**Alejandro Oliveros Emmanouil Manouselis Georgios Savvidis Johan Bini Rafael Silva**

**Applied Artificial Intelligence**

**D7062E, Artificial Intelligence and Pattern Recognition, Lp1, H22**

**Task 01 – Group 08**

1. **Introduction / The task**

This report is a partial requirement for the course Artificial Intelligence and Pattern Recognition(D7062E), held as part of the international master’s program Applied Artificial Intelligence at Luleå University of Technology, fall 2022. This is the first part of three of a project.

In this very first part, our task is to pre-process and prepare a training dataset that will be used by a couple of machine learning algorithms in later tasks of the project.

1. **Pre-processing the data**

In preparing the data in a form that will be meaningful in an AI algorithm, we create a pre-processing function (python) or process (rapidminer). In this preliminary step of the overall process, two issues are addressed, the normalization of raw data and the handling of missing values.

* 1. **Normalizing the data**

By normalizing them, the raw data are transformed so that they are representing the underlying gesture in a common scale and in reference to a relative to the observed person frame rather than a fixed frame of reference in space. For this, we build into our preprocessing function 2 toggleable sub-steps for Centering and Scaling each row.

We can exemplify the usefulness of these steps by visualizing the three different data points, all of the labels “dance” (indices 6, 10 and 89) on the same coordinate system.

In figure 1 below it is evident that the data represent 3 instances of the same gesture but recorded in a different absolute position in space. By translating all coordinates so that each datapoint does not have its euclidean center at point (0,0,0) the similarity of the three data points becomes more evident.

|  |  |
| --- | --- |
| Figure 1 : raw data | Figure 2 : centered data |

We can further exemplify the Scaling sub-step below in which the already centered data are also scaled so that each of the gesture instances occupies a cube of side 1. This is achieved by calculating, for each row, the overall width, height and depth of the 20 data points as the difference between the minimum and maximum corresponding coordinate and dividing each coordinate with the corresponding dimension.

|  |  |
| --- | --- |
| Figure 3 : centered data | Figure 4 : centered & scaled data |

* 1. **Handling missing values**

The structure of the training dataset was all explained in the assignment description and the original paper that was available for us. By investigating the training dataset, we found out that it had missing values. After looking into methods to fill missing values, we chose a couple of them:

1. Dropping/deleting all the rows that contained missing values.
2. Replacing the missing values with the mean of the column value it belongs to, after grouping the rows by gesture. I.e., first, we grouped the gestures and then we replaced the missing value with the mean for its corresponding gesture.
3. Replacing the missing values with the mean value of the whole column it belongs to. I.e., no grouping of gestures, only a straight calculation of the mean of all rows for that specific column on the dataset.
4. **The implementation / visualization of data**

For this very task of the project, we found there were too many students for too few tasks and, thus, we divided our group further into two groups. The first group was investigating Rapidminer whereas the second group was investigating python.

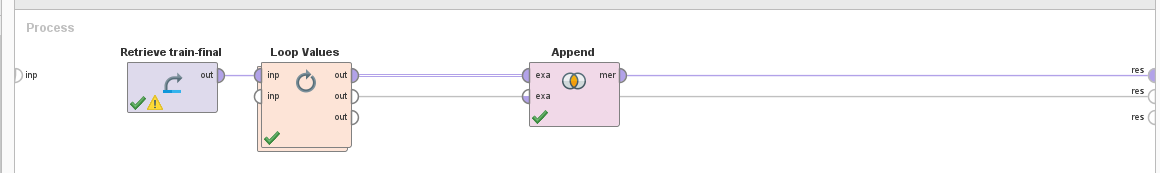
* 1. **Rapidminer**

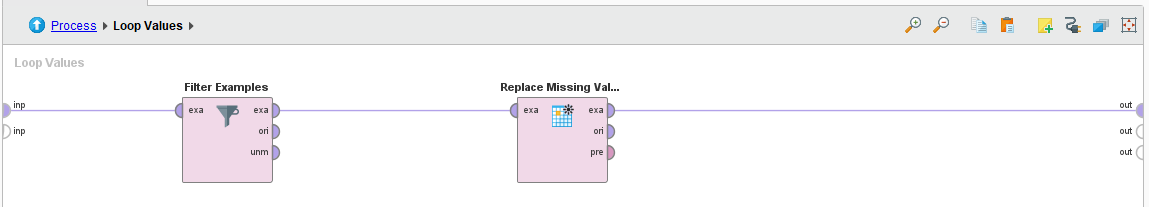
***Source code***

The processes files for the rapidminer project can be found here: <https://github.com/rasi10/D7062E-Part001/> . (*Project\_Task1.rmp*, *Project\_Task2.rmp* and *Project\_Task2\_Aggregated.rmp*) They are also attached together with this report in our compressed submitted file.

***Visualizations***

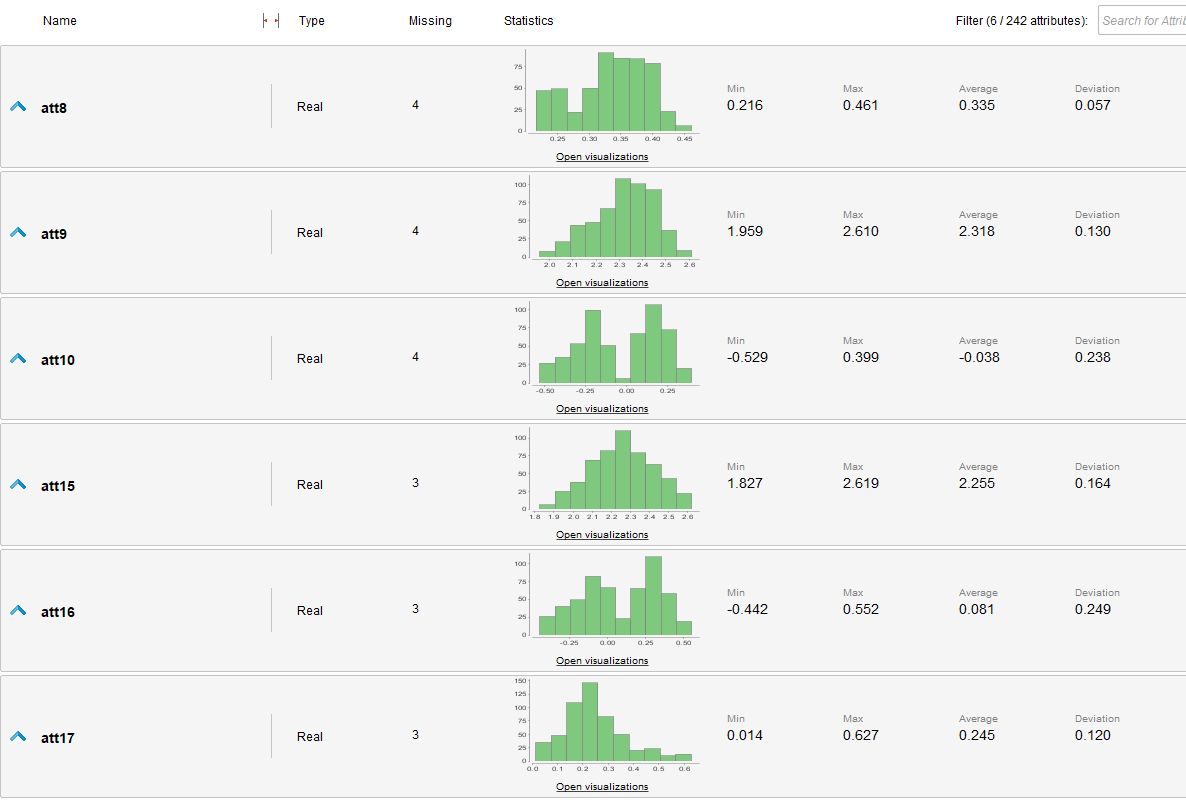
For handling the missing values, we used method *2* as described in section 2. We created through Loop Values examplesets for each gesture, and then for the missing values we took the average of each column. Afterwards we appended the examplesets.

****

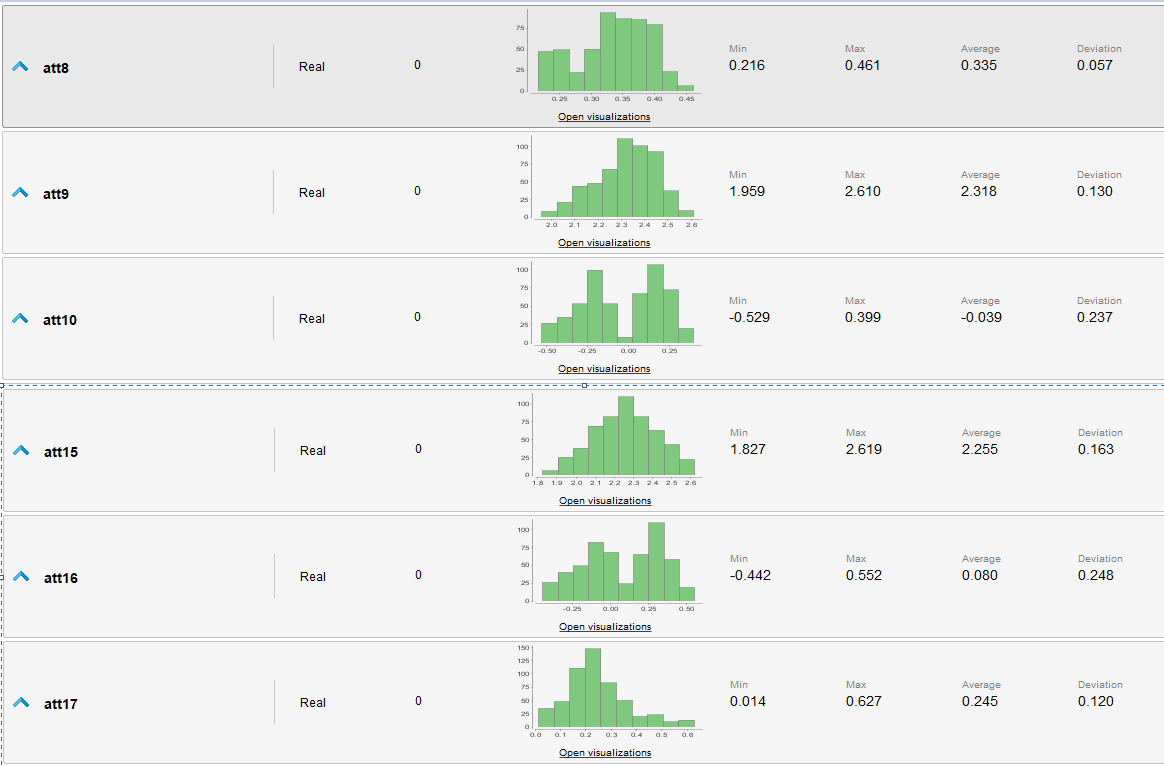
****

Train Dataset:

Before:

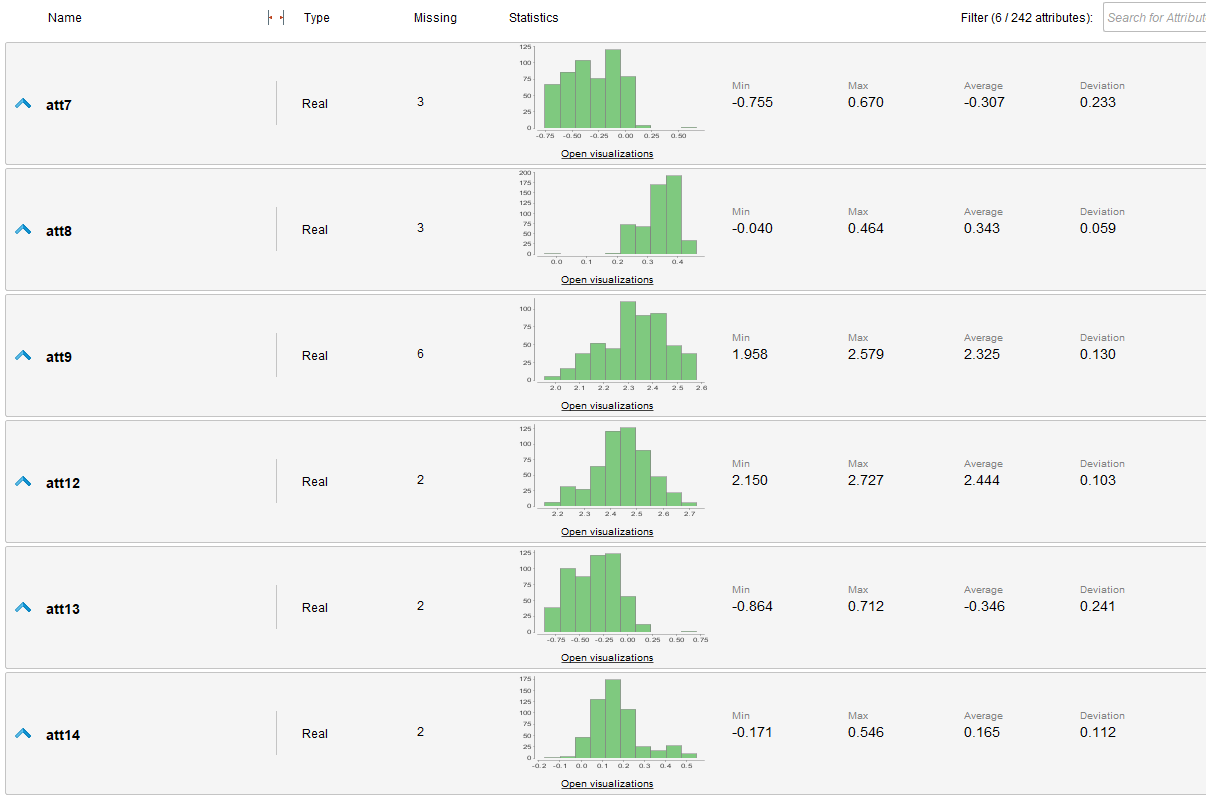


After the process has run:

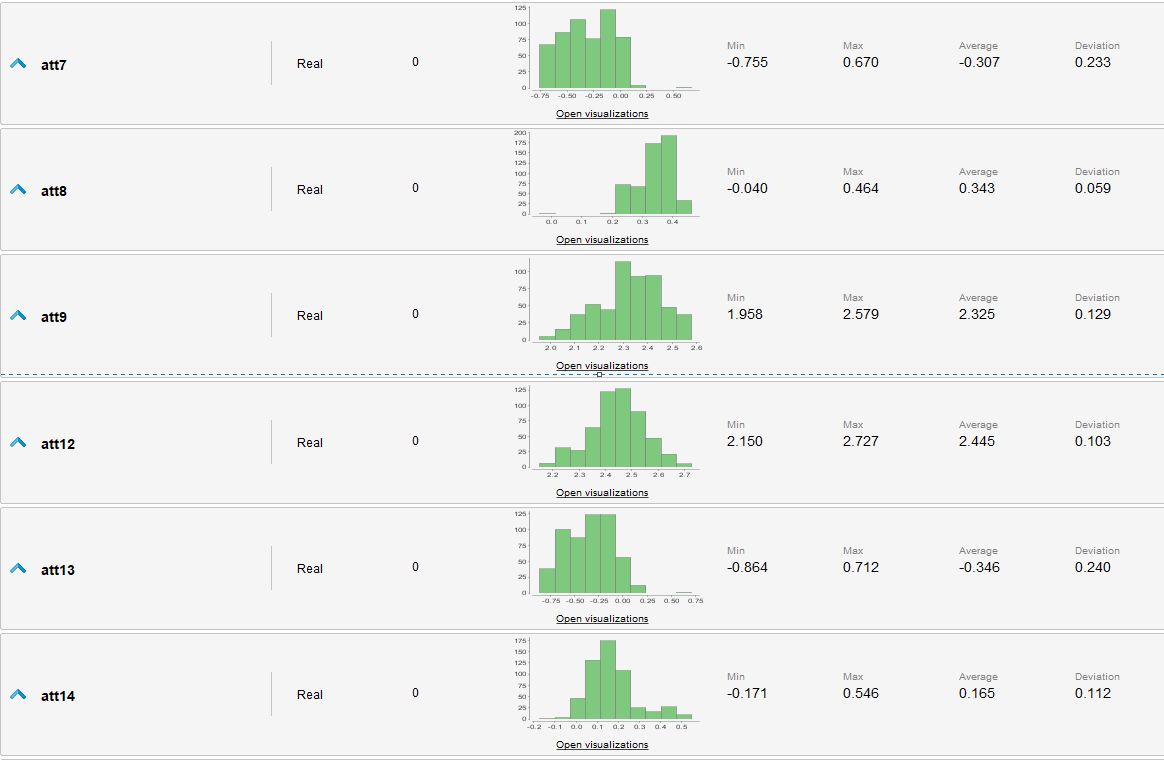


Test Dataset:

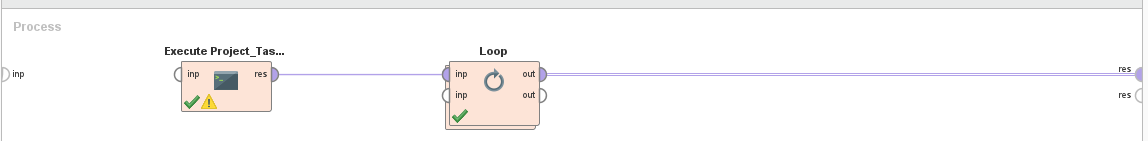
Before:

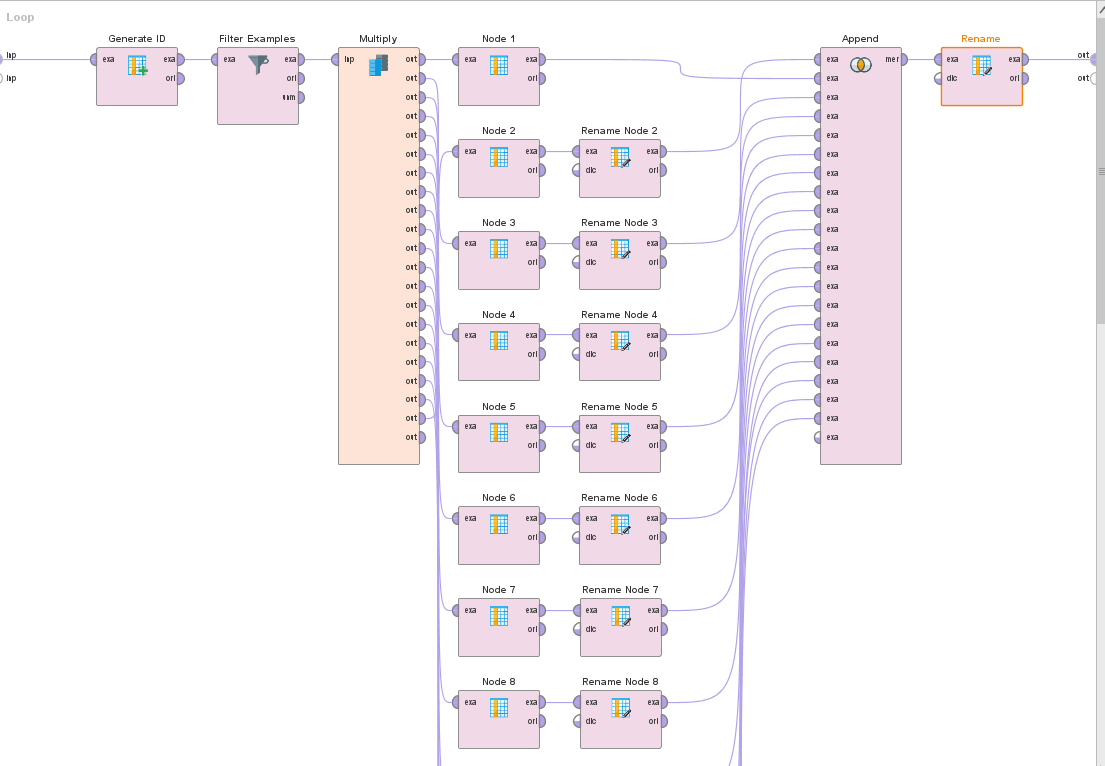


After:

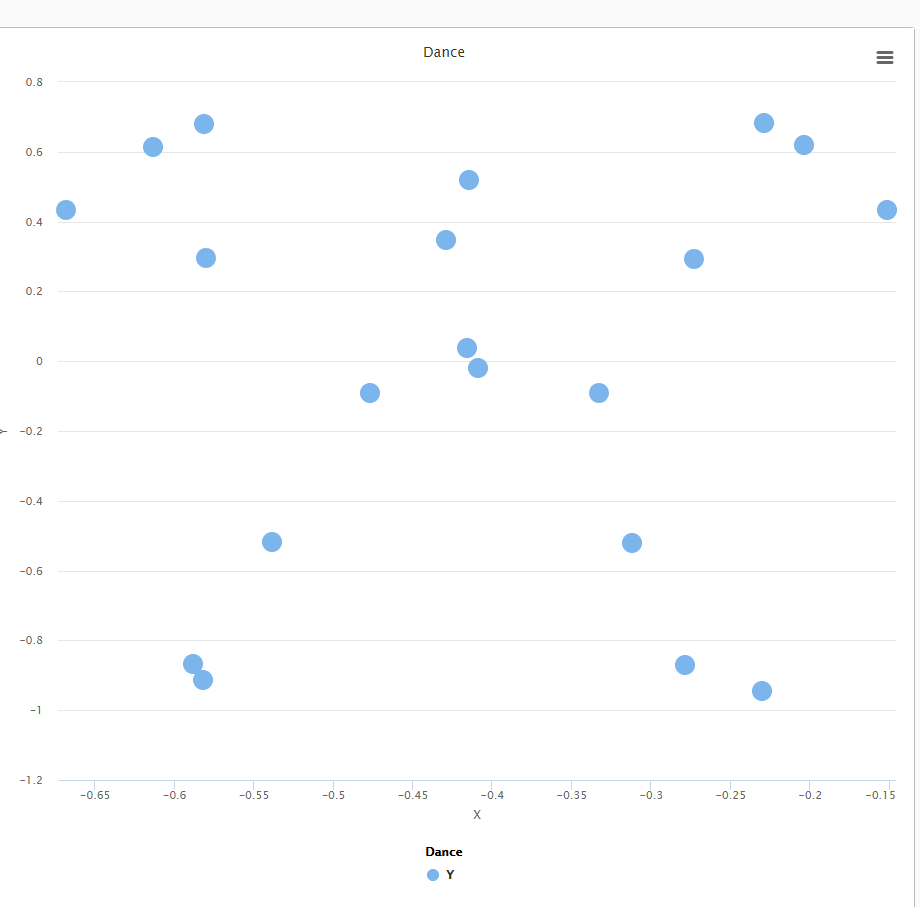


To visualize the gestures, we created examplesets for each row of the csv file. Within each exampleset, each joint coordinate was added in a different row. We did it for the mean values of each gesture.

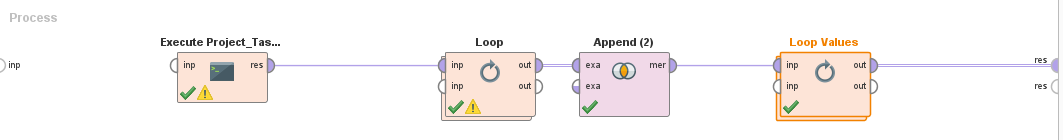


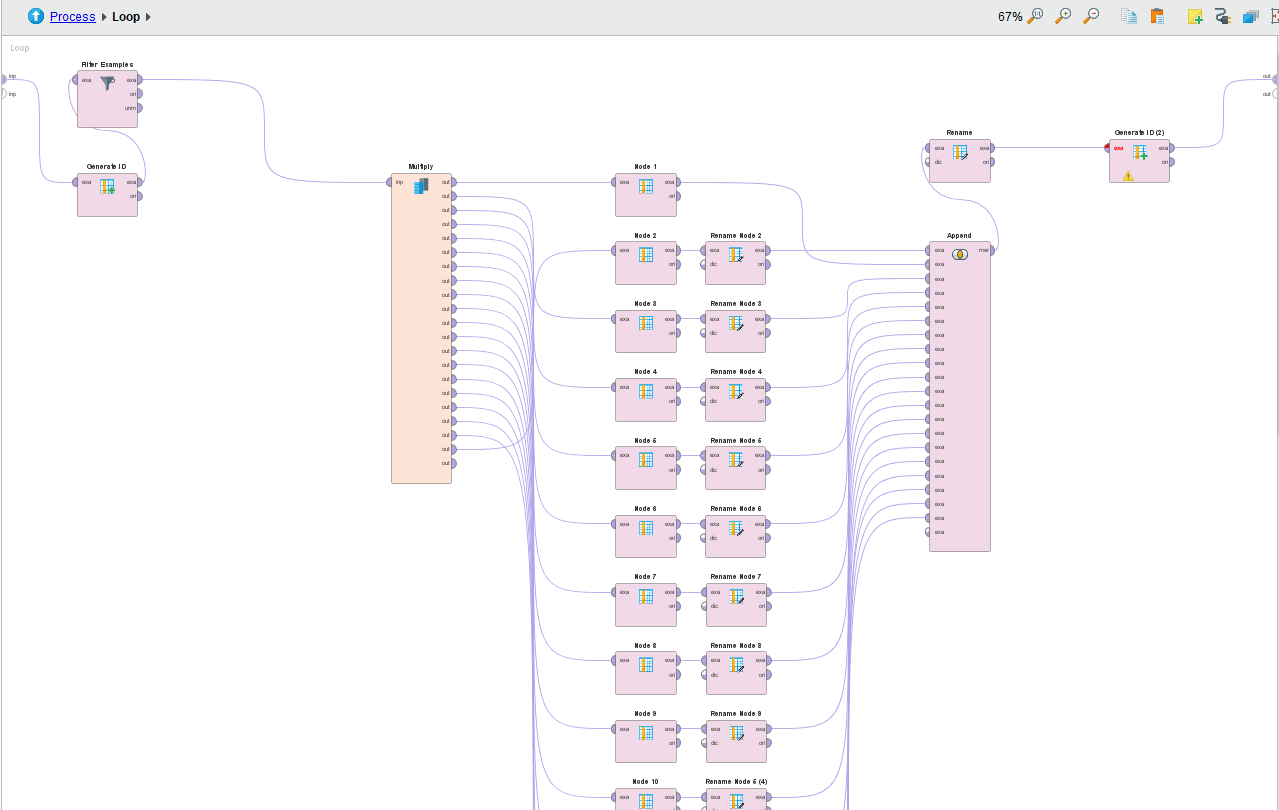


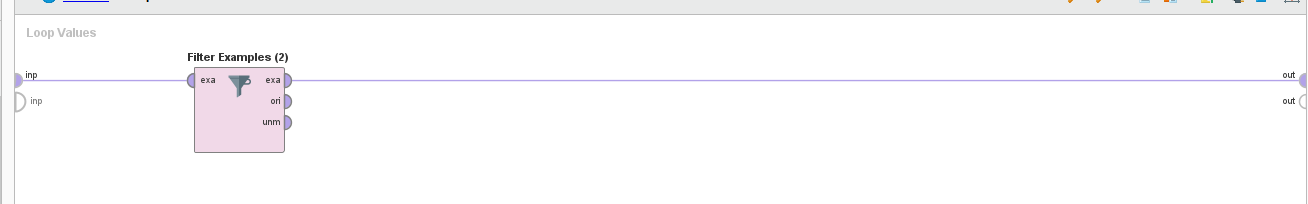
The plot of Dance gesture for a simple instance:



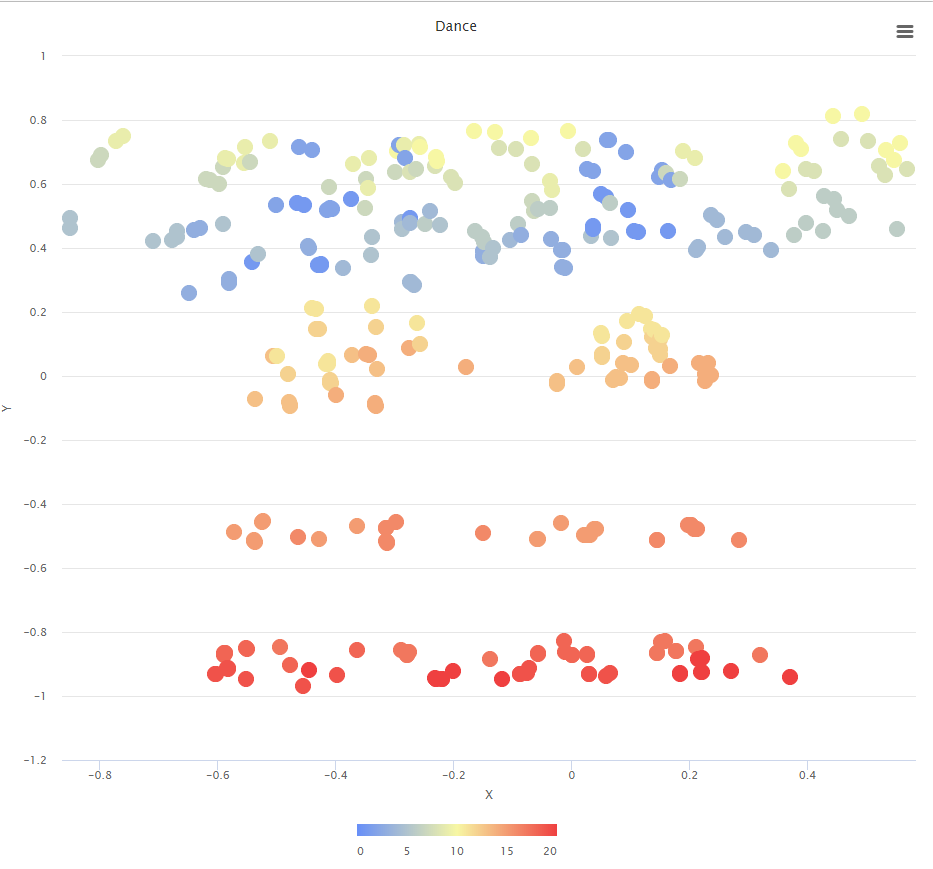
To visualize all of the instances of the same gesture in one graph we added another loop values operator. This way we are able to keep all instances of each gesture in the sample exampleset.







The dance move for all instances:



* 1. **Python**

***Source code***

The python script file for task 01 of the project can be found here: <https://github.com/rasi10/D7062E-Part001/> . (*D7062E\_project\_task01.py*) It is also attached together with this report in our compressed submitted file.

***Visualizations***

For the visualization we have created two functions:

* One (viz\_gesture)that visualizes a specified row of the frame. For example calling this function on row 34 of the normalized and preprocessed would produce the following result shown in figure 5 below.
* One (viz\_gesture\_group)that visualizes the “average gesture” of a specified gesture string label. This is achieved by grouping the data frame to be visualized by gesture label (using ‘mean’ as the grouping function) and then selecting the row of the specified gesture to be passed to the previous viz\_gesture.

|  |  |
| --- | --- |
| Figure 5  viz\_gesture(dfpp, 20, 34, links) | Figure 6  viz\_gesture\_group(dfpp,20,links,'love') |

The visualization of the links between the joining of the 20 points or the row is achieved by defining a 2-D array (links) of 20x20 size in which the connection between two joints is recorded as “1” and the non-existence of such connection as “0”. Then, after plotting the data points as a scatter plot, we can then go through this array and if the value “1” is found in position (m,n), a line is drawn between the points with the same numbers. This assumes that the correspondence between a specific body joint and the point position in the dataframe and this remains the same, or in other words, the data points were recorded using a device which could identify each body joint and record it in the correct place in the sequence of point (which by examining the data it seems to have been the case).

1. **Future work / improvements**

* Verify that the pre-processed data now works to be fed to the ML algorithms that we will investigate on Part 02 of the project. We have worked focused on Part 01 and did not try it yet, but maybe the output dataset needs some minor adjustments.
* The python code as it is at the moment has a few parameters hard-coded (eg. position of string label at column 240). Small adjustments on the code can be done so that the same code can be run on different datasets with varying numbers of points per gesture.
* The centering and scaling steps of the pre-processing function as implemented are applied only to the coordinate records. Although translation and scaling should not affect angles and their standard deviation, the data in the column recording the standard deviation of the position should be also scaled. (to be done)
* Similarly to the scaling of the standard deviation values, when grouping the data by their label to create the “average gesture” of the group, applying the same “mean” grouping function to the standard deviation may introduce errors.