In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.preprocessing import StandardScaler
from IPython.display import display, HTML
import re
```

In [2]:
```
data = pd.read_csv('human_motion_detection.csv', delimiter = ";")
df = data.copy()
```

In [3]:
```
def getting_primary_info(df):
    print("-------------------------------------------------------")
    print("Veri setinin şekli", df.shape)
    print("-------------------------------------------------------")
    print("Veri seti değişken tipleri:\n", df.dtypes)
    print("-------------------------------------------------------")
    print("Veri setinin ilk 5 satırı")
    display(HTML(df.head().to_html()))
    print("-------------------------------------------------------")
    print("Veri setinin istatistiki verileri")
    description = df.describe()
    display(HTML(description.to_html()))
    print("-------------------------------------------------------")
getting_primary_info(df)
```

```
-------------------------------------------------------
Veri setinin şekli (37161, 13)
-------------------------------------------------------
Veri seti değişken tipleri:
 gyro_x          float64
gyro_y          float64
gyro_z          float64
accel_x         float64
accel_y         float64
accel_z         float64
std_acc_30      float64
std_gyro_10     float64
mean_acc_20     float64
mean_gyro_20    float64
max_acc_15      float64
min_acc_20      float64
Output           object
dtype: object
-------------------------------------------------------
Veri setinin ilk 5 satırı
```

	gyro_x	gyro_y	gyro_z	accel_x	accel_y	accel_z	std_acc_30	std_gyro_10	mean_acc_20	mean_gyro_20	max_acc_15	min_acc_20	Output
0	0.49875	-0.64750	0.13125	0.685396	-0.630008	0.383141	0.0	0.0	0.0	0.0	0.0	0.0	sit
1	0.47250	-0.72625	0.12250	0.684420	-0.630191	0.383690	0.0	0.0	0.0	0.0	0.0	0.0	sit
2	0.39375	-0.63875	0.12250	0.687531	-0.629764	0.383507	0.0	0.0	0.0	0.0	0.0	0.0	sit
3	0.35875	-0.65625	0.09625	0.686616	-0.628971	0.384056	0.0	0.0	0.0	0.0	0.0	0.0	sit
4	0.29750	-0.60375	0.14000	0.685640	-0.631594	0.382714	0.0	0.0	0.0	0.0	0.0	0.0	sit

```
-------------------------------------------------------
Veri setinin istatistiki verileri
```

	gyro_x	gyro_y	gyro_z	accel_x	accel_y	accel_z	std_acc_30	std_gyro_10	mean_acc_20	mean_gyro_20	max_acc_15	min_acc_20
count	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000	37161.000000
mean	20.783674	-13.797805	25.534451	0.474828	-0.455960	0.562330	0.583307	25.162510	0.193229	10.842123	0.846625	-0.681360
std	41.377206	40.438210	46.621772	0.568782	0.483283	0.375338	0.295393	26.719618	0.135518	27.624567	0.429346	0.430828
min	-94.307503	-277.156250	-47.206249	-0.870000	-1.998604	-0.330925	0.000000	0.000000	-0.111540	-64.270000	-0.019581	-1.998604
25%	0.113750	-31.631250	-0.131250	0.038369	-0.630008	0.378078	0.480000	3.070000	0.135489	-0.970083	0.686128	-0.710000
50%	6.387500	-2.180000	2.563750	0.590000	-0.529663	0.504775	0.562331	14.239748	0.159617	1.589729	0.750000	-0.630000
75%	43.426250	2.310000	41.186249	0.692594	-0.112789	0.830000	0.611720	41.360000	0.250000	17.637667	1.012600	-0.460672
max	360.865002	208.390000	214.112503	1.998604	0.620000	1.998604	1.765029	125.698103	0.710211	119.218750	1.998604	0.134261

```
-------------------------------------------------------
```

In [5]:
```
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier

X, y = df.drop(columns="Output"), df["Output"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=47)

gb_model = GradientBoostingClassifier()
gb_model.fit(X_train, y_train)

rf_model = RandomForestClassifier()
rf_model.fit(X_train, y_train)

y_pred_gb = gb_model.predict(X_test)
accuracy_gb = accuracy_score(y_test, y_pred_gb)
print("Gradient Boosting Test Accuracy:", accuracy_gb)

y_pred_gb_train = gb_model.predict(X_train)
accuracy_gb_train = accuracy_score(y_train, y_pred_gb_train)
print("Gradient Boosting Train Accuracy:", accuracy_gb_train)
print("----------------------------")

y_pred_rf = rf_model.predict(X_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print("Random Forest Test Accuracy:", accuracy_rf)

y_pred_rf_train = rf_model.predict(X_train)
accuracy_rf_train = accuracy_score(y_train, y_pred_rf_train)
print("Random Forest Train Accuracy:", accuracy_rf_train)

tree_model = DecisionTreeClassifier()
tree_model.fit(X_train, y_train)
print("----------------------------")
y_pred = tree_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Decision Tree Test Accuracy:", accuracy)


tree_model.fit(X_train, y_train)

y_pred_train = tree_model.predict(X_train)
accuracy_train = accuracy_score(y_train, y_pred_train)
print("Decision Tree Train Accuracy:", accuracy_train)
```

```
Gradient Boosting Test Accuracy: 0.9982510426476524
Gradient Boosting Train Accuracy: 0.9997645317545748
----------------------------
Random Forest Test Accuracy: 0.9989237185524015
Random Forest Train Accuracy: 1.0
----------------------------
```

```
Decision Tree Test Accuracy: 0.9965020852953047
Decision Tree Train Accuracy: 1.0
```

In [ ]: