TAMILNADU MARGINAL WORKERS ASSESSMENT

Data Analytics with cognos – Phase 5

DOCUMENTATION

Team Members:

1.KARTHIKEYAN N (211521243081)

2.HARI KRISHNAMOORTHY M (211521243061)

3.ANJEEVARAGAVAN R (211521243014)

4.BALAJI V (211521243029)

5.KARTHIKEYAN P (211521243082)

Phase 5: Documentation

Problem Objectives:

The project involves analyzing the demographic characteristics of marginal workers in Tamil Nadu based on their age, industrial category, and sex. The objective is to perform a socioeconomic analysis and create visualizations to represent the distribution of marginal workers across different categories. This project includes defining objectives, designing the analysis approach, selecting appropriate visualization types, and performing the analysis using Python and data visualization libraries.

Consider conducting clustering analysis to identify patterns among different industrial categories and age groups.

Start the data analysis by loading and preprocessing the dataset. Load the dataset using python and data manipulation libraries (e.g., pandas).

Perform the demographic analysis and create visualizations.

Calculate the distribution of marginal workers based on age, industrial category, and sex using data aggregation and manipulation.

Create visualizations using data visualization libraries (e.g., Matplotlib, Seaborn).

Dataset Link:

https://tn.data.gov.in/resource/marginal-workers-classified-age-industrial-category-and-sex-scheduled-caste-2011-tamil

WE ARE GOING TO SEE IN THIS DOCUMENTATION ARE:

1.PROJECT OBJECTIVES

2.ANALYSIS

3.VISUALIZATION

Project objectives:

Define objectives such as analyzing marginal worker demographics, understanding age and gender distribution, and exploring industrial categories.

Analysis Approach:

Plan the steps to extract, clean, and analyze the dataset to derive insights.

Visualization Selection:

Determine suitable visualization types (e.g., bar charts, pie charts, heatmaps) to represent demographic distributions effectively.

Design Thinking:

Project Objectives:

Firstly, we are going to explore the analyzing Marginal Workers Demographics and detail observation of Dataset.

Marginal Workers Demographics:

According to the dataset, It consists of code for state, district and area name that shows the details as below, Table Code:

B0806SC

State Code:

`33 – Tamil Nadu

District Code:

000 – Tamil Nadu

602 – Thiruvallur

603 – Chennai

604 – Kancheepuram

605 – Vellore

606 – Tiruvannamalai

607 – Villupuram

608 – Salem

609 – Namakkal

610 – Erode

611 – The Nilgiris

612 – Dindigul

613 – Karur

614 – Tiruchirappalli

615 – Perambalur

616 – Ariyalur

617 – Cuddalore

618 – Nagapattinam

619 – Tiruvarur

620 – Thanjavur

621 – Pudukkottai

622 – Sivaganga

623 – Madurai

624 – Theni

625 – Virudhunagar

626 – Ramanathapuram

627 – Thoothukkudi

628 – Tirunelveli

629 – Kanniyakumari

630 – Dharmapuri

631 – Krishnagiri

632 – Coimbatore

633 – Tiruppur

The above mentioned are the code with their respective locations.

Total/Rural/Urban:

This column in the data set shows the type of location that inside the respective districts, here Total represents the sum of the Rural and Urban areas given in the State or Districts.

TOTAL = RURAL + URBAN

Total – represents the whole state.

Rural – represents the villages in state or districts.

Urban – represents the town or city in state or districts.

Age Groups:

In this Column, the ages are divided into certain groups as below,

5 – 14

15 – 34

35 – 59

60+

Age not stated

Genders:

In this data set, there are two types of genders are categorized, they are

Male

Female

Persons = Male + Female

Industrial Categories:

In Data set, Columns are categorized in the event of Industries that shows the type of industries and different phenomenon of works, the below following keys shows the type of industries that are categorized respective to the persons,

A: Agriculture, Forestry and Fishing;

B: Mining and Quarrying;

C: Manufacturing;

D: Electricity, Gas, steam and Air conditioning Supply;

E: Water Supply; (Sewerage, Waste Management and remediation activities);

F: Construction;

G: Wholesale and Retail Trade (Repair of motor vehicles and motor cycles);

H: Transportation and Storage;

I: Accommodation and food service activities;

J: Information and Communication;

K: Financial and Insurance activities;

L: Real Estate activities;

M: Professional, Scientific and Technical activities;

N: Administrative and support service activities;

O: Public Administration and Defence, Compulsory Social Security;

P: Education;

Q: Human Health and Social Work activities;

R: Arts, Entertainment and recreation;

S: Other Service Activities;

T: Activities of Households as Employers: Undifferentiated Goods and Services;

U: Activities of Extra-Territorial Organisations and Bodies.

Work Duration:

In Data set, some of the columns are categorized based on the duration of workers respects with the genders and ages, the duration types that shown in the dataset are:

>3 Months or More but less than 6 Months

>Worked for less than 3 Months

Those columns are describe the time period of person worked.

Gender Distribution:

In this data set, columns are categorized in industrial or works respects with the genders and ages, the below shown are gender distribution presents in the dataset,

Industrial Category - A - Cultivators – Persons;

Industrial Category - A - Cultivators – Males;

Industrial Category - A - Cultivators – Females;

Industrial Category - A - Agricultural labourers – Persons;

Industrial Category - A - Agricultural labourers – Males;

Industrial Category - A - Agricultural labourers – Females;

Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities – Persons;

Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities – Males;

Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities – Females;

Industrial Category - B – Persons;

Industrial Category - B – Males;

Industrial Category - B – Females;

Industrial Category - C - HHI – Persons;

Industrial Category - C - HHI – Males;

Industrial Category - C - HHI – Females;

Industrial Category - C - Non HHI – Persons;

Industrial Category - C - Non HHI – Males;

Industrial Category - C - Non HHI – Females;

Industrial Category - D & E – Persons;

Industrial Category - D & E – Males;

Industrial Category - D & E – Females;

Industrial Category - F – Persons;

Industrial Category - F – Males;

Industrial Category - F – Females;

Industrial Category - G - HHI – Persons;

Industrial Category - G - HHI – Males;

Industrial Category - G - HHI – Females;

Industrial Category - G - Non HHI – Persons;

Industrial Category - G - Non HHI – Males;

Industrial Category - G - Non HHI – Females;

Industrial Category - H – Persons;

Industrial Category - H – Males;

Industrial Category - H – Females;

Industrial Category - I – Persons;

Industrial Category - I – Males;

Industrial Category - I – Females;

Industrial Category - J - HHI – Persons;

Industrial Category - J - HHI – Males;

Industrial Category - J - HHI – Females;

Industrial Category - J - Non HHI – Persons;

Industrial Category - J - Non HHI – Males;

Industrial Category - J - Non HHI – Females;

Industrial Category - K to M – Persons;

Industrial Category - K to M – Males;

Industrial Category - K to M – Females;

Industrial Category - N to O – Persons;

Industrial Category - N to O – Males;

Industrial Category - N to O – Females; Industrial Category - P to Q – Persons;

Industrial Category - P to Q – Males;

Industrial Category - P to Q – Females;

Industrial Category - R to U - HHI – Persons;

Industrial Category - R to U - HHI – Males;

Industrial Category - R to U - HHI – Females;

Industrial Category - R to U - Non HHI – Persons;

Industrial Category - R to U - Non HHI – Males;

Industrial Category - R to U - Non HHI – Females;

ANALYSIS:

The steps for conducting a demographic analysis of marginal workers in Tamil Nadu using Python and data visualization libraries:

1. Define Objectives:

   Objective 1: Demographic Analysis

   - Define the criteria for identifying marginal workers.
   - Analyze the demographic characteristics, including age, industrial category, and sex.

   Objective 2: Socioeconomic Analysis

   - Explore socioeconomic factors affecting marginal workers.
   - Include variables related to education, income, and access to social services.

2. Data Extraction and Cleaning:

   - Obtain datasets containing information on marginal workers in Tamil Nadu.
   - Clean the data, handling missing values, outliers, and ensuring data consistency.
     - Handle Missing Values:
     - Identify and address missing values using appropriate methods like imputation or removal.
     - Data Transformation:

- Standardize formats, units, or any inconsistencies in the dataset.
- Remove Duplicates:
- Check for and remove any duplicate entries in the dataset.
- Handle Outliers:
- Identify and address outliers that could skew the analysis.

3. Data Exploration:
   - Use Pandas for initial exploration and summary statistics.
   - Visualize the distribution of demographic variables using histograms and other relevant plots.

4. Demographic Analysis:
   - Segment the data based on age groups, industrial categories, and sex.
   - Calculate proportions and percentages to understand the composition of each segment.

5. Socioeconomic Analysis:
   - Include variables related to education levels, income, and access to social services.
   - Use descriptive statistics and visualizations to analyze socioeconomic factors.

6. Data Visualization:

   Matplotlib and Seaborn:

   - Use Matplotlib and Seaborn for creating visualizations.
   - Plot bar charts for demographic distributions and box plots for socioeconomic factors.

   Pandas Plotting:

   - Leverage Pandas plotting capabilities for quick visualizations.

7. Interactive Visualizations:

   - Consider using libraries like Plotly for interactive visualizations.
   - Interactive charts can enhance the exploration of data and insights.

8. Geospatial Analysis (Optional):

   - If applicable, consider geospatial visualizations using libraries like GeoPandas for mapping the distribution of marginal workers across regions in Tamil Nadu.

9. Statistical Analysis:

   - Use statistical tests to identify significant differences in demographic and socioeconomic variables.
   - Perform correlation analysis to explore relationships.

10. Documentation and Reporting:
    - Document the data analysis process, including cleaning steps, analysis techniques, and visualization choices.
    - Prepare a detailed report with key findings, insights, and actionable recommendations.

11. Code Organization:
    - Organize Python code into functions or classes for modularity and reproducibility.
    - Add comments and documentation within the code for clarity.

12. Ethical Considerations:
    - Ensure compliance with ethical standards and data privacy regulations.
    - Anonymize data if necessary to protect individual privacy.

13. Stakeholder Engagement:
    - Engage with stakeholders to validate findings and gather additional insights.
    - Incorporate feedback into the analysis.

14. Iterative Analysis:
    - Be open to iterations based on initial findings and stakeholder feedback.

15. Knowledge Sharing:

- Share knowledge and insights gained during the analysis with relevant stakeholders and the broader community if applicable.

By following these steps, you can systematically conduct a demographic and socioeconomic analysis of marginal workers in Tamil Nadu, creating insightful visualizations using Python and data visualization libraries.

Overview of the process:

1.Import Libraries:

Begin by importing the necessary libraries, such as pandas for data manipulation.

2.Load the Dataset:

Use pd.read_csv() or other appropriate methods to load your dataset into a pandas DataFrame.

3.Explore the Dataset:

Display the initial rows, check for missing values, and explore basic statistics to understand the structure and content of the data.

4.Handle Missing Values:

Decide on an appropriate strategy for dealing with missing values, such as dropping rows or filling values based on a specific strategy.

5.Additional Preprocessing Steps:

Depending on the nature of your data, consider additional preprocessing steps such as feature scaling, handling outliers, processing date-time features, dealing with text data, feature engineering, or discretization.

6.Save Preprocessed Dataset (Optional):

Save the preprocessed dataset to a new file if significant changes have been made.

Loading the dataset:

1.Importing libraries

Here, for preprocessing the dataset and manipulate the data, pandas is the library used to frame the data.

Code:

```
Import pandas as pd
```

2.Loading the dataset

In this step, we are framing the data into the table using DataFrame in pandas, and display the head or 5 rows of the dataset.

Code:

```
# Replace with the actual filename
file_path='/Downloads/DDW_B06SC_3300_State_TAMIL_NADU-2011.csv '
```

df = pd.read_csv(file_path)

Preprocessing the dataset

3.Explore the dataset:

After framing data, the first few or five rows of the data in displayed using the head() function.

Code:

print(df.head())

Output:

Table Code State Code District Code Area Name Total/ Rural/ Urban \

0	B0806SC	`33	`000	State - TAMIL NADU	Total
1	B0806SC	`33	`000	State - TAMIL NADU	Total
2	B0806SC	`33	`000	State - TAMIL NADU	Total
3	B0806SC	`33	`000	State - TAMIL NADU	Total
4	B0806SC	`33	`000	State - TAMIL NADU	Total

Age group Worked for 3 months or more but less than 6 months - Persons \

0	Total	1200828
1	`5-14	27791
2	15-34	514340

3	35-59	542581
4	60+	115103

	Worked for 3 months or more but less than 6 months - Males \
0	589003
1	14125
2	259560
3	251957
4	62833

	Worked for 3 months or more but less than 6 months - Females \
0	611825
1	13666
2	254780
3	290624
4	52270

	Worked for less than 3 months - Persons ... \
0	221386 ...
1	2447 ...
2	92423 ...
3	99202 ...
4	27165 ...

	Industrial Category - N to O - Females	\
0	3565	
1	11	
2	1754	
3	1619	
4	175	

	Industrial Category - P to Q - Persons	\
0	11080	
1	122	
2	7536	
3	3205	
4	211	

	Industrial Category - P to Q - Males	\
0	4019	
1	71	
2	2718	
3	1131	
4	93	

	Industrial Category - P to Q - Females	\
0	7061	

1	51
2	4818
3	2074
4	118

	Industrial Category - R to U - HHI - Persons \
0	16833
1	427
2	8346
3	6591
4	1457

	Industrial Category - R to U - HHI - Males \
0	4266
1	169
2	2127
3	1487
4	483

	Industrial Category - R to U - HHI - Females \
0	12567
1	258
2	6219
3	5104

4	974

	Industrial Category - R to U - Non HHI - Persons \
0	122088
1	19305
2	68929
3	26498
4	7065

	Industrial Category - R to U - Non HHI - Males \
0	55801
1	9774
2	32803
3	9675
4	3394

	Industrial Category - R to U - Non HHI - Females
0	66287
1	9531
2	36126
3	16823
4	3671

[5 rows x 69 columns]

4.Check for missing values:

In this step, the missing values or null values, if it present in the data are separated and number of null values are shown through this code.

Code:

```
print("Missing values:\n", df.isnull().sum())
```

Output:

Missing values:

Table Code	0
State Code	0
District Code	0
Area Name	0
Total/ Rural/ Urban	0
..	
Industrial Category - R to U - HHI - Males	0
Industrial Category - R to U - HHI - Females	0
Industrial Category - R to U - Non HHI - Persons	0
Industrial Category - R to U - Non HHI - Males	0
Industrial Category - R to U - Non HHI - Females	0

Length: 69, dtype: int64

5.Check datatype:

In this step, the data type of the columns are discussed

Code:

```
print("Data Types:\n", df.dtypes)
```

Output:

```
Data Types:
 Table Code                                        object
State Code                                         object
District Code                                      object
Area Name                                          object
Total/ Rural/ Urban                                object
                                                     ...
Industrial Category - R to U - HHI - Males         int64
Industrial Category - R to U - HHI - Females       int64
Industrial Category - R to U - Non HHI - Persons   int64
Industrial Category - R to U - Non HHI - Males     int64
Industrial Category - R to U - Non HHI - Females   int64
Length: 69, dtype: object
```

6.Check basic statistics:

the statistics of the columns such as count, mean, std, min, max, 25%, 50%, 75% are shown through the describe() function command.

Code:

```
print("Summary Statistics:\n", df.describe())
```

Output:

Summary Statistics:

Worked for 3 months or more but less than 6 months - Persons \

count	5.940000e+02
mean	1.617277e+04
std	7.607172e+04
min	0.000000e+00
25%	2.872500e+02
50%	2.225500e+03
75%	9.628500e+03
max	1.200828e+06

Worked for 3 months or more but less than 6 months - Males \

count	594.000000
mean	7932.700337
std	36864.822704
min	0.000000
25%	147.250000
50%	1147.000000
75%	4770.500000
max	589003.000000

	Worked for 3 months or more but less than 6 months - Females \
count	594.000000
mean	8240.067340
std	39259.545337
min	0.000000
25%	144.000000
50%	1076.000000
75%	4887.500000
max	611825.000000

	Worked for less than 3 months - Persons \
count	594.000000
mean	2981.629630
std	13909.621137
min	0.000000
25%	27.000000
50%	430.000000
75%	1775.250000
max	221386.000000

	Worked for less than 3 months - Males \
count	594.000000

mean	1338.289562
std	6127.047670
min	0.000000
25%	14.250000
50%	198.500000
75%	774.250000
max	99368.000000

	Worked for less than 3 months - Females \
count	594.000000
mean	1643.340067
std	7808.832522
min	0.000000
25%	13.000000
50%	213.000000
75%	946.500000
max	122018.000000

	Industrial Category - A - Cultivators - Persons \
count	594.000000
mean	865.117845
std	4274.458077
min	0.000000
25%	9.000000

50%	69.500000
75%	466.000000
max	64235.000000

	Industrial Category - A - Cultivators - Males \
count	594.000000
mean	466.424242
std	2298.072295
min	0.000000
25%	5.000000
50%	35.500000
75%	244.250000
max	34632.000000

	Industrial Category - A - Cultivators - Females \
count	594.000000
mean	398.693603
std	1978.682322
min	0.000000
25%	4.000000
50%	32.000000
75%	204.750000
max	29603.000000

	Industrial Category - A - Agricultural labourers - Persons	... \
count	594.000000	...
mean	12225.616162	...
std	60458.382586	...
min	0.000000	...
25%	79.250000	...
50%	1094.000000	...
75%	6279.750000	...
max	907752.000000	...

	Industrial Category - N to O - Females \
count	594.000000
mean	48.013468
std	222.553500
min	0.000000
25%	0.000000
50%	2.000000
75%	18.000000
max	3565.000000

	Industrial Category - P to Q - Persons \
count	594.000000
mean	149.225589

std	696.553730
min	0.000000
25%	0.000000
50%	14.500000
75%	99.750000
max	11080.000000

	Industrial Category - P to Q - Males \
count	594.000000
mean	54.127946
std	253.067862
min	0.000000
25%	0.000000
50%	6.000000
75%	35.750000
max	4019.000000

	Industrial Category - P to Q - Females \
count	594.000000
mean	95.097643
std	444.011425
min	0.000000
25%	0.000000
50%	6.500000

	Industrial Category - R to U - HHI - Persons \
count	594.000000
mean	226.707071
std	1039.953069
min	0.000000
25%	0.000000
50%	27.000000
75%	126.750000
max	16833.000000

	Industrial Category - R to U - HHI - Males \
count	594.000000
mean	57.454545
std	265.230865
min	0.000000
25%	0.000000
50%	7.500000
75%	32.000000
max	4266.000000

	Industrial Category - R to U - HHI - Females \

count	594.000000
mean	169.252525
std	776.206806
min	0.000000
25%	0.000000
50%	20.000000
75%	97.500000
max	12567.000000

	Industrial Category - R to U - Non HHI - Persons \
count	594.000000
mean	1644.282828
std	7325.241597
min	0.000000
25%	64.500000
50%	263.500000
75%	994.000000
max	122088.000000

	Industrial Category - R to U - Non HHI - Males \
count	594.000000
mean	751.528620
std	3352.811737
min	0.000000

25%	34.000000
50%	123.000000
75%	447.750000
max	55801.000000

Industrial Category - R to U - Non HHI - Females

count	594.000000
mean	892.754209
std	3988.125301
min	0.000000
25%	30.500000
50%	135.000000
75%	500.000000
max	66287.000000

[8 rows x 63 columns]

8.Saving Preprocessed dataset:

In this step, if we made substantial changes to the dataset and want to save the preprocessed version, you can use the following Code

Code:

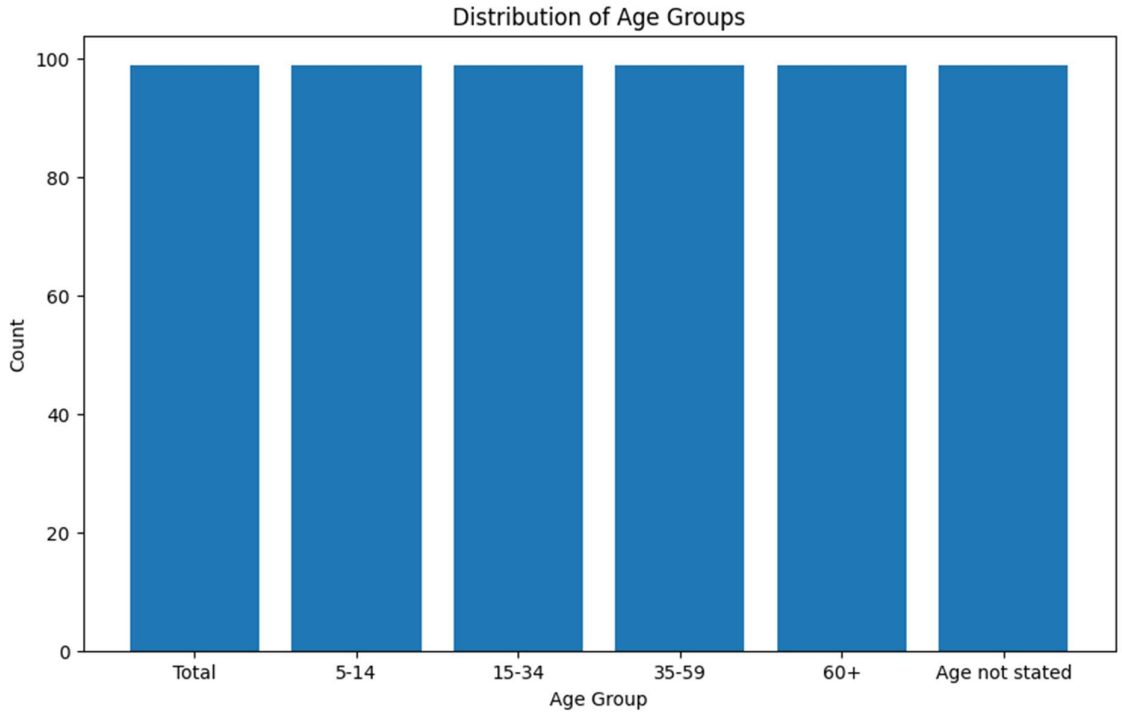
```
# Save the preprocessed dataset to a new CSV file
df.to_csv('preprocessed_dataset.csv', index=False)
```

9. Distribution of Age Groups:

Code:

```
import matplotlib.pyplot as plt

# Assuming 'Age group' is a column in your DataFrame
age_distribution = df['Age group'].value_counts()
plt.figure(figsize=(10, 6))
plt.bar(age_distribution.index, age_distribution.values)
plt.xlabel('Age Group')
plt.ylabel('Count')
plt.title('Distribution of Age Groups')
plt.show()
```

Output:



10. Distribution of Workers by Industrial Category:

Code:

```
import matplotlib.pyplot as plt


# Combine all the relevant columns for industrial categories
industrial_columns = ['Industrial Category - A - Cultivators - Persons',
    'Industrial Category - A - Cultivators - Males',
    'Industrial Category - A - Cultivators - Females',
    'Industrial Category - A - Agricultural labourers - Persons',
    'Industrial Category - A - Agricultural labourers - Males',
    'Industrial Category - A - Agricultural labourers - Females',
    'Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities - Persons',
    'Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities - Males',
    'Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities - Females',
    'Industrial Category - B - Persons', 'Industrial Category - B - Males',
    'Industrial Category - B - Females',
    'Industrial Category - C - HHI - Persons',
    'Industrial Category - C - HHI - Males',
    'Industrial Category - C - HHI - Females',
```

'Industrial Category - C - Non HHI - Persons',

'Industrial Category - C - Non HHI - Males',

'Industrial Category - C - Non HHI - Females',

'Industrial Category - D & E - Persons',

'Industrial Category - D & E - Males',

'Industrial Category - D & E - Females',

'Industrial Category - F - Persons', 'Industrial Category - F - Males',

'Industrial Category - F - Females',

'Industrial Category - G - HHI - Persons',

'Industrial Category - G - HHI - Males',

'Industrial Category - G - HHI - Females',

'Industrial Category - G - Non HHI - Persons',

'Industrial Category - G - Non HHI - Males',

'Industrial Category - G - Non HHI - Females',

'Industrial Category - H - Persons', 'Industrial Category - H - Males',

'Industrial Category - H - Females',

'Industrial Category - I - Persons', 'Industrial Category - I - Males',

'Industrial Category - I - Females',

'Industrial Category - J - HHI - Persons',

'Industrial Category - J - HHI - Males',

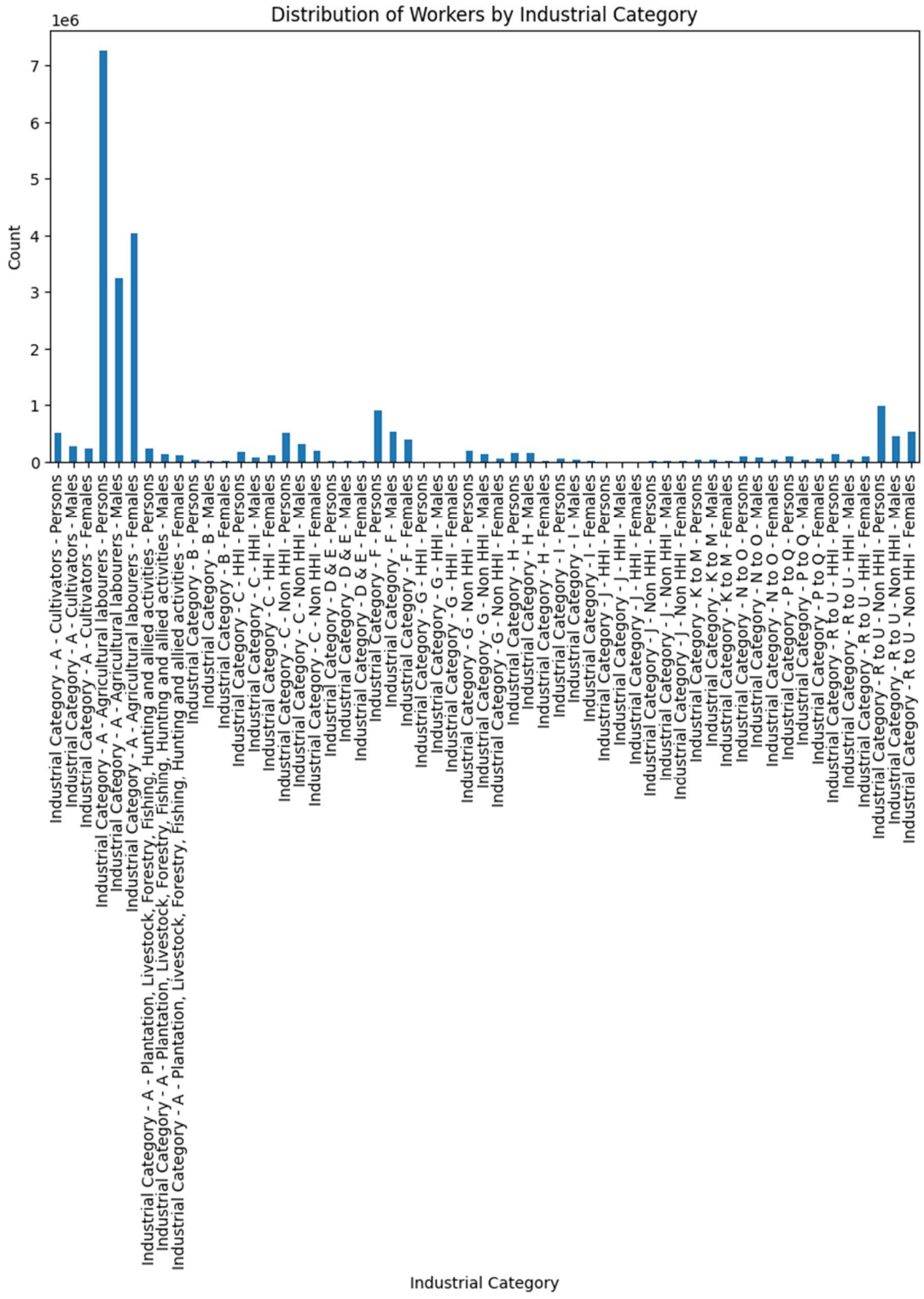
'Industrial Category - J - HHI - Females',

'Industrial Category - J - Non HHI - Persons',

```
        'Industrial Category - J - Non HHI - Males',
        'Industrial Category - J - Non HHI - Females',
        'Industrial Category - K to M - Persons',
        'Industrial Category - K to M - Males',
        'Industrial Category - K to M - Females',
        'Industrial Category - N to O - Persons',
        'Industrial Category - N to O - Males',
        'Industrial Category - N to O - Females',
        'Industrial Category - P to Q - Persons',
        'Industrial Category - P to Q - Males',
        'Industrial Category - P to Q - Females',
        'Industrial Category - R to U - HHI - Persons',
        'Industrial Category - R to U - HHI - Males',
        'Industrial Category - R to U - HHI - Females',
        'Industrial Category - R to U - Non HHI - Persons',
        'Industrial Category - R to U - Non HHI - Males',
        'Industrial Category - R to U - Non HHI - Females']

# Sum the counts across all the industrial category columns
industrial_counts = df[industrial_columns].sum()

# Create a bar chart to visualize the distribution of workers in each industrial category
plt.figure(figsize=(10, 5))
```

```
industrial_counts.plot(kind='bar', title='Distribution of Workers by Industrial Category')
plt.xlabel('Industrial Category')
plt.ylabel('Count')
plt.show()
```

Output:


Distribution of Workers by Industrial Category
1e6
Count
Industrial Category
Industrial Category - A - Cultivators - Persons
Industrial Category - A - Cultivators - Males
Industrial Category - A - Cultivators - Females
Industrial Category - A - Agricultural labourers - Persons
Industrial Category - A - Agricultural labourers - Males
Industrial Category - A - Agricultural labourers - Females
Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities - Persons
Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities - Males
Industrial Category - A - Plantation, Livestock, Forestry, Fishing, Hunting and allied activities - Females
Industrial Category - B - Persons
Industrial Category - B - Males
Industrial Category - B - Females
Industrial Category - C - HHI - Persons
Industrial Category - C - HHI - Males
Industrial Category - C - HHI - Females
Industrial Category - C - Non HHI - Persons
Industrial Category - C - Non HHI - Males
Industrial Category - C - Non HHI - Females
Industrial Category - D & E - Persons
Industrial Category - D & E - Males
Industrial Category - D & E - Females
Industrial Category - F - Persons
Industrial Category - F - Males
Industrial Category - F - Females
Industrial Category - G - HHI - Persons
Industrial Category - G - HHI - Males
Industrial Category - G - HHI - Females
Industrial Category - G - Non HHI - Persons
Industrial Category - G - Non HHI - Males
Industrial Category - G - Non HHI - Females
Industrial Category - H - Persons
Industrial Category - H - Males
Industrial Category - H - Females
Industrial Category - I - Persons
Industrial Category - I - Males
Industrial Category - I - Females
Industrial Category - J - HHI - Persons
Industrial Category - J - HHI - Males
Industrial Category - J - HHI - Females
Industrial Category - J - Non HHI - Persons
Industrial Category - J - Non HHI - Males
Industrial Category - J - Non HHI - Females
Industrial Category - K to M - Persons
Industrial Category - K to M - Males
Industrial Category - K to M - Females
Industrial Category - N to O - Persons
Industrial Category - N to O - Males
Industrial Category - N to O - Females
Industrial Category - P to Q - Persons
Industrial Category - P to Q - Males
Industrial Category - P to Q - Females
Industrial Category - R to U - HHI - Persons
Industrial Category - R to U - HHI - Males
Industrial Category - R to U - HHI - Females
Industrial Category - R to U - Non HHI - Persons
Industrial Category - R to U - Non HHI - Males
Industrial Category - R to U - Non HHI - Females

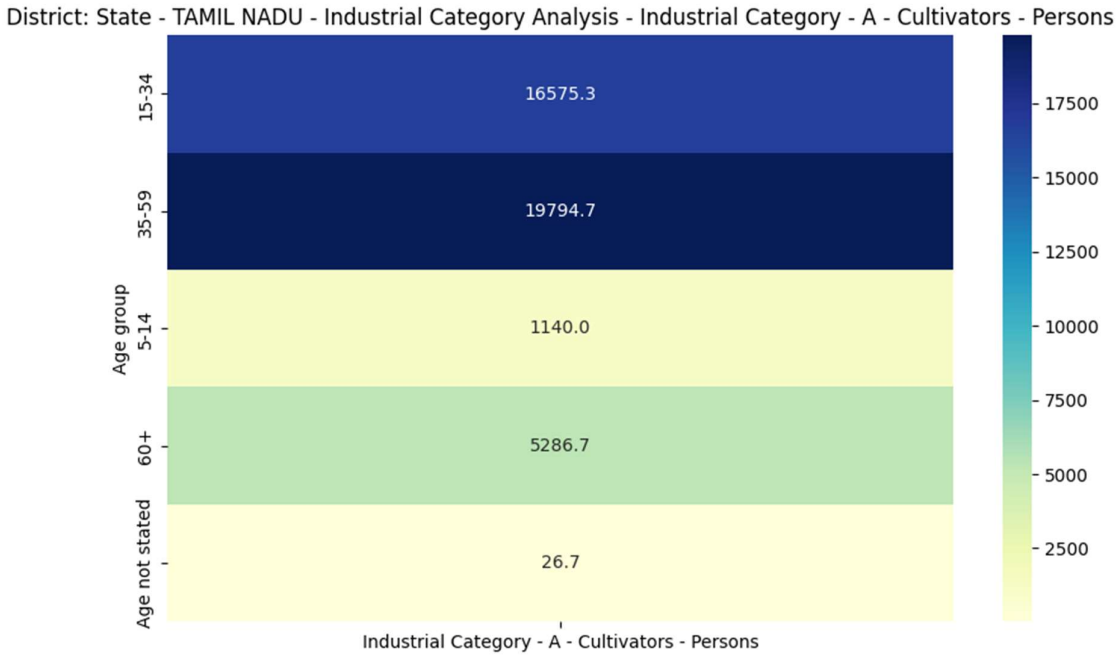
11.Heat map industrial category wise:

Code:

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

# Assuming df is your DataFrame after data cleaning and clustering

# List of industrial category columns
industrial_categories = [
    'Industrial Category - A - Cultivators - Persons',
    'Industrial Category - A - Cultivators - Males',
    'Industrial Category - A - Cultivators - Females',
    # Add all other industrial category columns here
]

# List of district names
districts = df['Area Name'].unique()

# Loop through districts
for district in districts:
    district_df = df[df['Area Name'] == district]
```

```
    # Loop through industrial categories
    for category in industrial_categories:
        # Create a pivot table for the specific category in the district
        pivot_table = district_df.pivot_table(index='Age group', values=category, aggfunc='mean')

        # Create a heatmap
        plt.figure(figsize=(10, 6))
        sns.heatmap(pivot_table, annot=True, fmt=".1f", cmap='YlGnBu')
        plt.title(f'District: {district} - Industrial Category Analysis - {category}')
        plt.show()
```

Output:



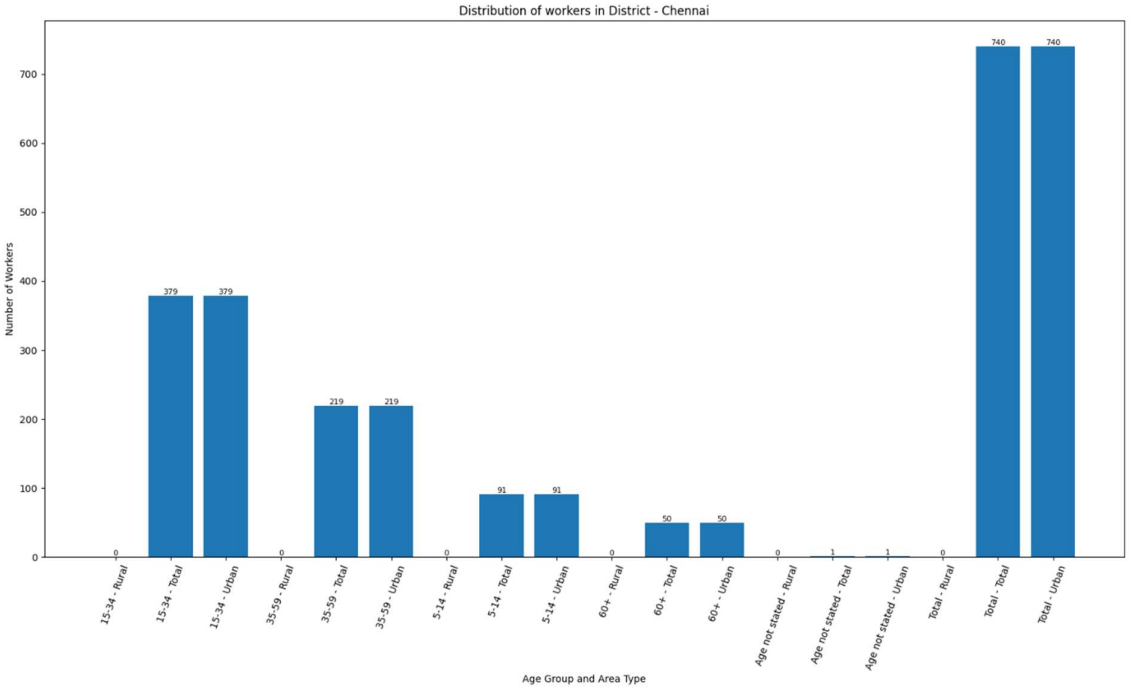
12.Visualization District Wise:

Code:

```
import matplotlib.pyplot as plt


# 'Area Name' represents the districts, 'Age group' represents the age groups, 'Total/ Rural/ Urban' represents rural or urban

# 'Industrial Category - A - Cultivators - Persons' represents the number of workers taken as sample'


# Grouping by 'Area Name', 'Age group', 'Total/ Rural/ Urban' and summing up the number of workers

grouped_data = df.groupby(['Area Name', 'Age group', 'Total/ Rural/ Urban'])['Industrial Category - A - Cultivators - Persons'].sum().reset_index()


# Create a separate plot for each district

districts = grouped_data['Area Name'].unique()


for district in districts:

    district_data = grouped_data[grouped_data['Area Name'] == district]

    plt.figure(figsize=(20, 10))

    bars = plt.bar(district_data['Age group'] + ' - ' + district_data['Total/ Rural/ Urban'], district_data['Industrial Category - A - Cultivators - Persons'])
```

```
    # Adding numbers on top of the bars
    for bar in bars:
        yval = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2, yval, round(yval), va='bottom', ha='center', fontsize=8, color='black')

    plt.title(f'Distribution of workers in {district}')
    plt.xlabel('Age Group and Area Type')
    plt.ylabel('Number of Workers')
    plt.xticks(rotation=70)
    plt.show()
```

Output:



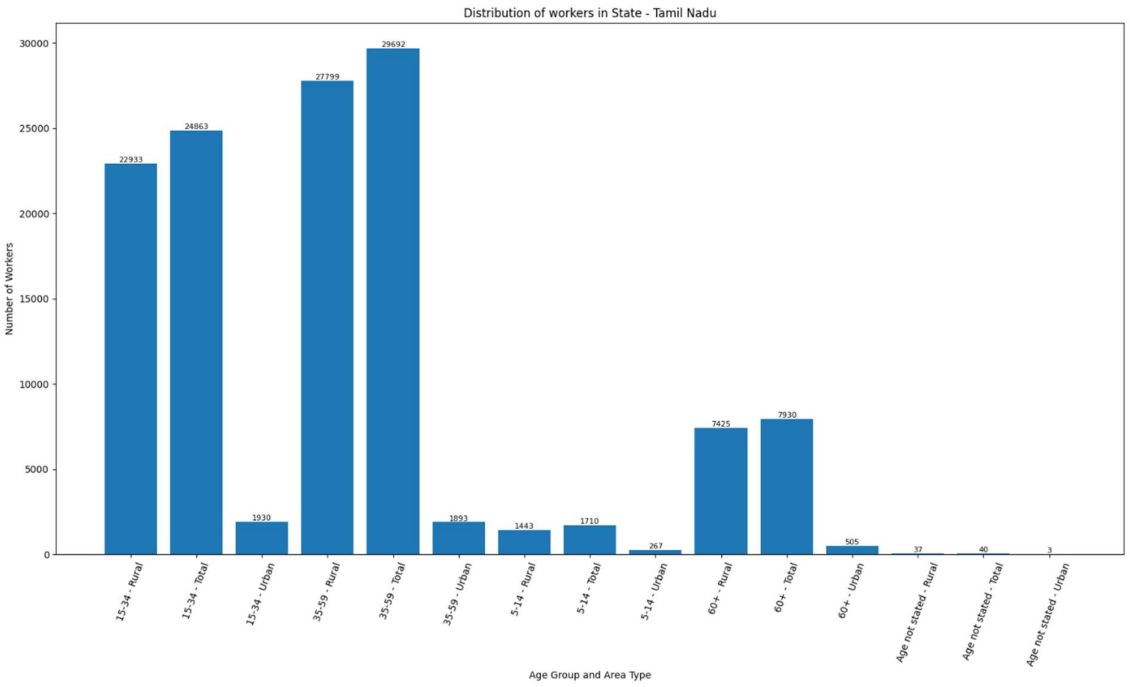
13.Visualization of full state:

Code:

```
# Entire State


import matplotlib.pyplot as plt


# Assuming 'df' is your DataFrame

# 'Area Name' represents the districts, 'Age group' represents the age groups, 'Total/ Rural/ Urban' represents rural or urban

# 'Industrial Category - A - Cultivators - Persons' represents the number of workers


# Filter data for State - Tamil Nadu

state_data = df[df['Area Name'] == 'State - TAMIL NADU']


# Grouping by 'Age group', 'Total/ Rural/ Urban' and summing up the number of workers

grouped_data = state_data.groupby(['Age group', 'Total/ Rural/ Urban'])['Industrial Category - A - Cultivators - Persons'].sum().reset_index()


# Create the bar chart

plt.figure(figsize=(20, 10))

bars = plt.bar(grouped_data['Age group'] + ' - ' + grouped_data['Total/ Rural/ Urban'], grouped_data['Industrial Category - A - Cultivators - Persons'])
```

```
# Adding numbers on top of the bars
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval, round(yval), va='bottom', ha='center', fontsize=8, color='black')

plt.title('Distribution of workers in State - Tamil Nadu')
plt.xlabel('Age Group and Area Type')
plt.ylabel('Number of Workers')
plt.xticks(rotation=70)
plt.show()
```

Output:

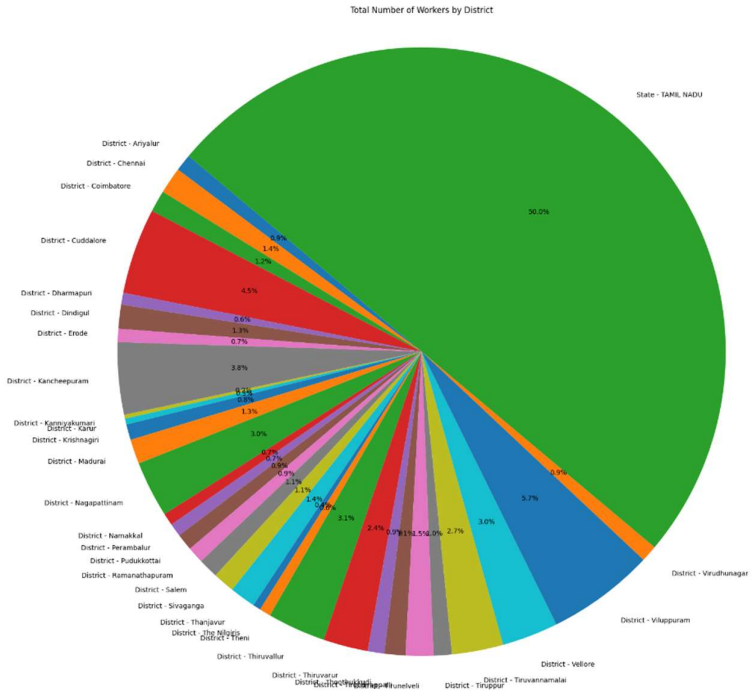


14.Total Number of workers by district:

Code:

```
import matplotlib.pyplot as plt


# Take the absolute values of district workers

district_workers = df.groupby('Area Name')['Worked for 3 months or more but less than 6 months - Persons'].sum().abs()


# Create a pie chart

plt.figure(figsize=(30, 18))

plt.pie(district_workers, labels=district_workers.index, autopct='%1.1f%%', startangle=140)

plt.title('Total Number of Workers by District')

plt.axis('equal')  # Equal aspect ratio ensures the pie chart is circular.


# Show the plot

plt.show()
```

Output:



15. Age wise Employment rate for categories : Worked for 3 to 6 months workers and workers who worked less than 3months

Code:

Define the relevant data for each age group

age_groups = ['5-9', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '40-49', '50-59', '60-69', '70-79', '80+']

worked_3_6_months = [48238, 76288, 257605, 478082, 554851, 483456, 502791, 824271, 539168, 324681, 103004, 22844]

worked_less_3_months = [2051, 6993, 41938, 81036, 91694, 79385, 84066, 137834, 96980, 70594, 25242, 5595]

Define the new total population for all age groups

new_total_population = 4942775

```
# Calculate the employment rate for each age group with the modified total population
employment_rate_age_group_modified = [
    ((worked_3_6 + worked_less_3) / new_total_population) * 100
    for worked_3_6, worked_less_3 in zip(worked_3_6_months, worked_less_3_months)
]

# Print the results
for age_group, rate in zip(age_groups, employment_rate_age_group_modified):
    print(f"Age Group: {age_group}\n- Employment Rate: {rate:.2f}%\n")
```

Output:

Age Group: 5-9

- Employment Rate: 1.02%

Age Group: 10-14

- Employment Rate: 1.68%

Age Group: 15-19

- Employment Rate: 6.06%

Age Group: 20-24

- Employment Rate: 11.31%

Age Group: 25-29

- Employment Rate: 13.08%

Age Group: 30-34

- Employment Rate: 11.39%

Age Group: 35-39

- Employment Rate: 11.87%

Age Group: 40-49

- Employment Rate: 19.46%

Age Group: 50-59

- Employment Rate: 12.87%

Age Group: 60-69

- Employment Rate: 8.00%

Age Group: 70-79

- Employment Rate: 2.59%

Age Group: 80+

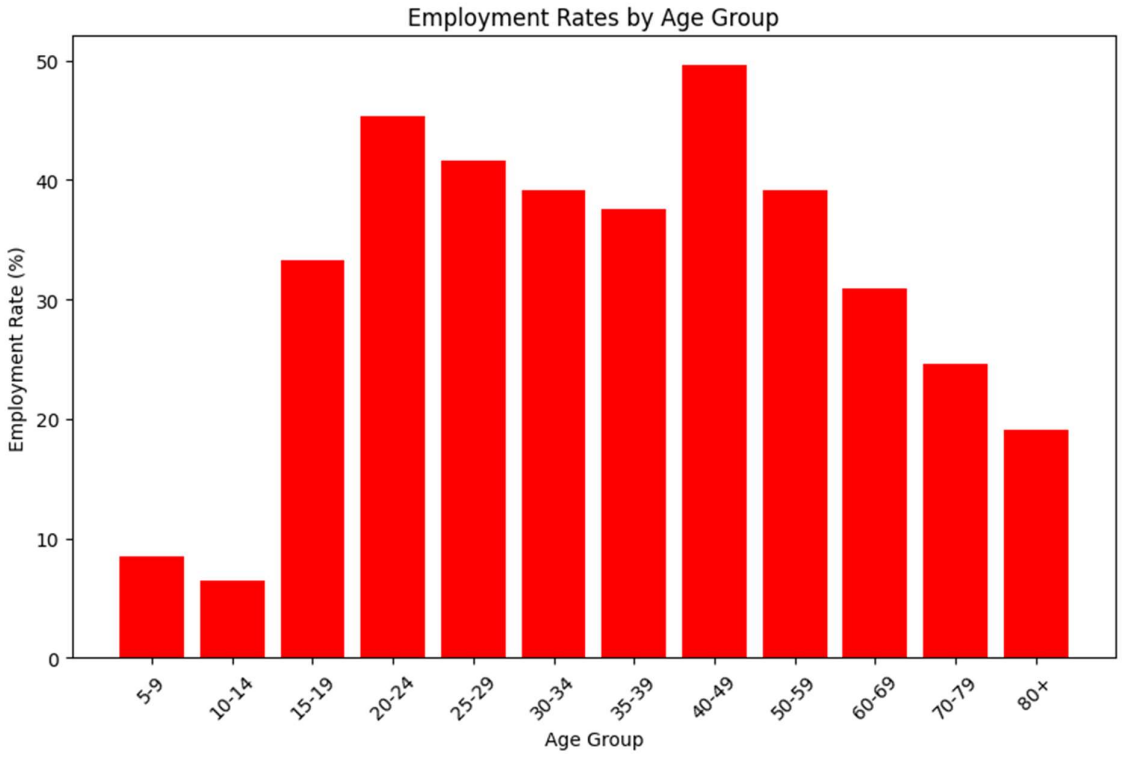
- Employment Rate: 0.58%

16.Employment rates by age groups:

Code:

```
import matplotlib.pyplot as plt


# Define the relevant data
age_groups = ['5-9', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '40-49', '50-59', '60-69', '70-79', '80+']
employment_rates = [8.53, 6.50, 33.23, 45.30, 41.65, 39.17, 37.61, 49.58, 39.11, 30.97, 24.60, 19.09]


# Create a bar chart
plt.figure(figsize=(10, 6))
plt.bar(age_groups, employment_rates, color='red')
plt.xlabel('Age Group')
plt.ylabel('Employment Rate (%)')
plt.title('Employment Rates by Age Group')
plt.xticks(rotation=45)  # Rotate x-axis labels for better visibility
plt.show()
```

Output:

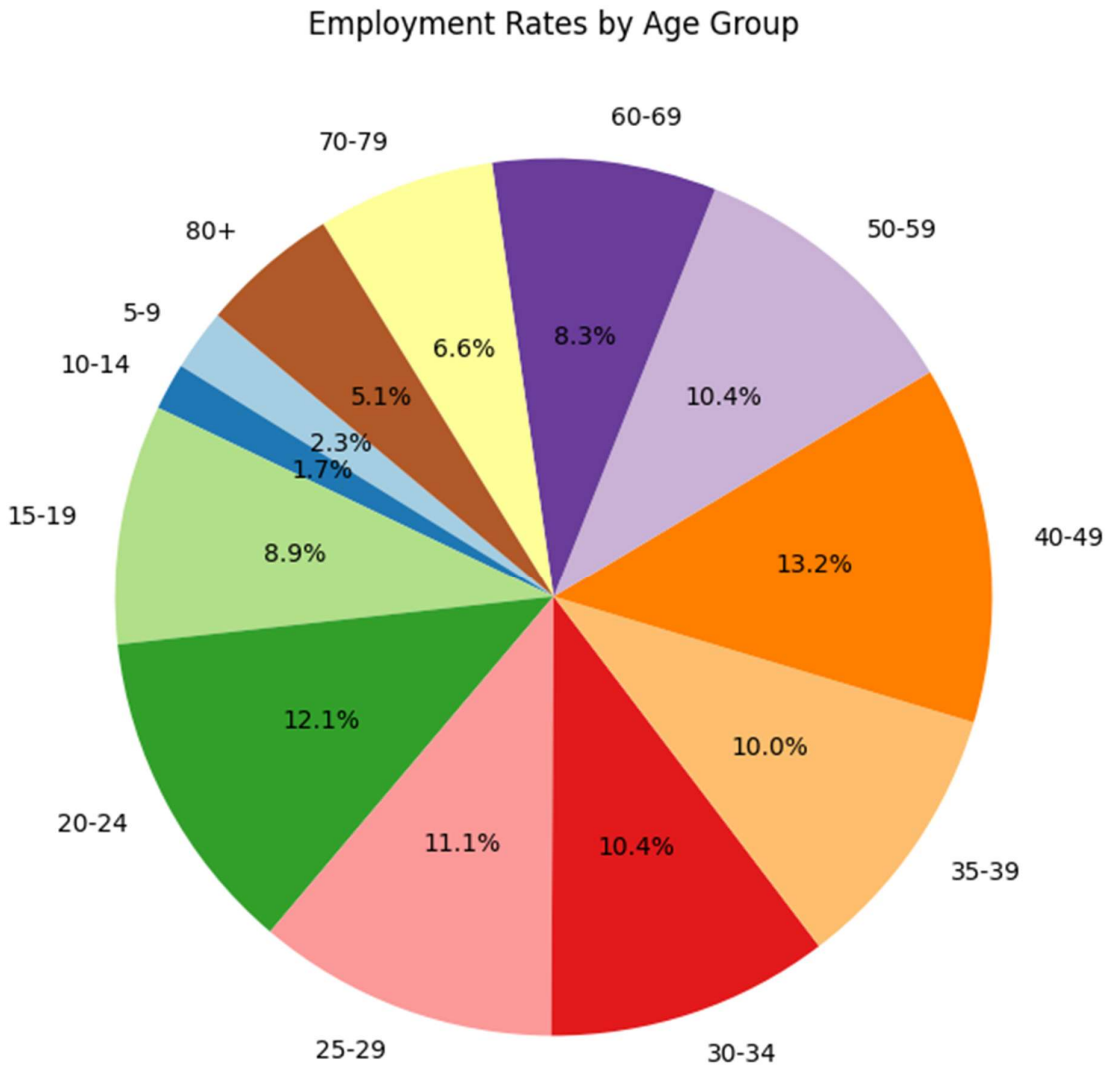


Code:

```
import matplotlib.pyplot as plt


# Define the relevant data
age_groups = ['5-9', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '40-49', '50-59', '60-69', '70-79', '80+']
employment_rates = [8.53, 6.50, 33.23, 45.30, 41.65, 39.17, 37.61, 49.58, 39.11, 30.97, 24.60, 19.09]


# Create a pie chart
plt.figure(figsize=(8, 8))
```

```
plt.pie(employment_rates, labels=age_groups,
autopct='%1.1f%%', startangle=140,
colors=plt.cm.Paired(range(len(age_groups))))

plt.title('Employment Rates by Age Group')

plt.show()
```

Output:



Code:

```
# Define the relevant data for each industrial category
category_a_cultivators = 393082
category_a_agricultural_labourers = 2372446
category_b = 14979
category_c_hhi = 10290
category_c_non_hhi = 4689
category_d_e = 154133
category_f = 53418
category_g_hhi = 100715
category_g_non_hhi = 306528
category_h = 188464
category_i = 118064
category_j_hhi = 7137
category_j_non_hhi = 6003
category_k_to_m = 1134
category_n_to_o = 390275
category_p_to_q = 241619
category_r_to_u_hhi = 148656
category_r_to_u_non_hhi = 510

# Define the total population
total_population = 4942775
```

```
# Calculate the employment rate for each industrial category
employment_rate_category_a = ((category_a_cultivators + category_a_agricultural_labourers) / total_population) * 100
employment_rate_category_b = (category_b / total_population) * 100
employment_rate_category_c_hhi = (category_c_hhi / total_population) * 100
employment_rate_category_c_non_hhi = (category_c_non_hhi / total_population) * 100
employment_rate_category_d_e = (category_d_e / total_population) * 100
employment_rate_category_f = (category_f / total_population) * 100
employment_rate_category_g_hhi = (category_g_hhi / total_population) * 100
employment_rate_category_g_non_hhi = (category_g_non_hhi / total_population) * 100
employment_rate_category_h = (category_h / total_population) * 100
employment_rate_category_i = (category_i / total_population) * 100
employment_rate_category_j_hhi = (category_j_hhi / total_population) * 100
employment_rate_category_j_non_hhi = (category_j_non_hhi / total_population) * 100
employment_rate_category_k_to_m = (category_k_to_m / total_population) * 100
```

```
employment_rate_category_n_to_o = (category_n_to_o /
total_population) * 100

employment_rate_category_p_to_q = (category_p_to_q /
total_population) * 100

employment_rate_category_r_to_u_hhi = (category_r_to_u_hhi
/ total_population) * 100

employment_rate_category_r_to_u_non_hhi =
(category_r_to_u_non_hhi / total_population) * 100


# Print the results

print(f"Category A Employment Rate:
{employment_rate_category_a:.2f}%")

print(f"Category B Employment Rate:
{employment_rate_category_b:.2f}%")

print(f"Category C HHI Employment Rate:
{employment_rate_category_c_hhi:.2f}%")

print(f"Category C Non HHI Employment Rate:
{employment_rate_category_c_non_hhi:.2f}%")

print(f"Category D & E Employment Rate:
{employment_rate_category_d_e:.2f}%")

print(f"Category F Employment Rate:
{employment_rate_category_f:.2f}%")

print(f"Category G HHI Employment Rate:
{employment_rate_category_g_hhi:.2f}%")

print(f"Category G Non HHI Employment Rate:
{employment_rate_category_g_non_hhi:.2f}%")

print(f"Category H Employment Rate:
{employment_rate_category_h:.2f}%")
```

```
print(f"Category I Employment Rate:
{employment_rate_category_i:.2f}%")

print(f"Category J HHI Employment Rate:
{employment_rate_category_j_hhi:.2f}%")

print(f"Category J Non HHI Employment Rate:
{employment_rate_category_j_non_hhi:.2f}%")

print(f"Category K to M Employment Rate:
{employment_rate_category_k_to_m:.2f}%")

print(f"Category N to O Employment Rate:
{employment_rate_category_n_to_o:.2f}%")

print(f"Category P to Q Employment Rate:
{employment_rate_category_p_to_q:.2f}%")

print(f"Category R to U HHI Employment Rate:
{employment_rate_category_r_to_u_hhi:.2f}%")

print(f"Category R to U Non HHI Employment Rate:
{employment_rate_category_r_to_u_non_hhi:.2f}%")
```

Output:

```
Category A Employment Rate: 55.95%

Category B Employment Rate: 0.30%

Category C HHI Employment Rate: 0.21%

Category C Non HHI Employment Rate: 0.09%

Category D & E Employment Rate: 3.12%

Category F Employment Rate: 1.08%

Category G HHI Employment Rate: 2.04%

Category G Non HHI Employment Rate: 6.20%
```

Category H Employment Rate: 3.81%

Category I Employment Rate: 2.39%

Category J HHI Employment Rate: 0.14%

Category J Non HHI Employment Rate: 0.12%

Category K to M Employment Rate: 0.02%

Category N to O Employment Rate: 7.90%

Category P to Q Employment Rate: 4.89%

Category R to U HHI Employment Rate: 3.01%

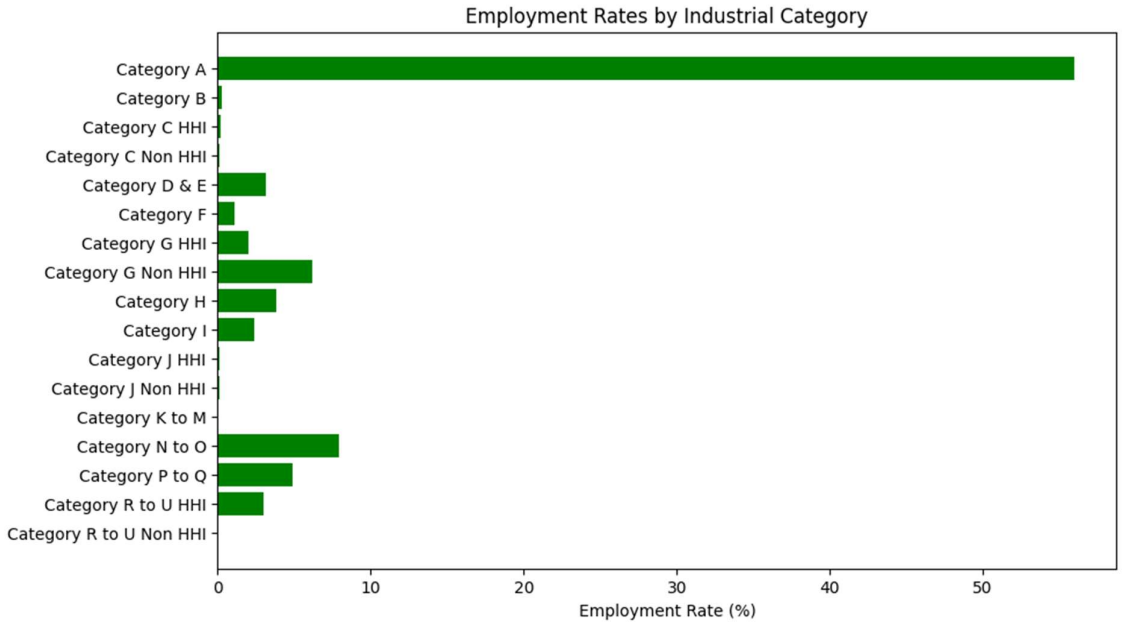
Category R to U Non HHI Employment Rate: 0.01%

Code:

```
#VISUALIZE FOR SLL CATEGORIES


import matplotlib.pyplot as plt


# Define the industrial categories and their respective employment rates
categories = ['Category A', 'Category B', 'Category C HHI', 'Category C Non HHI',
              'Category D & E', 'Category F', 'Category G HHI', 'Category G Non HHI',
              'Category H', 'Category I', 'Category J HHI', 'Category J Non HHI',
              'Category K to M', 'Category N to O', 'Category P to Q',
              'Category R to U HHI', 'Category R to U Non HHI']
```

```
employment_rates = [employment_rate_category_a,
employment_rate_category_b,
                    employment_rate_category_c_hhi,
employment_rate_category_c_non_hhi,
                    employment_rate_category_d_e,
employment_rate_category_f,
                    employment_rate_category_g_hhi,
employment_rate_category_g_non_hhi,
                    employment_rate_category_h,
employment_rate_category_i,
                    employment_rate_category_j_hhi,
employment_rate_category_j_non_hhi,
                    employment_rate_category_k_to_m,
employment_rate_category_n_to_o,
                    employment_rate_category_p_to_q,
employment_rate_category_r_to_u_hhi,
                    employment_rate_category_r_to_u_non_hhi]


# Create a bar chart
plt.figure(figsize=(10, 6))
plt.barh(categories, employment_rates, color='green')
plt.xlabel('Employment Rate (%)')
plt.title('Employment Rates by Industrial Category')
plt.gca().invert_yaxis()  # Invert y-axis for better visualization
plt.show()
```

Output:


Employment Rates by Industrial Category
Category A
Category B
Category C HHI
Category C Non HHI
Category D & E
Category F
Category G HHI
Category G Non HHI
Category H
Category I
Category J HHI
Category J Non HHI
Category K to M
Category N to O
Category P to Q
Category R to U HHI
Category R to U Non HHI
0
10
20
30
40
50
Employment Rate (%)

CONCLUSION:

In conclusion, the outlined data loading and preprocessing steps provide a foundational framework for preparing a dataset for analysis in Python using the pandas library. The demographic analysis and visualizations of marginal workers in Tamil Nadu have provided valuable insights into the composition of this vital workforce. We have observed a diverse age distribution, indicating that marginal workers span multiple age groups, potentially reflecting varying stages of life and career development. The examination of industrial categories has shed light on the prominent sectors where these workers are employed, which can be instrumental in tailoring workforce-related policies and interventions. Additionally, our analysis of gender distribution revealed the presence of both male and female workers, highlighting the need for gender sensitive labour policies.

These findings present opportunities for policymakers and labour organizations to design targeted strategies that consider the unique needs of marginal workers across different age groups, industrial sectors, and genders. By leveraging these insights, efforts can be made to enhance employment opportunities, job security, and working conditions for this vital workforce, thereby fostering greater socio-economic inclusivity in Tamil Nadu. This analysis, complemented by data visualizations, serves as a foundation for informed decision-making, and it encourages the development of initiatives that support and empower marginal workers in the region.