Lab 1 Report

Kyu Park, Dingyi Sun, Kendrick Xie

1. Complete summarize_sensor_trace() to compute the mean and variance of each of the 36 sensor attributes in a given CSV file.

```python
def summarize_sensor_trace(csv_file: str):
    df = pd.read_csv(csv_file)
    # list of lists of means and variances
    vals = []
    # list of column (attribute) names
    names = []
    for (columnName, columnData) in df.iteritems():
        # exclude average of time or unnamed column
        if columnName in ('time', 'Unnamed: 37'):
            continue

        names.append(columnName)
        vals.append([columnData.mean(), columnData.var()])
        # print("Mean of " + columnName + ": " +
str(columnData.mean()))
        # print("Variance of " + columnName + ": " +
str(columnData.var()))

    ret = pd.DataFrame(vals, columns = ['Mean', 'Variance'],
index = names)
    # print(ret)
    # returns a dataframe of means and variances of each
attribute
    return ret
```

The function summarize_sensor_trace() computes the mean and variance of each of the 36 sensor attributes of a given CSV file as such.

2. Complete visualize_sensor_trace() to graph a dimension of an attribute's value (e.g. controller_left_vel.x) as a function of time for a given CSV file. Remember to label your axes. You may also modify this function to graph all three dimensions of the given attribute (e.g. headset_rot) at once.

```python
def visualize_sensor_trace(csv_file: str = "", attribute: str =
"", single_attribute_mode : bool = True):
    # if the function is required to plot a time vs single
attribute
    if single_attribute_mode:
        # plot a attribute vs time plot
        df = pd.read_csv(csv_file)
        ax = df.plot(x = 'time', y = attribute)
        ax.set_xlabel("Time (milliseconds)")
        ax.set_ylabel(attribute)
        plt.show()
```

The function visualize_sensor_trace() graphs a dimension of a given attribute's value as a function time for a given CSV file if in single_attribute_mode.

3. Use the previous functions to determine attribute trends across all data samples for a given activity. Which attributes best help distinguish the sensor traces of one activity compared to another?

```python
def visualize_sensor_trace(csv_file: str = "", attribute: str =
"", single_attribute_mode : bool = True):
    # if the function is required to plot a time vs single
attribute
    if single_attribute_mode:
    else:
        # combine data samples for multiple activities
        Standing = combine_samples("STD")
        Sitting = combine_samples("SIT")
        Jogging = combine_samples("JOG")
        Stretching = combine_samples("STR")
        Overhead = combine_samples("OHD")
        Twisting = combine_samples("TWS")
```

```
        dataframes = [Standing, Sitting, Jogging, Stretching,
Overhead, Twisting]

        for i in range(12):
            compare_attributes(dataframes, i*3, i*3+3, "Mean")

        # Output graphs for variance of different activities
        for i in range(12):
            compare_attributes(dataframes, i*3, i*3+3,
"Variance")
```

We use visualize_sensor_trace() once again, but not in single_attribute_mode to compare attribute values (mean and variance) for activities. The function uses custom helper functions compare_attributes() and combine_samples(), as well as the previous function summarize_sensor_trace() to compute mean and variance for each attribute.

We have determined that the following mean values were meaningful distinguishing the sensor traces of one or more activity:

- headset_vel.x being comparatively negative (< -0.001000) suggests jogging
- headset_pos.y being negative (< 0.000000) suggests sitting
- headset_angularVel.x being comparatively large (> 0.010000) suggests twisting
- controller_left_vel.x being comparatively negative (< -0.020000) suggests stretching and relatively large (> 0.010000) suggests twisting
- controller_left_vel.y being very large (> 0.030000) suggests overhead
- controller_left_pos.x being very negative (< -0.400000) suggests stretching
- controller_left_pos.y being very negative (< -0.600000) suggests sitting
- controller_right_vel.x being large or positive (> 0.020000) suggests stretching
- controller_right_vel.y being very large (> 0.040000) suggests overhead
- controller_right_pos.x being very large suggests stretching (> 0.400000)
- controller_right_pos.y being very negative suggests sitting (< -0.600000)

The following are variance values that were meaningful distinguishing the sensor traces of one or more activity:

- headset_vel.y being very large (> 0.100000) suggests jogging
- headset_angularVel.y being very large (> 2.000000) suggests twisting

- controller_left_pos.y and controller_right_pos.y being relatively large or positive (> 0.120000) suggests overhead

These attributes were useful since at least one activity shows an irregular pattern (comparatively large or small) in each of these attributes.

4. Which attributes are less useful for distinguishing the differences between activities? Why?

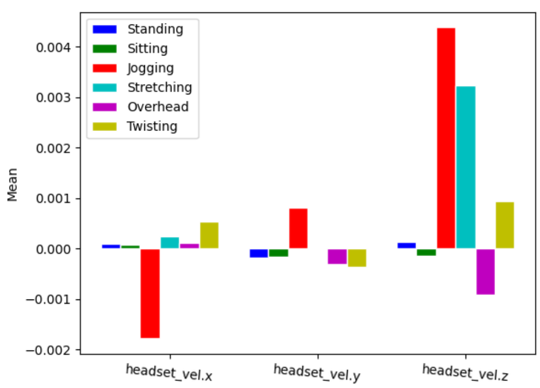
   None of the rotation attributes were very useful because their means and variances were not very different between different activities. Any other attributes not listed in best attributes were not very useful for the same reason. Additionally, if one of these attributes was relatively different for some activities, each activity was still close to at least one other activity.

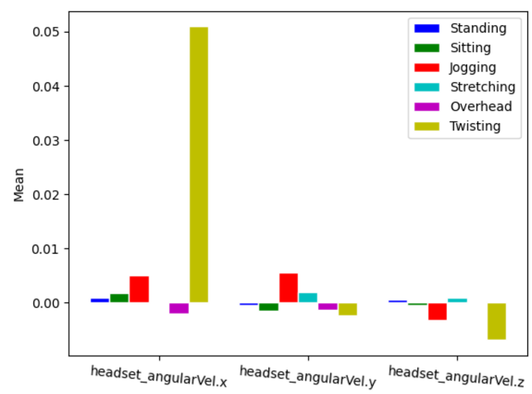
5. Compute the mean and variance of the significant attributes across all data samples, then present them in a table for all six activities. Also, include visualizations that highlight how motion patterns differ between activities for certain attributes. Explain how the statistics and visualizations support your answer to #3.

	Standing		Sitting		Jogging		Stretching		Overhead		Twisting	
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance
headset_vel.x	0.0001	0.0000	0.0001	0.0000	-0.0018	0.0114	0.0003	0.0005	0.0001	0.0003	0.0005	0.0290
headset_vel.y	-0.0002	0.0000	-0.0002	0.0000	0.0008	0.1224	0.0000	0.0003	-0.0003	0.0004	-0.0004	0.0006
headset_angular Vel.x	0.0008	0.0004	0.0018	0.0004	0.0050	0.2078	0.0002	0.0136	-0.0020	0.0310	0.0510	0.0479
headset_angular Vel.y	-0.0004	0.0004	-0.0015	0.0004	0.0055	0.0353	0.0019	0.0090	-0.0014	0.0089	-0.0024	2.9812
headset_pos.y	0.3361	0.0000	-0.2068	0.0000	0.2106	0.0006	0.2197	0.0000	0.1840	0.0000	0.0383	0.0000
controller_left_vel.x	0.0001	0.0002	0.0008	0.0000	-0.0024	0.0741	-0.0293	0.7910	-0.0008	0.0208	0.0109	0.4763
controller_left_vel.y	-0.0013	0.0001	-0.0001	0.0000	0.0033	0.3724	0.0027	0.0150	0.0419	1.0468	-0.0039	0.0069
controller_left_pos.x	-0.1156	0.0001	-0.0957	0.0000	-0.2109	0.0018	-0.4258	0.0543	-0.1699	0.0008	0.0095	0.0340
controller_left_pos.y	-0.4308	0.0001	-0.7071	0.0000	-0.3007	0.0042	0.0691	0.0008	0.1673	0.1311	-0.3586	0.0006

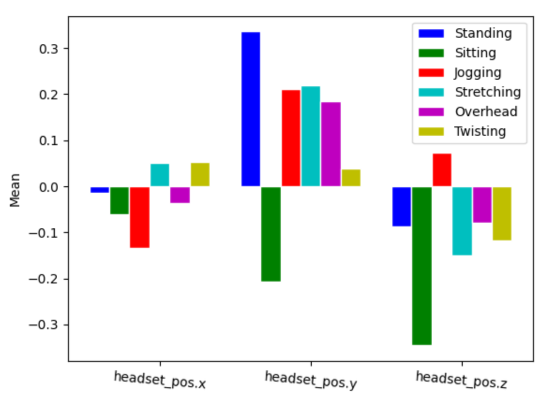
controller_right _vel.x	-0.0003	0.0002	0.0004	0.0001	0.0003	0.1450	0.0321	0.8160	-0.0038	0.0210	-0.0139	0.5182
controller_right _vel.y	0.0004	0.0001	0.0002	0.0000	0.0042	0.9561	0.0015	0.0116	0.0469	1.0531	-0.0004	0.0076
controller_right _pos.x	0.1292	0.0002	0.0797	0.0000	-0.0056	0.0028	0.5285	0.0668	0.1041	0.0008	0.1128	0.0383
controller_right _pos.y	-0.4325	0.0001	-0.7136	0.0000	-0.3441	0.0108	0.0540	0.0007	0.1693	0.1312	-0.3359	0.0006

Mean of Attributes:

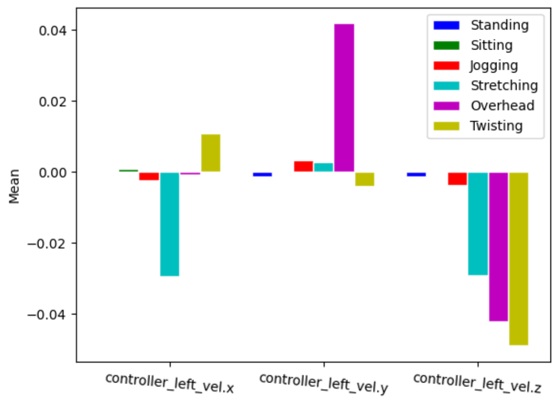


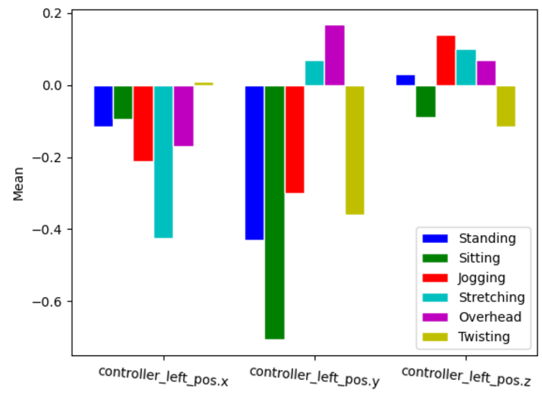


- headset_vel.x being comparatively negative (< -0.001000) suggests jogging
- headset_angularVel.x being comparatively large (> 0.010000) suggests twisting

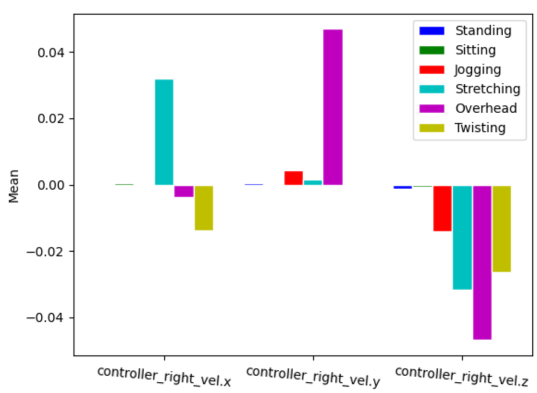


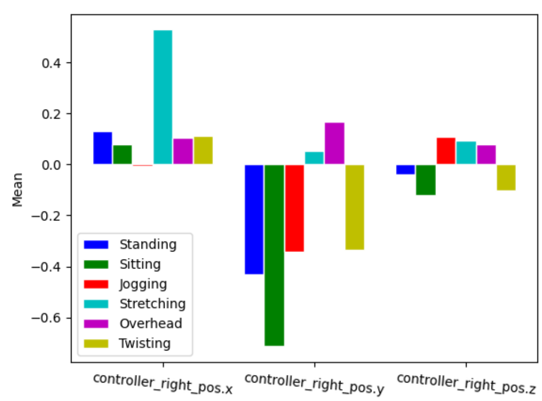
- headset_pos.y being negative (< 0.000000) suggests sitting





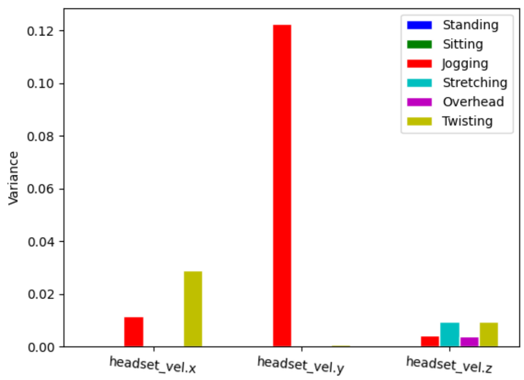
- controller_left_vel.x being comparatively negative (< -0.020000) suggests stretching and relatively large (> 0.010000) suggests twisting
- controller_left_vel.y being very large (> 0.030000) suggests overhead
- controller_left_pos.x being very negative (< -0.400000) suggests stretching
- controller_left_pos.y being very negative (< -0.600000) suggests sitting

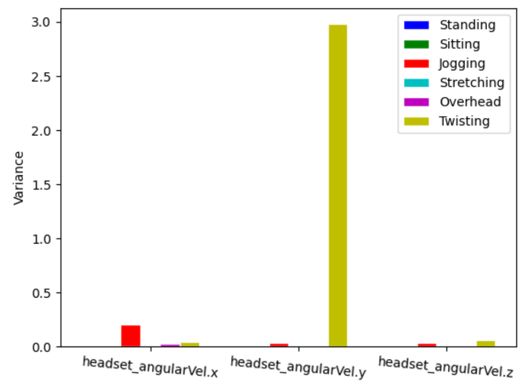




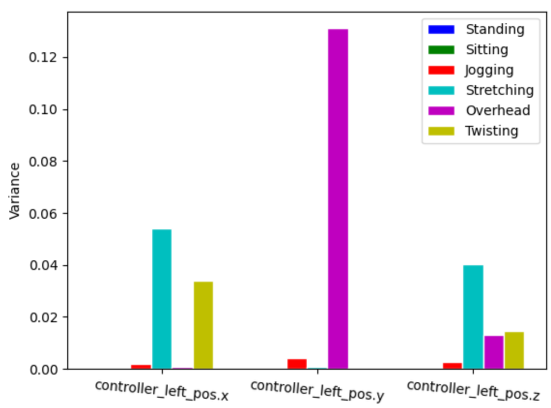
- controller_right_vel.x being large or positive (> 0.020000) suggests stretching
- controller_right_vel.y being very large (> 0.040000) suggests overhead
- controller_right_pos.x being very large suggests stretching (> 0.400000)
- controller_right_pos.y being very negative suggests sitting (< -0.600000)

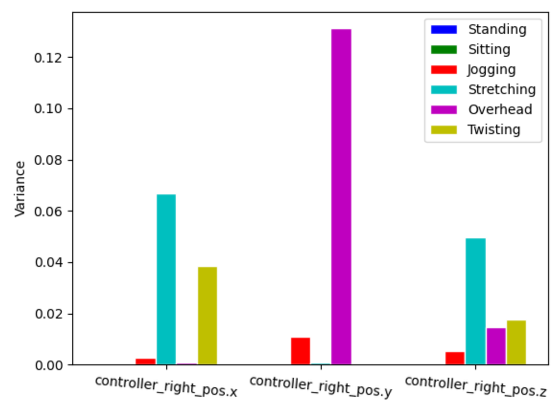
Variance of Attributes:





- headset_vel.y being very large (> 0.100000) suggests jogging
- headset_angularVel.y being very large (> 2.000000) suggests twisting





- controller_left_pos.y and controller_right_pos.y being relatively large or positive (> 0.120000) suggests overhead

6. Using the significant attributes found above, describe how you would design and implement a simple algorithm (e.g. using statistical thresholds) to determine which activity was performed for a given sensor trace. Your algorithm should work regardless of how long a data sample is.

   The Algorithm:

   > The algorithm calculates the weighted distance between two points that represent two activities (one is the standard value by averaging the training set, the another is the unknown activity we want to classify) in a 72-dimensional feature space

(constituted by mean and variance of all sensor data types). Each feature is assigned with an unique weight that corresponds to the relative importance of the feature in distinguishing different activities. Specifically, dist = ((f_input - f_standard)/f_standard)^2*w.

The initial weights are 1 for all features, and then we assign a weight ranging from 3 to 8 to features that is considered significant in previous steps of Lab1 to each feature. A demonstration of classification results is shown here:

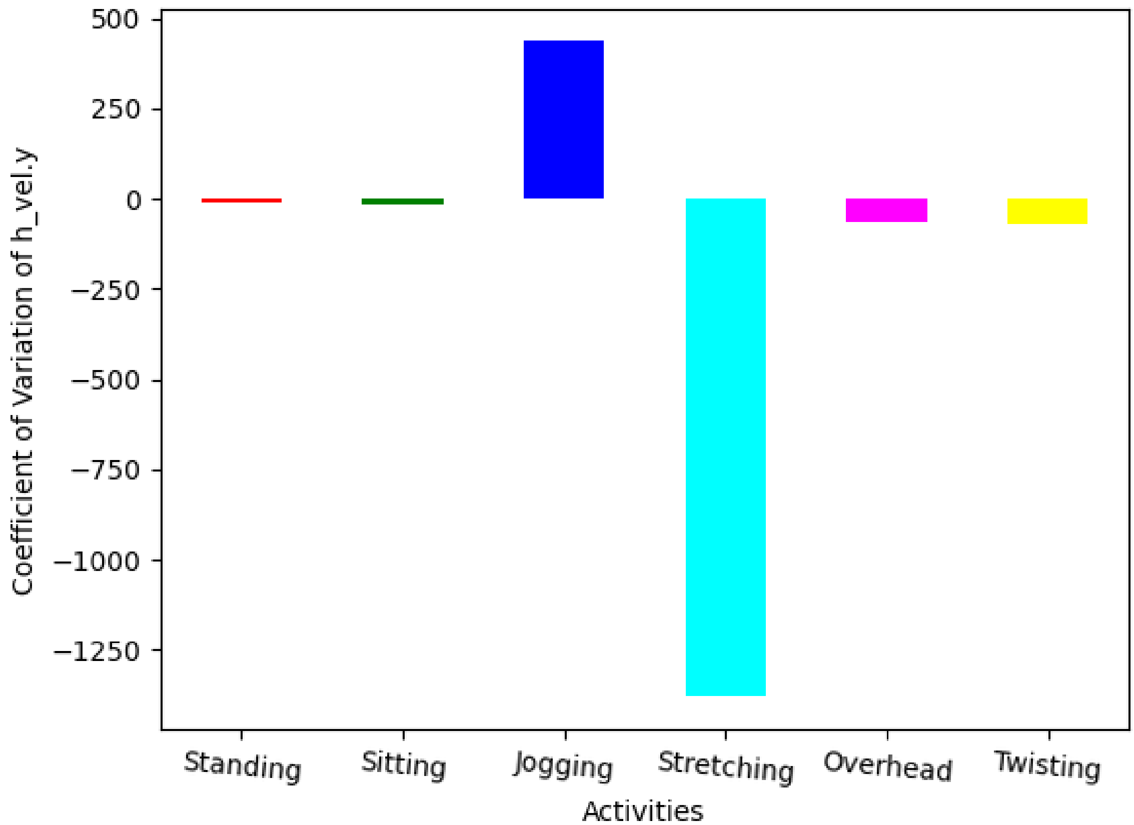
```
11467@Acht-Razor MINGW64 /d/code/mobile/mc-labs-digital-dominators/Python (main)
$ PY classify.py

TESTING INITIAL WEIGHTS
    Mean weights:  [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
    Var weights:   [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

Test 1
    Activity  JOG_P2_01.csv  is classified as:  JOG     Dists =  [2708, 49771, 99819, 1015580, 383013052, 4867896163]
    Activity  STD_P3_03.csv  is classified as:  TWS     Dists =  [256, 270, 319, 705, 772, 1633]
    Activity  STR_P2_02.csv  is classified as:  STR     Dists =  [474, 30488, 48039, 80666, 8436015477, 23606464319]
    Activity  OHD_P1_05.csv  is classified as:  OHD     Dists =  [143, 11147, 13019, 67053, 7067582897, 52020858907]
    Activity  TWS_P3_01.csv  is classified as:  TWS     Dists =  [19940, 33478, 63570, 95121, 6694169968, 21120826767]
    Activity  SIT_P1_03.csv  is classified as:  OHD     Dists =  [194, 316, 387, 427, 741, 4577313]

    Accuracy:  0.6666666666666666


Test 2
    Activity  JOG_P1_03.csv  is classified as:  JOG     Dists =  [4671, 35074, 77465, 2896231, 873502902, 8242159538]
    Activity  STD_P1_05.csv  is classified as:  STR     Dists =  [79, 82, 112, 323, 437, 702]
    Activity  STR_P3_03.csv  is classified as:  STR     Dists =  [4217, 25775, 44989, 67032, 6994655630, 25238610384]
    Activity  OHD_P3_02.csv  is classified as:  OHD     Dists =  [234, 1232, 32360, 62074, 3661224758, 17144594062]
    Activity  TWS_P2_03.csv  is classified as:  TWS     Dists =  [3112, 11334, 186552, 301266, 1500474266, 3935875946]
    Activity  SIT_P1_01.csv  is classified as:  JOG     Dists =  [567, 866, 1540, 2053, 3230656, 10556214]

    Accuracy:  0.6666666666666666


TESTING CORRECTED WEIGHTS
    Mean weights:  [1 1 1 1 1 1 3 8 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
    Var weights:   [1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 3 1 1 1 3 1 3 1 1 1 1 1 3 1 1 3 1 1 3 1 1]

Test 1
    Activity  JOG_P2_04.csv  is classified as:  JOG     Dists =  [3836, 31887, 74544, 9835692, 324446876, 10026769367]
    Activity  STD_P3_05.csv  is classified as:  STD     Dists =  [203, 216, 365, 686, 880, 1660]
    Activity  STR_P2_05.csv  is classified as:  STR     Dists =  [1251, 61310, 66584, 9033223, 6328190513, 26560590815]
    Activity  OHD_P1_04.csv  is classified as:  OHD     Dists =  [147, 4999, 22350, 74009, 3225954564, 55809902171]
    Activity  TWS_P3_04.csv  is classified as:  TWS     Dists =  [6850, 43264, 129915, 195161, 6012940078, 16590362996]
    Activity  SIT_P1_04.csv  is classified as:  SIT     Dists =  [179, 352, 490, 929, 8655, 9276784]

    Accuracy:  1.0


Test 2
    Activity  JOG_P1_04.csv  is classified as:  JOG     Dists =  [3456, 46355, 85565, 8663282, 1033941545, 21150651478]
    Activity  STD_P1_01.csv  is classified as:  STD     Dists =  [219, 309, 791, 1119, 1935, 2908]
    Activity  STR_P3_05.csv  is classified as:  STR     Dists =  [146, 38032, 41748, 65289, 88859112, 16740573683]
    Activity  OHD_P3_04.csv  is classified as:  OHD     Dists =  [326, 1873, 54010, 117047, 3420570778, 29449134710]
    Activity  TWS_P2_04.csv  is classified as:  TWS     Dists =  [3258, 9327, 79996, 123330, 3259776846, 5882486546]
    Activity  SIT_P1_02.csv  is classified as:  SIT     Dists =  [364, 383, 473, 1362, 8229984, 98659871]

    Accuracy:  1.0
```

We can conclude that, by assigning appropriate weights to significant features, we improve the accuracy of our classifier from 67% to 100%.

This algorithm is demonstrated in classify.py

7. What additional features you could derive from the provided raw data that would be helpful for distinguishing between activity types? Compute at least one feature, then graph it and show how it can be used to distinguish between two different activities.

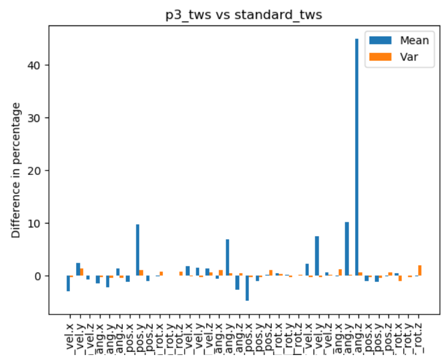
Since mean and variance could be obtained from the raw data, we can use these to compute the coefficient of variation, which is sqrt(variance) / mean. This could be useful in distinguishing between activity types, because we can standardize the variation in relation to the mean, and we can see how the pattern of the CV differs by each activity. It could provide more information than just simply using variance, because now we consider both variance and mean in coefficient of variation.

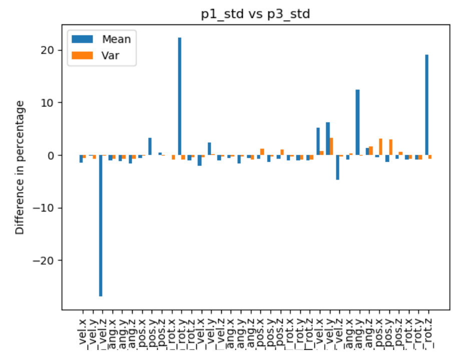


Here, in the CV of headset_vel.y, we observe that CV is high for jogging and extremely low for stretching. We could possibly use CV of headset_vel.y to determine whether an activity is jogging or stretching, or other activities. We have previously determined that the variance of headset_vel.y being very large (> 0.100000) suggests jogging, so it is not surprising that we the CV of headset_vel.y is high for jogging, but we could also additionally determine stretching by using CV, which could have not been done if we only used means and variance to determine traits.

8. Examine the data traces between group members for the same activity. Do you think individual group members can be identified by their data traces? Why/why not? What implications does this have on the security of VR systems?

Individual group members can be identified since we observe a significant difference in variance and mean of their data. Using the same algorithm described in Q6, we compare the features of different members and the averaged feature of all members:





We observe that among group members and when compared to the standard values, there can be a difference ranging from 0 - 60%. By producing an averaged feature list of each group member, we can identify an unknown activity by calculating the distance of it in feature space to that of all known members, and the minimum of these distances corresponds to the executioner of that activity.

Using this method, VR devices can identify if a user is authorized by comparing his/her pattern to the authorized users' stored in its harddrive.

This algorithm is demonstrated in person_compare.py

9. After completing the lab, clearly state the contribution of each individual in your team in your report.

Kyu Park:

- Task 1, 2, 3, 4, 5, 7
- Designed summarize_sensor_trace and visualize_sensor_trance functions and helped create functions to create bar graphs, designed algorithm for CV and made graphs for CV

Kendrick Xie:

- Task 1, 2, 3, 4, 5
- Helped with summarize_sensor_trace and visualize_sensor_trance functions, designed functions to create bar graphs, and made graphs for comparing mean and variance of attributes

Dingyi Sun:

- Task 6, 7, 8
- Design of classification and member identification algorithms.