Advanced Data Mining D7043E

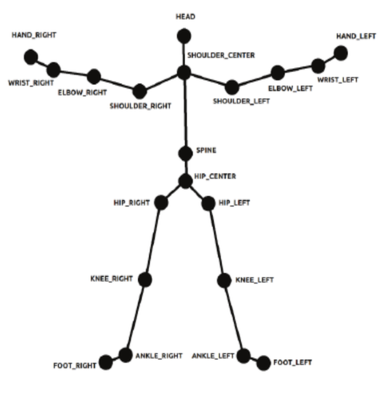
Final Project

Due Date: Exam week

For the final project of the course, each team shall implement a method for one of the two problem domains mentioned below (Human gesture analysis or U.S. Census data, not both). For the implementation of the project you have the option to use the RapidMiner Studio, the Python programming language, or any other tools that you wish to use. We can give support in the RapidMiner, Python, and Matlab.

Human gesture analysis

Your task in this project will be to train various classifiers (k-NN, Random Forest, MLP, SVM etc.), evaluate their performance on the test set and compare their results. Furthermore, you may use clustering and analysis tools to further investigate the data and report interesting findings.
The dataset used in this task, consists of samples of dynamic sign language gestures. The gestures were recorded using Microsoft Kinect. Microsoft Kinect produces the 3D skeleton of the body with 20 joints as shown in the figure below.



Thus, the data costs 60 (=20x3) dimensions. Each sign gesture is an ordered sequence of 20 joints representing their movement. The example below shows the 3 frames (60 dimension each) of a sign gesture.

-0.37722 0.676463 2.477132 -0.36725 0.493018 2.566909 -0.52942 0.35584 2.57627 -0.1943 0.431102 2.561145 -0.56758 0.087563 2.615399 -0.03334 0.579121 2.433389 -0.58943 -0.12229 2.591342 -0.11276 0.775669 2.344967 -0.57233 -0.21128 2.548121 -0.17889 0.813138 2.315454 -0.34965 0.147281 2.577553 -0.34631 0.078289 2.574845 -0.41984 -0.00526 2.57652 -0.26435 0.007698 2.571115 -0.45848 -0.46034 2.673417 -0.23598 -0.47961 2.666875 -0.46103 -0.84788 2.715237 -0.22958 -0.86515 2.683695 -0.46605 -0.89852 2.631478 -0.20799 -0.93692 2.647322

-0.37696 0.676517 2.47677 -0.36693 0.492976 2.56663 -0.52912 0.355951 2.576212 -0.19417 0.431174 2.561022 -0.56739 0.087649 2.61518 -0.03471 0.580535 2.43321 -0.58941 -0.12126 2.590974 -0.08724 0.764255 2.342603 -0.57263 -0.21045 2.548043 -0.14457 0.812098 2.321061 -0.34946 0.147283 2.57727 -0.34615 0.078288 2.574732 -0.41969 -0.00526 2.576389 -0.26428 0.007711 2.57107 -0.45848 -0.46306 2.674643 -0.23591 -0.48003 2.666955 -0.46135 -0.84594 2.716278 -0.22912 -0.866 2.683765 -0.48433 -0.92402 2.698534 -0.20776 -0.9373 2.633385

-0.37708 0.676532 2.476376 -0.36615 0.492745 2.566584 -0.52869 0.355584 2.575751 -0.1941 0.431192 2.560928 -0.5673 0.087411 2.61505 -0.03648 0.580642 2.432762 -0.5879 -0.11961 2.588753 -0.08555 0.764783 2.340891 -0.5722 -0.20986 2.547663 -0.11184 0.807713 2.319608 -0.34904 0.147191 2.577028 -0.34581 0.078215 2.574604 -0.41941 -0.00538 2.57626 -0.26406 0.007717 2.570943 -0.45842 -0.46289 2.674515 -0.23591 -0.48006

2.666903 -0.46149 -0.84789 2.716548 -0.22904 -0.86625 2.684294 -0.48464 -0.92603 2.699292 -0.20736 -0.93739 2.639276

Features Extracted

The cosine angles for each joint have been extracted costing another 60 dimensions. So, each gesture now is a sequence of positions and cosine angles of all joints.
$< positions(20 * 3 = 60)$, cosine angles $(20 * 3 = 60) > - >< 120$ features per frame $>$
i.e. each gesture is a sequence of 120 features.
Classifiers like k-NN, SVM and Random Forest require the data to be position vectors in any dimensions. They cannot handle sequential data. Therefore, we have represented the data by their MEAN (mean) and STANDARD DEVIATION (std).
i.e. Each gesture is: <mean (positions), std (positions), mean (angles), std (angles)>
i.e. $< 60, 60, 60, 60 >= 240 dimensions.$

Now, each gesture is 1x240 dimensional position vector.

Dataset CSV Files

To implement the task you are given a training and a test set to train and test your algorithms in csv file forms:

- train-final.csv: contains gestures for training
- test-final.csv: contains gestures for testing.

Each row in any CSV file contains:

- a set of 240 features representing a sign gesture
- gesture name (string)
- gesture ID (numeric)
- the candidate who performed the sign gesture (string)

Example

```
-0.332255162, 0.672471453, 2.450465228, -0.323645434, 0.486062191, 2.526748405,
-0.483694603, 0.370296328, 2.534973266, -0.172185644, 0.380791258, 2.530804253,
-0.513282554, 0.181775562, 2.440312633, -0.093926608, 0.209294813, 2.45882981,
-0.451918911, 0.264946205, 2.304346734, -0.119025985, 0.262768237, 2.303902215,
-0.425227826, 0.313142922, 2.290477456, -0.132248859, 0.309389838, 2.276056392,
-0.322625246, 0.135358034, 2.536521241, -0.3226637, 0.065258281, 2.536094316,
-0.392047651, -0.008026496, 2.535427114, -0.252877842, -0.004756406, 2.535192899,
-0.439505801, -0.467157328, 2.652029152, -0.211902319, -0.482045059, 2.64974581,
-0.449132459, -0.838430306, 2.691505038, -0.208712376, -0.851422575, 2.685142405,
-0.454246941, -0.889606135, 2.616739228, -0.189968842, -0.927571238, 2.635971658,
0.059753499, 0.002685665, 0.007119563, 0.050056279, 0.003261494, 0.004557349,
0.060112739, 0.004348804, 0.05482649, 0.061866062, 0.008459074, 0.047122457,
0.134410276, 0.021454011, 0.083326035, 0.166788961, 0.018002432, 0.114590371,
0.273614288, 0.038285436, 0.039651484, 0.32909226, 0.05334382, 0.122232151,
0.329314734, 0.038538801, 0.043688474, 0.375983125, 0.063189113, 0.127470542,
0.042062425, 0.001628359, 0.004863368, 0.042168542, 0.002269218, 0.00353412,
0.040706714, 0.004320257, 0.016910539, 0.041561043, 0.003661325, 0.009379221,
0.01716829, 0.004849589, 0.01032236, 0.01673762, 0.003227691, 0.004370211,
0.005509957, 0.002342745, 0.002214966, 0.00354215, 0.003453692, 0.00265156,
0.032057814, 0.006173736, 0.01587189, 0.001931744, 0.001818535, 0.002423289,
0.009201744, -0.023573893, 0.000558525, 0.006189902, -0.025115364, 0.029948193,
0.025388229, 0.041389905, -0.04730134, -0.00799405, 0.003017106, 0.098256794,
0.015583581, 0.116993443, -0.088289064, 0.018920917, -0.003245807, 0.056464415,
0.025235695, -0.038548577, 0.046386245, -0.030020816, -0.014784343, 0.023017322,
0.044550366, -0.022220313, -0.0144306, 0.009812549, -0.035997111, 0.014243838,
0.001013488, 0.01897849, 0.095908081, 0.014278947, 0.035131494, -0.007631017,
-0.005459036, 0.021428048, -0.033010996, -0.004590116, 0.031990516, 0.016206188,
0.047977112, -0.055423056, -0.000705523, -0.005481011, 0.017035178, 0.043681721,
0.035724834, 0.115419598, 0.022685414, 0.015954374, 0.078126562, 0.028050583,
```

0.091354824, 0.078863291, -0.029918646, -0.020105747, -0.041432565, 0.118327381, 0.928138377, 0.209511115, 0.201487581, 0.919161701, 0.214131248, 0.216765545, 0.706306191, 0.30047568, 0.442213354, 0.745180986, 0.320606856, 0.393360993, 0.689327034, 0.362353623, 0.418813135, 0.718505671, 0.257705861, 0.445917765, 0.779209717, 0.414830023, 0.317347734, 0.758191972, 0.333697935, 0.386498866, 0.821571379, 0.344500951, 0.308696432, 0.81343024, 0.278147466, 0.35094431, 0.911671188, 0.224283312, 0.208247102, 0.929694494, 0.283866356, 0.142641195, 0.79661676, 0.305589221, 0.358115783, 0.898272085, 0.280851919, 0.223404432, 0.658695664, 0.475447472, 0.401127032, 0.721980458, 0.481920461, 0.33873804, 0.394347689, 0.757080255, 0.348332688, 0.426959439, 0.604953419, 0.464156666, 0.751404229, 0.467515951, 0.306133099, 0.521694496, 0.54609408, 0.439480238, wind, 28

In the above example you can see the following:

- The first 240 numbers (red) are 240 features representing a sign gesture.
- wind, 28 (blue) represent the string and numeric label (basically both are same thing) of this sign gesture.

The 30 sign gestures presented in this dataset are the following:

afternoon	baby	big	born	bye	calendar
child	cloud	come	daily	dance	dark
day	enjoy	go	hello	home	love
my	name	no	rain	sorry	strong
study	thankyou	welcome	wind	yes	you

Some of these gestures are performed by single hand only and other gestures require both hands.

The task here would be to:

Task 1:

Do the CRISP-DM analysis of the project
Handle missing values.
Pre-process the attribute values so that they are in the appropriate form to be given as input for an algorithm.
Visualize the data with different techniques.

Task 2:

Using the training set, train several classifiers (Decision Tree, Random Forest, k-NN, MLP, SVM).
Evaluate their performance on the test set
For the two best classifiers,
reduce the dimensions to 90% using PCA and evaluate their performance on the test set again.
Reduce again to 80% and evaluate their performance on the test set again.