EXPERIMENT 5

No: 5

Aim:

To demonstrate handling of missing values, encoding categorical data, and feature scaling using Python libraries such as Pandas and Scikit-learn.

Algorithm:

1. Import required libraries such as NumPy and Pandas.
2. Load the dataset using read_csv() and display it.
3. Fill missing categorical values using mode().
4. Handle missing numerical values using SimpleImputer with mean strategy.
5. Encode categorical variables using OneHotEncoder.
6. Combine the encoded and numerical data into one dataset.
7. Apply StandardScaler and MinMaxScaler for feature scaling.
8. Display the transformed datasets.

Code:

```
import numpy as np
import pandas as pd
df = pd.read_csv('pre_process_datasample - pre_process_datasample.csv')
df
```

Output:

Country	Age	Salary	Purchased
France	44.0	72000.0	No
Spain	27.0	48000.0	Yes
Germany	30.0	54000.0	No
Spain	38.0	61000.0	No
Germany	40.0	NaN	Yes
France	35.0	58000.0	Yes
Spain	NaN	52000.0	No

Country	Age	Salary	Purchased
France	48.0	79000.0	Yes
Germany	50.0	83000.0	No
France	37.0	67000.0	Yes

---

df.head()

Output:

Country	Age	Salary	Purchased
France	44.0	72000.0	No
Spain	27.0	48000.0	Yes
Germany	30.0	54000.0	No
Spain	38.0	61000.0	No
Germany	40.0	NaN	Yes

```
df["Country"] = df["Country"].fillna(df["Country"].mode()[0])
from sklearn.impute import SimpleImputer
features = df.iloc[:, :-1].values
label = df.iloc[:, -1].values

imputer = SimpleImputer(strategy="mean", missing_values=np.nan)
features[:, 1:3] = imputer.fit_transform(features[:, 1:3])
features
```

Output:

```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['Germany', 40.0, 63777.77777777778],
       ['France', 35.0, 58000.0],
```

['Spain', 38.77777777777778, 52000.0],

['France', 48.0, 79000.0],

['Germany', 50.0, 83000.0],

['France', 37.0, 67000.0]], dtype=object)

```
from sklearn.preprocessing import OneHotEncoder
oh = OneHotEncoder(sparse_output=False)
Country = oh.fit_transform(features[:, [0]])
Country
```

Output:

```
array([[1., 0., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [1., 0., 0.],
       [0., 0., 1.],
       [1., 0., 0.],
       [0., 1., 0.],
       [1., 0., 0.]])
```

```
final_set = np.concatenate((Country, features[:, [1, 2]]), axis=1)
final_set
```

Output:

```
array([[1.0, 0.0, 0.0, 44.0, 72000.0],
       [0.0, 0.0, 1.0, 27.0, 48000.0],
       [0.0, 1.0, 0.0, 30.0, 54000.0],
       [0.0, 0.0, 1.0, 38.0, 61000.0],
       [0.0, 1.0, 0.0, 40.0, 63777.77777777778],
       [1.0, 0.0, 0.0, 35.0, 58000.0],
       [0.0, 0.0, 1.0, 38.77777777777778, 52000.0],
```

[1.0, 0.0, 0.0, 48.0, 79000.0],

[0.0, 1.0, 0.0, 50.0, 83000.0],

[1.0, 0.0, 0.0, 37.0, 67000.0]], dtype=object)

```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

sc.fit(final_set)

feat_standard_scaler = sc.transform(final_set)

feat_standard_scaler
```

Output:

array([[1.22474487e+00, -6.54653671e-01, -6.54653671e-01, 7.58874362e-01, 7.49473254e-01],

[-8.16496581e-01, -6.54653671e-01, 1.52752523e+00, -1.71150388e+00, -1.43817841e+00],

[-8.16496581e-01, 1.52752523e+00, -6.54653671e-01, -1.27555478e+00, -8.91265492e-01],

[-8.16496581e-01, -6.54653671e-01, 1.52752523e+00, -1.13023841e-01, -2.53200424e-01],

[-8.16496581e-01, 1.52752523e+00, -6.54653671e-01, 1.77608893e-01, 6.63219199e-16],

[1.22474487e+00, -6.54653671e-01, -6.54653671e-01, -5.48972942e-01, -5.26656882e-01],

[-8.16496581e-01, -6.54653671e-01, 1.52752523e+00, 0.00000000e+00, -1.07356980e+00],

[1.22474487e+00, -6.54653671e-01, -6.54653671e-01, 1.34013983e+00, 1.38753832e+00],

[-8.16496581e-01, 1.52752523e+00, -6.54653671e-01, 1.63077256e+00, 1.75214693e+00],

[1.22474487e+00, -6.54653671e-01, -6.54653671e-01, -2.58340208e-01, 2.93712492e-01]])

```
from sklearn.preprocessing import MinMaxScaler

mms = MinMaxScaler(feature_range=(0,1))

mms.fit(final_set)

feat_minmax_scaler = mms.transform(final_set)
```

feat_minmax_scaler

Output:

```
array([[1.        , 0.        , 0.        , 0.73913043, 0.68571429],
       [0.        , 0.        , 1.        , 0.        , 0.        ],
       [0.        , 1.        , 0.        , 0.13043478, 0.17142857],
       [0.        , 0.        , 1.        , 0.47826087, 0.37142857],
       [0.        , 1.        , 0.        , 0.56521739, 0.45079365],
       [1.        , 0.        , 0.        , 0.34782609, 0.28571429],
       [0.        , 0.        , 1.        , 0.51207729, 0.11428571],
       [1.        , 0.        , 0.        , 0.91304348, 0.88571429],
       [0.        , 1.        , 0.        , 1.        , 1.        ],
       [1.        , 0.        , 0.        , 0.43478261, 0.54285714]])
```

Result:

Thus, the Python program to handle missing values, encode categorical data, and scale numerical features using Pandas and Scikit-learn was executed successfully and the output was verified.