In [3]:
```
#import the requred libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
```

In [4]:
```
#if it will give any wanrting then it will be ignored
import warnings
warnings.filterwarnings('ignore')
```

In [5]:
```
df=pd.read_csv("/home/osboxes/Desktop/Fall_Deteciton_Dataset.csv")
df
```

Out[5]:

	ACTIVITY	TIME	SL	EEG	BP	HR	CIRCLUATION
0	Falling	4722.92	4019.64	-1600.00	13.0	79.0	317
1	Sitting	4059.12	2191.03	-1146.08	20.0	54.0	165
2	Sitting	4773.56	2787.99	-1263.38	46.0	67.0	224
3	Cramps	8271.27	9545.98	-2848.93	26.0	138.0	554
4	Cramps	7102.16	14148.80	-2381.15	85.0	120.0	809
...	...	...	...	...	...	...	...
16377	Cramps	9280.68	11417.00	-3021.64	36.0	156.0	654
16378	Falling	8479.69	9455.54	-2932.85	17.0	138.0	554
16379	Sitting	8872.53	27449.90	-2870.00	33.0	156.0	1364
16380	Cramps	7738.99	26466.40	-2920.24	97.0	156.0	1521
16381	Falling	9368.34	39149.10	-2970.00	21.0	196.0	1885

16382 rows × 7 columns

In [6]:
```
x=df.iloc[:,1:6]
y=df.iloc[:,-1]
x
```

Out[6]:

	TIME	SL	EEG	BP	HR
0	4722.92	4019.64	-1600.00	13.0	79.0
1	4059.12	2191.03	-1146.08	20.0	54.0
2	4773.56	2787.99	-1263.38	46.0	67.0
3	8271.27	9545.98	-2848.93	26.0	138.0
4	7102.16	14148.80	-2381.15	85.0	120.0
...	...	...	...	...	...
16377	9280.68	11417.00	-3021.64	36.0	156.0
16378	8479.69	9455.54	-2932.85	17.0	138.0

	TIME	SL	EEG	BP	HR
16379	8872.53	27449.90	-2870.00	33.0	156.0
16380	7738.99	26466.40	-2920.24	97.0	156.0
16381	9368.34	39149.10	-2970.00	21.0	196.0

In [7]:
```
y
```

Out[7]:
```
0         317
1         165
2         224
3         554
4         809
         ...
16377     654
16378     554
16379    1364
16380    1521
16381    1885
Name: CIRCLUATION, Length: 16382, dtype: int64
```

In [8]:
```
print("any null values: ",df.isnull().values.any())
print("null values: ",df.isnull().sum())
```

```
any null values:  True
null values:  ACTIVITY       0
TIME           0
SL             0
EEG            0
BP             2
HR             1
CIRCLUATION    0
dtype: int64
```

In [9]:
```
df.keys()
```

Out[9]:
```
Index(['ACTIVITY', 'TIME', 'SL', 'EEG', 'BP', 'HR', 'CIRCLUATION'], dt
ype='object')
```

In [13]:
```
from sklearn.impute import SimpleImputer

#Imputer object using the mean strategy and missing values types for im

imputer=SimpleImputer(missing_values=np.nan,strategy='mean')

#fitting the data to the imputer object

imputer=imputer.fit(x)

#imputing the data

x=imputer.transform(x)
print("imputed data: ",x)
```

```
imputed data:  [[ 4.72292e+03  4.01964e+03 -1.60000e+03  1.30000e+01
7.90000e+01]
 [ 4.05912e+03  2.19103e+03 -1.14608e+03  2.00000e+01  5.40000e+01]
 [ 4.77356e+03  2.78799e+03 -1.26338e+03  4.60000e+01  6.70000e+01]
 ...
 [ 8.87253e+03  2.74499e+04 -2.87000e+03  3.30000e+01  1.56000e+02]
 [ 7.73899e+03  2.64664e+04 -2.93924e+03  9.70000e+01  1.56000e+02]
```

In [14]:

```
print("any null values: ",df.isnull().values.any())
print("null values: ",df.isnull().sum())
```

```
any null values:  True
null values:  ACTIVITY       0
TIME          0
SL            0
EEG           0
BP            2
HR            1
CIRCLUATION    0
dtype: int64
```

In [ ]:

```
df
```

In [15]:

```
from sklearn.preprocessing import StandardScaler
en_sc=StandardScaler()
x=en_sc.fit_transform(x)
```

In [16]:

```
#split the data into train and test part
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,rando
```

In [17]:

```
#craete object of classifer
model=RandomForestClassifier(n_estimators=100)
```

In [81]:

```
#train the model using traing set
model.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
Input In [81], in <cell line: 2>()
      1 #train the model using traing set
----> 2 model.fit(x_train,y_train)

File ~/anaconda3/lib/python3.9/site-packages/sklearn/ensemble/_forest.
py:327, in BaseForest.fit(self, X, y, sample_weight)
    325 if issparse(y):
    326     raise ValueError("sparse multilabel-indicator for y is not
supported.")
--> 327 X, y = self._validate_data(
    328     X, y, multi_output=True, accept_sparse="csc", dtype=DTYPE
    329 )
    330 if sample_weight is not None:
    331     sample_weight = _check_sample_weight(sample_weight, X)

File ~/anaconda3/lib/python3.9/site-packages/sklearn/base.py:581, in B
aseEstimator._validate_data(self, X, y, reset, validate_separately, **
check_params)
    579         y = check_array(y, **check_y_params)
    580     else:
--> 581         X, y = check_X_y(X, y, **check_params)
    582     out = X, y
    584 if not no_val_X and check_params.get("ensure_2d", True):

File ~/anaconda3/lib/python3.9/site-packages/sklearn/utils/validation.
py:964, in check_X_y(X, y, accept_sparse, accept_large_sparse, dtype,
order, copy, force_all_finite, ensure_2d, allow_nd, multi_output, ensu
re_min_samples, ensure_min_features, y_numeric, estimator)
    961 if y is None:
    962     raise ValueError("y cannot be None")
--> 964 X = check_array(
    965     X,
    966     accept_sparse=accept_sparse,
    967     accept_large_sparse=accept_large_sparse,
    968     dtype=dtype,
    969     order=order,
    970     copy=copy,
    971     force_all_finite=force_all_finite,
    972     ensure_2d=ensure_2d,
    973     allow_nd=allow_nd,
    974     ensure_min_samples=ensure_min_samples,
    975     ensure_min_features=ensure_min_features,
    976     estimator=estimator,
    977 )
    979 y = _check_y(y, multi_output=multi_output, y_numeric=y_numeri
c)
    981 check_consistent_length(X, y)

File ~/anaconda3/lib/python3.9/site-packages/sklearn/utils/validation.
py:800, in check_array(array, accept_sparse, accept_large_sparse, dtyp
e, order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_samp
les, ensure_min_features, estimator)
```

```
    794             raise ValueError(
    795                 "Found array with dim %d. %s expected <= 2."
    796                 % (array.ndim, estimator_name)
    797             )
    799         if force_all_finite:
--> 800             _assert_all_finite(array, allow_nan=force_all_finite =
= "allow-nan")
    802 if ensure_min_samples > 0:
    803     n_samples = _num_samples(array)
```

In [ ]:

In [ ]: