Practical No-1

Date of Conduction: **Date of Checking:**

Data Wrangling, I

Perform the following operations using Python on any open source dataset (e.g., data.csv)

1. Import all the required Python Libraries.

2. Locate an open source data from the web (e.g. https://www.kaggle.com). Provide a clear description of the data and its source (i.e., URL of the web site).

3. Load the Dataset into pandas data frame.

4. Data Preprocessing: check for missing values in the data using pandas insult(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.

5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.

6. Turn categorical variables into quantitative variables in Python. In addition to the codes and outputs, explain every operation that you do in the above steps and explain everything that you do to import/read/scrape the data set.

Python Code:

 # 1. Import all the required Python Libraries.

```
import pandas as pd

import numpy as np
```

 # 2. Locate an open source data from the web.

```
# In this example, I'll use the Iris dataset available at UCI ML Repository.

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
```

 # 3. Load the Dataset into pandas data frame.

```
column_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'class']

iris_df = pd.read_csv(url, names=column_names)
```

```
# Display the first few rows of the dataset to verify the import.
print("First few rows of the Iris dataset:")
print(iris_df.head())

# 4. Data Preprocessing:
# Check for missing values using pandas info(), describe() functions.
print("\nInformation about the dataset:")
print(iris_df.info())

print("\nDescriptive statistics of the dataset:")
print(iris_df.describe())

# Variable Descriptions:
# - Sepal Length, Sepal Width, Petal Length, Petal Width: Numeric variables.
# - Class: Categorical variable representing the species of iris flowers.

# Check the dimensions of the data frame.
print("\nDimensions of the dataset (rows, columns):", iris_df.shape)

# 5. Data Formatting and Normalization:
# Summarize the types of variables by checking data types.
print("\nData Types of Variables:")
print(iris_df.dtypes)

# Ensure that numeric variables are in the correct data type.
# In this case, they are already in the correct data types (float64).

# 6. Turn categorical variables into quantitative variables.
# The 'class' variable is categorical; we can use one-hot encoding to convert it to quantitative.
iris_df = pd.get_dummies(iris_df, columns=['class'], drop_first=True)
```

```
# Display the updated dataframe.
print("\nUpdated DataFrame after one-hot encoding:")
print(iris_df.head())
```

Explanation:

- The code starts by importing necessary libraries, including Pandas for data manipulation and NumPy for numerical operations.
- The dataset URL is specified, and the read_csv function from Pandas is used to load the dataset into a Pandas DataFrame.
- The info() and describe() functions are used to obtain initial statistics and check for missing values.
- Variable descriptions are provided, and the dimensions of the DataFrame are printed.
- The data types of variables are displayed using dtypes.
- The 'class' variable is categorical, so one-hot encoding is applied using pd.get_dummies() to convert it into quantitative variables.
- The updated DataFrame is displayed.

Output:

```
"C:\Users\Ram Kumar Solanki\PycharmProjects\pythonProject\venv\Scripts\python.exe" "C:\Users\Ram Kumar Solanki\PycharmProjects\MBA_BFS\main.py"
First few rows of the Iris dataset:
   sepal_length  sepal_width  petal_length  petal_width        class
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa

Information about the dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
```

```
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   class         150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
None

Descriptive statistics of the dataset:
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000

Dimensions of the dataset (rows, columns): (150, 5)

Data Types of Variables:
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
class           object
```

dtype: object

Updated DataFrame after one-hot encoding:

	sepal_length	sepal_width	...	class_Iris-versicolor	class_Iris-virginica
0	5.1	3.5	...	False	False
1	4.9	3.0	...	False	False
2	4.7	3.2	...	False	False
3	4.6	3.1	...	False	False
4	5.0	3.6	...	False	False

[5 rows x 6 columns]

Process finished with exit code 0

Date :

Name &Signature of Instructor

Dr. Ram Kumar Solanki