TAMILNADU MARGINAL WORKERS ASSESSMENT

Data Analytics with cognos – Phase 3

DOCUMENTATION

Team Members:

1.Yazhini.B(au613021205062)

2.Dhanusiya.R(au613021205007)

3.Kavya.K(au613021205025)

4.Vishalatchi.Y(au613021205061)

5.Kiruthika.R(au613021205027)

Phase 3: Development Part 1

Problem Definition:

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

Dataset Link:

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

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
75%	64.000000
max	7061.000000

	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]

7.Additional Preprocessing steps:

Perform any other preprocessing steps that are specific to your dataset and analysis goals. This may include scaling numeric features, handling outliers, or creating new features.

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)
```

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. By following these steps, you can ensure that your data is in a suitable format and quality for further exploration and visualization tasks.