```python
#importing libraries
import numpy as np
import pandas as pd

Data = {'Name':['Alice', 'Bob', 'John','Lisa'],
        'Age':[21,20,22,23],
        'City':['San franscisco','New year','Los angeles','Chicago']}

df = pd.DataFrame(Data)
print(df)
```

```
    Name  Age            City
0  Alice   21  San franscisco
1    Bob   20        New year
2   John   22     Los angeles
3   Lisa   23         Chicago
```

```python
# Handling Missing Values
from sklearn.preprocessing import LabelEncoder
Data = {'Name':['Alice', 'Bob', 'John',None],
        'Age':[21,None,22,23],
        'City':['San franscisco','New year','Los angeles','Chicago']}

df = pd.DataFrame(Data)
print(df)

#Checking for missing value
print("Missing Values:\n",df.isnull())

#Dropping rows with missing values
df_cleaned = df.dropna()

#filling missing values with specified values
mean_values = df['Age'].mean()
df_filled = df.fillna(value={'Name':'unknown',
                             'Age': df['Age'].mean()})

print("\nDataframe after dropping missing values\n",df_cleaned)
print("\nDataframe after filling missing values\n",df_filled)

#encoding Categorical Data
label_encoder = LabelEncoder()
df['Encoded_city'] =
label_encoder.fit_transform(df['City'].astype(str))
print("\n After Encoding City")
print(df)
```

```
    Name   Age            City
0  Alice  21.0  San franscisco
1    Bob   NaN        New year
2   John  22.0     Los angeles
```

```
3   None  23.0         Chicago
Missing Values:
     Name    Age   City
0  False  False  False
1  False   True  False
2  False  False  False
3   True  False  False

Dataframe after dropping missing values
     Name   Age            City
0  Alice  21.0  San franscisco
2   John  22.0     Los angeles

Dataframe after filling missing values
       Name   Age            City
0    Alice  21.0  San franscisco
1      Bob  22.0        New year
2     John  22.0     Los angeles
3  unknown  23.0         Chicago

 After Encoding City
    Name   Age            City  Encoded_city
0  Alice  21.0  San franscisco             3
1    Bob   NaN        New year             2
2   John  22.0     Los angeles             1
3   None  23.0         Chicago             0
```

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

# Encoding Categorical Data
categories = ['red', 'blue', 'green', 'yellow', 'white']

label_encoder = LabelEncoder()
numeric_labels = label_encoder.fit_transform(categories)

print(categories)
print(numeric_labels)

# Creating DataFrame
data = {'Name': ['Alice', 'Bob', 'John', None],
        'Age': [21, None, 22, 23],
        'City': ['San Francisco', 'New York', 'Los Angeles',
'Chicago']}
df = pd.DataFrame(data)

# Encoding 'City' column
label_encoder = LabelEncoder()
df['City_Encoded'] = label_encoder.fit_transform(df['City'])
```

```
X = df[['Age', 'City_Encoded']]
y = df['Name']

# Splitting the data into training and test sets
x_train, x_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

print("Encoded DataFrame:")
print(df)
print("\nX_train:")
print(x_train)
print("\nX_test:")
print(x_test)
```

```
['red', 'blue', 'green', 'yellow', 'white']
[2 0 1 4 3]
Encoded DataFrame:
    Name   Age           City  City_Encoded
0  Alice  21.0  San Francisco             3
1    Bob   NaN       New York             2
2   John  22.0    Los Angeles             1
3   None  23.0        Chicago             0

X_train:
    Age  City_Encoded
3  23.0             0
0  21.0             3
2  22.0             1

X_test:
   Age  City_Encoded
1  NaN             2
```

```
# Independent variables (features)
X = df[['Name', 'Age', 'City']]

# Dependent variable (target)
y = df['City_Encoded']

print("Independent Variables (Features):\n", X)
print("Dependent Variable (Target):\n", y)
```

```
Independent Variables (Features):
     Name   Age           City
0  Alice  21.0  San Francisco
1    Bob   NaN       New York
2   John  22.0    Los Angeles
3   None  23.0        Chicago
Dependent Variable (Target):
 0    3
1    2
```

```
2    1
3    0
Name: City_Encoded, dtype: int64
```