DESCRIPTION

You are supposed to detect whether the person is running or walking based on the sensor data collected from iOS device. The dataset contains a single file which represents sensor data samples collected from accelerometer and gyroscope from iPhone 5c in 10 seconds interval and ~5.4/second frequency.

Objective: Practice classification based on Naive Bayes algorithm. Identify the predictors that can be influential.

Actions to Perform:

Load the kinematics dataset as measured on mobile sensors from the file “run_or_walk.csv.” List the columns in the dataset. Let the target variable “y” be the activity, and assign all the columns after it to “x.” Using Scikit-learn, fit a Gaussian Naive Bayes model and observe the accuracy. Generate a classification report using Scikit-learn. Repeat the model once using only the acceleration values as predictors and then using only the gyro values as predictors. Comment on the difference in accuracy between both models.

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plot
%matplotlib inline
```

In [2]:

```
df = pd.read_csv("run_or_walk.csv")
```

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88588 entries, 0 to 88587
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   date            88588 non-null  object 
 1   time            88588 non-null  object 
 2   username        88588 non-null  object 
 3   wrist           88588 non-null  int64  
 4   activity        88588 non-null  int64  
 5   acceleration_x  88588 non-null  float64
 6   acceleration_y  88588 non-null  float64
 7   acceleration_z  88588 non-null  float64
 8   gyro_x          88588 non-null  float64
 9   gyro_y          88588 non-null  float64
 10  gyro_z          88588 non-null  float64
dtypes: float64(6), int64(2), object(3)
memory usage: 7.4+ MB
```

In [5]:

```
#List the columns in the dataset
df.columns
```

Out[5]:

```
Index(['date', 'time', 'username', 'wrist', 'activity', 'acceleration_x',
       'acceleration_y', 'acceleration_z', 'gyro_x', 'gyro_y', 'gyro_z'],
      dtype='object')
```

In [6]:

```
#Let the target variable “y” be the activity, and assign all the columns after it to “x.”

from sklearn.model_selection import train_test_split
X,y = df.iloc[:, 5:].values,df.iloc[:,4].values
X_train, X_test, y_train,y_test = train_test_split(X,y,test_size=0.2, random_state=1)
```

In [7]:

```
print(X_train.shape)
print(y_test[0:10])
```

```
(70870, 6)
[1 0 0 1 1 1 0 1 1 1]
```

In [10]:

```
#Using Scikit-learn, fit a Gaussian Naive Bayes model and observe the accuracy

from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
```

In [12]:

```
classifier.fit(X_train,y_train)
```

Out[12]:

```
GaussianNB()
```

In [13]:

```
y_predict = classifier.predict(X_test)
```

In [14]:

```
from sklearn.metrics import accuracy_score
```

In [15]:

```
accuracy = accuracy_score(y_predict,y_test)
```

In [16]:

```
print(accuracy)
```

```
0.9554690145614629
```

In [17]:

```
from sklearn.metrics import confusion_matrix
```

In [18]:

```
conf_mat = confusion_matrix(y_predict,y_test)
```

In [19]:

```
print(conf_mat)
```

```
[[8583  699]
 [  90 8346]]
```

In [20]:

```
#Generate a classification report using Scikit-learn

from sklearn.metrics import classification_report
target_names = ["Walk","Run"]
```

In [21]:

```
print(classification_report(y_test,y_predict, target_names=target_names))
```

```
              precision    recall  f1-score   support

        Walk       0.92      0.99      0.96      8673
         Run       0.99      0.92      0.95      9045

    accuracy                           0.96     17718
   macro avg       0.96      0.96      0.96     17718
weighted avg       0.96      0.96      0.96     17718
```

In [22]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88588 entries, 0 to 88587
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   date            88588 non-null  object 
 1   time            88588 non-null  object 
 2   username        88588 non-null  object 
 3   wrist           88588 non-null  int64  
 4   activity        88588 non-null  int64  
 5   acceleration_x  88588 non-null  float64
 6   acceleration_y  88588 non-null  float64
 7   acceleration_z  88588 non-null  float64
 8   gyro_x          88588 non-null  float64
 9   gyro_y          88588 non-null  float64
 10  gyro_z          88588 non-null  float64
dtypes: float64(6), int64(2), object(3)
memory usage: 7.4+ MB
```

In [23]:

```
#Repeat the model once using only the acceleration values as predictors and then using only the gyro values as predictors.
#Comment on the difference in accuracy between both models.

from sklearn.model_selection import train_test_split
X,y = df.iloc[:, [5,6,7]].values,df.iloc[:, 4].values
X_train, X_test,y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=1)
```

In [25]:

```
classifier.fit(X_train,y_train)
y_predict = classifier.predict(X_test)
accuracy_score(y_predict, y_test)
```

Out[25]:

```
0.9565978101365843
```

In [26]:

```
print(conf_mat)
```

```
[[8583  699]
 [  90 8346]]
```

In [ ]: