Machine Learning Lab Sheet 01

Step 1: Setting up Google Colab

- Create a new notebook by visiting Google Colab https://colab.research.google.com/.

Step 1: Working with Libraries

Code Example 1: Importing Libraries

```
# Importing common libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Step 2: Loading and Manipulating Data

Code Example 2: Loading and Exploring Data with Pandas

```
# Upload a sample CSV file
from google.colab import files
uploaded = files.upload()
```

Code Example 3: Loading and Exploring Data with Pandas as a data frame

```
# import pandas module
import pandas as pd
   # making dataframe using the uploaded csv dataset
df = pd.read_csv("/content/missingdata.csv")
  # output the dataframe
print(df)
df.head()
```

Step3: Perform this example about the data that you have uploaded

```
# Import the Pandas library
import pandas as pd
# Create a sample DataFrame
data = {'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva'],
       'Age': [25, 32, 19, 45, 28],
       'Salary': [50000, 60000, 45000, 75000, 55000]}
df = pd.DataFrame(data)
print(df)
```

Code Example 4: **Using info() to get information about the DataFrame's structure and data types**

```
print("\nDataFrame information:")
df.info()
```

Code Example 5: **Using describe() to get summary statistics of the numeric columns**

```
print("\nSummary statistics of numeric columns:")
print(df.describe())
```

Step 4: Data Preprocessing: Removing missing values

```
# Data preprocessing: Removing missing values
data.dropna(inplace=True)

# Data preprocessing: Removing Duplicated records
# Remove duplicated records based on all columns
data = data.drop_duplicates()

# If you want to keep the first occurrence of each duplicated record and remove the rest, you can use the 'subset' and 'keep' parameters:

data = data.drop_duplicates(subset=None, keep='first')

# 'subset' allows you to specify a subset of columns to consider for duplicate removal. For example, if you only want to consider duplicates based on the 'column1' and 'column2' columns:

data = data.drop_duplicates(subset=['column1', 'column2'], keep='first')



# Perform the following code example that demonstrates how to handle missing data in a Pandas DataFrame by removing rows with missing values using dropna() and imputing missing values using the mean and forward-fill method with fillna():

import pandas as pd
import numpy as np

# Create a sample DataFrame with missing data
data = {
    'A': [1, 2, np.nan, 4, 5],
    'B': [np.nan, 2, 3, 4, np.nan],
    'C': [1, 2, 3, 4, 5]
}

df = pd.DataFrame(data)

# Display the original DataFrame
print("Original DataFrame:")
print(df)

# Removing rows with missing values using dropna()
df_dropna = df.dropna()

# Display the DataFrame after removing rows with missing values
print("\nDataFrame after removing rows with missing values:")
print(df_dropna)

# Imputing missing values with the mean value using fillna()
df_fill_mean = df.fillna(df.mean())

# Display the DataFrame after imputing missing values with the mean
print("\nDataFrame after imputing missing values with mean:")
print(df_fill_mean)

# Imputing missing values with forward fill (method='ffill') using fillna()
df_fill_ffill = df.fillna(method='ffill')

# Display the DataFrame after imputing missing values with forward fill
print("\nDataFrame after imputing missing values with forward fill:")
print(df_fill_ffill)
```

Step 5: IRIS dataset example

```
import pandas as pd
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt

# Load the Iris dataset
iris = load_iris()

# Create a DataFrame from the dataset
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
iris_df['target'] = iris.target

 #Display the first few rows of the dataset:

iris_df.head()
 # Get basic summary statistics of the dataset:
iris_df.describe()
  #Check the data types of each column:
iris_df.dtypes
#Check for missing values:
```

```
iris_df.isnull().sum()

# Count the unique values in the target column (species):
iris_df['target'].value_counts()
```

Step3:Data Transformation: Transform data through operations like feature scaling (e.g., Min-Max scaling, Z-score normalization), one-hot encoding.

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler,
LabelEncoder, OneHotEncoder

# Create a sample DataFrame with different data types
data = {
   'Age': [25, 32, 19, 45, 28],
   'Salary': [50000, 60000, 45000, 75000, 55000],
   'Gender': ['Male', 'Female', 'Male', 'Male', 'Female']
}

df = pd.DataFrame(data)

# Display the original DataFrame
print("Original DataFrame:")
print(df)

# Min-Max Scaling for 'Age' and 'Salary' columns
scaler = MinMaxScaler()
df[['Age', 'Salary']] = scaler.fit_transform(df[['Age', 'Salary']])

# Display the DataFrame after Min-Max scaling
print("\nDataFrame after Min-Max scaling for 'Age' and 'Salary' columns:")
print(df)

# Z-score Normalization for 'Age' and 'Salary' columns
scaler = StandardScaler()
df[['Age', 'Salary']] = scaler.fit_transform(df[['Age', 'Salary']] )

# Display the DataFrame after Z-score normalization
print("\nDataFrame after Z-score normalization for 'Age' and 'Salary'
columns:")
print(df)
```

```
# One-Hot Encoding for 'Gender' column
df = pd.get_dummies(df, columns=['Gender'], prefix=['Gender'])

# Display the DataFrame after one-hot encoding
print("\nDataFrame after one-hot encoding for 'Gender' column:")
print(df)

# perform the previous Data Transformation techniques for the iris dataset.
```

Save your Colab notebook for the Lab sheet-01 as : Yourname-LabSheet01 and upload it to the course Moodle assignment page.