Digital Signal Processing

Assignment 3

Team Members:
1)**IMT2017507** (Goli Akshay Sujith)
2)**IMT2017510** (Prakash K)

Introduction:

Human Activity recognition is formulated as a supervised classification problem, whose training data is obtained via an experiment having human subjects performing each of the activities. We aim to develop a model that is capable of recognizing multiple sets of daily activities under real-world conditions, using data collected by an accelerometer built into a cell phone.

Data Collection:

The data for human activities such as walking, Jogging, Jumping and climbing is collected using an accelerometer collector in smartphones. Here we'll be predicting the movement of a person based on data collected.

About the Dataset (Originally) :

The data is collected from the smartphones using the accelerometer collected and it is stored in CSV format for further explorations. The format of originally collected data is as follows:
Features: Time , X (acceleration), Y (acceleration), Z (acceleration)

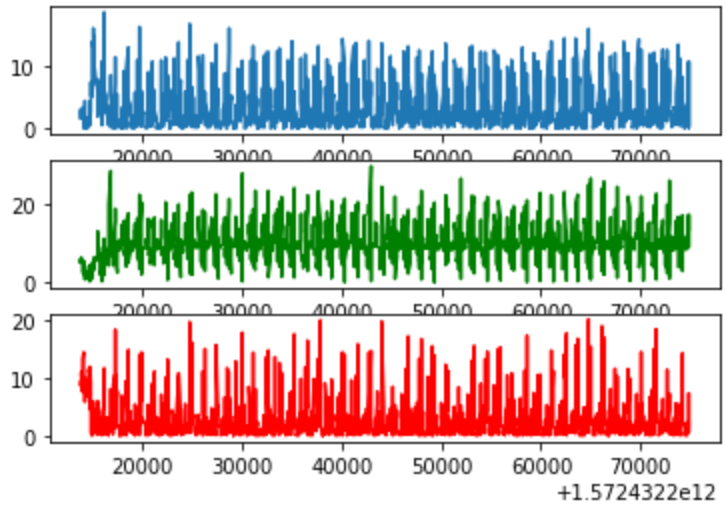
	time	x	y	z
0	1.572437e+12	-9.129990	2.803299	6.376724
1	1.572437e+12	-8.675095	1.946167	5.869156
2	1.572437e+12	-6.132446	0.610199	4.695999
3	1.572437e+12	-6.611298	1.103409	4.949783
4	1.572437e+12	-6.256958	2.501617	8.775711
...	...	...	...	...
842	1.572437e+12	-3.575439	4.153625	8.723038
843	1.572437e+12	-3.240250	3.751404	8.493195
844	1.572437e+12	-3.704727	3.349167	8.387848
845	1.572437e+12	-4.427780	3.325226	8.316025
846	1.572437e+12	-4.992813	3.526337	8.162796

IMG: Format of data collected using accelerometer collector for jumping

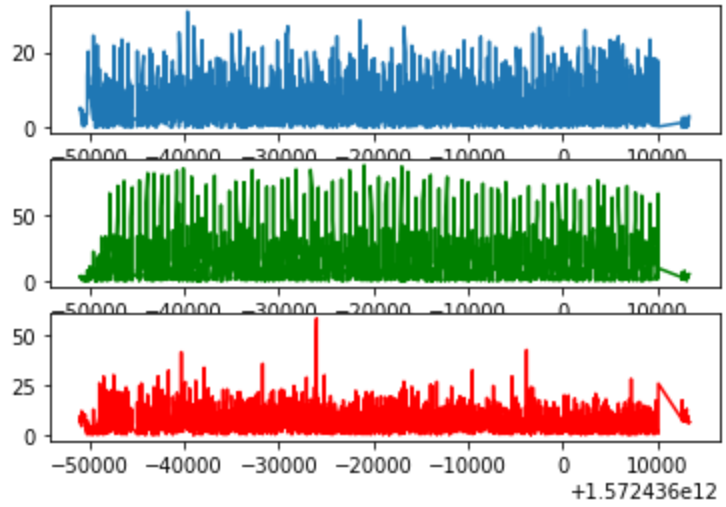
Data Exploration:

Acceleration Plotting:

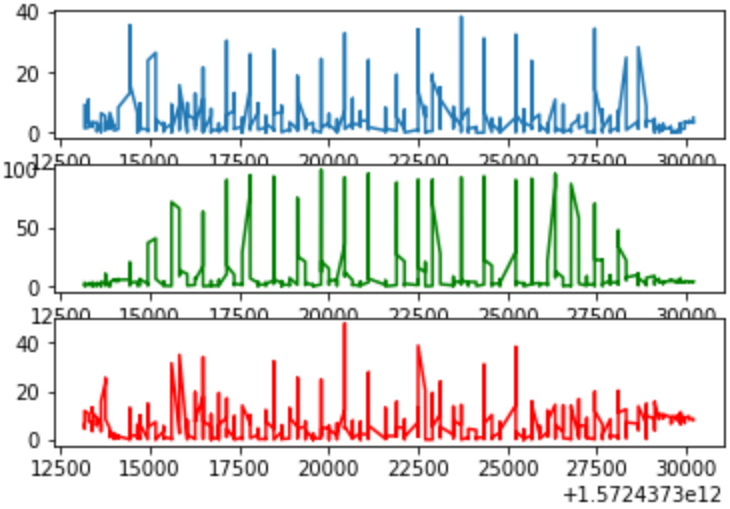
Time-Amp(X, Y, Z) graph for Walking (For one Datapoint)



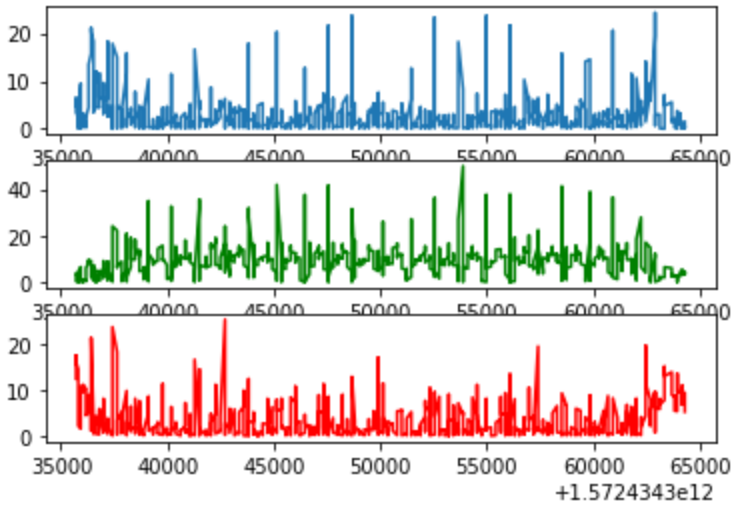
Time-Amp(X, Y, Z) graph for Jogging (For one Datapoint) :



Time-Amp graph(X, Y, Z) for Jumping (For one Datapoint) :



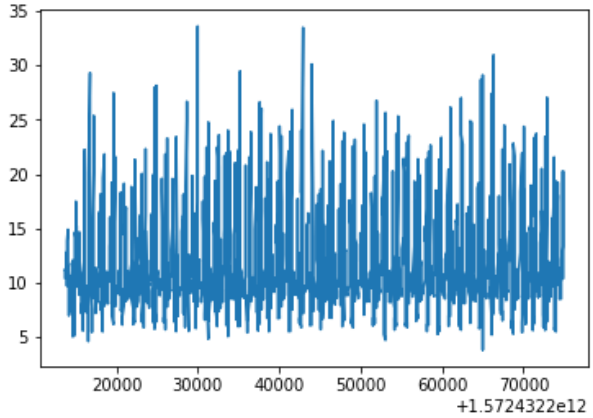
TIme-Amp graph(X, Y, Z) for Climbing (For one Datapoint) :



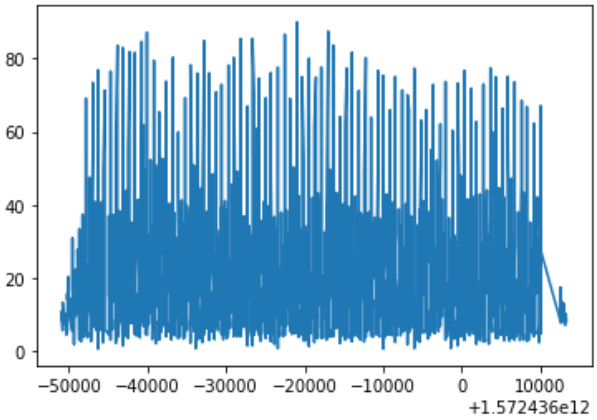
The data is collected by providing a specified sampling rate and three-dimensional accelerations of the data are collected. We see the variations of amplitude and peaks for different human activities which enables use for further explorations. The Blue indicates the X Acceleration, the Green indicates the Y Acceleration and the Red indicates the Z Acceleration. The Peak amplitude or MRA is around 10 units (For X), 20 units (For Y), 15 units (for Z) for walking and 20 units (for X),55 units (for Y),25 units (for Z) for Jogging which acts as classifiers. The time-Amp graph for jumping and climbing have variations of area density. For climbing, the Amplitudes for 2 axes are around the same value because the motion of change is along one axis.

Magnitude Graph for Human Activity:

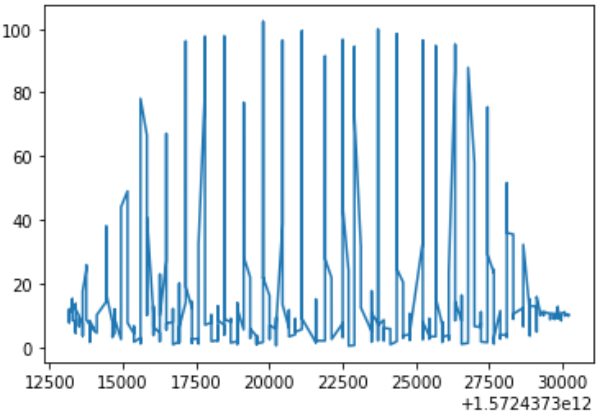
For Walking (For one Datapoint) :



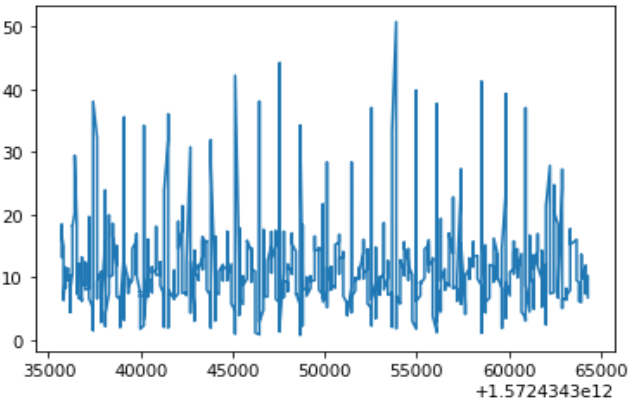
For Jogging (For one Datapoint):



For Jumping (For one Datapoint):



For Climbing (For one Datapoint):



The above graphs are magnitude graphs for every human activity recorded. We see the Amplitude variations for every activity i.e for Walking the Amp is around 25 units, for Jogging it is around 80 units, for Jumping it is around 100 units and for climbing it is around 20-40 units. We can also see the areal density variations for every activity recorded.

Feature Evaluation:

The accelerometer generates three-time series along the x-axis, y-axis, and z-axis, denoted by AX, AY and AZ, respectively. Each time series combines the linear acceleration due to body motion and due to gravity. we removed some samples at first and some samples at the end of every data file because there will be a chance of getting error values due to the starting and stopping of the sensor. we pick the magnitude of acceleration to be for feature construction.

Feature Extraction:

We used four-time series obtained in feature evaluation, we took a window size such that we didn't lose any data. we use the mean value for each window as one of the features. we also took a standard deviation(SD), root means square(RMS) and means absolute deviation(MAD) as one of the features.

Classification:

The Final data frame is divided into Train and test datasets and the train is fed to classification models. Before splitting the Dataset is shuffled.

Classification accuracy for different classifiers:

- The accuracy of the SVM kernel linear is 0.9469026548672567 (94.69%)
- The accuracy of the KNN is 0.9646017699115044 (96.46%)
- The accuracy of the Logistic Regression is 0.9380530973451328 (93.80%)

We evaluated the performance of the classifiers using pip toolkits (sklearn). Classifiers are trained and then tested on the test Dataset.
FInal DataFrame will look something like this before giving it to split and modal.

	x_mean	y_mean	z_mean	mag_mean	x_std	y_std	z_std	mag_std	x_mad	y_mad	z_mad	mag_mad	name
0	5.200556	-8.090993	-0.177448	11.928018	5.733349	5.397369	4.027406	5.318549	4.226741	3.905438	2.502137	3.884403	climbing
1	3.362202	-9.473108	0.504373	10.803490	3.461869	3.346298	2.713051	3.881255	2.489458	2.433577	2.026059	2.808169	climbing
2	3.958147	-9.441850	0.248405	10.872513	3.618126	3.437268	2.447784	4.184420	2.560808	2.389337	1.785103	2.923913	climbing
3	3.876458	-9.493899	0.339643	10.705184	2.953092	3.180301	1.978809	3.659064	2.158061	2.336617	1.529890	2.687143	climbing
4	4.105305	-9.298197	0.512715	10.733330	3.096942	3.418201	2.462904	3.957632	2.172996	2.329066	1.806167	2.739183	climbing
...	...	...	...	...	...	...	...	...	...	...	...	...	...
558	5.097750	-8.775361	-0.046456	10.885899	3.727581	2.503945	2.265694	3.119147	2.592401	1.776484	1.847204	2.427869	walking
559	5.097663	-8.813180	0.075428	10.957595	3.815652	2.341971	2.403828	3.058044	2.729255	1.726967	1.899133	2.430220	walking
560	5.142800	-8.952589	-0.058542	11.240913	4.281629	2.759532	2.355945	3.414827	3.199155	2.030993	1.882182	2.725671	walking
561	2.320073	-2.177994	5.398524	10.930446	4.665578	6.487062	5.354433	3.463127	2.996722	6.020700	4.539434	2.519211	walking
562	1.129177	2.097283	9.230640	9.796145	1.593754	1.260301	2.093236	1.843928	1.233566	1.033483	1.644604	1.501246	walking

Confusion Matrix:

For modal1 (SVM) :

```
Confusion Matrix :
[[24  0  0  4]
 [ 0 30  0  0]
 [ 0  0  6  0]
 [ 2  0  0 47]]
Accuracy Score : 0.9469026548672567
Report :
              precision    recall  f1-score   support

           0       0.92      0.86      0.89        28
           1       1.00      1.00      1.00        30
           2       1.00      1.00      1.00         6
           3       0.92      0.96      0.94        49

    accuracy                           0.95       113
   macro avg       0.96      0.95      0.96       113
weighted avg       0.95      0.95      0.95       113
```

For modal2 (KNeighborsClassifier):

```
Confusion Matrix :
[[27  0  0  1]
 [ 0 30  0  0]
 [ 0  1  5  0]
 [ 2  0  0 47]]
Accuracy Score : 0.9646017699115044
Report :
              precision    recall  f1-score   support

           0       0.93      0.96      0.95        28
           1       0.97      1.00      0.98        30
           2       1.00      0.83      0.91         6
           3       0.98      0.96      0.97        49

    accuracy                           0.96       113
   macro avg       0.97      0.94      0.95       113
weighted avg       0.97      0.96      0.96       113
```

For Modal3 (LogisticRegression) :

```
Confusion Matrix :
[[21  1  0  6]
 [ 0 30  0  0]
 [ 0  0  6  0]
 [ 0  0  0 49]]
Accuracy Score : 0.9380530973451328
Report :
              precision    recall  f1-score   support

           0       1.00      0.75      0.86        28
           1       0.97      1.00      0.98        30
           2       1.00      1.00      1.00         6
           3       0.89      1.00      0.94        49

    accuracy                           0.94       113
   macro avg       0.96      0.94      0.95       113
weighted avg       0.94      0.94      0.94       113
```