Assignment - 2

Roll No: 241110035
Name: Khushwant Kaswan

CS724: SENSING COMMUNICATIONS AND NETWORKING FOR SMART WIRELESS DEVICES

IIT Kanpur
MTech CSE

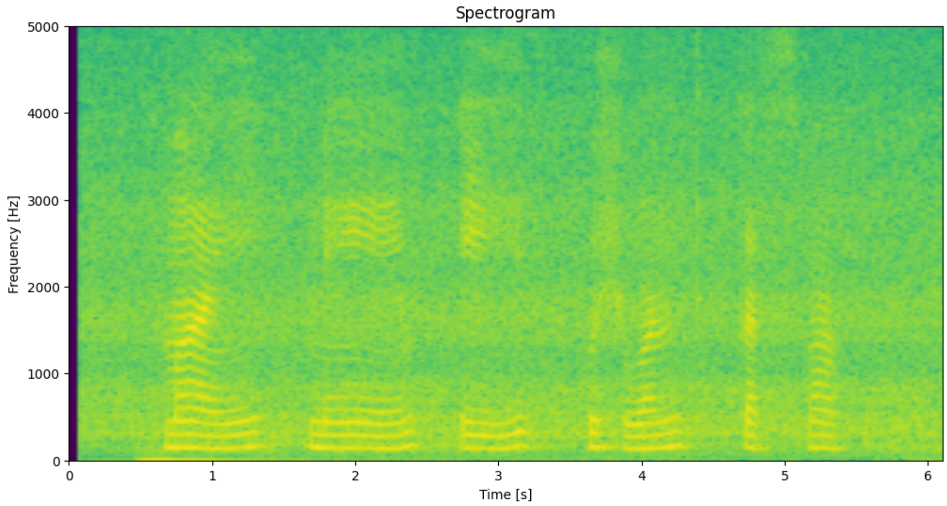
Question 1 : Spectrogram of Audio File

(a)

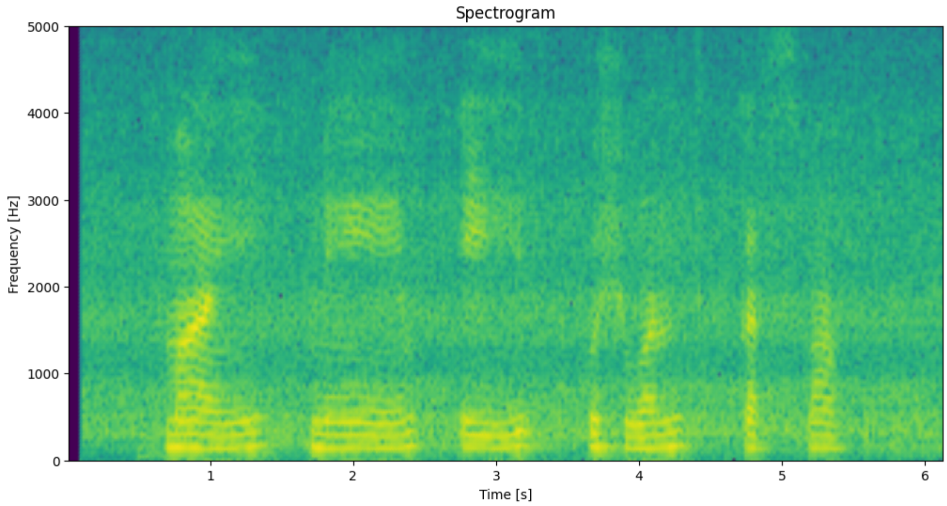
Recorded audo file of .wav format saying "My name is Khushwant Kaswan" and saved as "audio.wav"

(b)

Used numpy and Fourier Transform (numpy.fft) to create a spectogram and plotted using matplotlib.



To verify this, I plotted an direct spectogram of the audio file using `scipy`.



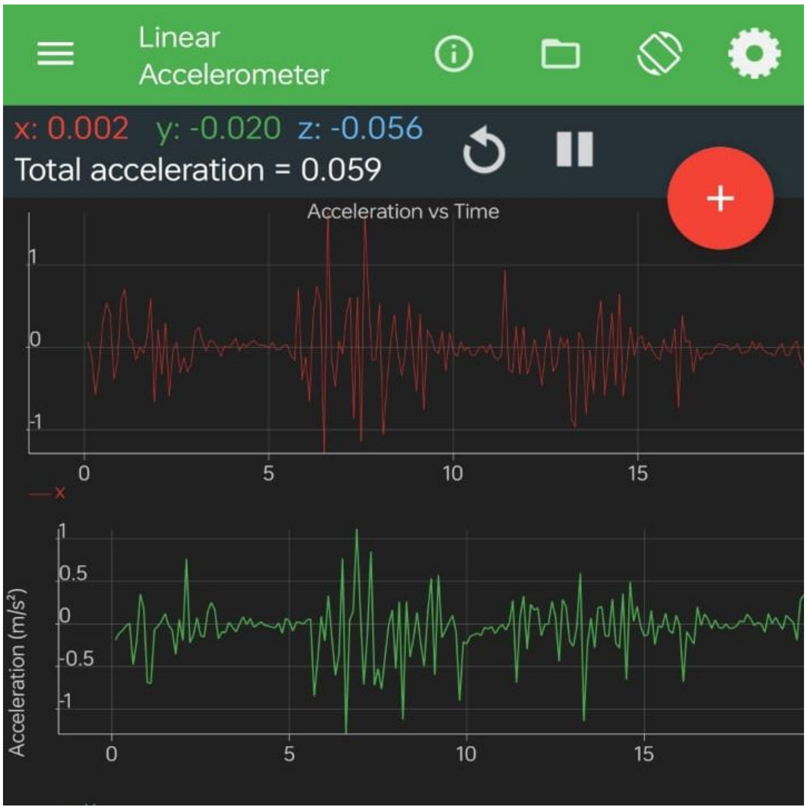
(c)

- Both spectograms(self and scipy generated) are completly same except the color difference.
- Horizontal lines or bands represent frequencies that remain constant during a particular time interval.
- Changes in frequency patterns over time (e.g., the appearance and disappearance of frequency bands) typically correspond to different spoken words or sounds in the audio.
- Each "block" of the spectrogram reflects the frequency content of the audio at that time ie individual words as they are spoken ie "My name is Khushwant Kaswan".
- So, the spectrogram indeed shows the time-frequency structure of speech, with each "spectrum" corresponding to a word or sound.

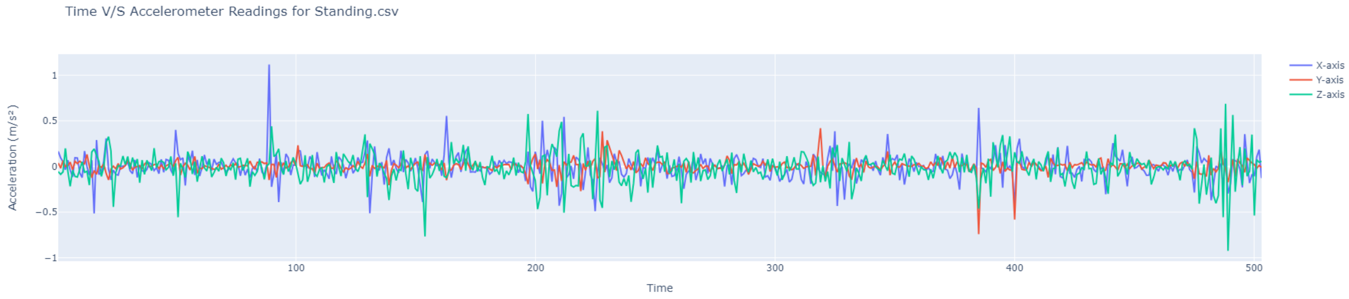
Question 2 : Human Posture Detection

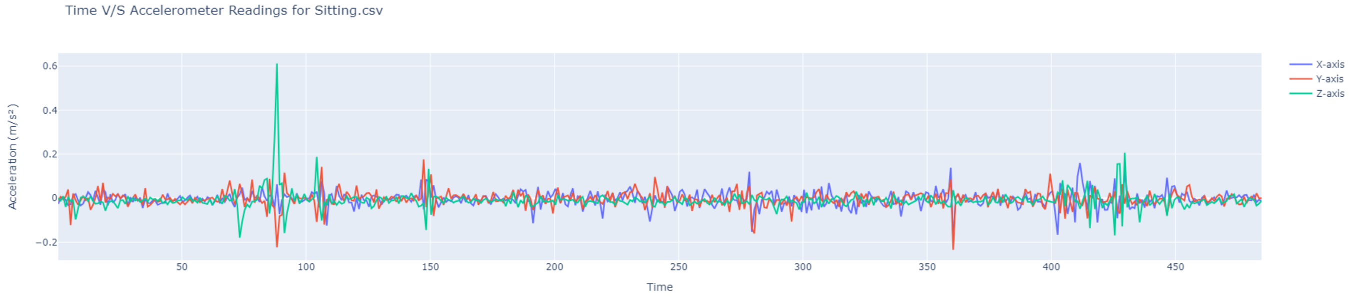
(a)

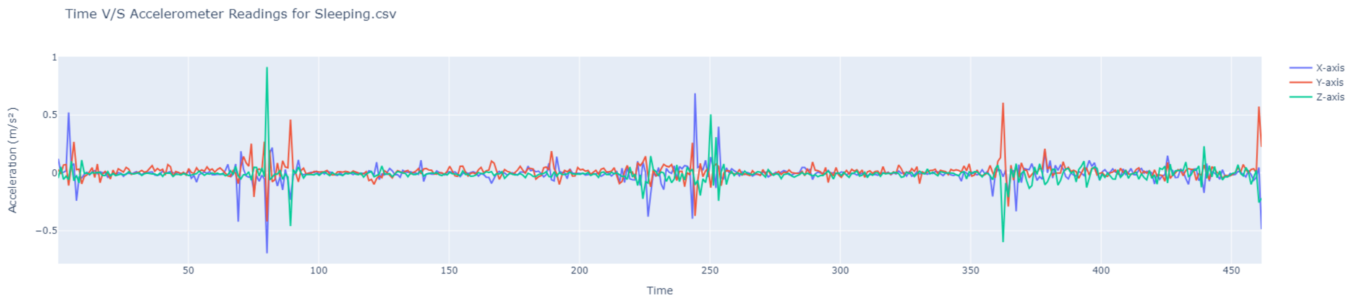
Used the Physics Toolbox app to record the accelerometer samples while sitting, standing and lying down straight.



Plotting them using python `plotly`.







(b)

Find out among these three human postures to which each data belongs to.

Created a test.csv containg 416 different test examples and we are going to classify them to "Sitting" "Sleeping" "Standing" and "Unknown".

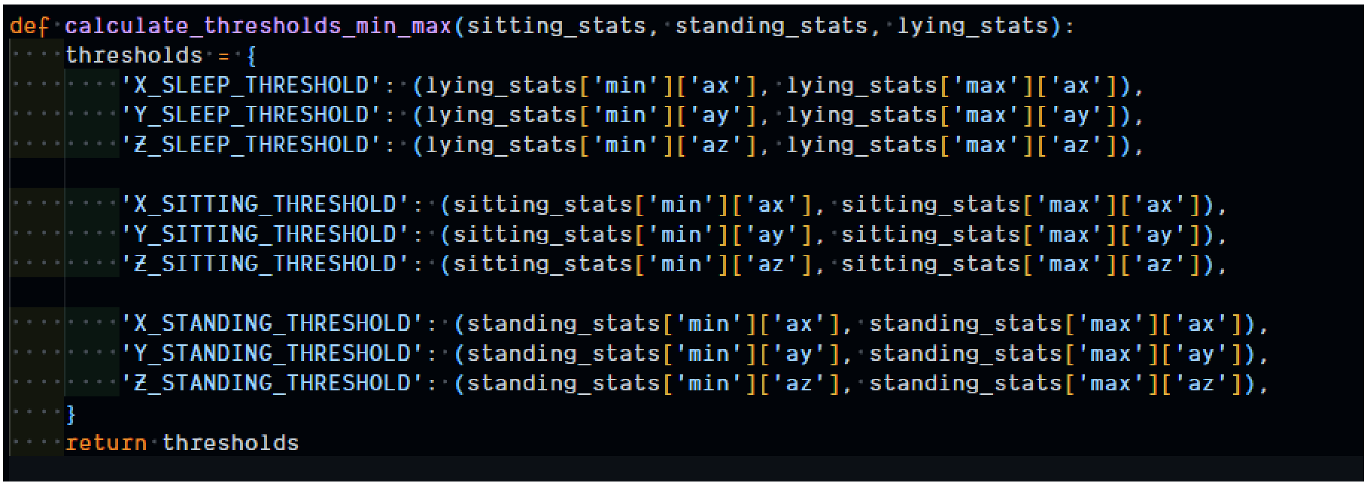
We have used 3 different methods

- Using normal thresholding with threshold chosen as (min---max) values of the acceleration-axis.
- Using normal thresholding with threshold chosen as (mean-2*SD---mean+2*SD) values of the acceleration-axis.
- Using Random Forest Classifier based Machine Learning Approach.

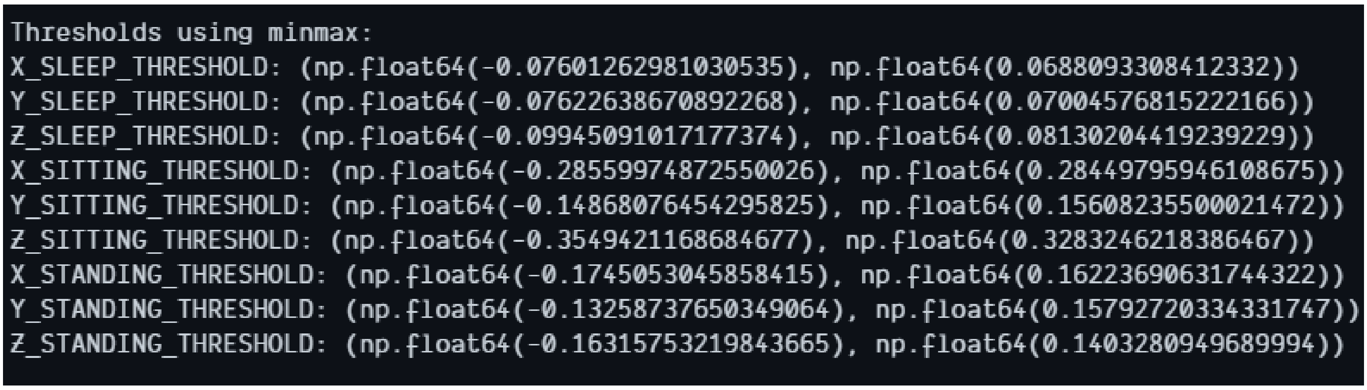
Using the statistics of already available data, we have mean standard-deviation min max.

M1.

For each posture, we create a x,y,z thresholds using minimum and maximum value of that axis.



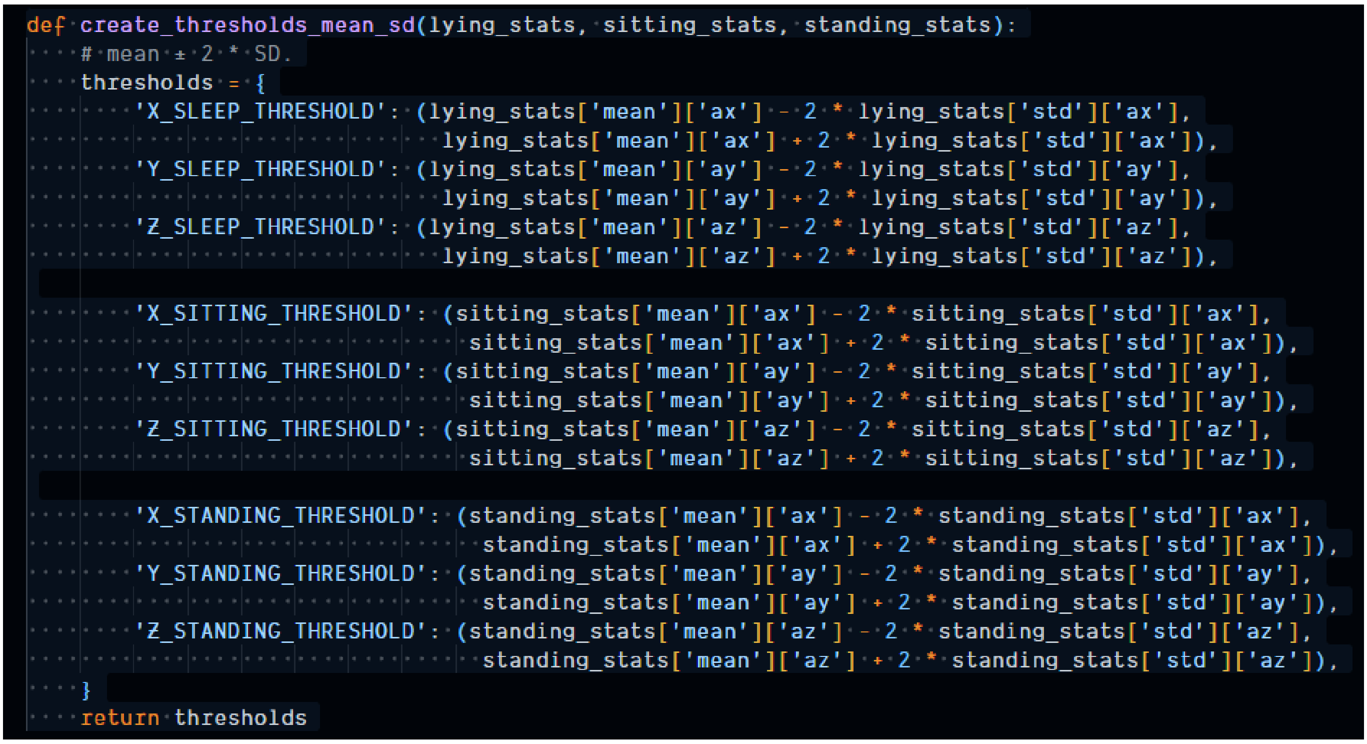
```python
def calculate_thresholds_min_max(sitting_stats, standing_stats, lying_stats):
    thresholds = {
        'X_SLEEP_THRESHOLD': (lying_stats['min']['ax'], lying_stats['max']['ax']),
        'Y_SLEEP_THRESHOLD': (lying_stats['min']['ay'], lying_stats['max']['ay']),
        'Z_SLEEP_THRESHOLD': (lying_stats['min']['az'], lying_stats['max']['az']),

        'X_SITTING_THRESHOLD': (sitting_stats['min']['ax'], sitting_stats['max']['ax']),
        'Y_SITTING_THRESHOLD': (sitting_stats['min']['ay'], sitting_stats['max']['ay']),
        'Z_SITTING_THRESHOLD': (sitting_stats['min']['az'], sitting_stats['max']['az']),

        'X_STANDING_THRESHOLD': (standing_stats['min']['ax'], standing_stats['max']['ax']),
        'Y_STANDING_THRESHOLD': (standing_stats['min']['ay'], standing_stats['max']['ay']),
        'Z_STANDING_THRESHOLD': (standing_stats['min']['az'], standing_stats['max']['az']),
    }
    return thresholds
```



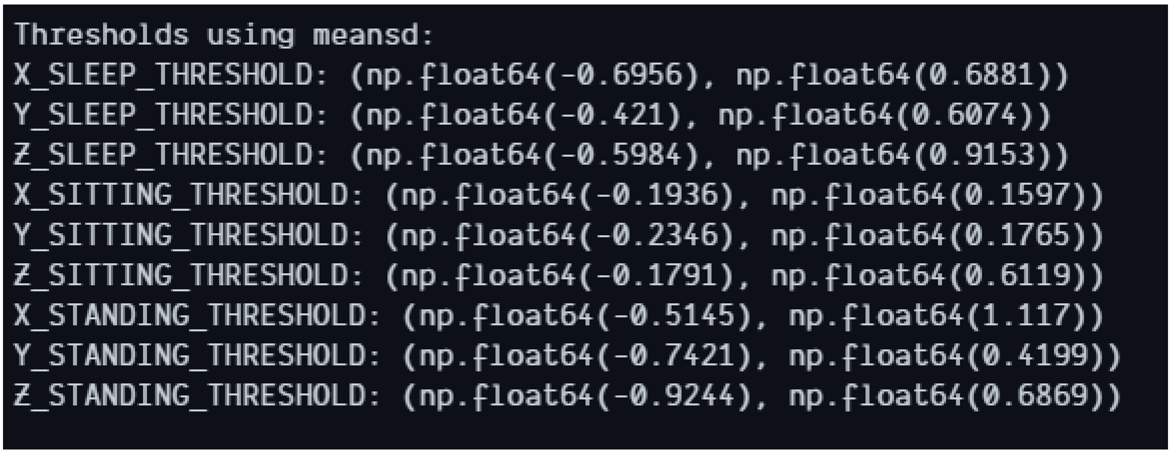
```
Thresholds using minmax:
X_SLEEP_THRESHOLD: (np.float64(-0.07601262981030535), np.float64(0.0688093308412332))
Y_SLEEP_THRESHOLD: (np.float64(-0.07622638670892268), np.float64(0.07004576815222166))
Z_SLEEP_THRESHOLD: (np.float64(-0.09945091017177374), np.float64(0.08130204419239229))
X_SITTING_THRESHOLD: (np.float64(-0.28559974872550026), np.float64(0.28449795946108675))
Y_SITTING_THRESHOLD: (np.float64(-0.14868076454295825), np.float64(0.15608235500021472))
Z_SITTING_THRESHOLD: (np.float64(-0.3549421168684677), np.float64(0.3283246218386467))
X_STANDING_THRESHOLD: (np.float64(-0.1745053045858415), np.float64(0.16223690631744322))
Y_STANDING_THRESHOLD: (np.float64(-0.13258737650349064), np.float64(0.15792720334331747))
Z_STANDING_THRESHOLD: (np.float64(-0.16315753219843665), np.float64(0.1403280949689994))
```

M2.

For each posture, we create a x,y,z thresholds between mean-2*SD to mean*+2SD for all axis.



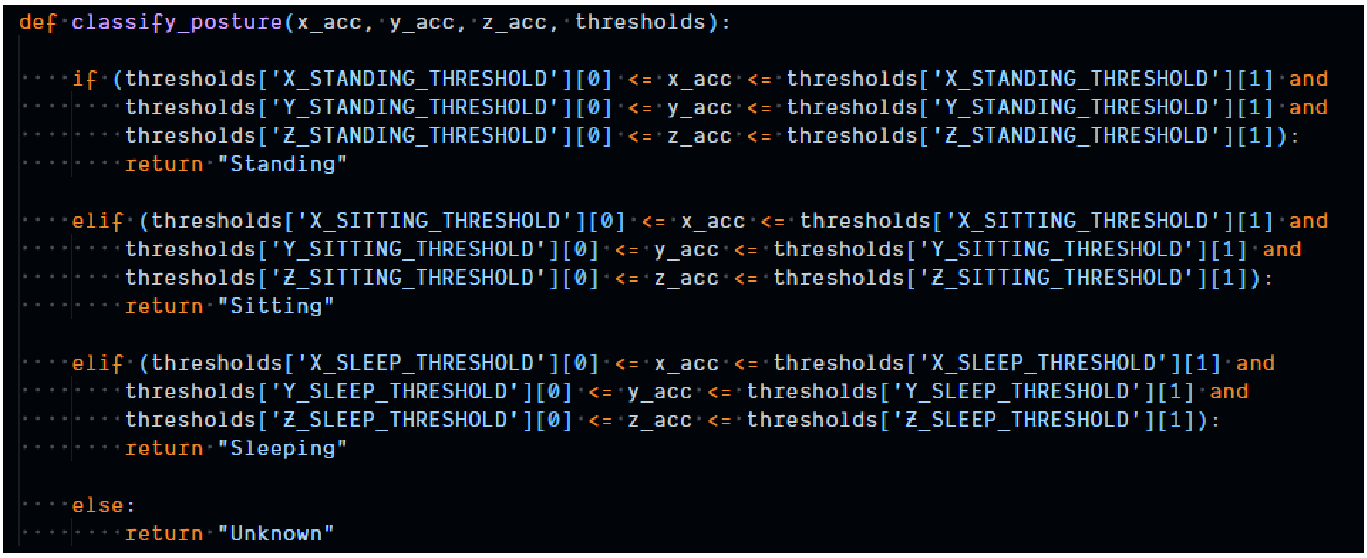
```python
def create_thresholds_mean_sd(lying_stats, sitting_stats, standing_stats):
    # mean ± 2 * SD.
    thresholds = {
        'X_SLEEP_THRESHOLD': (lying_stats['mean']['ax'] - 2 * lying_stats['std']['ax'],
                              lying_stats['mean']['ax'] + 2 * lying_stats['std']['ax']),
        'Y_SLEEP_THRESHOLD': (lying_stats['mean']['ay'] - 2 * lying_stats['std']['ay'],
                              lying_stats['mean']['ay'] + 2 * lying_stats['std']['ay']),
        'Z_SLEEP_THRESHOLD': (lying_stats['mean']['az'] - 2 * lying_stats['std']['az'],
                              lying_stats['mean']['az'] + 2 * lying_stats['std']['az']),

        'X_SITTING_THRESHOLD': (sitting_stats['mean']['ax'] - 2 * sitting_stats['std']['ax'],
                                sitting_stats['mean']['ax'] + 2 * sitting_stats['std']['ax']),
        'Y_SITTING_THRESHOLD': (sitting_stats['mean']['ay'] - 2 * sitting_stats['std']['ay'],
                                sitting_stats['mean']['ay'] + 2 * sitting_stats['std']['ay']),
        'Z_SITTING_THRESHOLD': (sitting_stats['mean']['az'] - 2 * sitting_stats['std']['az'],
                                sitting_stats['mean']['az'] + 2 * sitting_stats['std']['az']),

        'X_STANDING_THRESHOLD': (standing_stats['mean']['ax'] - 2 * standing_stats['std']['ax'],
                                 standing_stats['mean']['ax'] + 2 * standing_stats['std']['ax']),
        'Y_STANDING_THRESHOLD': (standing_stats['mean']['ay'] - 2 * standing_stats['std']['ay'],
                                 standing_stats['mean']['ay'] + 2 * standing_stats['std']['ay']),
        'Z_STANDING_THRESHOLD': (standing_stats['mean']['az'] - 2 * standing_stats['std']['az'],
                                 standing_stats['mean']['az'] + 2 * standing_stats['std']['az']),
    }
    return thresholds
```



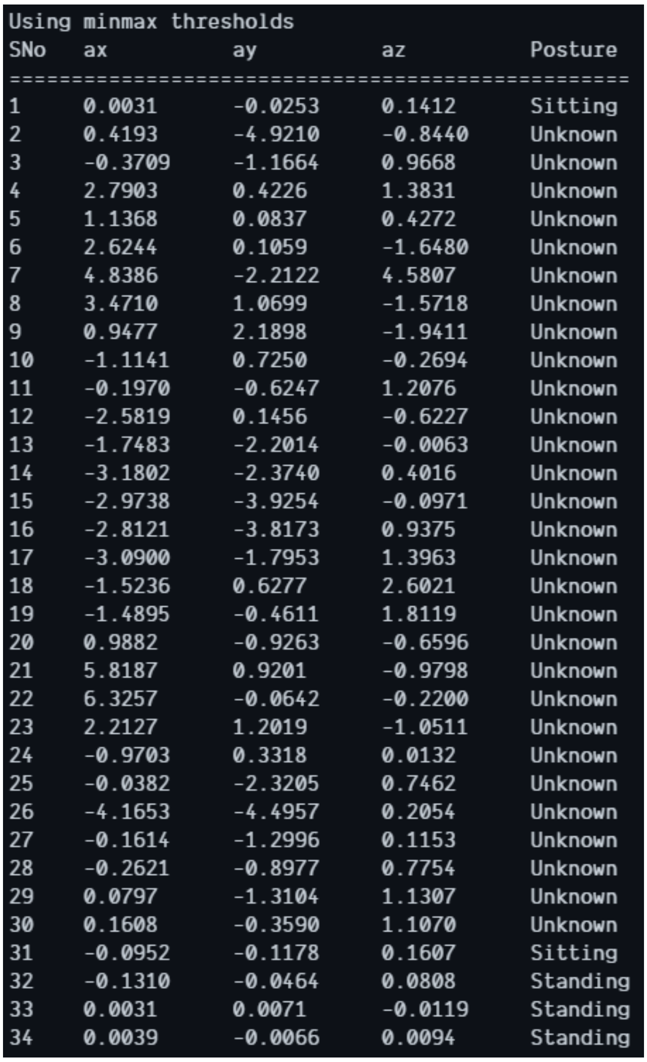
```
Thresholds using meansd:
X_SLEEP_THRESHOLD: (np.float64(-0.6956), np.float64(0.6881))
Y_SLEEP_THRESHOLD: (np.float64(-0.421), np.float64(0.6074))
Z_SLEEP_THRESHOLD: (np.float64(-0.5984), np.float64(0.9153))
X_SITTING_THRESHOLD: (np.float64(-0.1936), np.float64(0.1597))
Y_SITTING_THRESHOLD: (np.float64(-0.2346), np.float64(0.1765))
Z_SITTING_THRESHOLD: (np.float64(-0.1791), np.float64(0.6119))
X_STANDING_THRESHOLD: (np.float64(-0.5145), np.float64(1.117))
Y_STANDING_THRESHOLD: (np.float64(-0.7421), np.float64(0.4199))
Z_STANDING_THRESHOLD: (np.float64(-0.9244), np.float64(0.6869))
```

Testing using M1 and M2

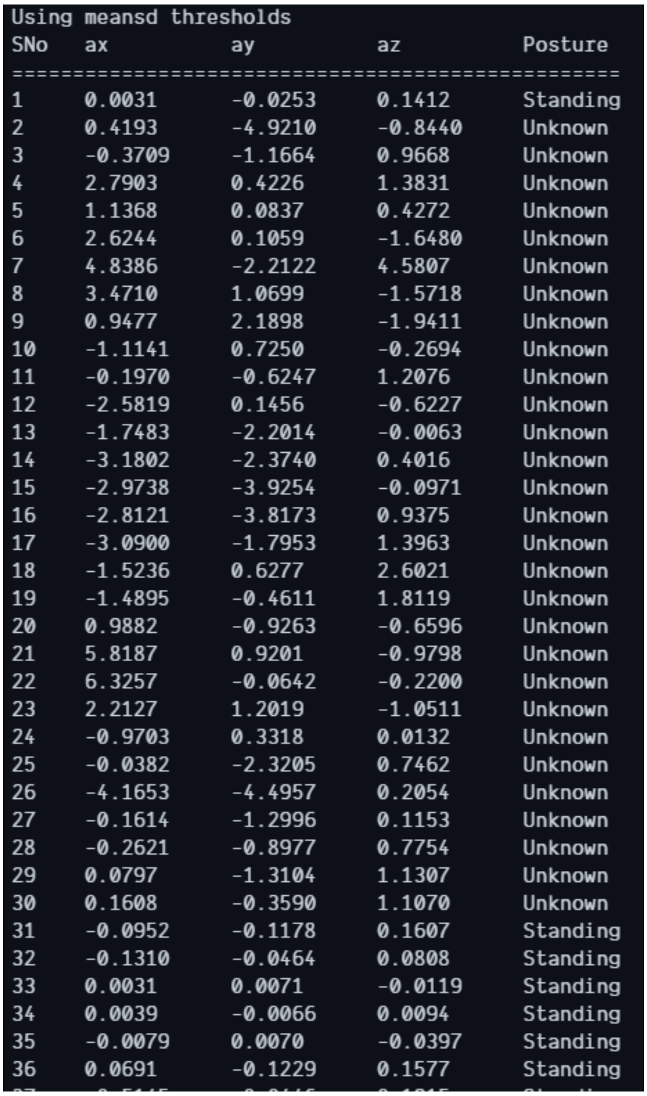
If all the x,y,z value lies betwee the threhold only then we classify the posture else Unknown.



```python
def classify_posture(x_acc, y_acc, z_acc, thresholds):

    if (thresholds['X_STANDING_THRESHOLD'][0] <= x_acc <= thresholds['X_STANDING_THRESHOLD'][1] and
        thresholds['Y_STANDING_THRESHOLD'][0] <= y_acc <= thresholds['Y_STANDING_THRESHOLD'][1] and
        thresholds['Z_STANDING_THRESHOLD'][0] <= z_acc <= thresholds['Z_STANDING_THRESHOLD'][1]):
        return "Standing"

    elif (thresholds['X_SITTING_THRESHOLD'][0] <= x_acc <= thresholds['X_SITTING_THRESHOLD'][1] and
        thresholds['Y_SITTING_THRESHOLD'][0] <= y_acc <= thresholds['Y_SITTING_THRESHOLD'][1] and
        thresholds['Z_SITTING_THRESHOLD'][0] <= z_acc <= thresholds['Z_SITTING_THRESHOLD'][1]):
        return "Sitting"

    elif (thresholds['X_SLEEP_THRESHOLD'][0] <= x_acc <= thresholds['X_SLEEP_THRESHOLD'][1] and
        thresholds['Y_SLEEP_THRESHOLD'][0] <= y_acc <= thresholds['Y_SLEEP_THRESHOLD'][1] and
        thresholds['Z_SLEEP_THRESHOLD'][0] <= z_acc <= thresholds['Z_SLEEP_THRESHOLD'][1]):
        return "Sleeping"

    else:
        return "Unknown"
```



```
Using minmax thresholds
SNo   ax          ay          az          Posture
==================================================
1     0.0031      -0.0253     0.1412      Sitting
2     0.4193      -4.9210     -0.8440     Unknown
3     -0.3709     -1.1664     0.9668      Unknown
4     2.7903      0.4226      1.3831      Unknown
5     1.1368      0.0837      0.4272      Unknown
6     2.6244      0.1059      -1.6480     Unknown
7     4.8386      -2.2122     4.5807      Unknown
8     3.4710      1.0699      -1.5718     Unknown
9     0.9477      2.1898      -1.9411     Unknown
10    -1.1141     0.7250      -0.2694     Unknown
11    -0.1970     -0.6247     1.2076      Unknown
12    -2.5819     0.1456      -0.6227     Unknown
13    -1.7483     -2.2014     -0.0063     Unknown
14    -3.1802     -2.3740     0.4016      Unknown
15    -2.9738     -3.9254     -0.0971     Unknown
16    -2.8121     -3.8173     0.9375      Unknown
17    -3.0900     -1.7953     1.3963      Unknown
18    -1.5236     0.6277      2.6021      Unknown
19    -1.4895     -0.4611     1.8119      Unknown
20    0.9882      -0.9263     -0.6596     Unknown
21    5.8187      0.9201      -0.9798     Unknown
22    6.3257      -0.0642     -0.2200     Unknown
23    2.2127      1.2019      -1.0511     Unknown
24    -0.9703     0.3318      0.0132      Unknown
25    -0.0382     -2.3205     0.7462      Unknown
26    -4.1653     -4.4957     0.2054      Unknown
27    -0.1614     -1.2996     0.1153      Unknown
28    -0.2621     -0.8977     0.7754      Unknown
29    0.0797      -1.3104     1.1307      Unknown
30    0.1608      -0.3590     1.1070      Unknown
31    -0.0952     -0.1178     0.1607      Sitting
32    -0.1310     -0.0464     0.0808      Standing
33    0.0031      0.0071      -0.0119     Standing
34    0.0039      -0.0066     0.0094      Standing
```

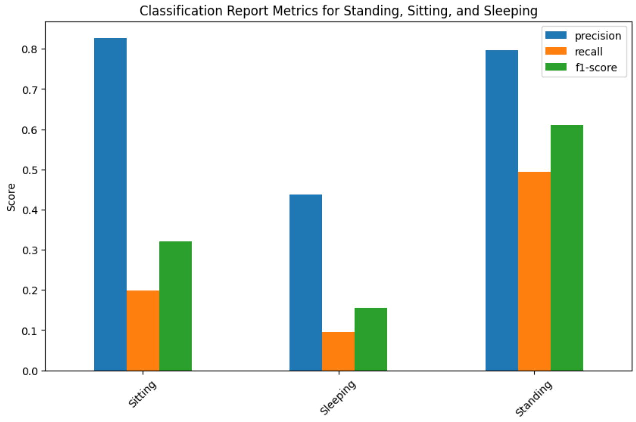


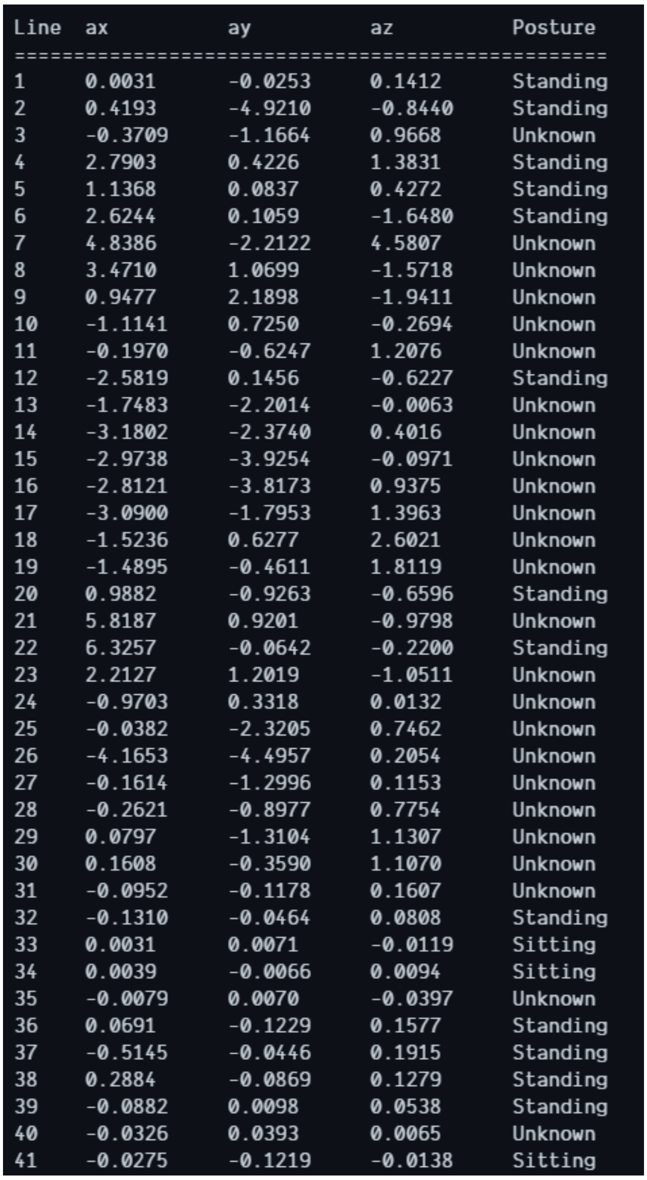
```
Using meansd thresholds
SNo   ax          ay          az          Posture
==================================================
1     0.0031      -0.0253     0.1412      Standing
2     0.4193      -4.9210     -0.8440     Unknown
3     -0.3709     -1.1664     0.9668      Unknown
4     2.7903      0.4226      1.3831      Unknown
5     1.1368      0.0837      0.4272      Unknown
6     2.6244      0.1059      -1.6480     Unknown
7     4.8386      -2.2122     4.5807      Unknown
8     3.4710      1.0699      -1.5718     Unknown
9     0.9477      2.1898      -1.9411     Unknown
10    -1.1141     0.7250      -0.2694     Unknown
11    -0.1970     -0.6247     1.2076      Unknown
12    -2.5819     0.1456      -0.6227     Unknown
13    -1.7483     -2.2014     -0.0063     Unknown
14    -3.1802     -2.3740     0.4016      Unknown
15    -2.9738     -3.9254     -0.0971     Unknown
16    -2.8121     -3.8173     0.9375      Unknown
17    -3.0900     -1.7953     1.3963      Unknown
18    -1.5236     0.6277      2.6021      Unknown
19    -1.4895     -0.4611     1.8119      Unknown
20    0.9882      -0.9263     -0.6596     Unknown
21    5.8187      0.9201      -0.9798     Unknown
22    6.3257      -0.0642     -0.2200     Unknown
23    2.2127      1.2019      -1.0511     Unknown
24    -0.9703     0.3318      0.0132      Unknown
25    -0.0382     -2.3205     0.7462      Unknown
26    -4.1653     -4.4957     0.2054      Unknown
27    -0.1614     -1.2996     0.1153      Unknown
28    -0.2621     -0.8977     0.7754      Unknown
29    0.0797      -1.3104     1.1307      Unknown
30    0.1608      -0.3590     1.1070      Unknown
31    -0.0952     -0.1178     0.1607      Standing
32    -0.1310     -0.0464     0.0808      Standing
33    0.0031      0.0071      -0.0119     Standing
34    0.0039      -0.0066     0.0094      Standing
35    -0.0079     0.0070      -0.0397     Standing
36    0.0691      -0.1229     0.1577      Standing
```

M3.

Using Random Forest Based Machine Learning Classifier.

- Add posture labels to the each dataset
- Combine all 3 datasets into one dataset
- Split the data into training and validation sets
- Train a Random Forest classifier
- Predict on validation data to check performance
- For each prediction, if the prediction confidence is below the threshold(0.7), assign "Unknown"
- Predict on the test.csv data
- For each test prediction, assign "Unknown" if prediction confidence is below the threshold(0.7)
- Show the classification report and test data with predicted postures





```
Line  ax          ay          az          Posture
================================================
1     0.0031      -0.0253     0.1412      Standing
2     0.4193      -4.9210     -0.8440     Standing
3     -0.3709     -1.1664     0.9668      Unknown
4     2.7903      0.4226      1.3831      Standing
5     1.1368      0.0837      0.4272      Standing
6     2.6244      0.1059      -1.6480     Standing
7     4.8386      -2.2122     4.5807      Unknown
8     3.4710      1.0699      -1.5718     Unknown
9     0.9477      2.1898      -1.9411     Unknown
10    -1.1141     0.7250      -0.2694     Unknown
11    -0.1970     -0.6247     1.2076      Unknown
12    -2.5819     0.1456      -0.6227     Standing
13    -1.7483     -2.2014     -0.0063     Unknown
14    -3.1802     -2.3740     0.4016      Unknown
15    -2.9738     -3.9254     -0.0971     Unknown
16    -2.8121     -3.8173     0.9375      Unknown
17    -3.0900     -1.7953     1.3963      Unknown
18    -1.5236     0.6277      2.6021      Unknown
19    -1.4895     -0.4611     1.8119      Unknown
20    0.9882      -0.9263     -0.6596     Standing
21    5.8187      0.9201      -0.9798     Unknown
22    6.3257      -0.0642     -0.2200     Standing
23    2.2127      1.2019      -1.0511     Unknown
24    -0.9703     0.3318      0.0132      Unknown
25    -0.0382     -2.3205     0.7462      Unknown
26    -4.1653     -4.4957     0.2054      Unknown
27    -0.1614     -1.2996     0.1153      Unknown
28    -0.2621     -0.8977     0.7754      Unknown
29    0.0797      -1.3104     1.1307      Unknown
30    0.1608      -0.3590     1.1070      Unknown
31    -0.0952     -0.1178     0.1607      Unknown
32    -0.1310     -0.0464     0.0808      Standing
33    0.0031      0.0071      -0.0119     Sitting
34    0.0039      -0.0066     0.0094      Sitting
35    -0.0079     0.0070      -0.0397     Unknown
36    0.0691      -0.1229     0.1577      Standing
37    -0.5145     -0.0446     0.1915      Standing
38    0.2884      -0.0869     0.1279      Standing
39    -0.0882     0.0098      0.0538      Standing
40    -0.0326     0.0393      0.0065      Unknown
41    -0.0275     -0.1219     -0.0138     Sitting
```