EXPERIMENT 1 : Setting up The Environment and Preprocessing The Data

AIM:

To set up a fully functional machine learning development environment and to perform data preprocessing operations like handling missing values, encoding categorical variables, feature scaling, and splitting datasets.

SOURCE CODE:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

import seaborn as sns

import matplotlib.pyplot as plt

df = sns.load_dataset('titanic')

display(df.head())

print(df.info())

print(df.describe())

print(df.isnull().sum())

df['age'].fillna(df['age'].median(), inplace=True)

df.drop(columns=['deck'], inplace=True)

le = LabelEncoder()

df['sex'] = le.fit_transform(df['sex'])

df.drop(columns=['embarked', 'class', 'who', 'alive', 'adult_male', 'alone', 'embark_town'],
inplace=True)

scaler = StandardScaler()

numerical_cols = ['age', 'fare']

df[numerical_cols] = scaler.fit_transform(df[numerical_cols])

X = df.drop('survived', axis=1)

y = df['survived']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("Training Data Shape:", X_train.shape)

print("Test Data Shape:", X_test.shape)

display(X_train.head())
```

OUTPUT:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype   
---  ------       --------------  -----   
 0   survived     891 non-null    int64   
 1   pclass       891 non-null    int64   
 2   sex          891 non-null    object  
 3   age          714 non-null    float64 
 4   sibsp        891 non-null    int64   
 5   parch        891 non-null    int64   
 6   fare         891 non-null    float64 
 7   embarked     889 non-null    object  
 8   class        891 non-null    category
 9   who          891 non-null    object  
 10  adult_male   891 non-null    bool    
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object  
 13  alive        891 non-null    object  
 14  alone        891 non-null    bool    
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
None
```

```
          survived      pclass         age       sibsp       parch        fare
count  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
mean     0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
std      0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
min      0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
25%      0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
50%      0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
75%      1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
max      1.000000    3.000000   80.000000    8.000000    6.000000  512.329200
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
Training Data Shape: (712, 6)
Test Data Shape: (179, 6)
/tmp/ipython-input-2443592645.py:24: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.

  df['age'].fillna(df['age'].median(), inplace=True)
```

	pclass	sex	age	sibsp	parch	fare
331	1	1	1.240235	0	0	-0.074583
733	2	1	-0.488887	0	0	-0.386671
382	3	1	0.202762	0	0	-0.488854
704	3	1	-0.258337	1	0	-0.490280
813	3	0	-1.795334	4	2	-0.018709