EXPERIMENT NO: 5

Experiment to understand feature scaling.

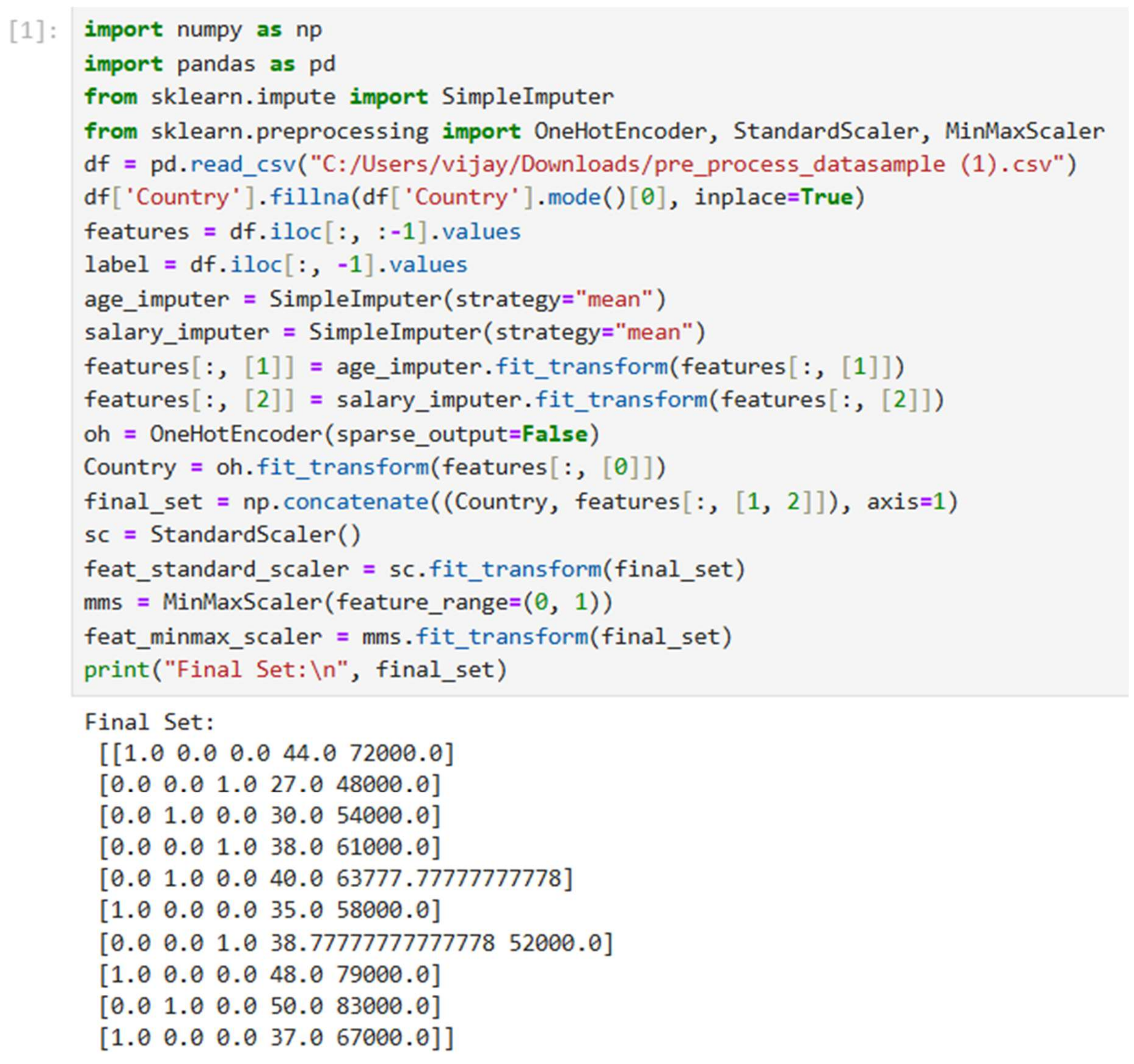
Aim:

To preprocess a dataset by handling missing values, encoding categorical variables, and scaling features using StandardScaler and MinMaxScaler.

Algorithm:

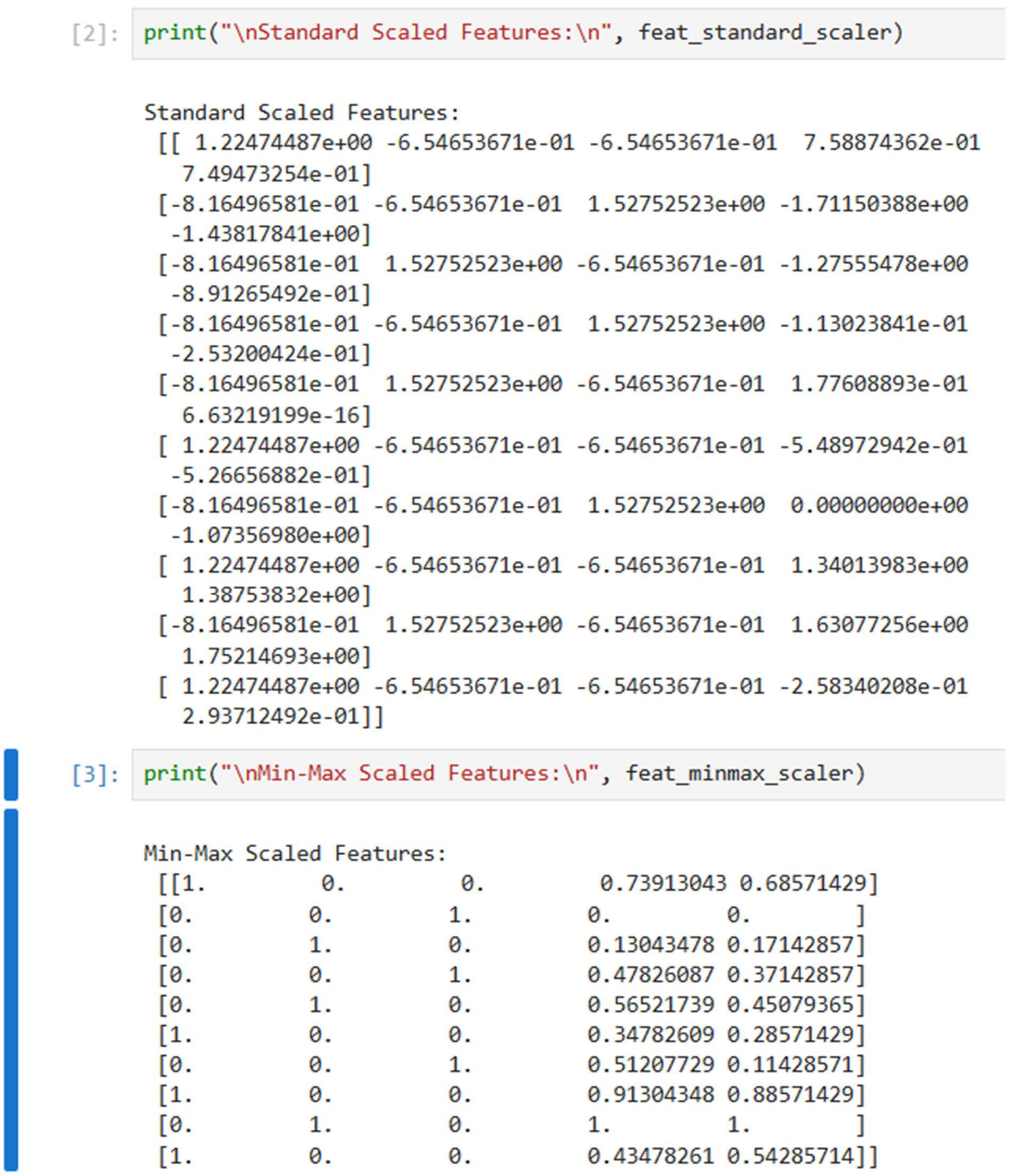
1) **Load Data: Read CSV file into a DataFrame.**
2) **Handle Missing Values: Fill missing Country values with mode; use mean imputation for Age and Salary.**
3) **Separate Features and Label: Extract features and target column.**
4) **Encode Categorical Data: Apply one-hot encoding to the Country column.**
5) **Combine Data: Concatenate encoded Country with numerical features (Age, Salary).**
6) **Scale Features:**
7) **Apply StandardScaler for mean-centered scaling.**
8) **Apply MinMaxScaler to scale features between 0 and 1.**
9) **Output: Print the final combined features and the scaled versions.**

Program:



```
[1]: import numpy as np
     import pandas as pd
     from sklearn.impute import SimpleImputer
     from sklearn.preprocessing import OneHotEncoder, StandardScaler, MinMaxScaler
     df = pd.read_csv("C:/Users/vijay/Downloads/pre_process_datasample (1).csv")
     df['Country'].fillna(df['Country'].mode()[0], inplace=True)
     features = df.iloc[:, :-1].values
     label = df.iloc[:, -1].values
     age_imputer = SimpleImputer(strategy="mean")
     salary_imputer = SimpleImputer(strategy="mean")
     features[:, [1]] = age_imputer.fit_transform(features[:, [1]])
     features[:, [2]] = salary_imputer.fit_transform(features[:, [2]])
     oh = OneHotEncoder(sparse_output=False)
     Country = oh.fit_transform(features[:, [0]])
     final_set = np.concatenate((Country, features[:, [1, 2]]), axis=1)
     sc = StandardScaler()
     feat_standard_scaler = sc.fit_transform(final_set)
     mms = MinMaxScaler(feature_range=(0, 1))
     feat_minmax_scaler = mms.fit_transform(final_set)
     print("Final Set:\n", final_set)
```

```
Final Set:
 [[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]]
```



```
[2]: print("\nStandard Scaled Features:\n", feat_standard_scaler)
```

```
Standard Scaled Features:
 [[ 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]]
```

```
[3]: print("\nMin-Max Scaled Features:\n", feat_minmax_scaler)
```

```
Min-Max Scaled Features:
 [[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:

The dataset is cleaned and numeric features are combined with one-hot encoded Country. Standard scaling centers the data, while Min-Max scaling normalizes it between 0 and 1, making it ready for machine learning.