Name : Suvidhi V. Pareek

Class / Div : TE / 03

Roll No : 09

| Experiment No.3 |
| --- |
| Perform data pre-processing |
| Date of Performance: 22/07/24 |
| Date of Submission: 29/07/24 |

**AIM:** To implement data preprocessing Algorithm

**Objective:-**Develop a program to implement data preprocessing algorithm

**Theory:**Why preprocess the data? Because data in the real world is dirty, incomplete and noisy. Incomplete in lacking attributes values and lacking attributes of interest or containing only aggregate value noisy in terms of containing errors or outliers and inconsistent containing discrepancies in names or codes. Now the question arises why is the data dirty? Because incomplete data may come from ―not applicable‖ data value when data has to be collected and the major issue is a different consideration between the times when the data was analyzed and human hardware and software issues are common. Noisy data may come from the when a human enters the wrong value at the time of data entry as Nobody is perfect. Errors in transmission of data and instruments that collect the faulty data. Inconsistent data may come from the different data sources. Duplicates records also need data cleaning.

Why data preprocessing is important? Data is not clean, Duplicity of data and the no quality data and the most important is no quality result so data preprocessing is important. Quality decisions must be based on the quality data. Data warehouse needs consistent integration of quality data. By the processing of data, data quality can be measures in term of accuracy, completeness, consistency, timeliness, believability, interpretability. There are three methods to handle the noisy data.

The different pre-processing steps that can be applied are:

1. Filling up the missing values
2. Removing duplicate data
3. Handling noisy data
4. Handling outliers
5. Scaling of data
6. Encoding of text or categorical values

**Code and output:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.impute import SimpleImputer

from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import OneHotEncoder, LabelEncoder

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.tree import DecisionTreeClassifier

# Load the dataset

dataset = pd.read\_csv("Data.csv")

x = dataset.iloc[:, :-1].values

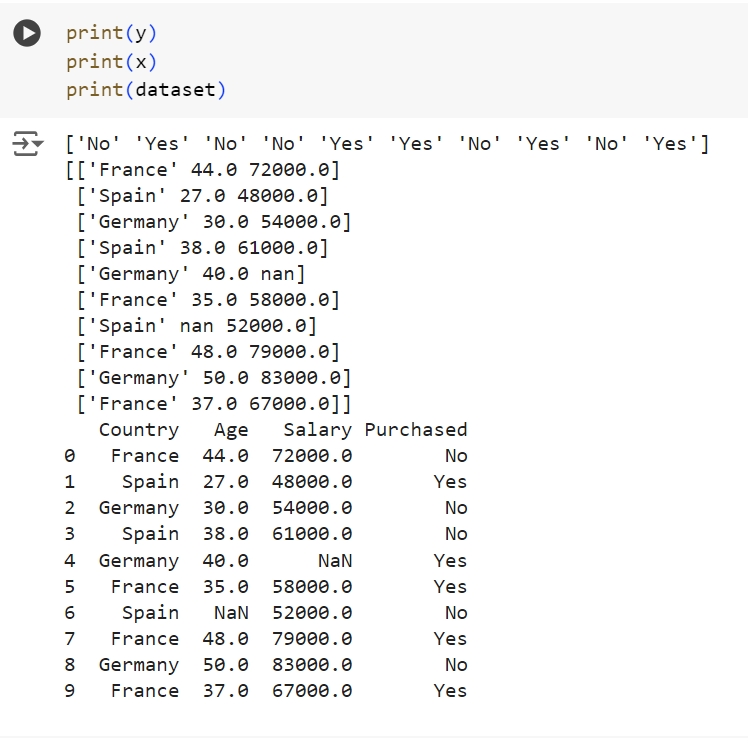
y = dataset.iloc[:, -1].values

# Print initial values

print(y)

print(x)

print(dataset)



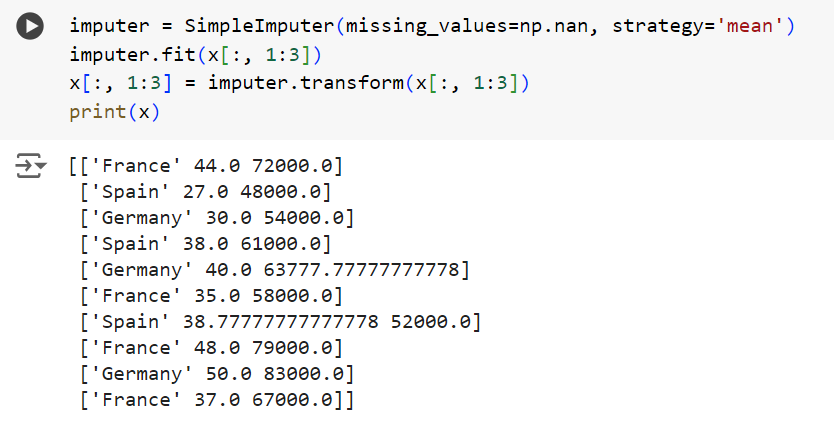
# Handle missing values

imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')

imputer.fit(x[:, 1:3])

x[:, 1:3] = imputer.transform(x[:, 1:3])

print(x)

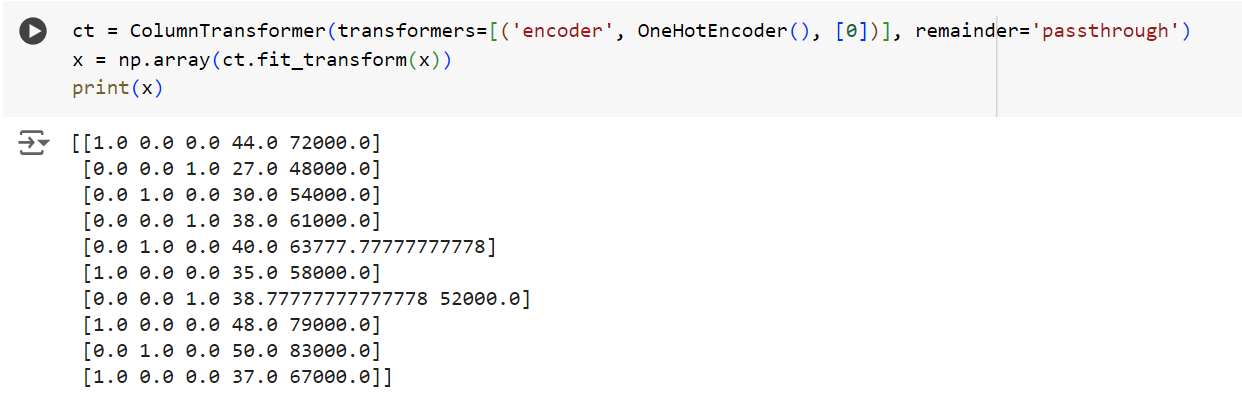


# Encode categorical data

ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [0])], remainder='passthrough')

x = np.array(ct.fit\_transform(x))

print(x)



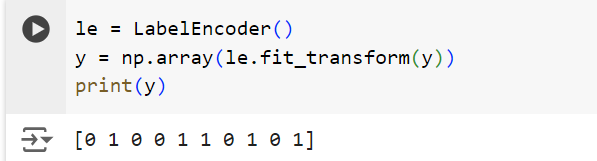
# Changed from X to x

# Encode the dependent variable

le = LabelEncoder()

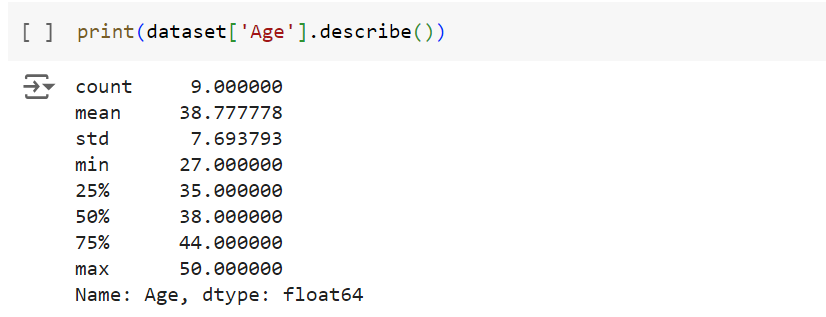
y = np.array(le.fit\_transform(y))

print(y)



# Display statistics of 'Age'

print(dataset['Age'].describe())



# Split the dataset into training and test sets

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=1)

# Feature scaling

sc = StandardScaler()

x\_train[:, 3:] = sc.fit\_transform(x\_train[:, 3:])

x\_test[:, 3:] = sc.transform(x\_test[:, 3:])

print(x\_test)

# Train a Decision Tree classifier

classifier = DecisionTreeClassifier(criterion='entropy', random\_state=0)

classifier.fit(x\_train, y\_train)

# Make predictions and evaluate the model

x\_predict = np.array(x\_test[:,:])

print(classifier.predict(x\_predict))

accuracy = classifier.score(x\_test, y\_test)

print(accuracy)

print(y\_test)



**Conclusion**: Comment on importance of pre-processing and what happens if the data is not pre-processed. Pre-processing is vital for cleaning and normalizing data, as unprocessed data can lead to inaccurate models and unreliable results.