Practical-2

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.

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.

```
Python version: 3.10.12
Numpy Version :  1.22.4
Pandas Version :  1.5.3
Skcit-learn Version :  1.2.2
Matplotlib Version :  3.7.1
Pickle version: 4.0
```

2. Write a python code to import the Sample.txt data. Further, apply the following processes on the imported data.
   a) Scale the dataset. Use the StandardScalar function of scikit learn to normalize the dataset. Ensure reproduciblity: Store the standard scalar object into your local file system, so that the same data normalization can be applied to the other data during the deployement.
   b) Split the data: Write the python code to split the normalized data into randomly selected proportion as per the constant ratio. For e.g., if the constant ratio is 0.8, then the code must randomly select the 80 % proportion of the data as the training dataset and remaining 20 % as the testing dataset.
   c) Store the snapshot of the data as the numpy file. Ensure the same dataset could be loaded into separate python code. (Extra question: how to ensure that the same random generation could be achieved on each execution.)

```
x =df.iloc[:,0].values
y=df.iloc[:,-1].values
```

```
x_train,x_test,y_train,y_test =train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_train = x_train.reshape(len(x_train),1)
x_test = x_test.reshape(len(x_test),1)
```

```
np.save("x_test",x_test)
np.save("y_test",y_test)
```

```
scaler =StandardScaler()
scaler.fit(x_train)
X_train_scaled = scaler.fit_transform(x_train)
X_test_scaled = scaler.transform(x_test)
```

```
regressor = LinearRegression()
regressor.fit(X_train_scaled,y_train)
```

```
LinearRegression()
```

```
y_pred = regressor.predict(X_test_scaled)
```

```
from sklearn.metrics import r2_score
```

```
print(r2_score(y_test,y_pred))
```

```
0.8018504692529483
```

```
import pickle
pickle.dump(scaler,open("scaler","wb"))
pickle.dump(regressor,open("regressor","wb"))
```

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.

In [ ]:
```
import pickle
scaler_model = pickle.load(open('scaler', 'rb'))
regressor_model = pickle.load(open('regressor', 'rb'))
```

In [ ]:
```
import numpy as np
```

In [ ]:
```
x_test =np.load("/content/x_test.npy")
```

In [ ]:
```
y_test = np.load("/content/y_test.npy")
```

In [ ]:
```
x_test = scaler_model.transform(x_test)
```

In [ ]:
```
y_pred = regressor_model.predict(x_test)
```

In [ ]:
```
from sklearn.metrics import r2_score
print(r2_score(y_test,y_pred))
```

```
0.8018504692529483
```