SOFTWARE AND CODE DOCUMENTATION

Tools Used:

- Python(Jupyter Notebooks) – for data cleaning, analysis and visualization
- Pandas,NumPy – for creating manipulation
- Matplotlib, Seaborn – for creating visualizations

Data Cleaning and Processing , EDA Code Snippet:

Data Cleaning and Saving CSV files

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [6]:
```
!pip install openpyxl
```

```
Collecting openpyxl
  Using cached openpyxl-3.1.5-py2.py3-none-any.whl.metadata (2.5 kB)
Collecting et-xmlfile (from openpyxl)
  Using cached et_xmlfile-2.0.0-py3-none-any.whl.metadata (2.7 kB)
Using cached openpyxl-3.1.5-py2.py3-none-any.whl (250 kB)
Using cached et_xmlfile-2.0.0-py3-none-any.whl (18 kB)
Installing collected packages: et-xmlfile, openpyxl
Successfully installed et-xmlfile-2.0.0 openpyxl-3.1.5
```

In [7]:
```
pd.read_excel('Datasets/HCES/Avg_MPCE.xlsx',skiprows=2)
```

Out[7]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unna
0	Food	NaN	NaN	NaN	NaN	NaN	
1	Non-food	NaN	NaN	NaN	NaN	NaN	
2	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN	NaN	NaN	

3 rows × 37 columns

In [8]:
```
pd.read_excel('Datasets/HCES/Abs_Perc.xlsx')
```

Out[8]:

	Statement 4: Absolute and percentage break-up of MPCE by item groups in 2023-24: All-India	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Ur
0	Item group	MPCE (Rs.)	NaN	NaN	NaN	NaN	
1	NaN	Rural	NaN	NaN	NaN	NaN	
2	cereals & cereal substitutes	206	NaN	NaN	NaN	NaN	
3	pulses & their products*	84	NaN	NaN	NaN	NaN	
4	sugar & salt	37	NaN	NaN	NaN	NaN	
5	milk & milk products	348	NaN	NaN	NaN	NaN	
6	vegetables	248	NaN	NaN	NaN	NaN	
7	fruits	158	NaN	NaN	NaN	NaN	
8	egg, fish & meat	203	NaN	NaN	NaN	NaN	
9	edible oil	114	NaN	NaN	NaN	NaN	
10	spices	135	NaN	NaN	NaN	NaN	
11	beverages, refreshments, processed food#	406	NaN	NaN	NaN	NaN	
12	food total	1939	NaN	NaN	NaN	NaN	
13	pan, tobacco & intoxicants	158	NaN	NaN	NaN	NaN	
14	fuel and light	252	NaN	NaN	NaN	NaN	
15	education	133	NaN	NaN	NaN	NaN	
16	medical	282	NaN	NaN	NaN	NaN	
17	conveyance	313	NaN	NaN	NaN	NaN	
18	consumer services excluding conveyance	217	NaN	NaN	NaN	NaN	
19	misc. goods, entertainment	256	NaN	NaN	NaN	NaN	
20	rent	23	NaN	NaN	NaN	NaN	

	Statement 4: Absolute and percentage break-up of MPCE by item groups in 2023-24: All-India	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Ur
21	taxes and cesses	9	NaN	NaN	NaN	NaN	
22	clothing, bedding & footwear	273	NaN	NaN	NaN	NaN	
23	durable goods	267	NaN	NaN	NaN	NaN	
24	non-food total	2183	NaN	NaN	NaN	NaN	
25	all items	4122	NaN	NaN	NaN	NaN	
26	*includes gram #includes purchased cooked...	NaN	NaN	NaN	NaN	NaN	
27	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN	NaN	NaN	

28 rows × 27 columns

In [11]:
```
pd.read_csv('Datasets/PLFS/employment_industry_extracted.csv')
```

Out[11]:

	industry \nas per NIC-2008 (2 digit NIC Division codes and corresponding descriptions)\n	self-employed	Unnamed: 2	Unnamed: 3	regular\nwage/\nsalary\n	c
0	NaN	own account worker, employer	helper in household enterprise	all self\nemployed\n	NaN	
1	(1)	(2)	(3)	(4)	(5)	
2	01-03 (agriculture)	61.6	22	83.6	1	
3	05-09 (mining & quarrying)	10.8	2	12.8	54.3	
4	10-33 (manufacturing)	29.3	4.9	34.3	51.4	
...	...	...	...	...	...	
170	55-56 (accommodation & food services)	40.5	16.5	57	33.5	
171	58-99 (other services)	20.6	1.7	22.4	75.8	
172	45-99 (tertiary)	38.3	6.6	44.9	51	
173	total	39	19.4	58.4	21.7	
174	sample workers	73565	31425	104990	47036	

175 rows × 7 columns

In [12]:
```
file_path = 'Datasets/PLFS/PLFS(2023-24)/employment_industry.xlsx'
xls = pd.ExcelFile(file_path)

xls.sheet_names
```

Out[12]:
```
['Sheet1', 'Sheet2', 'Sheet3']
```

In [13]:
```
df = pd.read_excel(xls, sheet_name = 'Sheet1', header=None)
df.head(20)
```

Out[13]:

	0	1	2	3	4	
0	Table (20): Percentage distribution of workers...	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	
3	industry \nas per NIC-2008 (2 digit NIC Divis...	self-employed	NaN	NaN	regular\nwage/ \nsalary\n	casual'
4	NaN	own account worker, employer	helper in household enterprise	all self\nemployed\n	NaN	
5	(1)	(2)	(3)	(4)	(5)	
6	01-03 (agriculture)	61.6	22	83.6	1	
7	05-09 (mining & quarrying)	10.8	2	12.8	54.3	
8	10-33 (manufacturing)	29.3	4.9	34.3	51.4	
9	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	
10	41-43 (construction)	12.8	0.5	13.3	3.6	
11	05-43 (secondary)	17.8	1.9	19.6	20.2	
12	45-47 (trade)	63	7.4	70.4	26.5	
13	49-53(transport)	53.3	0.5	53.9	34.1	
14	55-56 (accommodation & food services))	49	10.2	59.3	31.8	
15	58-99 (other services)	29.1	2	31	66.5	
16	45-99 (tertiary)	49.2	4.3	53.5	41.2	
17	total	47	12.4	59.4	15.8	
18	sample workers	33350	8170	41520	12126	
19	NaN	NaN	NaN	NaN	NaN	

In [17]:
```
df
```

Out[17]:

	0	1	2	3	4	
0	Table (20): Percentage distribution of workers...	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	
3	industry \nas per NIC-2008 (2 digit NIC Divis...	self-employed	NaN	NaN	regular\nwage/ \nsalary\n	casua
4	NaN	own account worker, employer	helper in household enterprise	all self\nemployed\n	NaN	
...	...	...	...	...	...	
174	55-56 (accommodation & food services)	40.5	16.5	57	33.5	
175	58-99 (other services)	20.6	1.7	22.4	75.8	
176	45-99 (tertiary)	38.3	6.6	44.9	51	
177	total	39	19.4	58.4	21.7	
178	sample workers	73565	31425	104990	47036	

179 rows × 7 columns

In [18]:

```
def remove_headers(df):
    df_clean = df[df.apply(lambda row: np.any(pd.to_numeric(row,errors='coer
    df_clean.reset_index(drop=True, inplace=True)
    return df_clean

df_cleaned = remove_headers(df)
df_cleaned
```

Out[18]:

	0	1	2	3	4	5	6
0	01-03 (agriculture)	61.6	22	83.6	1	15.4	100
1	05-09 (mining & quarrying)	10.8	2	12.8	54.3	32.9	100
2	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	100
3	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	100
4	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	100
...	...	...	...	...	...	...	...
112	55-56 (accommodation & food services)	40.5	16.5	57	33.5	9.5	100
113	58-99 (other services)	20.6	1.7	22.4	75.8	1.8	100
114	45-99 (tertiary)	38.3	6.6	44.9	51	4.1	100
115	total	39	19.4	58.4	21.7	19.8	100
116	sample workers	73565	31425	104990	47036	31044	183070

117 rows × 7 columns

In [19]:
```
df_cleaned.to_csv('PLFS_emp_industry.csv', index=False)
```

In [28]:
```
emp_industry = pd.read_csv('PLFS_emp_industry.csv')
```

In [29]:
```
emp_industry.head()
```

Out[29]:

	0	1	2	3	4	5	6
0	01-03 (agriculture)	61.6	22.0	83.6	1.0	15.4	100
1	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	100
2	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	100
3	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	100
4	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	100

In [30]:
```
col_names = ['industry','Own Account Worker, Employer','Helper in Household
emp_industry.columns = col_names
```

In [23]:
```
df
```

Out[23]:

	industry	Own Account Worker, Employer	Helper in Household Enterprise	All Self-Employed	Regular Wage/Salary	Casual Labour
0	01-03 (agriculture)	61.6	22.0	83.6	1.0	15.4
1	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9
2	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3
3	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4
4	41-43 (construction)	12.8	0.5	13.3	3.6	83.1
...	...	...	...	...	...	...
112	55-56 (accommodation & food services)	40.5	16.5	57.0	33.5	9.5
113	58-99 (other services)	20.6	1.7	22.4	75.8	1.8
114	45-99 (tertiary)	38.3	6.6	44.9	51.0	4.1
115	total	39.0	19.4	58.4	21.7	19.8
116	sample workers	73565.0	31425.0	104990.0	47036.0	31044.0

117 rows × 7 columns

In [31]:
```
df = pd.read_excel('Datasets/PLFS/PLFS(2023-24)/employment_state.xlsx', head
```

In [32]:
```
def remove_headers(df):
    df_clean = df[df.apply(lambda row: np.any(pd.to_numeric(row,errors='coer
    df_clean.reset_index(drop=True, inplace=True)
    return df_clean

df_cleaned = remove_headers(df)
df_cleaned
```

Out[32]:

	0	1	2	3	4	5	6
0	Andhra Pradesh	47.8	6.8	54.6	16.1	29.3	100
1	Arunachal Pradesh	60.1	12.8	72.9	19.9	7.2	100
2	Assam	47.9	6.1	54	17.2	28.8	100
3	Bihar	50.4	11.3	61.7	8.4	29.9	100
4	Chhattisgarh	46.9	20.7	67.6	13.2	19.3	100
...	...	...	...	...	...	...	...
325	Jammu & Kashmir	51.6	15.1	66.7	22.1	11.1	100
326	Ladakh	57.4	8.3	65.7	25.5	8.8	100
327	Lakshadweep	29	1.3	30.3	45.4	24.3	100
328	Puducherry	19.7	1.9	21.6	56.9	21.5	100
329	all India	39	19.4	58.4	21.7	19.8	100

330 rows × 7 columns

In [35]:
```
col_names = ['State/UT','Own account, worker, employer','Helper in household
df_cleaned.columns=col_names
```

In [36]:
```
df_cleaned
```

Out[36]:

	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All
0	Andhra Pradesh	47.8	6.8	54.6	16.1	29.3	100
1	Arunachal Pradesh	60.1	12.8	72.9	19.9	7.2	100
2	Assam	47.9	6.1	54	17.2	28.8	100
3	Bihar	50.4	11.3	61.7	8.4	29.9	100
4	Chhattisgarh	46.9	20.7	67.6	13.2	19.3	100
...	...	...	...	...	...	...	...
325	Jammu & Kashmir	51.6	15.1	66.7	22.1	11.1	100
326	Ladakh	57.4	8.3	65.7	25.5	8.8	100
327	Lakshadweep	29	1.3	30.3	45.4	24.3	100
328	Puducherry	19.7	1.9	21.6	56.9	21.5	100
329	all India	39	19.4	58.4	21.7	19.8	100

330 rows × 7 columns

In [37]:
```
df_cleaned.to_csv('PLFS_emp_state.csv')
```

In [44]:
```
file_path = 'Datasets/PLFS/PLFS(2023-24)/LFPR.xlsx'
age_15_29 = pd.read_excel(file_path, sheet_name='Age_15_29', header=None)
age_15_59 = pd.read_excel(file_path, sheet_name='Age_15_59', header=None)
age_15_above = pd.read_excel(file_path, sheet_name='Age_15_above', header=Nc
all_ages = pd.read_excel(file_path, sheet_name='All_age', header=None)
```

In [45]:
```
age_15_29 = remove_headers(age_15_29)
age_15_59 = remove_headers(age_15_59)
age_15_above = remove_headers(age_15_above)
all_ages = remove_headers(all_ages)
```

In [49]:
```
cols = pd.MultiIndex.from_tuples([
    ("", "State/UT"),
    ("Rural","Male"),('Rural','Female'),('Rural','Person'),
    ('Urban','Male'),('Urban','Female'),('Urban','Person'),
    ('Rural + Urban','Male'),('Rural + Urban','Female'),('Rural + Urban','Pe

])
age_15_29.columns = cols
age_15_59.columns = cols
age_15_above.columns = cols
all_ages.columns = cols
```

In [52]:
```
all_ages
```

Out[52]:

		Rural			Urban			Rural	
	State/UT	Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	59.7	40.7	50.1	59.3	24.5	41.3	59.5	35.8
1	Arunachal Pradesh	59.3	53.1	56.3	56	34.5	45.1	58.7	49.9
2	Assam	60.7	38.2	49.6	64.3	25.5	45	61.1	36.8
3	Bihar	48.1	21.1	34.9	48.2	12	30.9	48.1	20.3
4	Chhattisgarh	65.2	50.4	57.8	63.6	27.9	46.4	64.9	46.1
5	Delhi	54.7	13.8	37.8	54	14.5	35.9	54	14.5
6	Goa	58.1	21.8	40.5	54.9	23.7	39.5	56.3	22.9
7	Gujarat	62.8	44.3	53.8	62	23.6	43.7	62.5	35.8
8	Haryana	51.8	20.2	37.1	56.3	16.7	37.9	53.5	18.8
9	Himachal Pradesh	64.9	58.7	61.7	64.2	35.7	51.4	64.8	56.2
10	Jharkhand	53.1	40.4	46.8	52.1	15	34	52.9	35.8
11	Karnataka	60	34.6	47.3	59.8	23.6	42.1	59.9	30.5
12	Kerala	59.7	36.2	47.3	58.4	30.3	43.3	59.1	33.4
13	Madhya Pradesh	63.2	45.4	54.6	58.7	22.3	40.7	62	39.4
14	Maharashtra	61	38	49.6	61	23.7	43.1	61	32
15	Manipur	52.8	36.5	44.6	52.4	35.7	43.9	52.7	36.3
16	Meghalaya	54.8	48.7	51.7	53.1	37.8	45.1	54.6	47.1
17	Mizoram	51.1	30.5	41	49.2	30.2	39.6	50.2	30.4
18	Nagaland	57.3	45.2	51.1	52	35.8	44.1	55.7	42.7
19	Odisha	61.9	40.3	50.8	59.1	24.4	42	61.5	38
20	Punjab	62.4	27.7	45.2	62.6	19	41.4	62.5	24.4
21	Rajasthan	56.4	43	49.6	56.8	23.5	40.9	56.5	38
22	Sikkim	67.1	65.5	66.4	61.1	26.2	46	65.7	56.9
23	Tamil Nadu	60.4	44.5	52.3	59.2	24.4	41.3	59.8	35.2
24	Telangana	61.4	44	52.4	57.5	24.5	41.3	59.8	36.5
25	Tripura	64.8	39.1	51.9	57.3	25.9	41.3	63.5	36.8
26	Uttarakhand	56.6	41.8	49.3	55.6	18.6	37.2	56.4	35.9
27	Uttar Pradesh	54.7	28.1	41.5	56.9	13.5	36.2	55.2	25.2
28	West Bengal	63.6	33.7	48.6	64.3	26.8	45.4	63.8	31.7

				Rural			Urban	Rural	
	State/UT	Male	Female	Person	Male	Female	Person	Male	Female
29	Andaman & N. Island	67.5	44.6	56.5	67.9	30	49.3	67.7	38
30	Chandigarh	NaN	NaN	NaN	60.1	25	43.5	60.1	25
31	Dadra & Nagar Haveli & Daman & Diu	62.1	54.3	58.3	69.9	18.5	47.9	66.7	34.7
32	Jammu & Kashmir	55.7	41.9	49	58.2	25.8	42.5	56.2	38.8
33	Ladakh	55.5	46.4	51.3	61	26.3	45	56.2	43.7
34	Lakshadweep	73.3	14.5	45.7	56.3	12.4	35.4	61.2	13
35	Puducherry	58.6	37.1	48	58.2	24.4	40.2	58.4	28.9
36	all India	57.9	35.5	46.8	59	22.3	41	58.2	31.7

In [54]:
```
age_15_29.to_csv('LFPR_15_29.csv', index=False,header=True)
age_15_59.to_csv('LFPR_15_59.csv', index=False, header=True)
age_15_above.to_csv('LFPR_15_above.csv', index=False, header=True)
all_ages.to_csv('LFPR_all_ages.csv',index=False, header=True)
```

In [68]:
```
df = pd.read_excel('Datasets/PLFS/PLFS(2023-24)/Unemployment_Rate.xlsx', hea
df
```

Out[68]:

	0	1	2	3	4	5	6	7	8	
0	Table (18.1): Unemployment Rate (UR) (in per c...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	State/UT	rural	NaN	NaN	urban	NaN	NaN	rural + urban	NaN	
3	NaN	male	female	person	male	female	person	male	female	pe
4	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	
5	Andhra Pradesh	4.8	3	4.1	5.7	7	6.1	5.1	3.8	
6	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8	
7	Assam	4.7	9.6	6.1	7	17.2	9.5	4.9	10.3	
8	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9	
9	Chhattisgarh	2	1.8	1.9	6.9	12.3	8.4	2.9	3	
10	Delhi	3.6	19.5	6	2.2	1.1	2	2.3	1.5	
11	Goa	8	14.1	9.5	3.8	18.6	8.2	5.6	16.8	
12	Gujarat	0.7	0.3	0.6	2.4	4.6	2.9	1.4	1.4	
13	Haryana	3.9	1.7	3.4	4.2	3.6	4	4	2.4	
14	Himachal Pradesh	7.6	12.5	9.9	4.8	20.4	9.7	7.2	13.1	
15	Jharkhand	1.5	0.1	1.1	6.9	8.2	7.1	2.5	1	
16	Karnataka	2.7	1	2.1	4.3	4.6	4.4	3.3	2	
17	Kerala	5.6	17.4	9.7	5.7	15.9	9.2	5.6	16.7	
18	Madhya Pradesh	1.1	1.5	1.2	3.4	4.2	3.5	1.6	2.1	
19	Maharashtra	3.1	1.2	2.4	5	6.4	5.3	3.9	2.8	
20	Manipur	4.8	6.7	5.6	5.8	10.6	7.8	5.1	7.8	
21	Meghalaya	4	12.3	7.7	9.2	21.5	14.6	4.7	13.5	
22	Mizoram	1.5	1	1.3	3.2	3.9	3.4	2.2	2.4	
23	Nagaland	5.7	6.7	6.1	11.6	13.5	12.3	7.3	8.2	
24	Odisha	5.1	3.5	4.6	6.9	12.2	8.2	5.4	4.5	
25	Punjab	5.1	7.7	5.8	4.8	9.3	5.8	5	8.2	
26	Rajasthan	4.2	3.8	4.1	7	13.1	8.5	5	5.4	
27	Sikkim	4.6	3.5	4.1	1.8	7	3.1	4	3.8	

	0	1	2	3	4	5	6	7	8
28	Tamil Nadu	4.5	5.9	5.1	3.6	7.2	4.6	4.1	6.3
29	Telangana	4.8	3.4	4.2	6.4	11.1	7.7	5.5	5.4
30	Tripura	1.6	1.3	1.5	2.7	4.8	3.3	1.8	1.7
31	Uttarakhand	6.1	4.6	5.6	3.6	16.9	6.6	5.4	6.6
32	Uttar Pradesh	3.4	2.8	3.2	5.9	13.4	7.1	3.9	4.5
33	West Bengal	3.1	5.1	3.6	3.2	7.6	4.2	3.2	5.9
34	Andaman & N. Island	9.4	21.2	13.5	7.5	33.9	14.9	8.6	25.9
35	Chandigarh	NaN	NaN	NaN	4.4	16.3	7.7	4.4	16.3
36	Dadra & Nagar Haveli & Daman & Diu	5.4	0	3.1	2.1	3.9	2.4	3.3	1.2
37	Jammu & Kashmir	6	17.2	8.8	5.3	35.4	12.8	5.9	20.9
38	Ladakh	2.1	7.8	4.2	5.7	37.2	13.8	2.6	10.4
39	Lakshadweep	12.8	53.6	18.4	12.1	28.6	14.7	12.3	36.2
40	Puducherry	0.3	5.4	2.1	7	9.8	7.9	4.4	7.9
41	all India	3.4	3.7	3.5	4.8	8.7	5.7	3.8	4.9
42	Note: For Chandigarh entire area has been cons...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [69]:
```
df = remove_headers(df)
```

In [70]:
```
df
```

Out[70]:

	0	1	2	3	4	5	6	7	8	9
0	Andhra Pradesh	4.8	3	4.1	5.7	7	6.1	5.1	3.8	4.6
1	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8	7.5
2	Assam	4.7	9.6	6.1	7	17.2	9.5	4.9	10.3	6.5
3	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9	3.9
4	Chhattisgarh	2	1.8	1.9	6.9	12.3	8.4	2.9	3	3
5	Delhi	3.6	19.5	6	2.2	1.1	2	2.3	1.5	2.1
6	Goa	8	14.1	9.5	3.8	18.6	8.2	5.6	16.8	8.7
7	Gujarat	0.7	0.3	0.6	2.4	4.6	2.9	1.4	1.4	1.4
8	Haryana	3.9	1.7	3.4	4.2	3.6	4	4	2.4	3.6
9	Himachal Pradesh	7.6	12.5	9.9	4.8	20.4	9.7	7.2	13.1	9.9
10	Jharkhand	1.5	0.1	1.1	6.9	8.2	7.1	2.5	1	2.1
11	Karnataka	2.7	1	2.1	4.3	4.6	4.4	3.3	2	2.9
12	Kerala	5.6	17.4	9.7	5.7	15.9	9.2	5.6	16.7	9.5
13	Madhya Pradesh	1.1	1.5	1.2	3.4	4.2	3.5	1.6	2.1	1.7
14	Maharashtra	3.1	1.2	2.4	5	6.4	5.3	3.9	2.8	3.5
15	Manipur	4.8	6.7	5.6	5.8	10.6	7.8	5.1	7.8	6.2
16	Meghalaya	4	12.3	7.7	9.2	21.5	14.6	4.7	13.5	8.6
17	Mizoram	1.5	1	1.3	3.2	3.9	3.4	2.2	2.4	2.3
18	Nagaland	5.7	6.7	6.1	11.6	13.5	12.3	7.3	8.2	7.7
19	Odisha	5.1	3.5	4.6	6.9	12.2	8.2	5.4	4.5	5.1
20	Punjab	5.1	7.7	5.8	4.8	9.3	5.8	5	8.2	5.8
21	Rajasthan	4.2	3.8	4.1	7	13.1	8.5	5	5.4	5.1
22	Sikkim	4.6	3.5	4.1	1.8	7	3.1	4	3.8	3.9
23	Tamil Nadu	4.5	5.9	5.1	3.6	7.2	4.6	4.1	6.3	4.9
24	Telangana	4.8	3.4	4.2	6.4	11.1	7.7	5.5	5.4	5.4
25	Tripura	1.6	1.3	1.5	2.7	4.8	3.3	1.8	1.7	1.7
26	Uttarakhand	6.1	4.6	5.6	3.6	16.9	6.6	5.4	6.6	5.8
27	Uttar Pradesh	3.4	2.8	3.2	5.9	13.4	7.1	3.9	4.5	4.1
28	West Bengal	3.1	5.1	3.6	3.2	7.6	4.2	3.2	5.9	3.8
29	Andaman & N. Island	9.4	21.2	13.5	7.5	33.9	14.9	8.6	25.9	14.1
30	Chandigarh	NaN	NaN	NaN	4.4	16.3	7.7	4.4	16.3	7.7
31	Dadra & Nagar Haveli & Daman & Diu	5.4	0	3.1	2.1	3.9	2.4	3.3	1.2	2.7

	0	1	2	3	4	5	6	7	8	9
32	Jammu & Kashmir	6	17.2	8.8	5.3	35.4	12.8	5.9	20.9	9.6
33	Ladakh	2.1	7.8	4.2	5.7	37.2	13.8	2.6	10.4	5.4
34	Lakshadweep	12.8	53.6	18.4	12.1	28.6	14.7	12.3	36.2	16
35	Puducherry	0.3	5.4	2.1	7	9.8	7.9	4.4	7.9	5.6
36	all India	3.4	3.7	3.5	4.8	8.7	5.7	3.8	4.9	4.1

In [71]:
```
df.columns = cols
```

In [72]:
```
df
```

Out[72]:

		Rural			Urban			Rural	
	State/UT	Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	4.8	3	4.1	5.7	7	6.1	5.1	3.8
1	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8
2	Assam	4.7	9.6	6.1	7	17.2	9.5	4.9	10.3
3	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9
4	Chhattisgarh	2	1.8	1.9	6.9	12.3	8.4	2.9	3
5	Delhi	3.6	19.5	6	2.2	1.1	2	2.3	1.5
6	Goa	8	14.1	9.5	3.8	18.6	8.2	5.6	16.8
7	Gujarat	0.7	0.3	0.6	2.4	4.6	2.9	1.4	1.4
8	Haryana	3.9	1.7	3.4	4.2	3.6	4	4	2.4
9	Himachal Pradesh	7.6	12.5	9.9	4.8	20.4	9.7	7.2	13.1
10	Jharkhand	1.5	0.1	1.1	6.9	8.2	7.1	2.5	1
11	Karnataka	2.7	1	2.1	4.3	4.6	4.4	3.3	2
12	Kerala	5.6	17.4	9.7	5.7	15.9	9.2	5.6	16.7
13	Madhya Pradesh	1.1	1.5	1.2	3.4	4.2	3.5	1.6	2.1
14	Maharashtra	3.1	1.2	2.4	5	6.4	5.3	3.9	2.8
15	Manipur	4.8	6.7	5.6	5.8	10.6	7.8	5.1	7.8
16	Meghalaya	4	12.3	7.7	9.2	21.5	14.6	4.7	13.5
17	Mizoram	1.5	1	1.3	3.2	3.9	3.4	2.2	2.4
18	Nagaland	5.7	6.7	6.1	11.6	13.5	12.3	7.3	8.2
19	Odisha	5.1	3.5	4.6	6.9	12.2	8.2	5.4	4.5
20	Punjab	5.1	7.7	5.8	4.8	9.3	5.8	5	8.2
21	Rajasthan	4.2	3.8	4.1	7	13.1	8.5	5	5.4
22	Sikkim	4.6	3.5	4.1	1.8	7	3.1	4	3.8
23	Tamil Nadu	4.5	5.9	5.1	3.6	7.2	4.6	4.1	6.3
24	Telangana	4.8	3.4	4.2	6.4	11.1	7.7	5.5	5.4
25	Tripura	1.6	1.3	1.5	2.7	4.8	3.3	1.8	1.7
26	Uttarakhand	6.1	4.6	5.6	3.6	16.9	6.6	5.4	6.6
27	Uttar Pradesh	3.4	2.8	3.2	5.9	13.4	7.1	3.9	4.5
28	West Bengal	3.1	5.1	3.6	3.2	7.6	4.2	3.2	5.9

				Rural			Urban		Rural
	State/UT	Male	Female	Person	Male	Female	Person	Male	Female
29	Andaman & N. Island	9.4	21.2	13.5	7.5	33.9	14.9	8.6	25.9
30	Chandigarh	NaN	NaN	NaN	4.4	16.3	7.7	4.4	16.3
31	Dadra & Nagar Haveli & Daman & Diu	5.4	0	3.1	2.1	3.9	2.4	3.3	1.2
32	Jammu & Kashmir	6	17.2	8.8	5.3	35.4	12.8	5.9	20.9
33	Ladakh	2.1	7.8	4.2	5.7	37.2	13.8	2.6	10.4
34	Lakshadweep	12.8	53.6	18.4	12.1	28.6	14.7	12.3	36.2
35	Puducherry	0.3	5.4	2.1	7	9.8	7.9	4.4	7.9
36	all India	3.4	3.7	3.5	4.8	8.7	5.7	3.8	4.9

In [73]:
```
df.to_csv('Unemp_Rate.csv', index=False)
```

In [76]:
```
df = pd.read_excel('Datasets/PLFS/PLFS(2023-24)/WPR.xlsx', header=None)
df.head()
```

Out[76]:

	0	1	2	3	4	5	6	7	8	9
0	Table (17.1): Worker Population Ratio (WPR) (i...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	State/UT	rural	NaN	NaN	urban	NaN	NaN	rural + urban	NaN	NaN
3	NaN	male	female	person	male	female	person	male	female	person
4	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)

In [78]:
```
df = remove_headers(df)
```

In [79]:
```
df.head()
```

Out[79]:

	0	1	2	3	4	5	6	7	8	9
0	Andhra Pradesh	56.5	38.3	47.3	55.6	22.4	38.5	56.3	33.5	44.7
1	Arunachal Pradesh	52.6	47.5	50.2	50	27.2	38.4	52.2	44	48.2
2	Assam	57	22.8	40.1	59.1	17.3	38.3	57.2	22.2	39.9
3	Bihar	45.4	10.1	28.1	44.1	7.6	26.7	45.3	9.9	28
4	Chhattisgarh	62	45.4	53.7	58.2	23	41.3	61.2	41.2	51.3

In [80]:
```
df.columns = cols
```

In [81]:
```
df.head(0)
```

Out[81]:

			Rural			Urban			Rural + Urb
State/UT	**Male**	**Female**	**Person**	**Male**	**Female**	**Person**	**Male**	**Female**	**Pers**

In [82]:
```
df.to_csv('WPR.csv', index=False)
```

In [84]:
```
col = pd.MultiIndex.from_tuples([
    ("", 'Item group'),
    ('MPCE(Rs.)','Rural'),('MPCE(Rs.)','Urban'),
    ('%share in total MPCE','Rural'),('%share in total MPCE','Urban')
])
```

In [192…
```
df = pd.read_excel('Datasets/HCES/Abs_Perc.xlsx',header=None,skiprows=3)
df = df.dropna(axis=1, how='all')
df = df.dropna(how='all')
```

In [193…
```
df
```

Out[193…

	0	1	9	18	26
0	cereals & cereal substitutes	206.0	263.0	4.99	3.76
1	pulses & their products*	84.0	98.0	2.04	1.40
2	sugar & salt	37.0	40.0	0.89	0.57
3	milk & milk products	348.0	503.0	8.44	7.19
4	vegetables	248.0	288.0	6.03	4.12
5	fruits	158.0	271.0	3.85	3.87
6	egg, fish & meat	203.0	249.0	4.92	3.56
7	edible oil	114.0	127.0	2.77	1.82
8	spices	135.0	161.0	3.27	2.30
9	beverages, refreshments, processed food#	406.0	776.0	9.84	11.09
10	food total	1939.0	2776.0	47.04	39.68
11	pan, tobacco & intoxicants	158.0	166.0	3.84	2.37
12	fuel and light	252.0	391.0	6.11	5.59
13	education	133.0	418.0	3.24	5.97
14	medical	282.0	409.0	6.83	5.85
15	conveyance	313.0	592.0	7.59	8.46
16	consumer services excluding conveyance	217.0	400.0	5.25	5.72
17	misc. goods, entertainment	256.0	484.0	6.22	6.92
18	rent	23.0	460.0	0.56	6.58
19	taxes and cesses	9.0	23.0	0.21	0.33
20	clothing, bedding & footwear	273.0	396.0	6.63	5.66
21	durable goods	267.0	481.0	6.48	6.87
22	non-food total	2183.0	4220.0	52.96	60.32
23	all items	4122.0	6996.0	100.00	100.00
24	*includes gram #includes purchased cooked...	NaN	NaN	NaN	NaN
25	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN	NaN

In [194…
```
df.drop(24, inplace=True)
```

In [195…
```
df.drop(25, inplace=True)
```

In [196…
```
df.columns = col
df.head()
```

Out[196…

		MPCE(Rs.)		%share in total MPCE	
	Item group	Rural	Urban	Rural	Urban
0	cereals & cereal substitutes	206.0	263.0	4.99	3.76
1	pulses & their products*	84.0	98.0	2.04	1.40
2	sugar & salt	37.0	40.0	0.89	0.57
3	milk & milk products	348.0	503.0	8.44	7.19
4	vegetables	248.0	288.0	6.03	4.12

In [212…
```
df.to_csv('Abs_Perc_MPCE.csv')
```

In [213…
```
pd.read_csv('Abs_Perc_MPCE.csv', header=[0,1], index_col=0)
```

Out[213…

	Unnamed: 1_level_0	MPCE(Rs.)		%share in total MPCE	
	Item group	Rural	Urban	Rural	Urban
0	cereals & cereal substitutes	206.0	263.0	4.99	3.76
1	pulses & their products*	84.0	98.0	2.04	1.40
2	sugar & salt	37.0	40.0	0.89	0.57
3	milk & milk products	348.0	503.0	8.44	7.19
4	vegetables	248.0	288.0	6.03	4.12
5	fruits	158.0	271.0	3.85	3.87
6	egg, fish & meat	203.0	249.0	4.92	3.56
7	edible oil	114.0	127.0	2.77	1.82
8	spices	135.0	161.0	3.27	2.30
9	beverages, refreshments, processed food#	406.0	776.0	9.84	11.09
10	food total	1939.0	2776.0	47.04	39.68
11	pan, tobacco & intoxicants	158.0	166.0	3.84	2.37
12	fuel and light	252.0	391.0	6.11	5.59
13	education	133.0	418.0	3.24	5.97
14	medical	282.0	409.0	6.83	5.85
15	conveyance	313.0	592.0	7.59	8.46
16	consumer services excluding conveyance	217.0	400.0	5.25	5.72
17	misc. goods, entertainment	256.0	484.0	6.22	6.92
18	rent	23.0	460.0	0.56	6.58
19	taxes and cesses	9.0	23.0	0.21	0.33
20	clothing, bedding & footwear	273.0	396.0	6.63	5.66
21	durable goods	267.0	481.0	6.48	6.87
22	non-food total	2183.0	4220.0	52.96	60.32
23	all items	4122.0	6996.0	100.00	100.00

In [214…

```
df.rename(columns={'Unnamed: 1_level_0': ''})
```

Out[214…

		MPCE(Rs.)		%share in total MPCE	
	Item group	Rural	Urban	Rural	Urban
0	cereals & cereal substitutes	206.0	263.0	4.99	3.76
1	pulses & their products*	84.0	98.0	2.04	1.40
2	sugar & salt	37.0	40.0	0.89	0.57
3	milk & milk products	348.0	503.0	8.44	7.19
4	vegetables	248.0	288.0	6.03	4.12
5	fruits	158.0	271.0	3.85	3.87
6	egg, fish & meat	203.0	249.0	4.92	3.56
7	edible oil	114.0	127.0	2.77	1.82
8	spices	135.0	161.0	3.27	2.30
9	beverages, refreshments, processed food#	406.0	776.0	9.84	11.09
10	food total	1939.0	2776.0	47.04	39.68
11	pan, tobacco & intoxicants	158.0	166.0	3.84	2.37
12	fuel and light	252.0	391.0	6.11	5.59
13	education	133.0	418.0	3.24	5.97
14	medical	282.0	409.0	6.83	5.85
15	conveyance	313.0	592.0	7.59	8.46
16	consumer services excluding conveyance	217.0	400.0	5.25	5.72
17	misc. goods, entertainment	256.0	484.0	6.22	6.92
18	rent	23.0	460.0	0.56	6.58
19	taxes and cesses	9.0	23.0	0.21	0.33
20	clothing, bedding & footwear	273.0	396.0	6.63	5.66
21	durable goods	267.0	481.0	6.48	6.87
22	non-food total	2183.0	4220.0	52.96	60.32
23	all items	4122.0	6996.0	100.00	100.00

In [222…
```
df = pd.read_excel('Datasets/HCES/Avg_MPCE.xlsx', header=None, skiprows=3)
```

In [224…
```
df = df.dropna(axis=1, how='all')
df = df.dropna(how='all')
```

In [225…
```
df
```

Out[225…

	0	10	18	27	36
0	Food	1939.0	47.04	2776.0	39.68
1	Non-food	2183.0	52.96	4220.0	60.32
2	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN	NaN

In [228…
```
df.drop(2, inplace=True)
```

In [229…
```
df
```

Out[229…

	0	10	18	27	36
0	Food	1939.0	47.04	2776.0	39.68
1	Non-food	2183.0	52.96	4220.0	60.32

In [231…
```
col = pd.MultiIndex.from_tuples([
    ("",'Item group'),
    ('Rural India','Average MPCE(Rs.)'),('Rural India','Share in MPCE(%)'),
    ('Urban India','Average MPCE(Rs.)'),('Urban India','Share in MPCE(%)')
])

df.columns = col
```

In [232…
```
df
```

Out[232…

		Rural India		Urban India	
	Item group	**Average MPCE(Rs.)**	**Share in MPCE(%)**	**Average MPCE(Rs.)**	**Share in MPCE(%)**
0	Food	1939.0	47.04	2776.0	39.68
1	Non-food	2183.0	52.96	4220.0	60.32

In [233…
```
df.to_csv('Avg_MPCE.csv')
```

In [235…
```
df = pd.read_excel('Datasets/HCES/Avg_MPCE_State.xlsx',header=None,skiprows=
df
```

Out[235…

	0	1	2	3	4	5	6	7
0	Andhra Pradesh	5327.0	NaN	NaN	NaN	NaN	NaN	7182.0
1	Arunachal Pradesh	5995.0	NaN	NaN	NaN	NaN	NaN	9832.0
2	Assam	3793.0	NaN	NaN	NaN	NaN	NaN	6794.0
3	Bihar	3670.0	NaN	NaN	NaN	NaN	NaN	5080.0
4	Chhattisgarh	2739.0	NaN	NaN	NaN	NaN	NaN	4927.0
5	Delhi	7400.0	NaN	NaN	NaN	NaN	NaN	8534.0
6	Goa	8048.0	NaN	NaN	NaN	NaN	NaN	9726.0
7	Gujarat	4116.0	NaN	NaN	NaN	NaN	NaN	7175.0
8	Haryana	5377.0	NaN	NaN	NaN	NaN	NaN	8428.0
9	Himachal Pradesh	5825.0	NaN	NaN	NaN	NaN	NaN	9223.0
10	Jharkhand	2946.0	NaN	NaN	NaN	NaN	NaN	5393.0
11	Karnataka	4903.0	NaN	NaN	NaN	NaN	NaN	8076.0
12	Kerala	6611.0	NaN	NaN	NaN	NaN	NaN	7783.0
13	Madhya Pradesh	3441.0	NaN	NaN	NaN	NaN	NaN	5538.0
14	Maharashtra	4145.0	NaN	NaN	NaN	NaN	NaN	7363.0
15	Manipur	4531.0	NaN	NaN	NaN	NaN	NaN	5945.0
16	Meghalaya	3852.0	NaN	NaN	NaN	NaN	NaN	7839.0
17	Mizoram	5963.0	NaN	NaN	NaN	NaN	NaN	8709.0
18	Nagaland	5155.0	NaN	NaN	NaN	NaN	NaN	8022.0
19	Odisha	3357.0	NaN	NaN	NaN	NaN	NaN	5825.0
20	Punjab	5817.0	NaN	NaN	NaN	NaN	NaN	7359.0
21	Rajasthan	4510.0	NaN	NaN	NaN	NaN	NaN	6574.0
22	Sikkim	9377.0	NaN	NaN	NaN	NaN	NaN	13927.0
23	Tamil Nadu	5701.0	NaN	NaN	NaN	NaN	NaN	8165.0
24	Telangana	5435.0	NaN	NaN	NaN	NaN	NaN	8978.0
25	Tripura	6259.0	NaN	NaN	NaN	NaN	NaN	8034.0
26	Uttar Pradesh	3481.0	NaN	NaN	NaN	NaN	NaN	5395.0
27	Uttarakhand	5003.0	NaN	NaN	NaN	NaN	NaN	7486.0
28	West Bengal	3620.0	NaN	NaN	NaN	NaN	NaN	5775.0
29	Andaman & N Islands	7771.0	NaN	NaN	NaN	NaN	NaN	10453.0
30	Chandigarh	8857.0	NaN	NaN	NaN	NaN	NaN	13425.0
31	Dadra & Nagar Haveli and Daman & Diu	4311.0	NaN	NaN	NaN	NaN	NaN	6837.0

	0	1	2	3	4	5	6	7
32	Jammu & Kashmir	4774.0	NaN	NaN	NaN	NaN	NaN	6327.0
33	Ladakh	5010.0	NaN	NaN	NaN	NaN	NaN	7533.0
34	Lakshadweep	6350.0	NaN	NaN	NaN	NaN	NaN	6377.0
35	Puducherry	7598.0	NaN	NaN	NaN	NaN	NaN	8637.0
36	All-India	4122.0	NaN	NaN	NaN	NaN	NaN	6996.0
37	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [238…
```
df = df.dropna(axis=1,how='all')
df = df.dropna(how='all')
```

In [239…
```
df
```

Out[239…

	0	1	7
0	Andhra Pradesh	5327.0	7182.0
1	Arunachal Pradesh	5995.0	9832.0
2	Assam	3793.0	6794.0
3	Bihar	3670.0	5080.0
4	Chhattisgarh	2739.0	4927.0
5	Delhi	7400.0	8534.0
6	Goa	8048.0	9726.0
7	Gujarat	4116.0	7175.0
8	Haryana	5377.0	8428.0
9	Himachal Pradesh	5825.0	9223.0
10	Jharkhand	2946.0	5393.0
11	Karnataka	4903.0	8076.0
12	Kerala	6611.0	7783.0
13	Madhya Pradesh	3441.0	5538.0
14	Maharashtra	4145.0	7363.0
15	Manipur	4531.0	5945.0
16	Meghalaya	3852.0	7839.0
17	Mizoram	5963.0	8709.0
18	Nagaland	5155.0	8022.0
19	Odisha	3357.0	5825.0
20	Punjab	5817.0	7359.0
21	Rajasthan	4510.0	6574.0
22	Sikkim	9377.0	13927.0
23	Tamil Nadu	5701.0	8165.0
24	Telangana	5435.0	8978.0
25	Tripura	6259.0	8034.0
26	Uttar Pradesh	3481.0	5395.0
27	Uttarakhand	5003.0	7486.0
28	West Bengal	3620.0	5775.0
29	Andaman & N Islands	7771.0	10453.0
30	Chandigarh	8857.0	13425.0
31	Dadra & Nagar Haveli and Daman & Diu	4311.0	6837.0
32	Jammu & Kashmir	4774.0	6327.0

	0	1	7
33	Ladakh	5010.0	7533.0
34	Lakshadweep	6350.0	6377.0
35	Puducherry	7598.0	8637.0
36	All-India	4122.0	6996.0
37	Source: Factsheet-Household Consumption Expend...	NaN	NaN

In [241…
```
df.drop(37,inplace=True)
```

In [243…
```
col = pd.MultiIndex.from_tuples([
    ("",'State/UT'),
    ('Average MPCE(Rs.)','Rural'),('Average MPCE(Rs.)','Urban')
])

df.columns = col
df.head()
```

Out[243…

		Average MPCE(Rs.)	
	State/UT	Rural	Urban
0	Andhra Pradesh	5327.0	7182.0
1	Arunachal Pradesh	5995.0	9832.0
2	Assam	3793.0	6794.0
3	Bihar	3670.0	5080.0
4	Chhattisgarh	2739.0	4927.0

In [244…
```
df.to_csv('Avg_MPCE_State.csv')
```

In [245…
```
df = pd.read_excel('Datasets/HCES/Trend_Cereal_Food.xlsx',header=None,skiprc
```

In [247…
```
df = df.dropna(axis=1, how='all')
df = df.dropna(how='all')
```

In [251…
```
df.drop(3,inplace=True)
```

In [252…
```
col = pd.MultiIndex.from_tuples([
    ("",'Period'),
    ('Rural','%share of cereals in avg.'),('Rural','%share of food in avg.')
    ('Urban','%share of cereals in avg.'),('Urban','%share of food in avg.')
])
df.columns = col
df
```

Out[252…

		Rural		Urban	
	Period	%share of cereals in avg.	%share of food in avg.	%share of cereals in avg.	%share of food in avg.
0	2011-12	10.75	52.90	6.66	42.62
1	2022-23	4.91	46.38	3.64	39.17
2	2023-24	4.99	47.04	3.76	39.68

In [253…
```
df.to_csv('Trend_cereal_food.csv')
```

In [254…
```
df = pd.read_excel('Datasets/HCES/Trend_Consump.xlsx',header=None, skiprows=
```

In [255…
```
df = df.dropna(axis=1, how='all')
df = df.dropna(how='all')
```

In [256…
```
df
```

Out[256…

		0	17	30	42
0	Rural	1430.0	3773.0	4122.0	
1	Urban	2630.0	6459.0	6996.0	
2	Difference as % of Rural MPCE	83.9	71.2	69.7	
3	Note: The estimates of MPCE are based on Modif...	NaN	NaN	NaN	
4	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN	

In [257…
```
df.drop(3, inplace=True)
df.drop(4, inplace=True)
```

In [258…
```
df
```

Out[258…

		0	17	30	42
0	Rural	1430.0	3773.0	4122.0	
1	Urban	2630.0	6459.0	6996.0	
2	Difference as % of Rural MPCE	83.9	71.2	69.7	

In [259…
```
col = pd.MultiIndex.from_tuples([
    ("","Sector'),
    ('Average MPCE(Rs.) over different period','2011-12'),('Average MPCE(Rs.
])
df.columns = col
df
```

Out[259…

Average MPCE(Rs.) over different period

	Sector	2011-12	2022-23	2023-24
0	Rural	1430.0	3773.0	4122.0
1	Urban	2630.0	6459.0	6996.0
2	Difference as % of Rural MPCE	83.9	71.2	69.7

In [260…
```
df.to_csv('Trend_Consump.csv')
```

In [261…
```
df = pd.read_excel('Datasets/HCES/Trend_Rural.xlsx',header=None,skiprows=3)
```

In [262…
```
df = df.dropna(axis=1,how='all')
df = df.dropna(how='all')
```

In [263…
```
df
```

Out[263…

	0	26	37	47
0	cereal	10.69	4.89	4.97
1	cereal substitutes	0.06	0.02	0.02
2	gram	0.14	0.24	0.26
3	pulses and pulse products*	2.76	1.77	1.78
4	sugar & salt	1.83	0.93	0.89
5	milk and milk products	8.04	8.33	8.44
6	vegetables	6.62	5.38	6.03
7	fruits (fresh)	2.25	2.54	2.66
8	fruits (dry)	0.58	1.17	1.19
9	egg, fish & meat	4.79	4.91	4.92
10	edible oil	3.74	3.59	2.77
11	spices	3.50	2.98	3.27
12	beverages, processed food# etc.	7.90	9.62	9.84
13	food: total	52.90	46.38	47.04
14	pan, tobacco & intoxicants	3.21	3.79	3.84
15	fuel and light	7.98	6.66	6.11
16	toilet articles & other ...	4.01	5.12	5.20
17	education	3.49	3.30	3.24
18	medical (hospitalization)	2.15	2.36	2.25
19	medical (non- hospitalization)	4.50	4.77	4.58
20	conveyance	4.20	7.55	7.59
21	consumer services excluding conveyance&	3.99	5.08	5.25
22	entertainment	0.99	1.09	1.02
23	rent	0.45	0.78	0.56
24	other taxes & cesses	0.25	0.13	0.21
25	clothing & bedding$	5.99	5.24	5.67
26	footwear	1.02	0.86	0.96
27	durable goods	4.85	6.89	6.48
28	non-food: total	47.10	53.62	52.96
29	total expenditure	100.00	100.00	100.00
30	*excludes gram #includes purchased cooked ...	NaN	NaN	NaN
31	Source: Factsheet-Household Consumption Expend...	NaN	NaN	NaN

In [264…
```
df.drop(30, inplace=True)
df.drop(31, inplace=True)
```

In [265…
```
col = pd.MultiIndex.from_tuples([
    ("",'Item group'),
    ('%share in total MPCE','2011-12'),('%share in total MPCE','2022-23'),('
])

df.columns = col
df
```

Out[265…

		%share in total MPCE		
	Item group	2011-12	2022-23	2023-24
0	cereal	10.69	4.89	4.97
1	cereal substitutes	0.06	0.02	0.02
2	gram	0.14	0.24	0.26
3	pulses and pulse products*	2.76	1.77	1.78
4	sugar & salt	1.83	0.93	0.89
5	milk and milk products	8.04	8.33	8.44
6	vegetables	6.62	5.38	6.03
7	fruits (fresh)	2.25	2.54	2.66
8	fruits (dry)	0.58	1.17	1.19
9	egg, fish & meat	4.79	4.91	4.92
10	edible oil	3.74	3.59	2.77
11	spices	3.50	2.98	3.27
12	beverages, processed food# etc.	7.90	9.62	9.84
13	food: total	52.90	46.38	47.04
14	pan, tobacco & intoxicants	3.21	3.79	3.84
15	fuel and light	7.98	6.66	6.11
16	toilet articles & other ...	4.01	5.12	5.20
17	education	3.49	3.30	3.24
18	medical (hospitalization)	2.15	2.36	2.25
19	medical (non- hospitalization)	4.50	4.77	4.58
20	conveyance	4.20	7.55	7.59
21	consumer services excluding conveyance&	3.99	5.08	5.25
22	entertainment	0.99	1.09	1.02
23	rent	0.45	0.78	0.56
24	other taxes & cesses	0.25	0.13	0.21
25	clothing & bedding$	5.99	5.24	5.67
26	footwear	1.02	0.86	0.96
27	durable goods	4.85	6.89	6.48
28	non-food: total	47.10	53.62	52.96
29	total expenditure	100.00	100.00	100.00

In [266…

```
df.to_csv('Trend_Rural.csv')
```

In [267…
```
df = pd.read_excel('Datasets/HCES/Trend_Urban.xlsx', header=None,skiprows=3)
```

In [268…
```
df = df.dropna(axis=1,how='all')
df = df.dropna(how='all')
df.drop(30, inplace=True)
df.drop(31, inplace=True)
```

In [270…
```
col = pd.MultiIndex.from_tuples([
    ("",'Item group'),
    ('%share in total MPCE','2011-12'),('%share in total MPCE','2022-23'),('
])
df.columns = col
df.head()
```

Out[270…

	%share in total MPCE			
	Item group	2011-12	2022-23	2023-24
0	cereal	6.61	3.62	3.74
1	cereal substitutes	0.05	0.02	0.02
2	gram	0.11	0.18	0.18
3	pulses and pulse products*	1.93	1.21	1.22
4	sugar & salt	1.15	0.60	0.57

In [271…
```
df.to_csv('Trend_Urban.csv')
```

In [1]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
df
```

Out[3]:

	0	1	2	3	4	5	6	7	
0	NaN	1.1.01	Cereals and products	12.35	184.8	186.2	6.59	184.2	185
1	NaN	1.1.02	Meat and fish	4.38	210.9	208.1	2.73	219.6	217
2	NaN	1.1.03	Egg	0.49	190.4	197.1	0.36	194.8	200
3	NaN	1.1.04	Milk and products	7.72	182.2	182.3	5.33	182.3	182
4	NaN	1.1.05	Oils and fats	4.21	162.6	162.4	2.81	156.7	156
5	NaN	1.1.06	Fruits	2.88	174.6	172.7	2.90	182.7	178
6	NaN	1.1.07	Vegetables	7.46	199.9	188.4	4.41	246.0	234
7	NaN	1.1.08	Pulses and products	2.95	202.9	204.2	1.73	209.3	210
8	NaN	1.1.09	Sugar and Confectionery	1.7	129.7	130.1	0.97	130.9	131
9	NaN	1.1.10	Spices	3.11	249.8	249.1	1.79	239.7	238
10	NaN	1.2.11	Non-alcoholic beverages	1.37	181.8	182	1.13	169.0	169
11	NaN	1.1.12	Prepared meals, snacks, sweets etc.	5.56	193.7	194.3	5.54	201.8	202
12	1	NaN	Food and beverages	54.18	190.2	188.8	36.29	196.6	195
13	2	NaN	Pan, tobacco and intoxicants	3.26	202.9	203.1	1.36	208.5	208
14	NaN	3.1.01	Clothing	6.32	194.4	194.8	4.72	184.5	184
15	NaN	3.1.02	Footwear	1.04	189.8	190.3	0.85	171.0	171
16	3	NaN	Clothing and footwear	7.36	193.7	194.2	5.57	182.4	182
17	4	NaN	Housing	-	-	-	21.67	177.9	176
18	5	NaN	Fuel and light	7.94	182.4	183.1	5.58	175.8	175
19	NaN	6.1.01	Household goods and services	3.75	182	182.5	3.87	172.3	172
20	NaN	6.1.02	Health	6.83	191.9	192.5	4.81	186.2	186
21	NaN	6.1.03	Transport and communication	7.6	171.7	171.8	9.73	161.7	161
22	NaN	6.1.04	Recreation and amusement	1.37	176.4	177	2.04	171.8	171
23	NaN	6.1.05	Education	3.46	185.2	185.3	5.62	180.4	180
24	NaN	6.1.06	Personal care and effects	4.25	186.7	188.1	3.47	187.9	189

	0	1	2	3	4	5	6	7	
25	6	NaN	Miscellaneous	27.26	182.5	183	29.53	174.4	174
26	General Index (All Groups)	NaN	NaN	100	188.2	187.6	100.00	184.2	183
27	Consumer Food Price Index (CFPI)	NaN	NaN	47.25	190.1	188.3	29.62	196.8	194
28	Notes:	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
29	NaN	1. Prov. : Provisional.	NaN	NaN	NaN	NaN	NaN	NaN	Na
30	NaN	2. CFPI : Out of 12 sub-groups contained in ...	NaN	NaN	NaN	NaN	NaN	NaN	Na
31	NaN	3. - : CPI (Rural) for housing is not compiled.	NaN	NaN	NaN	NaN	NaN	NaN	Na

In [4]:
```
df = df.dropna(axis=1,how='all')
df = df.dropna(how='all')
```

In [5]:
```
df
```

Out[5]:

	0	1	2	3	4	5	6	7	
0	NaN	1.1.01	Cereals and products	12.35	184.8	186.2	6.59	184.2	185
1	NaN	1.1.02	Meat and fish	4.38	210.9	208.1	2.73	219.6	217
2	NaN	1.1.03	Egg	0.49	190.4	197.1	0.36	194.8	200
3	NaN	1.1.04	Milk and products	7.72	182.2	182.3	5.33	182.3	182
4	NaN	1.1.05	Oils and fats	4.21	162.6	162.4	2.81	156.7	156
5	NaN	1.1.06	Fruits	2.88	174.6	172.7	2.90	182.7	178
6	NaN	1.1.07	Vegetables	7.46	199.9	188.4	4.41	246.0	234
7	NaN	1.1.08	Pulses and products	2.95	202.9	204.2	1.73	209.3	210
8	NaN	1.1.09	Sugar and Confectionery	1.7	129.7	130.1	0.97	130.9	131
9	NaN	1.1.10	Spices	3.11	249.8	249.1	1.79	239.7	238
10	NaN	1.2.11	Non-alcoholic beverages	1.37	181.8	182	1.13	169.0	169
11	NaN	1.1.12	Prepared meals, snacks, sweets etc.	5.56	193.7	194.3	5.54	201.8	202
12	1	NaN	Food and beverages	54.18	190.2	188.8	36.29	196.6	195
13	2	NaN	Pan, tobacco and intoxicants	3.26	202.9	203.1	1.36	208.5	208
14	NaN	3.1.01	Clothing	6.32	194.4	194.8	4.72	184.5	184
15	NaN	3.1.02	Footwear	1.04	189.8	190.3	0.85	171.0	171
16	3	NaN	Clothing and footwear	7.36	193.7	194.2	5.57	182.4	182
17	4	NaN	Housing	-	-	-	21.67	177.9	176
18	5	NaN	Fuel and light	7.94	182.4	183.1	5.58	175.8	175
19	NaN	6.1.01	Household goods and services	3.75	182	182.5	3.87	172.3	172
20	NaN	6.1.02	Health	6.83	191.9	192.5	4.81	186.2	186
21	NaN	6.1.03	Transport and communication	7.6	171.7	171.8	9.73	161.7	161
22	NaN	6.1.04	Recreation and amusement	1.37	176.4	177	2.04	171.8	171
23	NaN	6.1.05	Education	3.46	185.2	185.3	5.62	180.4	180
24	NaN	6.1.06	Personal care and effects	4.25	186.7	188.1	3.47	187.9	189

	0	1	2	3	4	5	6	7	
25	6	NaN	Miscellaneous	27.26	182.5	183	29.53	174.4	174
26	General Index (All Groups)	NaN	NaN	100	188.2	187.6	100.00	184.2	183
27	Consumer Food Price Index (CFPI)	NaN	NaN	47.25	190.1	188.3	29.62	196.8	194
28	Notes:	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
29	NaN	1. Prov. : Provisional.	NaN	NaN	NaN	NaN	NaN	NaN	Na
30	NaN	2. CFPI : Out of 12 sub-groups contained in ...	NaN	NaN	NaN	NaN	NaN	NaN	Na
31	NaN	3. - : CPI (Rural) for housing is not compiled.	NaN	NaN	NaN	NaN	NaN	NaN	Na

In [6]:
```
df.drop(28, inplace=True)
```

In [7]:
```
df.drop(29, inplace=True)
df.drop(30, inplace=True)
df.drop(31, inplace=True)
```

In [8]:
```
df
```

Out[8]:

	0	1	2	3	4	5	6	7	8
0	NaN	1.1.01	Cereals and products	12.35	184.8	186.2	6.59	184.2	185.6
1	NaN	1.1.02	Meat and fish	4.38	210.9	208.1	2.73	219.6	217.5
2	NaN	1.1.03	Egg	0.49	190.4	197.1	0.36	194.8	200.8
3	NaN	1.1.04	Milk and products	7.72	182.2	182.3	5.33	182.3	182.5
4	NaN	1.1.05	Oils and fats	4.21	162.6	162.4	2.81	156.7	156.7
5	NaN	1.1.06	Fruits	2.88	174.6	172.7	2.90	182.7	178.9
6	NaN	1.1.07	Vegetables	7.46	199.9	188.4	4.41	246.0	234.7
7	NaN	1.1.08	Pulses and products	2.95	202.9	204.2	1.73	209.3	210.1
8	NaN	1.1.09	Sugar and Confectionery	1.7	129.7	130.1	0.97	130.9	131.4
9	NaN	1.1.10	Spices	3.11	249.8	249.1	1.79	239.7	238.7
10	NaN	1.2.11	Non-alcoholic beverages	1.37	181.8	182	1.13	169.0	169.2
11	NaN	1.1.12	Prepared meals, snacks, sweets etc.	5.56	193.7	194.3	5.54	201.8	202.4
12	1	NaN	Food and beverages	54.18	190.2	188.8	36.29	196.6	195.3
13	2	NaN	Pan, tobacco and intoxicants	3.26	202.9	203.1	1.36	208.5	208.4
14	NaN	3.1.01	Clothing	6.32	194.4	194.8	4.72	184.5	184.8
15	NaN	3.1.02	Footwear	1.04	189.8	190.3	0.85	171.0	171.2
16	3	NaN	Clothing and footwear	7.36	193.7	194.2	5.57	182.4	182.7
17	4	NaN	Housing	-	-	-	21.67	177.9	176.9
18	5	NaN	Fuel and light	7.94	182.4	183.1	5.58	175.8	175.4
19	NaN	6.1.01	Household goods and services	3.75	182	182.5	3.87	172.3	172.7
20	NaN	6.1.02	Health	6.83	191.9	192.5	4.81	186.2	186.8
21	NaN	6.1.03	Transport and communication	7.6	171.7	171.8	9.73	161.7	161.9
22	NaN	6.1.04	Recreation and amusement	1.37	176.4	177	2.04	171.8	171.9
23	NaN	6.1.05	Education	3.46	185.2	185.3	5.62	180.4	180.3
24	NaN	6.1.06	Personal care and effects	4.25	186.7	188.1	3.47	187.9	189.4

	0	1	2	3	4	5	6	7	8
25	6	NaN	Miscellaneous	27.26	182.5	183	29.53	174.4	174.8
26	General Index (All Groups)	NaN	NaN	100	188.2	187.6	100.00	184.2	183.6
27	Consumer Food Price Index (CFPI)	NaN	NaN	47.25	190.1	188.3	29.62	196.8	194.9

In [ ]:

EDA

1. Periodic Labor Force Survey

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
emp_industry = pd.read_csv('Cleaned Data/PLFS/PLFS_emp_industry.csv',header=
```

In [3]:
```
emp_industry
```

Out[3]:

	0	1	2	3	4	5	6
	01-03 (agriculture)	**61.6**	**22**	**83.6**	**1**	**15.4**	**100**
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	100
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	100
2	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	100
3	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	100
4	05-43 (secondary)	17.8	1.9	19.6	20.2	60.1	100
...	...	...	...	...	...	...	...
111	55-56 (accommodation & food services)	40.5	16.5	57.0	33.5	9.5	100
112	58-99 (other services)	20.6	1.7	22.4	75.8	1.8	100
113	45-99 (tertiary)	38.3	6.6	44.9	51.0	4.1	100
114	total	39.0	19.4	58.4	21.7	19.8	100
115	sample workers	73565.0	31425.0	104990.0	47036.0	31044.0	183070

116 rows × 7 columns

This dataset requires a column name and feature engineering

In [4]:
```
col = pd.MultiIndex.from_tuples([
    ('Industry as per NIC-2008',''),
    ('Self-Employed','own account worker, employer'),('Self-Employed','helpe
    ('regular wage/salary',''),('casual labour',''),('all','')
])
emp_industry.columns = col
```

In [5]:
```
emp_industry
```

Out[5]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	
2	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	
3	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	
4	05-43 (secondary)	17.8	1.9	19.6	20.2	60.1	
...	...	...	...	...	...	...	
111	55-56 (accommodation & food services)	40.5	16.5	57.0	33.5	9.5	
112	58-99 (other services)	20.6	1.7	22.4	75.8	1.8	
113	45-99 (tertiary)	38.3	6.6	44.9	51.0	4.1	
114	total	39.0	19.4	58.4	21.7	19.8	
115	sample workers	73565.0	31425.0	104990.0	47036.0	31044.0	1

116 rows × 7 columns

In [6]:
```
emp_industry.iloc[0:13]
```

Out[6]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	
2	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	
3	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	
4	05-43 (secondary)	17.8	1.9	19.6	20.2	60.1	
5	45-47 (trade)	63.0	7.4	70.4	26.5	3.1	
6	49-53(transport)	53.3	0.5	53.9	34.1	12.0	
7	55-56 (accommodation & food services))	49.0	10.2	59.3	31.8	8.9	
8	58-99 (other services)	29.1	2.0	31.0	66.5	2.4	
9	45-99 (tertiary)	49.2	4.3	53.5	41.2	5.3	
10	total	47.0	12.4	59.4	15.8	24.9	
11	sample workers	33350.0	8170.0	41520.0	12126.0	16323.0	69
12	01-03 (agriculture)	29.9	51.0	80.9	0.7	18.4	

In [7]:
```
categories = ['Male', 'Female', 'Persons'] * 9

rows_per_set = 13

category_col = [categories[i // rows_per_set] for i in range(len(emp_industr
emp_industry['Category'] = category_col
emp_industry
```

Out[7]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	
2	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	
3	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	
4	05-43 (secondary)	17.8	1.9	19.6	20.2	60.1	
...	...	...	...	...	...	...	
111	55-56 (accommodation & food services)	40.5	16.5	57.0	33.5	9.5	
112	58-99 (other services)	20.6	1.7	22.4	75.8	1.8	
113	45-99 (tertiary)	38.3	6.6	44.9	51.0	4.1	
114	total	39.0	19.4	58.4	21.7	19.8	
115	sample workers	73565.0	31425.0	104990.0	47036.0	31044.0	1

116 rows × 8 columns

In [8]:
```
emp_industry.iloc[0:13]
```

Out[8]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	
2	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	
3	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	
4	05-43 (secondary)	17.8	1.9	19.6	20.2	60.1	
5	45-47 (trade)	63.0	7.4	70.4	26.5	3.1	
6	49-53(transport)	53.3	0.5	53.9	34.1	12.0	
7	55-56 (accommodation & food services))	49.0	10.2	59.3	31.8	8.9	
8	58-99 (other services)	29.1	2.0	31.0	66.5	2.4	
9	45-99 (tertiary)	49.2	4.3	53.5	41.2	5.3	
10	total	47.0	12.4	59.4	15.8	24.9	
11	sample workers	33350.0	8170.0	41520.0	12126.0	16323.0	69
12	01-03 (agriculture)	29.9	51.0	80.9	0.7	18.4	

In [9]:
```
emp_industry.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 116 entries, 0 to 115
Data columns (total 8 columns):
 #   Column                                              Non-Null Count  Dtype
---  ------                                              --------------  -----
 0   (Industry as per NIC-2008, )                        116 non-null    object
 1   (Self-Employed, own account worker, employer)       116 non-null    float6
4
 2   (Self-Employed, helper in household enterprise)     116 non-null    float6
4
 3   (Self-Employed, all self employed)                  116 non-null    float6
4
 4   (regular wage/salary, )                             116 non-null    float6
4
 5   (casual labour, )                                   116 non-null    float6
4
 6   (all, )                                             116 non-null    int64
 7   (Category, )                                        116 non-null    object
dtypes: float64(5), int64(1), object(2)
memory usage: 7.4+ KB
```

In [10]:
```
emp_industry['Self-Employed','own account worker, employer'].iloc[0:11].mear
```

Out[10]:
```
np.float64(33.85454545454545)
```

Column Types:

- `Numerical` : own account worker, employer; helper in household enterprise all self ; all self employed; regular wage/salary; casual labour; all
- `Categorical` : Category
- `Mixed` : Industry as per NIC-2008

Removing the sample worker row in the table because it represents the survey sample size, not actual employment data.

In [11]:
```
emp_industry = emp_industry[~emp_industry['Industry as per NIC-2008'].str.cc
```

In [12]:
```
emp_industry[~emp_industry['Industry as per NIC-2008'].str.contains('total',
```

Out[12]:

	Self-Employed			regular wage/salary	casual labour	all
	own account worker, employer	helper in household enterprise	all self employed			
count	98.000000	98.000000	98.000000	98.000000	98.000000	98.0
mean	30.255102	9.597959	39.852041	40.972449	19.184694	100.0
std	18.307078	12.661403	25.182137	26.746819	24.223365	0.0
min	0.000000	0.000000	0.700000	0.700000	0.500000	100.0
25%	16.800000	1.425000	18.400000	20.125000	4.100000	100.0
50%	26.550000	4.750000	34.600000	36.100000	9.650000	100.0
75%	45.900000	12.825000	60.350000	64.175000	22.750000	100.0
max	70.200000	51.000000	86.200000	93.700000	98.600000	100.0

Univariate Analysis on Numerical Columns

1. own account worker, employer:

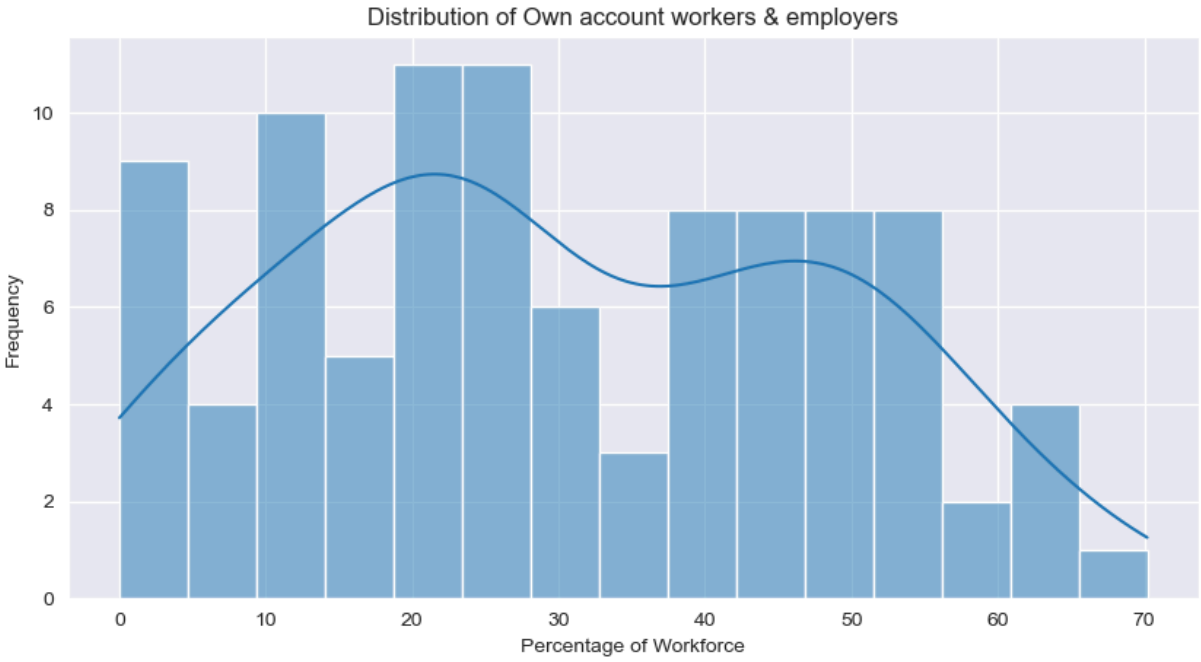
- Mean val: 30.5%
- STD: 18%
- 25%: 17%, 50%: 27%, 75%: 46%, max: 70% (there might be a outlier)

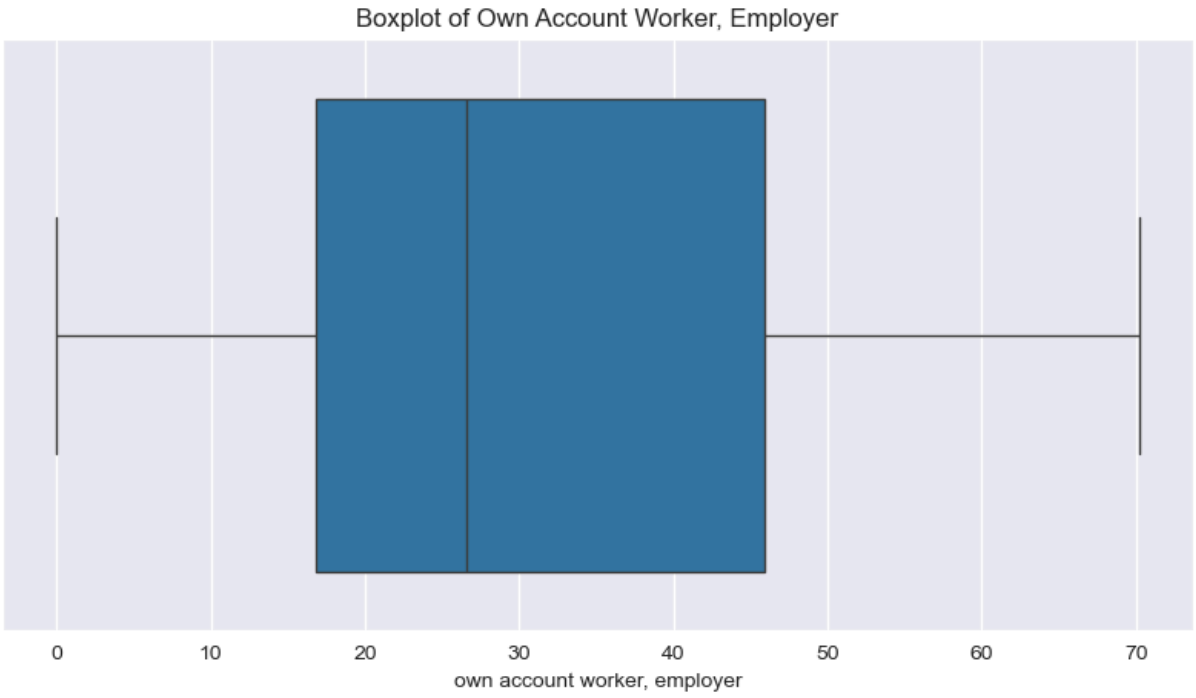
In [13]:
```
emp = emp_industry[~emp_industry['Industry as per NIC-2008'].str.contains('t
# filtering out the dataset without total rows for univariate analysis
```

In [14]:
```
emp['Self-Employed']['own account worker, employer'].skew()
```

Out[14]:
```
np.float64(0.15239918350142975)
```

In [15]:
```
sns.set_style('darkgrid')

plt.figure(figsize=(10,5))
sns.histplot(emp['Self-Employed']['own account worker, employer'], kde=True,
plt.title("Distribution of Own account workers & employers")
plt.xlabel('Percentage of Workforce')
plt.ylabel('Frequency')
plt.show()

#Boxplot
plt.figure(figsize=(10,5))
sns.boxplot(x=emp['Self-Employed']['own account worker, employer'])
plt.title('Boxplot of Own Account Worker, Employer')
plt.show()
```





Conclusion:

- Shape of the histogram is almost a bimodial distribution with peaks around 10-30% and 40-50%
- This shows that industries have very few or moderate amount of own acount workers, with fewer industries having extreme values(70%)
- Boxplot tells us that there are no outliers in this distribution which indicates that there are no extreme deviations.
- The spread is natural and there are no usual spikes.

In [16]:
```
emp_industry[~emp_industry['Industry as per NIC-2008'].str.contains('total',
```

Out[16]:

	Self-Employed			regular wage/salary	casual labour	all
	own account worker, employer	helper in household enterprise	all self employed			
count	98.000000	98.000000	98.000000	98.000000	98.000000	98.0
mean	30.255102	9.597959	39.852041	40.972449	19.184694	100.0
std	18.307078	12.661403	25.182137	26.746819	24.223365	0.0
min	0.000000	0.000000	0.700000	0.700000	0.500000	100.0
25%	16.800000	1.425000	18.400000	20.125000	4.100000	100.0
50%	26.550000	4.750000	34.600000	36.100000	9.650000	100.0
75%	45.900000	12.825000	60.350000	64.175000	22.750000	100.0
max	70.200000	51.000000	86.200000	93.700000	98.600000	100.0

2. Helper in Household Enterprise

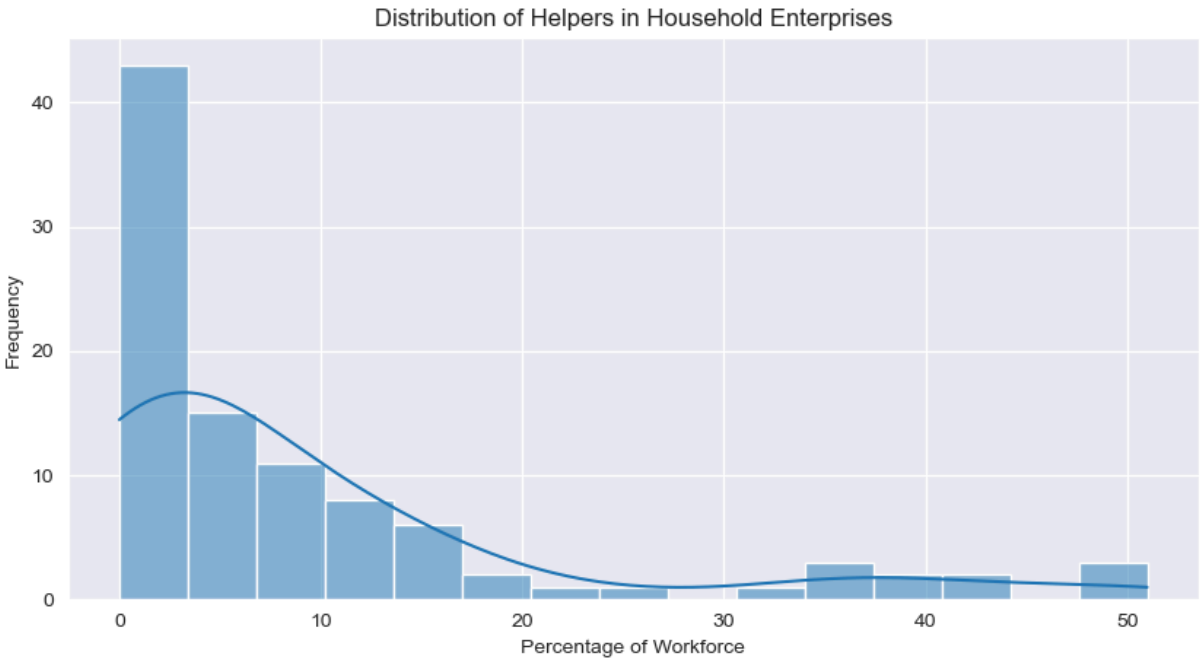
In [17]:
```
emp_industry[~emp_industry['Industry as per NIC-2008'].str.contains('total',
```

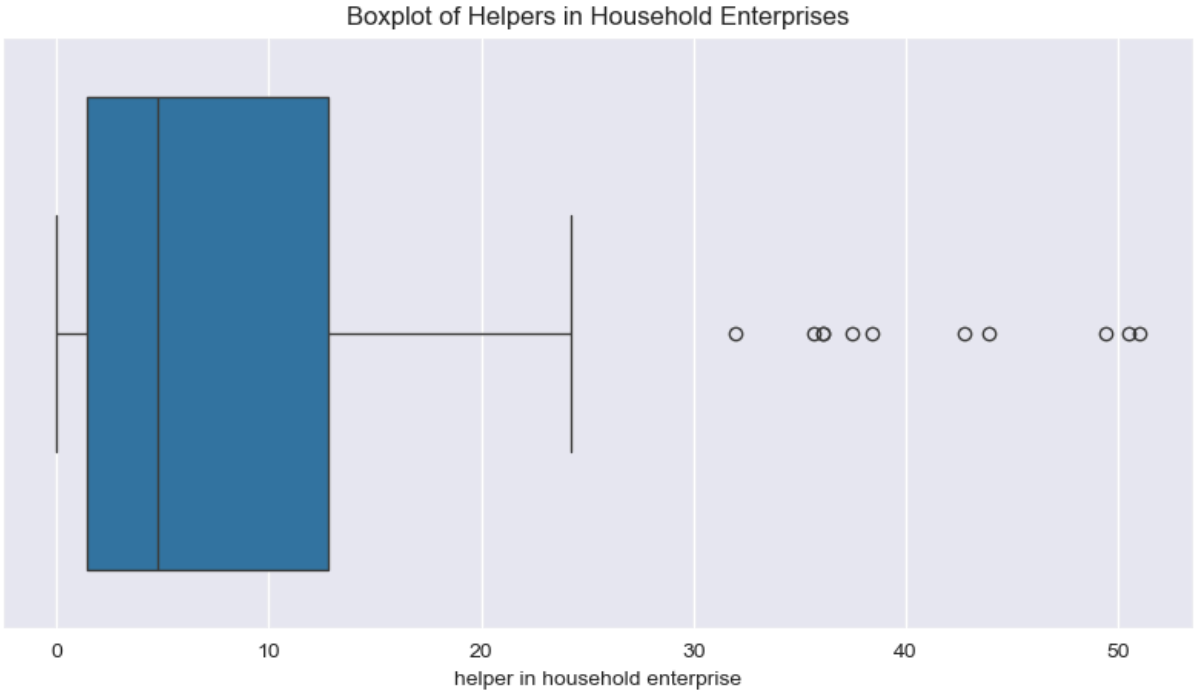
Out[17]:

		Self-Employed			regular wage/salary	cas labo
		own account worker, employer	helper in household enterprise	all self employed		
Self-Employed	**own account worker, employer**	1.000000	0.299664	0.877456	-0.448935	-0.416
	helper in household enterprise	0.299664	1.000000	0.720553	-0.477479	-0.221
	all self employed	0.877456	0.720553	1.000000	-0.566276	-0.414
regular wage/salary		-0.448935	-0.477479	-0.566276	1.000000	-0.515
casual labour		-0.416766	-0.221592	-0.414387	-0.515460	1.000
all		NaN	NaN	NaN	NaN	N

In [18]:
```
plt.figure(figsize=(10,5))
sns.histplot(emp['Self-Employed']['helper in household enterprise'],kde=True
```

```
plt.title('Distribution of Helpers in Household Enterprises')
plt.xlabel('Percentage of Workforce')
plt.ylabel('Frequency')
plt.show()

plt.figure(figsize=(10,5))
sns.boxplot(x=emp['Self-Employed']['helper in household enterprise'])
plt.title('Boxplot of Helpers in Household Enterprises')
plt.show()
```



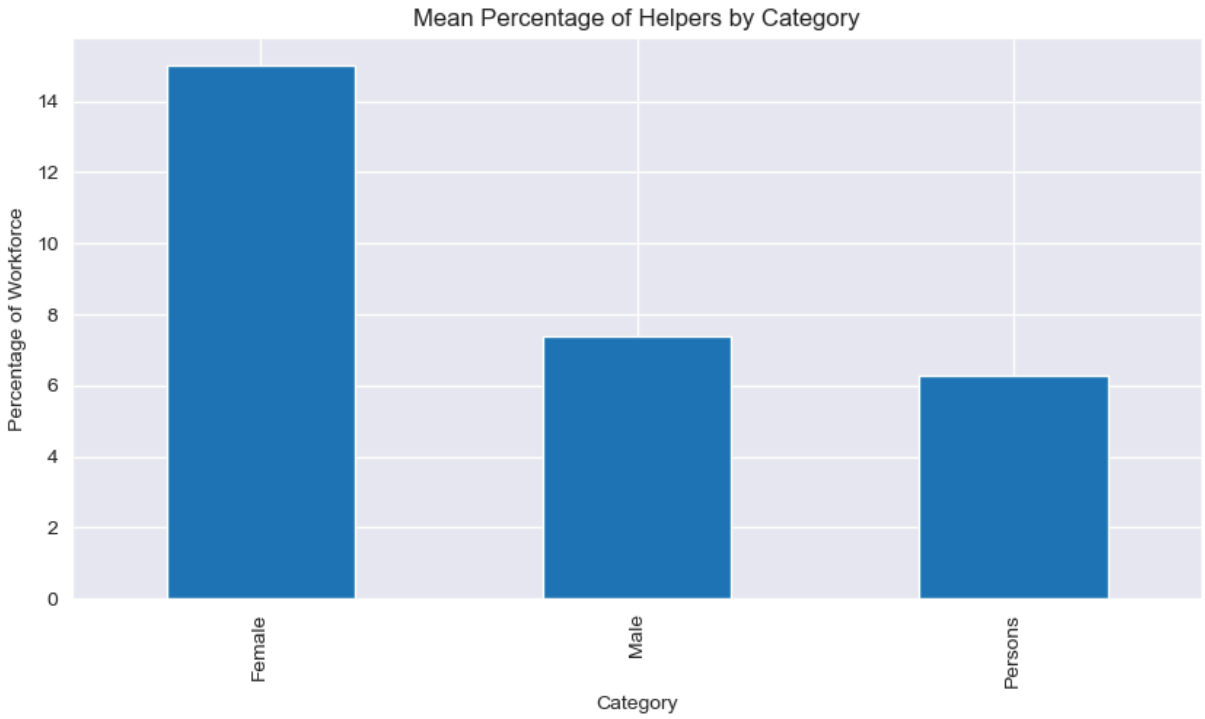


Conclusion:

- Histogram & KDE:

    - The distribution is right-skewed(positively), showing that most industries have low values of "Helpers in Household Enterprise"
    - The highest frequency is near 0% showing most industries employ very few people
    - long tail extends towards 20-50%, meaning few industries have some values of Helpers in household enterprise
- Boxplot:
    - median is low
    - There are several outliers beyond 30%
    - IQR is narrow, means that most values are concentrated in the lower range
- After doing groupby, we find that females have the highest mean out of the 3 category hence the outliers

In [19]:
```
h = [('Self-Employed','helper in household enterprise')]
c = emp.groupby('Category')[h].describe()
c[('Self-Employed','helper in household enterprise','mean')].plot(kind='bar'
plt.ylabel('Percentage of Workforce')
plt.show()
```



In [20]:
```
d = emp.groupby('Industry as per NIC-2008')[h].describe()
d
```

Out[20]:

	Self-Employed							
	helper in household enterprise							
	count	mean	std	min	25%	50%	75%	max
Industry as per NIC-2008								
01-03 (agriculture)	8.0	33.587500	13.327141	13.4	23.575	35.9	39.700	51.0
05-09 (mining & quarrying)	9.0	2.888889	3.737126	0.1	1.300	1.7	2.000	12.0
05-43 (secondary)	9.0	5.777778	3.920707	1.9	2.900	4.1	9.500	12.3
10-33 (manufacturing)	9.0	8.666667	4.123106	4.0	4.900	7.8	13.700	13.9
35-39 (electricity and water supply)	9.0	2.900000	1.987461	0.2	1.500	2.7	4.900	5.6
41-43 (construction)	9.0	0.822222	0.376755	0.5	0.500	0.7	1.100	1.4
45-47 (trade)	9.0	20.077778	14.014615	7.4	9.900	13.5	32.000	43.9
45-99 (tertiary)	9.0	8.222222	4.421758	4.3	5.000	6.6	9.800	17.4
49-53(transport)	9.0	0.466667	0.212132	0.0	0.400	0.5	0.600	0.7
55-56 (accommodation & food services)	8.0	24.700000	16.008569	8.7	13.600	17.4	38.825	49.4
55-56 (accommodation & food services))	1.0	10.200000	NaN	10.2	10.200	10.2	10.200	10.2
58-99 (other services)	9.0	1.744444	0.229734	1.3	1.700	1.7	1.800	2.1

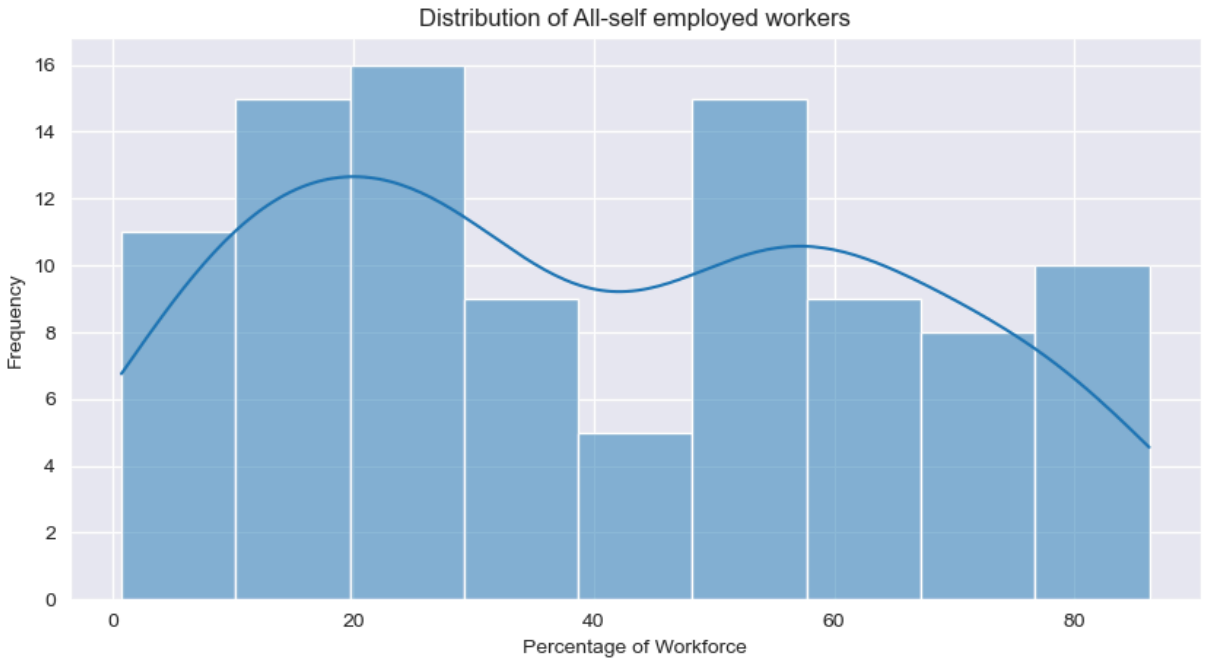
In [21]:
```
emp.head(2)
```

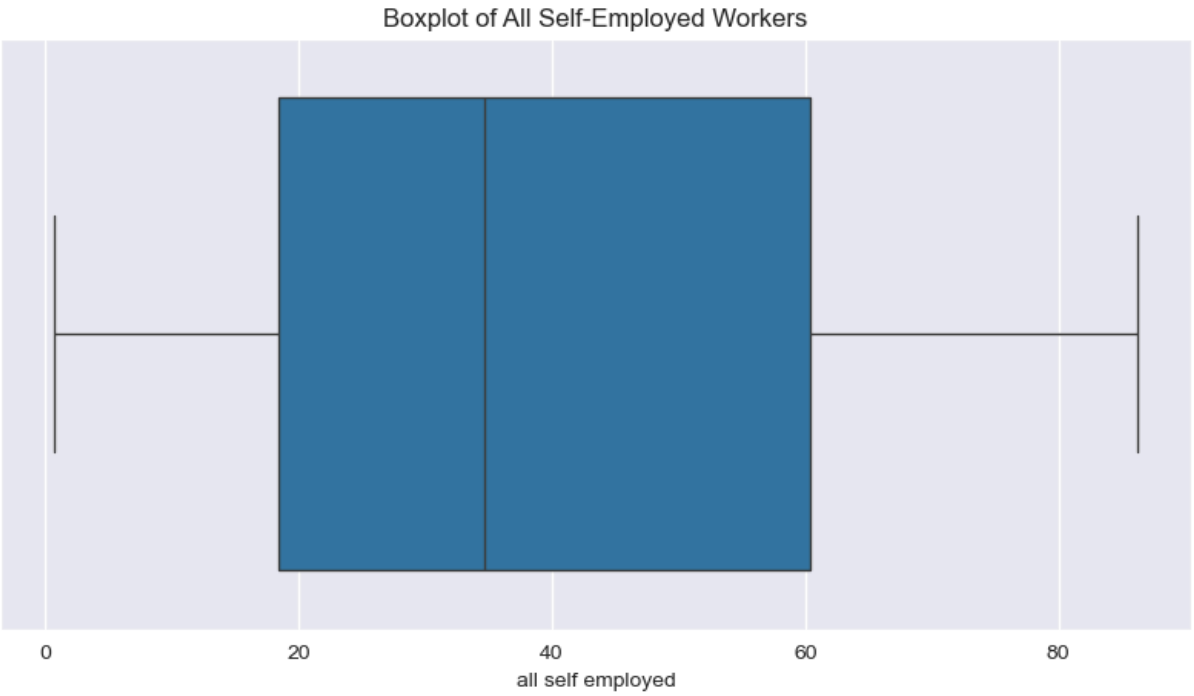
Out[21]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	all
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	100
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	100

3. All Self Employed

In [22]:
```
plt.figure(figsize=(10,5))
n = len(emp['Self-Employed']['all self employed'])
bins = int(np.sqrt(n)) # Square root rule for finding binsize
sns.histplot(emp['Self-Employed']['all self employed'], kde=True, bins=bins)
plt.xlabel('Percentage of Workforce')
plt.ylabel('Frequency')
plt.title('Distribution of All-self employed workers')
plt.show()

plt.figure(figsize=(10,5))
sns.boxplot(x=emp['Self-Employed']['all self employed'])
plt.title('Boxplot of All Self-Employed Workers')
plt.show()
```



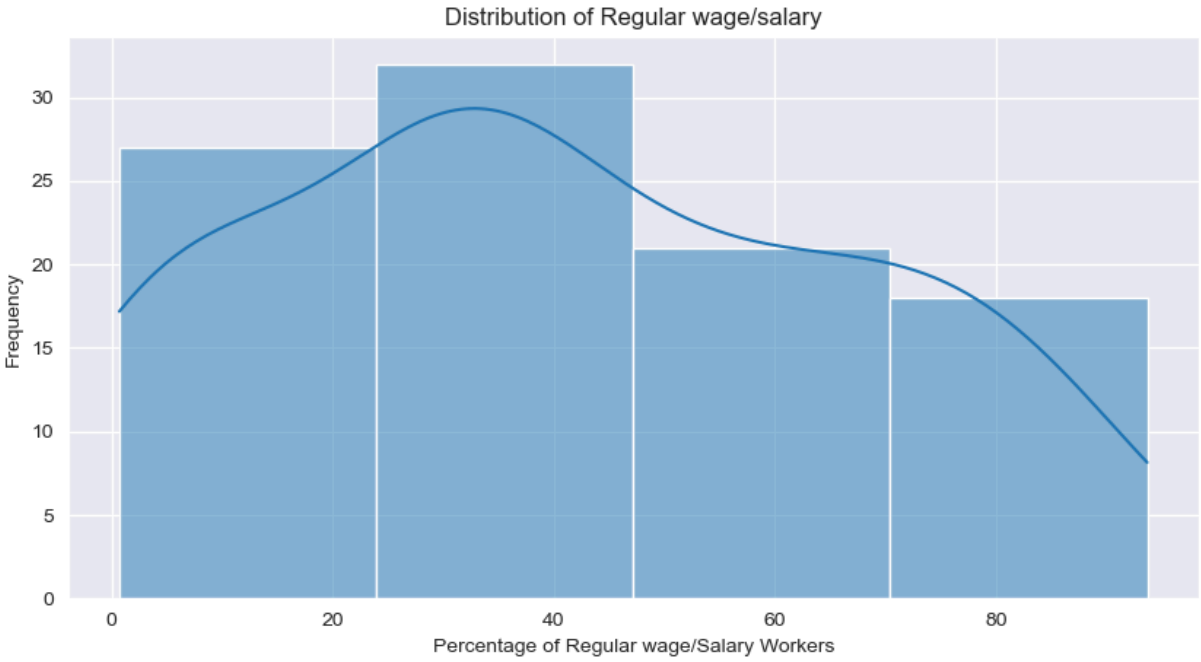


Conclusions

- Distribution is bimodial (two peaks)
- Frequency distribution is spread out
- No outliers present
- median line is below and near 40%
- IQR is large , the middle 50% of data is quite spread out.

4. Regular wage/Salary

In [23]:
```
plt.figure(figsize=(10,5))
n = len(emp['regular wage/salary'])
bins = int(np.sqrt(n)) # Square root rule for finding binsize
sns.histplot(emp['regular wage/salary'],kde=True, bins=4)
plt.xlabel('Percentage of Regular wage/Salary Workers')
plt.ylabel('Frequency')
plt.title('Distribution of Regular wage/salary')
plt.show()

plt.figure(figsize=(8,4))
sns.boxplot(x=emp['regular wage/salary'])
plt.title('Boxplot of Regular wage/salary')
plt.show()
```

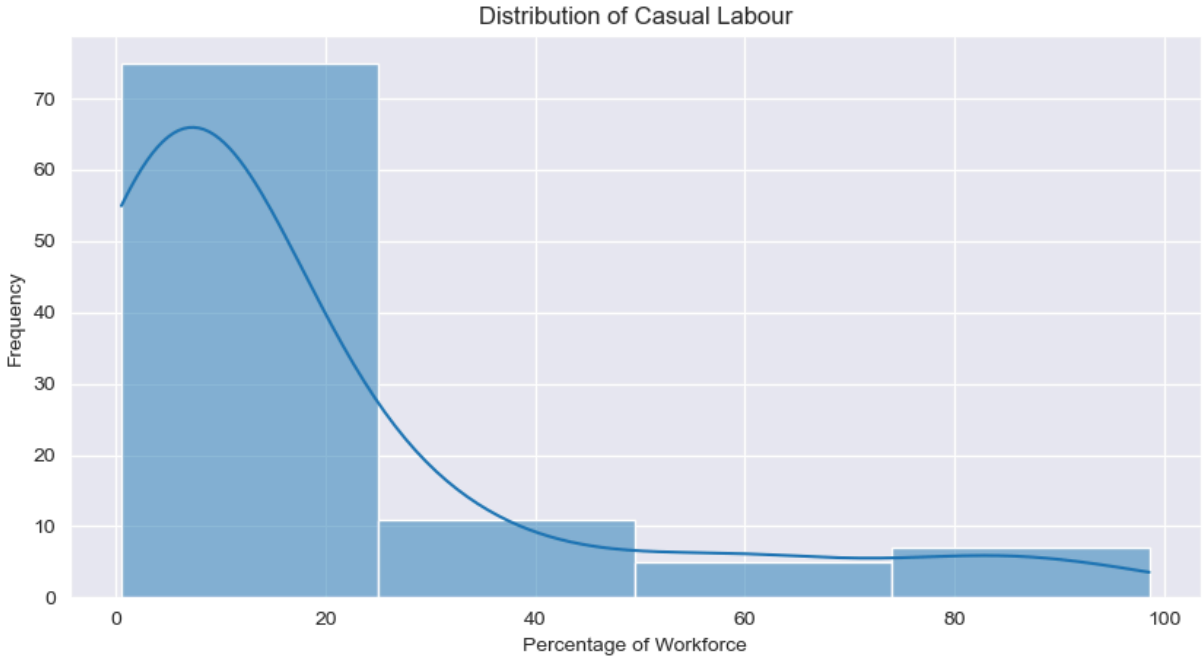


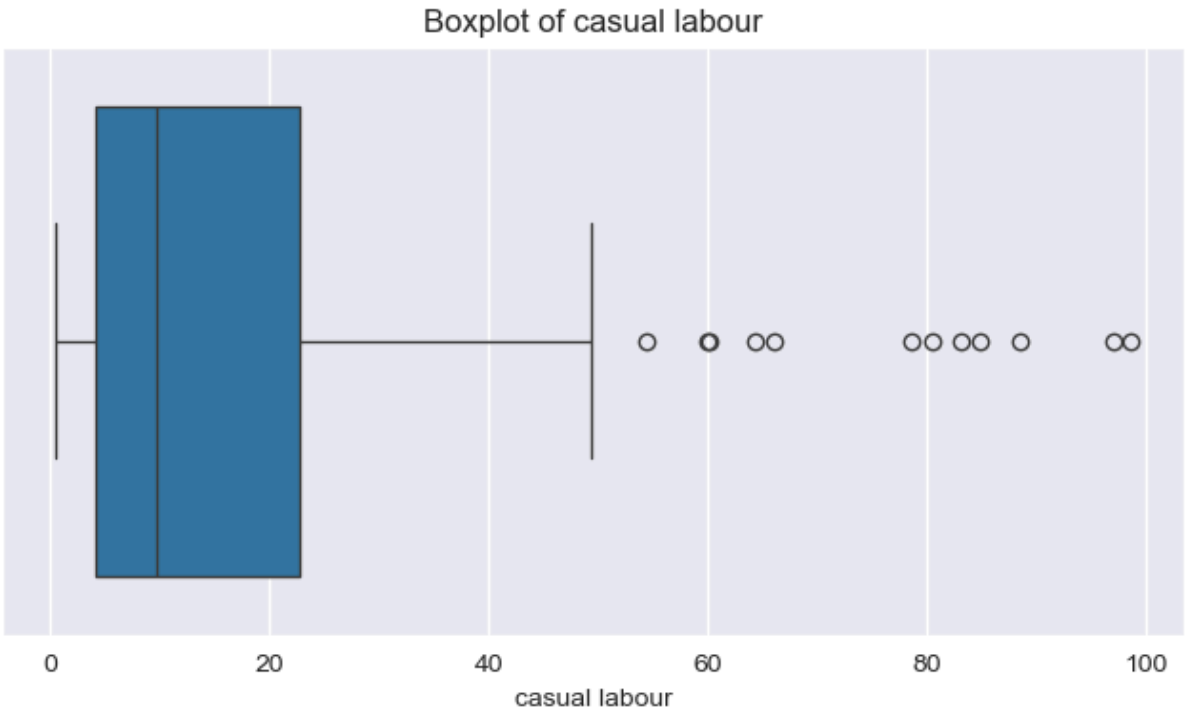
Conclusions

- Highest frequency is observed in the 30-40% range meaning most regions fall within this percentage
- Right-skewed, suggests a slight decline as we move toward higher percentages, fewer regions have a very high percentage of regular wage/salary.
- No outliers
- median value below and near 40% and IQR covering the middle which is quite spread out

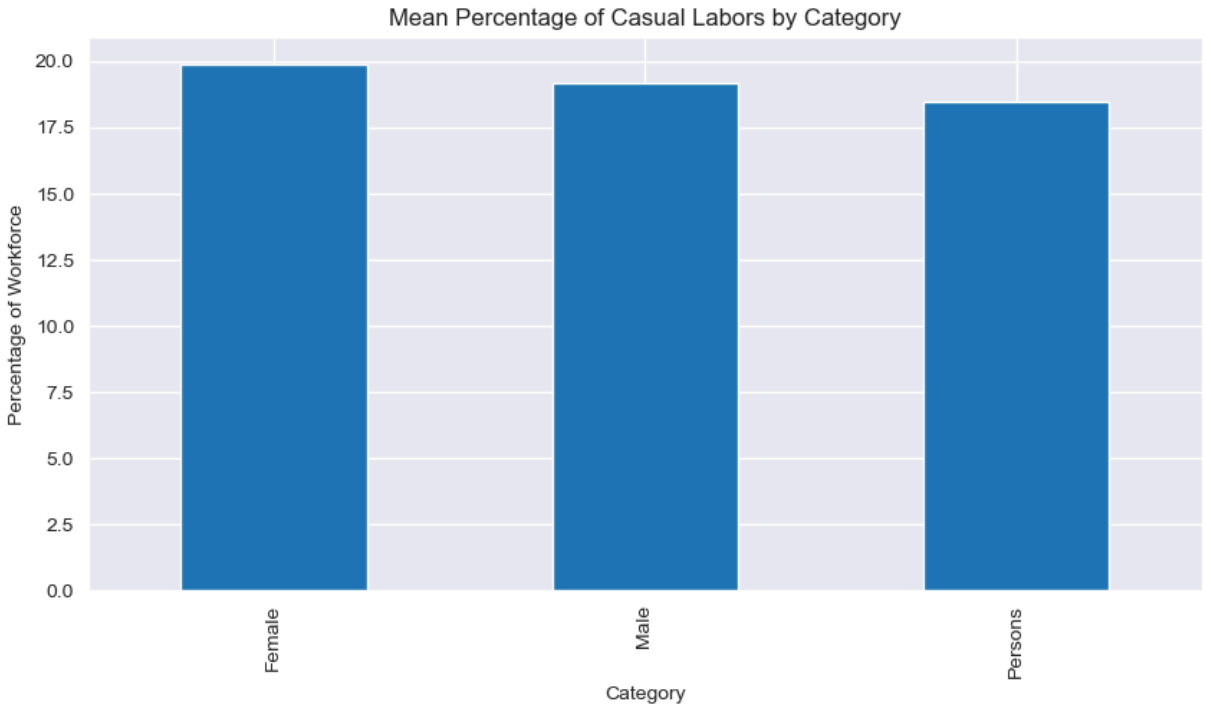
5. Casual Labor

In [24]:
```
plt.figure(figsize=(10,5))
sns.histplot(emp['casual labour'], kde=True, bins=4)
plt.title('Distribution of Casual Labour')
plt.xlabel('Percentage of Workforce')
plt.ylabel('Frequency')
plt.show()

plt.figure(figsize=(8,4))
sns.boxplot(x=emp['casual labour'])
plt.title('Boxplot of casual labour')
plt.show()
```





In [25]:
```
# h = [('Self-Employed','helper in household enterprise')]
c = emp.groupby('Category')['casual labour'].describe()
c['mean'].plot(kind='bar', figsize=(10,5), title='Mean Percentage of Casual
plt.ylabel('Percentage of Workforce')
plt.show()
```



Conclusions

- Histogram

    - Right Skewed Distribution: shows that most regions have a low percentage of casual labor with highest frequency in 0-20% range.
    - Long tail towards higher percentages: few regions where casual labor is significantly higher, above 50% but less frequent.
- Boxplot
    - Median close to the lower quartile
    - Outliers present, data has multiple outliers above 50% some regions have unusually high percentage of casual labor.
    - IQR is relatively small means that majority of the data falls within a narrow range (under 30%)
    - Casual Labor is generally low in most industries but a few industries deviate significantly
- Possible Analysis:
    - Investigate the industries with high casual labor participation - are they rural,industrial or economically underdeveloped?
    - Compare casual labor trends with regular wage/salary and self-employment for deeper insights.

Univariate Analysis on Categorical Columns

In [26]:
```
emp.head(2)
```

Out[26]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	all
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	100
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	100

In [27]:
```
emp['Industry as per NIC-2008'].value_counts()
```

Out[27]:
```
Industry as per NIC-2008
05-09 (mining & quarrying)                 9
10-33 (manufacturing)                      9
35-39 (electricity and  water supply)      9
41-43 (construction)                       9
05-43 (secondary)                          9
45-47 (trade)                              9
49-53( transport)                          9
58-99 (other services)                     9
45-99 (tertiary)                           9
01-03 (agriculture)                        8
55-56 (accommodation & food services)      8
55-56 (accommodation & food services))     1
Name: count, dtype: int64
```

In [28]:
```
emp[emp['Industry as per NIC-2008'] == '55-56 (accommodation & food services
```

Out[28]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	all
		own account worker, employer	helper in household enterprise	all self employed			
7	55-56 (accommodation & food services))	49.0	10.2	59.3	31.8	8.9	100

In [29]:
```
emp['Industry as per NIC-2008'] = emp['Industry as per NIC-2008'].replace('5
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\400728202.py:1: SettingWit
hCopyWarning:
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
  emp['Industry as per NIC-2008'] = emp['Industry as per NIC-2008'].replace
('55-56 (accommodation & food services))','55-56 (accommodation & food servi
ces) ')
```

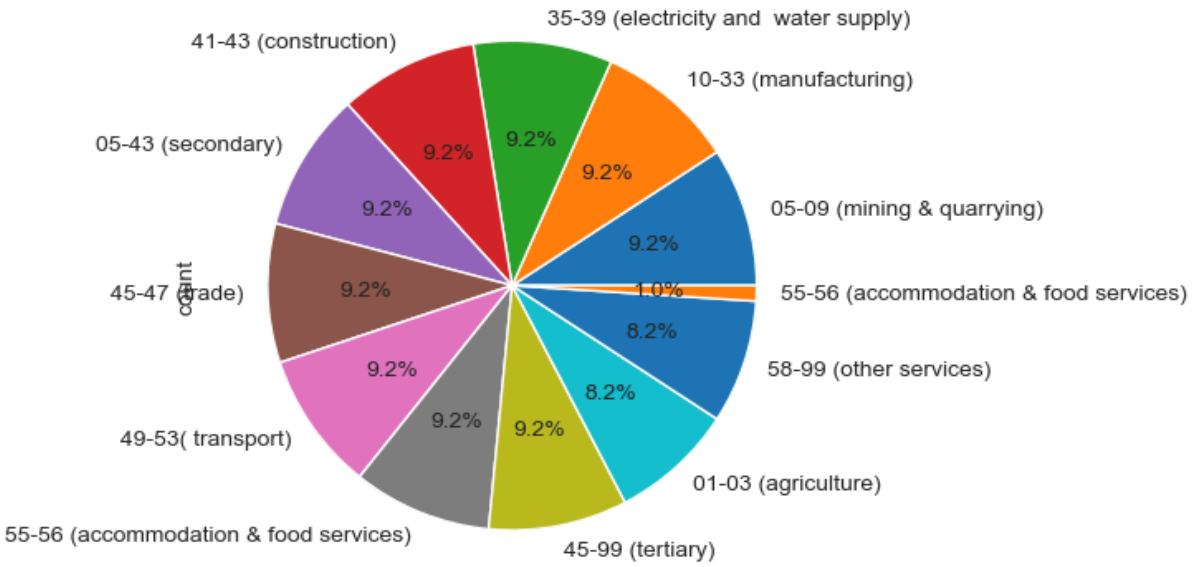
In [30]:
```
emp['Industry as per NIC-2008'].value_counts()
```

Out[30]:
```
Industry as per NIC-2008
05-09 (mining & quarrying)                9
10-33 (manufacturing)                     9
35-39 (electricity and  water supply)     9
41-43 (construction)                      9
05-43 (secondary)                         9
45-47 (trade)                             9
49-53( transport)                         9
58-99 (other services)                    9
45-99 (tertiary)                          9
01-03 (agriculture)                       8
55-56 (accommodation & food services)     8
55-56 (accommodation & food services)     1
Name: count, dtype: int64
```

In [31]:
```
emp.loc[8,'Industry as per NIC-2008'] = '55-56 (accommodation & food service
```

In [32]:
```
emp['Industry as per NIC-2008'].value_counts().plot(kind='pie',autopct='%0.1
```

Out[32]: <Axes: ylabel='count'>



In [33]:
```
emp.groupby('Category')[[('Self-Employed','helper in household enterprise')]
```

Out[33]:

Category	Self-Employed helper in household enterprise
Female	49.4
Male	51.0
Persons	21.7

In [34]:
```
emp.head()
```

Out[34]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	all
		own account worker, employer	helper in household enterprise	all self employed			
0	05-09 (mining & quarrying)	10.8	2.0	12.8	54.3	32.9	100
1	10-33 (manufacturing)	29.3	4.9	34.3	51.4	14.3	100
2	35-39 (electricity and water supply)	11.1	0.2	11.3	84.4	4.4	100
3	41-43 (construction)	12.8	0.5	13.3	3.6	83.1	100
4	05-43 (secondary)	17.8	1.9	19.6	20.2	60.1	100

Workforce Distribution & Trends

1. How are workers distributed across different employment types ?

2. Which Industries have the highest and lowest percentage of regular wage/salary workers?

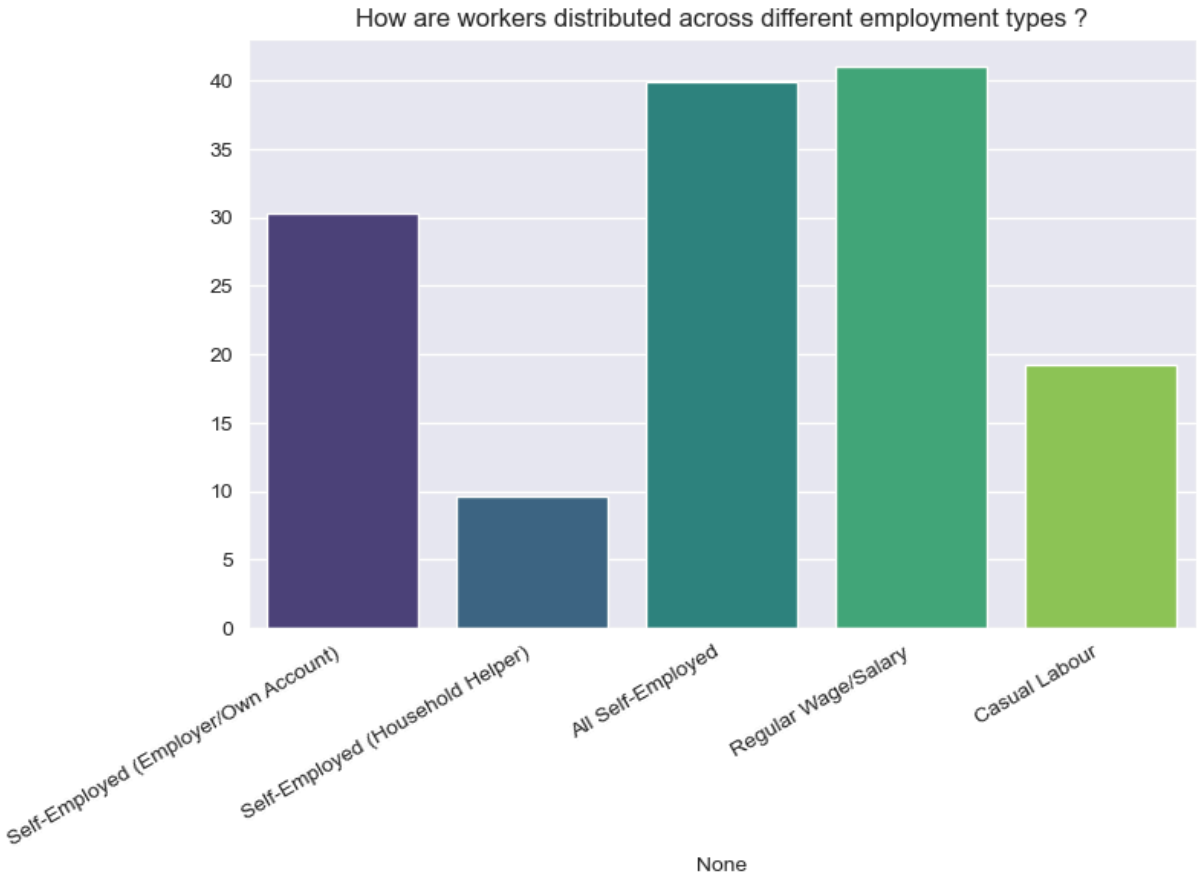
In [35]:
```
total = emp_industry[emp_industry['Industry as per NIC-2008'].str.contains('
total
```

Out[35]:

	Industry as per NIC-2008	Self-Employed			regular wage/salary	casual labour	all	Cat
		own account worker, employer	helper in household enterprise	all self employed				
10	total	47.0	12.4	59.4	15.8	24.9	100	
23	total	31.2	42.3	73.5	7.8	18.7	100	I
36	total	41.0	23.7	64.7	12.7	22.5	100	P
49	total	35.1	4.7	39.8	46.8	13.4	100	
62	total	28.5	13.8	42.3	49.4	8.3	100	I
75	total	33.4	7.0	40.4	47.5	12.1	100	P
88	total	43.5	10.1	53.6	24.9	21.5	100	
101	total	30.7	36.7	67.4	15.9	16.7	100	I
114	total	39.0	19.4	58.4	21.7	19.8	100	P

In [36]:
```
import matplotlib.pyplot as plt
```

In [37]:
```
employment_types = emp[[('Self-Employed','own account worker, employer'),('S

employment_distribution = employment_types.mean()

employment_distribution.index = ['Self-Employed (Employer/Own Account)',
                                 'Self-Employed (Household Helper)',
                                 'All Self-Employed',
                                 'Regular Wage/Salary',
                                 'Casual Labour']

plt.figure(figsize=(8,5))
sns.barplot(x=employment_distribution.index, y=employment_distribution.value
plt.title('How are workers distributed across different employment types ?')
# plt.ylable('Workforce')
plt.xticks(rotation=30,ha='right')
plt.show()
```



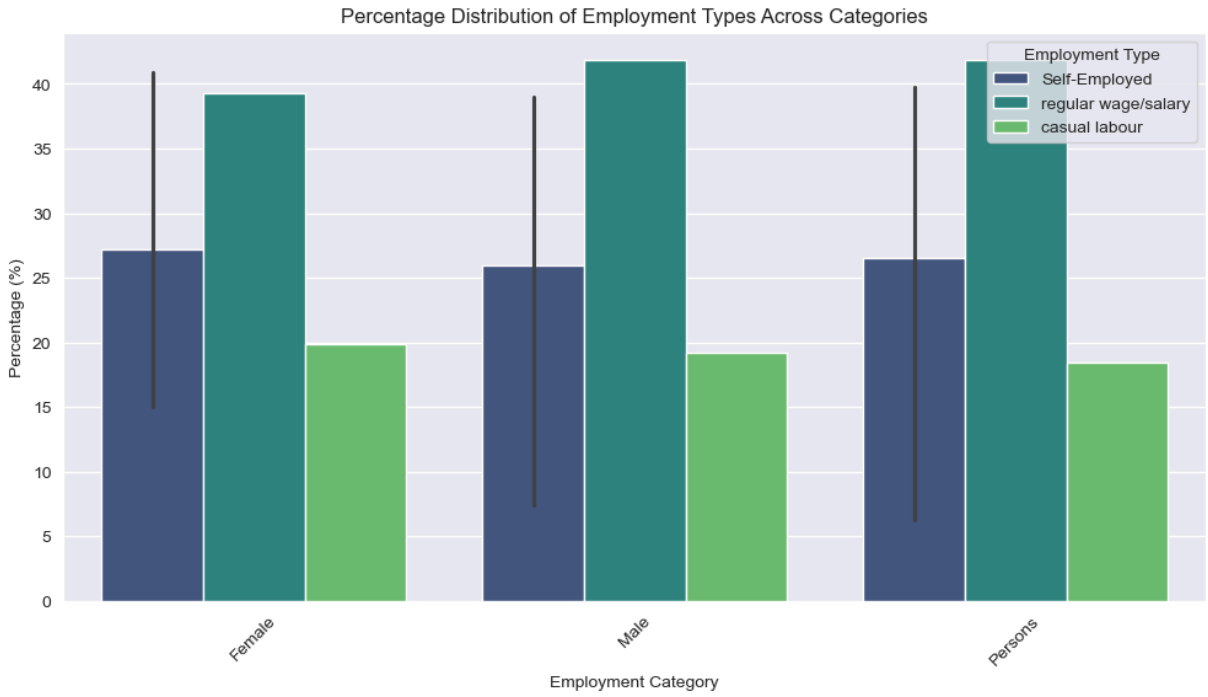
Key Insights:

- Which employment type has the highest percentage?: Regular Wage/Salary Employment type
- Which one is the least common?: Self-Employed(Household Helper)
- How do the employment types compare in terms of gender?: There is a greater number of females in self-employed type and for regular wage/salary majority is male and in casual labor too number of female worker is greater than male workers
- Does any type have an outlier (very high or low percentage)?: Yes, Out of all the employment types, Helper in Household enterprise and casual labors have outliers. The reason we have outliers in Household enterprise is because majority of workers are female and similary majority workers are female in the casual labour type

In [38]:
```
emp_types = [('Self-Employed','own account worker, employer'),('Self-Employe
grp = emp.groupby('Category')[emp_types].mean()
grp_t = grp.T
```

In [39]:
```
emp_types = [
    ('Self-Employed', 'own account worker, employer'),
    ('Self-Employed', 'helper in household enterprise'),
    ('Self-Employed', 'all self employed'),
```

```
    ('regular wage/salary', ''),
    ('casual labour', '')
]

# Group by 'Category' and compute mean percentage
grp = emp.groupby('Category')[emp_types].mean()
grp = grp.reset_index()
grp
grp_melted = grp.melt(id_vars=[('Category','')], var_name='Employment Type',

plt.figure(figsize=(12, 6))
sns.barplot(data=grp_melted, x=('Category',''), y='Percentage', hue='Employm

# Formatting
plt.title("Percentage Distribution of Employment Types Across Categories")
plt.ylabel("Percentage (%)")
plt.xlabel("Employment Category")
plt.xticks(rotation=45)
plt.legend(title="Employment Type")

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



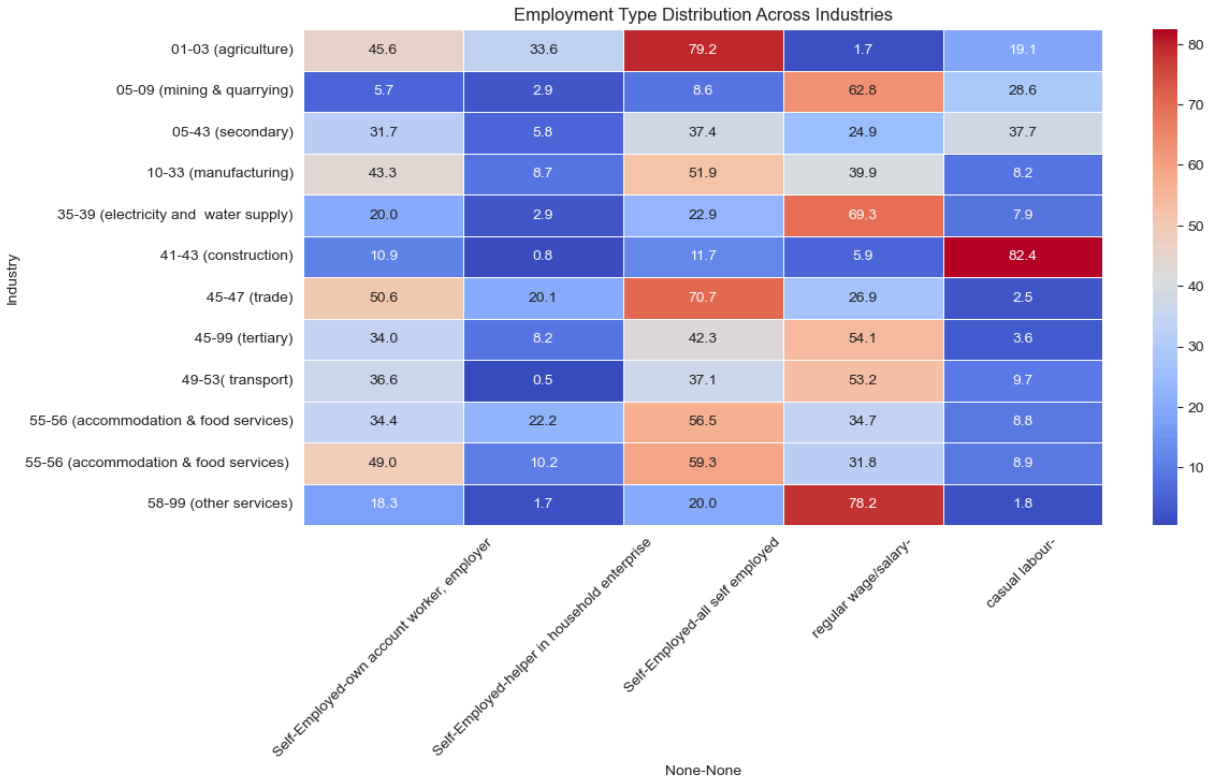
In [40]:
```
grp_melted
```

Out[40]:

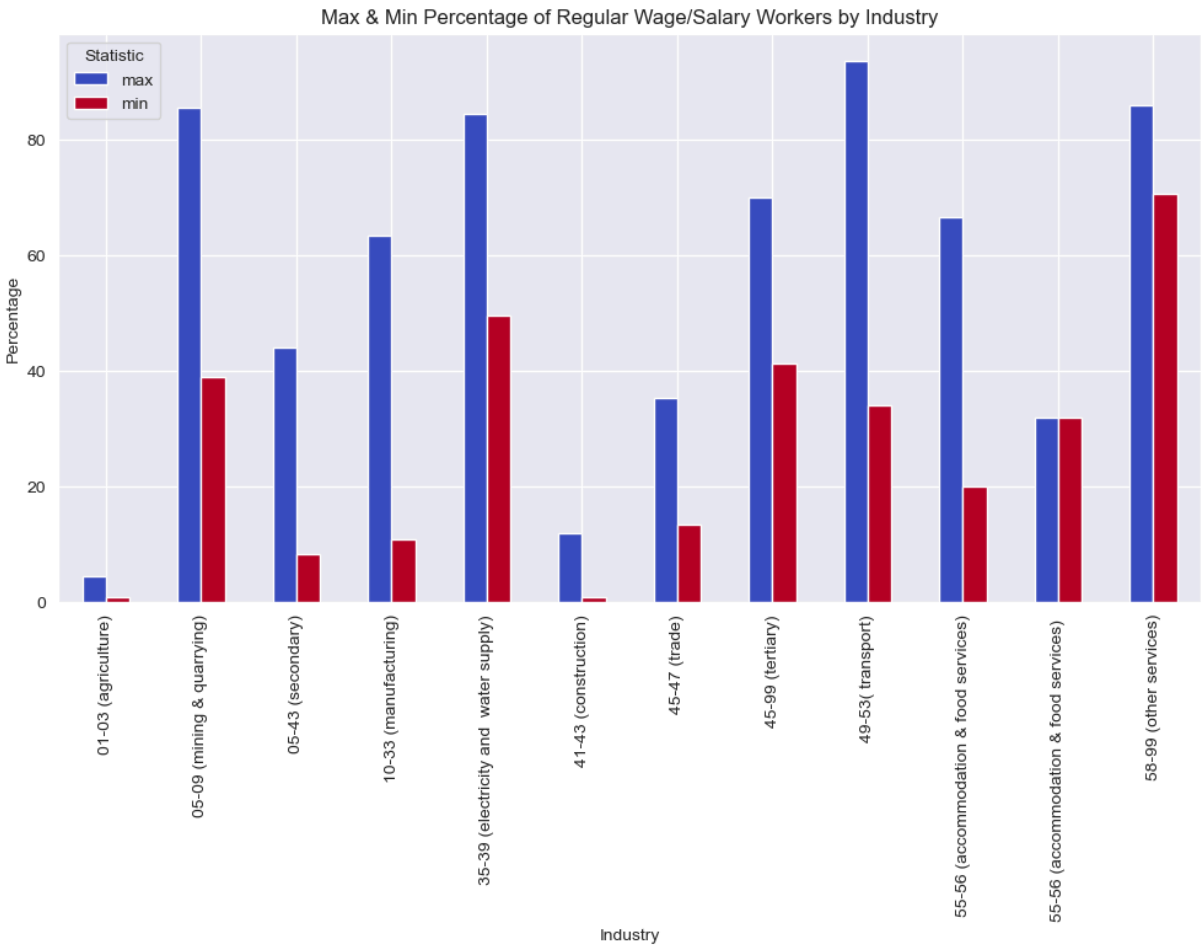
	(Category,)	Employment Type	Percentage
0	Female	Self-Employed	25.833333
1	Male	Self-Employed	31.596970
2	Persons	Self-Employed	33.431250
3	Female	Self-Employed	15.018182
4	Male	Self-Employed	7.381818
5	Persons	Self-Employed	6.293750
6	Female	Self-Employed	40.851515
7	Male	Self-Employed	38.975758
8	Persons	Self-Employed	39.725000
9	Female	regular wage/salary	39.263636
10	Male	regular wage/salary	41.872727
11	Persons	regular wage/salary	41.806250
12	Female	casual labour	19.900000
13	Male	casual labour	19.154545
14	Persons	casual labour	18.478125

In [41]:

```
plt.figure(figsize=(12,6))
industry_grouped = emp.groupby("Industry as per NIC-2008")[emp_types].mean()
industry_grouped
# Plot heatmap
sns.heatmap(industry_grouped, annot=True, cmap="coolwarm", fmt=".1f", linewi
plt.title("Employment Type Distribution Across Industries")
plt.ylabel("Industry")
plt.xticks(rotation=45)
plt.show()
```



In [42]:
```
reg = emp.groupby('Industry as per NIC-2008')['regular wage/salary'].agg(['m

reg.plot(kind='bar',figsize=(12,6),colormap='coolwarm')

plt.title('Max & Min Percentage of Regular Wage/Salary Workers by Industry')
plt.ylabel('Percentage')
plt.xlabel('Industry')
plt.xticks(rotation=90)  # Rotate x labels for readability
plt.legend(title='Statistic')

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



Industry-Specific Insights

1. What percentage of workers are in casual labour across different industries?

2. Which industries have the highest rates of self-employment?

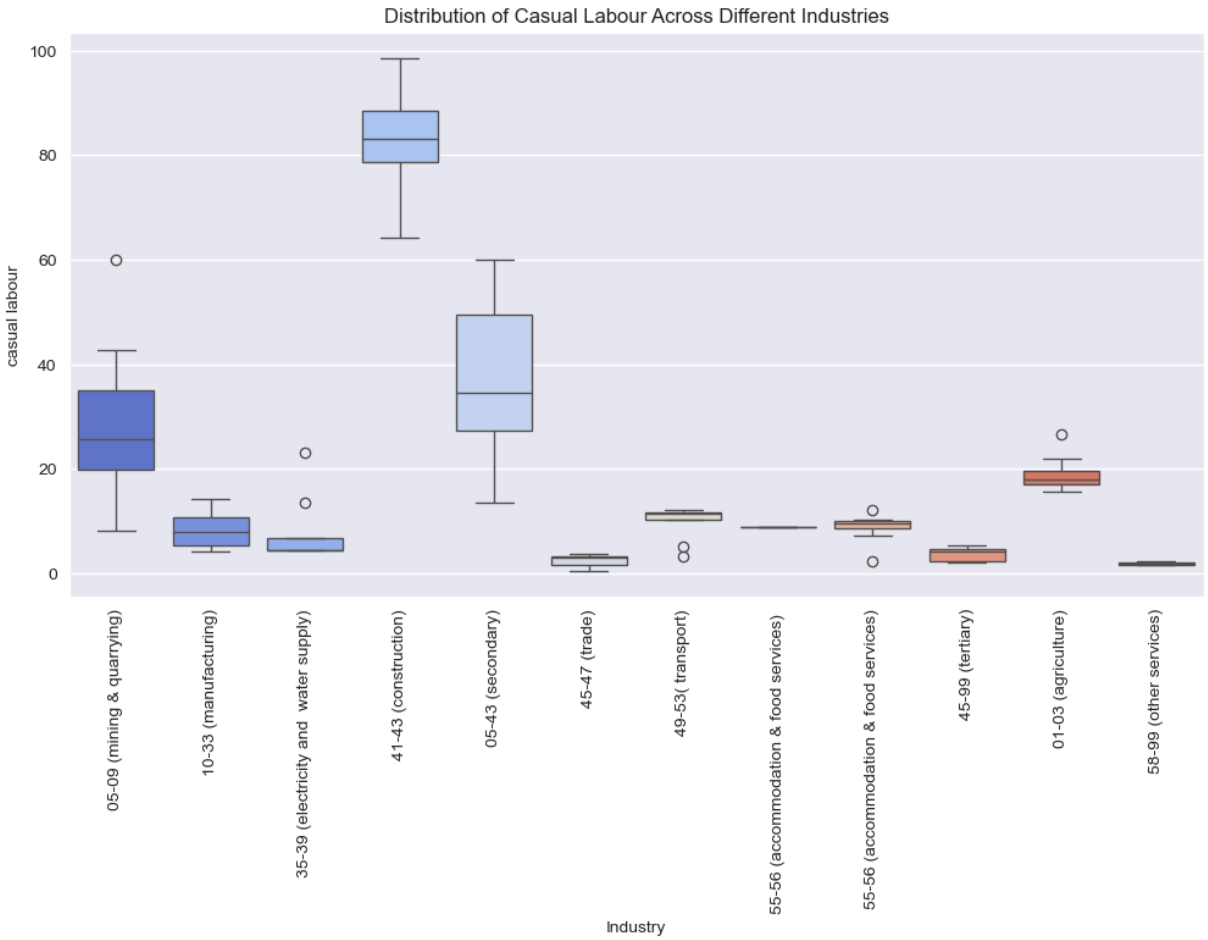
In [43]:
```
emp.groupby('Industry as per NIC-2008')['casual labour'].describe()
```

Out[43]:

Industry as per NIC-2008	count	mean	std	min	25%	50%	75%	max
01-03 (agriculture)	8.0	19.100000	3.614257	15.5	16.95	18.05	19.525	26.7
05-09 (mining & quarrying)	9.0	28.611111	16.366157	8.1	19.90	25.60	35.000	60.0
05-43 (secondary)	9.0	37.655556	15.334692	13.4	27.20	34.50	49.400	60.1
10-33 (manufacturing)	9.0	8.155556	3.375319	4.1	5.30	7.80	10.700	14.3
35-39 (electricity and water supply)	9.0	7.877778	6.394485	4.4	4.40	4.40	6.800	23.0
41-43 (construction)	9.0	82.411111	11.887645	64.3	78.60	83.10	88.500	98.6
45-47 (trade)	9.0	2.466667	1.137981	0.5	1.60	3.00	3.300	3.6
45-99 (tertiary)	9.0	3.633333	1.242980	2.0	2.20	4.10	4.600	5.3
49-53(transport)	9.0	9.722222	3.248760	3.2	10.30	11.30	11.700	12.0
55-56 (accommodation & food services)	9.0	8.800000	2.711088	2.4	8.60	9.50	9.900	12.0
55-56 (accommodation & food services)	1.0	8.900000	NaN	8.9	8.90	8.90	8.900	8.9
58-99 (other services)	8.0	1.800000	0.297610	1.5	1.50	1.80	1.950	2.3

In [44]:

```
plt.figure(figsize=(12,6))
sns.boxplot(data=emp,x='Industry as per NIC-2008',y='casual labour', palette
plt.title('Distribution of Casual Labour Across Different Industries')
# plt.ylabel('Percentage of Casual Labour')
plt.xlabel('Industry')
plt.xticks(rotation=90)
plt.show()
```



Gender & Employment Type Analysis

Are men more likely to be self-employed compared to women? (refer to above graph on category and employment type)

Do women have a higher self-employment rates by gender using a bar-chart or grouped bar-chart

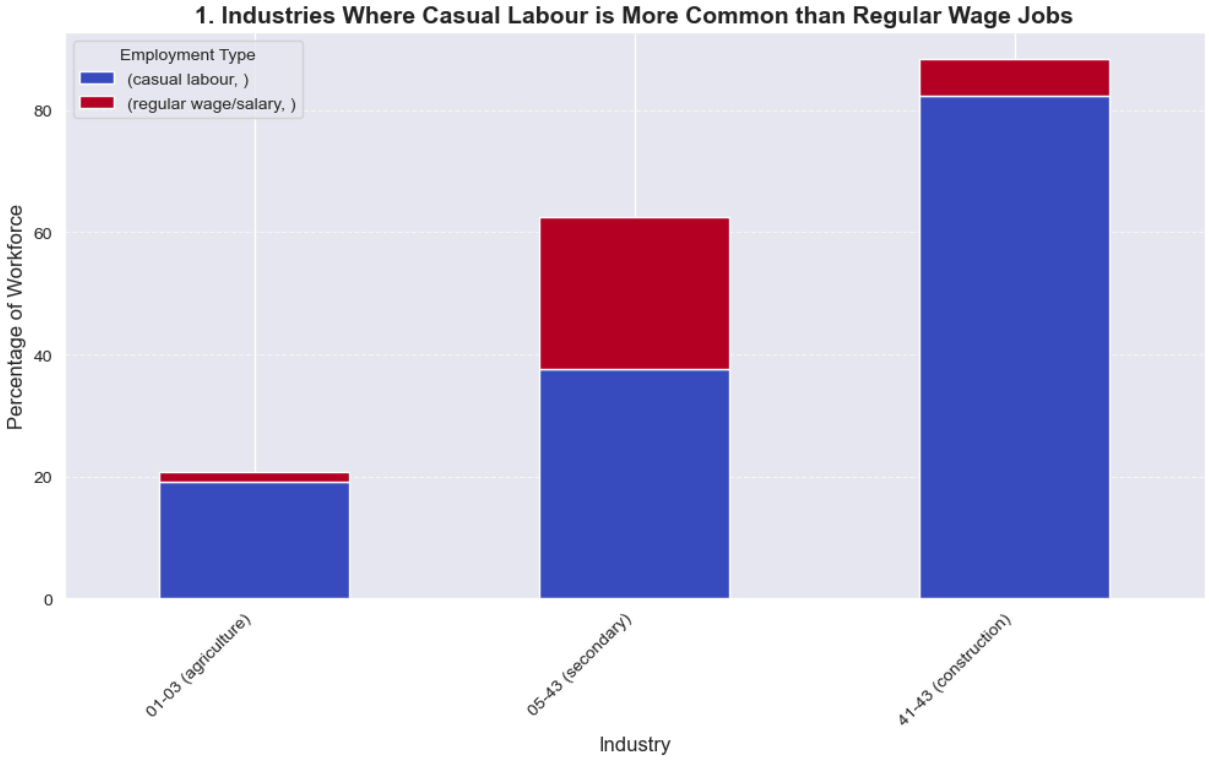
Casual Labour & Informal Economy

1. Are there industries where casual labour is more common than regular wage/salary jobs?

2. What does the distribution of casual labour looks like across industries?

In [45]:
```
col = [('casual labour',''),('regular wage/salary','')]
industry_grp = emp.groupby('Industry as per NIC-2008')[col].mean()
industry_grp
high_casual_labour = industry_grp[industry_grp['casual labour'] > industry_g
```

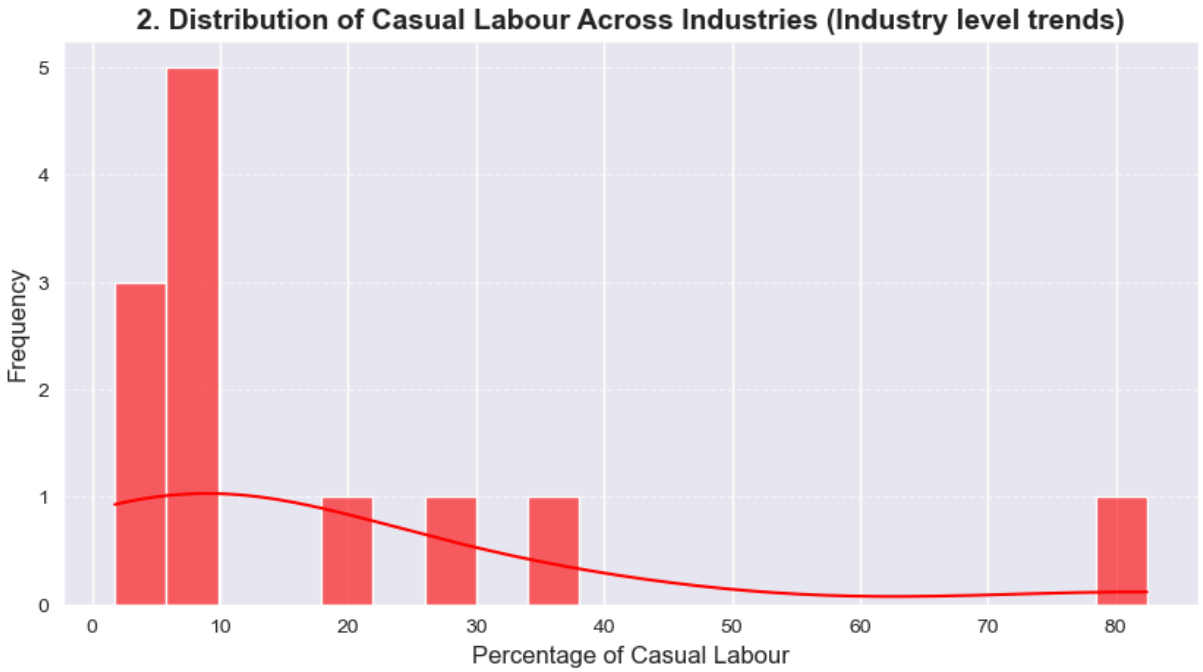
```
high_casual_labour.plot(kind='bar', stacked=True, figsize=(12,6), colormap='

plt.title('1. Industries Where Casual Labour is More Common than Regular Wag
plt.ylabel('Percentage of Workforce', fontsize=12)
plt.xlabel('Industry', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.legend(title="Employment Type")
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Show plot
plt.show()
```



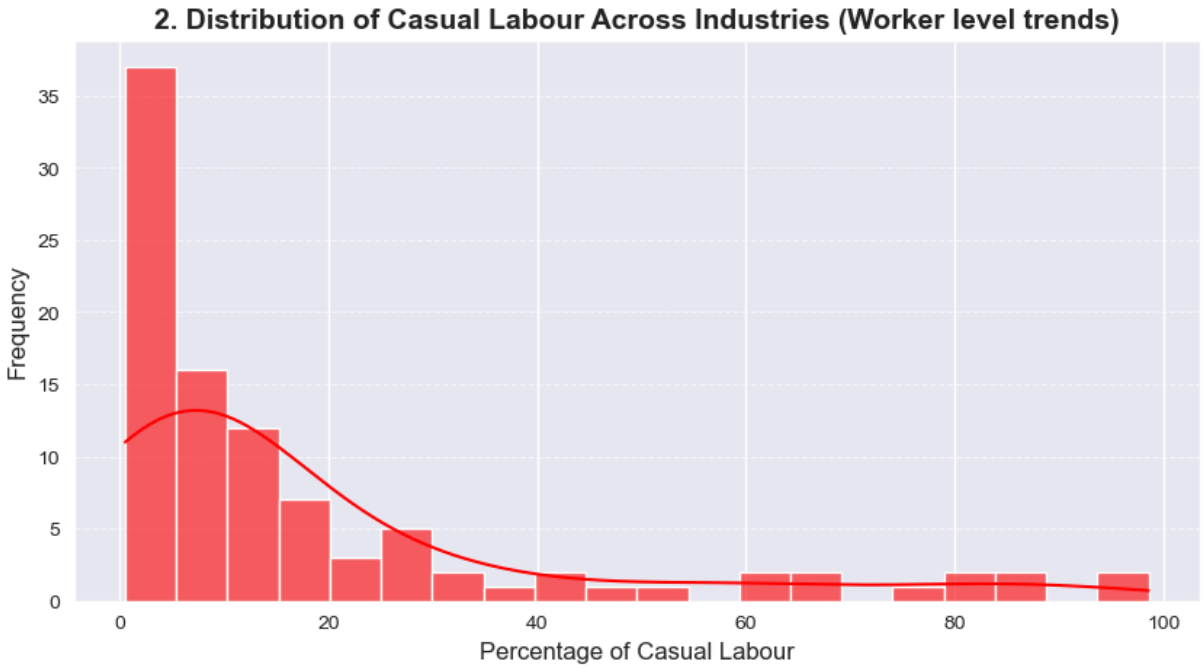
Conclusions:

- Construction has the highest percentage of casual labour
- Agriculture is also heavily casual labour dependent
- Secondary industries have a more mixed employment pattern

In [46]:
```
plt.figure(figsize=(10,5))

sns.histplot(industry_grp['casual labour'],bins=20,kde=True,color='red',alph
plt.title("2. Distribution of Casual Labour Across Industries (Industry leve
plt.xlabel('Percentage of Casual Labour',fontsize=12)
plt.ylabel('Frequency',fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

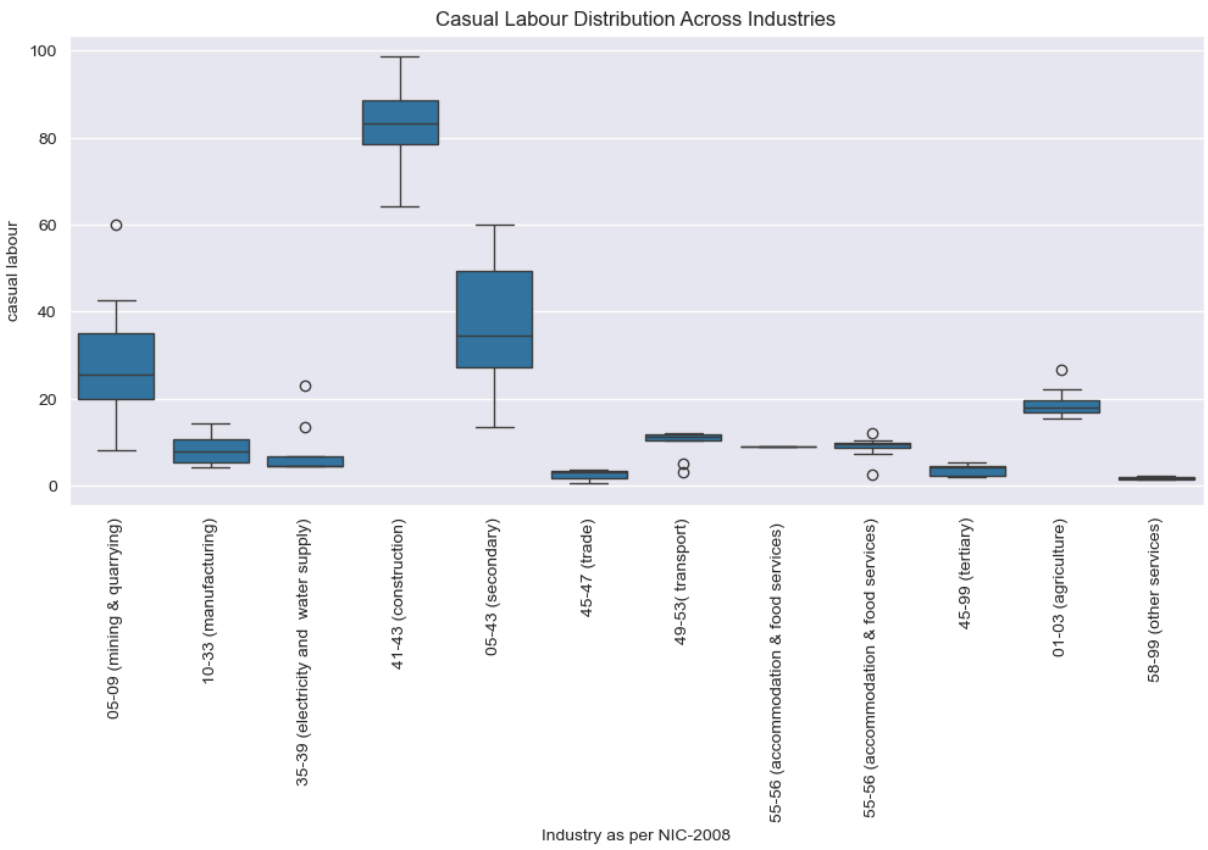


In [47]:
```
plt.figure(figsize=(10,5))

sns.histplot(emp['casual labour'],bins=20,kde=True,color='red',alpha=0.6)
plt.title("2. Distribution of Casual Labour Across Industries (Worker level
plt.xlabel('Percentage of Casual Labour',fontsize=12)
plt.ylabel('Frequency',fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



In [48]:
```
plt.figure(figsize=(12,5))
sns.boxplot(data=emp, x="Industry as per NIC-2008", y="casual labour")
plt.xticks(rotation=90)
```

```
plt.title("Casual Labour Distribution Across Industries")
plt.show()
```



Conclusions:

- Industry-level Trends Graph:
  - Each bar represents average casual labour percentage per industry.
  - Shows most industries have a casual labour percentage below 20%, but a few industries have much higher rates(~80%)
  - The KDE(curve) suggests that industries mostly fall into low casual labour categories
  - Helps analyze which industries rely more or less on casual labour
- Worker-level Trends Graph:
  - Each bar represents individual data points of casual labour
  - Shows most workers fall in the lower casual labour range (0-20%), but some workers are in industries with high casual labour (80-100%)
  - KDE suggests a skewed distribution meaning casual labour is more common in specific industries
  - Helps understand how widespread casual labour is across workers rather than just industries.

EDA on PLFS Employment State dataset

In [49]:
```
emp_state = pd.read_csv('Cleaned Data/PLFS/PLFS_emp_state.csv',index_col=Fal
```

In [50]:
```
emp_state
```

Out[50]:

	Unnamed: 0	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	C l
0	0	Andhra Pradesh	47.8	6.8	54.6	16.1	
1	1	Arunachal Pradesh	60.1	12.8	72.9	19.9	
2	2	Assam	47.9	6.1	54.0	17.2	
3	3	Bihar	50.4	11.3	61.7	8.4	
4	4	Chhattisgarh	46.9	20.7	67.6	13.2	
...	...	...	...	...	...	...	...
325	325	Jammu & Kashmir	51.6	15.1	66.7	22.1	
326	326	Ladakh	57.4	8.3	65.7	25.5	
327	327	Lakshadweep	29.0	1.3	30.3	45.4	
328	328	Puducherry	19.7	1.9	21.6	56.9	
329	329	all India	39.0	19.4	58.4	21.7	

330 rows × 8 columns

In [51]:
```
emp_state = emp_state.drop('Unnamed: 0', axis=1)
```

In [52]:
```
emp_state.head()
```

Out[52]:

	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All
0	Andhra Pradesh	47.8	6.8	54.6	16.1	29.3	100
1	Arunachal Pradesh	60.1	12.8	72.9	19.9	7.2	100
2	Assam	47.9	6.1	54.0	17.2	28.8	100
3	Bihar	50.4	11.3	61.7	8.4	29.9	100
4	Chhattisgarh	46.9	20.7	67.6	13.2	19.3	100

In [53]:
```
emp_state.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 330 entries, 0 to 329
Data columns (total 7 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   State/UT                        330 non-null    object
 1   Own account, worker, employer   330 non-null    float64
 2   Helper in household enterprise  330 non-null    float64
 3   All self employed               330 non-null    float64
 4   Regular wage/salary             330 non-null    float64
 5   Casual labour                   330 non-null    float64
 6   All                             330 non-null    int64
dtypes: float64(5), int64(1), object(1)
memory usage: 18.2+ KB
```

In [54]:
```
categories = ['Male', 'Female', 'Persons'] * 9

rows_per_set = 36

category_col = [categories[i // rows_per_set] for i in range(len(emp_state))]
emp_state['Category'] = category_col
emp_state
```

Out[54]:

	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All
0	Andhra Pradesh	47.8	6.8	54.6	16.1	29.3	100
1	Arunachal Pradesh	60.1	12.8	72.9	19.9	7.2	100
2	Assam	47.9	6.1	54.0	17.2	28.8	100
3	Bihar	50.4	11.3	61.7	8.4	29.9	100
4	Chhattisgarh	46.9	20.7	67.6	13.2	19.3	100
...	...	...	...	...	...	...	...
325	Jammu & Kashmir	51.6	15.1	66.7	22.1	11.1	100
326	Ladakh	57.4	8.3	65.7	25.5	8.8	100
327	Lakshadweep	29.0	1.3	30.3	45.4	24.3	100
328	Puducherry	19.7	1.9	21.6	56.9	21.5	100
329	all India	39.0	19.4	58.4	21.7	19.8	100

330 rows × 8 columns

In [55]:
```
emp_state.iloc[55]
```

Out[55]:
```
State/UT                           Odisha
Own account, worker, employer        27.2
Helper in household enterprise       48.6
All self employed                    75.8
Regular wage/salary                   5.3
Casual labour                        18.9
All                                   100
Category                           Female
Name: 55, dtype: object
```

In [56]:
```
# Filtered out all India rows from the dataframe
emp_1 = emp_state[~emp_state['State/UT'].str.contains('all India',case=False
all_india = emp_state[emp_state['State/UT'].str.contains('all India',case=Fa
```

In [57]:
```
all_india.sort_values(by='Category',inplace=True)
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\4198808067.py:1: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  all_india.sort_values(by='Category',inplace=True)
```

In [58]:
```
all_india.reset_index(drop=True,inplace=True)
```

In [59]:
```
all_india
```

Out[59]:

	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All	Categ
0	all India	31.2	42.3	73.5	7.8	18.7	100	Fer
1	all India	35.1	4.7	39.8	46.8	13.4	100	Fer
2	all India	43.5	10.1	53.6	24.9	21.5	100	Fer
3	all India	47.0	12.4	59.4	15.8	24.9	100	
4	all India	33.4	7.0	40.4	47.5	12.1	100	
5	all India	39.0	19.4	58.4	21.7	19.8	100	
6	all India	41.0	23.7	64.7	12.7	22.5	100	Per
7	all India	28.5	13.8	42.3	49.4	8.3	100	Per
8	all India	30.7	36.7	67.4	15.9	15.9	100	Per

In [60]:
```
emp_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 321 entries, 0 to 328
Data columns (total 8 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   State/UT                        321 non-null    object
 1   Own account, worker, employer   321 non-null    float64
 2   Helper in household enterprise  321 non-null    float64
 3   All self employed               321 non-null    float64
 4   Regular wage/salary             321 non-null    float64
 5   Casual labour                   321 non-null    float64
 6   All                             321 non-null    int64
 7   Category                        321 non-null    object
dtypes: float64(5), int64(1), object(2)
memory usage: 22.6+ KB
```

Column Types:

- `Numerical` : own account worker, employer; helper in household enterprise all self ; all self employed; regular wage/salary; casual labour; all
- `Categorical` : Category
- `Mixed` : State/UT

Univariate Analysis on Numerical Cols

Own account, worker, employer

In [61]:
```
emp_1.head()
```

Out[61]:

	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All	Ca
0	Andhra Pradesh	47.8	6.8	54.6	16.1	29.3	100	
1	Arunachal Pradesh	60.1	12.8	72.9	19.9	7.2	100	
2	Assam	47.9	6.1	54.0	17.2	28.8	100	
3	Bihar	50.4	11.3	61.7	8.4	29.9	100	
4	Chhattisgarh	46.9	20.7	67.6	13.2	19.3	100	

In [62]:
```
emp_1.describe()
```

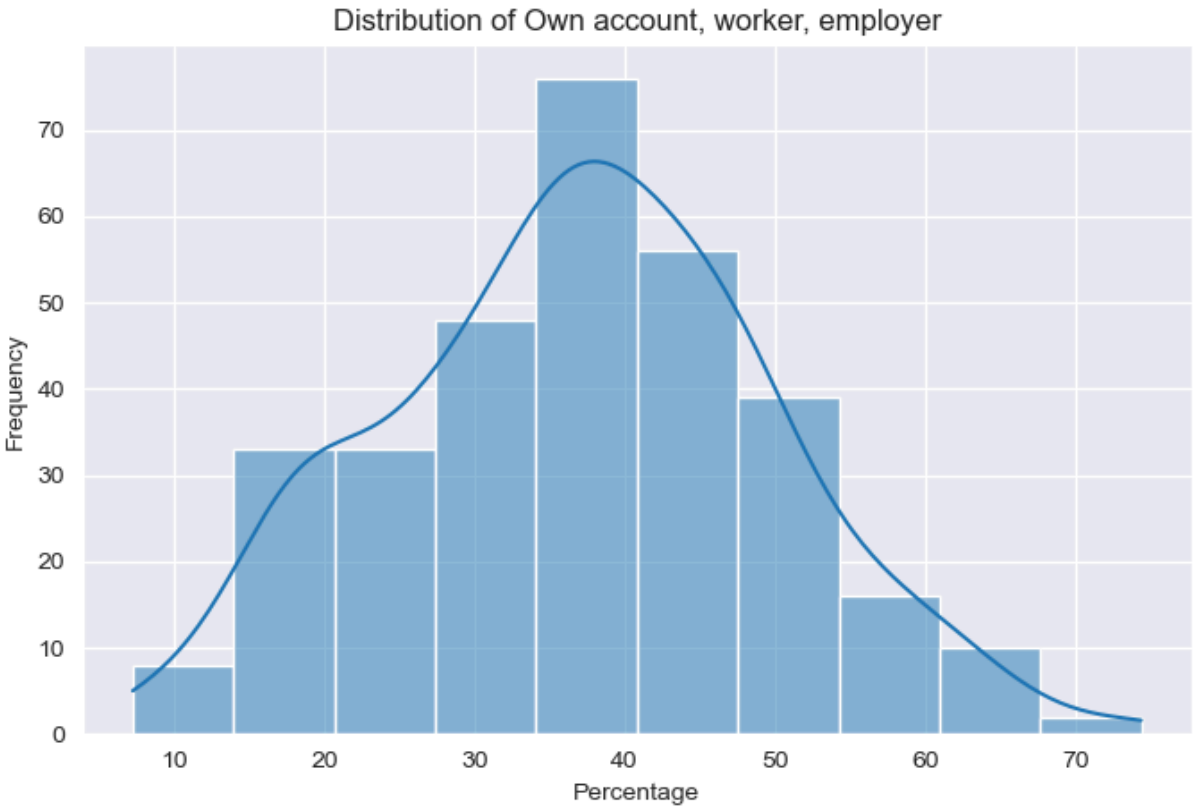
Out[62]:

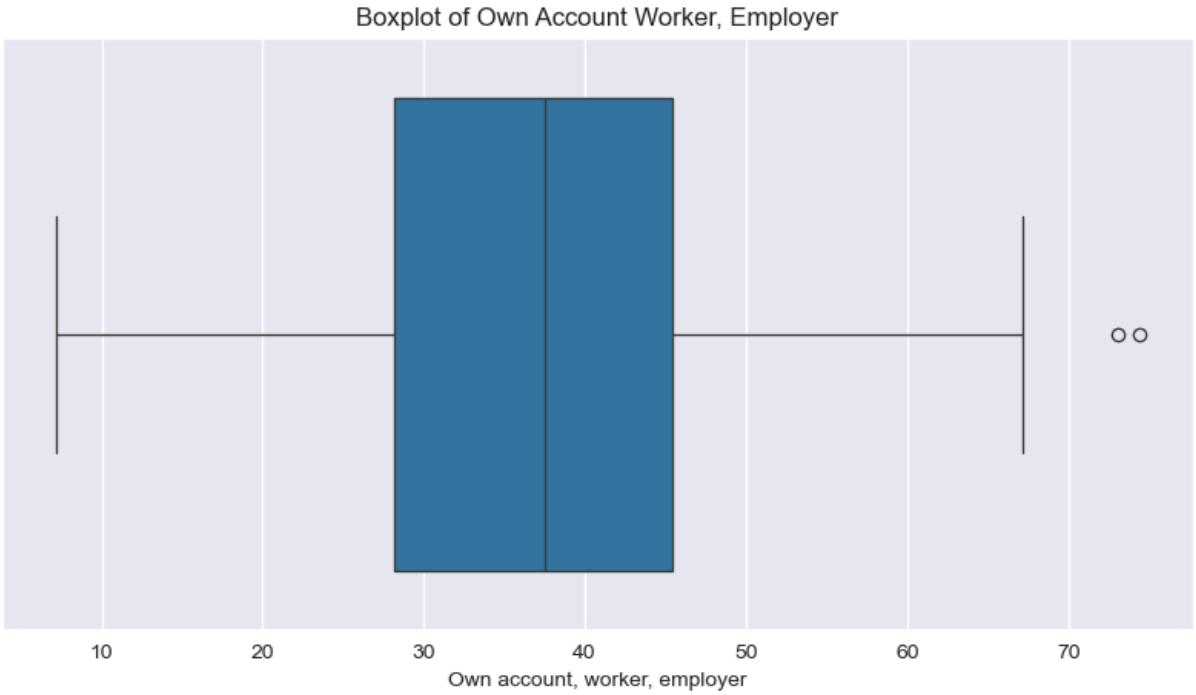
	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All
count	321.000000	321.000000	321.000000	321.000000	321.000000	321.0
mean	37.002181	15.409657	52.413396	33.019003	14.566978	100.0
std	12.816546	13.696010	17.909787	19.157565	9.485046	0.0
min	7.200000	0.000000	13.300000	2.200000	0.000000	100.0
25%	28.200000	5.700000	38.800000	16.600000	7.200000	100.0
50%	37.500000	10.700000	52.200000	30.900000	13.100000	100.0
75%	45.400000	22.200000	65.600000	46.800000	20.000000	100.0
max	74.400000	67.100000	94.500000	86.700000	47.200000	100.0

In [63]:
```
emp_1['Own account, worker, employer'].skew()
```

Out[63]:
```
np.float64(0.06893745791627617)
```

In [64]:
```
plt.figure(figsize=(8,5))
sns.histplot(emp_1['Own account, worker, employer'],kde=True,bins=10)
plt.title('Distribution of Own account, worker, employer')
plt.xlabel('Percentage')
plt.ylabel('Frequency')
plt.show()

#Boxplot
plt.figure(figsize=(10,5))
sns.boxplot(x=emp_1['Own account, worker, employer'])
plt.title('Boxplot of Own Account Worker, Employer')
plt.show()
```





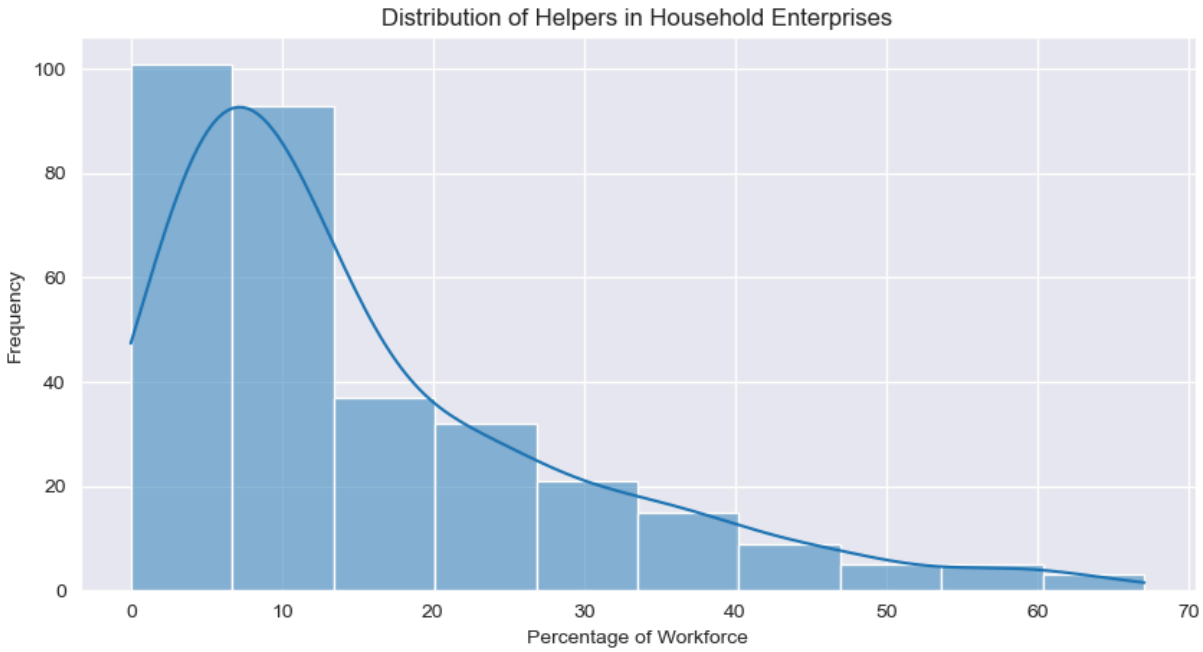
Conclusions:

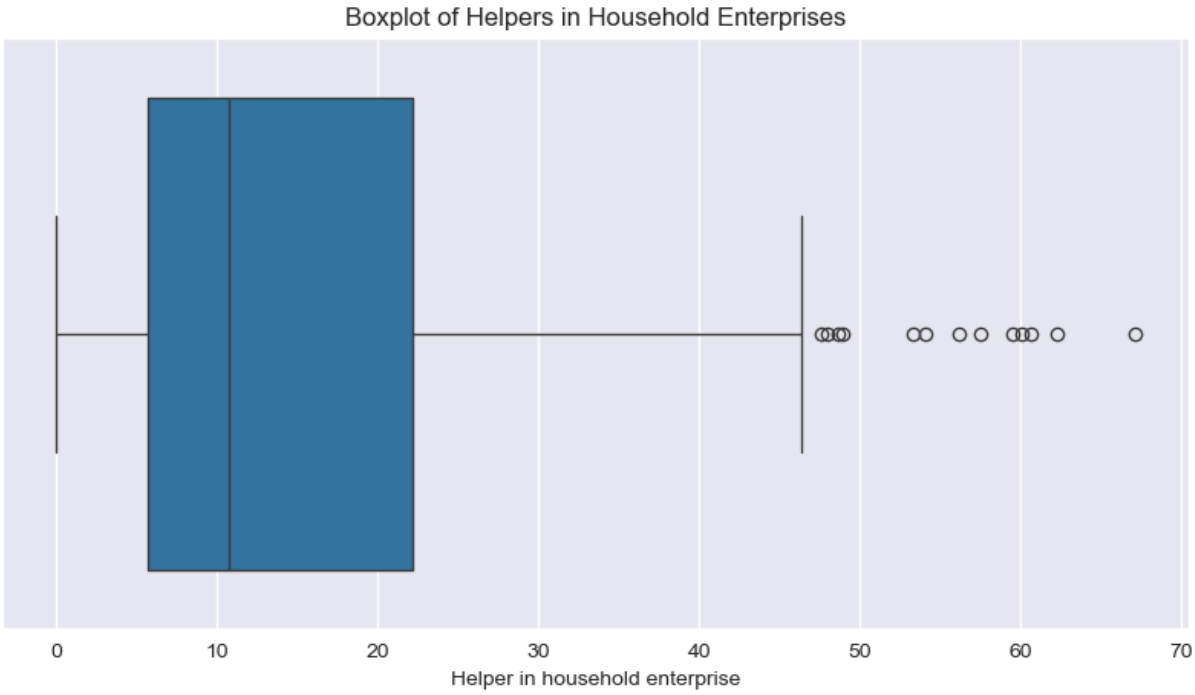
- Histogram:
  - shows normal distribution as there almost 0 skewness in the distribution, this tells us that Own account worker, employer are normally distributed all over india

    - Here percentage peaks at 30-50%
- Boxplot:
    - 2 outliers are present in this data
    - Median value is above 30 and below 40 with a wide IQR range indicating siginificant variablility across states.
    - Outliers exist above 65-67%

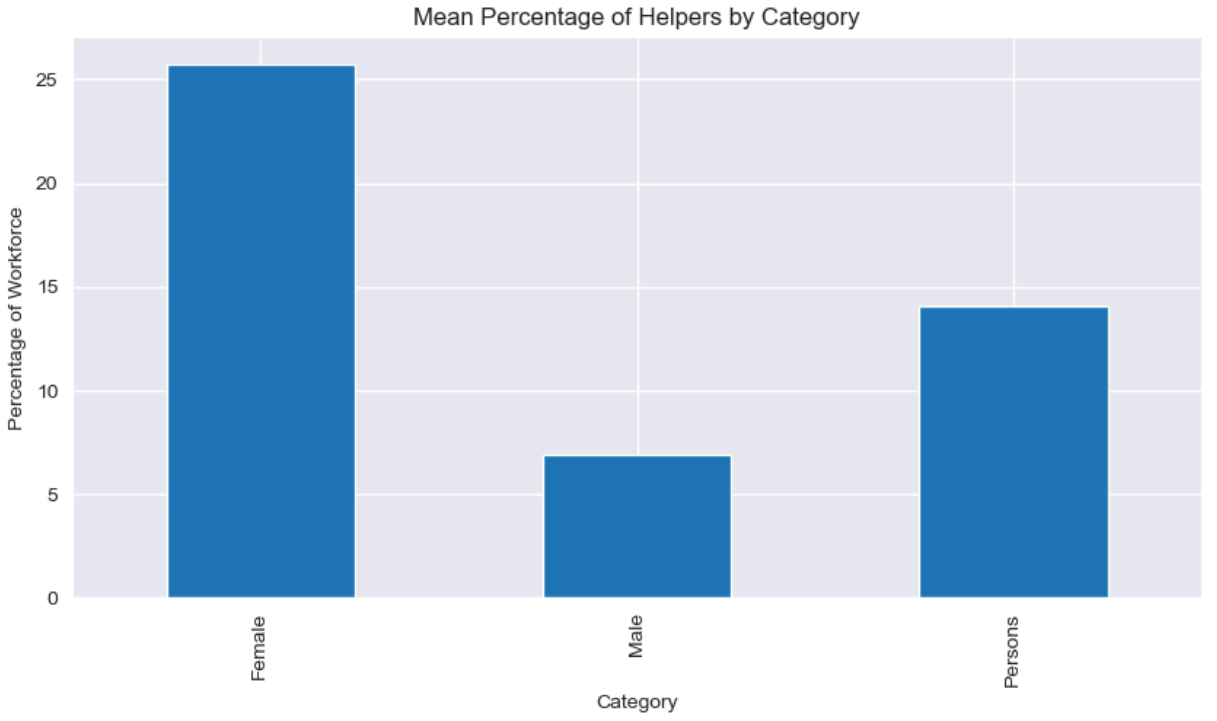
Helper in Household Enterprise

In [65]:
```
plt.figure(figsize=(10,5))
sns.histplot(emp_1['Helper in household enterprise'],kde=True,bins=10)
plt.title('Distribution of Helpers in Household Enterprises')
plt.xlabel('Percentage of Workforce')
plt.ylabel('Frequency')
plt.show()

plt.figure(figsize=(10,5))
sns.boxplot(x=emp_1['Helper in household enterprise'])
plt.title('Boxplot of Helpers in Household Enterprises')
plt.show()
```





In [66]:
```
c = emp_1.groupby('Category')['Helper in household enterprise'].describe()
c.loc[:,'mean'].plot(kind='bar', figsize=(10,5), title='Mean Percentage of H
plt.ylabel('Percentage of Workforce')
plt.show()
```



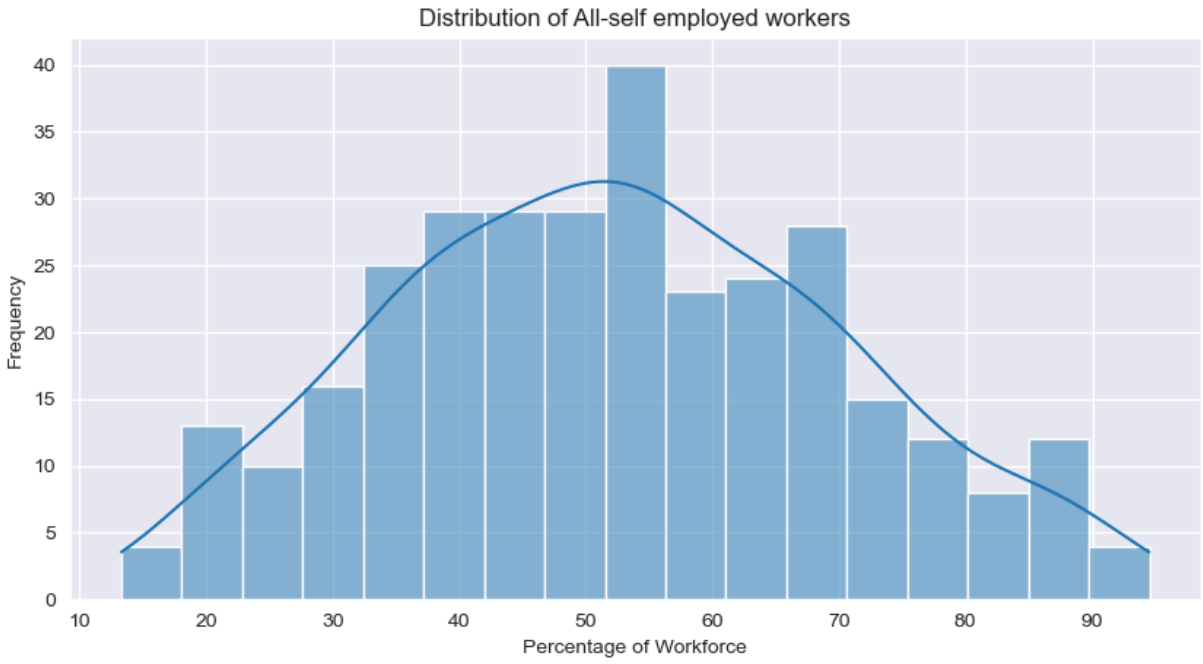
Conclusion:

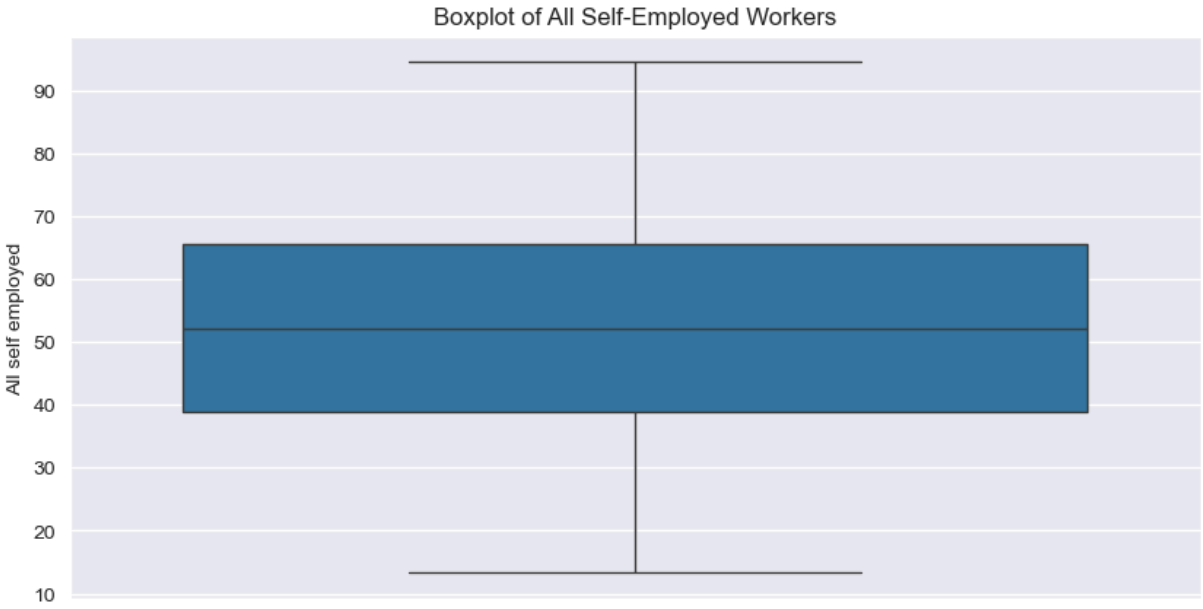
- Histogram & KDE:
  - The distribution is right-skewed(positively), showing that most States have low values of "Helpers in Household Enterprise"

    - The highest frequency is near 0-10% showing most states very few people employed in this type.
    - long tail extends towards 20-70%, meaning few states have some values of Helpers in household enterprise
- Boxplot:
    - median is low
    - There are several outliers beyond 45%
    - IQR is narrow, means that most values are concentrated in the lower range
- After doing groupby, we find that females have the highest mean out of the 3 category hence the outliers

All self employed

In [67]:
```
plt.figure(figsize=(10,5))
n = len(emp_1['All self employed'])
bins = int(np.sqrt(n)) # Square root rule for finding binsize
sns.histplot(emp_1['All self employed'], kde=True, bins=bins)
plt.xlabel('Percentage of Workforce')
plt.ylabel('Frequency')
plt.title('Distribution of All-self employed workers')
plt.show()

plt.figure(figsize=(10,5))
sns.boxplot(emp_1['All self employed'])
plt.title('Boxplot of All Self-Employed Workers')
plt.show()
```

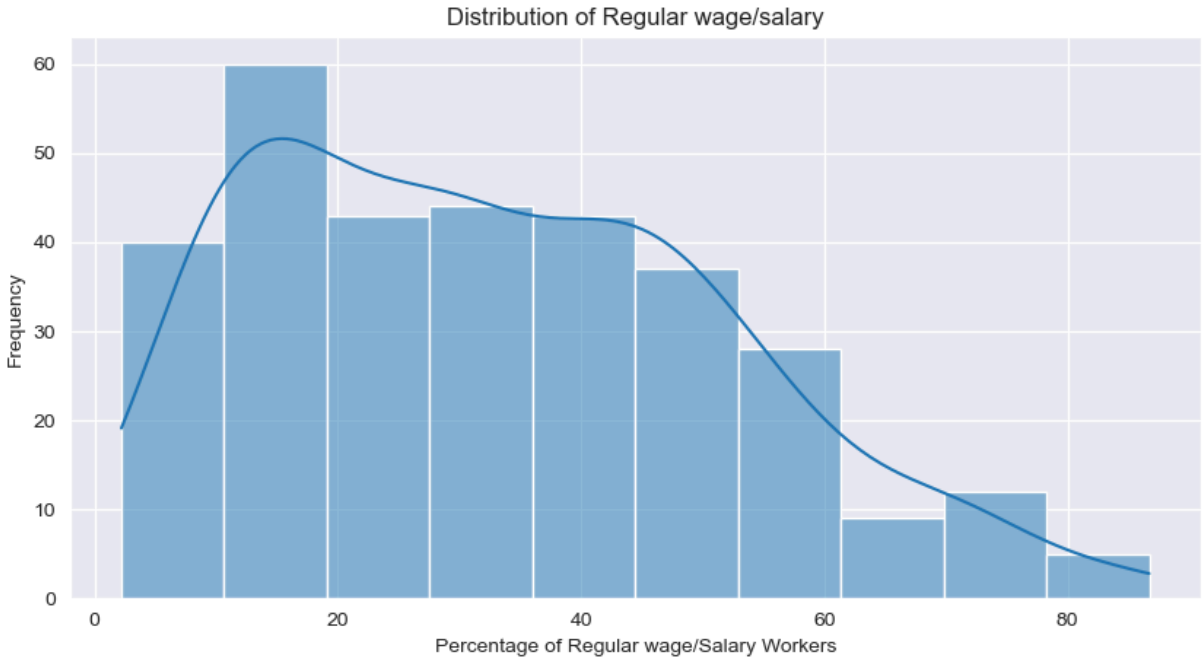




Conclusions

- Distribution is roughly normal, the peak frequency occurs around 50-55% of the workforce
- Most data falls between 30-70% of the workforce
- The relatively symmetric shape indicates most states have similar self-employment percentages with fewer cases at the extreme
- Boxplot indicates:
  - median is around 50-55%
  - IQR range approx between 40-65%
  - min: around 15%
  - max: around 90%

In [68]:
```
plt.figure(figsize=(10,5))
n = len(emp_1['Regular wage/salary'])
bins = int(np.sqrt(n)) # Square root rule for finding binsize
sns.histplot(emp_1['Regular wage/salary'],kde=True, bins=10)
plt.xlabel('Percentage of Regular wage/Salary Workers')
plt.ylabel('Frequency')
plt.title('Distribution of Regular wage/salary')
plt.show()

plt.figure(figsize=(8,4))
sns.boxplot(x=emp_1['Regular wage/salary'])
plt.title('Boxplot of Regular wage/salary')
plt.show()
```



Histogram (Distribution of Regular Wage/Salary)

- Primary Peak: The most prominent feature is the peak in the 10-20% range with a frequency of approximately 60 workers. This indicates that the largest concentration of workers earn between 10-20% of their regular wage/salary.
- Right-Skewed Distribution: The distribution shows a clear right skew, with a long tail extending toward higher percentage values. This means that while most workers are concentrated in the lower percentage ranges, there are progressively fewer workers as the percentage increases.

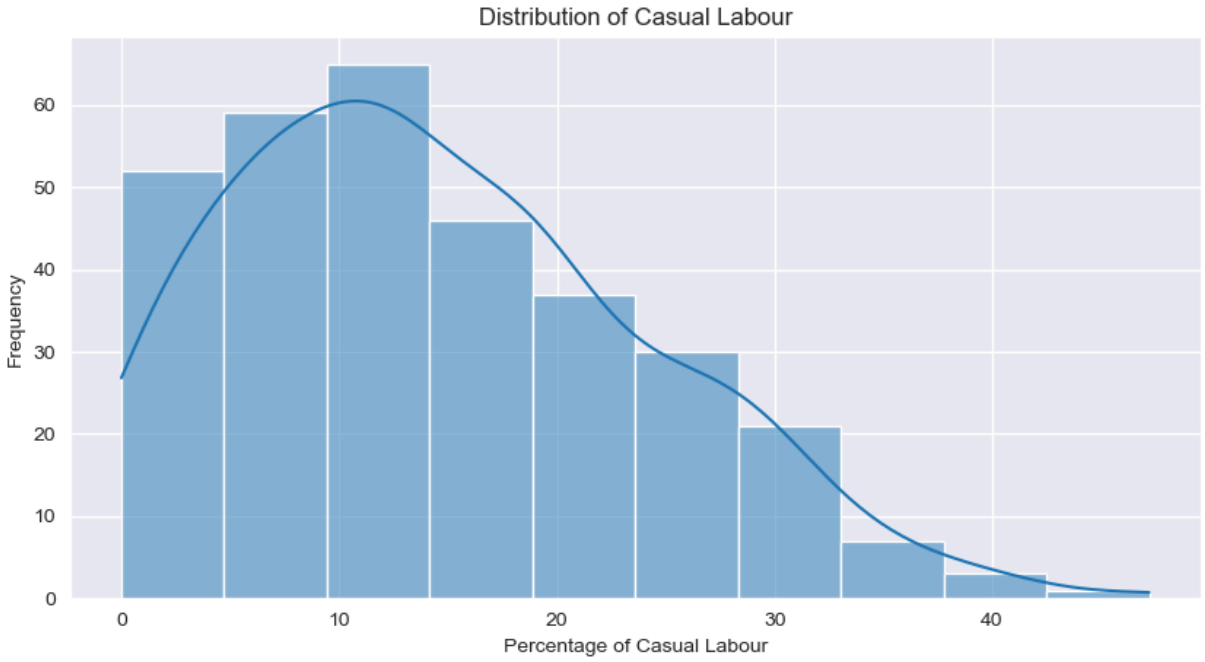
- Declining Frequency Pattern: After the initial peak, there's a steady decline in frequency as the percentage increases. The frequency drops significantly after the 20% mark.
- Low Frequency in Higher Ranges: Very few workers appear in the 60-80% range, showing that high percentages of regular wage/salary are relatively uncommon in this population.

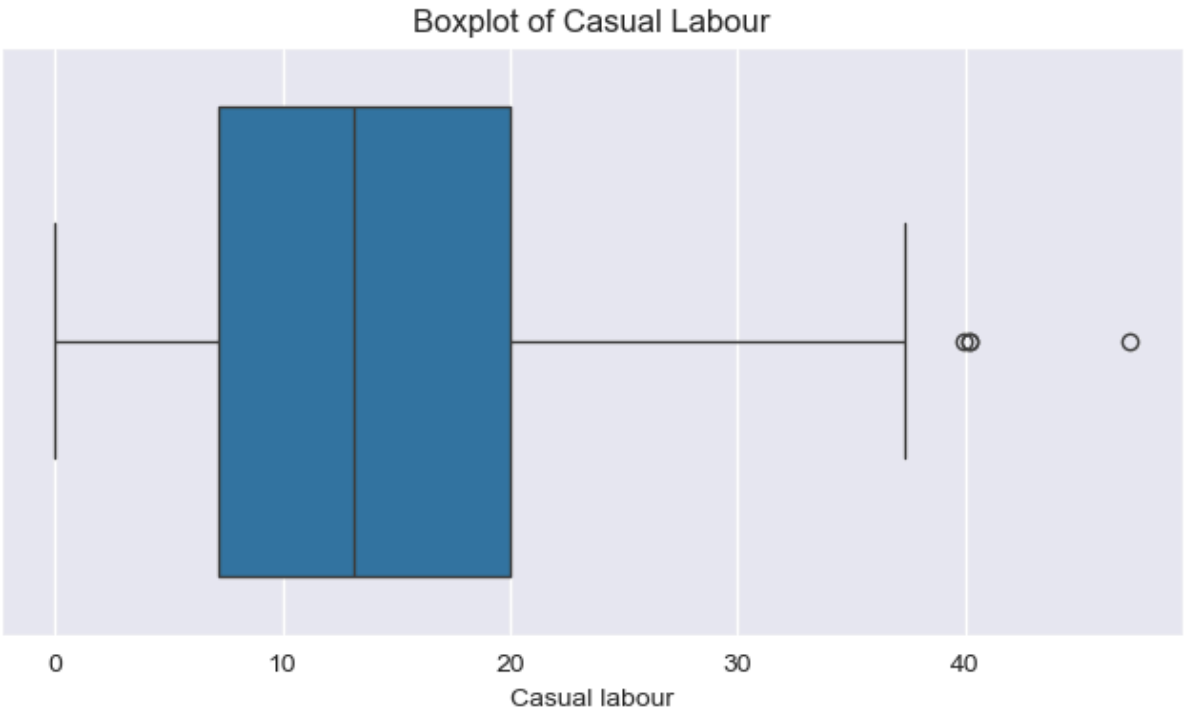
Boxplot (Boxplot of Regular Wage/Salary)

- Median Position: The median (vertical line inside the box) is positioned around 40%, indicating that half of the workers earn less than 40% of their regular wage/salary and half earn more.
- Interquartile Range (IQR): The box spans from approximately 20% to 60%, representing the middle 50% of the data. This 40% spread shows considerable variability in the central portion of the distribution.
- Whisker Extent: The whiskers extend from near 0% to about 80%, showing the full range of the data without outliers. The longer right whisker (from median to upper end) compared to the left whisker reinforces the right-skewed nature of the distribution.
- Data Concentration: The width of the box relative to the whiskers indicates that while the data spans a wide range (0-80%), there's a notable concentration in the 20-60% range.
- Overall Insights Bimodal Tendency: Looking at both graphs together, there might be a slight bimodal tendency, with concentrations around the 10-20% range and another smaller concentration around 40-50%.
- Inequality Indicator: The right-skewed distribution suggests inequality in wage/salary distribution, with many workers at lower percentages and fewer at higher percentages.
- Central Tendency vs. Mode Discrepancy: There's an interesting discrepancy between the mode (10-20% from the histogram) and the median (around 40% from the boxplot), which further emphasizes the skewed nature of the distribution.
- Smooth Transition: The density curve overlaid on the histogram shows a relatively smooth transition, suggesting that while there are peaks, the distribution changes gradually rather than having sharp cutoffs between groups.

These features collectively paint a picture of a wage/salary distribution where most workers earn a relatively small percentage of their regular wage/salary, with progressively fewer workers earning higher percentages.

In [69]:
```
plt.figure(figsize=(10,5))
n = len(emp_1['Casual labour'])
bins = int(np.sqrt(n)) # Square root rule for finding binsize
sns.histplot(emp_1['Casual labour'],kde=True, bins=10)
plt.xlabel('Percentage of Casual Labour')
plt.ylabel('Frequency')
plt.title('Distribution of Casual Labour')
plt.show()

plt.figure(figsize=(8,4))
sns.boxplot(x=emp_1['Casual labour'])
plt.title('Boxplot of Casual Labour')
plt.show()
```





Histogram: "Distribution of Casual Labour"

- Type of Graph: Histogram with a density curve.
- Key Features:
  - The distribution peaks around the 10% range, with the highest frequency of approximately 60 workers.
  - The distribution is right-skewed, with a long tail extending toward higher percentages.
  - Frequencies decrease steadily after the peak, with very few workers in the 30-40% range.

Boxplot: "Boxplot of Casual Labour"

Key Features:

- The median is around 15%.
- The interquartile range (IQR) spans from approximately 15% to 20%.
- There are outliers beyond the upper whisker, around the 40% mark.

Histogram Features:

- Primary Peak: The distribution has a prominent peak at approximately 10% with a frequency of about 60 workers, indicating this is the most common percentage of casual labor.
- Right Skewness: The distribution shows significant right skewness, with a long tail extending toward higher percentages (30-40%).

- Rapid Decline: After the peak, there's a steep decline in frequency, showing that higher percentages of casual labor become increasingly uncommon.
- Range: The distribution primarily spans from 0% to about 40%, with very few data points beyond this range.

Boxplot Features:

- Median: Positioned at approximately 15%, showing the central tendency of casual labor percentage.
- Interquartile Range (IQR): Spans from about 15% to 20%, representing the middle 50% of the data.
- Outliers: Several outliers appear beyond the upper whisker, around the 40% mark, representing unusual cases with higher casual labor percentages.
- Compact Distribution: The relatively narrow IQR suggests consistency in the lower range of casual labor percentages.

In [70]:
```
emp_1.head()
```

Out[70]:

	State/UT	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All	Ca
0	Andhra Pradesh	47.8	6.8	54.6	16.1	29.3	100	
1	Arunachal Pradesh	60.1	12.8	72.9	19.9	7.2	100	
2	Assam	47.9	6.1	54.0	17.2	28.8	100	
3	Bihar	50.4	11.3	61.7	8.4	29.9	100	
4	Chhattisgarh	46.9	20.7	67.6	13.2	19.3	100	

In [71]:
```
emp_1.corr(numeric_only=True)
```

Out[71]:

	Own account, worker, employer	Helper in household enterprise	All self employed	Regular wage/salary	Casual labour	All
Own account, worker, employer	1.000000	-0.088219	0.648131	-0.568142	-0.076185	NaN
Helper in household enterprise	-0.088219	1.000000	0.701378	-0.607635	-0.096818	NaN
All self employed	0.648131	0.701378	1.000000	-0.871175	-0.128369	NaN
Regular wage/salary	-0.568142	-0.607635	-0.871175	1.000000	-0.375074	NaN
Casual labour	-0.076185	-0.096818	-0.128369	-0.375074	1.000000	NaN
All	NaN	NaN	NaN	NaN	NaN	NaN

Further Analysis

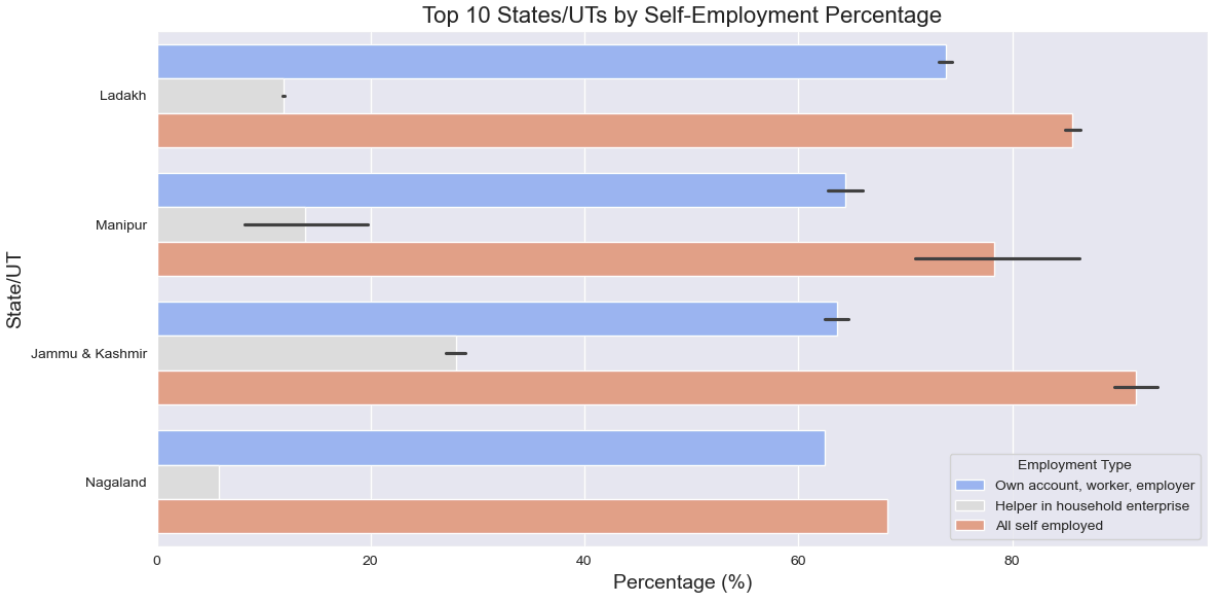
1. Regional Patterns:

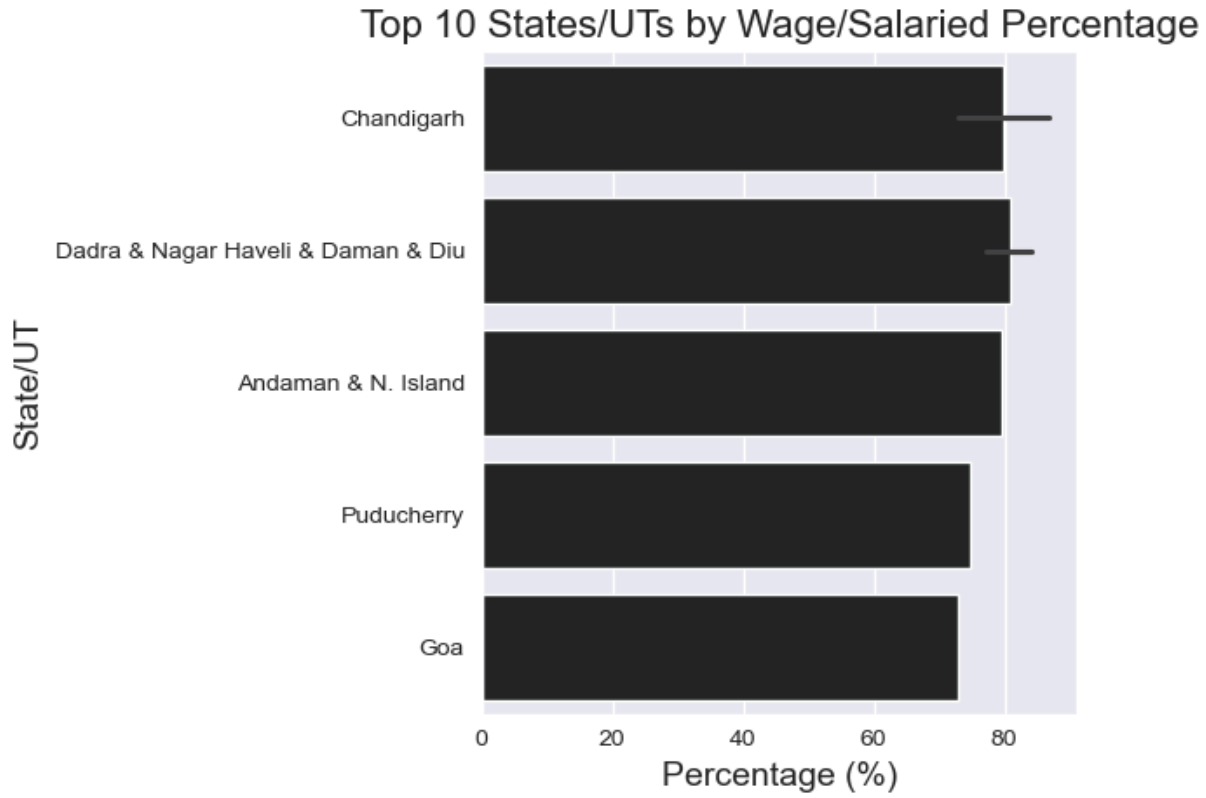
- How does employment status vary by state or union territory?
- Are there noticeable regional clusters or patterns in the percentages of self-employed, wage/salary, and casual labor workers?

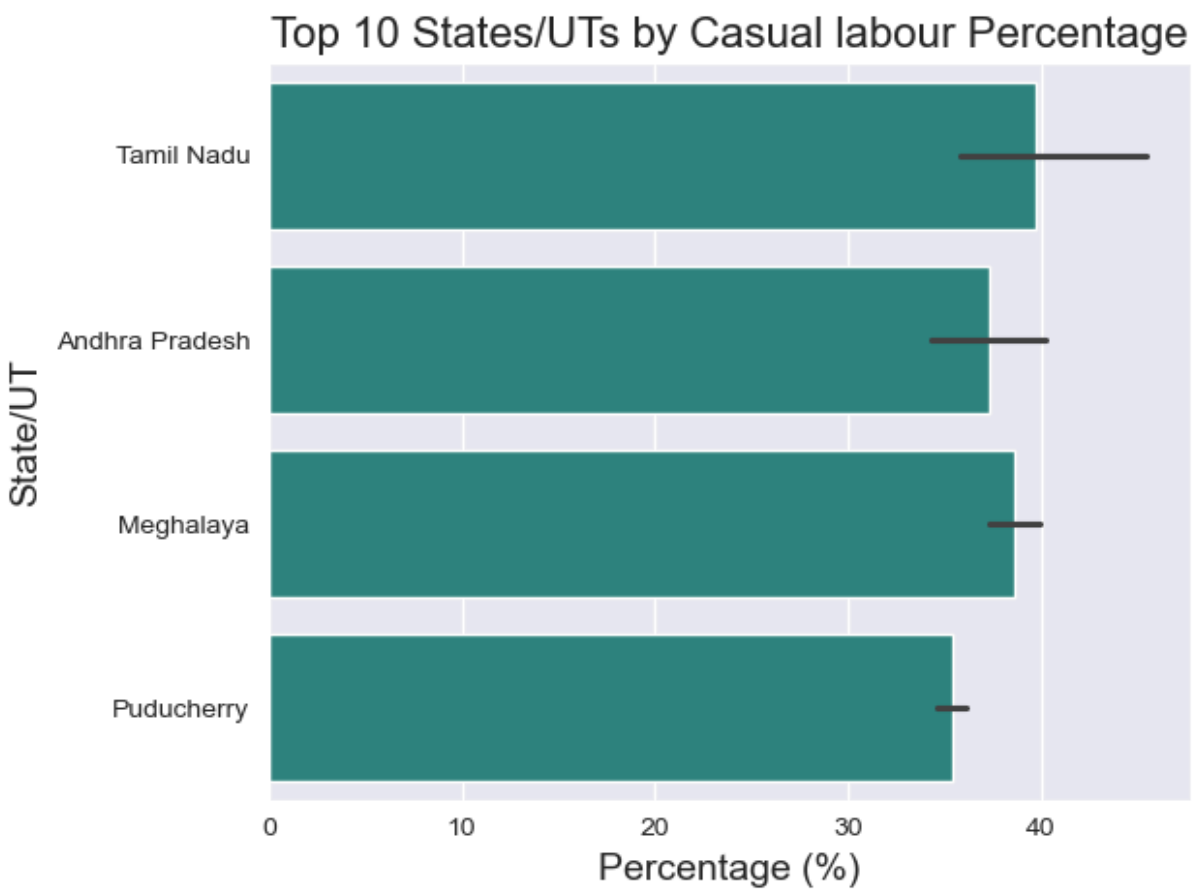
In [72]:
```
cols = ['Own account, worker, employer','Helper in household enterprise','Al
emp_grp = emp_1.groupby('State/UT')[cols].describe()
```

In [73]:
```
# Sorting for Self-Employment
top_self_emp = emp_1.sort_values(
    ['Own account, worker, employer', 'Helper in household enterprise', 'All
    ascending=[False, False, False]
).head(10)

bottom_self_emp = emp_1.sort_values(
    ['Own account, worker, employer', 'Helper in household enterprise', 'All
    ascending=[True, True, True]
).head(10)

# Sorting for Regular Wage/Salary
top_wage_emp = emp_1.sort_values('Regular wage/salary', ascending=False).hea
bottom_wage_emp = emp_1.sort_values('Regular wage/salary', ascending=True).h

# Sorting for Casual Labour
top_casual_emp = emp_1.sort_values('Casual labour', ascending=False).head(10
bottom_casual_emp = emp_1.sort_values('Casual labour', ascending=True).head(
```

```
# **Fixing Visualization**
# Melt the dataframe for plotting
top_self_emp_melted = top_self_emp.melt(id_vars=['State/UT'],
                                        value_vars=['Own account, worker, em
                                                    'Helper in household ent
                                                    'All self employed'],
                                        var_name='Employment Type',
                                        value_name='Percentage')

# Create the bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x='Percentage', y='State/UT', hue='Employment Type', data=top_se

plt.title('Top 10 States/UTs by Self-Employment Percentage', fontsize=16)
plt.xlabel('Percentage (%)', fontsize=14)
plt.ylabel('State/UT', fontsize=14)
plt.legend(title='Employment Type')
plt.tight_layout()
plt.show()

top_wage_emp_melted = top_wage_emp.melt(id_vars=['State/UT'],
                                        value_vars='Regular wage/salary',
                                        var_name = 'Employment Type',
                                        value_name='Percentage')
sns.barplot(x='Percentage',y='State/UT',hue='Employment Type',data=top_wage_
plt.title('Top 10 States/UTs by Wage/Salaried Percentage', fontsize=16)
plt.xlabel('Percentage (%)', fontsize=14)
plt.ylabel('State/UT', fontsize=14)
plt.tight_layout()
plt.show()

top_casual_emp_melted = top_casual_emp.melt(id_vars=['State/UT'],
                                            value_vars='Casual labour',
                                            var_name = 'Employment Type',
                                            value_name='Percentage')
sns.barplot(x='Percentage',y='State/UT',hue='Employment Type', data=top_casu
plt.title('Top 10 States/UTs by Casual labour Percentage', fontsize=16)
plt.xlabel('Percentage (%)', fontsize=14)
plt.ylabel('State/UT', fontsize=14)
plt.tight_layout()
plt.show()
```


Top 10 States/UTs by Self-Employment Percentage
State/UT
Ladakh
Manipur
Jammu & Kashmir
Nagaland
0
20
40
60
80
Percentage (%)
Employment Type
Own account, worker, employer
Helper in household enterprise
All self employed

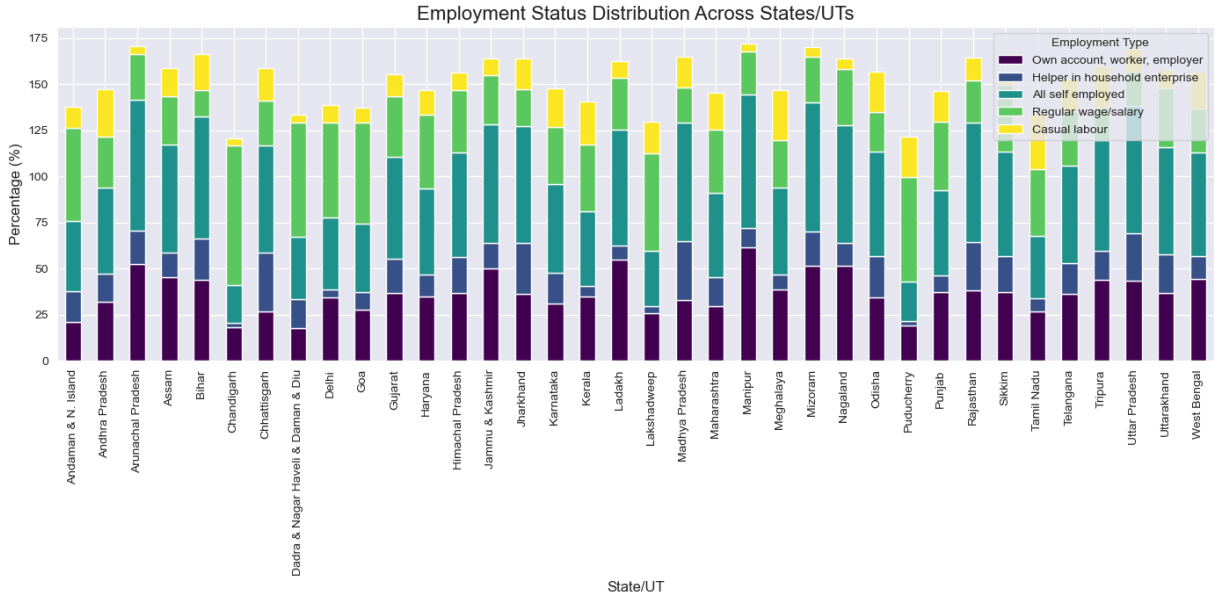

Top 10 States/UTs by Wage/Salaried Percentage
State/UT
Chandigarh
Dadra & Nagar Haveli & Daman & Diu
Andaman & N. Island
Puducherry
Goa
0
20
40
60
80
Percentage (%)



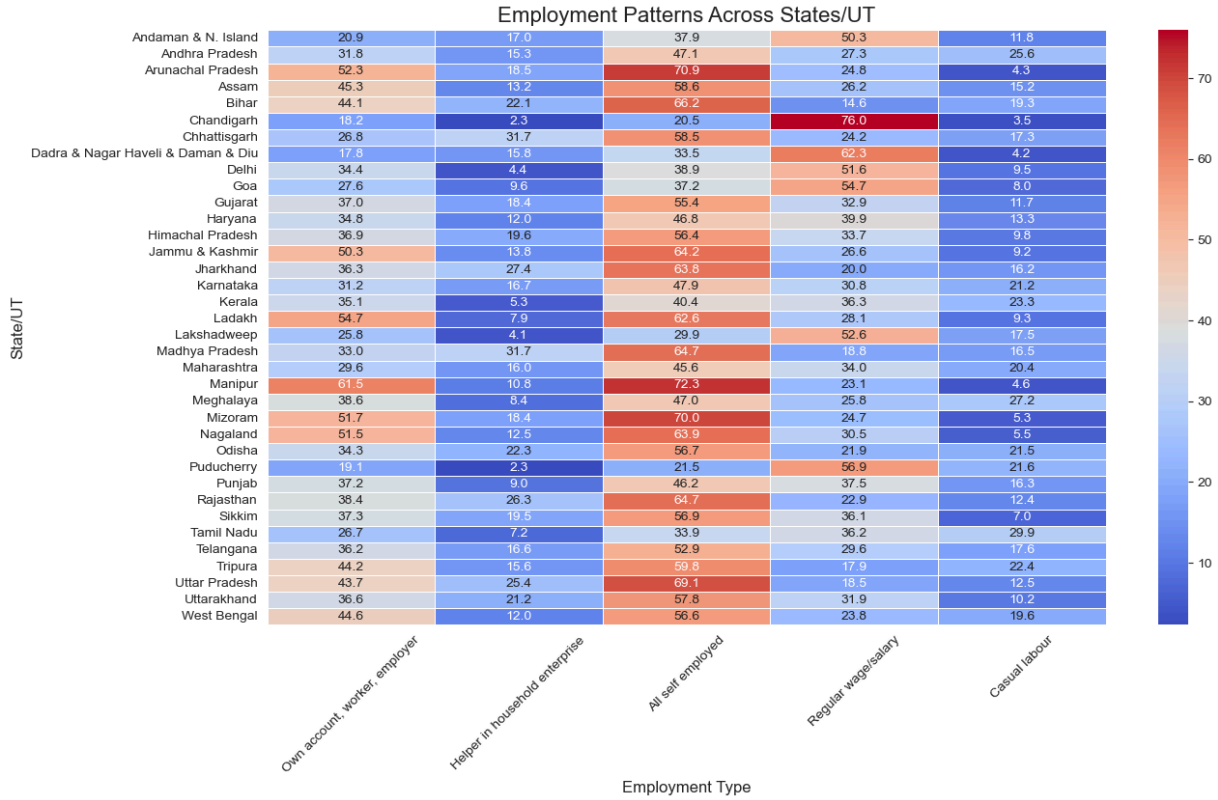
In [74]:
```
# Grouping by state to analyze employment trends
emp_statewise = emp_1.groupby('State/UT')[['Own account, worker, employer',
                                           'All self employed', 'Regular wag
```

In [75]:
```
# Plotting the employment distribution across states
plt.figure(figsize=(14,7))

# Stacked bar chart
emp_statewise.plot(kind='bar', stacked=True, colormap='viridis', figsize=(14

plt.title('Employment Status Distribution Across States/UTs', fontsize=16)
plt.xlabel('State/UT', fontsize=12)
plt.ylabel('Percentage (%)', fontsize=12)
plt.legend(title="Employment Type")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

```
<Figure size 1400x700 with 0 Axes>
```



In [76]:
```
plt.figure(figsize=(14,8))
sns.heatmap(emp_statewise,cmap='coolwarm',annot=True,fmt='.1f',linewidth=0.5
plt.title('Employment Patterns Across States/UT',fontsize=16)
plt.xlabel('Employment Type',fontsize=12)
plt.ylabel('State/UT',fontsize=12)
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.show()
```



Gender Based Analysis

- Splitting by the category column:
  - Compare how employment patterns differ among the gender categories.
- Self-Employment Rates by Gender:
  - Examine and compare the self employment percentages between male and female groups.

In [77]:
```
gen = emp_1.groupby('Category')[cols].mean()
```

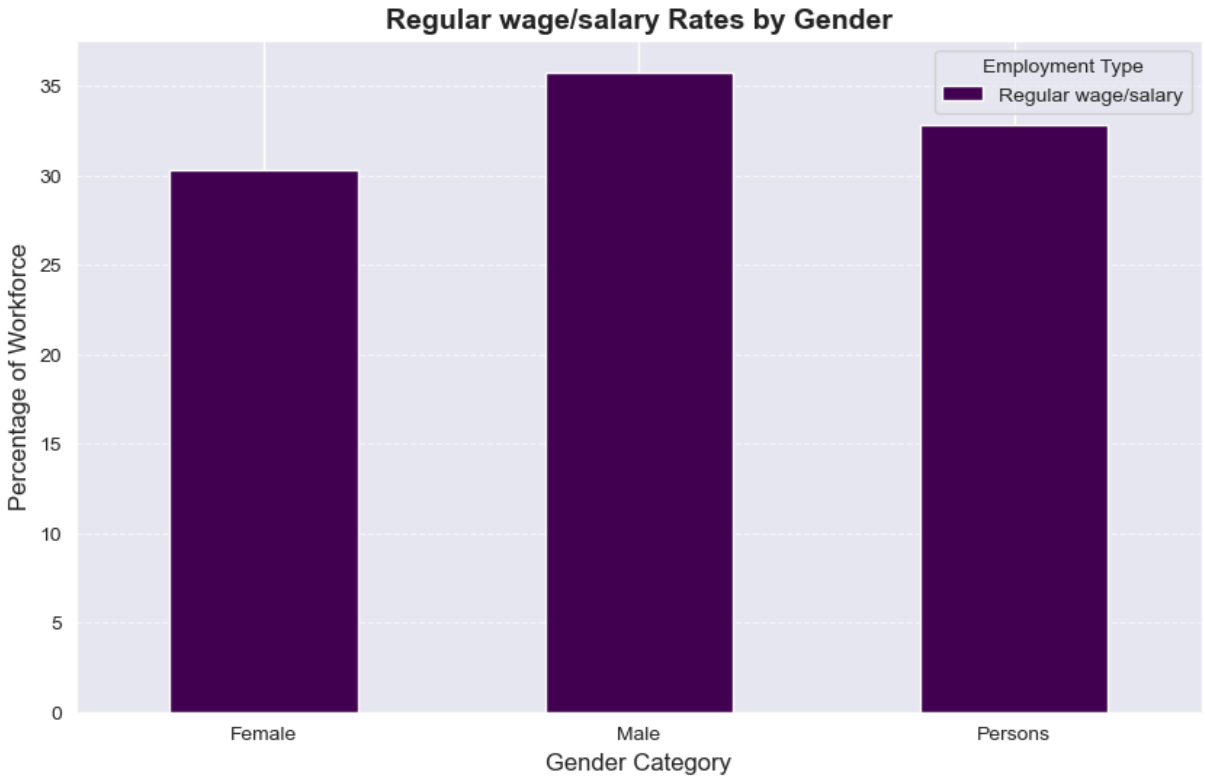
In [ ]:
```

```

In [78]:
```
plt.figure(figsize=(10,6))
gen[['Own account, worker, employer', 'Helper in household enterprise', 'All
plt.title('Self-Employment Rates by Gender', fontsize=14, fontweight='bold')
plt.xlabel('Gender Category', fontsize=12)
plt.ylabel('Percentage of Workforce', fontsize=12)
plt.legend(title="Employment Type")
plt.xticks(rotation=0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```

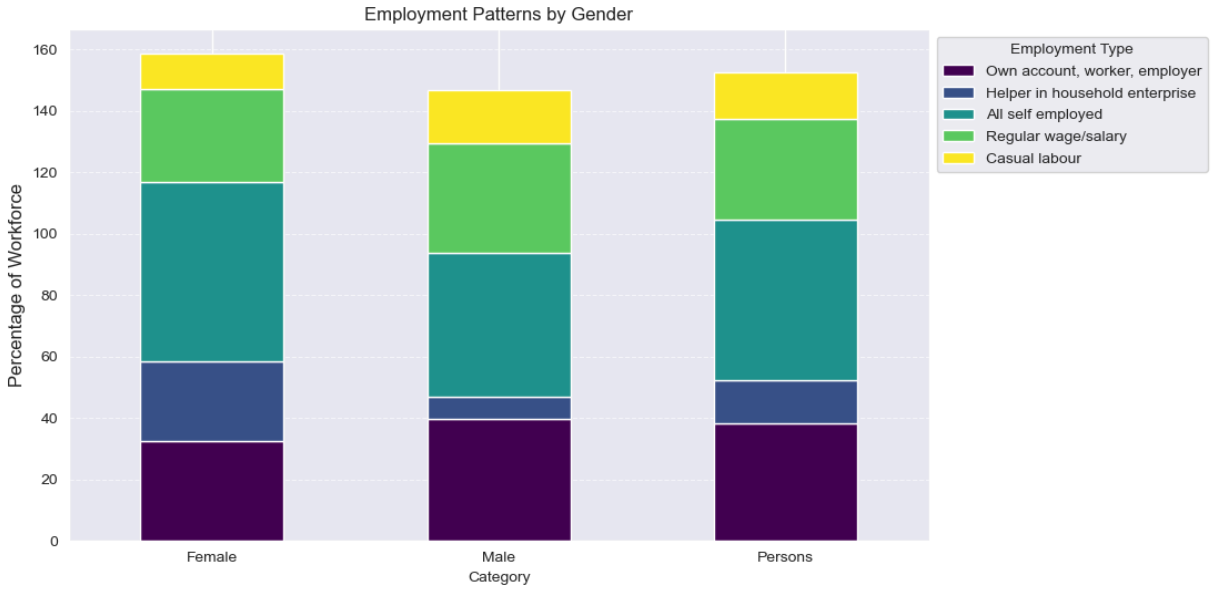
In [79]:
```
plt.figure(figsize=(10,6))
gen['Regular wage/salary'].plot(kind='bar', colormap='viridis', figsize=(10,
plt.title('Regular wage/salary Rates by Gender', fontsize=14, fontweight='bo
plt.xlabel('Gender Category', fontsize=12)
plt.ylabel('Percentage of Workforce', fontsize=12)
plt.legend(title="Employment Type")
plt.xticks(rotation=0)
```

```
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



In [80]:
```
plt.figure(figsize=(10,6))
gen['Casual labour'].plot(kind='bar', colormap='viridis', figsize=(10,6))
plt.title('Casual Labour Rates by Gender', fontsize=14, fontweight='bold')
plt.xlabel('Gender Category', fontsize=12)
plt.ylabel('Percentage of Workforce', fontsize=12)
plt.legend(title="Employment Type")
plt.xticks(rotation=0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

In [81]:
```
gen.plot(kind='bar',stacked=True,colormap='viridis',figsize=(10,6))
plt.title('Employment Patterns by Gender')
plt.ylabel('Percentage of Workforce',fontsize=12)
plt.legend(title='Employment Type', loc='upper left', bbox_to_anchor=(1,1))
plt.xticks(rotation=0)
plt.grid(axis='y',linestyle='--',alpha=0.7)
plt.show()
```



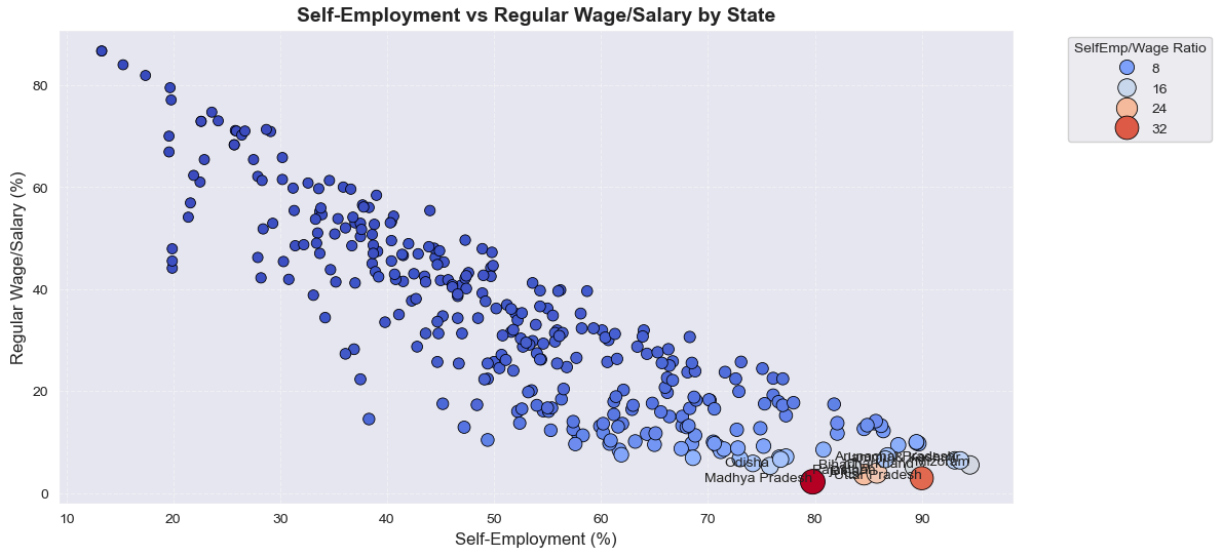
Sectoral Comparisions

- Highest Self-Employed States versus Wage/Salary Workers:

    - Identify states where self-employment percentage is highest relative to regular wage/salary workers.
- Correlations Between Employment Types:
    - Explore if there's a statistical correlation between different categories.

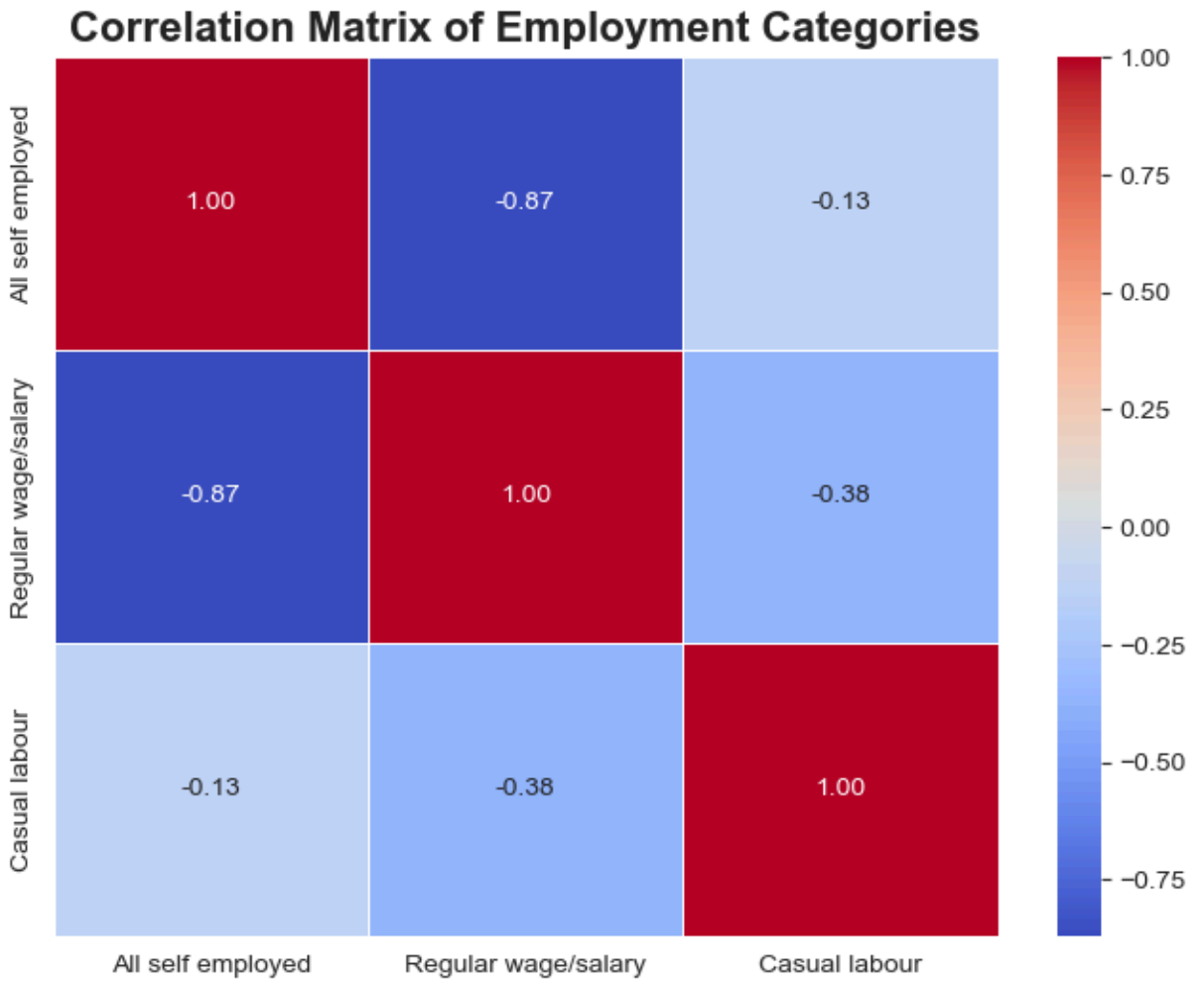
In [82]:
```
emp_1['SelfEmp_Wage_Ratio'] = emp_1['All self employed']/emp_1['Regular wage

top_self_emp_states = emp_1.sort_values('SelfEmp_Wage_Ratio',ascending=False
bottom_self_emp_states = emp_1.sort_values('SelfEmp_Wage_Ratio').head(10)

plt.figure(figsize=(12,6))
sns.scatterplot(data=emp_1,x='All self employed',y='Regular wage/salary',siz

for i, row in top_self_emp_states.iterrows():
    plt.text(row['All self employed'], row['Regular wage/salary'], row['Stat

plt.title('Self-Employment vs Regular Wage/Salary by State', fontsize=14, fo
plt.xlabel('Self-Employment (%)', fontsize=12)
plt.ylabel('Regular Wage/Salary (%)', fontsize=12)
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend(title="SelfEmp/Wage Ratio", bbox_to_anchor=(1.05, 1), loc='upper

plt.show()
```

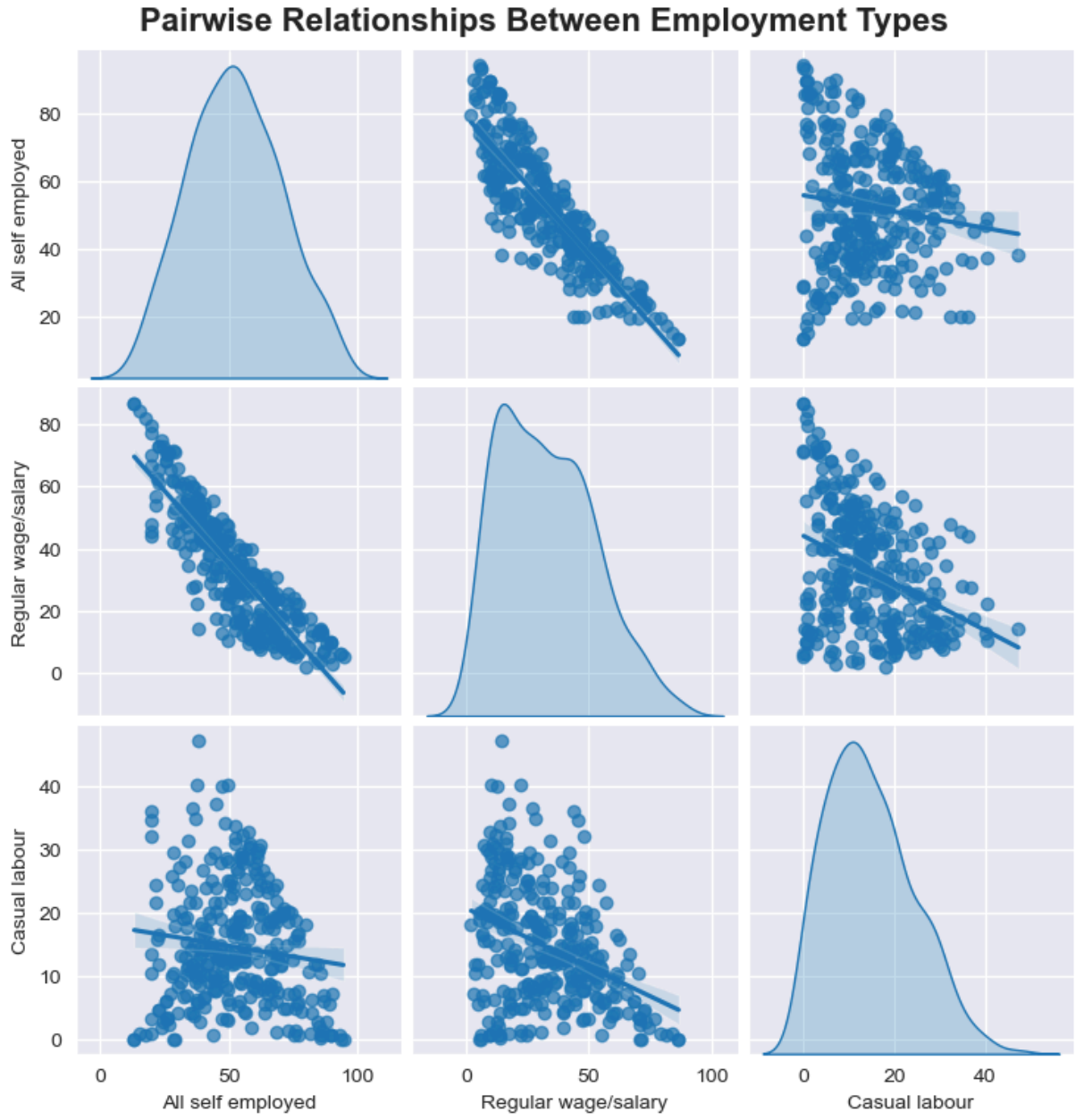
```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\1436415267.py:1: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  emp_1['SelfEmp_Wage_Ratio'] = emp_1['All self employed']/emp_1['Regular wa
ge/salary']
```



In [83]:
```
# Exploring Correlations Between Employment Types
emp_cols = ['All self employed','Regular wage/salary','Casual labour']

corr_matrix = emp_1[emp_cols].corr(method='pearson')
```

```
plt.figure(figsize=(8,6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm',fmt='.2f',linewidths=0.
plt.title('Correlation Matrix of Employment Categories',fontsize=16,fontweig
plt.show()

plt.figure(figsize=(12,5))
for i,col in enumerate(emp_cols[1:]):
    plt.subplot(1,2,i+1)
    sns.regplot(x=emp_1['All self employed'],y=emp_1[col], scatter_kws={'alp
    plt.title(f"Self-Employment vs {col}")
    plt.xlabel('Self-Employment(%)')
    plt.ylabel(f'{col}(%)')

plt.tight_layout()
plt.show()

sns.pairplot(emp_1[emp_cols],kind='reg',diag_kind='kde',plot_kws={'scatter_k
plt.suptitle('Pairwise Relationships Between Employment Types',fontsize=16,
plt.show()
```





Unemployment Rate Dataset (PLFS)

In [84]:
```
unemp_rate = pd.read_csv('Cleaned Data/PLFS/Unemp_Rate.csv',header=None,skip
unemp_rate.head()
```

Out[84]:

	0	1	2	3	4	5	6	7	8	9
0	State/UT	Male	Female	Person	Male	Female	Person	Male	Female	Person
1	Andhra Pradesh	4.8	3	4.1	5.7	7	6.1	5.1	3.8	4.6
2	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8	7.5
3	Assam	4.7	9.6	6.1	7	17.2	9.5	4.9	10.3	6.5
4	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9	3.9

In [85]:
```
col = pd.MultiIndex.from_tuples([
    ('State/UT',''),
    ('Rural','Male'),('Rural','Female'),('Rural','Person'),
    ('Urban','Male'),('Urban','Female'),('Urban','Person'),
    ('Rural+Urban','Male'),('Rural+Urban','Female'),('Rural+Urban','Person')
])
```

In [86]:
```
unemp_rate.columns = col
unemp_rate.head()
```

Out[86]:

	State/UT	Rural			Urban			Rural+Ur		
		Male	Female	Person	Male	Female	Person	Male	Female	Per
0	State/UT	Male	Female	Person	Male	Female	Person	Male	Female	Pe
1	Andhra Pradesh	4.8	3	4.1	5.7	7	6.1	5.1	3.8	
2	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8	
3	Assam	4.7	9.6	6.1	7	17.2	9.5	4.9	10.3	
4	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9	

In [87]:
```
unemp_rate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38 entries, 0 to 37
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   (State/UT, )           38 non-null     object
 1   (Rural, Male)          37 non-null     object
 2   (Rural, Female)        37 non-null     object
 3   (Rural, Person)        37 non-null     object
 4   (Urban, Male)          38 non-null     object
 5   (Urban, Female)        38 non-null     object
 6   (Urban, Person)        38 non-null     object
 7   (Rural+Urban, Male)    38 non-null     object
 8   (Rural+Urban, Female)  38 non-null     object
 9   (Rural+Urban, Person)  38 non-null     object
dtypes: object(10)
memory usage: 3.1+ KB
```

In [88]:
```
unemp_rate
```

Out[88]:

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
0	State/UT	Male	Female	Person	Male	Female	Person	Male	Female
1	Andhra Pradesh	4.8	3	4.1	5.7	7	6.1	5.1	3.8
2	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8
3	Assam	4.7	9.6	6.1	7	17.2	9.5	4.9	10.3
4	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9
5	Chhattisgarh	2	1.8	1.9	6.9	12.3	8.4	2.9	3
6	Delhi	3.6	19.5	6	2.2	1.1	2	2.3	1.5
7	Goa	8	14.1	9.5	3.8	18.6	8.2	5.6	16.8
8	Gujarat	0.7	0.3	0.6	2.4	4.6	2.9	1.4	1.4
9	Haryana	3.9	1.7	3.4	4.2	3.6	4	4	2.4
10	Himachal Pradesh	7.6	12.5	9.9	4.8	20.4	9.7	7.2	13.1
11	Jharkhand	1.5	0.1	1.1	6.9	8.2	7.1	2.5	1
12	Karnataka	2.7	1	2.1	4.3	4.6	4.4	3.3	2
13	Kerala	5.6	17.4	9.7	5.7	15.9	9.2	5.6	16.7
14	Madhya Pradesh	1.1	1.5	1.2	3.4	4.2	3.5	1.6	2.1
15	Maharashtra	3.1	1.2	2.4	5	6.4	5.3	3.9	2.8
16	Manipur	4.8	6.7	5.6	5.8	10.6	7.8	5.1	7.8
17	Meghalaya	4	12.3	7.7	9.2	21.5	14.6	4.7	13.5
18	Mizoram	1.5	1	1.3	3.2	3.9	3.4	2.2	2.4
19	Nagaland	5.7	6.7	6.1	11.6	13.5	12.3	7.3	8.2
20	Odisha	5.1	3.5	4.6	6.9	12.2	8.2	5.4	4.5
21	Punjab	5.1	7.7	5.8	4.8	9.3	5.8	5	8.2
22	Rajasthan	4.2	3.8	4.1	7	13.1	8.5	5	5.4
23	Sikkim	4.6	3.5	4.1	1.8	7	3.1	4	3.8
24	Tamil Nadu	4.5	5.9	5.1	3.6	7.2	4.6	4.1	6.3
25	Telangana	4.8	3.4	4.2	6.4	11.1	7.7	5.5	5.4
26	Tripura	1.6	1.3	1.5	2.7	4.8	3.3	1.8	1.7
27	Uttarakhand	6.1	4.6	5.6	3.6	16.9	6.6	5.4	6.6
28	Uttar Pradesh	3.4	2.8	3.2	5.9	13.4	7.1	3.9	4.5

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
29	West Bengal	3.1	5.1	3.6	3.2	7.6	4.2	3.2	5.9
30	Andaman & N. Island	9.4	21.2	13.5	7.5	33.9	14.9	8.6	25.9
31	Chandigarh	NaN	NaN	NaN	4.4	16.3	7.7	4.4	16.3
32	Dadra & Nagar Haveli & Daman & Diu	5.4	0	3.1	2.1	3.9	2.4	3.3	1.2
33	Jammu & Kashmir	6	17.2	8.8	5.3	35.4	12.8	5.9	20.9
34	Ladakh	2.1	7.8	4.2	5.7	37.2	13.8	2.6	10.4
35	Lakshadweep	12.8	53.6	18.4	12.1	28.6	14.7	12.3	36.2
36	Puducherry	0.3	5.4	2.1	7	9.8	7.9	4.4	7.9
37	all India	3.4	3.7	3.5	4.8	8.7	5.7	3.8	4.9

In [89]:
```
unemp_rate.loc[31,('Rural','Male')] = 0
unemp_rate.loc[31,('Rural','Female')] = 0
unemp_rate.loc[31,('Rural','Person')] = 0
```

In [90]:
```
unemp_rate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38 entries, 0 to 37
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   (State/UT, )           38 non-null     object
 1   (Rural, Male)          38 non-null     object
 2   (Rural, Female)        38 non-null     object
 3   (Rural, Person)        38 non-null     object
 4   (Urban, Male)          38 non-null     object
 5   (Urban, Female)        38 non-null     object
 6   (Urban, Person)        38 non-null     object
 7   (Rural+Urban, Male)    38 non-null     object
 8   (Rural+Urban, Female)  38 non-null     object
 9   (Rural+Urban, Person)  38 non-null     object
dtypes: object(10)
memory usage: 3.1+ KB
```

In [91]:
```
for category in ['Rural','Urban','Rural+Urban']:
    for group in ['Male','Female','Person']:
        unemp_rate[(category,group)] = pd.to_numeric(unemp_rate[(category,gr
```

In [92]:
```
unemp_rate.info()
```

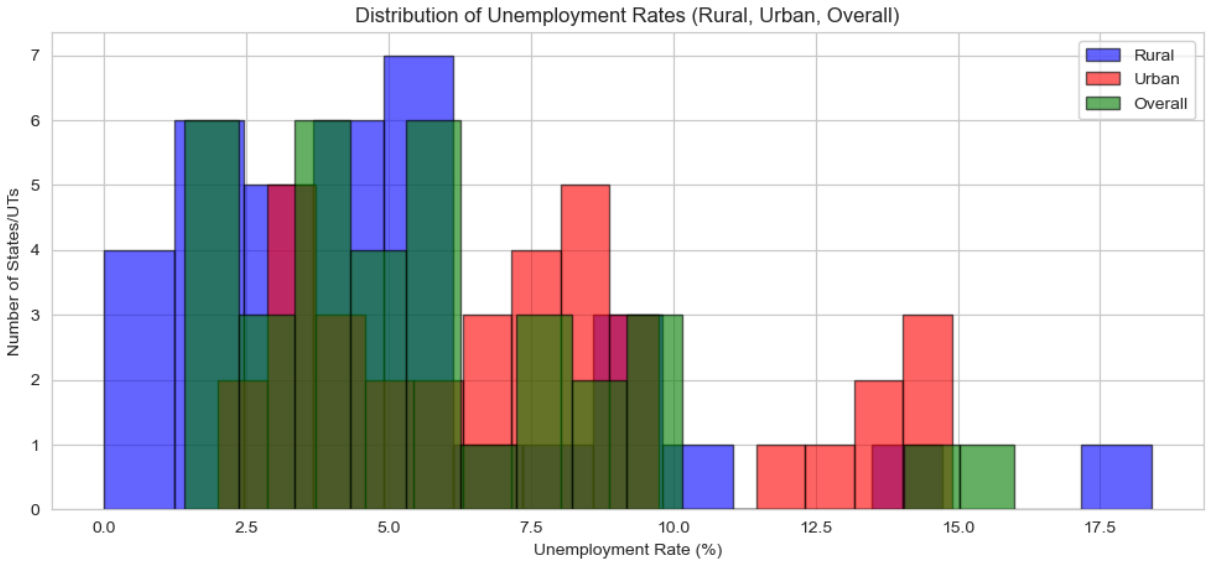
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38 entries, 0 to 37
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   (State/UT, )           38 non-null     object 
 1   (Rural, Male)          37 non-null     float64
 2   (Rural, Female)        37 non-null     float64
 3   (Rural, Person)        37 non-null     float64
 4   (Urban, Male)          37 non-null     float64
 5   (Urban, Female)        37 non-null     float64
 6   (Urban, Person)        37 non-null     float64
 7   (Rural+Urban, Male)    37 non-null     float64
 8   (Rural+Urban, Female)  37 non-null     float64
 9   (Rural+Urban, Person)  37 non-null     float64
dtypes: float64(9), object(1)
memory usage: 3.1+ KB
```

In [93]:
```
unemp_rate_f = unemp_rate[~unemp_rate['State/UT'].str.contains('all India',c
all_ = unemp_rate[unemp_rate['State/UT'].str.contains('all India',case=False
```

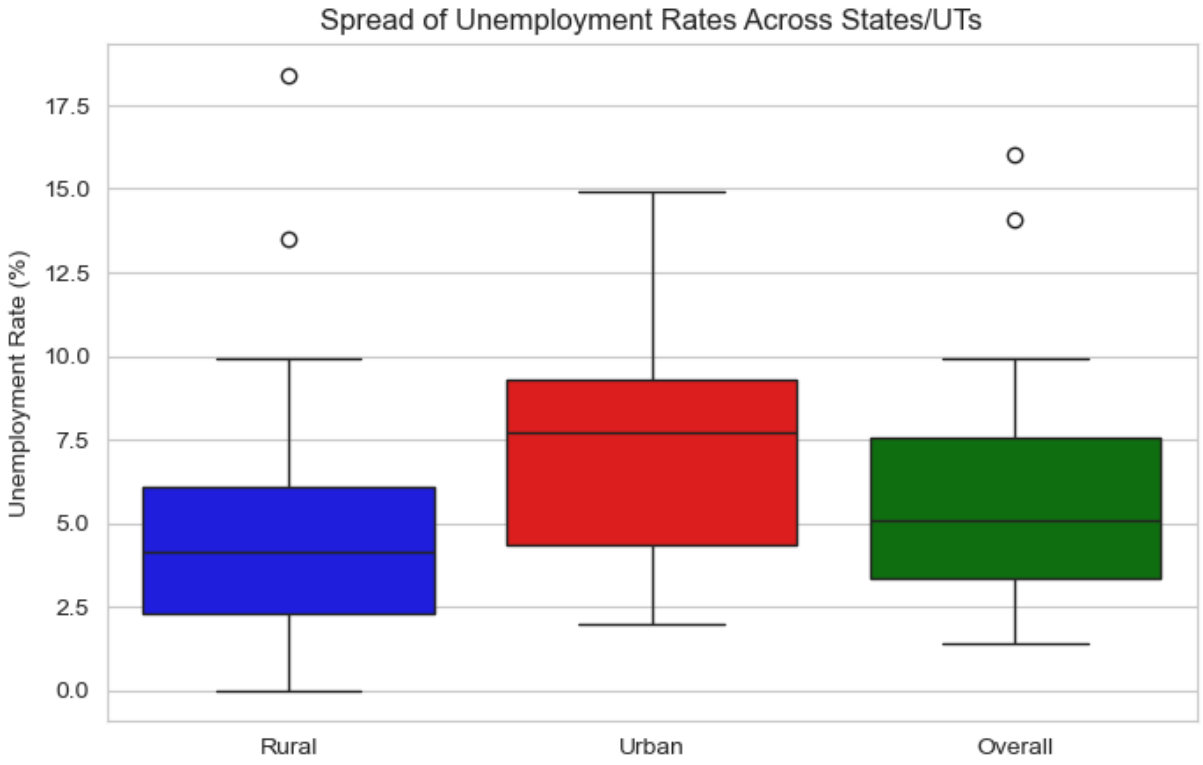
Unemployment Rate Distribution

- Histogram of rural, urban, and overall unemployment rates.
- Boxplots to show the spread of unemployment rates across states.

In [94]:
```
sns.set_style('whitegrid')

plt.figure(figsize=(12, 5))
plt.hist(unemp_rate_f[("Rural", "Person")], bins=15, alpha=0.6, label="Rural
plt.hist(unemp_rate_f[("Urban", "Person")], bins=15, alpha=0.6, label="Urbar
plt.hist(unemp_rate_f[("Rural+Urban", "Person")], bins=15, alpha=0.6, label=

# Labels and Title
plt.xlabel("Unemployment Rate (%)")
plt.ylabel("Number of States/UTs")
plt.title("Distribution of Unemployment Rates (Rural, Urban, Overall)")
plt.legend()
plt.show()
```



In [95]:
```
# --- KDE (Density Plot) ---
plt.figure(figsize=(10, 5))
sns.kdeplot(unemp_rate_f[("Rural", "Person")], fill=True, color="blue", labe
sns.kdeplot(unemp_rate_f[("Urban", "Person")], fill=True, color="red", label
sns.kdeplot(unemp_rate_f[("Rural+Urban", "Person")], fill=True, color="greer

# Labels and Title
plt.xlabel("Unemployment Rate (%)")
plt.ylabel("Density")
plt.title("Distribution of Unemployment Rates (Rural, Urban, Overall)")
plt.legend()
plt.show()

plt.figure(figsize=(8, 5))
sns.boxplot(data=unemp_rate_f[[("Rural", "Person"), ("Urban", "Person"), ("F
            palette=["blue", "red", "green"])

plt.title("Spread of Unemployment Rates Across States/UTs")
plt.ylabel("Unemployment Rate (%)")
plt.xticks(ticks=[0, 1, 2], labels=["Rural", "Urban", "Overall"])
plt.show()
```

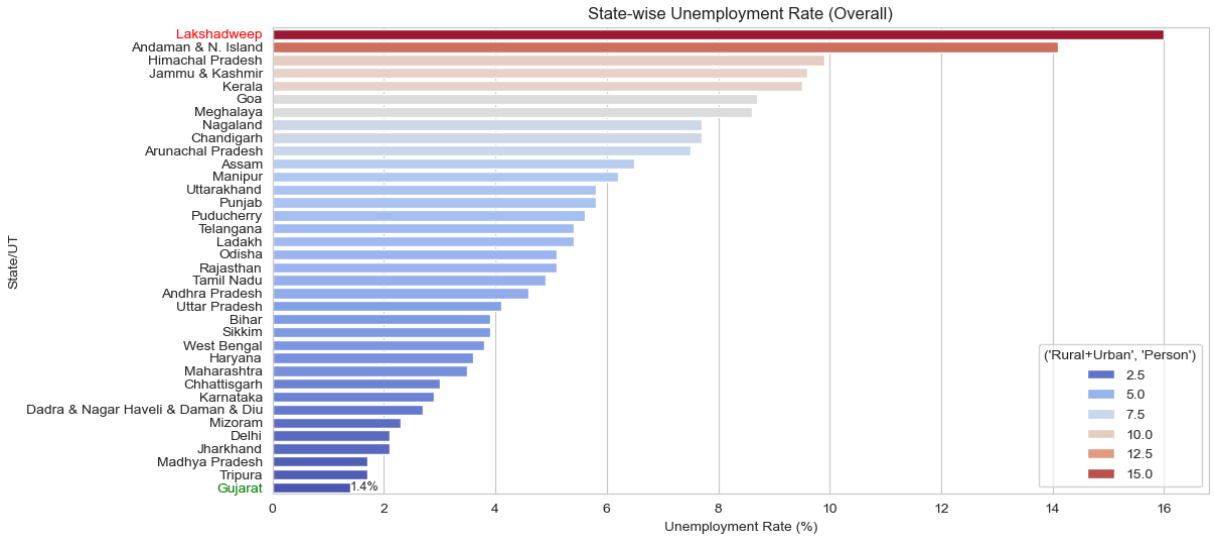


State-wise Comparision

- Bar charts comparing unemployment rates across states
- Highlight the states with the highest and lowest unemployment

In [96]:
```
unemp_rate_sorted = unemp_rate_f.sort_values(by=('Rural+Urban','Person'),asc
unemp_rate_sorted.drop(36,inplace=True)
top_state = unemp_rate_sorted.iloc[0]
bottom_state = unemp_rate_sorted.iloc[-1]
```

```
plt.figure(figsize=(12,6))
barplot = sns.barplot(
    data=unemp_rate_sorted,
    y=('State/UT',''),
    x=('Rural+Urban','Person'),
    hue = ('Rural+Urban','Person'),
    palette='coolwarm'
)

barplot.bar_label(barplot.containers[0], fmt='%0.1f%%',fontsize=9)
barplot.get_yticklabels()[0].set_color('red')
barplot.get_yticklabels()[-1].set_color('green')

plt.xlabel('Unemployment Rate (%)')
plt.ylabel('State/UT')
plt.title('State-wise Unemployment Rate (Overall)')

plt.show()
```

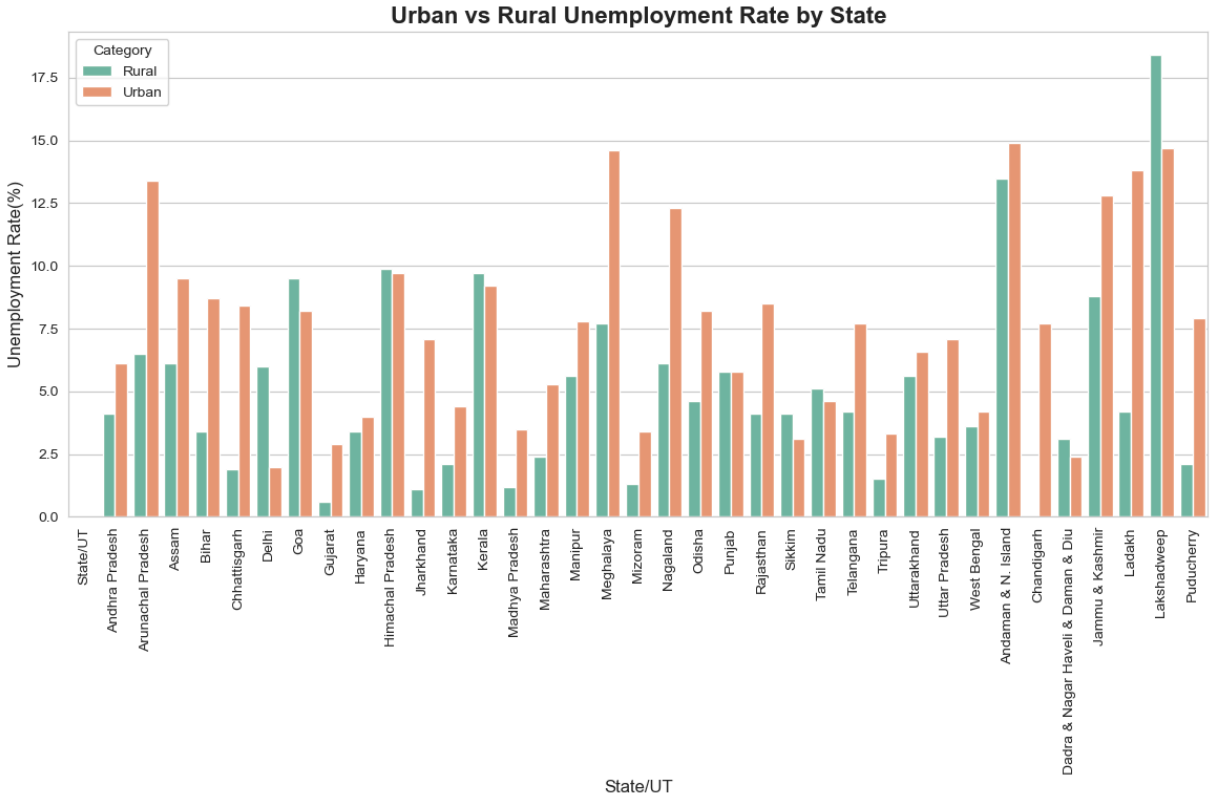


Urban vs Rural Unemployment Trends

- Compare Rural and Urban Unemployment
  - Grouped bar chart or side by side violin plots for rural vs urban unemployment rates
  - Line chart comparing national-level rural and urban unemployment over time (if you have time-series data).
- Ratio of rural to urban unemployment
  - Heatmap or scatter plot showing the rural-to-urban unemployment ratio across states.

In [97]:
```
#Grouped Bar chart
plt.figure(figsize=(14,6))
unemp_melted = unemp_rate_f.melt(id_vars=[('State/UT','')],
                                 value_vars=[('Rural','Person'),('Urban','Person')
                                 var_name='Category',value_name='Unemployment Rate
```

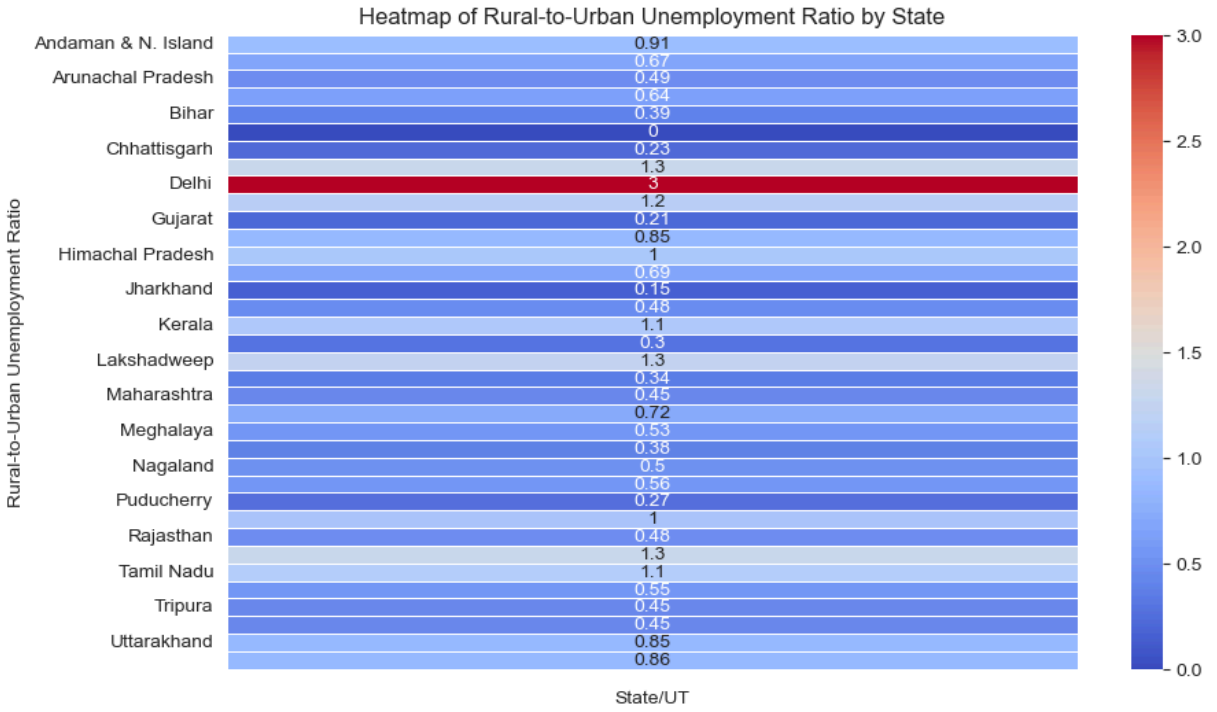
```
sns.barplot(data=unemp_melted,
            x=('State/UT',''),
            y='Unemployment Rate',
            hue='Category',
            palette='Set2')
plt.xticks(rotation=90)
plt.xlabel('State/UT',fontsize=12)
plt.ylabel('Unemployment Rate(%)',fontsize=12)
plt.title('Urban vs Rural Unemployment Rate by State',fontsize=16,fontweight
plt.legend(title='Category')
plt.show()
```



In [98]:
```
#Violin Plot
plt.figure(figsize=(8,6))
sns.violinplot(data=unemp_melted,
               x='Category',
               y='Unemployment Rate',
               palette='muted',
               hue='Category')
plt.xlabel('Category')
plt.ylabel('Unemployment Rate(%)')
plt.title('Distribution of Rural vs. Urban Unemployment Rates')
plt.show()
```

In [99]:
```
# Ensure the column names are correctly accessed
if ('Rural', 'Person') in unemp_rate_f.columns and ('Urban', 'Person') in ur
    unemp_rate_f['Ratio'] = unemp_rate_f[('Rural', 'Person')].astype(float)
else:
    raise KeyError("Column names ('Rural', 'Person') or ('Urban', 'Person')

# Ensure 'State/UT' is used correctly
state_col = 'State/UT' if 'State/UT' in unemp_rate_f.columns else ('State/UT

# Create pivot table
heatmap_data = unemp_rate_f.pivot_table(index=state_col, values='Ratio')

# Plot heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(heatmap_data, cmap='coolwarm', annot=True, linewidths=0.5)

plt.xlabel('State/UT')
plt.ylabel('Rural-to-Urban Unemployment Ratio')
plt.title('Heatmap of Rural-to-Urban Unemployment Ratio by State')

plt.show()
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\609659057.py:3: SettingWit
hCopyWarning:
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
  unemp_rate_f['Ratio'] = unemp_rate_f[('Rural', 'Person')].astype(float) /
unemp_rate_f[('Urban', 'Person')].astype(float)
```

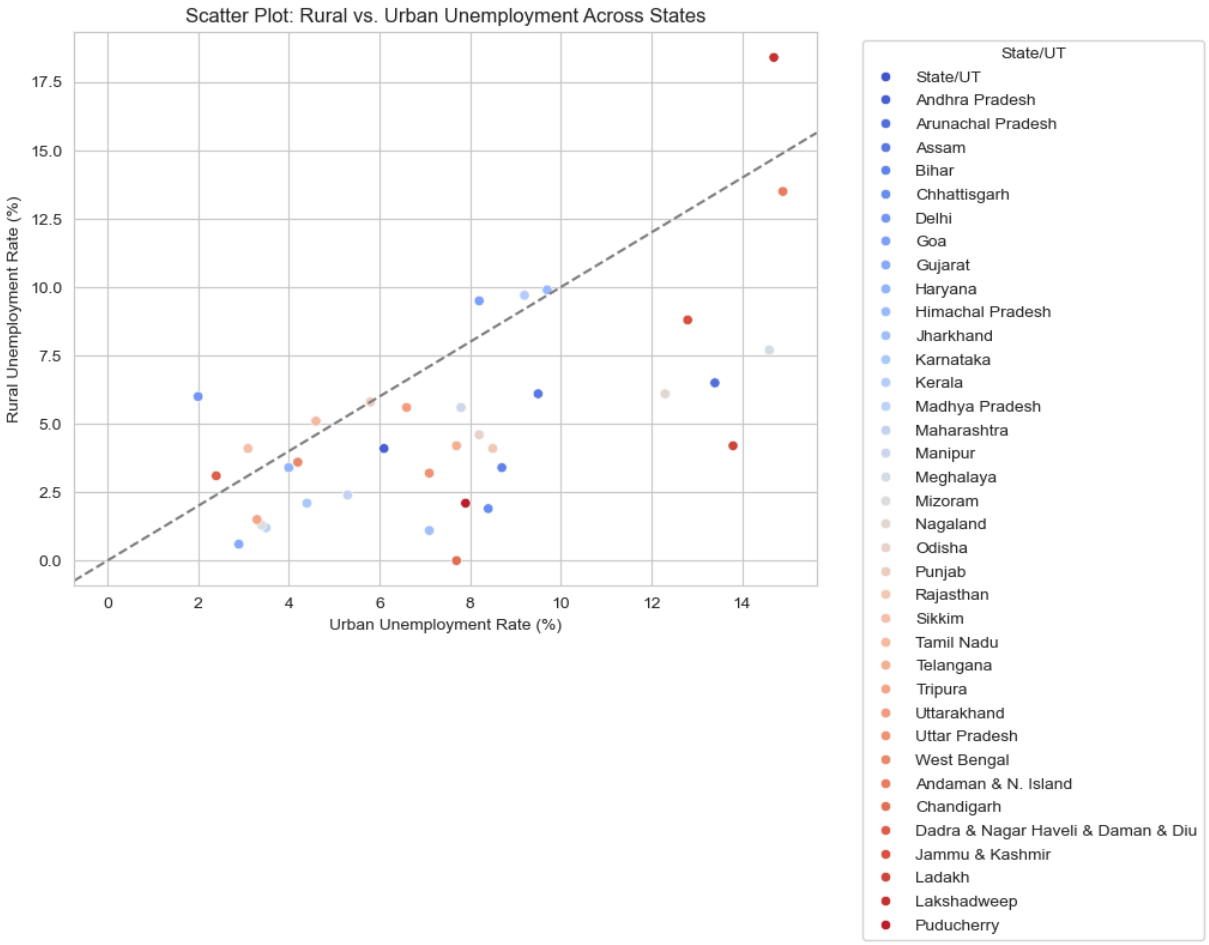


In [100…
```
unemp_rate_f.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 37 entries, 0 to 36
Data columns (total 11 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   (State/UT, )           37 non-null     object
 1   (Rural, Male)          36 non-null     float64
 2   (Rural, Female)        36 non-null     float64
 3   (Rural, Person)        36 non-null     float64
 4   (Urban, Male)          36 non-null     float64
 5   (Urban, Female)        36 non-null     float64
 6   (Urban, Person)        36 non-null     float64
 7   (Rural+Urban, Male)    36 non-null     float64
 8   (Rural+Urban, Female)  36 non-null     float64
 9   (Rural+Urban, Person)  36 non-null     float64
 10  (Ratio, )              36 non-null     float64
dtypes: float64(10), object(1)
memory usage: 3.5+ KB
```

In [101…
```
# Scatter plot
plt.figure(figsize=(8,6))
sns.scatterplot(data=unemp_rate_f,
                x=('Urban','Person'),
```

```
                y=('Rural','Person'),
                hue=('State/UT',''),
                palette='coolwarm')
plt.xlabel("Urban Unemployment Rate (%)")
plt.ylabel("Rural Unemployment Rate (%)")
plt.title("Scatter Plot: Rural vs. Urban Unemployment Across States")
plt.axline((0, 0), slope=1, color="gray", linestyle="--")  # Line y=x for re
plt.legend(title="State/UT", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



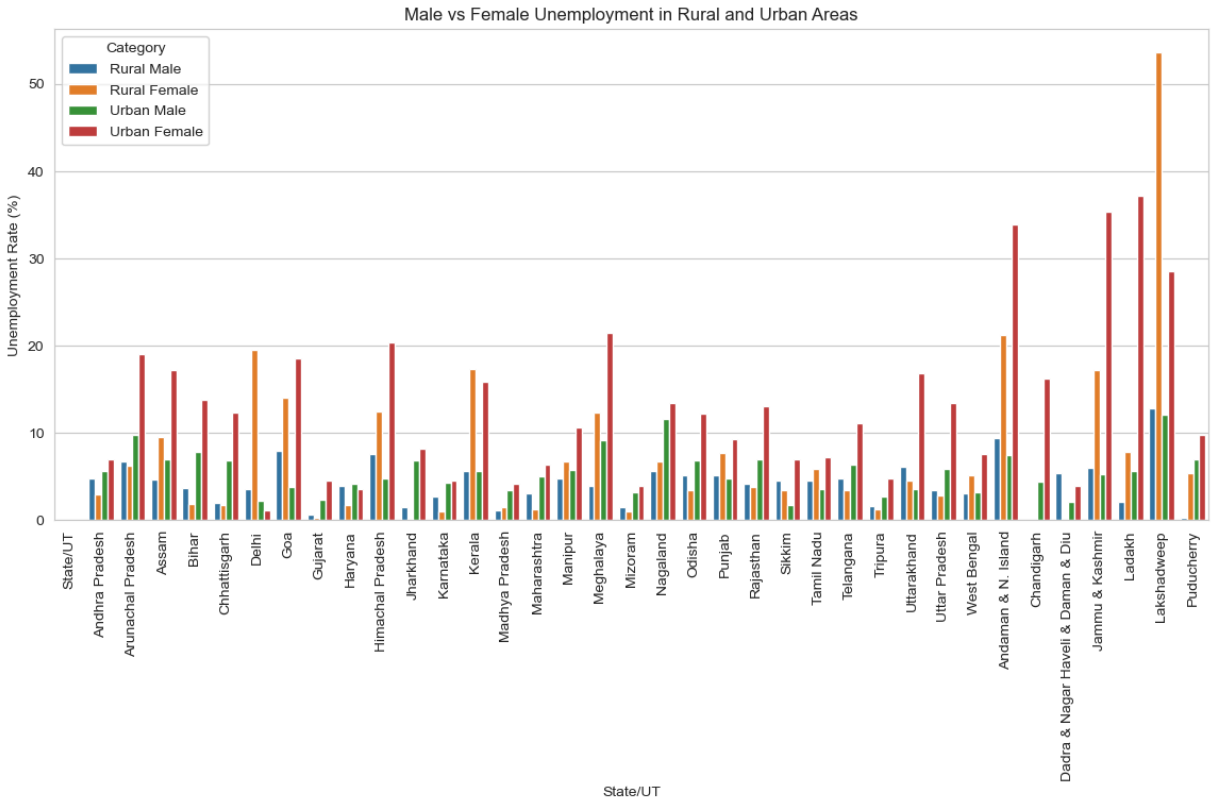
Gender-based Unemployment Trends

- Male vs Female Unemployment
  - Side by side bar chart for male vs female unemployment in rural and urban areas.
  - States where female unemployment is significantly higher than male unemployment.

In [102…

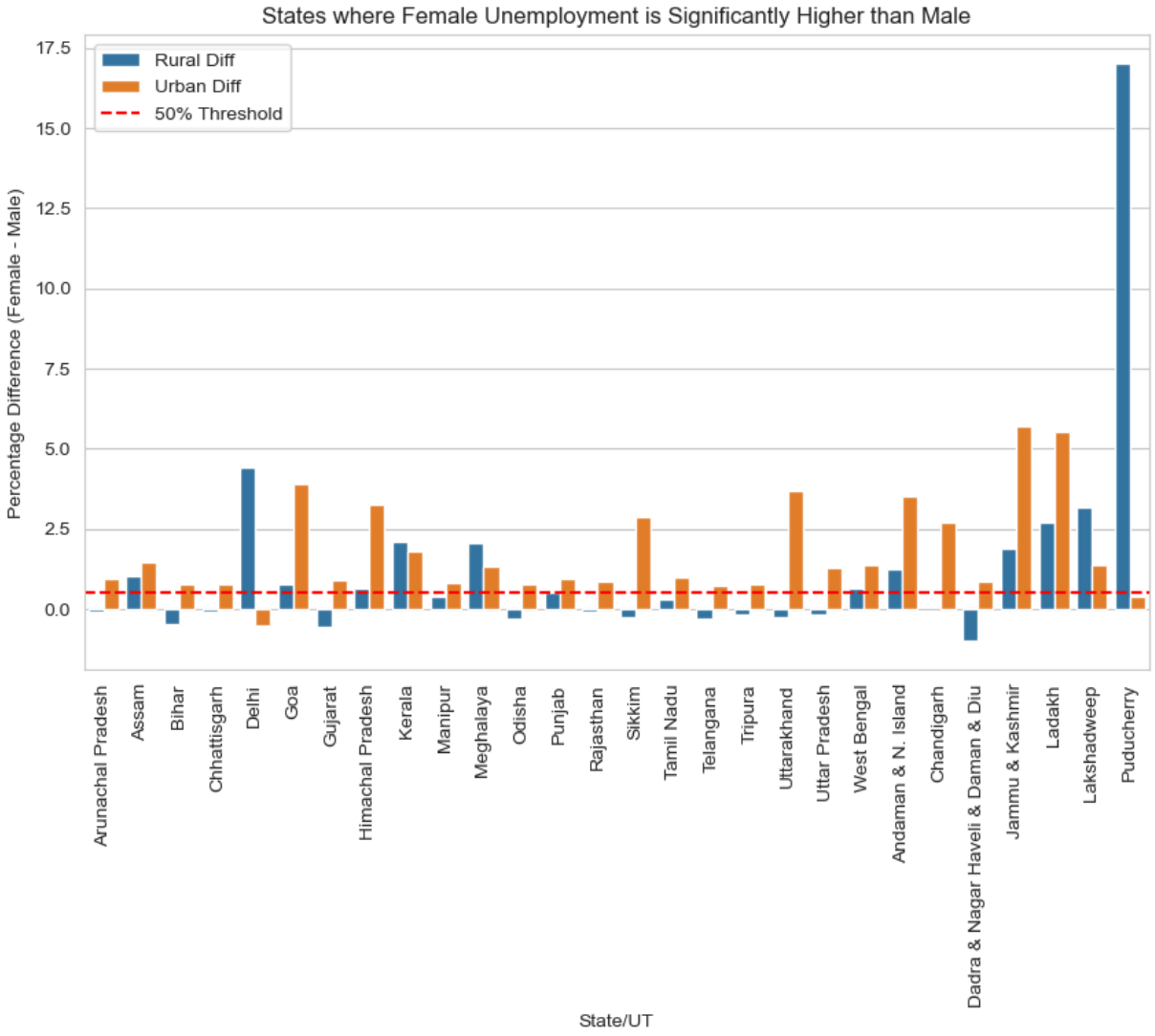
```
gender_unemp = unemp_rate_f[[('State/UT',''),('Rural','Male'),('Rural','Fema
gender_unemp.columns = ['State/UT','Rural Male','Rural Female','Urban Male',

#melt the df
gender_unemp_melted = gender_unemp.melt(id_vars=['State/UT'], var_name='Cate
```

```
plt.figure(figsize=(14,6))
sns.barplot(data=gender_unemp_melted, x='State/UT',y='Unemployment Rate',hue
plt.xticks(rotation=90)
plt.ylabel('Unemployment Rate (%)')
plt.xlabel('State/UT')
plt.title('Male vs Female Unemployment in Rural and Urban Areas')
plt.legend(title='Category')
plt.show()

# Identifying States where female unemployment is significantly higher
gender_unemp['Rural Diff'] = (gender_unemp['Rural Female'] - gender_unemp['F
gender_unemp['Urban Diff'] = (gender_unemp['Urban Female'] - gender_unemp['U

#filter states where female unemployment is significantly higher
threshold = 0.5
high_female_unemp_states = gender_unemp[(gender_unemp['Rural Diff'] > thresh
print("States where Female Unemployment is significantly higher than Male: "
print(high_female_unemp_states[['State/UT','Rural Diff','Urban Diff']])

plt.figure(figsize=(10,6))
sns.barplot(data=high_female_unemp_states.melt(id_vars=['State/UT'], value_v
           x='State/UT',y='Difference',hue='Category')

plt.xticks(rotation=90)
plt.ylabel('Percentage Difference (Female - Male)')
plt.xlabel('State/UT')
plt.title('States where Female Unemployment is Significantly Higher than Mal
plt.axhline(y=0.5, color='r', linestyle='--', label="50% Threshold")
plt.legend()
plt.show()
```



States where Female Unemployment is significantly higher than Male:

	State/UT	Rural Diff	Urban Diff
2	Arunachal Pradesh	-0.074627	0.948980
3	Assam	1.042553	1.457143
4	Bihar	-0.486486	0.769231
5	Chhattisgarh	-0.100000	0.782609
6	Delhi	4.416667	-0.500000
7	Goa	0.762500	3.894737
8	Gujarat	-0.571429	0.916667
10	Himachal Pradesh	0.644737	3.250000
13	Kerala	2.107143	1.789474
16	Manipur	0.395833	0.827586
17	Meghalaya	2.075000	1.336957
20	Odisha	-0.313725	0.768116
21	Punjab	0.509804	0.937500
22	Rajasthan	-0.095238	0.871429
23	Sikkim	-0.239130	2.888889
24	Tamil Nadu	0.311111	1.000000
25	Telangana	-0.291667	0.734375
26	Tripura	-0.187500	0.777778
27	Uttarakhand	-0.245902	3.694444
28	Uttar Pradesh	-0.176471	1.271186
29	West Bengal	0.645161	1.375000
30	Andaman & N. Island	1.255319	3.520000
31	Chandigarh	NaN	2.704545
32	Dadra & Nagar Haveli & Daman & Diu	-1.000000	0.857143
33	Jammu & Kashmir	1.866667	5.679245
34	Ladakh	2.714286	5.526316
35	Lakshadweep	3.187500	1.363636
36	Puducherry	17.000000	0.400000

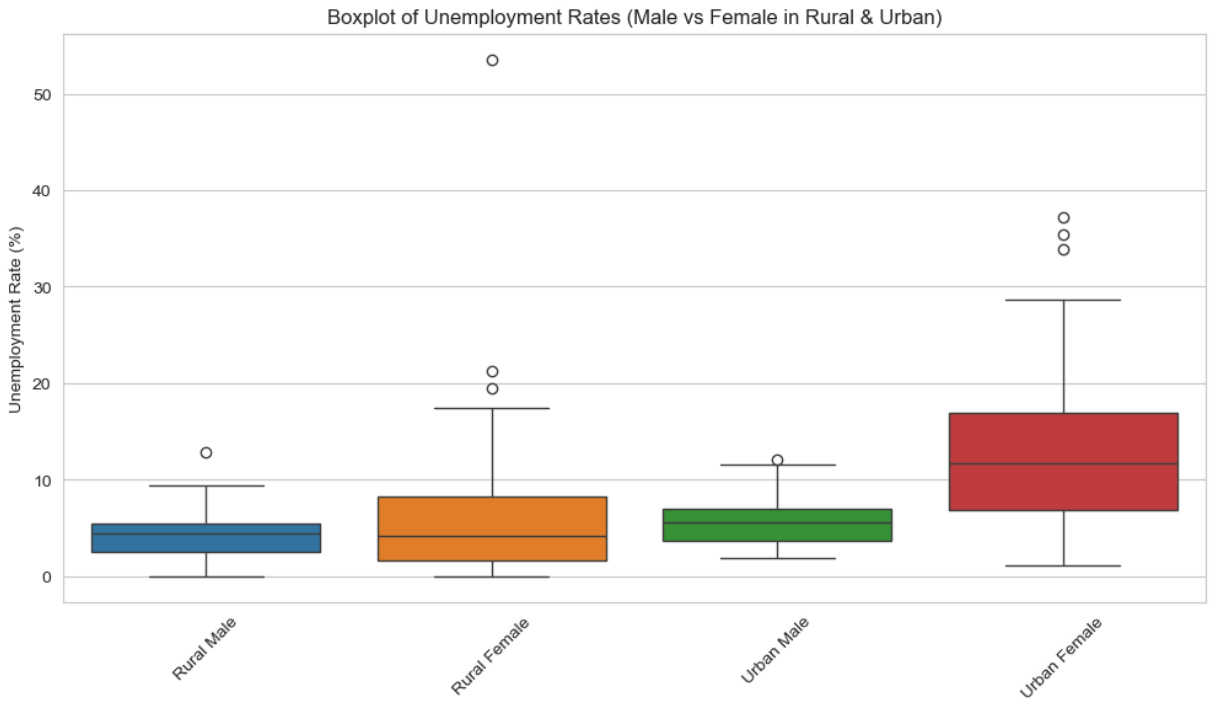


Identify Anomalies & Trends

- Outlier Analysis
  - Use boxplots or scatter plots to detect states with extreme unemployment values

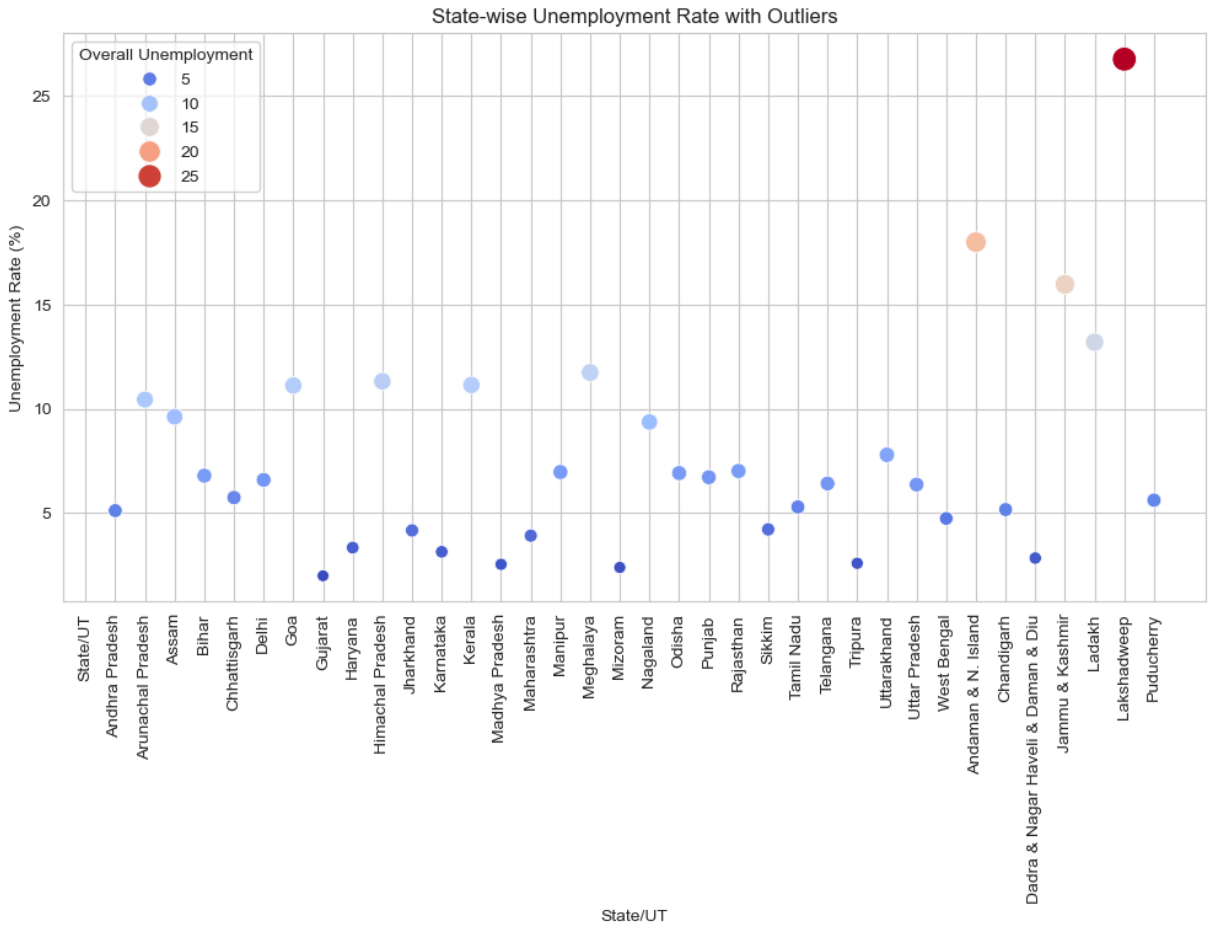
In [103…

```
unemp_data = unemp_rate_f[[('State/UT',''),('Rural','Male'),('Rural','Female
unemp_data.columns = ['State/UT', 'Rural Male', 'Rural Female', 'Urban Male'
unemp_data['Overall Unemployment'] = (unemp_data['Rural Male'] + unemp_data[
                                      unemp_data['Urban Male'] + unemp_data[

# 1️⃣ Outlier Analysis - Boxplots
plt.figure(figsize=(12, 6))
sns.boxplot(data=unemp_data[['Rural Male', 'Rural Female', 'Urban Male', 'Ur
plt.title('Boxplot of Unemployment Rates (Male vs Female in Rural & Urban)')
plt.ylabel('Unemployment Rate (%)')
plt.xticks(rotation=45)
plt.show()
```



In [104…

```
plt.figure(figsize=(12, 6))
sns.scatterplot(data=unemp_data, x='State/UT', y='Overall Unemployment', hue
plt.xticks(rotation=90)
plt.ylabel('Unemployment Rate (%)')
plt.xlabel('State/UT')
plt.title('State-wise Unemployment Rate with Outliers')
plt.show()
```



In [105…

```
Q1 = unemp_data['Overall Unemployment'].quantile(0.25)
Q3 = unemp_data['Overall Unemployment'].quantile(0.75)
IQR = Q3 - Q1
outliers = unemp_data[(unemp_data['Overall Unemployment'] > (Q3 + 1.5 * IQR)
print("States with Extreme Unemployment Rates (Outliers):")
print(outliers[['State/UT', 'Overall Unemployment']])
```

```
States with Extreme Unemployment Rates (Outliers):
       State/UT  Overall Unemployment
35  Lakshadweep                26.775
```

In [106…

```
unemp_rate_f
```

Out[106…

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
0	State/UT	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Andhra Pradesh	4.8	3.0	4.1	5.7	7.0	6.1	5.1	3.8
2	Arunachal Pradesh	6.7	6.2	6.5	9.8	19.1	13.4	7.2	7.8
3	Assam	4.7	9.6	6.1	7.0	17.2	9.5	4.9	10.3
4	Bihar	3.7	1.9	3.4	7.8	13.8	8.7	4.1	2.9
5	Chhattisgarh	2.0	1.8	1.9	6.9	12.3	8.4	2.9	3.0
6	Delhi	3.6	19.5	6.0	2.2	1.1	2.0	2.3	1.5
7	Goa	8.0	14.1	9.5	3.8	18.6	8.2	5.6	16.8
8	Gujarat	0.7	0.3	0.6	2.4	4.6	2.9	1.4	1.4
9	Haryana	3.9	1.7	3.4	4.2	3.6	4.0	4.0	2.4
10	Himachal Pradesh	7.6	12.5	9.9	4.8	20.4	9.7	7.2	13.1
11	Jharkhand	1.5	0.1	1.1	6.9	8.2	7.1	2.5	1.0
12	Karnataka	2.7	1.0	2.1	4.3	4.6	4.4	3.3	2.0
13	Kerala	5.6	17.4	9.7	5.7	15.9	9.2	5.6	16.7
14	Madhya Pradesh	1.1	1.5	1.2	3.4	4.2	3.5	1.6	2.1
15	Maharashtra	3.1	1.2	2.4	5.0	6.4	5.3	3.9	2.8
16	Manipur	4.8	6.7	5.6	5.8	10.6	7.8	5.1	7.8
17	Meghalaya	4.0	12.3	7.7	9.2	21.5	14.6	4.7	13.5
18	Mizoram	1.5	1.0	1.3	3.2	3.9	3.4	2.2	2.4
19	Nagaland	5.7	6.7	6.1	11.6	13.5	12.3	7.3	8.2
20	Odisha	5.1	3.5	4.6	6.9	12.2	8.2	5.4	4.5
21	Punjab	5.1	7.7	5.8	4.8	9.3	5.8	5.0	8.2
22	Rajasthan	4.2	3.8	4.1	7.0	13.1	8.5	5.0	5.4
23	Sikkim	4.6	3.5	4.1	1.8	7.0	3.1	4.0	3.8
24	Tamil Nadu	4.5	5.9	5.1	3.6	7.2	4.6	4.1	6.3
25	Telangana	4.8	3.4	4.2	6.4	11.1	7.7	5.5	5.4
26	Tripura	1.6	1.3	1.5	2.7	4.8	3.3	1.8	1.7
27	Uttarakhand	6.1	4.6	5.6	3.6	16.9	6.6	5.4	6.6
28	Uttar Pradesh	3.4	2.8	3.2	5.9	13.4	7.1	3.9	4.5

	State/UT			Rural			Urban		Rura
		Male	Female	Person	Male	Female	Person	Male	Female
29	West Bengal	3.1	5.1	3.6	3.2	7.6	4.2	3.2	5.9
30	Andaman & N. Island	9.4	21.2	13.5	7.5	33.9	14.9	8.6	25.9
31	Chandigarh	0.0	0.0	0.0	4.4	16.3	7.7	4.4	16.3
32	Dadra & Nagar Haveli & Daman & Diu	5.4	0.0	3.1	2.1	3.9	2.4	3.3	1.2
33	Jammu & Kashmir	6.0	17.2	8.8	5.3	35.4	12.8	5.9	20.9
34	Ladakh	2.1	7.8	4.2	5.7	37.2	13.8	2.6	10.4
35	Lakshadweep	12.8	53.6	18.4	12.1	28.6	14.7	12.3	36.2
36	Puducherry	0.3	5.4	2.1	7.0	9.8	7.9	4.4	7.9

WPR Dataset (PLFS)

In [107…
```
wpr = pd.read_csv('Cleaned Data/PLFS/WPR.csv',header=None,skiprows=2)
wpr_f = wpr[~wpr[0].str.contains('all India',case=False, na=False)]
wpr_all = wpr[wpr[0].str.contains('all India',case=False, na=False)]
```

In [108…
```
wpr_f.columns = col
wpr_f
```

Out[108…

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	56.5	38.3	47.3	55.6	22.4	38.5	56.3	33.5
1	Arunachal Pradesh	52.6	47.5	50.2	50.0	27.2	38.4	52.2	44.0
2	Assam	57.0	22.8	40.1	59.1	17.3	38.3	57.2	22.2
3	Bihar	45.4	10.1	28.1	44.1	7.6	26.7	45.3	9.9
4	Chhattisgarh	62.0	45.4	53.7	58.2	23.0	41.3	61.2	41.2
5	Delhi	52.8	11.1	35.5	52.5	13.9	34.9	52.5	13.9
6	Goa	53.5	18.4	36.5	52.8	19.3	36.3	53.1	18.9
7	Gujarat	62.1	35.1	48.9	60.1	17.6	39.9	61.2	27.9
8	Haryana	49.5	19.2	35.5	53.9	16.0	36.3	51.2	18.0
9	Himachal Pradesh	57.0	46.9	51.9	60.8	28.2	46.1	57.5	44.8
10	Jharkhand	49.5	20.3	34.9	47.0	10.6	29.3	49.0	18.5
11	Karnataka	57.6	32.5	45.1	57.0	21.5	39.6	57.4	28.4
12	Kerala	53.1	22.4	36.9	51.9	20.6	35.1	52.6	21.5
13	Madhya Pradesh	58.6	19.8	39.9	55.3	14.1	35.0	57.7	18.3
14	Maharashtra	58.8	36.4	47.7	57.8	21.3	40.3	58.4	30.1
15	Manipur	50.1	33.4	41.7	48.9	31.3	40.0	49.8	32.8
16	Meghalaya	51.7	36.2	43.8	48.0	29.4	38.3	51.1	35.2
17	Mizoram	50.3	30.1	40.5	47.7	29.1	38.2	49.1	29.6
18	Nagaland	53.1	38.5	45.6	45.2	28.1	36.8	50.8	35.7
19	Odisha	57.2	25.5	40.9	54.1	17.4	36.0	56.7	24.3
20	Punjab	58.9	23.2	41.2	59.5	16.4	38.5	59.1	20.6
21	Rajasthan	53.0	32.4	42.6	52.4	17.5	35.7	52.9	28.6
22	Sikkim	63.4	61.2	62.3	59.8	24.1	44.3	62.5	53.1
23	Tamil Nadu	56.5	35.1	45.7	56.7	21.6	38.6	56.6	28.9
24	Telangana	56.8	39.8	48.1	53.6	21.0	37.6	55.5	32.6
25	Tripura	62.8	36.4	49.5	55.3	22.4	38.4	61.5	33.9
26	Uttarakhand	50.7	27.1	39.0	53.2	13.6	33.6	51.3	23.7
27	Uttar Pradesh	50.3	14.4	32.5	52.8	9.8	32.2	50.8	13.4
28	West Bengal	60.7	17.1	38.9	61.3	16.9	39.0	60.9	17.1

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
29	Andaman & N. Island	60.3	30.4	46.0	62.4	18.2	40.7	61.2	24.9
30	Chandigarh	NaN	NaN	NaN	56.9	20.7	39.7	56.9	20.7
31	Dadra & Nagar Haveli & Daman & Diu	58.6	50.3	54.6	68.0	17.3	46.3	64.2	32.3
32	Jammu & Kashmir	50.4	15.9	33.8	54.5	13.2	34.6	51.2	15.4
33	Ladakh	53.7	36.1	45.6	56.7	15.3	37.6	54.1	33.3
34	Lakshadweep	63.9	6.1	36.9	49.5	8.4	30.0	53.7	7.8
35	Puducherry	56.9	30.2	43.7	53.0	20.6	35.7	54.5	24.0

In [109…
```
wpr_f.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 36 entries, 0 to 35
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   (State/UT, )           36 non-null     object
 1   (Rural, Male)          35 non-null     float64
 2   (Rural, Female)        35 non-null     float64
 3   (Rural, Person)        35 non-null     float64
 4   (Urban, Male)          36 non-null     float64
 5   (Urban, Female)        36 non-null     float64
 6   (Urban, Person)        36 non-null     float64
 7   (Rural+Urban, Male)    36 non-null     float64
 8   (Rural+Urban, Female)  36 non-null     float64
 9   (Rural+Urban, Person)  36 non-null     float64
dtypes: float64(9), object(1)
memory usage: 3.1+ KB
```

In [110…
```
# Filling missing values
wpr_f.loc[30,('Rural','Male')] = 0
wpr_f.loc[30,('Rural','Female')] = 0
wpr_f.loc[30,('Rural','Person')] = 0
```

In [111…
```
wpr_f.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 36 entries, 0 to 35
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   (State/UT, )           36 non-null     object
 1   (Rural, Male)          36 non-null     float64
 2   (Rural, Female)        36 non-null     float64
 3   (Rural, Person)        36 non-null     float64
 4   (Urban, Male)          36 non-null     float64
 5   (Urban, Female)        36 non-null     float64
 6   (Urban, Person)        36 non-null     float64
 7   (Rural+Urban, Male)    36 non-null     float64
 8   (Rural+Urban, Female)  36 non-null     float64
 9   (Rural+Urban, Person)  36 non-null     float64
dtypes: float64(9), object(1)
memory usage: 4.1+ KB
```

In [112…
```
wpr_f.describe()
```

Out[112…

	Rural			Urban			
	Male	**Female**	**Person**	**Male**	**Female**	**Person**	**M**
count	36.000000	36.000000	36.000000	36.000000	36.000000	36.000000	36.00
mean	54.036111	29.044444	41.808333	54.600000	19.247222	37.438889	55.20
std	10.345217	13.222350	10.011033	5.145928	5.968656	4.080846	4.50
min	0.000000	0.000000	0.000000	44.100000	7.600000	26.700000	45.30
25%	51.450000	19.650000	36.900000	52.275000	15.825000	35.550000	51.27
50%	56.500000	30.300000	42.150000	54.300000	18.750000	38.250000	55.00
75%	58.650000	36.400000	47.400000	57.900000	22.400000	39.625000	57.87
max	63.900000	61.200000	62.300000	68.000000	31.300000	46.300000	64.20

Gender Based Analysis

Male vs. Female WPR across states

In [113…
```
plt.figure(figsize=(12,6))
wpr_f_sorted = wpr_f.sort_values(by=[('Rural+Urban','Person')],ascending=Fal
sns.barplot(y=wpr_f_sorted['State/UT'],x=wpr_f_sorted[('Rural+Urban','Male')
sns.barplot(y=wpr_f_sorted['State/UT'],x=wpr_f_sorted[('Rural+Urban','Female
plt.xlabel('Worker Population Ratio (%)')
plt.ylabel('State/UT')
plt.title("Worker Population Ratio by Gender")
plt.legend()
plt.show()
```

Urban vs Rural Divide

In [114…

```
wpr_f[('Rural_Urban_Diff', 'Male')] = wpr_f[('Rural', 'Male')] - wpr_f[('Urb
wpr_f[('Rural_Urban_Diff', 'Female')] = wpr_f[('Rural', 'Female')] - wpr_f[(
wpr_f[('Rural_Urban_Diff', 'Person')] = wpr_f[('Rural', 'Person')] - wpr_f[(

wpr_f
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\339968114.py:1: SettingWit
hCopyWarning:
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
  wpr_f[('Rural_Urban_Diff', 'Male')] = wpr_f[('Rural', 'Male')] - wpr_f[('U
rban', 'Male')]
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\339968114.py:2: SettingWit
hCopyWarning:
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
  wpr_f[('Rural_Urban_Diff', 'Female')] = wpr_f[('Rural', 'Female')] - wpr_f
[('Urban', 'Female')]
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\339968114.py:3: SettingWit
hCopyWarning:
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
  wpr_f[('Rural_Urban_Diff', 'Person')] = wpr_f[('Rural', 'Person')] - wpr_f
[('Urban', 'Person')]
```

Out[114…

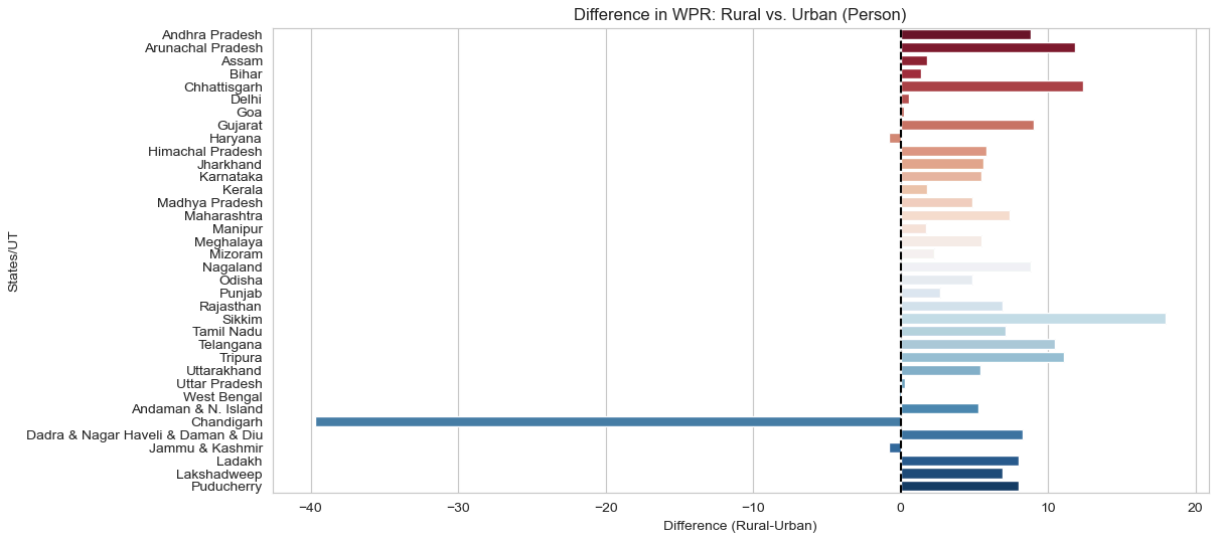
	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	56.5	38.3	47.3	55.6	22.4	38.5	56.3	33.5
1	Arunachal Pradesh	52.6	47.5	50.2	50.0	27.2	38.4	52.2	44.0
2	Assam	57.0	22.8	40.1	59.1	17.3	38.3	57.2	22.2
3	Bihar	45.4	10.1	28.1	44.1	7.6	26.7	45.3	9.9
4	Chhattisgarh	62.0	45.4	53.7	58.2	23.0	41.3	61.2	41.2
5	Delhi	52.8	11.1	35.5	52.5	13.9	34.9	52.5	13.9
6	Goa	53.5	18.4	36.5	52.8	19.3	36.3	53.1	18.9
7	Gujarat	62.1	35.1	48.9	60.1	17.6	39.9	61.2	27.9
8	Haryana	49.5	19.2	35.5	53.9	16.0	36.3	51.2	18.0
9	Himachal Pradesh	57.0	46.9	51.9	60.8	28.2	46.1	57.5	44.8
10	Jharkhand	49.5	20.3	34.9	47.0	10.6	29.3	49.0	18.5
11	Karnataka	57.6	32.5	45.1	57.0	21.5	39.6	57.4	28.4
12	Kerala	53.1	22.4	36.9	51.9	20.6	35.1	52.6	21.5
13	Madhya Pradesh	58.6	19.8	39.9	55.3	14.1	35.0	57.7	18.3
14	Maharashtra	58.8	36.4	47.7	57.8	21.3	40.3	58.4	30.1
15	Manipur	50.1	33.4	41.7	48.9	31.3	40.0	49.8	32.8
16	Meghalaya	51.7	36.2	43.8	48.0	29.4	38.3	51.1	35.2
17	Mizoram	50.3	30.1	40.5	47.7	29.1	38.2	49.1	29.6
18	Nagaland	53.1	38.5	45.6	45.2	28.1	36.8	50.8	35.7
19	Odisha	57.2	25.5	40.9	54.1	17.4	36.0	56.7	24.3
20	Punjab	58.9	23.2	41.2	59.5	16.4	38.5	59.1	20.6
21	Rajasthan	53.0	32.4	42.6	52.4	17.5	35.7	52.9	28.6
22	Sikkim	63.4	61.2	62.3	59.8	24.1	44.3	62.5	53.1
23	Tamil Nadu	56.5	35.1	45.7	56.7	21.6	38.6	56.6	28.9
24	Telangana	56.8	39.8	48.1	53.6	21.0	37.6	55.5	32.6
25	Tripura	62.8	36.4	49.5	55.3	22.4	38.4	61.5	33.9
26	Uttarakhand	50.7	27.1	39.0	53.2	13.6	33.6	51.3	23.7
27	Uttar Pradesh	50.3	14.4	32.5	52.8	9.8	32.2	50.8	13.4
28	West Bengal	60.7	17.1	38.9	61.3	16.9	39.0	60.9	17.1

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
29	Andaman & N. Island	60.3	30.4	46.0	62.4	18.2	40.7	61.2	24.9
30	Chandigarh	0.0	0.0	0.0	56.9	20.7	39.7	56.9	20.7
31	Dadra & Nagar Haveli & Daman & Diu	58.6	50.3	54.6	68.0	17.3	46.3	64.2	32.3
32	Jammu & Kashmir	50.4	15.9	33.8	54.5	13.2	34.6	51.2	15.4
33	Ladakh	53.7	36.1	45.6	56.7	15.3	37.6	54.1	33.3
34	Lakshadweep	63.9	6.1	36.9	49.5	8.4	30.0	53.7	7.8
35	Puducherry	56.9	30.2	43.7	53.0	20.6	35.7	54.5	24.0

In [115…

```
plt.figure(figsize=(12,6))
sns.barplot(y=wpr_f['State/UT'], x=wpr_f[('Rural_Urban_Diff','Person')], pal
plt.axvline(x=0, color='black',linestyle='--')

plt.title('Difference in WPR: Rural vs. Urban (Person)')
plt.xlabel('Difference (Rural-Urban)')
plt.ylabel('States/UT')
plt.show()
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\2980443822.py:2: FutureWar
ning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the
same effect.

  sns.barplot(y=wpr_f['State/UT'], x=wpr_f[('Rural_Urban_Diff','Person')], p
alette='RdBu')
```



Top & Bottom 5 states

In [116…
```
# Define correct column references
state_col = ('State/UT', '')  # Ensure correct MultiIndex format
wpr_col = ('Rural+Urban', 'Person')  # MultiIndex column

# Top 5 states based on WPR (Rural+Urban)
top_states = wpr_f.nlargest(5, wpr_col)[[state_col, wpr_col]]

# Bottom 5 states based on WPR (Rural+Urban)
bottom_states = wpr_f.nsmallest(5, wpr_col)[[state_col, wpr_col]]

# Print results
print('Top 5 states by WPR: \n', top_states)
print('Bottom 5 states by WPR: \n', bottom_states)
```

```
Top 5 states by WPR:
                                State/UT Rural+Urban
                                              Person
22                                Sikkim        58.2
4                             Chhattisgarh        51.3
9                        Himachal Pradesh        51.1
31  Dadra & Nagar Haveli & Daman & Diu        49.9
1                        Arunachal Pradesh        48.2
Bottom 5 states by WPR:
            State/UT Rural+Urban
                           Person
3              Bihar        28.0
34       Lakshadweep        31.9
27     Uttar Pradesh        32.4
10         Jharkhand        33.9
32   Jammu & Kashmir        33.9
```

In [117…
```
wpr_f.columns.tolist()
```

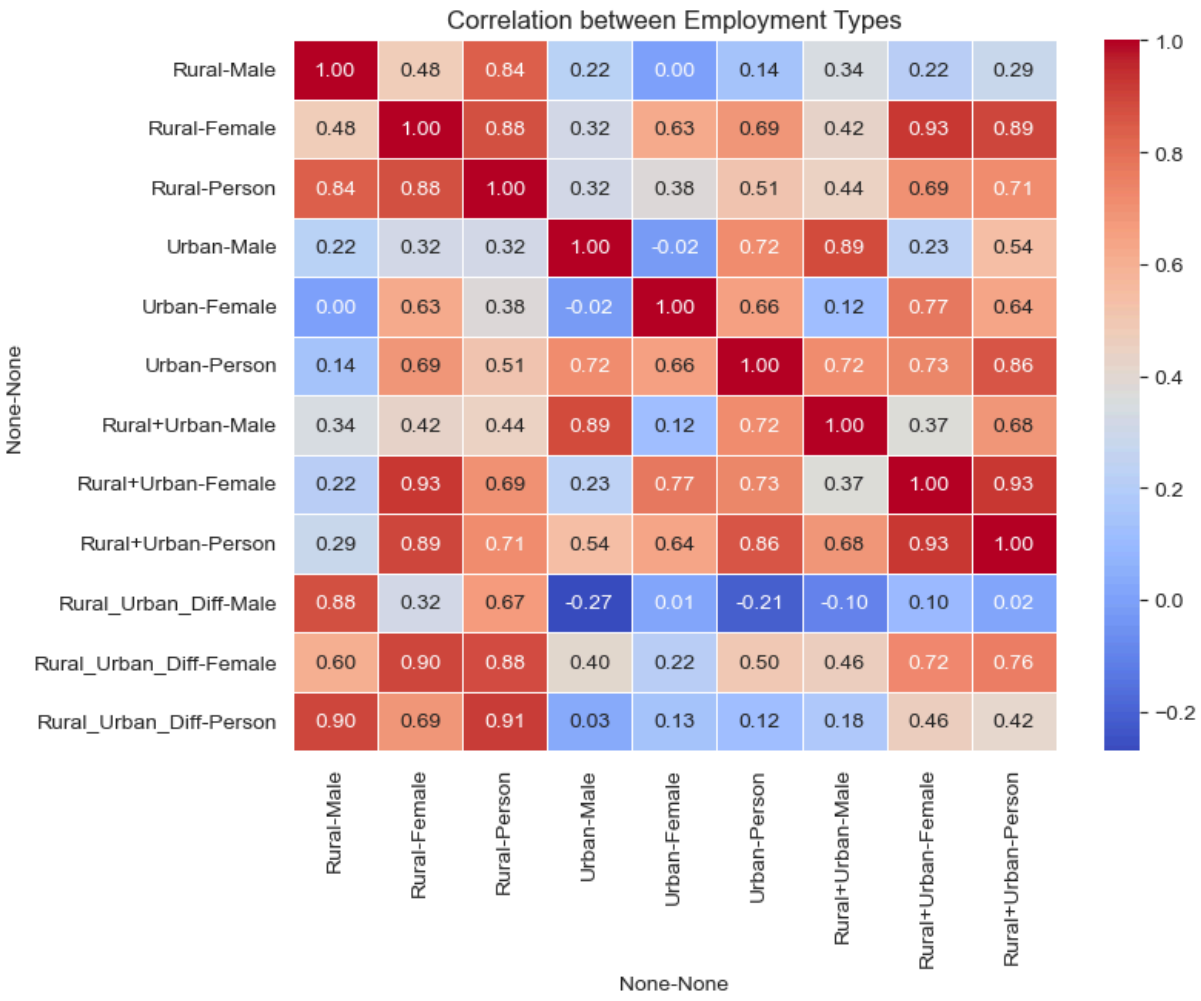
Out[117…
```
[('State/UT', ''),
 ('Rural', 'Male'),
 ('Rural', 'Female'),
 ('Rural', 'Person'),
 ('Urban', 'Male'),
 ('Urban', 'Female'),
 ('Urban', 'Person'),
 ('Rural+Urban', 'Male'),
 ('Rural+Urban', 'Female'),
 ('Rural+Urban', 'Person'),
 ('Rural_Urban_Diff', 'Male'),
 ('Rural_Urban_Diff', 'Female'),
 ('Rural_Urban_Diff', 'Person')]
```

Correlation Analysis

In [118…
```
corr_matrix = wpr_f.corr(numeric_only = True).drop('Rural_Urban_Diff',axis=1
```

```
plt.figure(figsize=(8,6))
sns.heatmap(corr_matrix,annot=True,cmap='coolwarm',fmt='.2f',linewidths=0.5)
plt.title('Correlation between Employment Types')
plt.show()
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\3560686108.py:1: Performan
ceWarning: dropping on a non-lexsorted multi-index without a level parameter
may impact performance.
  corr_matrix = wpr_f.corr(numeric_only = True).drop('Rural_Urban_Diff',axis
=1)
```

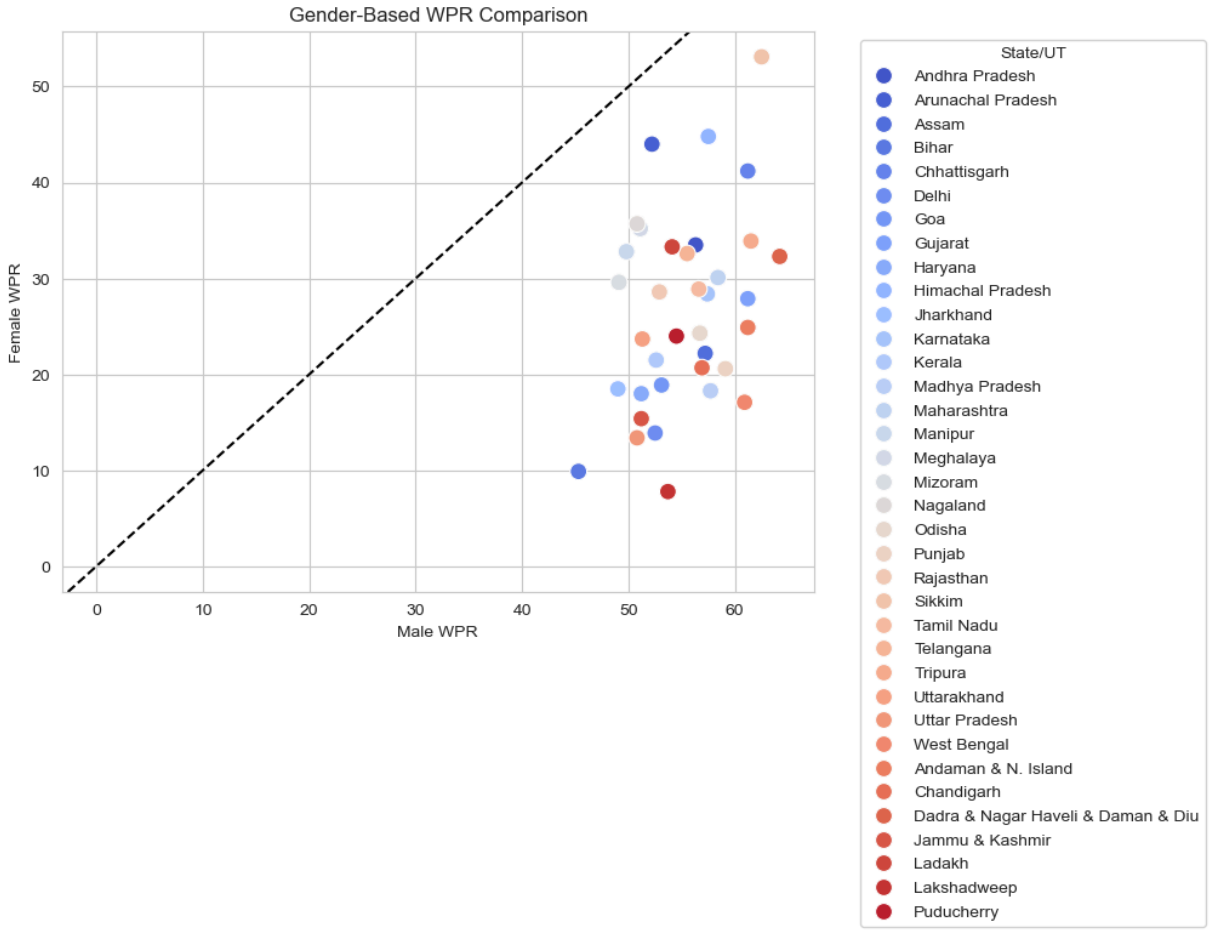


Scatter Plot for Outliers

Checking for states where Male WPR is high but Female WPR is very low

In [119…

```
plt.figure(figsize=(8,6))
sns.scatterplot(x=wpr_f['Rural+Urban','Male'], y=wpr_f['Rural+Urban','Female
plt.axline((0,0), slope=1, color="black", linestyle="--")  # Reference line
plt.xlabel("Male WPR")
plt.ylabel("Female WPR")
plt.title("Gender-Based WPR Comparison")
plt.legend(title="State/UT", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



All values are below the line which indicates there is a severe gender gap in workforce population

In [120…
```
lfpr_15_29 = pd.read_csv('Cleaned Data/PLFS/LFPR_15_29.csv',header=None,skip
lfpr_15_29.head()
```

Out[120…

	0	1	2	3	4	5	6	7	8	9
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9	45.6
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1	50.5
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1	51.3
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8	34.6
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6	61.7

In [121…
```
lfpr_15_29_f = lfpr_15_29[~lfpr_15_29[0].str.contains('all India',case=False
```

In [122…
```
lfpr_15_29_f
```

Out[122…

	0	1	2	3	4	5	6	7	8	9
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9	45.6
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1	50.5
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1	51.3
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8	34.6
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6	61.7
5	Delhi	63.0	11.9	47.0	46.6	18.1	34.6	47.1	18.0	34.9
6	Goa	62.7	20.4	43.3	47.9	32.9	40.2	54.4	28.1	41.5
7	Gujarat	75.4	49.0	62.7	68.6	28.1	50.2	72.5	40.6	57.5
8	Haryana	56.8	12.9	37.3	59.6	17.4	40.9	57.9	14.7	38.7
9	Himachal Pradesh	70.3	60.3	65.3	62.1	37.4	52.2	68.9	57.5	63.3
10	Jharkhand	67.9	43.4	55.2	49.8	17.3	34.4	64.1	38.6	51.1
11	Karnataka	62.4	20.0	42.3	57.7	25.9	42.6	60.5	22.4	42.4
12	Kerala	51.5	30.5	40.9	51.0	30.2	40.3	51.2	30.3	40.6
13	Madhya Pradesh	80.3	43.9	62.9	64.9	21.7	43.9	76.4	38.1	58.1
14	Maharashtra	57.6	25.5	42.9	64.2	28.0	47.6	60.5	26.6	44.9
15	Manipur	37.9	24.1	30.8	39.9	26.5	33.2	38.5	24.8	31.5
16	Meghalaya	70.4	61.8	66.1	47.8	34.5	41.1	66.5	57.2	61.8
17	Mizoram	32.4	22.8	28.3	24.6	21.8	23.3	29.0	22.3	26.0
18	Nagaland	64.5	49.4	56.5	47.7	28.1	37.9	58.6	42.5	50.2
19	Odisha	69.7	39.4	54.0	55.9	27.6	42.5	67.3	37.6	52.1
20	Punjab	66.3	28.4	47.6	66.1	22.0	45.2	66.2	25.9	46.6
21	Rajasthan	66.5	40.5	53.1	58.5	24.1	43.0	64.2	36.5	50.4
22	Sikkim	67.2	56.4	61.8	58.2	25.7	45.5	64.1	48.8	57.0
23	Tamil Nadu	60.2	24.7	42.8	57.4	22.9	39.6	59.0	23.8	41.3
24	Telangana	67.7	35.5	51.7	61.9	27.4	45.0	65.2	32.1	48.8
25	Tripura	64.3	32.3	49.8	47.1	20.1	33.4	61.7	30.1	47.1
26	Uttarakhand	62.7	43.0	52.8	57.7	20.5	38.5	61.4	36.9	49.0
27	Uttar Pradesh	65.0	23.5	44.0	60.3	15.0	38.9	64.0	21.8	42.9
28	West Bengal	70.3	31.5	49.7	66.5	30.4	47.3	69.2	31.2	49.0
29	Andaman & N. Island	72.0	67.7	70.1	78.4	39.6	59.9	75.1	53.4	65.1
30	Chandigarh	NaN	NaN	NaN	65.3	34.1	50.0	65.3	34.1	50.0
31	Dadra & Nagar Haveli & Daman & Diu	72.0	65.5	69.1	80.1	17.7	54.4	76.6	40.1	61.0

	0	1	2	3	4	5	6	7	8	9
32	Jammu & Kashmir	57.3	41.2	49.8	52.8	31.9	43.2	56.5	39.6	48.7
33	Ladakh	45.0	33.0	38.9	57.8	28.8	45.3	46.9	32.5	39.8
34	Lakshadweep	82.6	41.8	66.2	60.1	15.3	42.3	66.7	23.1	49.3
35	Puducherry	56.8	28.3	42.3	62.2	27.5	42.9	60.1	27.8	42.7

In [123…
```
col = pd.MultiIndex.from_tuples([
    ('State/UT',''),
    ('Rural','Male'),('Rural','Female'),('Rural','Person'),
    ('Urban','Male'),('Urban','Female'),('Urban','Person'),
    ('Rural+Urban','Male'),('Rural+Urban','Female'),('Rural+Urban','Person')
])
```

In [124…
```
lfpr_15_29_f.columns = col
```

In [125…
```
lfpr_15_29_f.head()
```

Out[125…

	State/UT	Rural			Urban			Rural+		
		Male	Female	Person	Male	Female	Person	Male	Female	P
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9	
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1	
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1	
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8	
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6	

In [126…
```
lfpr_15_29_f.loc[30,('Rural','Male')] = 0
lfpr_15_29_f.loc[30,('Rural','Female')] = 0
lfpr_15_29_f.loc[30,('Rural','Person')] = 0
```

In [127…
```
lfpr_15_59 = pd.read_csv('Cleaned Data/PLFS/LFPR_15_59.csv',header=None,skip

lfpr_15_59_f = lfpr_15_59[~lfpr_15_59[0].str.contains('all India',case=False

lfpr_15_59_f.columns = col

lfpr_15_59_f.loc[30,('Rural','Male')] = 0
lfpr_15_59_f.loc[30,('Rural','Female')] = 0
lfpr_15_59_f.loc[30,('Rural','Person')] = 0
```

In [128…
```
lfpr_15_above = pd.read_csv('Cleaned Data/PLFS/LFPR_15_above.csv',header=Non

lfpr_15_above_f = lfpr_15_above[~lfpr_15_above[0].str.contains('all India',c

lfpr_15_above_f.columns = col
```

```
lfpr_15_above_f.loc[30,('Rural','Male')] = 0
lfpr_15_above_f.loc[30,('Rural','Female')] = 0
lfpr_15_above_f.loc[30,('Rural','Person')] = 0
```

In [129…
```
lfpr_all_ages = pd.read_csv('Cleaned Data/PLFS/LFPR_all_ages.csv',header=Nor

lfpr_all_ages_f = lfpr_all_ages[~lfpr_all_ages[0].str.contains('all India',c

lfpr_all_ages_f.columns = col

lfpr_all_ages_f.loc[30,('Rural','Male')] = 0
lfpr_all_ages_f.loc[30,('Rural','Female')] = 0
lfpr_all_ages_f.loc[30,('Rural','Person')] = 0
```

In [130…
```
frames = [lfpr_15_29_f,lfpr_15_59_f,lfpr_15_above_f,lfpr_all_ages_f]
lfpr_res = pd.concat(frames)
```

In [131…
```
lfpr_res.reset_index(drop=True,inplace=True)
```

In [132…
```
lfpr_res
```

Out[132…

	State/UT	Rural			Urban			Rur	
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6
...	...	...	...	...	...	...	...	...	...
139	Dadra & Nagar Haveli & Daman & Diu	62.1	54.3	58.3	69.9	18.5	47.9	66.7	34.1
140	Jammu & Kashmir	55.7	41.9	49.0	58.2	25.8	42.5	56.2	38.8
141	Ladakh	55.5	46.4	51.3	61.0	26.3	45.0	56.2	43.1
142	Lakshadweep	73.3	14.5	45.7	56.3	12.4	35.4	61.2	13.6
143	Puducherry	58.6	37.1	48.0	58.2	24.4	40.2	58.4	28.9

144 rows × 10 columns

In [133…
```
lfpr_res.iloc[0:36]
```

Out[133…

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6
5	Delhi	63.0	11.9	47.0	46.6	18.1	34.6	47.1	18.0
6	Goa	62.7	20.4	43.3	47.9	32.9	40.2	54.4	28.1
7	Gujarat	75.4	49.0	62.7	68.6	28.1	50.2	72.5	40.6
8	Haryana	56.8	12.9	37.3	59.6	17.4	40.9	57.9	14.7
9	Himachal Pradesh	70.3	60.3	65.3	62.1	37.4	52.2	68.9	57.5
10	Jharkhand	67.9	43.4	55.2	49.8	17.3	34.4	64.1	38.6
11	Karnataka	62.4	20.0	42.3	57.7	25.9	42.6	60.5	22.4
12	Kerala	51.5	30.5	40.9	51.0	30.2	40.3	51.2	30.3
13	Madhya Pradesh	80.3	43.9	62.9	64.9	21.7	43.9	76.4	38.1
14	Maharashtra	57.6	25.5	42.9	64.2	28.0	47.6	60.5	26.6
15	Manipur	37.9	24.1	30.8	39.9	26.5	33.2	38.5	24.8
16	Meghalaya	70.4	61.8	66.1	47.8	34.5	41.1	66.5	57.2
17	Mizoram	32.4	22.8	28.3	24.6	21.8	23.3	29.0	22.3
18	Nagaland	64.5	49.4	56.5	47.7	28.1	37.9	58.6	42.5
19	Odisha	69.7	39.4	54.0	55.9	27.6	42.5	67.3	37.6
20	Punjab	66.3	28.4	47.6	66.1	22.0	45.2	66.2	25.9
21	Rajasthan	66.5	40.5	53.1	58.5	24.1	43.0	64.2	36.5
22	Sikkim	67.2	56.4	61.8	58.2	25.7	45.5	64.1	48.8
23	Tamil Nadu	60.2	24.7	42.8	57.4	22.9	39.6	59.0	23.8
24	Telangana	67.7	35.5	51.7	61.9	27.4	45.0	65.2	32.1
25	Tripura	64.3	32.3	49.8	47.1	20.1	33.4	61.7	30.1
26	Uttarakhand	62.7	43.0	52.8	57.7	20.5	38.5	61.4	36.9
27	Uttar Pradesh	65.0	23.5	44.0	60.3	15.0	38.9	64.0	21.8
28	West Bengal	70.3	31.5	49.7	66.5	30.4	47.3	69.2	31.2

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
29	Andaman & N. Island	72.0	67.7	70.1	78.4	39.6	59.9	75.1	53.4
30	Chandigarh	0.0	0.0	0.0	65.3	34.1	50.0	65.3	34.1
31	Dadra & Nagar Haveli & Daman & Diu	72.0	65.5	69.1	80.1	17.7	54.4	76.6	40.1
32	Jammu & Kashmir	57.3	41.2	49.8	52.8	31.9	43.2	56.5	39.6
33	Ladakh	45.0	33.0	38.9	57.8	28.8	45.3	46.9	32.5
34	Lakshadweep	82.6	41.8	66.2	60.1	15.3	42.3	66.7	23.1
35	Puducherry	56.8	28.3	42.3	62.2	27.5	42.9	60.1	27.8

In [134…

```
categories = ['15-29 Age','15-59 Age','15 and above Age','All Ages'] * 9

rows_per_set = 36

category_col = [categories[i // rows_per_set] for i in range(len(lfpr_res))]
lfpr_res['Age Category'] = category_col
lfpr_res
```

Out[134…

	State/UT	Rural			Urban			Rur	
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6
...	...	...	...	...	...	...	...	...	...
139	Dadra & Nagar Haveli & Daman & Diu	62.1	54.3	58.3	69.9	18.5	47.9	66.7	34.1
140	Jammu & Kashmir	55.7	41.9	49.0	58.2	25.8	42.5	56.2	38.8
141	Ladakh	55.5	46.4	51.3	61.0	26.3	45.0	56.2	43.1
142	Lakshadweep	73.3	14.5	45.7	56.3	12.4	35.4	61.2	13.6
143	Puducherry	58.6	37.1	48.0	58.2	24.4	40.2	58.4	28.9

144 rows × 11 columns

HCES

1. Absolute and percentage breakup of MPCE by item-groups in 2023-24 All India

In [135…
```
ab_perc_mpce = pd.read_csv('Cleaned Data/HCES/Abs_Perc_MPCE.csv',header=None
ab_perc_mpce.reset_index(drop=True,inplace=True)
```

In [136…
```
ab_perc_mpce = ab_perc_mpce.drop(0,axis=1)
```

In [137…
```
cols = pd.MultiIndex.from_tuples([
    ('Item group',''),
    ('MPCE(Rs.)','Rural'),('MPCE(Rs.)','Urban'),
    ('% share in total MPCE','Rural'),('% share in total MPCE','Urban')
])
ab_perc_mpce.columns = cols
```

In [138…]
```
ab_perc_mpce
```

Out[138…]

	Item group	MPCE(Rs.)		% share in total MPCE	
		Rural	Urban	Rural	Urban
0	cereals & cereal substitutes	206	263	4.99	3.76
1	pulses & their products*	84	98	2.04	1.40
2	sugar & salt	37	40	0.89	0.57
3	milk & milk products	348	503	8.44	7.19
4	vegetables	248	288	6.03	4.12
5	fruits	158	271	3.85	3.87
6	egg, fish & meat	203	249	4.92	3.56
7	edible oil	114	127	2.77	1.82
8	spices	135	161	3.27	2.30
9	beverages, refreshments, processed food#	406	776	9.84	11.09
10	food total	1939	2776	47.04	39.68
11	pan, tobacco & intoxicants	158	166	3.84	2.37
12	fuel and light	252	391	6.11	5.59
13	education	133	418	3.24	5.97
14	medical	282	409	6.83	5.85
15	conveyance	313	592	7.59	8.46
16	consumer services excluding conveyance	217	400	5.25	5.72
17	misc. goods, entertainment	256	484	6.22	6.92
18	rent	23	460	0.56	6.58
19	taxes and cesses	9	23	0.21	0.33
20	clothing, bedding & footwear	273	396	6.63	5.66
21	durable goods	267	481	6.48	6.87
22	non-food total	2183	4220	52.96	60.32
23	all items	4122	6996	100.00	100.00

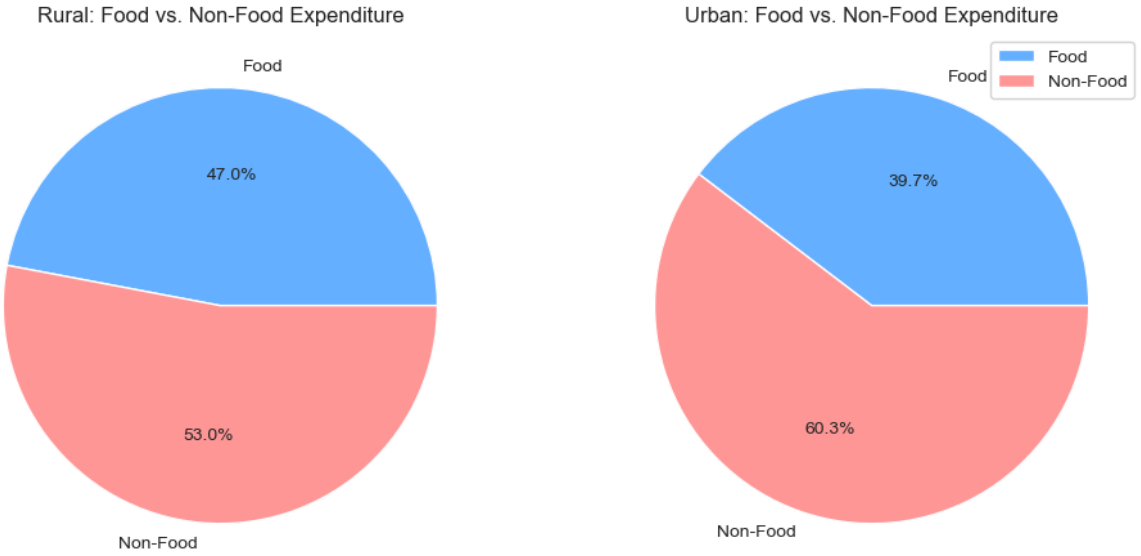
In [139…]
```
ab_perc_mpceF = ab_perc_mpce[~ab_perc_mpce['Item group'].str.contains('all i
ab_perc_mpceA = ab_perc_mpce[ab_perc_mpce['Item group'].str.contains('all it
```

Analysis

1. Food vs. Non-Food Expenditure (Essentials vs. Discretionary Spending)

In [140…

```
data = {
    'Category':['Food','Non-Food'],
    "Rural": [1939,2183],
    'Urban': [2776,4220]
}
df = pd.DataFrame(data)

fig,axes = plt.subplots(1,2,figsize=(12,6))

#Rural Pie Chart
axes[0].pie(df['Rural'],labels=df['Category'],autopct='%0.1f%%',colors=["#66
axes[0].set_title('Rural: Food vs. Non-Food Expenditure')

#Urban Pie Chart
axes[1].pie(df['Urban'],labels=df['Category'],autopct='%0.1f%%',colors=["#66
axes[1].set_title('Urban: Food vs. Non-Food Expenditure')
plt.legend()
plt.show()
```



Food spending percentage is higher in rural areas. It suggests they spend more on necessities.

Urban areas spend more on non food, it shows a shift towards discretionary and service-based epenses.

2. Major Spending Categories (Top 5 in Rural & Urban)

In [141…

```
df = ab_perc_mpceF[~ab_perc_mpceF['Item group'].isin(['food total','non-fooc
df
```

Out[141…

	Item group	MPCE(Rs.)		% share in total MPCE	
		Rural	Urban	Rural	Urban
0	cereals & cereal substitutes	206	263	4.99	3.76
1	pulses & their products*	84	98	2.04	1.40
2	sugar & salt	37	40	0.89	0.57
3	milk & milk products	348	503	8.44	7.19
4	vegetables	248	288	6.03	4.12
5	fruits	158	271	3.85	3.87
6	egg, fish & meat	203	249	4.92	3.56
7	edible oil	114	127	2.77	1.82
8	spices	135	161	3.27	2.30
9	beverages, refreshments, processed food#	406	776	9.84	11.09
11	pan, tobacco & intoxicants	158	166	3.84	2.37
12	fuel and light	252	391	6.11	5.59
13	education	133	418	3.24	5.97
14	medical	282	409	6.83	5.85
15	conveyance	313	592	7.59	8.46
16	consumer services excluding conveyance	217	400	5.25	5.72
17	misc. goods, entertainment	256	484	6.22	6.92
18	rent	23	460	0.56	6.58
19	taxes and cesses	9	23	0.21	0.33
20	clothing, bedding & footwear	273	396	6.63	5.66
21	durable goods	267	481	6.48	6.87

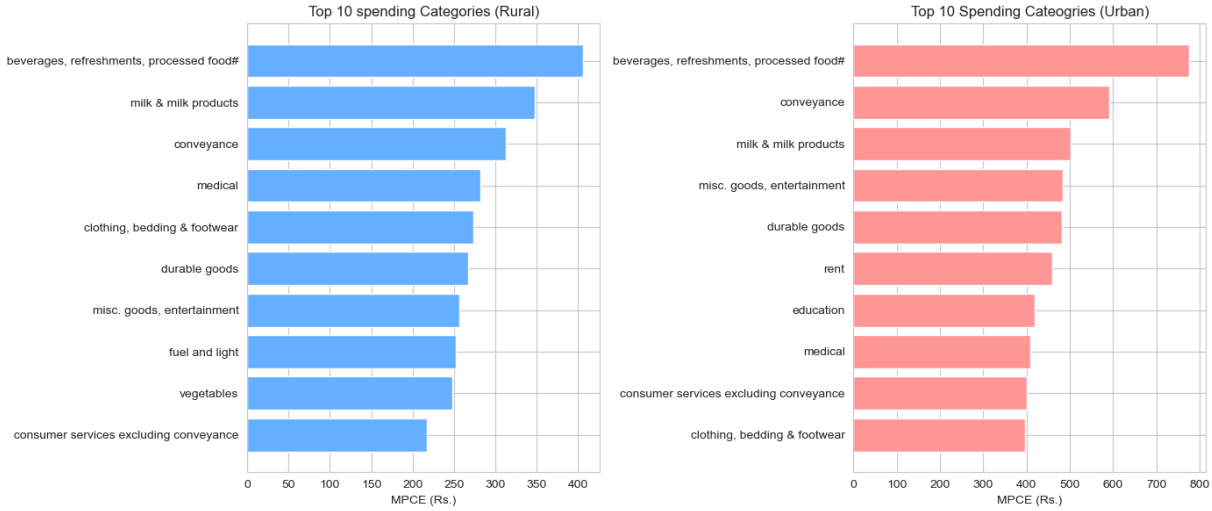
In [142…

```
top5_rural = df.nlargest(10, ('MPCE(Rs.)',"Rural"))
top5_urban = df.nlargest(10, ('MPCE(Rs.)','Urban'))
```

In [143…

```
fig, axes = plt.subplots(1,2,figsize=(14,6))

#Rural Plot
axes[0].barh(top5_rural['Item group'], top5_rural[('MPCE(Rs.)','Rural')],col
axes[0].set_title('Top 10 spending Categories (Rural)')
axes[0].invert_yaxis()
axes[0].set_xlabel('MPCE (Rs.)')

#Urban Plot
axes[1].barh(top5_urban['Item group'], top5_urban[('MPCE(Rs.)','Urban')], co
```

```
axes[1].set_title("Top 10 Spending Cateogries (Urban)")
axes[1].invert_yaxis()
axes[1].set_xlabel('MPCE (Rs.)')

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



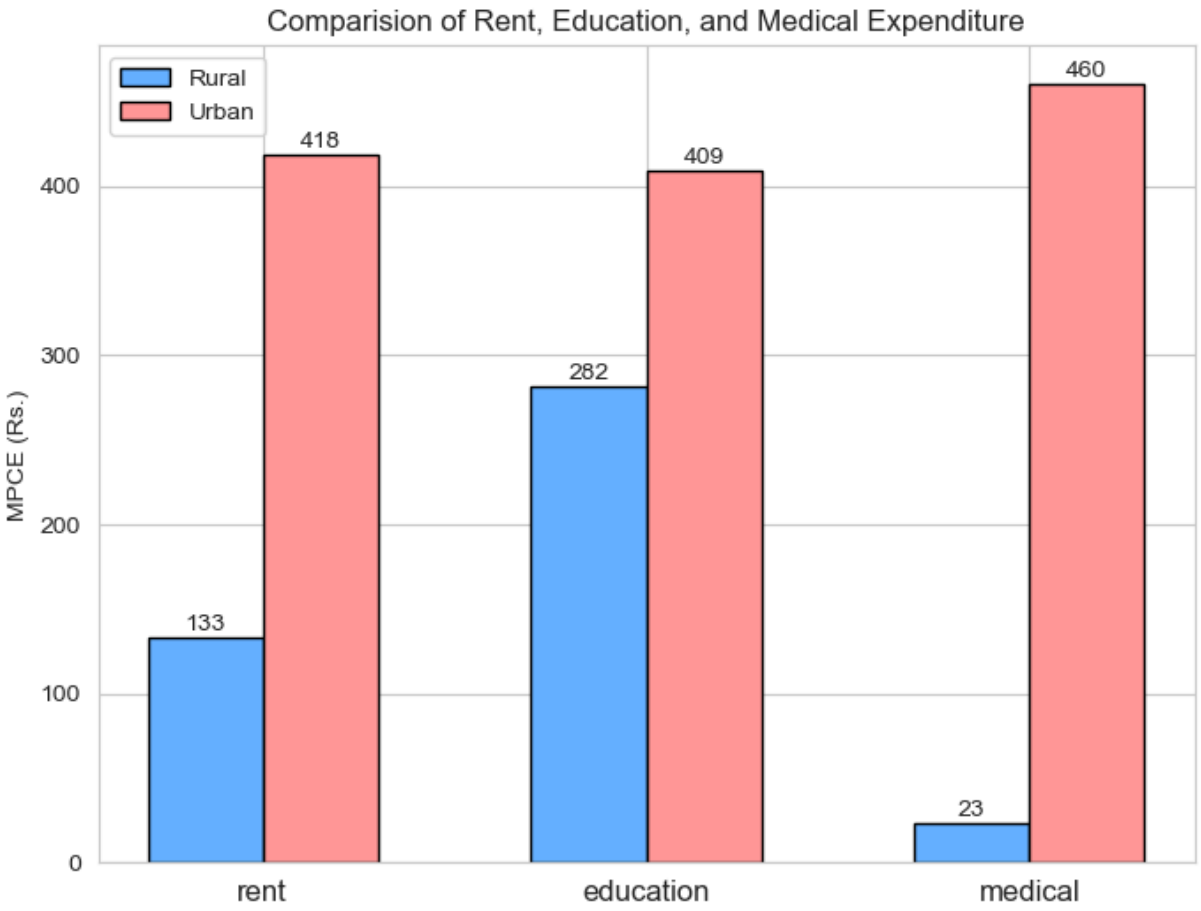
Top 1 Spending Categories is same for both Rural & Urban

3. Rent, Education and Medical Expenditure Comparisiion

In [144…

```
categories = ['rent','education','medical']
df_selected = ab_perc_mpceF[ab_perc_mpceF['Item group'].isin(categories)]

x = np.arange(len(categories))
width = 0.3

fig,ax = plt.subplots(figsize=(8,6))

bars1 = ax.bar(x-width/2, df_selected[('MPCE(Rs.)','Rural')],width,label='Ru
bars2 = ax.bar(x+width/2,df_selected[('MPCE(Rs.)','Urban')],width,label='Urb

ax.set_xticks(x)
ax.set_xticklabels(categories,fontsize=12)
ax.set_ylabel('MPCE (Rs.)')
ax.set_title('Comparision of Rent, Education, and Medical Expenditure')
ax.legend()

#display values on top of bars
for bars in [bars1, bars2]:
    for bar in bars:
        height = bar.get_height()
        ax.annotate(f'{height:.0f}', xy=(bar.get_x() + bar.get_width()/2, he
                    xytext=(0, 3), textcoords="offset points", ha='center',
```

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

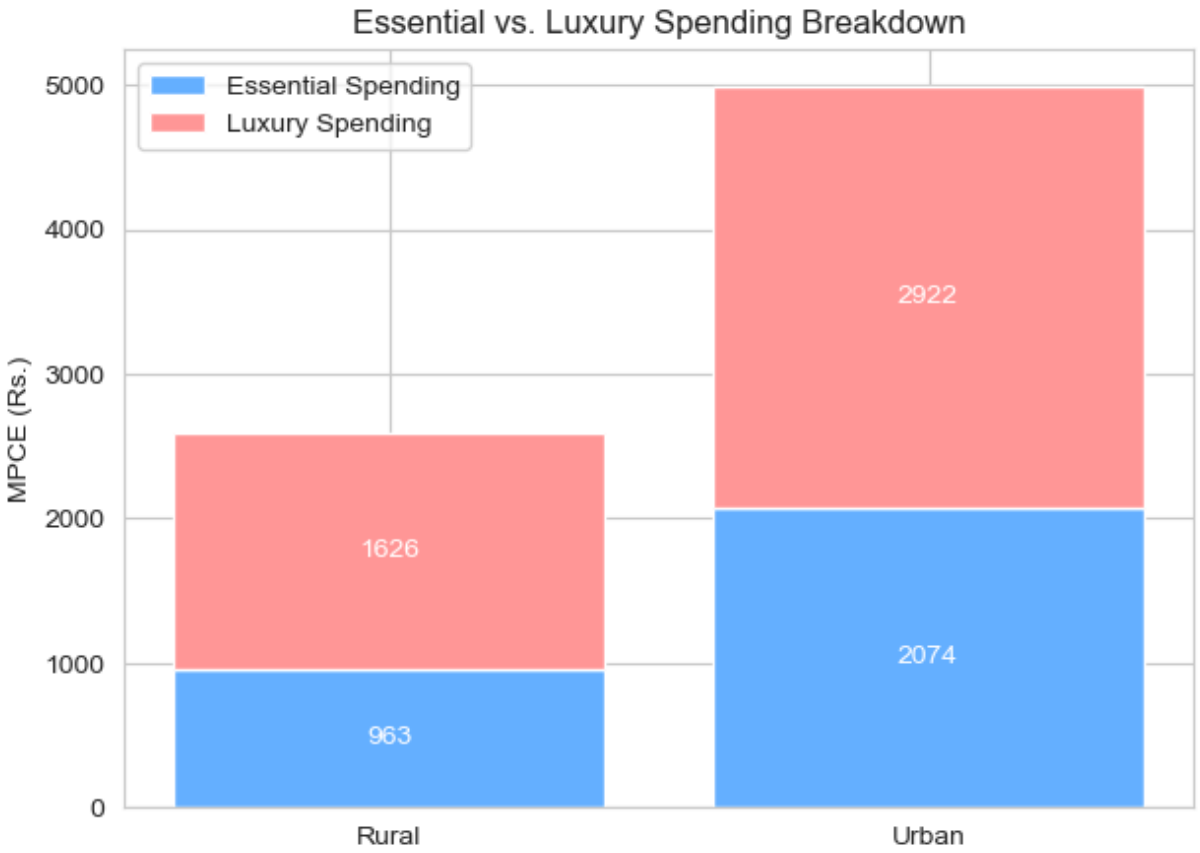


4. Essential vs Luxury Spending Breakdown

In [145…

```
# Define essential & luxury categories
essential_items = ["food total", "fuel and light", "medical", "education", "
luxury_items = ["misc. goods, entertainment", "consumer services excluding c
                "beverages, refreshments, processed food#", "conveyance", "t

# Compute total spending in each category
essential_rural = df[df["Item group"].isin(essential_items)][('MPCE(Rs.)',"R
luxury_rural = df[df["Item group"].isin(luxury_items)][('MPCE(Rs.)',"Rural")
essential_urban = df[df["Item group"].isin(essential_items)][('MPCE(Rs.)',"U
luxury_urban = df[df["Item group"].isin(luxury_items)][('MPCE(Rs.)',"Urban")

# Create DataFrame for visualization
data = {
    "Category": ["Rural", "Urban"],
    "Essential": [essential_rural, essential_urban],
    "Luxury": [luxury_rural, luxury_urban]
}
df_vis = pd.DataFrame(data)

# Plot Stacked Bar Chart
fig, ax = plt.subplots(figsize=(7, 5))
ax.bar(df_vis["Category"], df_vis["Essential"], label="Essential Spending",
```

```
ax.bar(df_vis["Category"], df_vis["Luxury"], bottom=df_vis["Essential"], lab

# Labels & Title
ax.set_ylabel("MPCE (Rs.)")
ax.set_title("Essential vs. Luxury Spending Breakdown")
ax.legend()

# Show values on bars
for i, row in df_vis.iterrows():
    ax.text(i, row["Essential"] / 2, f"{row['Essential']:.0f}", ha="center",
    ax.text(i, row["Essential"] + row["Luxury"] / 2, f"{row['Luxury']:.0f}",

# Show plot
plt.show()
```



Average MPCE and share of food and non food

In [146…
```
avg_mpce = pd.read_csv('Cleaned Data/HCES/Avg_MPCE.csv',header=None,skiprows
```

In [147…
```
avg_mpce.drop(0,axis=1,inplace=True)
```

In [148…
```
c = pd.MultiIndex.from_tuples([
    ('Item group',''),
    ('Rural India','Average MPCE(Rs.)'),('Rural India','Share in MPCE(%)'),
    ('Urban India','Average MPCE(Rs.)'),('Urban India','Share in MPCE(%)')
])
avg_mpce.columns = c
```

In [149…

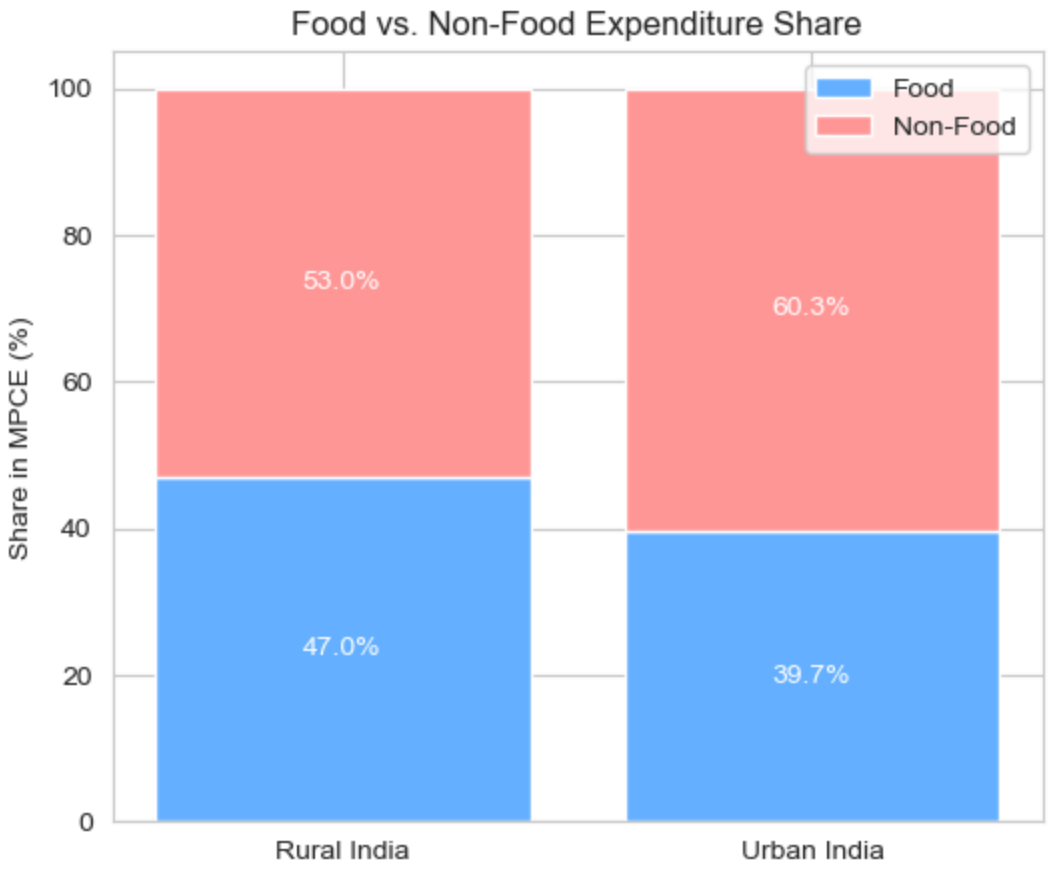
```
avg_mpce
```

Out[149…

	Item group	Rural India		Urban India	
		Average MPCE(Rs.)	Share in MPCE(%)	Average MPCE(Rs.)	Share in MPCE(%)
0	Food	1939.0	47.04	2776.0	39.68
1	Non-food	2183.0	52.96	4220.0	60.32

In [150…

```
categories = ["Rural India", "Urban India"]
food_expense = [47.04, 39.68]  # Share in MPCE (%)
non_food_expense = [52.96, 60.32]  # Share in MPCE (%)

# Plot
fig, ax = plt.subplots(figsize=(6, 5))
bars1 = ax.bar(categories, food_expense, label="Food", color="#66b3ff")
bars2 = ax.bar(categories, non_food_expense, bottom=food_expense, label="Nor

# Labels
ax.set_ylabel("Share in MPCE (%)")
ax.set_title("Food vs. Non-Food Expenditure Share")
ax.legend()

# Show values on bars
for bars in [bars1, bars2]:
    for bar in bars:
        height = bar.get_height()
        ax.text(bar.get_x() + bar.get_width()/2, bar.get_y() + height/2, f"{
                ha='center', va='center', color="white", fontsize=10)

# Show plot
plt.show()
```



Average MPCE for each state

In [151…
```
avg_MPCE_state = pd.read_csv('Cleaned Data/HCES/Avg_MPCE_State.csv',header=N
```

In [152…
```
avg_MPCE_state
```

Out[152…

	0	1	2	3
0	0	Andhra Pradesh	5327.0	7182.0
1	1	Arunachal Pradesh	5995.0	9832.0
2	2	Assam	3793.0	6794.0
3	3	Bihar	3670.0	5080.0
4	4	Chhattisgarh	2739.0	4927.0
5	5	Delhi	7400.0	8534.0
6	6	Goa	8048.0	9726.0
7	7	Gujarat	4116.0	7175.0
8	8	Haryana	5377.0	8428.0
9	9	Himachal Pradesh	5825.0	9223.0
10	10	Jharkhand	2946.0	5393.0
11	11	Karnataka	4903.0	8076.0
12	12	Kerala	6611.0	7783.0
13	13	Madhya Pradesh	3441.0	5538.0
14	14	Maharashtra	4145.0	7363.0
15	15	Manipur	4531.0	5945.0
16	16	Meghalaya	3852.0	7839.0
17	17	Mizoram	5963.0	8709.0
18	18	Nagaland	5155.0	8022.0
19	19	Odisha	3357.0	5825.0
20	20	Punjab	5817.0	7359.0
21	21	Rajasthan	4510.0	6574.0
22	22	Sikkim	9377.0	13927.0
23	23	Tamil Nadu	5701.0	8165.0
24	24	Telangana	5435.0	8978.0
25	25	Tripura	6259.0	8034.0
26	26	Uttar Pradesh	3481.0	5395.0
27	27	Uttarakhand	5003.0	7486.0
28	28	West Bengal	3620.0	5775.0
29	29	Andaman & N Islands	7771.0	10453.0
30	30	Chandigarh	8857.0	13425.0
31	31	Dadra & Nagar Haveli and Daman & Diu	4311.0	6837.0
32	32	Jammu & Kashmir	4774.0	6327.0

	0	1	2	3
33	33	Ladakh	5010.0	7533.0
34	34	Lakshadweep	6350.0	6377.0
35	35	Puducherry	7598.0	8637.0
36	36	All-India	4122.0	6996.0

In [153…
```
avg_MPCE_stateF = avg_MPCE_state[~avg_MPCE_state[1].str.contains('All-India'
```

In [154…
```
avg_MPCE_stateF.drop(0,axis=1,inplace=True)
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\1758681681.py:1: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  avg_MPCE_stateF.drop(0,axis=1,inplace=True)
```

In [155…
```
col = pd.MultiIndex.from_tuples([
    ('State/UT',''),
    ('Average MPCE(Rs.)','Rural'),('Average MPCE(Rs.)','Urban')
])
avg_MPCE_stateF.columns = col
```

In [156…
```
avg_MPCE_stateF
```

Out[156…

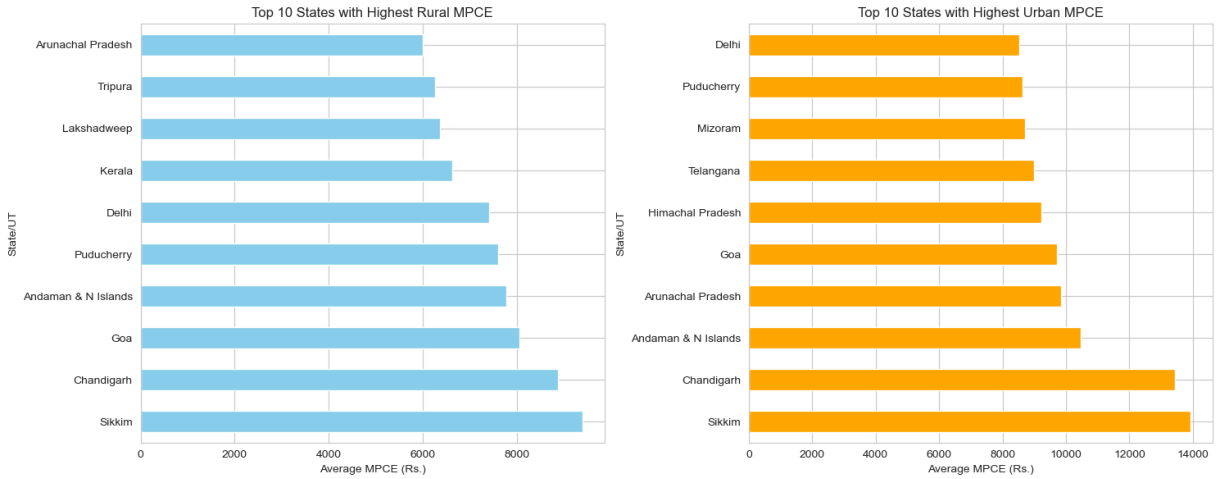
	State/UT	Average MPCE(Rs.)	
		Rural	Urban
0	Andhra Pradesh	5327.0	7182.0
1	Arunachal Pradesh	5995.0	9832.0
2	Assam	3793.0	6794.0
3	Bihar	3670.0	5080.0
4	Chhattisgarh	2739.0	4927.0
5	Delhi	7400.0	8534.0
6	Goa	8048.0	9726.0
7	Gujarat	4116.0	7175.0
8	Haryana	5377.0	8428.0
9	Himachal Pradesh	5825.0	9223.0
10	Jharkhand	2946.0	5393.0
11	Karnataka	4903.0	8076.0
12	Kerala	6611.0	7783.0
13	Madhya Pradesh	3441.0	5538.0
14	Maharashtra	4145.0	7363.0
15	Manipur	4531.0	5945.0
16	Meghalaya	3852.0	7839.0
17	Mizoram	5963.0	8709.0
18	Nagaland	5155.0	8022.0
19	Odisha	3357.0	5825.0
20	Punjab	5817.0	7359.0
21	Rajasthan	4510.0	6574.0
22	Sikkim	9377.0	13927.0
23	Tamil Nadu	5701.0	8165.0
24	Telangana	5435.0	8978.0
25	Tripura	6259.0	8034.0
26	Uttar Pradesh	3481.0	5395.0
27	Uttarakhand	5003.0	7486.0
28	West Bengal	3620.0	5775.0
29	Andaman & N Islands	7771.0	10453.0
30	Chandigarh	8857.0	13425.0
31	Dadra & Nagar Haveli and Daman & Diu	4311.0	6837.0

	State/UT	Average MPCE(Rs.)	
		Rural	Urban
32	Jammu & Kashmir	4774.0	6327.0
33	Ladakh	5010.0	7533.0
34	Lakshadweep	6350.0	6377.0
35	Puducherry	7598.0	8637.0

1. Top 10 States with highest MPCE (Rural & Urban)

In [157…

```
df_rural_top10 = avg_MPCE_stateF.nlargest(10,('Average MPCE(Rs.)','Rural'))
df_urban_top10 = avg_MPCE_stateF.nlargest(10, ('Average MPCE(Rs.)','Urban'))

fig, axes = plt.subplots(1, 2, figsize=(15, 6))

df_rural_top10.plot(kind='barh', x='State/UT', y=('Average MPCE(Rs.)','Rural
axes[0].set_xlabel("Average MPCE (Rs.)")
axes[0].set_title("Top 10 States with Highest Rural MPCE")

df_urban_top10.plot(kind='barh', x='State/UT', y=('Average MPCE(Rs.)','Urbar
axes[1].set_xlabel("Average MPCE (Rs.)")
axes[1].set_title("Top 10 States with Highest Urban MPCE")

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

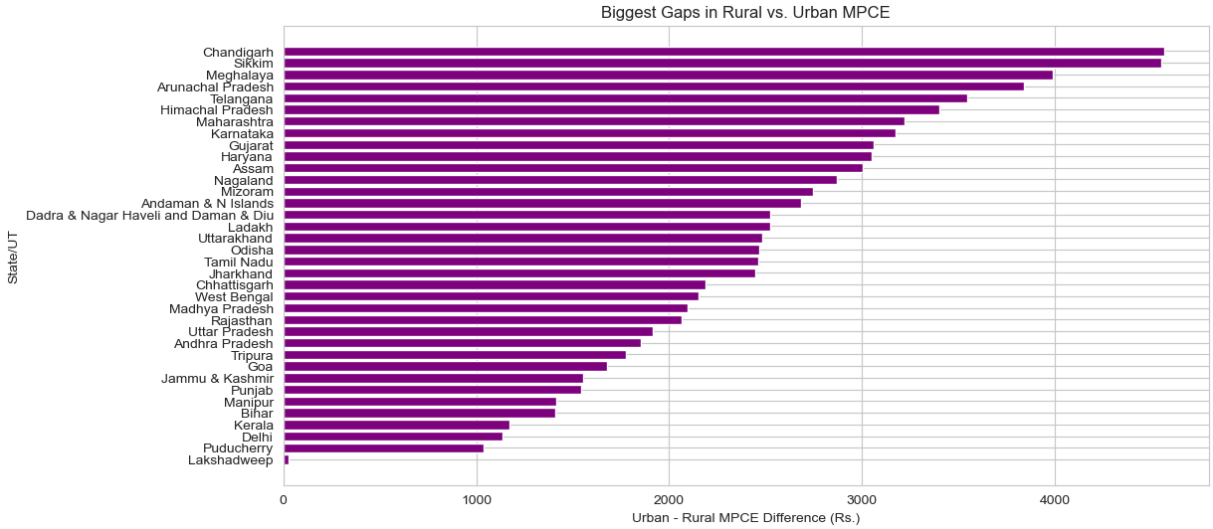


In [158…

```
avg_MPCE_stateF['MPCE Difference'] = avg_MPCE_stateF[('Average MPCE(Rs.)','U

avg_sorted = avg_MPCE_stateF.sort_values(by='MPCE Difference',ascending=Fals
plt.figure(figsize=(12, 6))
plt.barh(avg_sorted["State/UT"], avg_sorted["MPCE Difference"], color="purpl
plt.xlabel("Urban - Rural MPCE Difference (Rs.)")
plt.ylabel("State/UT")
plt.title("Biggest Gaps in Rural vs. Urban MPCE")
plt.gca().invert_yaxis()  # To show the largest difference at the top
```

```
plt.show()
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\2209780450.py:1: SettingWi
thCopyWarning:
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
  avg_MPCE_stateF['MPCE Difference'] = avg_MPCE_stateF[('Average MPCE(R
s.)','Urban')] - avg_MPCE_stateF[('Average MPCE(Rs.)','Rural')]
```



In [183…

```
import json
avg_MPCE_stateF["Total_MPCE"] = (avg_MPCE_stateF[('Average MPCE(Rs.)',"Rural

# Load the GeoJSON file for India states
with open("india_state_geo.json", "r") as f:
    india_states = json.load(f)

# Ensure state names in both data and GeoJSON match
avg_MPCE_stateF["State/UT"] = avg_MPCE_stateF["State/UT"].str.title()
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\171751136.py:2: SettingWit
hCopyWarning:


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

C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\171751136.py:9: SettingWit
hCopyWarning:


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
```

In [165…
```
avg_MPCE_stateF.head(2)
```

Out[165…

	State/UT	Average MPCE(Rs.)		MPCE Difference	Total_MPCE
		Rural	Urban		
0	Andhra Pradesh	5327.0	7182.0	1855.0	6254.5
1	Arunachal Pradesh	5995.0	9832.0	3837.0	7913.5

In [172…
```
avg_MPCE_stateF_copy = avg_MPCE_stateF.copy()
avg_MPCE_stateF_copy.columns = ['_'.join(col).strip() if isinstance(col, tup
avg_MPCE_stateF_copy.head(2)
```

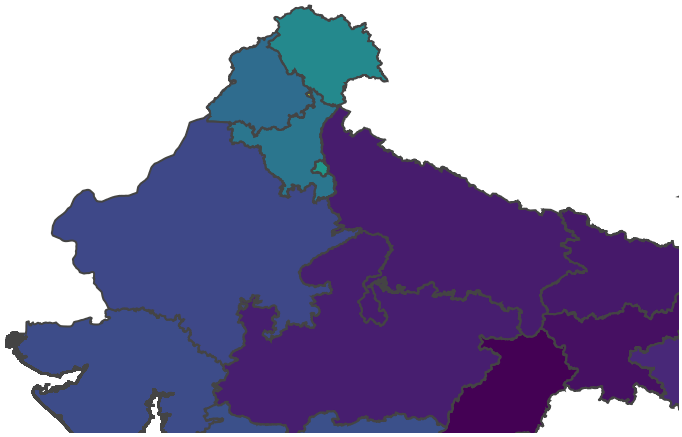
Out[172…

	State/UT_	Average MPCE(Rs.)_Rural	Average MPCE(Rs.)_Urban	MPCE Difference_	Total_MPCE_
0	Andhra Pradesh	5327.0	7182.0	1855.0	6254.5
1	Arunachal Pradesh	5995.0	9832.0	3837.0	7913.5

In [184…
```
import plotly.express as px
fig = px.choropleth(
    avg_MPCE_stateF_copy,
    geojson = india_states,
    locations = 'State/UT_',
    featureidkey = 'properties.NAME_1',
    color = 'Total_MPCE_',
    color_continuous_scale = px.colors.sequential.Viridis,
    range_color =[avg_MPCE_stateF_copy['Total_MPCE_'].min(), avg_MPCE_stateF
    title='State-wise MPCE in India'
)
```

```
fig.update_geos(fitbounds='locations',visible=False)
fig.update_layout(margin={'r':0, 't':30, 'l':0, 'b':0})
fig.update_layout(transition={'duration':0})
fig.show()
```

State-wise MPCE in India



Trend in Cereal and Food (HCES)

In [186…
```
trend_food = pd.read_csv('Cleaned Data/HCES/Trend_cereal_food.csv',header=Nc
trend_food
```

Out[186…

	0	1	2	3	4	5
0	0	2011-12	10.75	52.90	6.66	42.62
1	1	2022-23	4.91	46.38	3.64	39.17
2	2	2023-24	4.99	47.04	3.76	39.68

In [188…
```
trend_food = trend_food.drop(0,axis=1)
col = pd.MultiIndex.from_tuples([
    ('Period',''),
    ('Rural','%share of cereals in avg.'),('Rural','%share of food in avg.')
    ('Urban','%share of cereals in avg.'),('Urban','%share of food in avg.')
```

```
])
trend_food.columns = col
trend_food
```

Out[188…

	Period	Rural		Urban	
		%share of cereals in avg.	%share of food in avg.	%share of cereals in avg.	%share of food in avg.
0	2011-12	10.75	52.90	6.66	42.62
1	2022-23	4.91	46.38	3.64	39.17
2	2023-24	4.99	47.04	3.76	39.68

Insights:

- `Declining trend in cereal share:` The share of cereals in MPCE has consistently decreased from 2011-12 to 2023-24 in both rural (10.75% → 4.99%) and urban (6.66% → 3.76%) areas.
- `Declining share of food overall:` The total food share in MPCE has also reduced over time, indicating a possible shift in expenditure patterns towards non-food items.
- `Rural vs Urban Differences:`
  - Rural areas have always had a higher share of cereals in MPCE compared to urban areas.
  - The drop in cereal share is sharper in rural areas, possibly due to dietary diversification.
  - Urban areas show a more gradual decline in both cereals and food share.

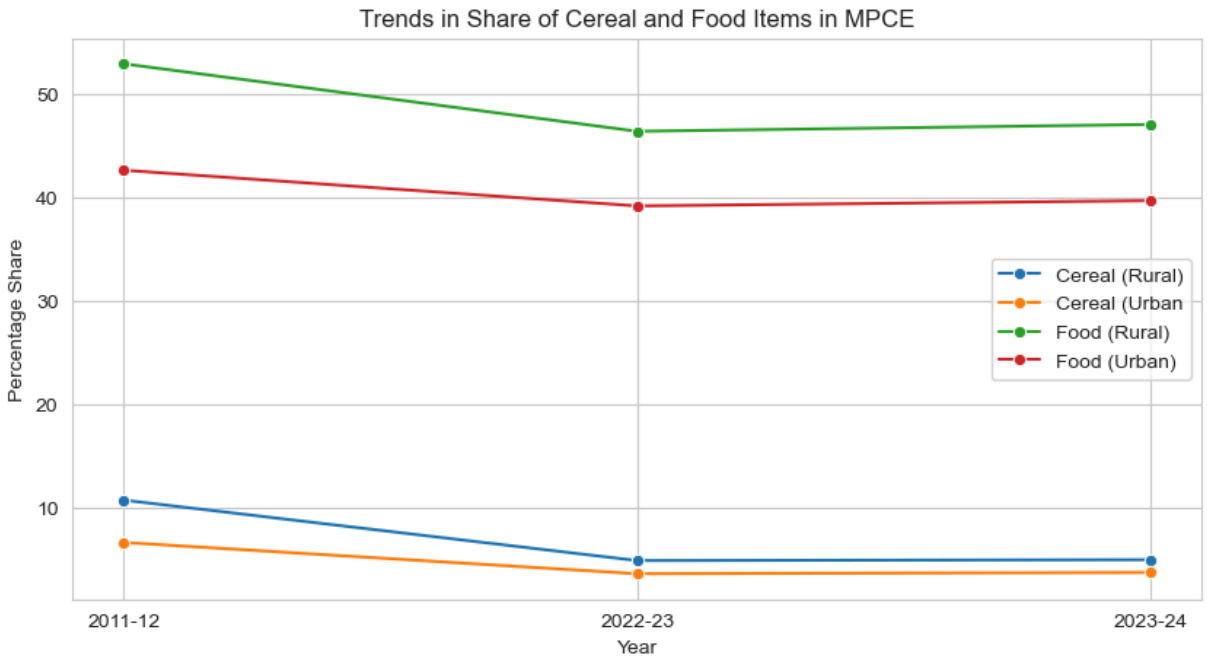
In [193…

```
trend_food.columns
trend_food_copy = trend_food.copy()
trend_food_copy.columns = ['Period','Rural_Cereal','Rural_Food','Urban_Cerea
```

Trend over time

In [197…

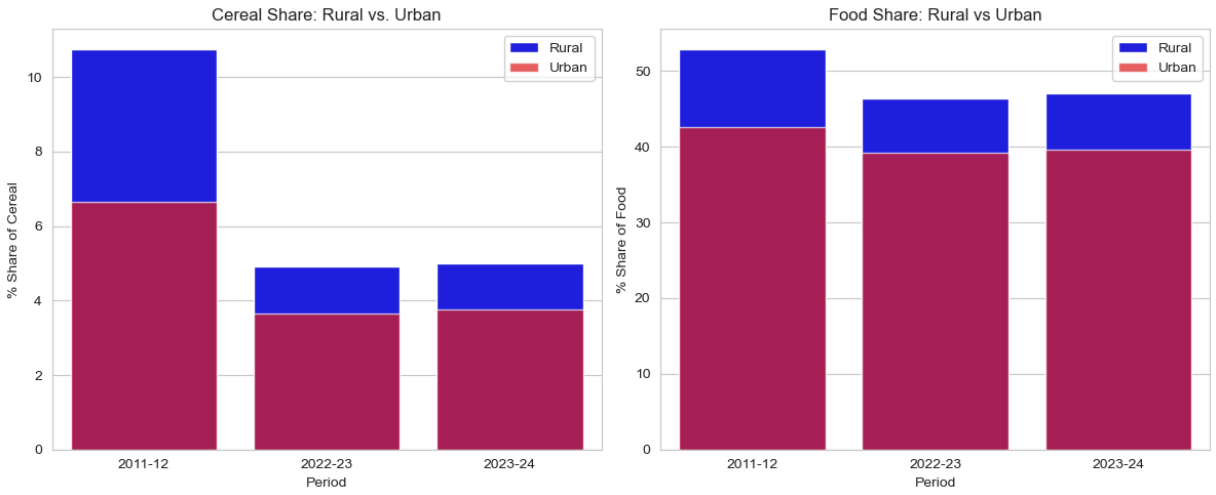
```
plt.figure(figsize=(10,5))
sns.lineplot(x='Period',y='Rural_Cereal',data=trend_food_copy,marker='o',lab
sns.lineplot(x='Period',y='Urban_Cereal',data=trend_food_copy,marker='o',lab
sns.lineplot(x='Period',y='Rural_Food',data=trend_food_copy,marker='o',label
sns.lineplot(x='Period',y='Urban_Food',data=trend_food_copy,marker='o',label
plt.xlabel('Year')
plt.ylabel('Percentage Share')
plt.title('Trends in Share of Cereal and Food Items in MPCE')
plt.legend()
plt.grid(True)
plt.show()
```



Rural vs. Urban Comparision (Bar chart)

In [200…

```
fig, ax = plt.subplots(1,2,figsize=(12,5))

sns.barplot(x='Period',y='Rural_Cereal',data=trend_food_copy, ax=ax[0],color
sns.barplot(x='Period',y='Urban_Cereal',data=trend_food_copy,ax=ax[0],color=
ax[0].set_title('Cereal Share: Rural vs. Urban')
ax[0].set_ylabel('% Share of Cereal')
ax[0].legend

sns.barplot(x='Period',y='Rural_Food',data=trend_food_copy,ax=ax[1],color='b
sns.barplot(x='Period',y='Urban_Food',data=trend_food_copy,ax=ax[1],color='r
ax[1].set_title('Food Share: Rural vs Urban')
ax[1].set_ylabel('% Share of Food')
ax[1].legend()

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



Percentage Decline Calculation

In [202…

```
def percentage_decline(initial, final):
    return round(((initial-final)/initial)*100,2)

cereal_rural_decline = percentage_decline(10.75,4.99)
cereal_urban_decline = percentage_decline(6.66, 3.76)
food_rural_decline = percentage_decline(52.90, 47.04)
food_urban_decline = percentage_decline(42.62, 39.68)

print(f"Percentage decline in cereal consumption (Rural): {cereal_rural_decl
print(f"Percentage decline in cereal consumption (Urban): {cereal_urban_decl
print(f"Percentage decline in food consumption (Rural): {food_rural_decline}
print(f"Percentage decline in food consumption (Urban): {food_urban_decline}
```

```
Percentage decline in cereal consumption (Rural): 53.58%
Percentage decline in cereal consumption (Urban): 43.54%
Percentage decline in food consumption (Rural): 11.08%
Percentage decline in food consumption (Urban): 6.9%
```

Trend in level of Consumption since 2011-12 All-India

In [205…

```
trend_consump = pd.read_csv('Cleaned Data/HCES/Trend_Consump.csv',header=Nor
trend_consump = trend_consump.drop(0,axis=1)
col = pd.MultiIndex.from_tuples([
    ('Sector',''),
    ('Average MPCE(Rs.) over different period','2011-12'),('Average MPCE(Rs.
])
trend_consump.columns = col
trend_consump
```

Out[205…

	Sector	Average MPCE(Rs.) over different period		
		2011-12	2022-23	2023-24
0	Rural	1430.0	3773.0	4122.0
1	Urban	2630.0	6459.0	6996.0
2	Difference as % of Rural MPCE	83.9	71.2	69.7

Insights:

1. Steady Increase in MPCE Across both sectors
   - The Rural MPCE has grown from ₹1430 in 2011-12 to ₹4122 in 2023-24, an almost 3× increase.
   - The Urban MPCE has grown from ₹2630 in 2011-12 to ₹6996 in 2023-24, also increasing almost 3×.
   - This indicates a steady rise in per capita spending, which may be attributed to factors like inflation, increased income, and improved living

standards.

2. Urban-Rural in Consumption
   - The absolute difference in MPCE between Urban and Rural areas has increased over time, but the relative difference (as % of Rural MPCE) has decreased:
     - 2011-12: Urban MPCE was 83.9% higher than Rural.
     - 2022-23: Gap reduced to 71.2%.
     - 2023-24: Further declined to 69.7%.
   - This suggests that rural consumption levels are catching up with urban areas, indicating economic improvement in rural regions.
3. Potential Factors behind the trends
   - Higher Economic Growth & Inflation: The increase in MPCE suggests that household expenditures have risen due to both economic growth and inflation.
   - Government Schemes & Rural Development: The narrowing Urban-Rural gap suggests that rural welfare programs and better income opportunities may have helped bridge the divide.
   - Lifestyle Changes: Increased spending could reflect a shift in consumption patterns, with more spending on non-essential goods, technology, and services.

In [215…
```
trend_consump.columns = ['_'.join(col).strip() for col in trend_consump.colu
```

In [219…
```
trend_consump_melt = trend_consump.melt(id_vars=['Sector_'],var_name='Year',
trend_consump_melt['Year'] = trend_consump_melt['Year'].str.extract(r'(\d{4}
trend_consump_melt['Year'] = trend_consump_melt['Year'].astype(str)
```

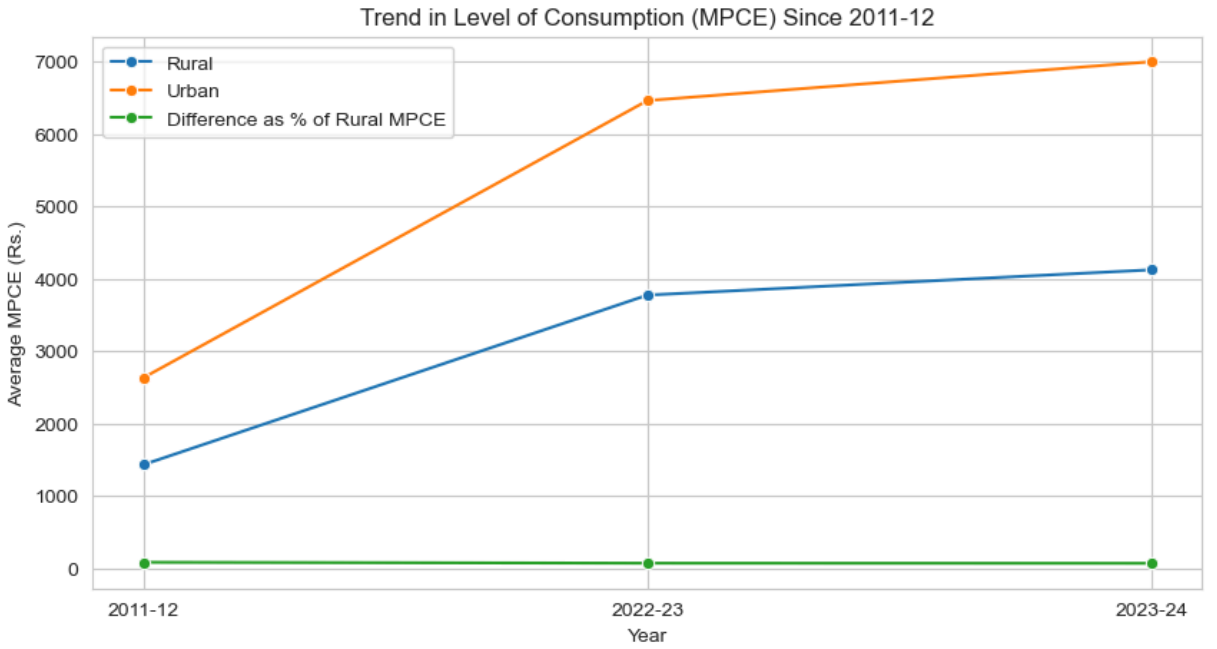
In [220…
```
trend_consump_melt
```

Out[220…

	Sector_	Year	MPCE
0	Rural	2011-12	1430.0
1	Urban	2011-12	2630.0
2	Difference as % of Rural MPCE	2011-12	83.9
3	Rural	2022-23	3773.0
4	Urban	2022-23	6459.0
5	Difference as % of Rural MPCE	2022-23	71.2
6	Rural	2023-24	4122.0
7	Urban	2023-24	6996.0
8	Difference as % of Rural MPCE	2023-24	69.7

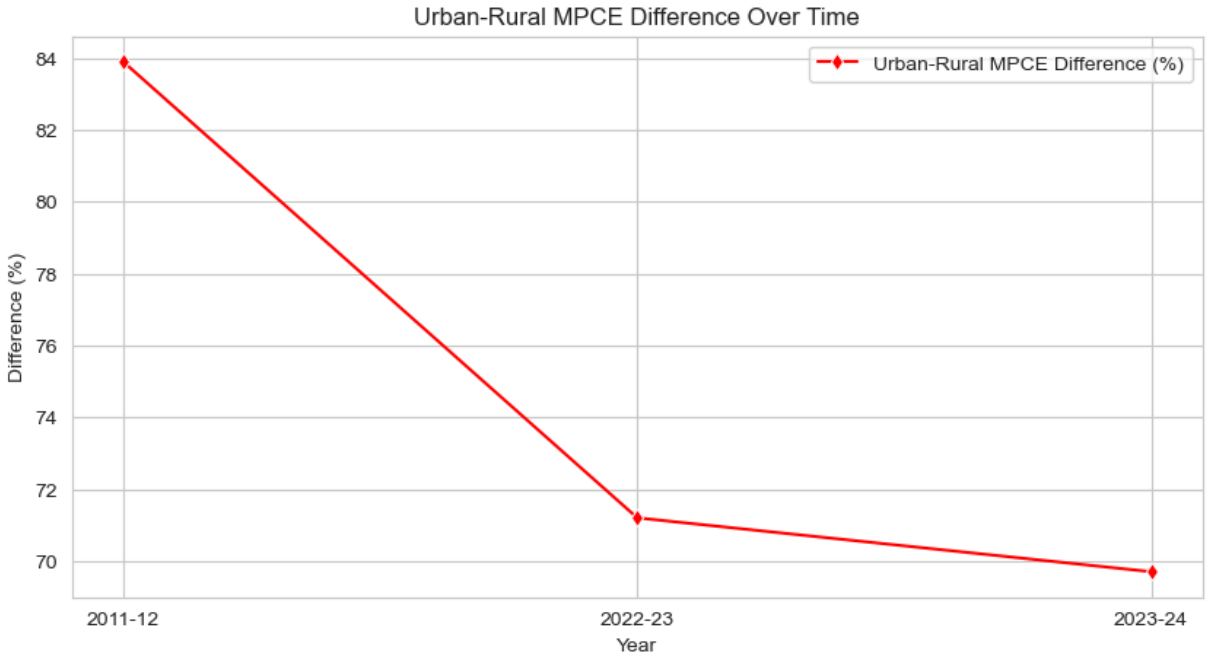
In [221…
```
plt.figure(figsize=(10,5))
sns.lineplot(x='Year',y='MPCE',hue='Sector_',data=trend_consump_melt,marker=
```

```
plt.xlabel('Year')
plt.ylabel('Average MPCE (Rs.)')
plt.title('Trend in Level of Consumption (MPCE) Since 2011-12')
plt.legend()
plt.grid(True)
plt.show()
```



In [227…

```
df_diff = trend_consump_melt[trend_consump_melt['Sector_'] == 'Difference as
plt.figure(figsize=(10,5))
sns.lineplot(x='Year',y='MPCE',data=df_diff,marker='d',color='red',label='Ur

plt.xlabel('Year')
plt.ylabel('Difference (%)')
plt.title('Urban-Rural MPCE Difference Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



In [288…
```
lfpr_res.to_csv('LFPS_res.csv',index=False,header=True)
```

Combined Datasets (PLFS+HCES)

Merging Datasets:

- Merging PLFS LFPR_res.csv with Avg MPCE_State: Analyze how LFPR correlates with household expenditure at the state level.

1. lfpr_res and avg_MPCE_stateF

In [289…
```
lfpr_res
```

Out[289…

	State/UT	Rural			Urban			Rur
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	63.0	33.6	48.6	57.2	22.4	39.3	61.2	29.9
1	Arunachal Pradesh	57.0	49.3	53.2	44.3	32.5	38.1	54.8	46.1
2	Assam	67.6	36.2	51.9	65.2	25.9	46.0	67.3	35.1
3	Bihar	55.5	15.3	35.4	42.1	10.7	27.6	54.1	14.8
4	Chhattisgarh	74.5	55.1	65.2	60.9	32.3	46.7	71.9	50.6
...	...	...	...	...	...	...	...	...	...
139	Dadra & Nagar Haveli & Daman & Diu	62.1	54.3	58.3	69.9	18.5	47.9	66.7	34.7
140	Jammu & Kashmir	55.7	41.9	49.0	58.2	25.8	42.5	56.2	38.8
141	Ladakh	55.5	46.4	51.3	61.0	26.3	45.0	56.2	43.7
142	Lakshadweep	73.3	14.5	45.7	56.3	12.4	35.4	61.2	13.6
143	Puducherry	58.6	37.1	48.0	58.2	24.4	40.2	58.4	28.9

144 rows × 11 columns

In [292…
```
avg_MPCE_stateF.head(5)
```

Out[292…

	State/UT	Average MPCE(Rs.)		MPCE Difference	Total_MPCE
		Rural	Urban		
0	Andhra Pradesh	5327.0	7182.0	1855.0	6254.5
1	Arunachal Pradesh	5995.0	9832.0	3837.0	7913.5
2	Assam	3793.0	6794.0	3001.0	5293.5
3	Bihar	3670.0	5080.0	1410.0	4375.0
4	Chhattisgarh	2739.0	4927.0	2188.0	3833.0

In [303…
```
final_lfpr = lfpr_res[lfpr_res['Age Category'] == 'All Ages']
final_lfpr.reset_index(drop=True,inplace=True)
```

In [304…
```
final_lfpr = final_lfpr.drop(('Age Category',''),axis=1)
```

In [305…
```
final_lfpr
```

Out[305…

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
0	Andhra Pradesh	59.7	40.7	50.1	59.3	24.5	41.3	59.5	35.8
1	Arunachal Pradesh	59.3	53.1	56.3	56.0	34.5	45.1	58.7	49.9
2	Assam	60.7	38.2	49.6	64.3	25.5	45.0	61.1	36.8
3	Bihar	48.1	21.1	34.9	48.2	12.0	30.9	48.1	20.3
4	Chhattisgarh	65.2	50.4	57.8	63.6	27.9	46.4	64.9	46.1
5	Delhi	54.7	13.8	37.8	54.0	14.5	35.9	54.0	14.5
6	Goa	58.1	21.8	40.5	54.9	23.7	39.5	56.3	22.9
7	Gujarat	62.8	44.3	53.8	62.0	23.6	43.7	62.5	35.8
8	Haryana	51.8	20.2	37.1	56.3	16.7	37.9	53.5	18.8
9	Himachal Pradesh	64.9	58.7	61.7	64.2	35.7	51.4	64.8	56.2
10	Jharkhand	53.1	40.4	46.8	52.1	15.0	34.0	52.9	35.8
11	Karnataka	60.0	34.6	47.3	59.8	23.6	42.1	59.9	30.5
12	Kerala	59.7	36.2	47.3	58.4	30.3	43.3	59.1	33.4
13	Madhya Pradesh	63.2	45.4	54.6	58.7	22.3	40.7	62.0	39.4
14	Maharashtra	61.0	38.0	49.6	61.0	23.7	43.1	61.0	32.0
15	Manipur	52.8	36.5	44.6	52.4	35.7	43.9	52.7	36.3
16	Meghalaya	54.8	48.7	51.7	53.1	37.8	45.1	54.6	47.1
17	Mizoram	51.1	30.5	41.0	49.2	30.2	39.6	50.2	30.4
18	Nagaland	57.3	45.2	51.1	52.0	35.8	44.1	55.7	42.7
19	Odisha	61.9	40.3	50.8	59.1	24.4	42.0	61.5	38.0
20	Punjab	62.4	27.7	45.2	62.6	19.0	41.4	62.5	24.4
21	Rajasthan	56.4	43.0	49.6	56.8	23.5	40.9	56.5	38.0
22	Sikkim	67.1	65.5	66.4	61.1	26.2	46.0	65.7	56.9
23	Tamil Nadu	60.4	44.5	52.3	59.2	24.4	41.3	59.8	35.2
24	Telangana	61.4	44.0	52.4	57.5	24.5	41.3	59.8	36.5
25	Tripura	64.8	39.1	51.9	57.3	25.9	41.3	63.5	36.8
26	Uttarakhand	56.6	41.8	49.3	55.6	18.6	37.2	56.4	35.9
27	Uttar Pradesh	54.7	28.1	41.5	56.9	13.5	36.2	55.2	25.2
28	West Bengal	63.6	33.7	48.6	64.3	26.8	45.4	63.8	31.7

	State/UT	Rural			Urban			Rura	
		Male	Female	Person	Male	Female	Person	Male	Female
29	Andaman & N. Island	67.5	44.6	56.5	67.9	30.0	49.3	67.7	38.0
30	Chandigarh	0.0	0.0	0.0	60.1	25.0	43.5	60.1	25.0
31	Dadra & Nagar Haveli & Daman & Diu	62.1	54.3	58.3	69.9	18.5	47.9	66.7	34.7
32	Jammu & Kashmir	55.7	41.9	49.0	58.2	25.8	42.5	56.2	38.8
33	Ladakh	55.5	46.4	51.3	61.0	26.3	45.0	56.2	43.7
34	Lakshadweep	73.3	14.5	45.7	56.3	12.4	35.4	61.2	13.0
35	Puducherry	58.6	37.1	48.0	58.2	24.4	40.2	58.4	28.9

In [306…
```
merged_df1 = pd.merge(final_lfpr,avg_MPCE_stateF,on='State/UT',how='inner')
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\889140574.py:1: PerformanceWarning:

dropping on a non-lexsorted multi-index without a level parameter may impact performance.
```

In [308…
```
merged_df1.isnull().sum()
```

Out[308…
```
State/UT                  0
Rural              Male   0
                   Female 0
                   Person 0
Urban              Male   0
                   Female 0
                   Person 0
Rural+Urban        Male   0
                   Female 0
                   Person 0
Average MPCE(Rs.)  Rural  0
                   Urban  0
MPCE Difference           0
Total_MPCE                0
dtype: int64
```

In [310…
```
merged_df1.corr(numeric_only=True)
```

Out[310…

		Rural			U		
		Male	Female	Person	Male	Female	Pe
Rural	Male	1.000000	0.550619	0.877186	0.210371	0.019344	0.11
	Female	0.550619	1.000000	0.881337	0.273687	0.545457	0.57
	Person	0.877186	0.881337	1.000000	0.268362	0.315764	0.38
Urban	Male	0.210371	0.273687	0.268362	1.000000	0.113226	0.63
	Female	0.019344	0.545457	0.315764	0.113226	1.000000	0.82
	Person	0.110738	0.579378	0.389932	0.632933	0.821906	1.00
Rural+Urban	Male	0.350964	0.366930	0.400110	0.859434	0.187484	0.61
	Female	0.240597	0.915576	0.659114	0.296608	0.710511	0.73
	Person	0.307570	0.859653	0.665304	0.548655	0.625684	0.81
Average MPCE(Rs.)	Rural	-0.260472	-0.235045	-0.267774	-0.008028	0.059724	0.07
	Urban	-0.377915	-0.010845	-0.206207	0.098788	0.261399	0.31
MPCE Difference		-0.326858	0.354403	0.022508	0.207117	0.418617	0.50
Total_MPCE		-0.336915	-0.115362	-0.242193	0.052719	0.177102	0.21

To Analyze the relationship between employment trends and consumption patterns across different states in India

In [336…

```
merged_df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34 entries, 0 to 33
Data columns (total 14 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   State/UT_                34 non-null     object
 1   Rural_Male               34 non-null     float64
 2   Rural_Female             34 non-null     float64
 3   Rural_Person             34 non-null     float64
 4   Urban_Male               34 non-null     float64
 5   Urban_Female             34 non-null     float64
 6   Urban_Person             34 non-null     float64
 7   Rural+Urban_Male         34 non-null     float64
 8   Rural+Urban_Female       34 non-null     float64
 9   Rural+Urban_Person       34 non-null     float64
 10  Average MPCE(Rs.)_Rural  34 non-null     float64
 11  Average MPCE(Rs.)_Urban  34 non-null     float64
 12  MPCE Difference_         34 non-null     float64
 13  Total_MPCE_              34 non-null     float64
dtypes: float64(13), object(1)
memory usage: 3.8+ KB
```

In [333…

```
merged_df1.columns = ['_'.join(col).strip() if isinstance(col, tuple) else c
merged_df1.head(2)
```

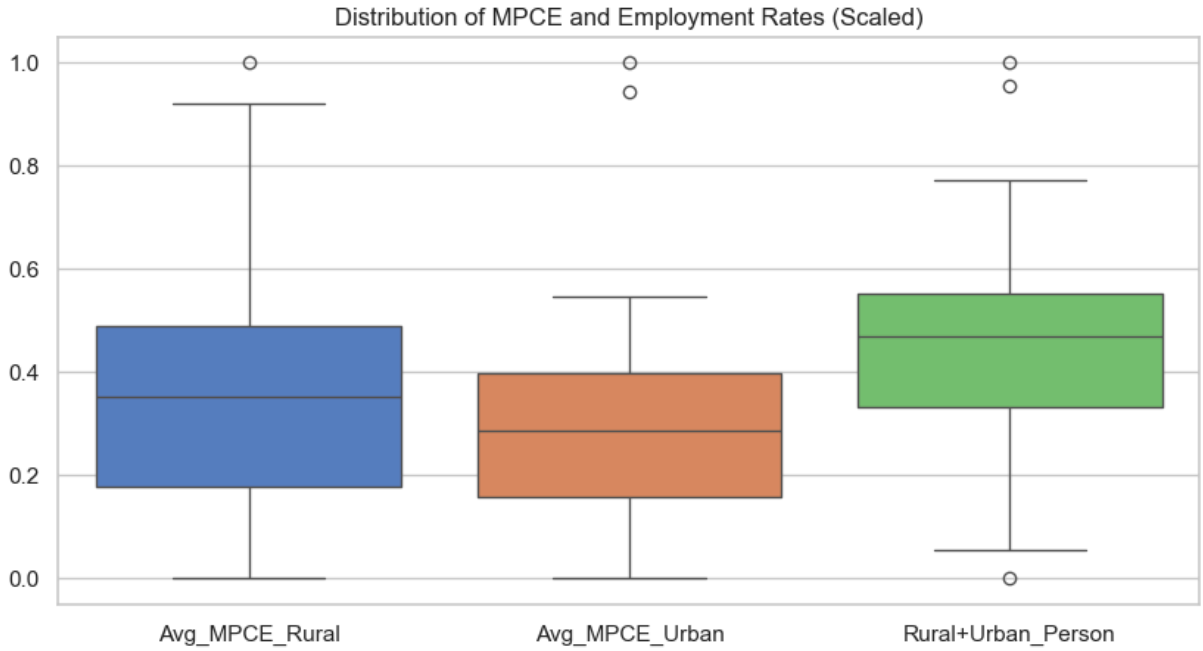
Out[333…

	State/UT_	Rural_Male	Rural_Female	Rural_Person	Urban_Male	Urban_Fer
0	Andhra Pradesh	59.7	40.7	50.1	59.3	
1	Arunachal Pradesh	59.3	53.1	56.3	56.0	

Distribution of MPCE and Employment Rates (Scaled) Boxplot Analysis

In [349…

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(merged_df1[['Average MPCE(Rs.)_Rural', 'A

scaled_df = merged_df1.copy()
scaled_df[['Avg_MPCE_Rural', 'Avg_MPCE_Urban', 'Rural+Urban_Person']] = scal

# Plot after scaling
plt.figure(figsize=(10,5))
sns.boxplot(data=scaled_df[['Avg_MPCE_Rural', 'Avg_MPCE_Urban', 'Rural+Urbar
plt.title('Distribution of MPCE and Employment Rates (Scaled)')
plt.show()
```

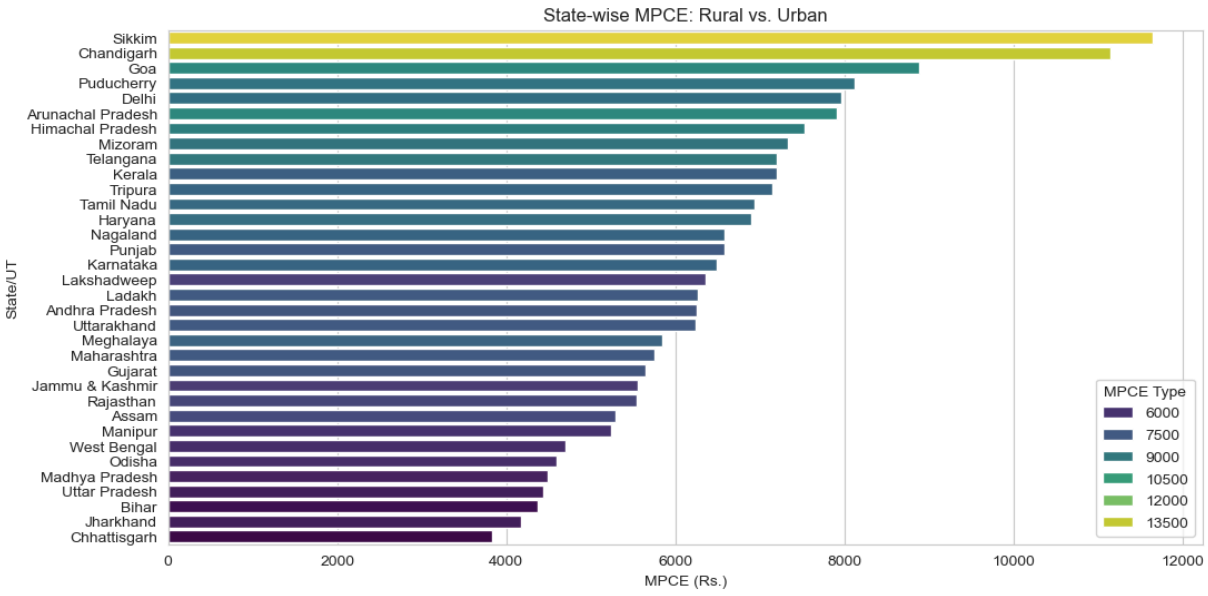


Conclusions:

-

In [315…

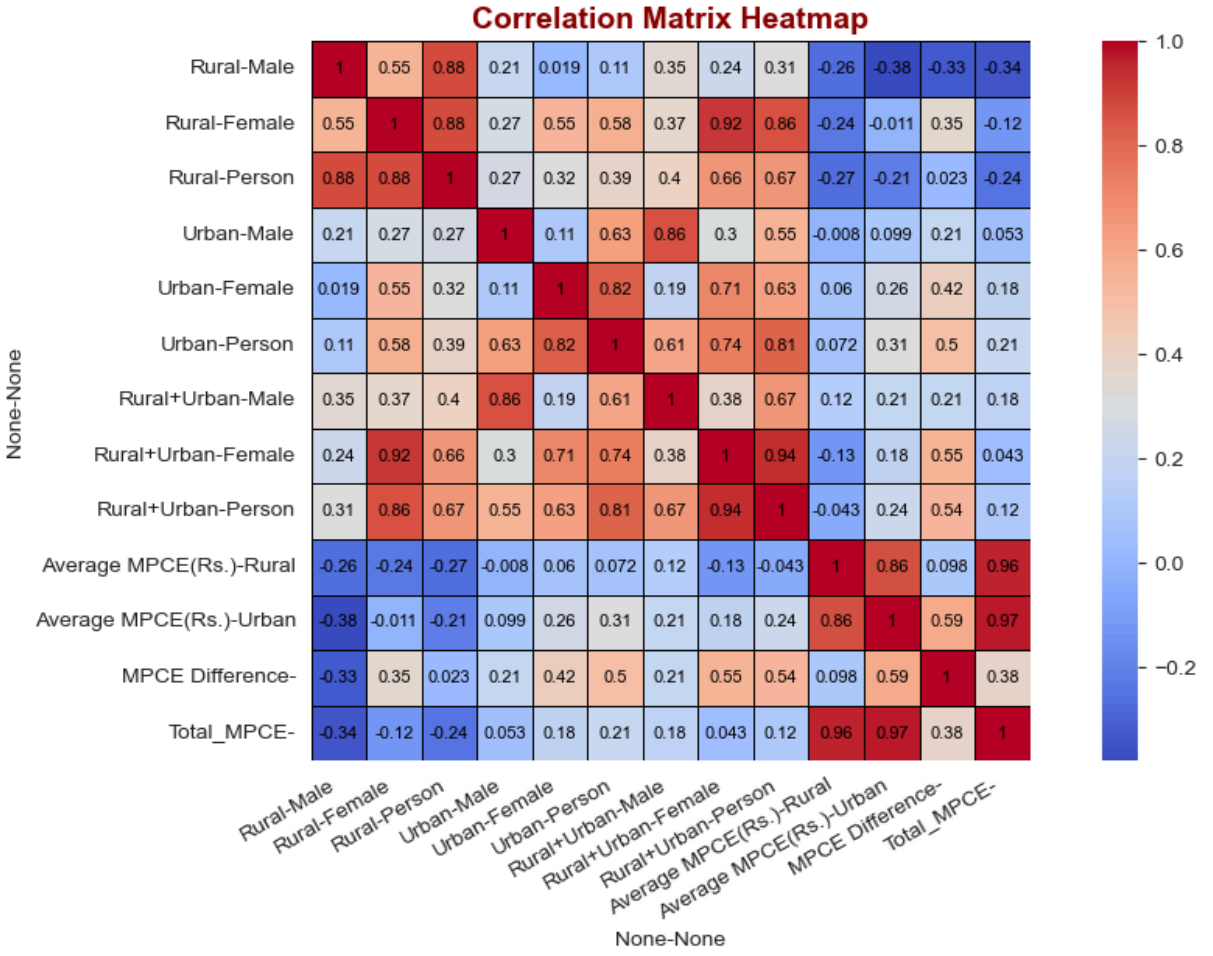
```
plt.figure(figsize=(12, 6))
df_sorted = merged_df1.sort_values(by="Total_MPCE", ascending=False)
sns.barplot(data=df_sorted, x="Total_MPCE", y="State/UT", hue=('Average MPCE
plt.xlabel("MPCE (Rs.)")
plt.ylabel("State/UT")
plt.title("State-wise MPCE: Rural vs. Urban")
plt.legend(title="MPCE Type")
plt.show()
```



Correlation Analysis

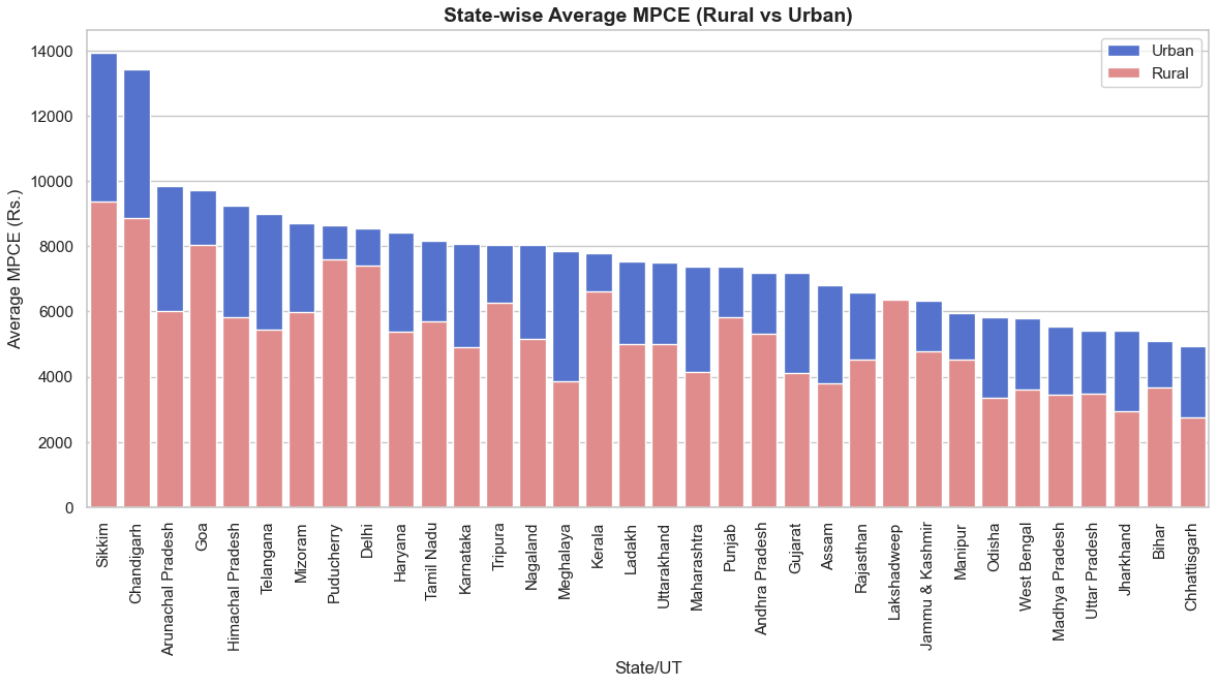
In [330…

```
plt.figure(figsize=(12,6))
sns.heatmap(merged_df1.corr(numeric_only=True),annot=True,cmap='coolwarm',li
plt.title('Correlation Matrix Heatmap',fontsize=14,fontweight='bold',color='
plt.xticks(rotation=30,ha='right',fontsize=10)
plt.show()
```



In [351…

```
# 1️⃣ State-wise MPCE Analysis (Bar Plot)
plt.figure(figsize=(14, 6))
statewise_data = merged_df1[['State/UT_', 'Average MPCE(Rs.)_Rural', 'Averag

sns.barplot(data=statewise_data, x='State/UT_', y='Average MPCE(Rs.)_Urban',
sns.barplot(data=statewise_data, x='State/UT_', y='Average MPCE(Rs.)_Rural',

plt.xticks(rotation=90)
plt.title('State-wise Average MPCE (Rural vs Urban)', fontsize=14, fontweigh
plt.ylabel('Average MPCE (Rs.)')
plt.xlabel('State/UT')
plt.legend()
plt.show()
```



In [357…

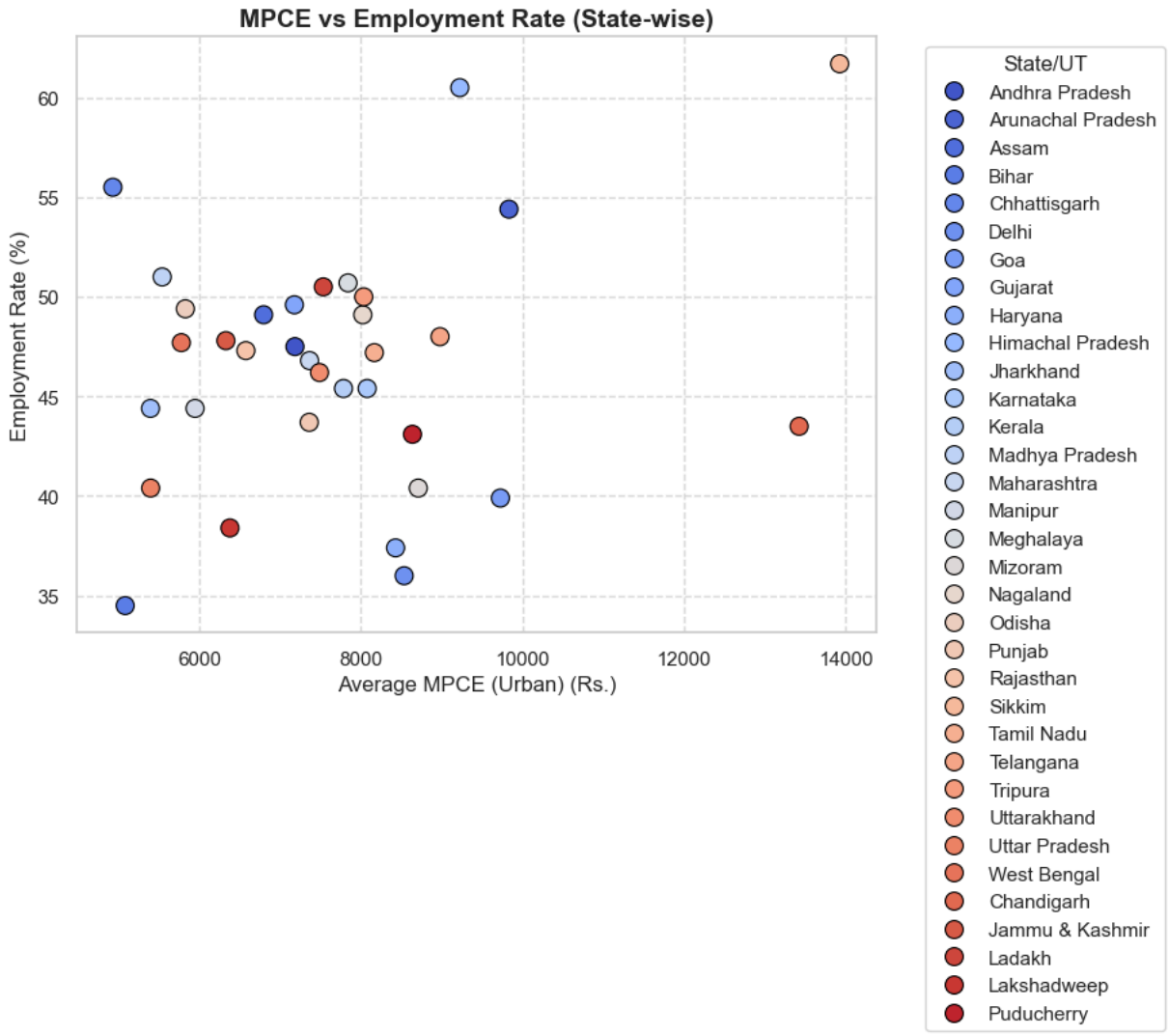
```
top_states = merged_df1.nlargest(5, 'Average MPCE(Rs.)_Urban')[['State/UT_',
bottom_states = merged_df1.nsmallest(5, 'Average MPCE(Rs.)_Urban')[['State/U

# Display results
print("🔝 Top 5 States with Highest MPCE (Urban):\n", top_states)
print("\n🔻 Bottom 5 States with Lowest MPCE (Urban):\n", bottom_states)
```

```
🔝 Top 5 States with Highest MPCE (Urban):
            State/UT_  Average MPCE(Rs.)_Urban
22             Sikkim                  13927.0
29         Chandigarh                  13425.0
1    Arunachal Pradesh                   9832.0
6                  Goa                   9726.0
9     Himachal Pradesh                   9223.0

🔻 Bottom 5 States with Lowest MPCE (Urban):
          State/UT_  Average MPCE(Rs.)_Urban
4      Chhattisgarh                  4927.0
3             Bihar                  5080.0
10        Jharkhand                  5393.0
27    Uttar Pradesh                  5395.0
13   Madhya Pradesh                  5538.0
```

MPCE vs Employment Rate (State-Wise)

In [354…

```
plt.figure(figsize=(8, 6))
sns.scatterplot(data=merged_df1, x='Average MPCE(Rs.)_Urban', y='Rural+Urbar

plt.title('MPCE vs Employment Rate (State-wise)', fontsize=14, fontweight='b
plt.xlabel('Average MPCE (Urban) (Rs.)')
plt.ylabel('Employment Rate (%)')
plt.grid(True, linestyle='--', alpha=0.7)
plt.legend(title='State/UT',bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



In [ ]:
```
merged_df1.columns = ['_'.join(col).strip() if isinstance(col, tuple) else c
```

In [423…
```
unemp_rate_f.columns = ['_'.join(col).strip() if isinstance(col, tuple) else
```

In [422…
```
trend_consump
```

Out[422…

	Sector_	Average MPCE(Rs.) over different period_2011-12	Average MPCE(Rs.) over different period_2022-23	Average MPCE(Rs.) over different period_2023-24
0	Rural	1430.0	3773.0	4122.0
1	Urban	2630.0	6459.0	6996.0
2	Difference as % of Rural MPCE	83.9	71.2	69.7

In [425…
```
trend_consump_long = trend_consump.melt(id_vars=['Sector_'], var_name='Year'
trend_consump_long['Year'] = trend_consump_long['Year'].str.extract('(\d{4}-
```

```
<>:2: SyntaxWarning:

invalid escape sequence '\d'

<>:2: SyntaxWarning:

invalid escape sequence '\d'

C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\925743468.py:2: SyntaxWarn
ing:

invalid escape sequence '\d'
```

In [426…
```
trend_consump_long
```

Out[426…

	Sector_	Year	MPCE
0	Rural	2011-12	1430.0
1	Urban	2011-12	2630.0
2	Difference as % of Rural MPCE	2011-12	83.9
3	Rural	2022-23	3773.0
4	Urban	2022-23	6459.0
5	Difference as % of Rural MPCE	2022-23	71.2
6	Rural	2023-24	4122.0
7	Urban	2023-24	6996.0
8	Difference as % of Rural MPCE	2023-24	69.7

Unemployment Analysis

1. Gender Disparity

In [427…
```
unemp_rate_f.head(1)
```

Out[427…

	State/UT_	Rural_Male	Rural_Female	Rural_Person	Urban_Male	Urban_Fer
1	Andhra Pradesh	4.8	3.0	4.1	5.7	

In [428…
```
gender_gap = unemp_rate_f[['State/UT_','Rural+Urban_Female','Rural+Urban_Mal
gender_gap['Gap'] = gender_gap['Rural+Urban_Female'] - gender_gap['Rural+Urb
gender_gap.sort_values('Gap', ascending=False)
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\872014814.py:2: SettingWit
hCopyWarning:


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
```

Out[428…

	State/UT_	Rural+Urban_Female	Rural+Urban_Male	Gap
35	Lakshadweep	36.2	12.3	23.9
30	Andaman & N. Island	25.9	8.6	17.3
33	Jammu & Kashmir	20.9	5.9	15.0
31	Chandigarh	16.3	4.4	11.9
7	Goa	16.8	5.6	11.2
13	Kerala	16.7	5.6	11.1
17	Meghalaya	13.5	4.7	8.8
34	Ladakh	10.4	2.6	7.8
10	Himachal Pradesh	13.1	7.2	5.9
3	Assam	10.3	4.9	5.4
36	Puducherry	7.9	4.4	3.5
21	Punjab	8.2	5.0	3.2
16	Manipur	7.8	5.1	2.7
29	West Bengal	5.9	3.2	2.7
24	Tamil Nadu	6.3	4.1	2.2
27	Uttarakhand	6.6	5.4	1.2
19	Nagaland	8.2	7.3	0.9
28	Uttar Pradesh	4.5	3.9	0.6
2	Arunachal Pradesh	7.8	7.2	0.6
14	Madhya Pradesh	2.1	1.6	0.5
22	Rajasthan	5.4	5.0	0.4
18	Mizoram	2.4	2.2	0.2
5	Chhattisgarh	3.0	2.9	0.1
8	Gujarat	1.4	1.4	0.0
25	Telangana	5.4	5.5	-0.1
26	Tripura	1.7	1.8	-0.1
23	Sikkim	3.8	4.0	-0.2
6	Delhi	1.5	2.3	-0.8
20	Odisha	4.5	5.4	-0.9
15	Maharashtra	2.8	3.9	-1.1
4	Bihar	2.9	4.1	-1.2
1	Andhra Pradesh	3.8	5.1	-1.3
12	Karnataka	2.0	3.3	-1.3

	State/UT_	Rural+Urban_Female	Rural+Urban_Male	Gap
11	Jharkhand	1.0	2.5	-1.5
9	Haryana	2.4	4.0	-1.6
32	Dadra & Nagar Haveli & Daman & Diu	1.2	3.3	-2.1

Insight: States like Kerala, Goa and Chandigarh show high female unemployment rates (>15%) compared to males

2. Urban-Rural Divide

In [430…

```
unemp_rate_f['Urban_Rural_Gap'] = unemp_rate_f['Urban_Person'] - unemp_rate_
unemp_rate_f.sort_values('Urban_Rural_Gap',ascending=False)
```

```
C:\Users\Aabhas\AppData\Local\Temp\ipykernel_7124\4236042522.py:1: SettingWi
thCopyWarning:


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
```

Out[430…

	State/UT_	Rural_Male	Rural_Female	Rural_Person	Urban_Male	Urban_
34	Ladakh	2.1	7.8	4.2	5.7	
31	Chandigarh	0.0	0.0	0.0	4.4	
2	Arunachal Pradesh	6.7	6.2	6.5	9.8	
17	Meghalaya	4.0	12.3	7.7	9.2	
5	Chhattisgarh	2.0	1.8	1.9	6.9	
19	Nagaland	5.7	6.7	6.1	11.6	
11	Jharkhand	1.5	0.1	1.1	6.9	
36	Puducherry	0.3	5.4	2.1	7.0	
4	Bihar	3.7	1.9	3.4	7.8	
22	Rajasthan	4.2	3.8	4.1	7.0	
33	Jammu & Kashmir	6.0	17.2	8.8	5.3	
28	Uttar Pradesh	3.4	2.8	3.2	5.9	
20	Odisha	5.1	3.5	4.6	6.9	
25	Telangana	4.8	3.4	4.2	6.4	
3	Assam	4.7	9.6	6.1	7.0	
15	Maharashtra	3.1	1.2	2.4	5.0	
12	Karnataka	2.7	1.0	2.1	4.3	
14	Madhya Pradesh	1.1	1.5	1.2	3.4	
8	Gujarat	0.7	0.3	0.6	2.4	
16	Manipur	4.8	6.7	5.6	5.8	
18	Mizoram	1.5	1.0	1.3	3.2	
1	Andhra Pradesh	4.8	3.0	4.1	5.7	
26	Tripura	1.6	1.3	1.5	2.7	
30	Andaman & N. Island	9.4	21.2	13.5	7.5	
27	Uttarakhand	6.1	4.6	5.6	3.6	
9	Haryana	3.9	1.7	3.4	4.2	
29	West Bengal	3.1	5.1	3.6	3.2	
21	Punjab	5.1	7.7	5.8	4.8	
10	Himachal Pradesh	7.6	12.5	9.9	4.8	

	State/UT_	Rural_Male	Rural_Female	Rural_Person	Urban_Male	Urban_
24	Tamil Nadu	4.5	5.9	5.1	3.6	
13	Kerala	5.6	17.4	9.7	5.7	
32	Dadra & Nagar Haveli & Daman & Diu	5.4	0.0	3.1	2.1	
23	Sikkim	4.6	3.5	4.1	1.8	
7	Goa	8.0	14.1	9.5	3.8	
35	Lakshadweep	12.8	53.6	18.4	12.1	
6	Delhi	3.6	19.5	6.0	2.2	

Insight: Urban unemployment is higher in states like Ladakh (9.6%) while rural unemployment dominates in Punjab (equal rates(0))

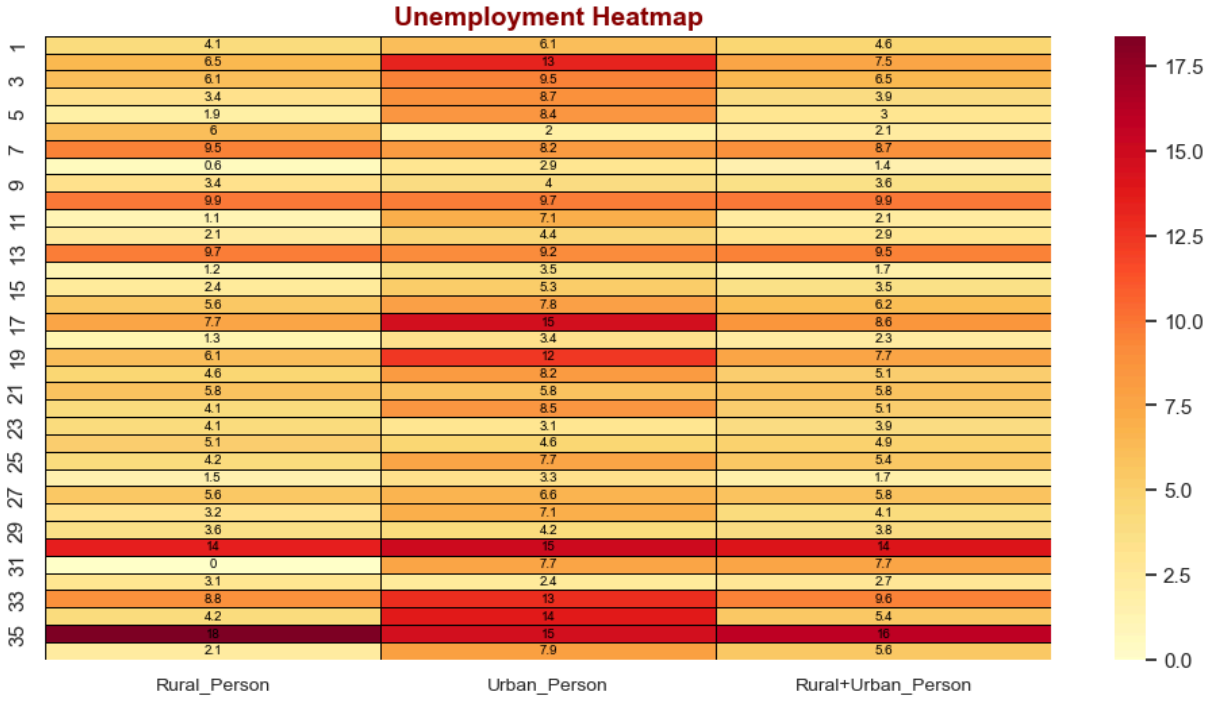
MPCE (Consumption) Analysis

1. Rural vs Urban MPCE Growth

Unemployment Heatmap

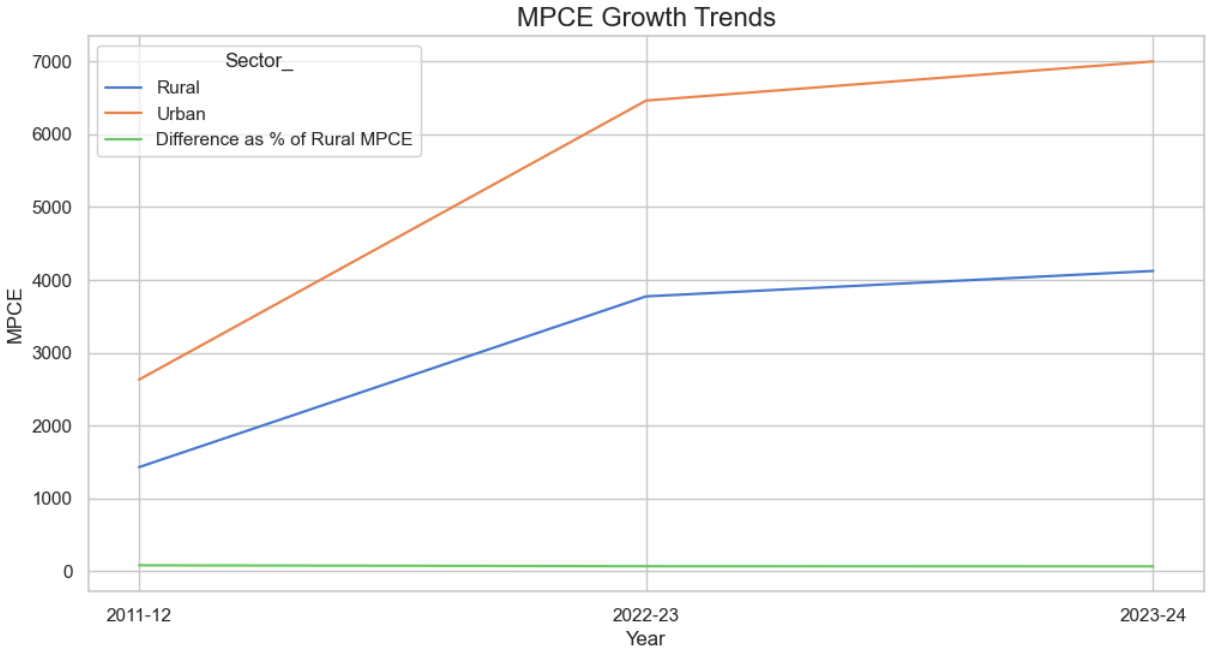
In [440…

```
plt.figure(figsize=(12,6))
sns.heatmap(unemp_rate_f[['Rural_Person', 'Urban_Person', 'Rural+Urban_Persc
plt.title('Unemployment Heatmap',fontsize=14,fontweight='bold',color='darkre
plt.xticks(rotation=0,fontsize=10)
plt.show()
```



In [443…

```
plt.figure(figsize=(12,6))
sns.lineplot(data=trend_consump_long, x='Year', y='MPCE', hue='Sector_')
plt.title('MPCE Growth Trends',fontsize=16)
plt.show()
```



Combined Insights:

1. **Gender & Sectoral Gaps**:
   - High female unemployment correlates with lower MPCE growth in states like Kerala (16.7% female unemployment).

- Urban areas show higher MPCE but also higher unemployment volatility (e.g., Chandigarh: 16.3% female urban unemployment).

2. **Policy Implications**:
   - **Rural Focus**: Boost employment schemes (e.g., MGNREGA) to align with MPCE growth.
   - **Urban Focus**: Address female unemployment (e.g., skill development) to stabilize consumption.

In [444…

```
%who_ls DataFrame |
```

Out[444…

```
['ab_perc_mpce',
 'ab_perc_mpceA',
 'ab_perc_mpceF',
 'all_',
 'all_india',
 'avg_MPCE_state',
 'avg_MPCE_stateF',
 'avg_MPCE_stateF_copy',
 'avg_mpce',
 'avg_sorted',
 'bottom_casual_emp',
 'bottom_self_emp',
 'bottom_self_emp_states',
 'bottom_states',
 'bottom_wage_emp',
 'corr_matrix',
 'cpi',
 'd',
 'df',
 'df_diff',
 'df_rural_top10',
 'df_selected',
 'df_sorted',
 'df_urban_top10',
 'df_vis',
 'emp',
 'emp_1',
 'emp_grp',
 'emp_industry',
 'emp_state',
 'emp_statewise',
 'employment_types',
 'final_emp',
 'final_lfpr',
 'gen',
 'gen_index_cpi',
 'gender_gap',
 'gender_unemp',
 'gender_unemp_melted',
 'grp',
 'grp_melted',
 'grp_t',
 'heatmap_data',
 'high_casual_labour',
 'high_female_unemp_states',
 'industry_grouped',
 'industry_grp',
 'lfpr_15_29',
 'lfpr_15_29_f',
 'lfpr_15_59',
 'lfpr_15_59_f',
 'lfpr_15_above',
 'lfpr_15_above_f',
 'lfpr_all_ages',
 'lfpr_all_ages_f',
 'lfpr_res',
```

```
'merged_df1',
'outliers',
'reg',
'rural_unemp',
'scaled_df',
'statewise_data',
'top5_rural',
'top5_urban',
'top_casual_emp',
'top_casual_emp_melted',
'top_self_emp',
'top_self_emp_melted',
'top_self_emp_states',
'top_states',
'top_wage_emp',
'top_wage_emp_melted',
'total',
'trend_consump',
'trend_consump_long',
'trend_consump_melt',
'trend_food',
'trend_food_copy',
'unemp_data',
'unemp_melted',
'unemp_rate',
'unemp_rate_f',
'unemp_rate_sorted',
'wpr',
'wpr_all',
'wpr_f',
'wpr_f_sorted']
```

In [ ]: