TN Marginal Workers Assessment
PHASE 3:DEVELOPMENT PART 1

Introduction:

Data loading and pre-processing are fundamental steps in any data-driven project, whether it's machine learning, data analysis, or any other data-centric task. These crucial processes involve gathering raw data from various sources and transforming it into a suitable format for analysis or model training

Proper data preprocessing is essential to ensure that the data used for analysis or machine learning is accurate, consistent, and suitable for the intended purpose

Data preprocessing is the process of cleaning and preparing raw data before it is used in data analysis.

The steps are:

- *Data Cleaning*
- *Data Transformation*
- *Data Reduction*
- *Data Splitting*

Data Cleaning:

Removing or correcting any errors or inconsistencies in the data, such as missing values, duplicates, or outliers.

Data Transformation:

This can include converting data into a suitable format, scaling or normalizing features, and encoding categorical variables into numerical values.

Data Reduction:

Sometimes, data may be too large or contain redundant information. Data reduction techniques, like dimensionality reduction, can be applied to reduce the data's complexity.

Data Splitting:

Dividing the data into training and testing sets for model evaluation and validation.

The given data set:

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

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Table Cod	State Cod	District Co	Area Nam	Total/ Rur	Age group	Worked fo	Worked fo	Worked fo	Worked fo	Worked fo	Worked fo	Industrial	Industrial	Industrial	Industrial	Industrial	Industrial	Industrial	Industrial	Industr
2	B0806SC	`33	`000	State - TAI	Total	Total	1200828	589003	611825	221386	99368	122018	64235	34632	29603	907752	404844	502908	29410	16268	131
3	B0806SC	`33	`000	State - TAI	Total	`5-14	27791	14125	13666	2447	1247	1200	1710	825	885	6398	3130	3268	190	107	
4	B0806SC	`33	`000	State - TAI	Total	15-34	514340	259560	254780	92423	43892	48531	24863	12711	12152	345420	152968	192452	9430	5443	3
5	B0806SC	`33	`000	State - TAI	Total	35-59	542581	251957	290624	99202	40691	58511	29692	15927	13765	450052	192771	257281	15744	8230	7
6	B0806SC	`33	`000	State - TAI	Total	60+	115103	62833	52270	27165	13465	13700	7930	5151	2779	105325	55730	49595	4028	2470	1
7	B0806SC	`33	`000	State - TAI	Total	Age not st	1013	528	485	149	73	76	40	18	22	557	245	312	18	18	
8	B0806SC	`33	`000	State - TAI	Rural	Total	966645	459738	506907	174443	73663	100780	59637	32189	27448	824698	364131	460567	19758	11033	87
9	B0806SC	`33	`000	State - TAI	Rural	`5-14	17239	8713	8526	1977	985	992	1443	684	759	6005	2922	3083	144	80	
10	B0806SC	`33	`000	State - TAI	Rural	15-34	406847	198575	208272	71974	31917	40057	22933	11766	11167	316885	138622	178263	6687	3909	2
11	B0806SC	`33	`000	State - TAI	Rural	35-59	444800	199573	245227	77922	29808	48114	27799	14887	12912	406147	172178	233969	10307	5468	4
12	B0806SC	`33	`000	State - TAI	Rural	60+	97011	52498	44513	22446	10902	11544	7425	4835	2590	95151	50192	44959	2608	1564	1
13	B0806SC	`33	`000	State - TAI	Rural	Age not st	748	379	369	124	51	73	37	17	20	510	217	293	12	12	
14	B0806SC	`33	`000	State - TAI	Urban	Total	234183	129265	104918	46943	25705	21238	4598	2443	2155	83054	40713	42341	9652	5235	4
15	B0806SC	`33	`000	State - TAI	Urban	`5-14	10552	5412	5140	470	262	208	267	141	126	393	208	185	46	27	
16	B0806SC	`33	`000	State - TAI	Urban	15-34	107493	60985	46508	20449	11975	8474	1930	945	985	28535	14346	14189	2743	1534	1
17	B0806SC	`33	`000	State - TAI	Urban	35-59	97781	52384	45397	21280	10883	10397	1893	1040	853	43905	20593	23312	5437	2762	2
18	B0806SC	`33	`000	State - TAI	Urban	60+	18092	10335	7757	4719	2563	2156	505	316	189	10174	5538	4636	1420	906	5
19	B0806SC	`33	`000	State - TAI	Urban	Age not st	265	149	116	25	22	3	3	1	2	47	28	19	6	6	
20	B0806SC	`33	`602	District - T	Total	Total	74448	39295	35153	15866	8004	7862	3066	1663	1403	42579	20345	22234	1519	1025	4
21	B0806SC	`33	`602	District - T	Total	`5-14	2521	1284	1237	147	82	65	122	56	66	330	154	176	12	12	
22	B0806SC	`33	`602	District - T	Total	15-34	33568	18049	15519	6529	3654	2875	1225	632	593	15591	7257	8334	570	387	1
23	B0806SC	`33	`602	District - T	Total	35-59	32568	16771	15797	7718	3529	4189	1414	792	622	22192	10446	11746	788	532	2
24	B0806SC	`33	`602	District - T	Total	60+	5716	3147	2569	1465	739	726	305	183	122	4441	2476	1965	149	94	
25	B0806SC	`33	`602	District - T	Total	Age not st	75	44	31	7	0	7	0	0	0	25	12	13	0	0	

Algorithm to perform data loading and cleaning

Step1: Import necessary libraries:

import pandas as pd

Step2: Load the data:

Use pd.read_csv()

Step3: Explore the data:

Use functions like head(), info(), and describe()

Step4: Data Cleaning:

Identify and address missing values, duplicates, and other data quality issues.

Step5: Data Transformation

Step6: Save the cleaned data

PROGRAM:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
# Import the data set
df = pd.read_csv("C:\\Users\lenovo\Downloads\DDW_B06SC_3300_State_TAMIL_NADU-2011.csv")
# Drop columns that aren't useful
df.drop('Sl. No.', axis=1, inplace=True)
```

```
# Handle missing data

# In this case, we will drop the rows with missing values.

df.dropna(inplace=True)

# Encode categorical data

# The data set contains categorical data in the `Industrial category` column. We can encode this data using the `pd.get_dummies()` function.

df = pd.get_dummies(df, columns=['Industrial category'])

# Split the data set into training and test sets

# We will split the data set into 80% training data and 20% test data.

X_train, X_test, y_train, y_test =
train_test_split(df.drop('Scheduled Caste', axis=1),
df['Scheduled Caste'], test_size=0.2, random_state=42)

# Feature scaling

# The data set contains features with different scales, so we will scale them using the `StandardScaler()` class.

scaler = StandardScaler()

scaler.fit(X_train)

X_train = scaler.transform(X_train)

X_test = scaler.transform(X_test)

# Save the preprocessed data set

X_train.to_csv("X_train.csv", index=False)

X_test.to_csv("X_test.csv", index=False)
```

```
y_train.to_csv("y_train.csv", index=False)

y_test.to_csv("y_test.csv", index=False)
```

To start the data analysis by loading and pre-processing the dataset.

ALGORITHM:

Step 1: *Import libraries*

```
import pandas as pd

import matplotlib.pyplot as plt
```

Step 2: *Load the dataset.*

```
df =
pd.read_csv("C:\\Users\lenovo\Downloads\DDW_B06SC_3300_State_TAMIL_NADU-2011.csv")
```

Step 3: *Create a bar chart of the number of marginal workers by age group*

```
plt.figure(figsize=(10, 6))

df['Age'].value_counts().plot(kind='bar',
color=plt.cm.viridis.colors)

plt.xlabel('Age group')

plt.ylabel('Number of marginal workers')

plt.title('Number of marginal workers by age group')
```

```
plt.show()
```

Step 4: *Create a pie chart of the number of marginal workers by industrial category*

```
plt.figure(figsize=(10, 6))

df['Industrial category'].value_counts().plot(kind='pie', autopct="%1.1f%%")

plt.title('Percentage of marginal workers by industrial category')

plt.show()
```

Step 5: *Create a stacked bar chart of the number of marginal workers by sex and industrial category*

```
plt.figure(figsize=(10, 6))

df.groupby(['Industrial category', 'Sex'])['Age'].count().unstack().plot(kind='bar', stacked=True)

plt.xlabel('Industrial category')

plt.ylabel('Number of marginal workers')

plt.title('Number of marginal workers by sex and industrial category')

plt.legend()

plt.show()
```

PROGRAM:

```
import pandas as pd

import matplotlib.pyplot as plt


df =
pd.read_csv("C:\\Users\lenovo\Downloads\DDW_B06SC_330
0_State_TAMIL_NADU-2011.csv")


plt.figure(figsize=(10, 6))

df['Age'].value_counts().plot(kind='bar',
color=plt.cm.viridis.colors)

plt.xlabel('Age group')

plt.ylabel('Number of marginal workers')

plt.title('Number of marginal workers by age group')

plt.show()


plt.figure(figsize=(10, 6))

df['Industrial category'].value_counts().plot(kind='pie',
autopct="%1.1f%%")

plt.title('Percentage of marginal workers by industrial
category')

plt.show()


plt.figure(figsize=(10, 6))
```

```
df.groupby(['Industrial category',
'Sex'])['Age'].count().unstack().plot(kind='bar',
stacked=True)

plt.xlabel('Industrial category')

plt.ylabel('Number of marginal workers')plt.title('Number of
marginal workers by sex and industrial category')

plt.legend()

plt.show()
```