**A2 | Disha Singh**

**P.S. :** Since my data “Disha\_Singh” folder did not capture pressure.txt data, I had to use the

folder “Cody” as test data.

**Step 1: Plot figures**

A picture containing graphical user interface

Description automatically generatedDiagram

Description automatically generated

**Stationary**: The **accelerometer magnitude is least**(~10) with **almost no peaks** among all activities. **Variance for accelerometer is 0** as the person is stationary. **Barometer slope is almost constant**.

**Walking flat surface**: There are **more peaks** in accelerometer magnitude than stationary. All other plotted metrics are **very similar to stationary** **except** that the accelerometer magnitude **amplitude** **is not consistent** and is very small too – it **oscillates up and down** centering at the stationary level amplitude. The accelerometer **variance** even though not very considerable, is **not 0** like stationary. The **mean peak value** for accelerometer magnitude will be definitely lesser than other movement activities (<20).

**Running**: All windows taken during running had similar variance and this was the **highest accelerometer variance** among all activities (~20). Accelerometer magnitude peaks crossing 80(varied from 0 to 80). The highest amplitudes

were witnessed for running which were very high than the other magnitudes (<20). **Barometric pressure is consistently high** with **most peaks > 1026.5**

**Elevator up**: **Pressure keeps falling** while going up the elevator with a **constant accelerometer magnitude ~10. Barometer slope keeps falling linearly.** Accelerometer magnitude variance close to 0.

**Elevator down**: **Pressure keeps rising** while going up the elevator with a **constant accelerometer magnitude ~10. Barometer slope keeps rising linearly.** Accelerometer magnitude variance close to 0.

**Walking stairs up:** A slight **accelerometer magnitude variance <5**. **Pressure keeps decreasing** with a linearly **decreasing slope**. The **accelerometer magnitude signal oscillates between 0 and 20** with about 10% of the peaks crossing 20.

Lowest **barometer slope** value achieved is usually lesser than that achieved during elevator up. Same is true for pressure.

**Walking stairs down:** A slight **accelerometer magnitude variance <5**. **Pressure keeps increasin**g with a linearly **increasing slope**. The **accelerometer magnitude signal oscillates between 0 and 20** with about 10% of the peaks crossing 20. Highest **barometer slope** value achieved is usually lesser than that achieved during elevator down. Same is true for pressure.

**Step 2: Windowing and feature extraction**

I have computed sampling rate in the file computeSamplingRate.m . It approximately yielded in fs = 203Hz

=>203 samples/s. But since we were asked to use default sampling frequency of 32Hz. I have taken sampling rate to be 32 samples/s.

Therefore, the shift for 2s => 64 samples/s and the window length of 10s => 32\*10 = 320 samples/s. To modify the data length according to this, I have made changes in computeRawData.m: Line 76-82.

Text, letter

Description automatically generated

This clipping of some data from the end is done to enable proper windowing and shifting of data in

extractFeatures.m: Line 12 :



Please see extractFeatures.m: Line 17-85 to see all extracted features. The implementation of frequency domain features along with conversion from time domain to frequency domain(FFT) is in class FREQUENCYDOMAIN.m

**Continued…**

**Step 3: KNN and Random Forest on all features**

I have implemented the **KNN classifier in knn.m** and **randomForest classifer in RandomForest.m** separately. Please

run these scripts separately to see performance results.

Within randomForest.m change *domain* variable in Line:6 to *‘all’* in order to run the classifier on both time domain

and frequency features.

**KNN Performance with all 30 features:**

**testAccuracy: 0.19**

**Actual (same order)**

**Predicted**

|  |
| --- |
| 'Stationary' |
| 'Walking-flat-surface' |
| 'Running' |
| 'Elevator-up' |
| 'Elevator-down' |
| 'Walking-up-stairs' |
| 'Walking-down-stairs' |

1 46 0 0 1 0 0

0 58 0 2 0 0 0

0 56 0 0 0 0 0

0 27 0 0 4 0 0

0 29 0 0 3 0 0

0 50 0 0 0 0 0

0 47 0 0 0 0 0

**Activity wise Performance Metrics:**

Activity | Precision | Recall | F-1 Score

Stationary | 1.00 | 0.02 | 0.04

Walking-flat-surface | 0.19 | 0.97 | 0.31

Running | NaN | 0.00 | NaN

Elevator-up | 0.00 | 0.00 | NaN

Elevator-down | 0.38 | 0.09 | 0.15

Walking-up-stairs | NaN | 0.00 | NaN

Walking-down-stairs | NaN | 0.00 | NaN

**RandomForest Performance with all 30 features:**

**testAccuracy:0.94**

**Actual (same order)**

**Predicted**

|  |
| --- |
| 'Stationary' |
| 'Walking-flat-surface' |
| 'Running' |
| 'Elevator-up' |
| 'Elevator-down' |
| 'Walking-up-stairs' |
| 'Walking-down-stairs' |

48 0 0 0 0 0 0

1 59 0 0 0 0 0

0 0 51 2 0 2 1

4 0 0 27 0 0 0

2 0 0 2 28 0 0

0 0 0 0 0 50 0

0 6 0 0 0 1 40

**Activity wise Performance Metrics:**

Activity | Precision | Recall | F-1 Score

Stationary | 0.87 | 1.00 | 0.93

Walking-flat-surface | 0.91 | 0.98 | 0.94

Running | 1.00 | 0.91 | 0.95

Elevator-up | 0.87 | 0.87 | 0.87

Elevator-down | 1.00 | 0.88 | 0.93

Walking-up-stairs | 0.94 | 1.00 | 0.97

Walking-down-stairs | 0.98 | 0.85 | 0.91

**Comments:**

1. On the test data, Random Forest(94% accuracy) performs way better than KNN (19% accuracy).
2. Places where Random Forest makes a mistake is that it classifies a very few number of elevator up and

down instances as Stationary. For random forest all metrics are >=85% showing that it is an overall a

good classifier.

1. KNN on the contrary classifies almost everything as walking-flat-surface and therefore, the recall for

walking flat surface is very high. Even the precision for stationary is very high. But all other metrics ~0.

**Step 4: Random Forest on time domain and frequency features separately**

In order to switch between the feature domain, change the *domain* variable: Line 6 in randomForest.m to

‘time’ or ‘freq’.

**RandomForest Performance with only time domain features:**

**testAccuracy:0.92**

**Actual (same order)**

**Predicted**

|  |
| --- |
| 'Stationary' |
| 'Walking-flat-surface' |
| 'Running' |
| 'Elevator-up' |
| 'Elevator-down' |
| 'Walking-up-stairs' |
| 'Walking-down-stairs' |

48 0 0 0 0 0 0

1 59 0 0 0 0 0

0 0 47 2 0 2 5

5 0 0 26 0 0 0

5 0 0 1 26 0 0

0 0 0 0 0 50 0

0 6 0 0 0 0 41

**Activity wise Performance Metrics:**

Activity | Precision | Recall | F-1 Score

Stationary | 0.81 | 1.00 | 0.90

Walking-flat-surface | 0.91 | 0.98 | 0.94

Running | 1.00 | 0.84 | 0.91

Elevator-up | 0.90 | 0.84 | 0.87

Elevator-down | 1.00 | 0.81 | 0.90

Walking-up-stairs | 0.96 | 1.00 | 0.98

Walking-down-stairs | 0.89 | 0.87 | 0.88

**Performance with only frequency domain features RF:**

**testAccuracy:0.54**

**Actual (same order)**

**Predicted**

|  |
| --- |
| 'Stationary' |
| 'Walking-flat-surface' |
| 'Running' |
| 'Elevator-up' |
| 'Elevator-down' |
| 'Walking-up-stairs' |
| 'Walking-down-stairs' |

47 0 0 1 0 0 0

50 0 2 4 0 4 0

0 0 53 1 0 2 0

13 0 0 15 3 0 0

9 0 0 14 6 2 1

1 0 0 0 0 38 11

1 0 3 0 0 26 17

**Activity wise Performance Metrics:**

Activity | Precision | Recall | F-1 Score

Stationary | 0.39 | 0.98 | 0.56

Walking-flat-surface | NaN | 0.00 | NaN

Running | 0.91 | 0.95 | 0.93

Elevator-up | 0.43 | 0.48 | 0.45

Elevator-down | 0.67 | 0.19 | 0.29

Walking-up-stairs | 0.53 | 0.76 | 0.62

Walking-down-stairs | 0.59 | 0.36 | 0.45

**Comments:**

1. Overall test accuracy of time domain features (92%) is more than frequency domain features (54%).
2. Frequency domain fails to find any walking-flat-surface and elevator-down instances efficiently.
3. On the contrary, time domain features successfully classify all activities.
4. That means**, time domain features are the main features** that are essential for activity recognition, even

though **frequency domain features assist them** to **differentiate between few confusing activity instances**,

thereby increasing precision and recall when we classify using all features.

**Step 5: Cross Validation on Random Forest**

I have implemented cross validation with Random Forest Classifier in a separate file: crossValidation.m

**RandomForest Performance with Cross Validation:**

Overall Confusion Matrix:

A picture containing chart

Description automatically generated

**Cross-Validated Activity wise Performance Metrics:**

Activity | Precision | Recall | F-1 Score

Stationary | 0.83 | 0.83 | 0.83

Walking-flat-surface | 0.76 | 0.88 | 0.82

Running | 0.58 | 0.79 | 0.67

Elevator-up | 0.62 | 0.62 | 0.62

Elevator-down | 0.65 | 0.68 | 0.66

Walking-up-stairs | 0.87 | 0.63 | 0.73

Walking-down-stairs | 0.60 | 0.43 | 0.50

-x-x-x-