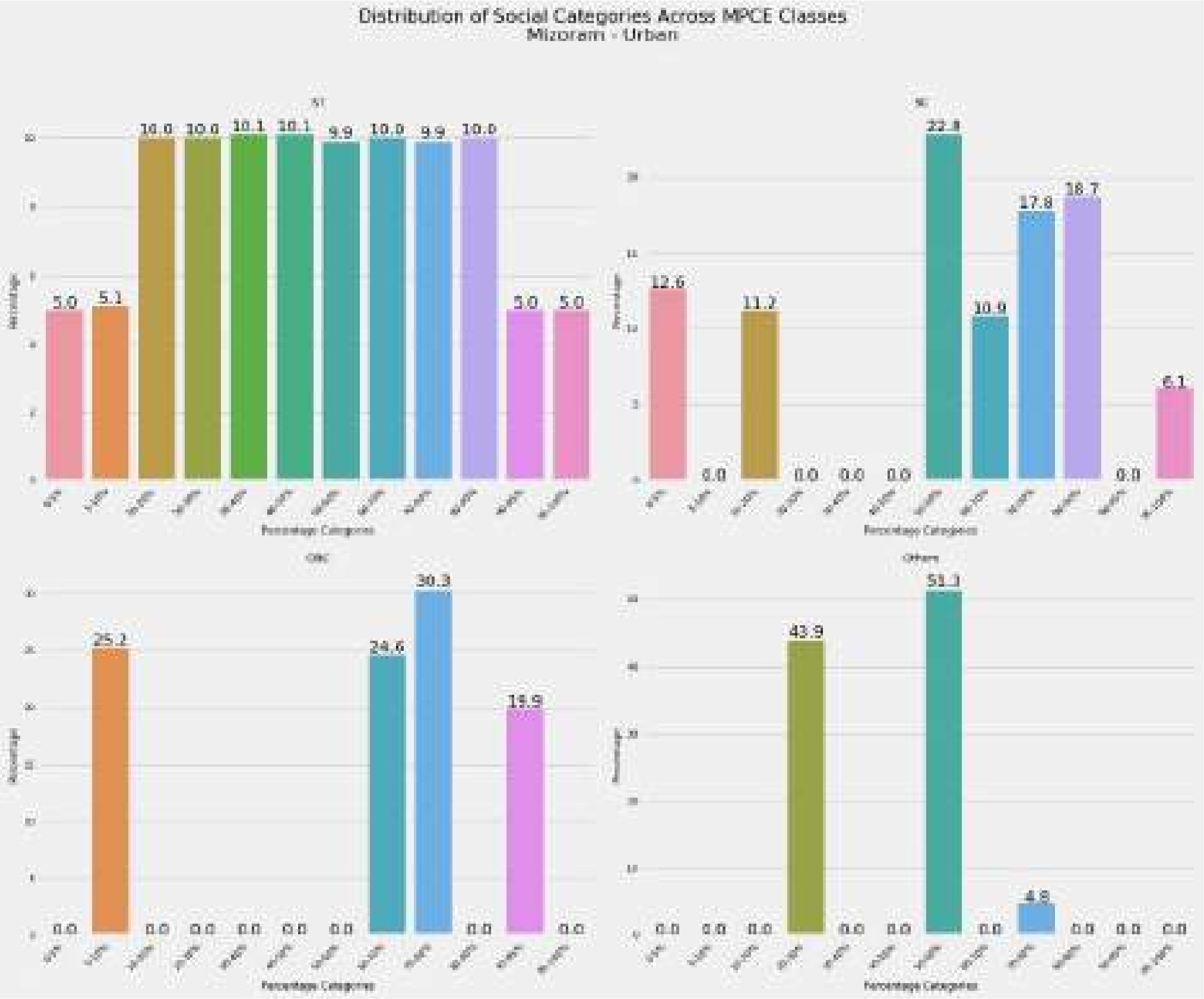

Distribution of Social Categories Across MPCE Classes
Manipur - Rural
ST
7.7
6.9
11.4
10.6
10.8
10.5
8.9
9.7
8.7
7.0
3.6
4.2
Percentage
Percentage Categories
SC
0.0
2.7
5.8
8.5
7.1
4.3
8.5
7.5
16.2
16.8
11.2
11.5
Percentage
Percentage Categories
OBC
4.2
4.1
9.6
10.0
9.5
10.3
13.4
10.2
9.1
10.4
5.4
3.8
Percentage
Percentage Categories
Others
0.6
2.2
8.2
8.9
9.2
9.5
8.2
10.9
13.2
15.4
6.5
7.3
Percentage
Percentage Categories

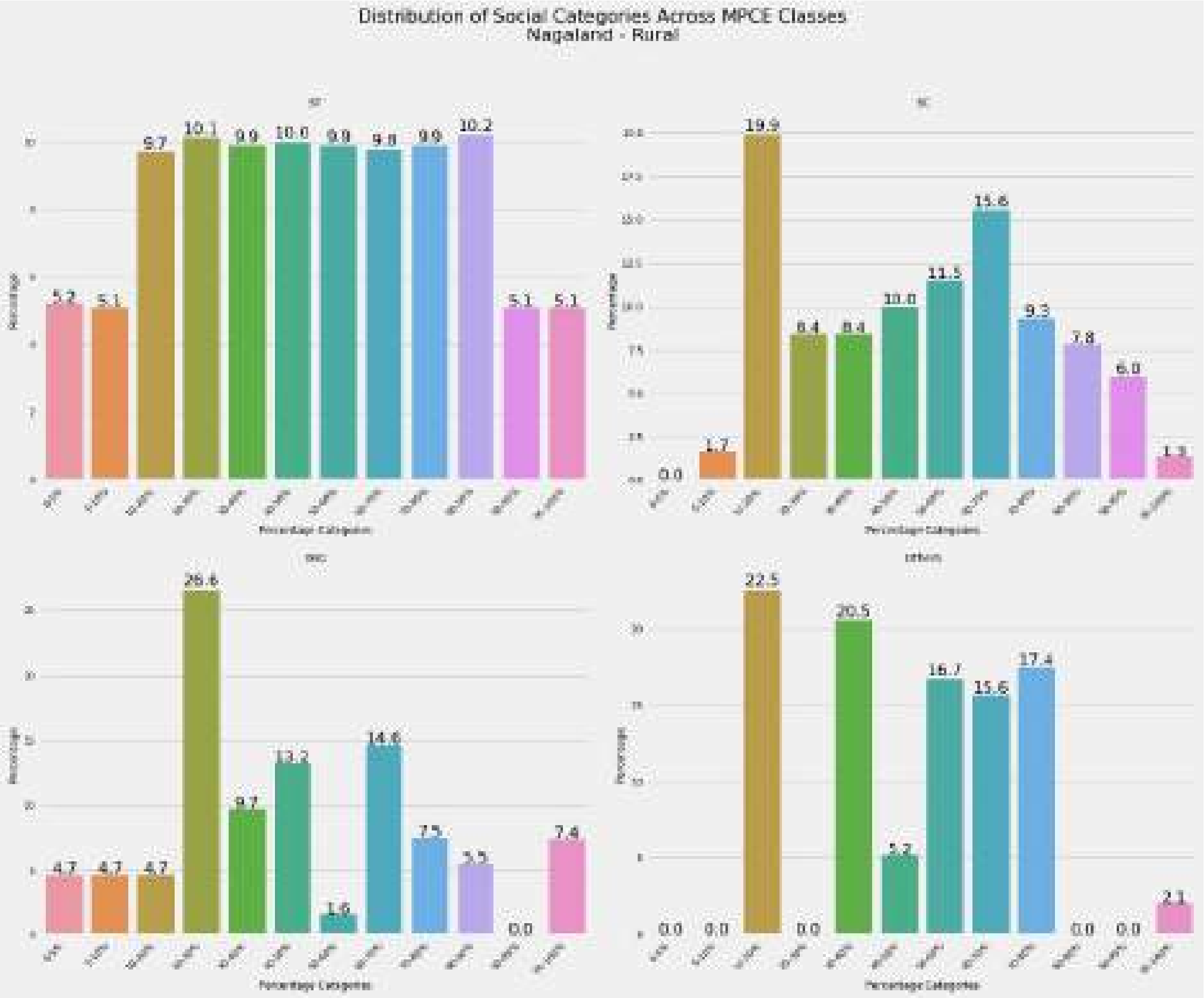

Distribution of Social Categories Across MPCE Classes
Manipur - Urban
ST
Percentage
2.5
4.7
14.9
9.6
12.3
14.4
8.7
7.6
7.2
10.9
4.5
2.6
Percentage Categories
SC
Percentage
3.3
9.7
15.8
15.4
9.8
10.2
6.9
9.8
7.0
7.5
1.1
3.5
Percentage Categories
OBC
Percentage
8.9
5.2
11.5
10.6
9.5
9.3
10.7
11.0
9.1
7.8
3.9
2.6
Percentage Categories
Others
Percentage
1.8
3.7
5.2
8.3
9.9
9.6
10.4
9.9
12.6
12.9
7.3
8.7
Percentage Categories

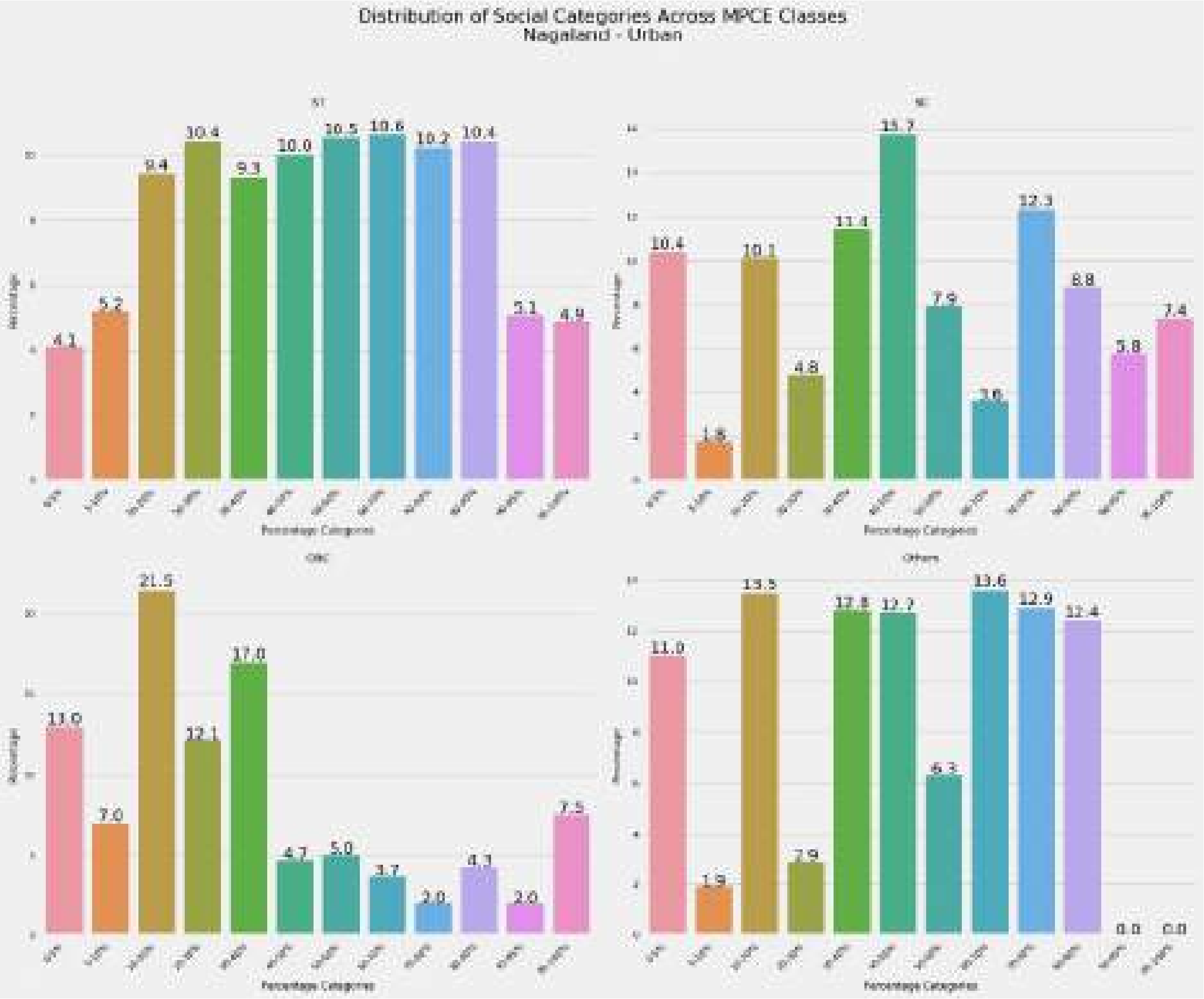
Distribution of Social Categories Across MPCE Classes
Meghalaya - Rural
ST
5.2
5.2
10.6
10.4
9.9
10.2
10.1
9.9
9.8
9.7
4.6
4.3
Percentage
Percentage Categories
SC
0.0
4.5
0.0
4.2
12.5
9.2
9.6
6.4
21.8
12.7
9.9
9.2
Percentage
Percentage Categories
OBC
0.0
0.0
0.0
0.0
0.0
0.0
26.7
0.0
0.0
0.0
26.7
46.7
Percentage
Percentage Categories
Others
2.3
0.0
0.0
2.6
11.2
5.0
7.0
14.7
10.7
16.7
12.0
17.8
Percentage
Percentage Categories

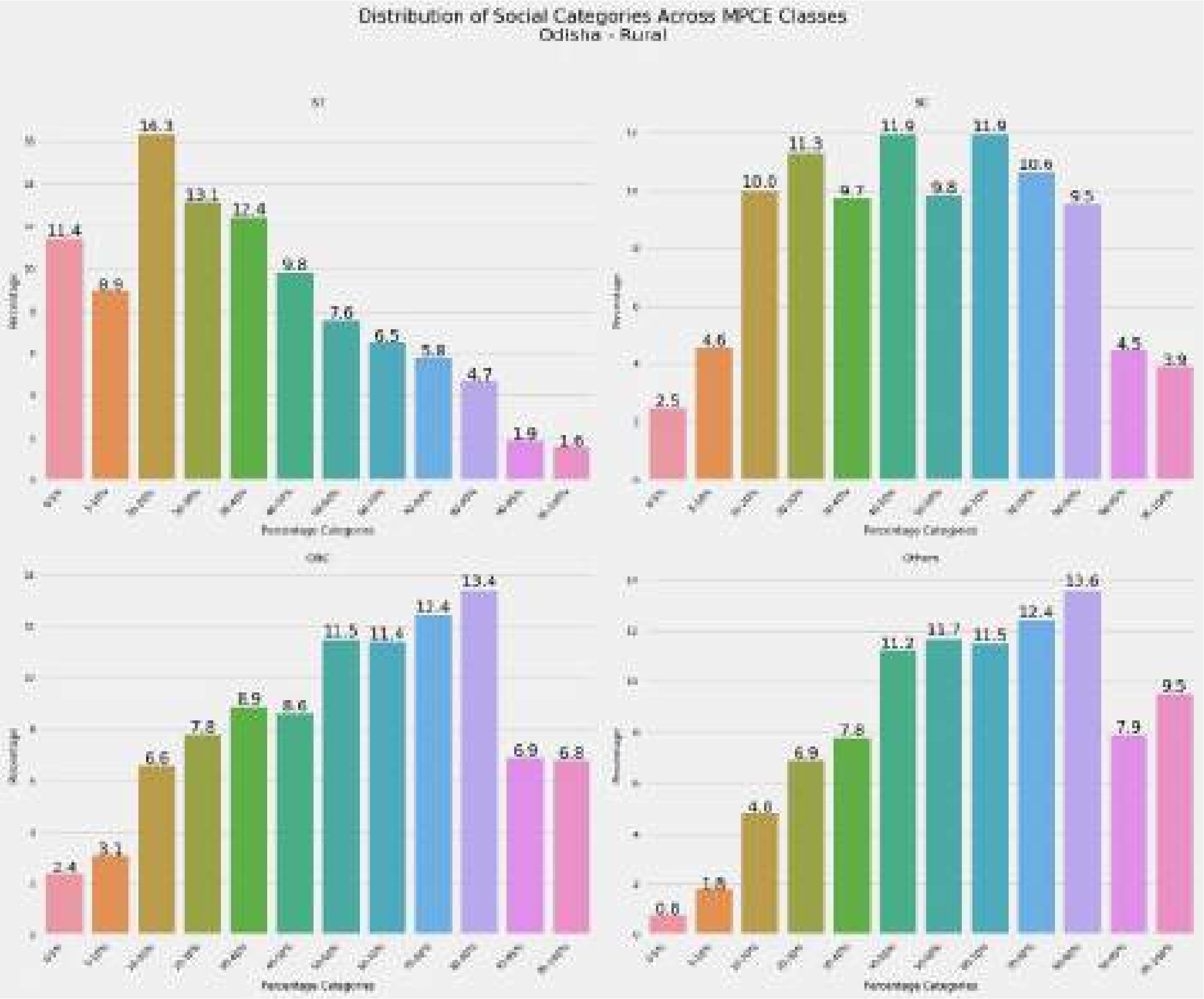

Distribution of Social Categories Across MPCE Classes
Meghalaya - Urban
ST
Percentage
Percentage Categories
6.4
6.1
12.4
12.1
10.3
9.1
10.2
9.1
9.1
7.0
3.5
4.7
SC
Percentage
Percentage Categories
2.1
5.9
4.7
10.5
14.7
18.5
14.8
13.9
3.8
9.6
0.8
0.5
OBC
Percentage
Percentage Categories
0.0
8.6
0.0
6.4
15.8
27.2
0.0
15.8
15.8
10.5
0.0
0.0
Others
Percentage
Percentage Categories
0.2
0.5
2.0
2.3
8.4
11.4
9.2
13.0
13.5
22.0
10.8
6.7

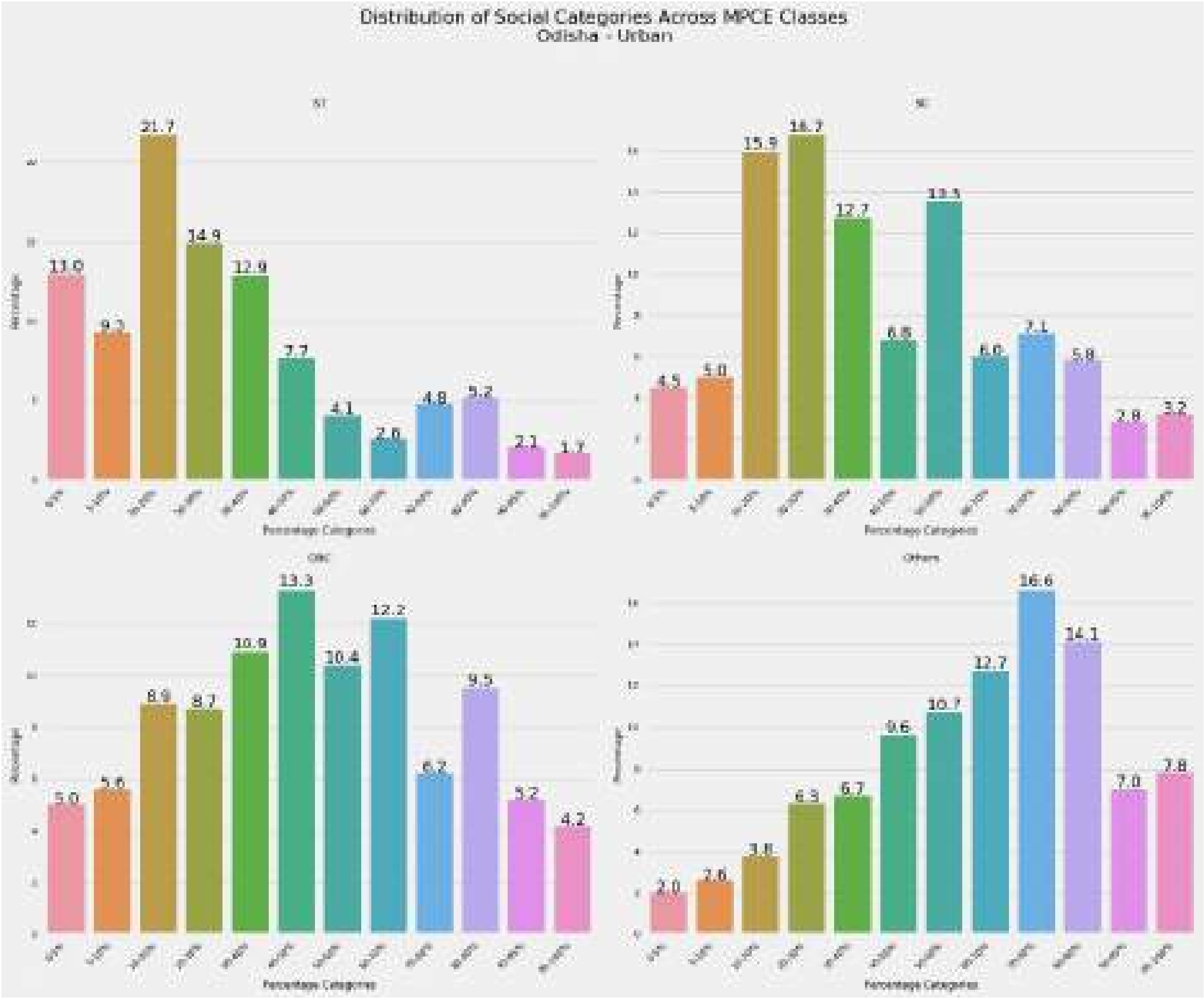

Distribution of Social Categories Across MPCE Classes
Mizoram - Rural
ST
Percentage
4.7
5.1
9.3
9.9
9.9
10.2
10.2
10.1
10.3
10.2
5.1
5.1
Percentage Categories
SC
Percentage
16.8
0.0
45.4
14.4
15.1
3.6
3.6
0.0
0.0
0.0
0.0
1.0
Percentage Categories
OBC
Percentage
24.5
18.9
37.7
18.9
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
Percentage Categories
Others
Percentage
0.04
0.02
0.00
-0.02
-0.04
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
Percentage Categories

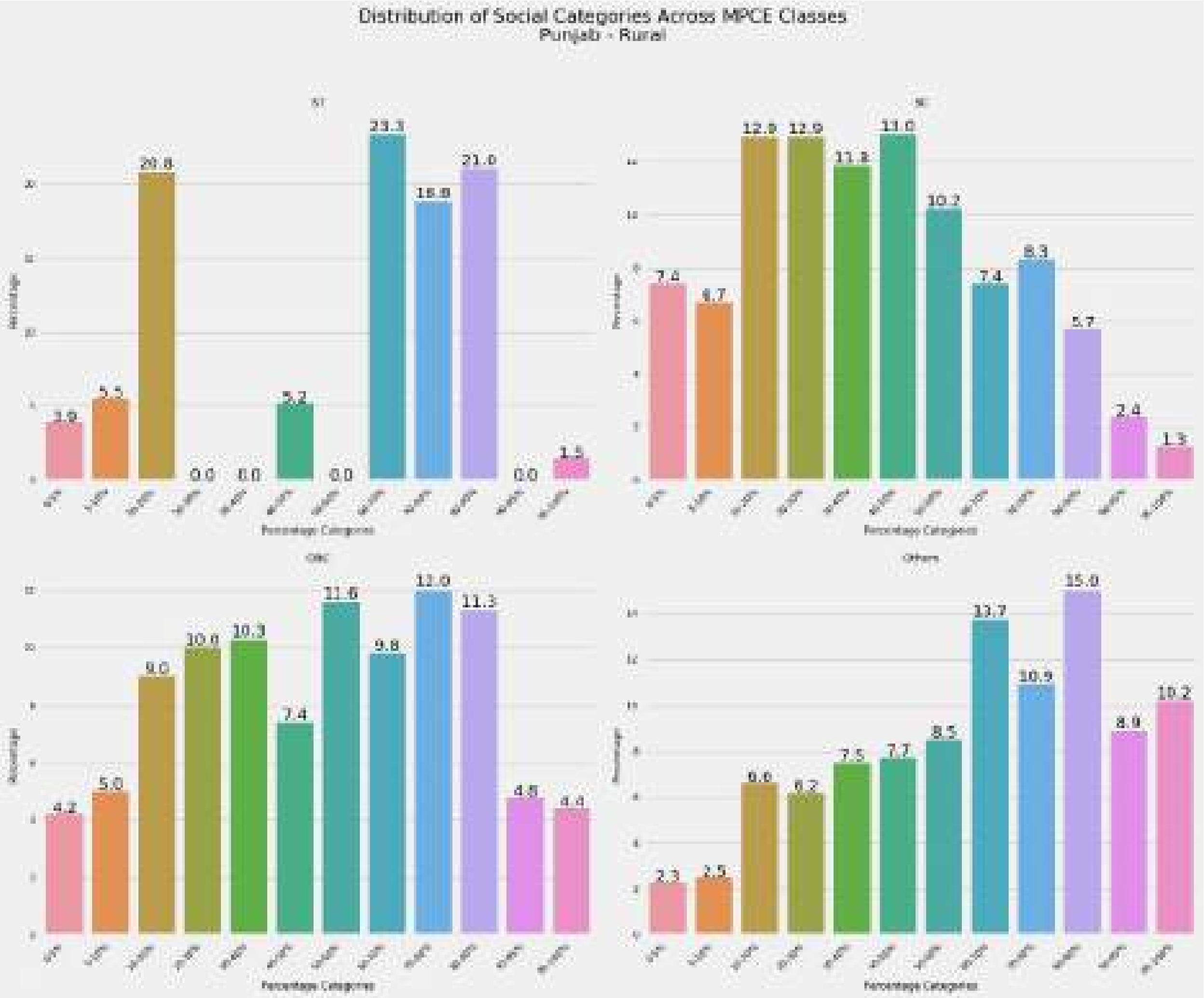

Distribution of Social Categories Across MPCE Classes
Mizoram - Urban
ST
Percentage
5.0
5.1
10.0
10.0
10.1
10.1
9.9
10.0
9.9
10.0
5.0
5.0
Percentage Categories
SC
Percentage
12.6
0.0
11.7
0.0
0.0
0.0
22.8
10.9
17.8
18.7
0.0
6.1
Percentage Categories
OBC
Percentage
0.0
25.2
0.0
0.0
0.0
0.0
0.0
24.6
30.3
0.0
19.9
0.0
Percentage Categories
Others
Percentage
0.0
0.0
0.0
43.9
0.0
0.0
51.3
0.0
4.8
0.0
0.0
0.0
Percentage Categories

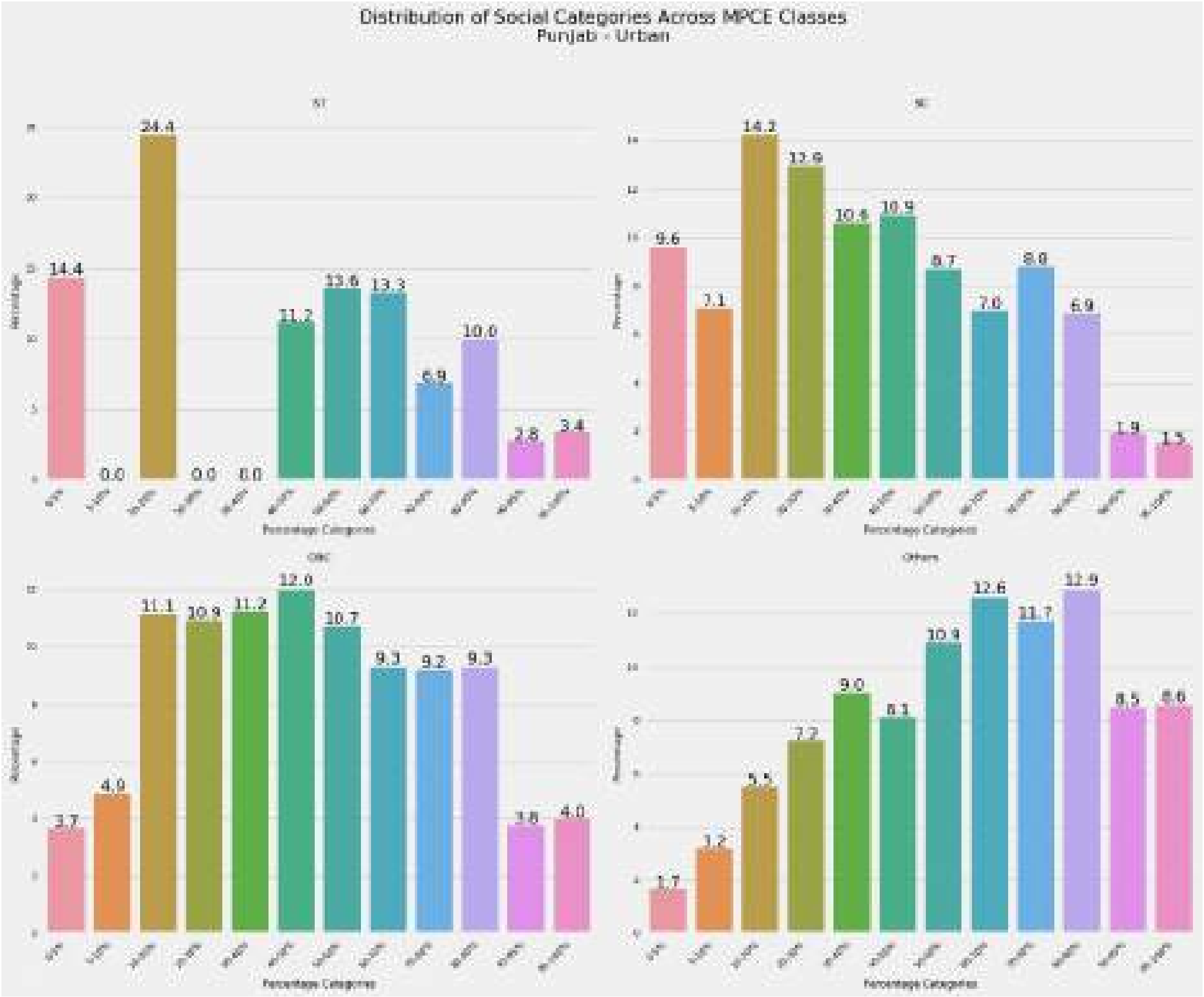

Distribution of Social Categories Across MPCE Classes
Nagaland - Rural
ST
Percentage
0
2
4
6
8
10
5.2
5.1
9.7
10.1
9.9
10.0
9.9
9.8
9.9
10.2
5.1
5.1
Percentage Categories
SC
Percentage
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0.0
1.7
19.9
8.4
8.4
10.0
11.5
15.6
9.3
7.8
6.0
1.3
Percentage Categories
OBC
Percentage
0
5
10
15
20
25
4.7
4.7
4.7
26.6
9.7
13.2
1.6
14.6
7.5
5.5
0.0
7.4
Percentage Categories
Others
Percentage
0
5
10
15
20
0.0
0.0
22.5
0.0
20.5
5.2
16.7
15.6
17.4
0.0
0.0
2.1
Percentage Categories

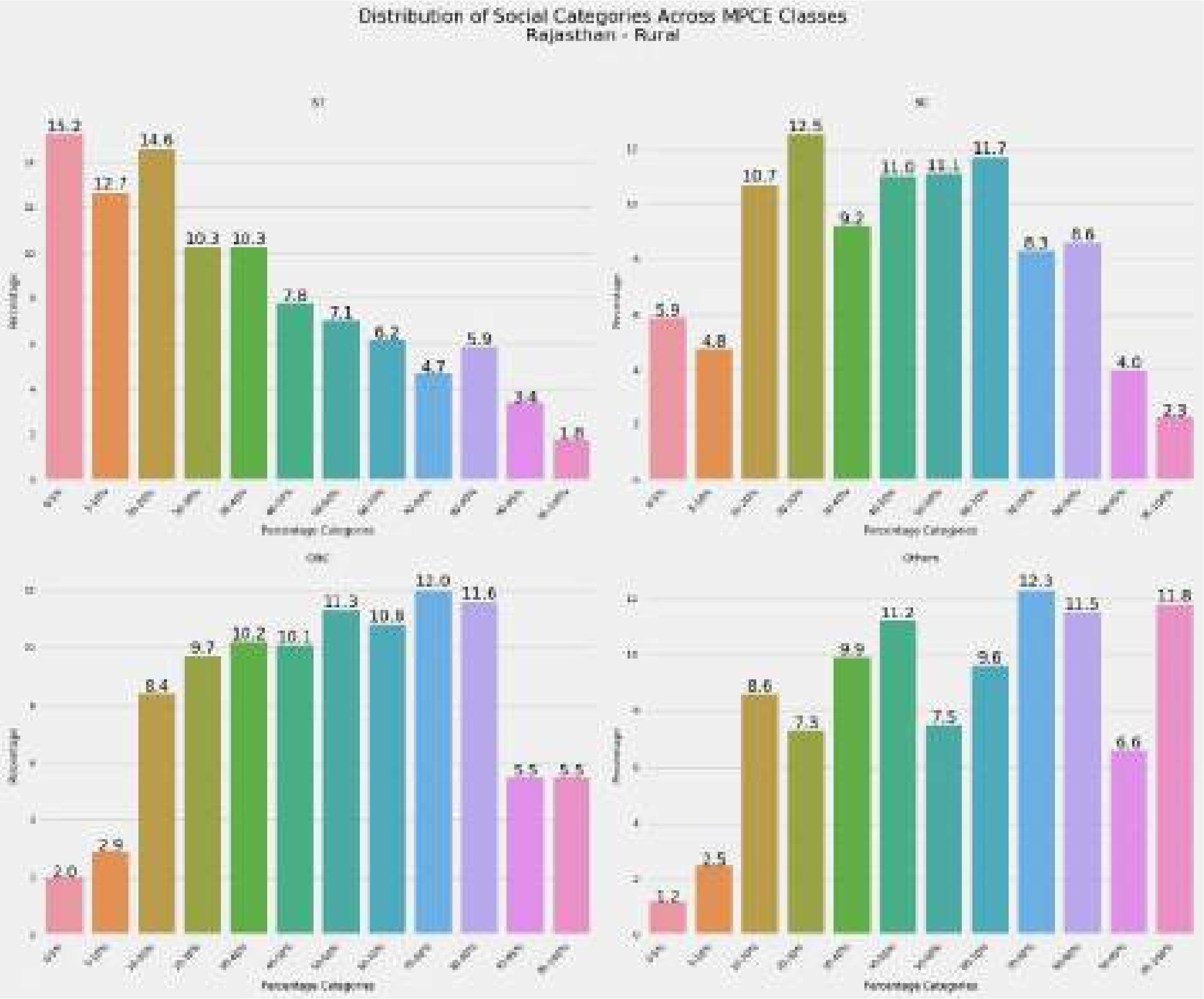

Distribution of Social Categories Across MPCE Classes
Nagaland - Urban
ST
4.1
5.2
9.4
10.4
9.3
10.0
10.5
10.6
10.2
10.4
5.1
4.9
Percentage
Percentage Categories
SC
10.4
1.8
10.1
4.8
11.4
13.7
7.9
3.6
12.3
8.8
5.8
7.4
Percentage
Percentage Categories
OBC
13.0
7.0
21.5
12.1
17.0
4.7
5.0
3.7
2.0
4.3
2.0
7.5
Percentage
Percentage Categories
Others
11.0
1.9
13.5
2.9
12.8
12.7
6.3
13.6
12.9
12.4
0.0
0.0
Percentage
Percentage Categories

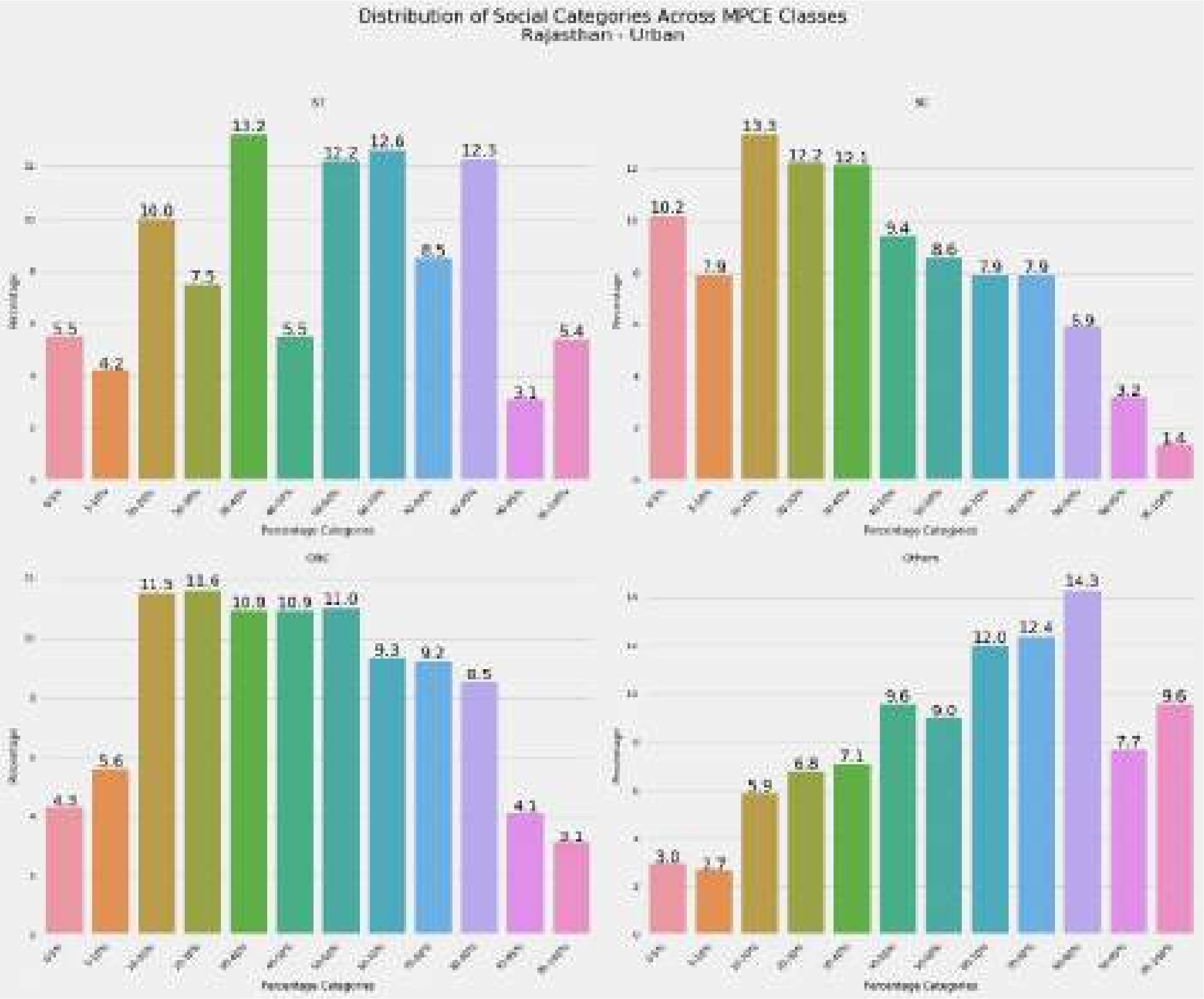

Distribution of Social Categories Across MPCE Classes
Odisha - Rural
ST
Percentage
Percentage Categories
11.4
8.9
16.3
13.1
12.4
9.8
7.6
6.5
5.8
4.7
1.9
1.6
SC
Percentage
Percentage Categories
2.5
4.6
10.0
11.3
9.7
11.9
9.8
11.9
10.6
9.5
4.5
3.9
OBC
Percentage
Percentage Categories
2.4
3.1
6.6
7.8
8.9
8.6
11.5
11.4
12.4
13.4
6.9
6.8
Others
Percentage
Percentage Categories
0.8
1.8
4.8
6.9
7.8
11.2
11.7
11.5
12.4
13.6
7.9
9.5

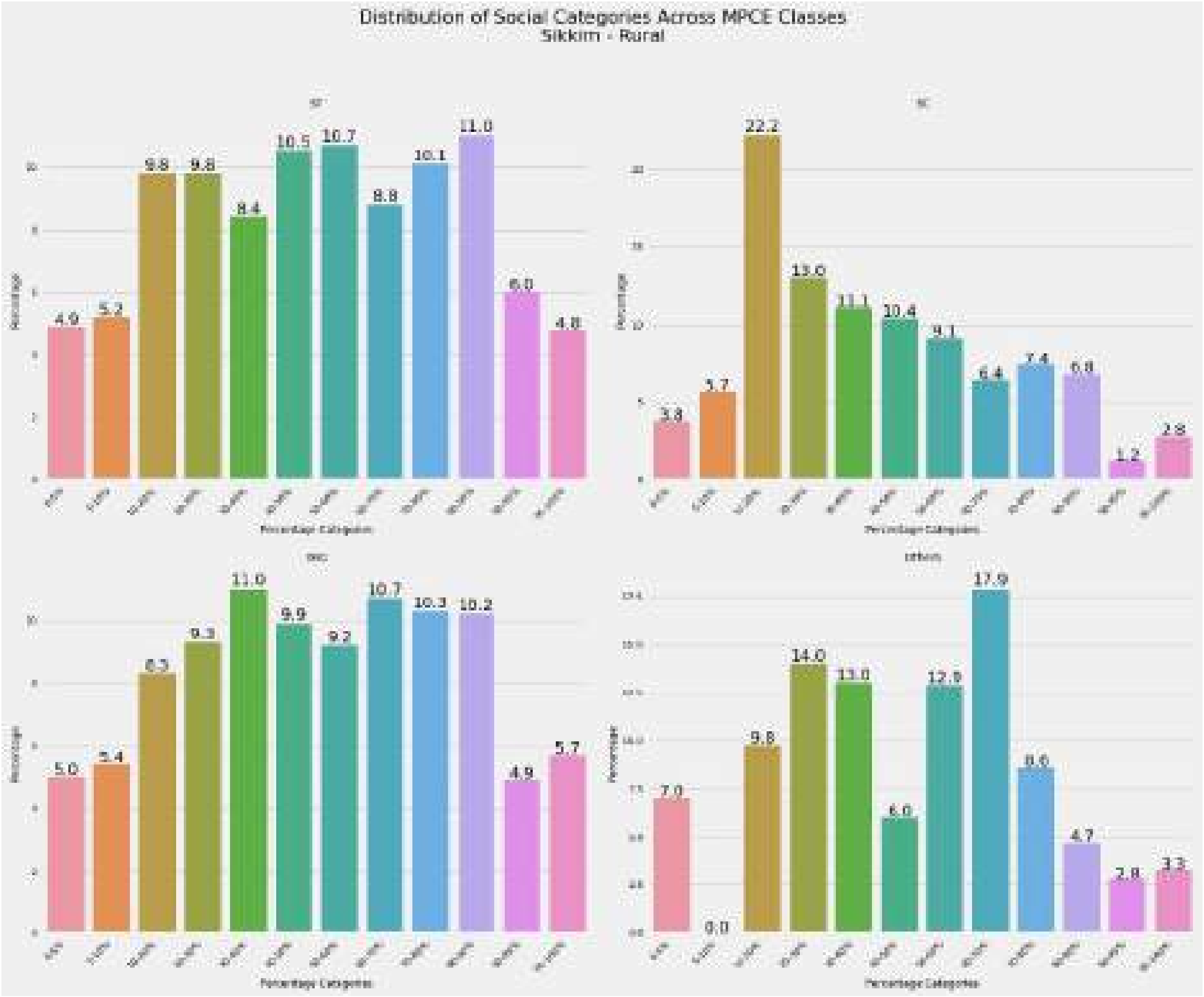

Distribution of Social Categories Across MPCE Classes
Odisha - Urban
ST
Percentage
0
5
10
15
20
13.0
9.3
21.7
14.9
12.9
7.7
4.1
2.6
4.8
5.2
2.1
1.7
Percentage Categories
SC
Percentage
0
2
4
6
8
10
12
14
16
4.5
5.0
15.9
16.7
12.7
6.8
13.5
6.0
7.1
5.8
2.8
3.2
Percentage Categories
OBC
Percentage
0
2
4
6
8
10
12
5.0
5.6
8.9
8.7
10.9
13.3
10.4
12.2
6.2
9.5
5.2
4.2
Percentage Categories
Others
Percentage
0
2
4
6
8
10
12
14
16
2.0
2.6
3.8
6.3
6.7
9.6
10.7
12.7
16.6
14.1
7.0
7.8
Percentage Categories

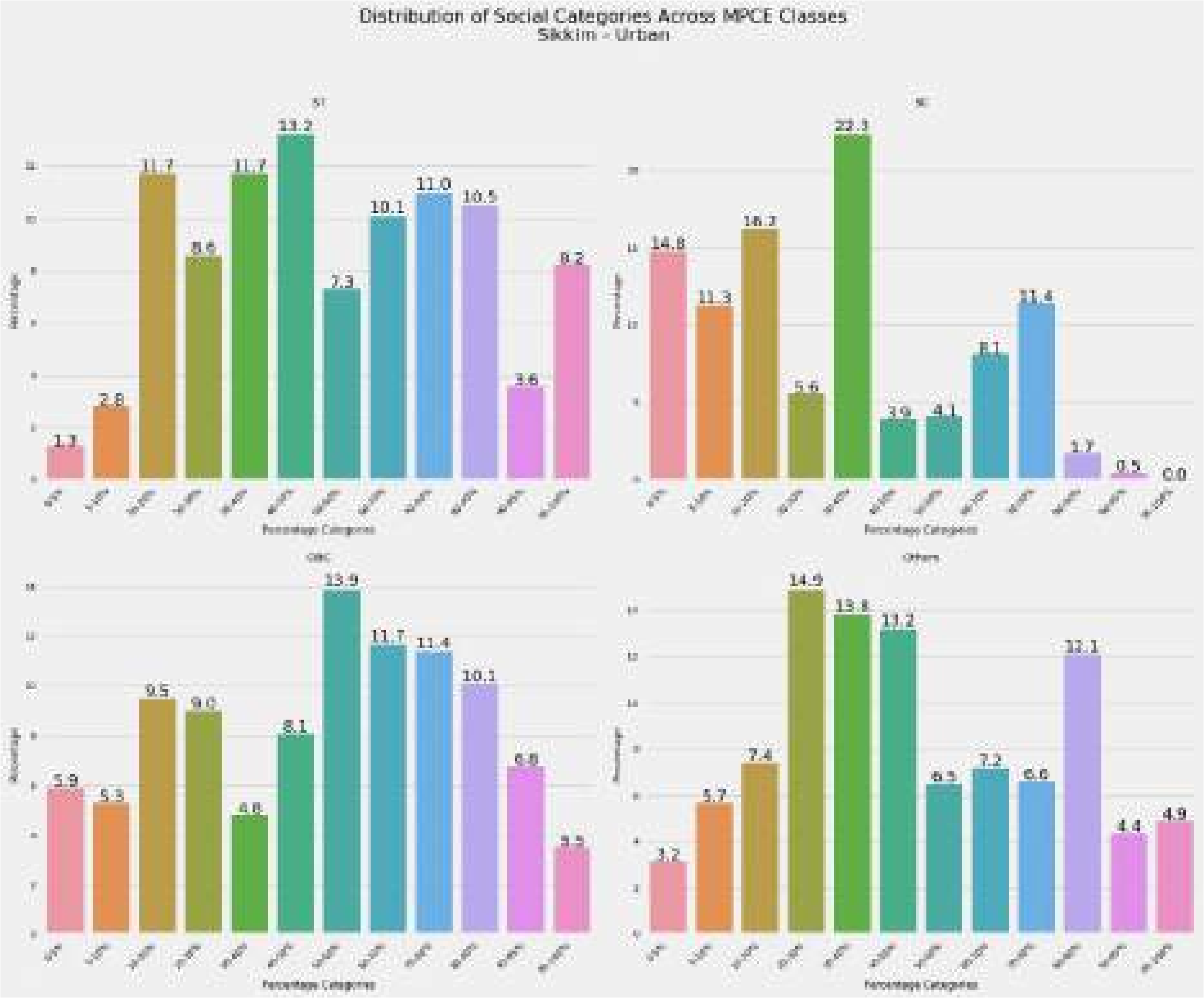

Distribution of Social Categories Across MPCE Classes
Punjab - Rural
ST
Percentage
Percentage Categories
0
5
10
15
20
3.9
5.5
20.8
0.0
0.0
5.2
0.0
23.3
18.8
21.0
0.0
1.5
SC
Percentage
Percentage Categories
0
2
4
6
8
10
12
7.4
6.7
12.9
12.9
11.8
13.0
10.2
7.4
8.3
5.7
2.4
1.3
OBC
Percentage
Percentage Categories
0
2
4
6
8
10
12
4.2
5.0
9.0
10.0
10.3
7.4
11.6
9.8
12.0
11.3
4.8
4.4
Others
Percentage
Percentage Categories
0
2
4
6
8
10
12
14
2.3
2.5
6.6
6.2
7.5
7.7
8.5
13.7
10.9
15.0
8.9
10.2

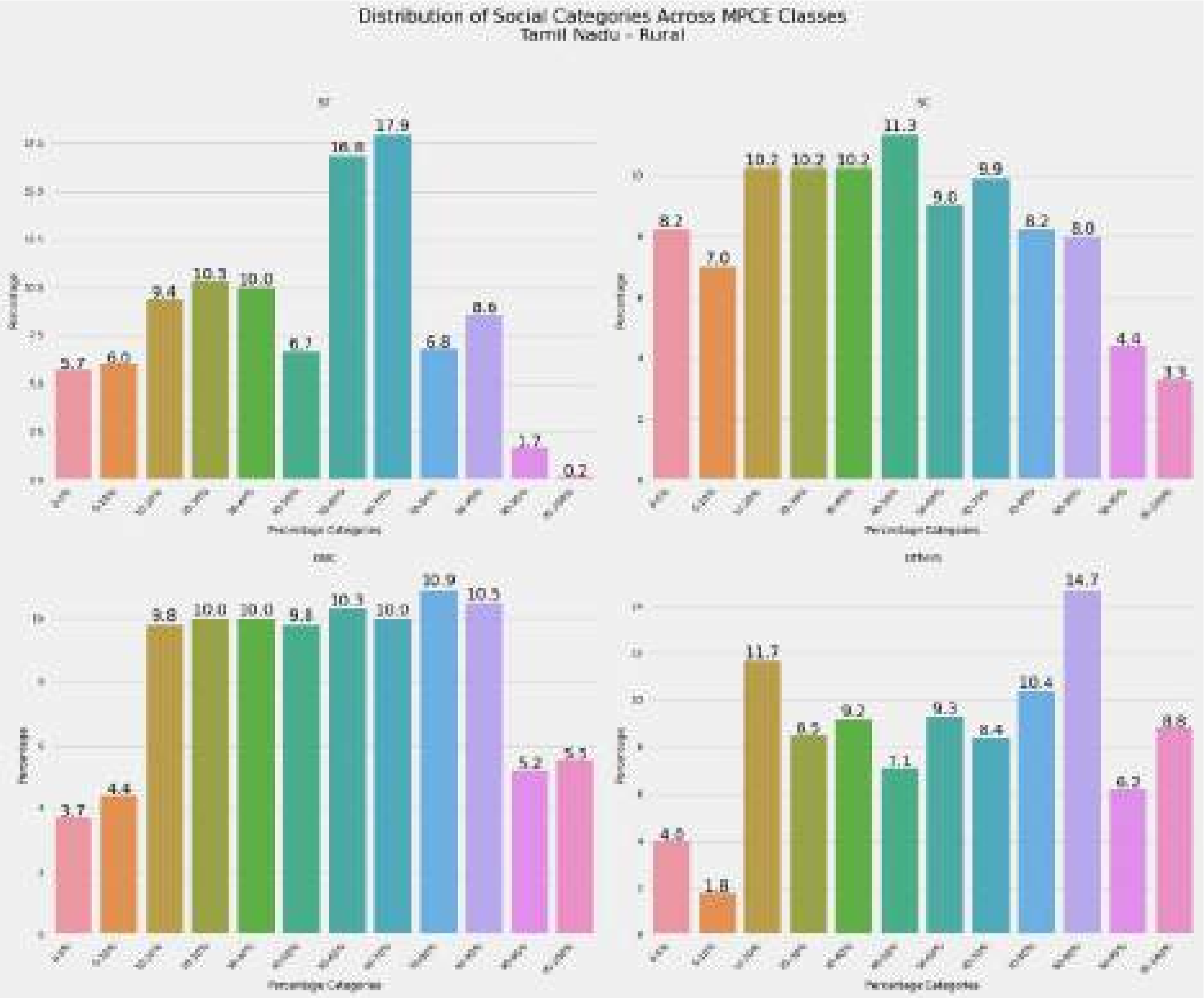
Distribution of Social Categories Across MPCE Classes
Punjab - Urban
ST
Percentage
Percentage Categories
0
5
10
15
20
25
14.4
0.0
24.4
0.0
0.0
11.2
13.6
13.3
6.9
10.0
2.8
3.4
SC
Percentage
Percentage Categories
0
2
4
6
8
10
12
14
9.6
7.1
14.2
12.9
10.6
10.9
8.7
7.0
8.8
6.9
1.9
1.5
OBC
Percentage
Percentage Categories
0
2
4
6
8
10
12
3.7
4.9
11.1
10.9
11.2
12.0
10.7
9.3
9.2
9.3
3.8
4.0
Others
Percentage
Percentage Categories
0
2
4
6
8
10
12
1.7
3.2
5.5
7.2
9.0
8.1
10.9
12.6
11.7
12.9
8.5
8.6

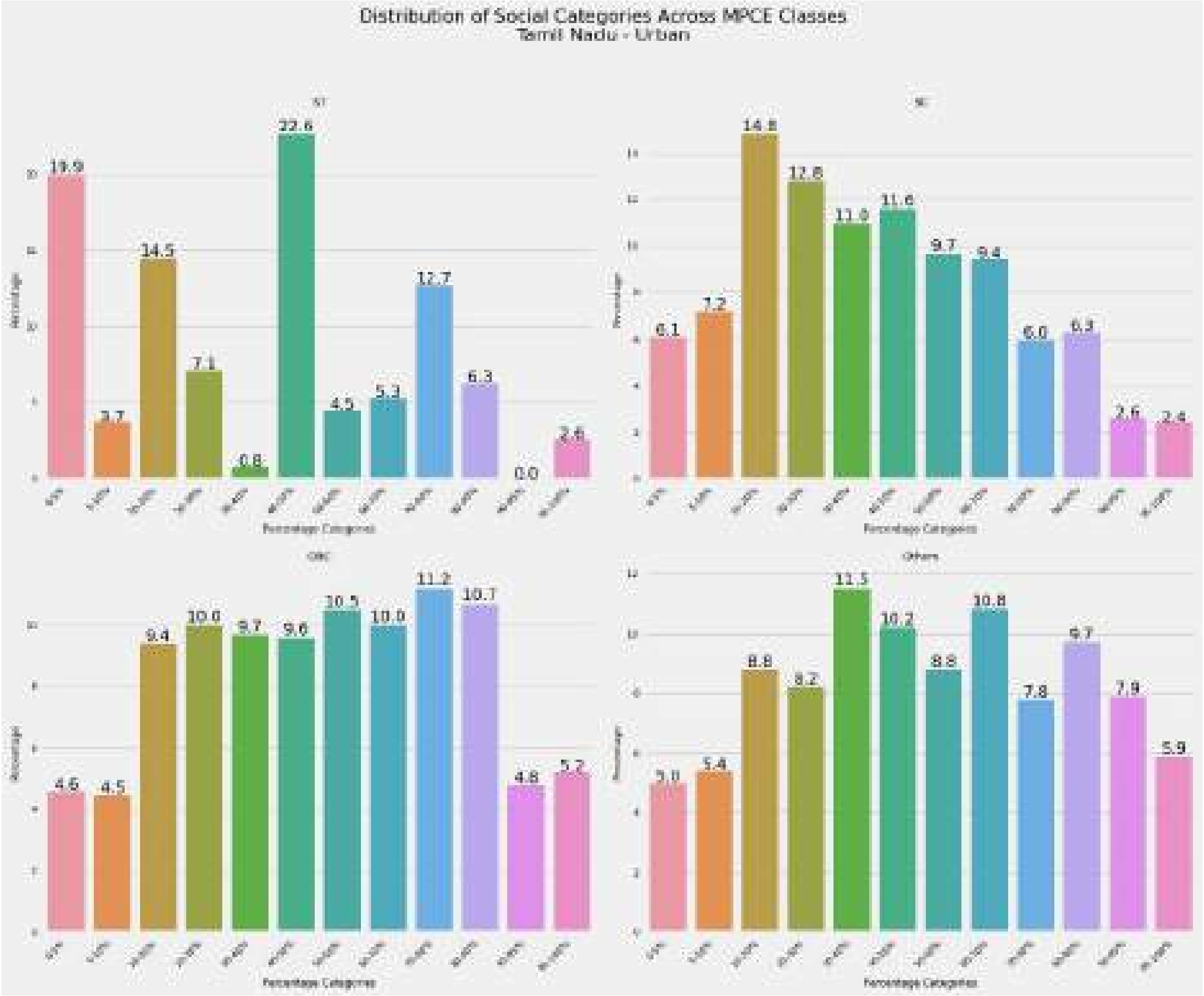

Distribution of Social Categories Across MPCE Classes
Rajasthan - Rural
ST
15.2
12.7
14.6
10.3
10.3
7.8
7.1
6.2
4.7
5.9
3.4
1.8
Percentage
Percentage Categories
SC
5.9
4.8
10.7
12.5
9.2
11.0
11.1
11.7
8.3
8.6
4.0
2.3
Percentage
Percentage Categories
OBC
2.0
2.9
8.4
9.7
10.2
10.1
11.3
10.8
12.0
11.6
5.5
5.5
Percentage
Percentage Categories
Others
1.2
2.5
8.6
7.3
9.9
11.2
7.5
9.6
12.3
11.5
6.6
11.8
Percentage
Percentage Categories

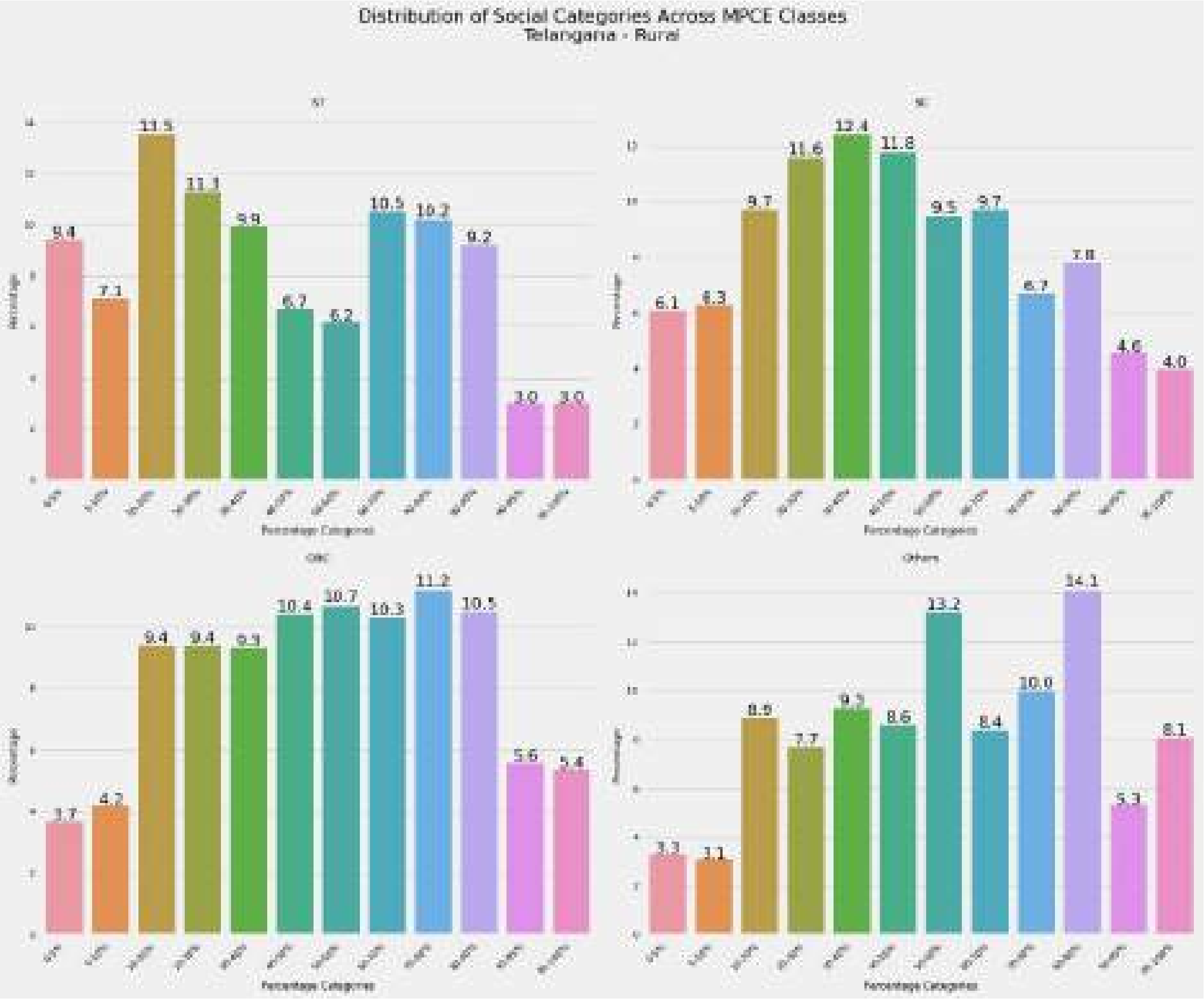

Distribution of Social Categories Across MPCE Classes
Rajasthan - Urban
ST
5.5
4.2
10.0
7.5
13.2
5.5
12.2
12.6
8.5
12.3
3.1
5.4
Percentage
Percentage Categories
SC
10.2
7.9
13.3
12.2
12.1
9.4
8.6
7.9
7.9
5.9
3.2
1.4
Percentage
Percentage Categories
OBC
4.3
5.6
11.5
11.6
10.9
10.9
11.0
9.3
9.2
8.5
4.1
3.1
Percentage
Percentage Categories
Others
3.0
2.7
5.9
6.8
7.1
9.6
9.0
12.0
12.4
14.3
7.7
9.6
Percentage
Percentage Categories

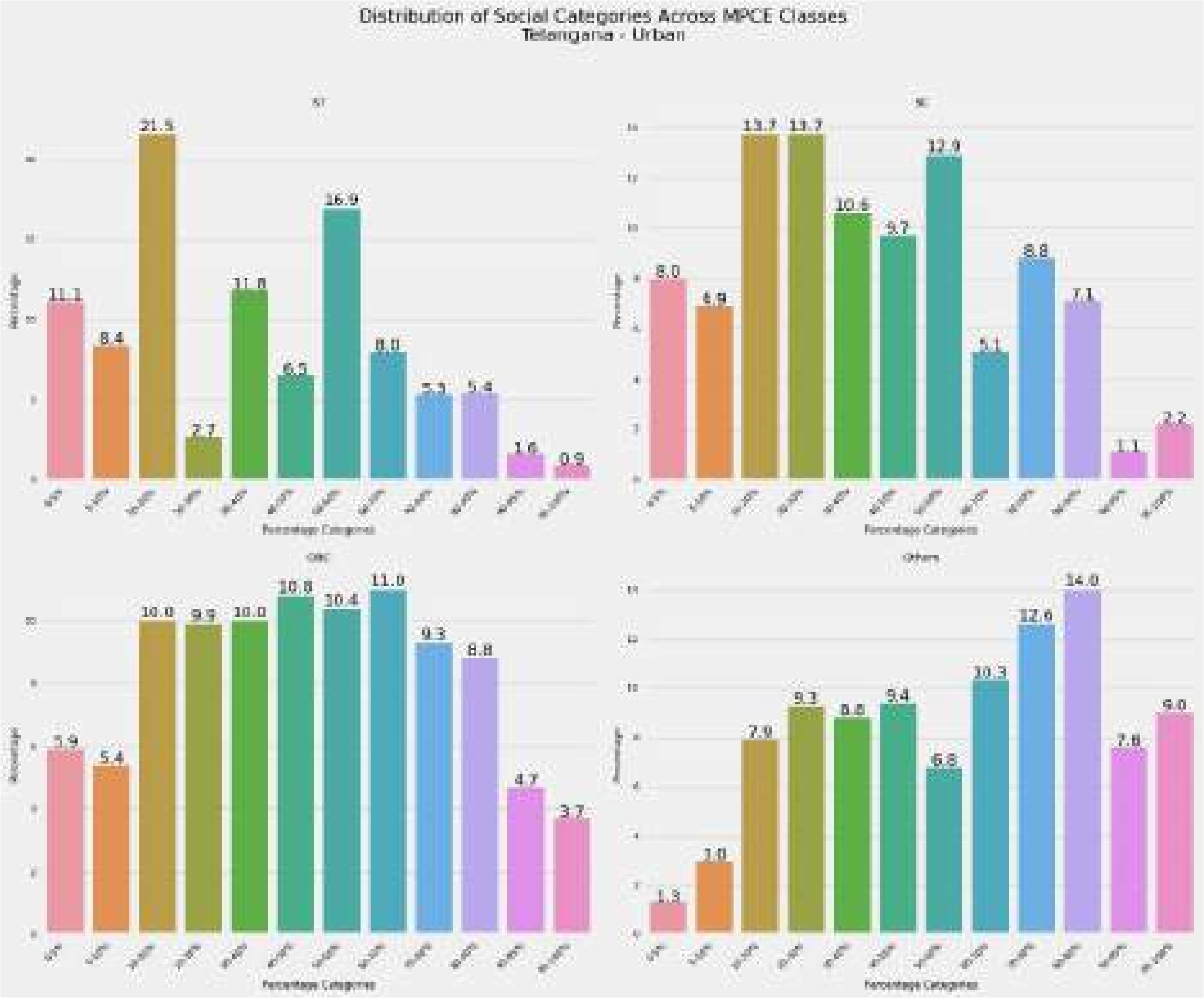

Distribution of Social Categories Across MPCE Classes
Sikkim - Rural
ST
Percentage
4.9
5.2
9.8
9.8
8.4
10.5
10.7
8.8
10.1
11.0
6.0
4.8
Percentage Categories
SC
Percentage
3.8
5.7
22.2
13.0
11.1
10.4
9.1
6.4
7.4
6.8
1.2
2.8
Percentage Categories
OBC
Percentage
5.0
5.4
8.3
9.3
11.0
9.9
9.2
10.7
10.3
10.2
4.9
5.7
Percentage Categories
Others
Percentage
7.0
0.0
9.8
14.0
13.0
6.0
12.9
17.9
8.6
4.7
2.8
3.3
Percentage Categories


Distribution of Social Categories Across MPCE Classes
Sikkim - Urban
ST
1.3
2.8
11.7
8.6
11.7
13.2
7.3
10.1
11.0
10.5
3.6
8.2
Percentage
Percentage Categories
SC
14.8
11.3
16.2
5.6
22.3
3.9
4.1
8.1
11.4
1.7
0.5
0.0
Percentage
Percentage Categories
OBC
5.9
5.3
9.5
9.0
4.8
8.1
13.9
11.7
11.4
10.1
6.8
3.5
Percentage
Percentage Categories
Others
3.2
5.7
7.4
14.9
13.8
13.2
6.5
7.2
6.6
12.1
4.4
4.9
Percentage
Percentage Categories


Distribution of Social Categories Across MPCE Classes
Tamil Nadu - Rural
ST
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Percentage
5.7
6.0
9.4
10.3
10.0
6.7
16.8
17.9
6.8
8.6
1.7
0.2
Percentage Categories
SC
0
2
4
6
8
10
Percentage
8.2
7.0
10.2
10.2
10.2
11.3
9.0
9.9
8.2
8.0
4.4
3.3
Percentage Categories
OBC
0
2
4
6
8
10
Percentage
3.7
4.4
9.8
10.0
10.0
9.8
10.3
10.0
10.9
10.5
5.2
5.5
Percentage Categories
Others
0
2
4
6
8
10
12
14
Percentage
4.0
1.8
11.7
8.5
9.2
7.1
9.3
8.4
10.4
14.7
6.2
8.8
Percentage Categories


Distribution of Social Categories Across MPCE Classes
Tamil Nadu - Urban
ST
19.9
3.7
14.5
7.1
0.8
22.6
4.5
5.3
12.7
6.3
0.0
2.6
Percentage
Percentage Categories
SC
6.1
7.2
14.8
12.8
11.0
11.6
9.7
9.4
6.0
6.3
2.6
2.4
Percentage
Percentage Categories
OBC
4.6
4.5
9.4
10.0
9.7
9.6
10.5
10.0
11.2
10.7
4.8
5.2
Percentage
Percentage Categories
Others
5.0
5.4
8.8
8.2
11.5
10.2
8.8
10.8
7.8
9.7
7.9
5.9
Percentage
Percentage Categories


Distribution of Social Categories Across MPCE Classes
Telangana - Rural
ST
0
2
4
6
8
10
12
14
Percentage
9.4
7.1
13.5
11.1
9.9
6.7
6.2
10.5
10.2
9.2
3.0
3.0
Percentage Categories
SC
0
2
4
6
8
10
12
Percentage
6.1
6.3
9.7
11.6
12.4
11.8
9.5
9.7
6.7
7.8
4.6
4.0
Percentage Categories
OBC
0
2
4
6
8
10
Percentage
3.7
4.2
9.4
9.4
9.3
10.4
10.7
10.3
11.2
10.5
5.6
5.4
Percentage Categories
Others
0
2
4
6
8
10
12
14
Percentage
3.3
3.1
8.9
7.7
9.3
8.6
13.2
8.4
10.0
14.1
5.3
8.1
Percentage Categories


Distribution of Social Categories Across MPCE Classes
Telangana - Urban
ST
Percentage
11.1
8.4
21.5
2.7
11.8
6.5
16.9
8.0
5.3
5.4
1.6
0.9
Percentage Categories
SC
Percentage
8.0
6.9
13.7
13.7
10.6
9.7
12.9
5.1
8.8
7.1
1.1
2.2
Percentage Categories
OBC
Percentage
5.9
5.4
10.0
9.9
10.0
10.8
10.4
11.0
9.3
8.8
4.7
3.7
Percentage Categories
Others
Percentage
1.3
3.0
7.9
9.3
8.8
9.4
6.8
10.3
12.6
14.0
7.6
9.0
Percentage Categories

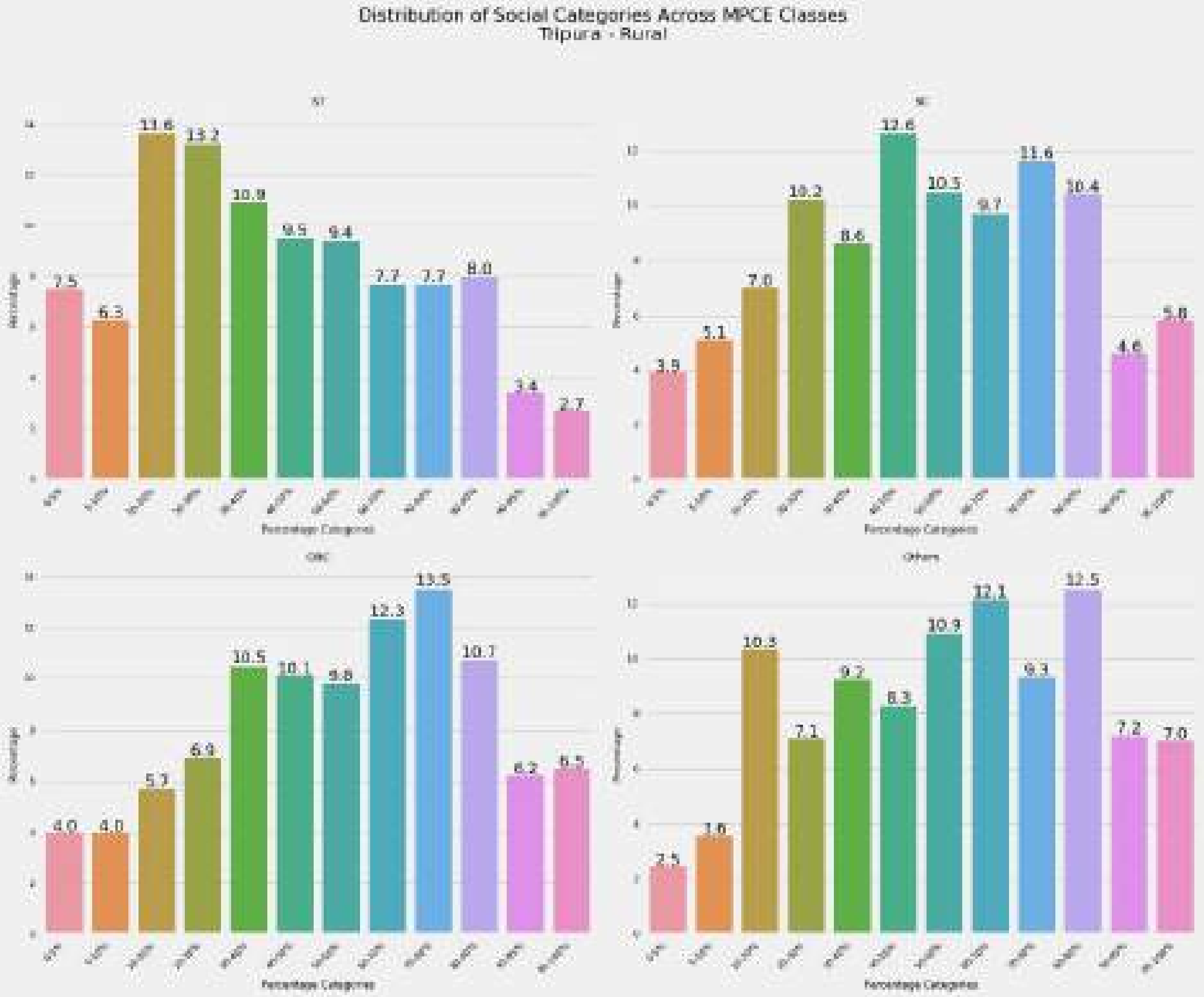


Figure 9 Average MPCE (Rs) by social categories

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# List of columns to plot
social_columns = ['ST', 'SC', 'OBC', 'Others', 'All']

# Get unique states
states = data_a9['state'].unique()

# Create plots for each state
for state in states:
    # Create a figure with two subplots (Rural and Urban)
    fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12, 10))

    # Process Rural data
    rural_data = data_a9[(data_a9['state'] == state) &
```

```
                        (data_a9['col0'] == 'Rural') &
                        (data_a9['col2'] == 'Avg. MPCE (Rs.)')]

    # Process Urban data
    urban_data = data_a9[(data_a9['state'] == state) &
                        (data_a9['col0'] == 'Urban') &
                        (data_a9['col2'] == 'Avg. MPCE (Rs.)')]

    if not rural_data.empty and not urban_data.empty:
        # Plot Rural data
        rural_values = rural_data[social_columns].values[0]
        sns.barplot(x=social_columns, y=rural_values, ax=ax1)

        # Customize Rural subplot
        ax1.set_title(f'{state} - Rural', pad=10)
        ax1.set_ylabel('Average MPCE (Rs.)')
        ax1.set_xticklabels(social_columns, rotation=45,
horizontalalignment='right')

        # Add value labels on Rural bars
        for i, v in enumerate(rural_values):
            if pd.notna(v):  # Check if value is not NaN
                ax1.text(i, v + v*0.01, f'{v:.0f}', ha='center')

        # Plot Urban data
        urban_values = urban_data[social_columns].values[0]
        sns.barplot(x=social_columns, y=urban_values, ax=ax2)

        # Customize Urban subplot
        ax2.set_title(f'{state} - Urban', pad=10)
        ax2.set_ylabel('Average MPCE (Rs.)')
        ax2.set_xticklabels(social_columns, rotation=45,
horizontalalignment='right')

        # Add value labels on Urban bars
        for i, v in enumerate(urban_values):
            if pd.notna(v):  # Check if value is not NaN
                ax2.text(i, v + v*0.01, f'{v:.0f}', ha='center')

        # Add main title for the state
        fig.suptitle(f'Average MPCE (Rs.) by Social Category in
{state}',
                    fontsize=16, y=1.02)

        # Adjust layout
        plt.tight_layout()

        # Show plot
        plt.show()
        # Optional: Add a small pause between plots
        plt.pause(0.5)
```