Practical-2

AIM : Introduction to reproducible Machine Learning Operations

The aim of the practial is to get the hands-on experience of reproducing the machine learning operations at each stage. Student needs to apply the following steps in the practical.

**Step 1 :**

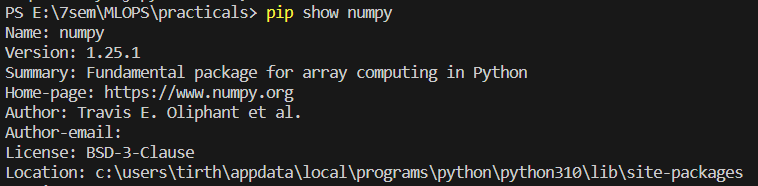
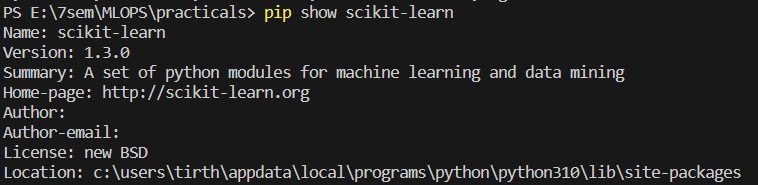
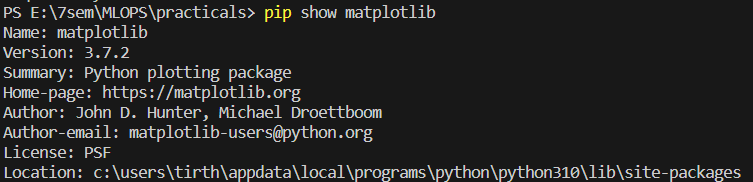
Ensure that the numpy, scikit learn, and matplotlib libraries are available in your system. Create the requirements.txt file and make a note of the versions of these libraries.

**To check the Versions of Libraries** :

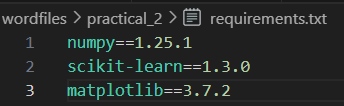
pip show numpy

pip show scikit-learn

pip show matplotlib

**Create requirements.txt:**

****

**Step 2 :**

1. **Import the data and scale it using StandardScaler:**

import numpy as np

from sklearn.preprocessing import StandardScaler

import joblib

# Load the data from sample.csv using np.genfromtxt

data = np.genfromtxt('sample.csv', delimiter=',')

# # Load the data from Sample.txt

# data = np.loadtxt('sample.csv')

# Create a StandardScaler object and fit\_transform the data

scaler = StandardScaler()

scaled\_data = scaler.fit\_transform(data)

# Store the scaler object for reproducibility

joblib.dump(scaler, 'scaler\_object.joblib')

# Now scaled\_data contains the normalized dataset

1. **Splitting the normalized data:**

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

# Split the data into training and testing sets

train\_data, test\_data = train\_test\_split(scaled\_data, test\_size=0.2, random\_state=42)

# train\_data and test\_data now contain the training and testing datasets

1. **Storing the snapshot of the data as a numpy file:**

# Save the datasets as numpy files

np.save('train\_data.npy', train\_data)

np.save('test\_data.npy', test\_data)

**Step 3 :**

Apply the linear regression algorithm on the dataset and assess the prediction on the test dataset.

**a) Store the trained model into the local file system to ensure the reproducibility of the**

**prediction. Import the model and the test dataset into other python file. Check whether the same prediction is obtained in the latter case.**

from sklearn.linear\_model import LinearRegression

import joblib

# Assume you already have the train\_data and test\_data loaded

# Separate features and target variable

X\_train = train\_data[:, :-1]

y\_train = train\_data[:, -1]

# Create and train the linear regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Assess the prediction on the test dataset

X\_test = test\_data[:, :-1]

y\_test = test\_data[:, -1]

# Make predictions

predictions = model.predict(X\_test)

# Store the trained model for reproducibility

joblib.dump(model, 'linear\_regression\_model.joblib')

# Now, in another Python file, you can load the model and test data for prediction

# Load the model

loaded\_model = joblib.load('linear\_regression\_model.joblib')

# Make predictions on the test data

loaded\_predictions = loaded\_model.predict(X\_test)

# Check if the predictions are the same

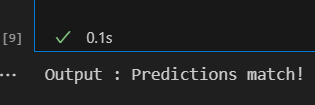
if np.array\_equal(predictions, loaded\_predictions):

    print("Output : Predictions match!")

else:

    print("Output : Predictions differ!")

**Output :**

****