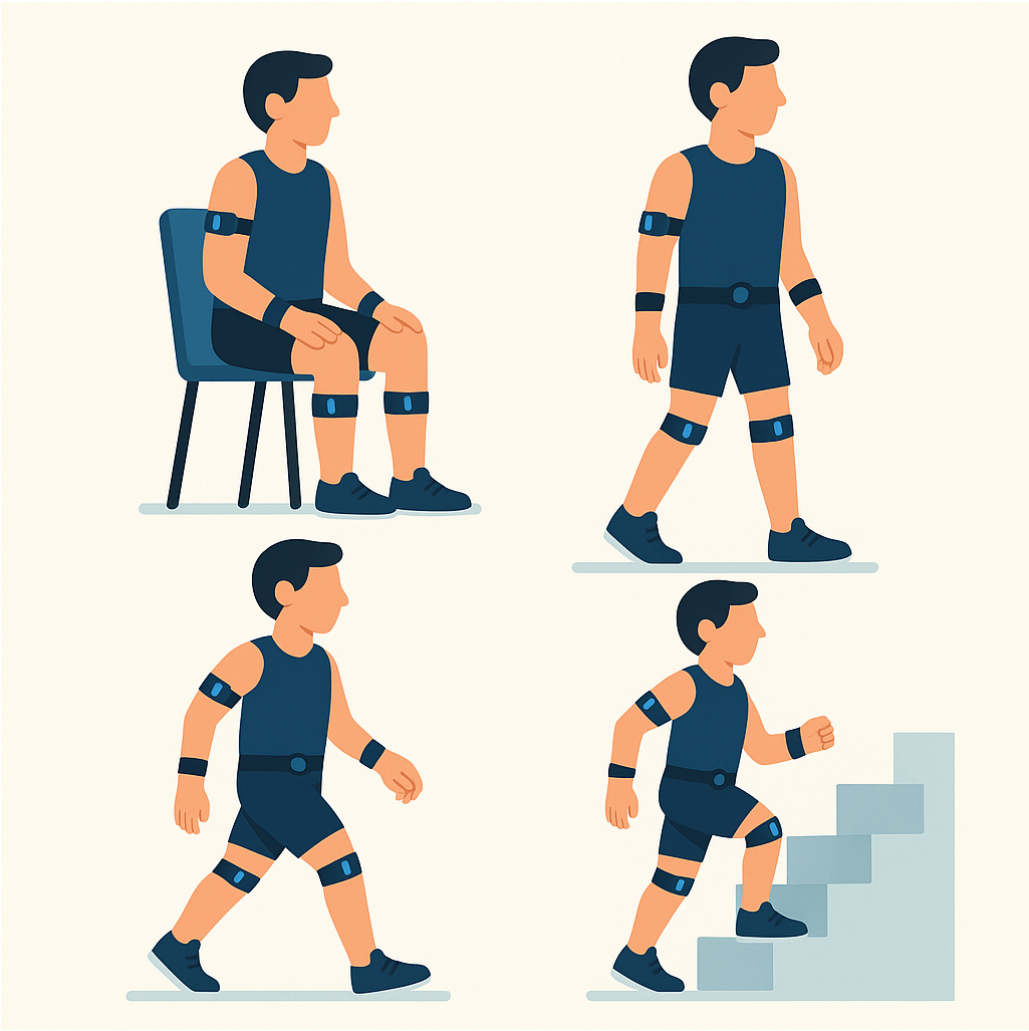
Introduction to Machine Learning - MIA 2025/2026
Practical assignment

Detection of Human Activities



Introduction

Objective: To exercise core concepts of a data analysis pipeline, going through the stages of data preparation, cleaning, extraction of descriptive features, their selection/reduction, and machine learning.

Context: The problem proposed in this practical assignment is a typical classification task, namely the recognition of human physical activities. This is a context of growing importance in multiple situations, encompassing, for example, medical, recreational, and wellness applications. Regardless of the specific problem and its potential applications, this work will allow you to practise and internalise key concepts for any data analysis pipeline.

Figure 1: Device locations

We will use the FORTH-TRACE benchmark dataset[1]. This dataset was collected using 5 sensors (see Figure 1), including sensors for acceleration, angular velocity, and magnetic field variation, placed on both the upper and lower parts of the body. It consists of data acquired from 15 participants using a protocol that involved 16 distinct activities listed in Table 1. The original data can be downloaded using this link, which contains the following files (part → participant, dev → device):

- partX/partXdev1.csv
- partX/partXdev2.csv
- partX/partXdev3.csv
- partX/partXdev4.csv
- partX/partXdev5.csv

where X corresponds to the participant ID and 1–5 to the device ID (see Table 2).

[1] Katerina Karagiannaki,Athanasia Panousopoulou,Panagiotis Tsakalides, A Benchmark Study on Feature Selection for Human Activity Recognition, UBICOMP/ISWC '16, https://dl.acm.org/doi/pdf/10.1145/2968219.2971421

Each CSV file follows the format below:

- Column 1: Device ID
- Column 2: accelerometer x
- Column 3: accelerometer y
- Column 4: accelerometer z
- Column 5: gyroscope x
- Column 6: gyroscope y
- Column 7: gyroscope z
- Column 8: magnetometer x
- Column 9: magnetometer y
- Column 10: magnetometer z
- Column 11: Timestamp
- Column 12: Activity Label

Table 1: Ativities

Label	Activity
1	Stand
2	Sit
3	Sit and Talk
4	Walk
5	Walk and Talk
6	Climb Stair (up/down)
7	Climb Stair (up/down) and talk
8	Stand-> Sit
9	Sit-> Stand
10	Stand-> Sit and talk
11	Sit->Stand and talk
12	Stand-> walk
13	Walk-> stand
14	Stand -> climb stairs (up/down), stand -> climb stairs (up/down) and talk
15	Climb stairs (up/down) -> walk
16	Climb stairs (up/down) and talk -> walk and talk

Table 2: Device identifier

ID	Dispositivo
1	Left wrist
2	Right wrist
3	Chest
4	Right thigh
5	Left shin

ASSIGNMENT INSTRUCTIONS

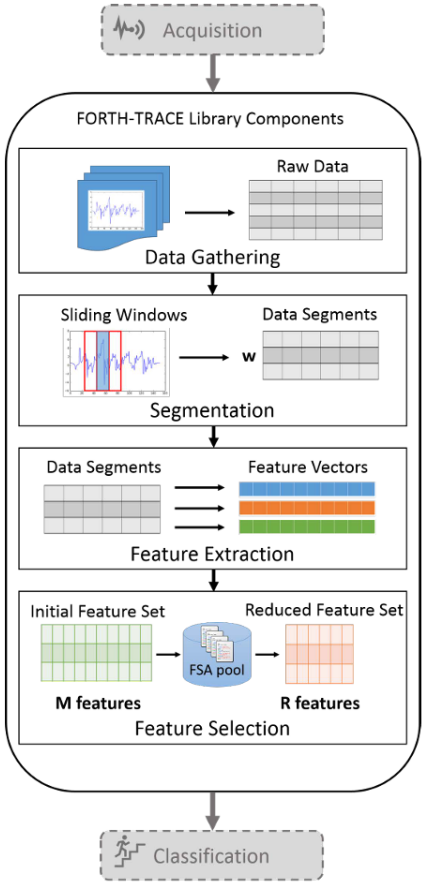


Figure 2 - Expected pipeline

Introductory note: *For simplicity, consider only activities 1 to 4 and device 1. Feel free to expand to more activities and more devices as you see fit.*

1. Create an IPython notebook with your name and number. In this notebook, you should place your code and the analysis, discussions, and answers asked in the assignment. Alternatively, you can code in separate python files and submit the code and a PDF file with the report (analysis, discussions, and answers to the assignment).

2. **Download the data** using the link provided above and implement a routine that loads the data for a given individual and returns it as a NumPy array.

   You may use, for example, the CSV library (https://docs.python.org/3/library/csv.html).

3. **Outlier analysis and handling:** the goal is to identify and treat outliers using different **univariate** and **multivariate** approaches. For this purpose we will use the **magnitude (modules)** of the acceleration, gyroscope, and magnetometer vectors. *Let*

$$\vec{t} = \left(t_x, t_y, t_z\right)$$

be the acceleration, gyroscope, or magnetic vector. The respective module is determined by:

$$\left\|\vec{t}\right\| = \sqrt{t_x^2 + t_y^2 + t_z^2}$$

3.1. Develop a routine that simultaneously displays the boxplot for each activity (on the horizontal axis) covering all subjects, for each of the following transformed variables (the magnitudes of the acceleration, gyroscope, and magnetometer vectors), and separated by device (if you are considering more than one device). The use of the matplotlib library is suggested, in particular, the `matplotlib.pyplot.boxplot` class.

3.2. Analyse and comment on the density of outliers present in the transformed dataset, i.e., in the magnitudes of the acceleration, gyroscope, and magnetometer vectors for each activity. Apply the IQR method (Tukey's definition, the same used by `matplotlib.pyplot.boxplot`). Note that the density is determined by using

$$d = \frac{n_o}{n_r} \times 100$$

where n_o is the number of points classified as outliers and n_r is the total number of points.

3.3. Write a routine that receives an array of samples of a variable and identifies outliers using the Z-Score test for a variable k (input parameter).

3.4. Using the implemented Z-score, flag all samples considered outliers in the magnitudes of the acceleration, gyroscope, and magnetometer vectors. Present plots separated by activity (analogous to question 3.1) in which these points appear in red, while the remaining points appear in blue. Use $k = 3$, 3.5, and 4.

3.5. Compare and discuss the results obtained in 3.1 and 3.4, as well as the respective outlier densities obtained with the two methods.

3.6. Develop a routine that implements the k-means algorithm for n(input value) clusters.

3.7. Determine the outliers in the transformed dataset using k-means. To do this, you may use the space of the sensor magnitudes. Try different numbers of clusters and compare with the results obtained in 3.4. Illustrate the results graphically using 3D plots (see, for example, https://medium.com/data-science/an-easy-introduction-to-3d-plotting-with-matplotlib-801561999725).

3.7.1. **Bonus:** You may carry out an analogous study using the DBSCAN algorithm (the use of the `sklearn` library is suggested).

4. **Feature engineering:** the goal is to compress the problem space by extracting discriminative features that enable effective solutions to the classification problem.

4.1. Using the variables applied in item 3.1, determine the statistical significance of their mean values across the different activities. Note that you may assess distribution normality using, for example, the Kolmogorov–Smirnov test (see SciPy documentation). To review the choice of statistical tests, the reference is suggested[2]. Comment.

4.2. Develop the routines needed to extract the temporal and spectral feature set suggested in the article[3]. To do this, you should:

- Read the article and identify the set of temporal and spectral features specified by the authors.
- Consider only 5-second sliding windows with 50% overlap to segment the data for feature extraction. If a segment spans more than one activity, discard that segment. See in particular Figure 2 and Table 1 of article[1] (also in Figure 2).
- For each feature, implement a routine for its extraction.
- Using the routines developed in the previous item, write the code required to extract the feature vector at each instant.

Note: You may use the NumPy and SciPy libraries. Any other library must be identified.

[2] Jean-Baptist du Prel, Bernd Röhrig, Gerhard Hommel, Maria Blettner, Choosing Statistical Tests, Deutsches Arzteblatt, v107(19), 2010. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2881615/

[3] Mi Zhang, Alexander A Sawchuk, A. Sawchuk, A feature selection-based framework for human activity recognition using wearable multimodal sensors, BodyNets '11: Proceedings of the 6th International Conference on Body Area Networks, November 2011 pp 92–98, https://pdfs.semanticscholar.org/8522/ce2bfce1ab65b133e411350478183e79fae7.pdf

4.3. Develop the code needed to implement PCA on a feature set; you may use existing implementations.

4.4. Determine the importance of each principal component in explaining the variability of the feature space. Note that you should normalise the features using the z-score. How many dimensions should be used to explain 75% of the feature set?

4.4.1. Indicate how you could obtain the features corresponding to this compression and provide an example for an instant of your choice.

4.4.2. Indicate the advantages and limitations of this approach.

4.5. Develop the code needed to implement Fisher Score and ReliefF. You may use existing implementations.

4.6. Identify the 10 best features according to Fisher Score and ReliefF and compare the results.

4.6.1. Indicate how you could obtain the features corresponding to this selection and provide an example for an instant of your choice.

4.6.2. Indicate the advantages and limitations of this approach.

5. **Data Augmentation:** the objective is to generate synthetic examples to add variety to the training set.

5.1. Analyse the balance between the number of exemples available for the set of activities you are considering. Is the dataset balanced?

5.2. Create a function that receives the dataset and implements the SMOTE method to generate K new samples for a given activity A.

5.3. Use the function before to generate and visualize 3 new samples of activity 4 of participant 3. Use only the samples of that participant in the process. For

the visualization, use a 2D scatter plot considering only two features, make the color of the points diferente by activity and highlight the synthetic ones.

6. **Data splitting:** Split the dataset into train, validation and test sets.

    6.1. Use a TVT split of 60-20-20% split, within each subject (data of a single subject is present in all sets)

    6.2. Use the same split but at the subject level: put 9 subjects for training, 3 for validation, and 3 for testing.

    6.3. Create a function to receive the true labels and predicted labels and return a set of classification metrics: confusion matrix, accuracy, f1 score, precision, recall, etc.

7. **Instance-based classifier: k-Nearest Neighbors.**

    7.1. Implement a k-Nearest Neighbors classifier. (I expect you to implement your own version, but then you can use the sci-kit learn version if you need better performance. See information in: sklearn_kNN).

8. **Evaluation**:

    8.1. The following steps should be repeated for the two scenarios of question 6:

    a) Using all features

    b) Using the PCA-transformed features

    c) Using the top 10 features selected by ReliefF

    8.2. **Hyperparameter tuning:** Select the best value of k using only the train and validation data. Then, with the best k, retrain with the dataset (train + validation) and evaluate on the test set.

    8.3. **Report and analyse your results.**

9. **Go further:** Discuss what can be done more to improve your classification system. If you have the time, implemente and evaluate some of those ideas. The sky is the limit!