fallDetection353 Project by Amritpal Singh, David Liu, Sedat Demiriz

Data Recording - by all group members:

Our data was recorded using the app Physics Toolbox our instructor recommended. It proved suitable for our needs as it was able to give us the tools necessary for recording a person's movements along with various other functions not as relevant to our goal of primarily focusing on Linear Acceleration readings. Data was collected on 2 iPhone devices and a single Android device.

These readings consisted of columns for:

- time
- aX (acceleration along the X axis of the phone)
- aY (acceleration along the Y axis of the phone)
- aZ (acceleration along the Z axis of the phone)
- aT (total acceleration of the phone, calculated by:)

$$a_{total}=\sqrt{{a_x}^2+{a_y}^2+{a_z}^2}$$

We considered several actions that would give out similar accelerometer readings to that of falling, such as laying down, sitting down and dropping the device. All of these movements produced similar-looking readings and values in accelerometer data. Each of the 4 types of actions was recorded 20 times using the methodology described below using each member's own device.

We recorded our actions following a standardized procedure to make the readings consistent (also outlined on the README on the Git repository):

1. Beginning to record accelerometer data
2. Waiting 2 seconds to let the action to be fully recorded
3. Performing the corresponding action to the data type being collected
4. Waiting 5 seconds to let any residual movement to cease
5. Ending the accelerometer data recording

The raw data from these recordings can be found in the data-all folder as well as the folders labelled with the group members' initials on the repository if individual recordings are required for any detailed analysis. The renaming from the default names generated by Physics Toolbox was performed using the provided numbercsv.sh Bash script.

Noise in the Data: - by Amritpal Singh:

Due to the nature of sensors, a noticeable amount of noise was recorded alongside our desired signals. With the noise present, it was going to be difficult to make out the signal on its own and therefore low-pass high-pass filtering was used on all collected data from all types of actions to reduce the effect of noise on the signal.

For initial analysis, the X axes of all files of each action type were appended together, and similarly were the Y and Z axes. This collection of data contains a significant amount of noise, this is where the low-pass high-pass filtering was used. The filter increased the quality of our results a considerable amount, being able to reduce the effect the noise on the signal allowed us to better notice the trends in the data and compare the signals of different actions against each other.

Variation of data from activity to activity:

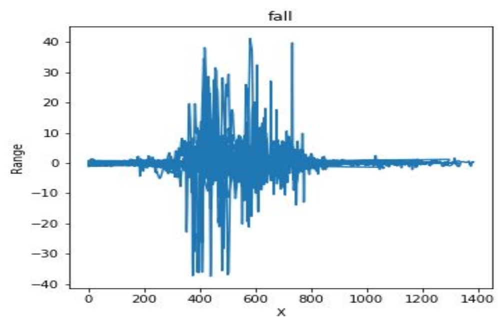
Our main goal was to analyze how the four types of actions related to each other and how we could differentiate between them. In each type of action, there is a variation in accelerometer readings from the X, Y and Z axes. We noticed that some actions have larger changes in these values compared to the rest.

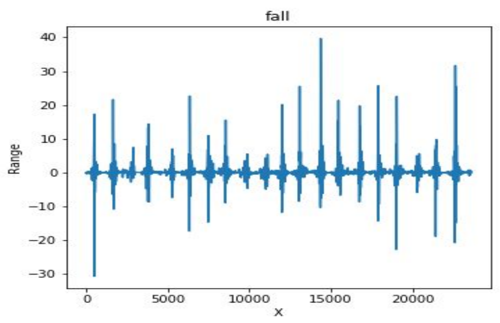
Based on this observation, plots and graphs were used to visualize the acceleration in each of the axes for each type of action. The graphs here show the effect filtering had on our readings. The plot of our data prior to the application of the filter does not give the same amount of clarity about the data compared to the plot after the filter was applied. After appending all X, Y and Z axes together and plotting the results, we were able to tell how many times the action occurred. This also showcases how the ranges of the accelerometer readings vary between each type of action.

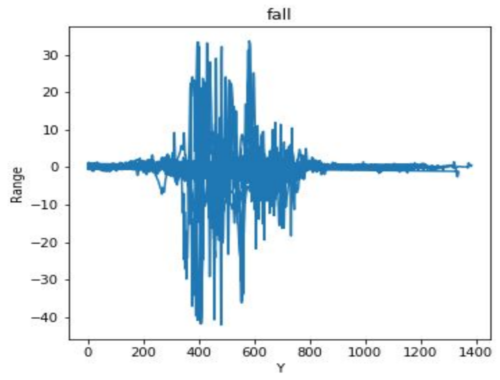
- Activity 1 - Walking and Falling

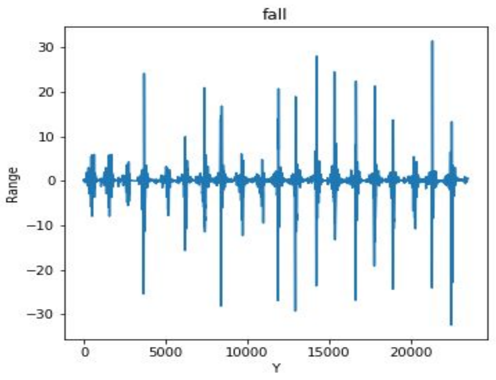
The graphs below consider the falling motion of a person while they were walking. The recorded data was put through low-pass high-pass filtering to remove noise. Each pair of graphs corresponds to pre-filtered and post-filtered data of the accelerometer data from the X, Y and Z axes.

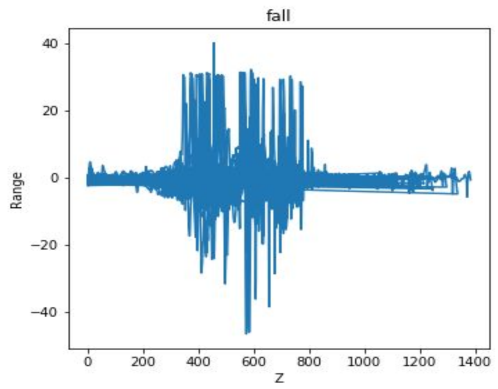
The filtered data is quite easy to understand compared to the unfiltered data. Notice that the filtered plots show the times at which the falls occur. The change in the range of movement in all axes is quite high during the falls: a range of values (-30, 40) in the X axis, a range of values (-30, 30) in the Y axis and a range of values (-40, 40) is observed. We can confirm that there was a large change in linear acceleration when the fall occurred.

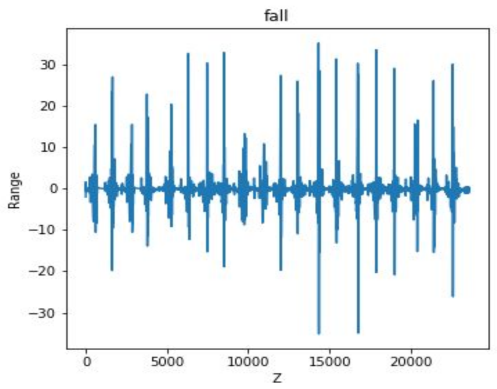








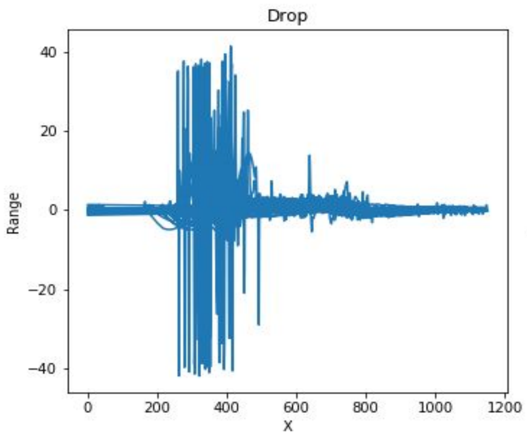


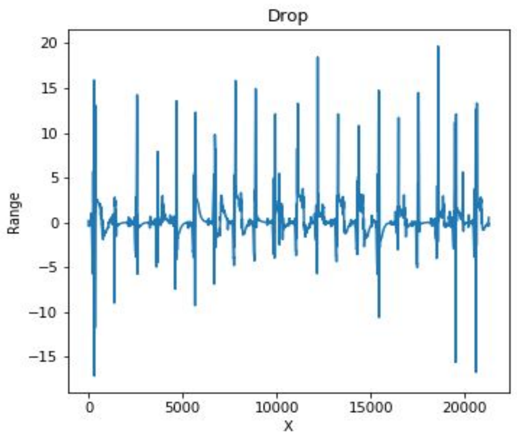


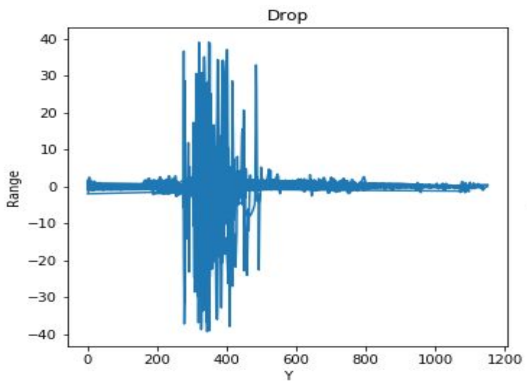
- Activity 2 - Dropping the Phone

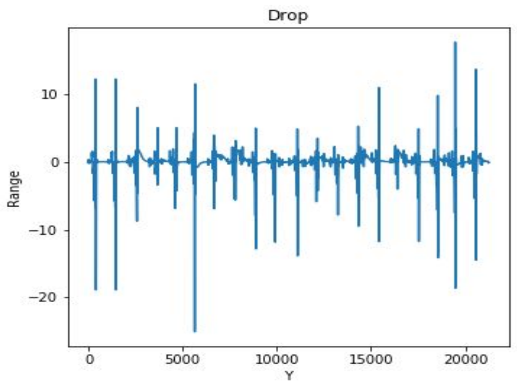
20 peaks can be seen on the filtered graphs below, which correspond to the 20 drops of the device. There are also numerous flatter sections between the large peaks during

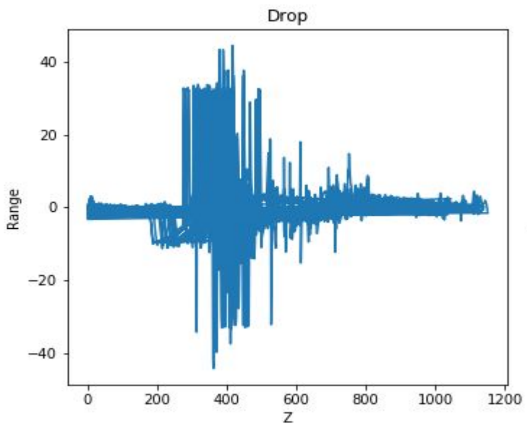
which the device was not moving as much, pointing to the times in the data collection routine where the device was kept steady for several seconds at a time. The linear acceleration measurement values in these graphs range from (-40, 40) for X, (-40, 40) for Y and (-40, 40) for Z.

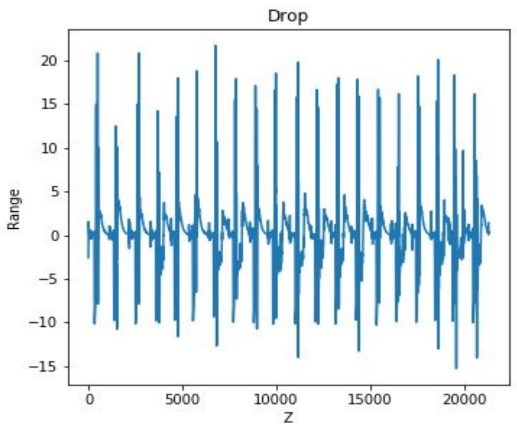






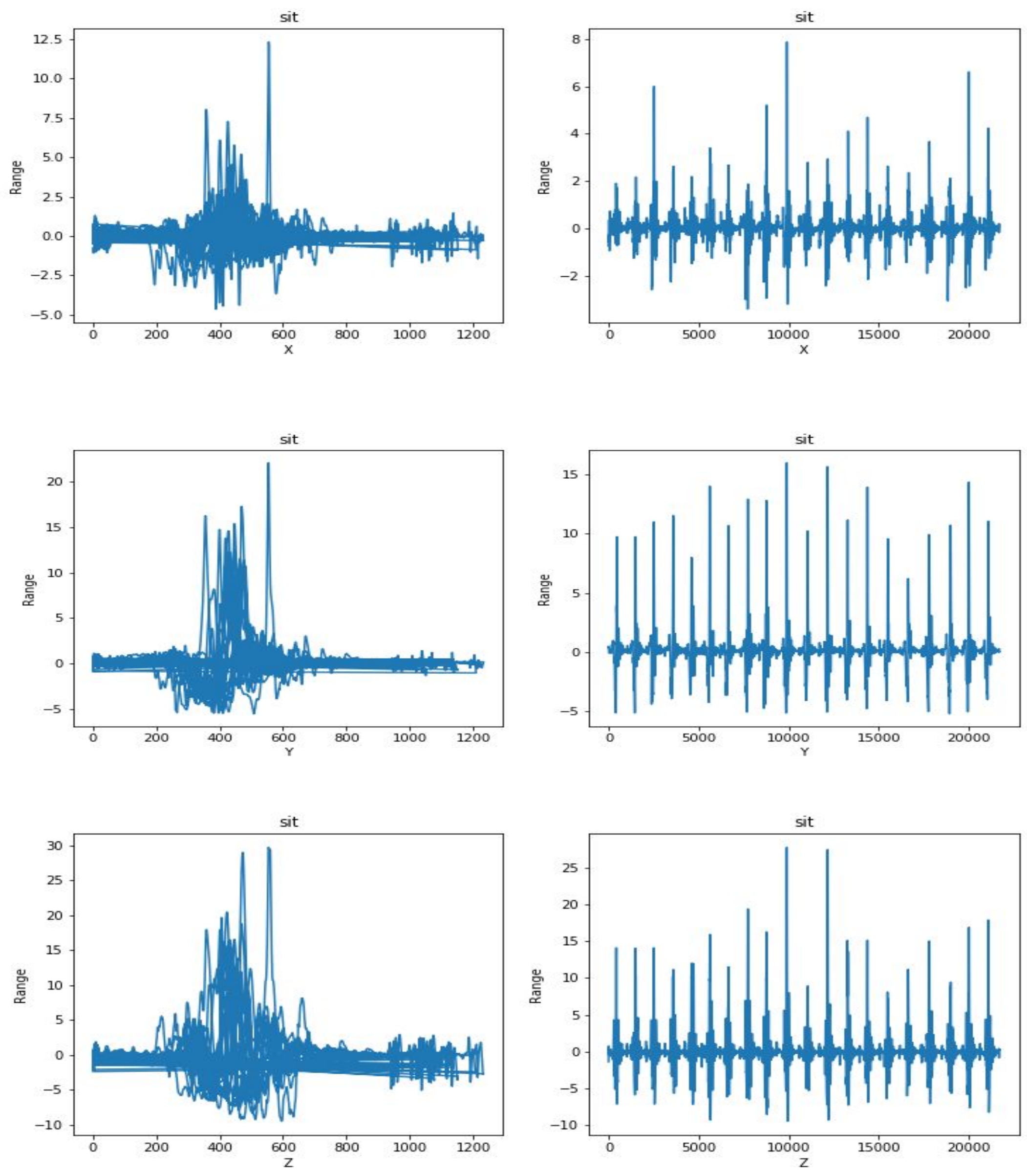






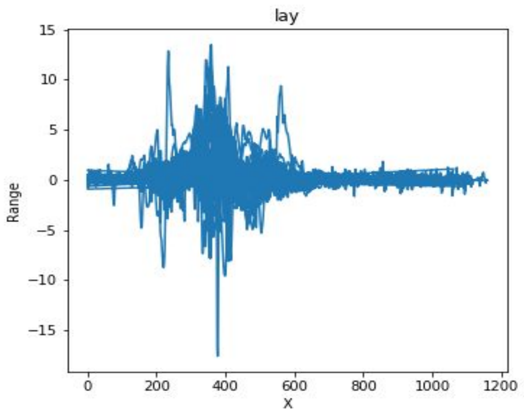
- Activity 3 - Sitting Down

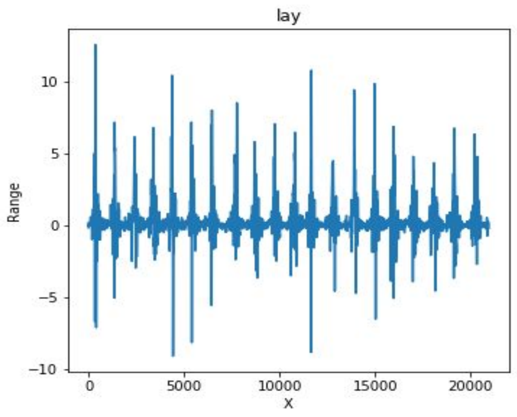
In this activity, the data was collected while performing a sitting action, and the similar use of low-pass high-pass filtering to remove noise from the data. The change in linear acceleration variation is the lowest in all three axes: (-5, 12) for the X axis, (-5, 20) for the Y axis and (-10, 30) for the Z axis. Comparing the action of sitting down with other actions confirms that sitting does not have as much variation in its linear acceleration values. The peaks in all axes are very closely related to each other and are quite uniform compared to other actions.

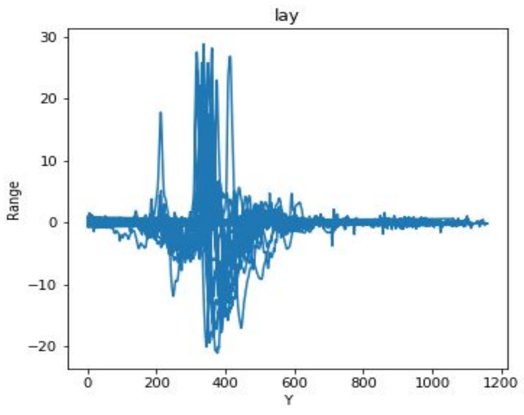


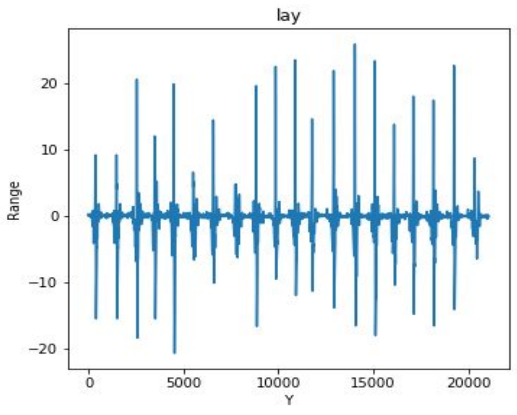
- Activity 4 - Laying Down

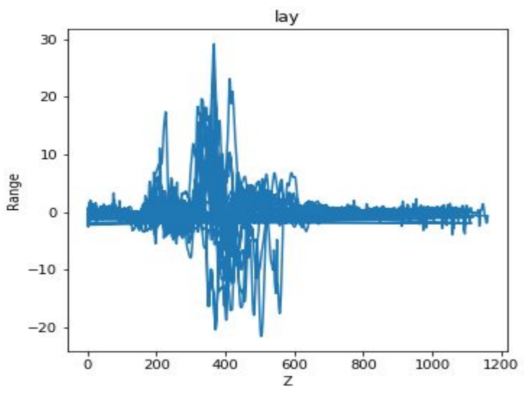
This activity features the action of laying down and once again filtering via low-pass high-pass. We thought this action would produce similar linear acceleration values to those of sitting down. However, this is not entirely right as seems from our data our peaks have turned out to be higher than those of sitting down. The range for X axis was (-15, 15), the range for the Y axis was (-20, 30) and the range for Z was (-20, 30).

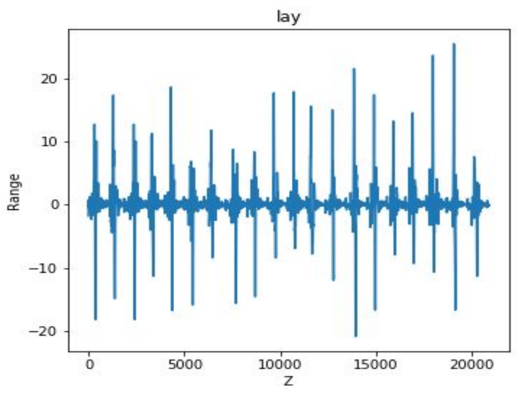












Data Overview

Falling Down:

The change in linear acceleration for the fall action is the second highest besides the drop action and the patterns closely relate to each other. This is likely to be difficult to differentiate using classifiers as the values are largely similar. The statistical analysis will require the extraction of meaningful values to let any sort of classifier to be able to correctly differentiate between these two action types.

There are large changes in linear acceleration values when the action was being performed. These values were generally less than or equal to those of the drop action. Value ranges are:

- (-30, 40) in the X axis
- (-30, 30) in the Y axis
- (-40, 40) in the Z axis

Device Dropping:

All axes in this action's data we have collected have distinctly high linear acceleration values, often rivaling those of the fall action. This is expected because the linear acceleration changes rapidly when the phone starts falling and then collides with the ground.

The ranges we have discovered for this action are the highest recorded ones from our selection of actions and their data. The distinctive variations show how quickly the change in linear acceleration occurs as the device is being dropped. Value ranges are:

- (-40, 40) in the X axis
- (-40, 40) in the Y axis
- (-40, 40) in the Z axis

Sitting Down:

This action has the lowest values out of all the actions when it comes to linear acceleration. This is likely due to sitting down being a controlled motion unlike the falls and drops where the body in question is likely to be experiencing freefall at some point, during which the acceleration values will be at their highest. The sit also has the least distance travelled from a standing position so acceleration will not be able to reach its potential peak values before the slow collision with the seat occurs.

Value ranges are:

- (-5, 12) in the X axis
- (-5, 20) in the Y axis
- (-10, 20) in the Z axis

Laying Down:

Linear acceleration is observed to be higher than that of the sit action but lower than those of the fall and drop actions. During a person laying down, according to our data, the linear acceleration does not change as much as in other scenarios.

As you can see, the value ranges are lesser than those of the fall or drop actions. Value ranges are:

- (-15, 15) in the X axis
- (-20, 30) in the Y axis
- (-20, 30) in the Z axis

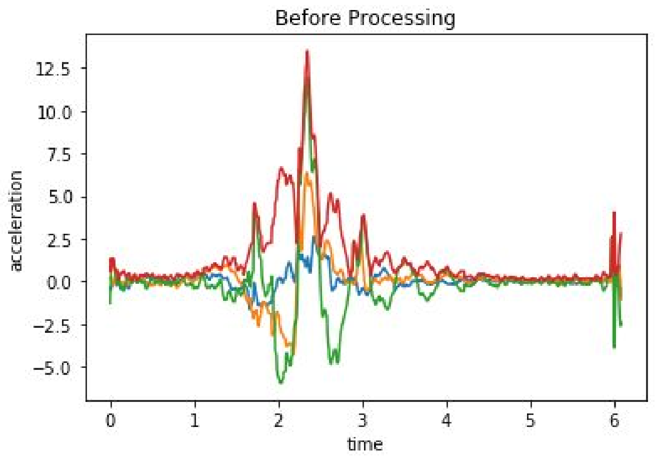
Overall, it can be concluded that the activity of dropping the device has the highest changes in linear acceleration we have seen, with falling being a close second due to the similar nature of the actions where the movement is not controlled by the person. In contrast, when laying down and then sitting down, which have the lower values out of all the measurements, the body is under control from the person and therefore is not likely to accelerate the same way as the actions mentioned before.

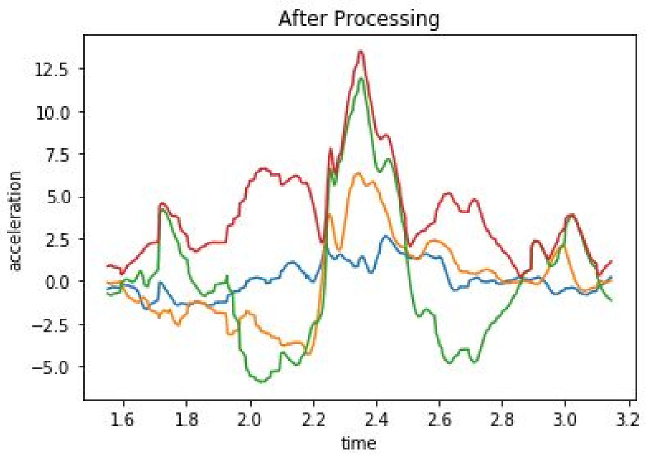
Data Processing and Statistics: - by David Liu

Most of the collected data contained large stretches of near zero acceleration due to the controlled pauses before performing any given action and after said action before the recording is ended. Each .csv file containing a particular action required trimming near the beginning and the end which has allowed us to see the patterns forming distinctly without being interrupted due to the recording starting late or ending prematurely.

Each action lies within an approximate duration of two seconds. In order to find the window where the data requires a trim the max value of the vector sum is computed. From the max value of the vector sum, we only keep the data from +/- 0.9 time frames of the time where the max vector sum occurs. The figures below illustrate the time versus acceleration graph prior and after trimming has been done.

Note the much clearer and distinct peak, being stretched thanks to the extra space the plot has received from the ends being trimmed off. Thanks to the trimming, we also have managed to avoid the lesser but consistent peaks at the beginning and the end of the recording, likely to be caused by the user interacting with the device when starting and cutting the recording.





After processing each .csv file for each action, the average acceleration in teh X, Y and Z axes relative to the phone is calculated along with the average vector sum. Additionally the maximum values of vector sum and acceleration in X, Y, Z axes was extracted. The values obtained were then stored in rows of a pandas dataframe. Each created dataframe represents its own separate action of falling, sitting, laying down or dropping. Each row of the dataframes represents a single instance of the given action.

- Analysis of Variance Test

An ANOVA test was performed on each of the various attributes collected from the data processed data. Each of the acceleration averages in all axes X, Y, Z and averages from the vector sum were tested. Therefore 4 separate ANOVA tests were performed to determine if there was a difference in the means of each separate action for each attribute. Additionally, the same was performed for maximum values for a total of 8 separate ANOVA tests.

The results of each of the 8 ANOVA tests produced p-values significantly lower than α = 0.05. Therefore, the means of the tested attributes for each action have some significant differences.

- Post Hoc Analysis

Due to the significance of the 8 ANOVA tests, 8 separate Post Hoc Analyses were performed to complement the ANOVA tests. For the testing of average vector sums, it was found that dropping our device and falling have similar means which seems consistent with the ranges we observed and conclusions we drew in the plot from the previous section. Likewise, for the average acceleration in the X axis drop vs fall and fall vs lie have similar means. The remaining data regarding the fall action versus other actions against the remaining attributes resulted in significant differences in means.

As a result of this Post Hoc Analysis, it was revealed that it could be possible to identify a falling action versus sitting and lying down. However, it may be difficult to differentiate between dropping a phone and falling.

Sample Tukey HSD Analysis result:

```
>>>print(posthoc_avg_aT)

   Multiple Comparison of Means - Tukey HSD, FWER=0.05
==============================================================
  group1       group2     meandiff  p-adj    lower     upper    reject
---------------------------------------------------------------------------------------------
drop_avg_aT  fall_avg_aT   0.4701   0.4491  -0.3513    1.2914    False
drop_avg_aT  lie_avg_aT   -1.3672   0.001   -2.1886   -0.5459    True
drop_avg_aT  sit_avg_aT   -2.864    0.001   -3.6853   -2.0426    True
fall_avg_aT  lie_avg_aT   -1.8373   0.001   -2.6587   -1.0159    True
fall_avg_aT  sit_avg_aT   -3.3341   0.001   -4.1554   -2.5127    True
 lie_avg_aT  sit_avg_aT   -1.4968   0.001   -2.3181   -0.6754    True
---------------------------------------------------------------------------------------------
```

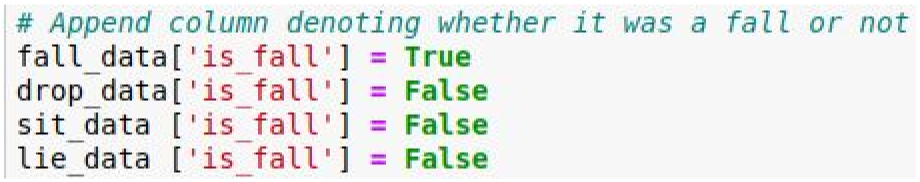
Machine Learning and Classifiers - Sedat Demiriz

The project title itself implies a classification problem: we are not trying to build a model to predict future falls, nor are we trying to find a trend in the actions we recorded via regressions. We are simply looking at training a model to be able to decide whether a data pattern it has been fed belongs to that of a fall or not. We are not looking to individually classify all 4 actions we recorded to be able to differentiate between them. All we really care about is whether a certain reading is likely to be a fall or not.

From this starting point, the question becomes whether there is any significant difference between the readings we get from a fall compared to any other action a person might take that involves a similar motion such as sitting down or dropping their phone.

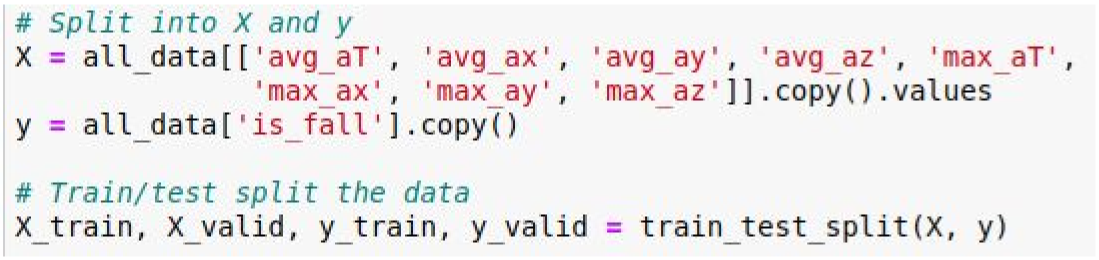
According to the results of my group members have obtained after noise filtering and statistical analysis of means and maximum from our gathered data, there indeed was a noticable difference between the actions. Drops and falls were close in their data patterns, and would likely be difficult to differentiate from each other. Sits and lie-downs on the other hand were very different in their readings and were likely to be able to be differentiable from the other two actions.

I was able to reshape my group members' findings into a single pandas dataframe, and added a column depicting whether the corresponding reading belonged to a fall or not via True/False values.



```
# Append column denoting whether it was a fall or not
fall_data['is_fall'] = True
drop_data['is_fall'] = False
sit_data ['is_fall'] = False
lie_data ['is_fall'] = False
```

With 8 values to draw from, the average and maximum columns from the dataframe were stacked to form individual observations and the ninth boolean value was used to denote whether the observation was a fall or not. The data was split using the train and test split routine. This reshaped data was then fed into several classification algorithms from sklearn.



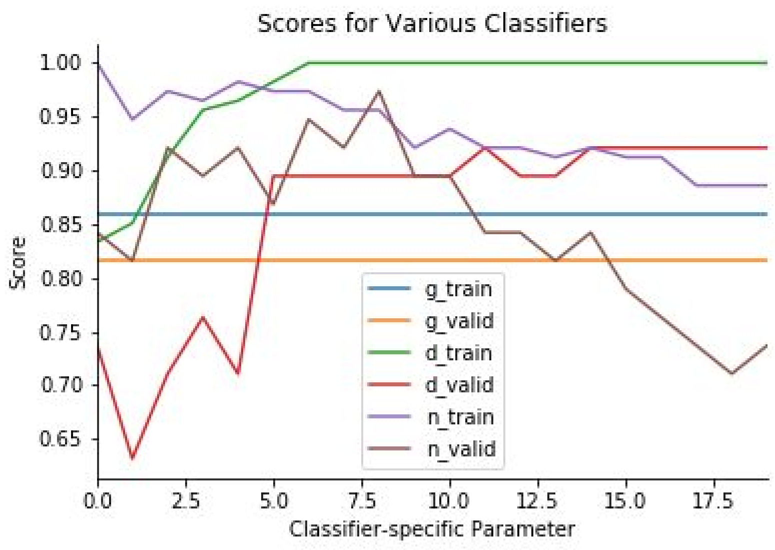
```
# Split into X and y
X = all_data[['avg_aT', 'avg_ax', 'avg_ay', 'avg_az', 'max_aT',
              'max_ax', 'max_ay', 'max_az']].copy().values
y = all_data['is_fall'].copy()

# Train/test split the data
X_train, X_valid, y_train, y_valid = train_test_split(X, y)
```

Based on the various classifiers covered in class, I opted to go with three variants: a Naive Bayes Gaussian Classifier, a N Nearest Neighbors Classifier as well as a Decision Tree Classifier. My considerations were:

- A NB Gaussian classification would give a baseline accuracy from which we would be able to improve by applying more sophisticated classifiers.
- A N Nearest Neighbors classification could be useful if even despite not having drastically different values between fall and drop actions they could cluster separately on the 8 axes of data that we have. The other two actions are already easier to classify as different values.
- A Decision Tree classification could catch small numerical differences and split the data space into distinct parts where even the small differences between groups would likely be caught.

After running several manual model fits and scoring runs, I observed some interesting scores using different algorithms as well as within the same algorithm with different parameters: the number of neighbors or the decision tree depth used. Not satisfied with the small range of values I could manually test and note down, I wrote a score benchmarking suite of functions that would train models for a given parameter and represent all the results on a single plot for visualization:



The plot contains the training and validation results for:

- g: the NB Gaussian Classifier
- d: the Decision Tree Classifier
- n: the N Nearest Neighbors Classifier

Due to the different training and test splits that occur with each run of the benchmarker, the plot is not static, with some classifiers occasionally scoring higher during certain runs than what is shown in this specific plot.

Please do note that the X axis of the plot is not meant to be consistent or represent any comparison of the methods at some value. The plot is simply there to find out what values for N Nearest Neighbors and Decision Tree depth gave the best results for our dataset. The NB Gaussian method does not have any parameters to tune in the benchmarker, and therefore the results remain constant throughout.

When it comes to scoring each method of classification, the highest peak value we observe for any validation score is around low x values (generally at or lower than x = 8), for the N Nearest Neighbors Classifier. A close runner up was the Decision Tree Classifier, with accuracy scores about 5% behind. The Gaussian Classifier gave an ok reading overall, and was a correct choice for a baseline algorithm.

All in all, at 8 Nearest Neighbors, the NNN Classifier performed its best and the best overall, achieving a 96% score on our data, proving to be the most reliable method to use for Fall Detection using our methodology.

Accomplishment Statements - Amritpal Singh

- Collected linear acceleration data 20 times for each of fall, drop, lay and sit activity through the Physics Toolbox application.
- Merge and append all the data dimension together for each activity.
- Filtered the noise from data through low pass high pass filtering.
- Each activity data visualized through graphs and plots for each dimension.
- Computed that dropping the phone has the highest variation in linear acceleration.

Fall Detection Project Overview – David Liu

- Collected various data using Physics Tool Box application which resulted in 20 different data points for 4 distinct actions.
- Created a Python program to processed raw data for reducing unnecessary noise which highlighted key areas of interests necessary for further analysis.
- Produced csv files containing potential attribute candidates from processed data which condensed collected data into a ready to be used state for further analysis.
- Performed ANOVA tests and subsequent complementary post hoc analysis which resulted in the identification of candidate attributes for machine learning classification.

Accomplishment Statements - Sedat Demiriz

- Collected linear acceleration data using the Physics Toolbox application, for a total of 20 .csv file format recordings per each of the 4 types of actions.
- Designed and implemented a Classifier-based Machine Learning system in Python, using the sklearn library's built-in classifier methods of Gaussian, N Nearest Neighbors and Decision Trees that was able to successfully and consistently differentiate between falls and other similar types of actions from data collected via handheld device linear accelerometers.
- Implemented an additional benchmarking step to be able to observe what classification model performed the best for any given run of the script, visualized by a plot output and best accuracy score values for each type of classifier.
- Wrote a Bash script to automatically rename and number .csv files in a given folder to be able to later use the name template for easy access and imports of the files into Python or potentially other environments.
- Managed and kept up to date a working Git repository to unify group members' working environment and keep access to all directories and files consistent.