```
import os
import numpy as np

import tensorflow as tf
from matplotlib import pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix ,
classification_report
```

IMPORTING DONE

SETTING UP VARIABLES

```
TrainingDirect = os.path.join('Data','training')
trainaccnpy = os.path.join(TrainingDirect,'trainACC.npy')
traingyronpy = os.path.join(TrainingDirect,'trainGYRO.npy')
trainlabelsnpy = os.path.join(TrainingDirect,'trainLABELS.npy')

TestingDirect = os.path.join('Data' , 'testing')
testaccnpy = os.path.join(TestingDirect,'testACC.npy')
testgyronpy = os.path.join(TestingDirect,'testGYRO.npy')
testlabelsnpy = os.path.join(TestingDirect,'testLABELS.npy')
```

VARIABLES SET UP COMPLETE

READING NUMPY FILES AND COMBINING

```
trainXacc = np.load(trainaccnpy)
trainXgyro = np.load(traingyronpy)
trainYlabels = np.load(trainlabelsnpy)

testXacc= np.load(testaccnpy)
testXgyro = np.load(testgyronpy)
testYlabels = np.load(testlabelsnpy)

#combining acc and gyro variables
trainX = np.concatenate([trainXacc,trainXgyro] , axis=2)
testX = np.concatenate([testXacc,testXgyro] , axis=2)
print(testX.shape)
trainX.shape
```

```
(90, 268, 6)
```

```
(87, 268, 6)
```

FILES READ AND COMBINED

NORMALIZIGN AND EXTRACTING FEATURES

```
trainXreshaped= trainX.reshape(-1,trainX.shape[-1])
testXreshaped = testX.reshape(-1,testX.shape[-1])
trainXreshaped.shape
```

```
(23316, 6)
```

```
trainXscaled = StandardScaler().fit_transform(trainXreshaped)
testXscaled = StandardScaler().fit_transform(testXreshaped)
trainXscaled.shape
```

```
(23316, 6)
```

```
def Extractor(xyz):
    mn=np.mean(xyz,axis=1,)
    st=np.std(xyz,axis=1,)
    mx=np.max(xyz,axis=1,)
    mns=np.min(xyz,axis=1,)

    ext=np.concatenate([mn,st,mx,mns],axis=1)

    return ext


trainXfeatures = Extractor(trainX)
testXfeatures = Extractor(testX)

trainYlabels.shape
```

```
(87,)
```

```
scaler = StandardScaler()
trainXfeaturesscaled =scaler.fit_transform(trainXfeatures)
testXfeaturesscaled=scaler.fit_transform(testXfeatures)
trainXfeaturesscaled.shape
```

```
(87, 24)
```

```
trainYlabels.shape
```

```
(87,)
```

FINISHED NORMALIZING AND EXTRACTING

APPLYING LOGISTIC REGRESSION

```
LogReg = LogisticRegression(max_iter=3000)
LogReg.fit(trainXfeaturesscaled, trainYlabels)
```

```
predLogReg = LogReg.predict(testXfeaturesscaled)
predLogReg.shape
```

```
(90,)
```

LOGICTIC REGRESSION COMPLETED

Using Forest Classifier

```
RanClass = RandomForestClassifier()
RanClass.fit(trainXfeaturesscaled, trainYlabels)
predRanClass = RanClass.predict(testXfeaturesscaled)
```

DONE USING FOREST CLASSIFIER

GENERATING REPORTS

```
print("Logistic Regression Accuracy:", accuracy_score(testYlabels,
predLogReg))
print("Confusion Matrix of Logistic Regression :",
confusion_matrix(testYlabels, predLogReg))
print()
```

```
Logistic Regression Accuracy: 0.9888888888888889
Confusion Matrix of Logistic Regression : [[47  1]
 [ 0 42]]
```

```
print("Random Forest Classifier Accuracy:",
accuracy_score(testYlabels, predRanClass))
print("Confusion Matrix of Random Forest Classifier :",
confusion_matrix(testYlabels, predLogReg))
```

```
Random Forest Classifier Accuracy: 0.9444444444444444
Confusion Matrix of Random Forest Classifier : [[47  1]
 [ 0 42]]
```