ELEC 292: Introduction to Data Science
Project Instructions

Goal: The goal of the project is to build a desktop app that can distinguish between 'walking' and 'jumping' with reasonable accuracy, using the data collected from the accelerometers of a smartphone.

Description: The project involves building a small and simple desktop application that accepts accelerometer data (*x*, *y*, and *z* axes) in CSV format, and writes the outputs into a separate CSV file. The output CSV file contains the labels ('walking' or 'jumping') for the corresponding input data. For classification purposes, the system will use a simple classifier, i.e., logistic regression.

In order to accomplish the goal of the final project and complete the report, the following 7 steps are required:

1. Data collection
2. Data storage
3. Visualization
4. Pre-processing
5. Feature extraction & normalization
6. Training and testing a classifier
7. Creating a simple desktop application with a simple UI that shows the output

Step 1. Data collection

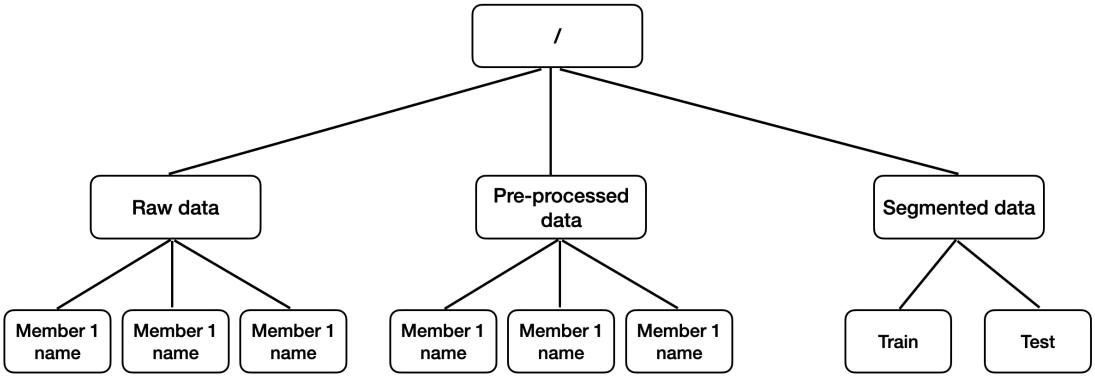
In this step, you need to collect data using your smart phones while 'walking' and 'jumping'. There are a number of different apps you can use to collect accelerometer data from your smartphone. As an example, you may use an app called Phyphox, which works on both iOS and Android, and allows you to output the recorded signals as a CSV file. Other apps would also be acceptable.

Data collection protocol: Recall that when collecting data, the diversity of the dataset will allow your system to work better when deployed. (**a**) Therefore, to maximize diversity, each team member must participate in the data collection process to create a total of 3 subsets (1 per member). (**b**) To further maximize diversity in your dataset, for each subset, several trials should be performed where the phone is placed in different positions. For example, you can place the phone in your front pocket, back pocket, pocket of a jacket, carry it in your hand, etc. The phone

itself can also be oriented differently. (**c**) The duration of data collection by each member must exceed 5 minutes. Please note that it is important that you collect a roughly *balanced* dataset. In other words, the amount of time dedicated to each user, each trial, and each action ('walking' vs. 'jumping'), should be roughly the same.

Step 2. Data storage

After transferring your dataset (all the subsets) to a computer and labeling them based on the actions, store the dataset in an HDF5 file. This HDF5 file must be organized as follows:



It is always a good idea to keep the data as originally collected, which is why we have the structure that we see on the left side of this image. Next, we should also store the pre-processed data as shown in the middle branch (see Step 4 of the project below for details). This will provide us with a backup of the finalized pre-processed data for model training, eliminating the need to redo this step. Lastly, in order to create a simple machine learning system, you need to create separate training and test splits. To do so, divide each signal into 5-second windows, shuffle the segmented data, and use 90% for training and 10% for testing. This new dataset must also be stored in the HDF5 file as shown on the left side.

Step 3. Visualization

Data visualize is a critical step in the field of data science and will allow you to find issues in the data early on and also become familiar with the data that you will be working with. So, in this step, you will need to visualize a few samples from your dataset (all three axes) and from both classes ('walking' and 'jumping'). A simple acceleration vs. time plot would be a good start. But also think about additional creative ways of showing the data with the goal of representing your dataset. Provide some visualization for the meta-data for your dataset and sensors too. Don't forget to use good visualization principles.

Step 4. Pre-processing

Remember, *garbage in, garbage out*! Almost any dataset, no matter how careful you were during collection, will inevitably contain some noise. First, ensure to fill in any missing data. Second, the data will likely contain noise, which should be reduced, e.g., using a moving average filter – you

can find the right filter parameters with trial and error. You should visualize the data after pre-processing and compare the signals with the original raw data.

Step 5. Feature extraction & normalization

From each time window (the 5-second segments that you created and stored in the HDF5 file), extract a minimum of 10 different features. These features could be maximum, minimum, range, mean, median, variance, skewness, etc. Additional features may be explored as well. After feature extraction has been performed, you will be required to apply a normalization technique for preventing features with larger scales from disproportionately influencing the results. Common normalization techniques are min-max scaling, z-score standardization, etc.

Step 6. Training and testing a classifier

Using the features from the preprocessed training set, train a *logistic regression* model to classify the data into 'walking' and 'jumping' classes. Once training is complete, apply it on the test set and record the accuracy. You should also monitor and record the training curves during the training process and present them in your report. Note that during the training phase, your test set must not leak into the training set (no overlap between the segments used for training and testing). In terms of accuracy, note that for a binary machine learning model, 50% is considered chance-level, so you should aim to achieve higher performance, e.g., 80% or higher (the higher the better).

Step 7. Deploying the trained classifier in a desktop app

The last step is to deploy your final model in a desktop app. For building a simple graphical user interface in Python, you can use Tkinter or PyQt5 libraries. As mentioned, this app must accept an input file in CSV format and generate a CSV file as the output, which includes the labels (walking or jumping) for each window in the input file. Run a demo for your built app in which you input a CSV file and the app generates a plot which represents the outputs. Once deployed, how did you test the system to ensure it works as intended?

Deliverables

Report: Write a report for the project. The project should contain the following.

- A title page containing the following: Course: ELEC 292
  Project Report
  Group Number: _____
  Names, Student Numbers, and Email Addresses: _____ Date: _____
- After the title page, the rest of the document must be in 12-point Times New Roman font, single spaced, 1 inch margins, with the text justified, and with page numbers in the bottom center of each page.
- Every student must submit a **separate copy** that is identical to their teammates. This is done as a signoff, indicating that each member has participated and agrees with the content.
- As a rule of thumb, the report should be between 15 to 20 pages including references and figures.
- Note that where you refer to online sources (articles, websites, etc.) the references must be mentioned in the reference section of the document (in the end of the document), and the references should be *referred to in the text*. Here is a brief description of how proper citation and references should be used: https://labwrite.ncsu.edu/res/res-citsandrefs.html. In this report, you must use the IEEE format for references.
- Note that in your report, you must not "copy-paste" text from other resources, even if you provide proper citations. Text should be read, understood, and *paraphrased*, with proper citation of the original reference.
- Proper editing (grammar, typos, etc.) is expected for the reports. Using tools like Grammarly or other writing tools are allowed/encouraged.
- The report should clearly describe each step and provide the requested material.
- The report must have the following **sections:**

  1. Data Collection: How did you collect the data, label them, transfer them to a PC, and what challenges did you deal with during data collection. How did you overcome them? Mention all the hardware and software used.

  2. Data Storage: Provide a full description of the way you stored the collected data.

  3. Visualization: Provide all the plots that you created for visualization purposes, and provide appropriate descriptions for each of them. What did you learn? Knowing what you learn from the plots, if you were to re-do your data collection, how would you do things differently?

  4. Preprocessing: Clearly describe the measures you took for preprocessing, and how it impacted the data (you should use a few plots here too). Why did

you choose the parameters that you did (e.g., size of the moving average filter)?

5. Feature Extraction & normalization: What features did you extract and why? References may be useful here. Explain the process of feature extraction and normalization, then justify your choices.

6. Training and testing the classifier: Provide a description of the way you trained the logistic regression model. This section must include the learning curves and accuracies on the training and test sets. What parameters did you use here? Justify your answers.

7. Model deployment: This section should include the details of how you deployed the trained model into a desktop app. Provide screenshots of the GUI you created along with its description, and justify your design choices.

- At the end of the report, a **Participation Report** must be added. Please note that the project should be done together and collaboratively. It is not acceptable for one person to do the technical work and another to simply write the report. Having said this, a reasonable division of work is allowed for type up or other simple tasks. At the end of the report, provide a table that clearly shows which members have been present in each phase of the project and contributed to each question. Please note that should someone not pull roughly 1/3 of the weight of the project, they will lose points.

Demo video: Record your screen while running a demo with the created app. The video should feature all team members and show short snippets of your data collection process, as well as the app in action. The video should also explain your project in a few sentences. It should be between 1 to 3 minutes. Include clear audio explanations, showcasing app functionality, and showing data collection setups.

Bonus: This part of the final project is not mandatory and serves as a bonus deliverable, which can gain up to *5 bonus points (out of 100) on your project!* The app that you created, works offline. In other words, the app is not able to classify activities from your smart phone in ***real-time***. For the bonus component of the project, our goal is to build a desktop app which can read the accelerometer data from your smart phone in real-time and classify it immediately. Your smart phone would need to send the accelerometer data to the app in real-time, and the app would show the class of action (e.g., 'walking') in real-time.

Hint: For reading the accelerometer data online, you may use the 'Enable remote access' option of the Phyphox app. By doing so, you will have access to the accelerometer data in a web page. You can then use Beautiful Soup and Selenium libraries to read the data. Alternative ways include using Bluetooth to send the data to the PC in real-time.

Note that in order to make the system function in real-time, the pre-processing and classifier implementation may need to be updated/extended to allow such a functionality.

Deliverables for the <u>bonus</u> component:

1. The report should be extended by 2-3 pages. These additional materials should include:
    - All the details of how the data was transferred to the PC in real-time
    - A description of any changes made to the desktop app and its GUI
    - A description of any changes made to the pre-processing and trained classifier in order to allow the system to function in real-time
2. The video should clearly show that a person is carrying a phone in their pocket and the desktop app is classifying their actions in real-time

General note: If you attempted anything but could not get it to work, whether for the main part of the project or the bonus component, you should mention what you did, what is your hypothesis for it not working, and how things should likely change to make it work, to receive some partial marks.

Submission instructions: The following items will need to be submitted in OnQ:

1. Your project report in **PDF** format
2. Your saved HDF5 file
3. The demo video
4. Your clean and executable Python code (for the main part and bonus if attempted), which contains the code for ranging from (1) visualization, (2) pre-processing, (3) feature extraction, (4) training and running the model

***** ***Note*** that each member of the team must submit an individual copy of the report, identical to their teammates.

Grading: A 5-point scale will be used for grading different aspects of the project. This 5-point scale will be as follows:

Quality	**Grade**	**Definition**
Excellent	4/4	Explanations are complete, clear, and easy to understand
Good	3/4	Explanations are lacking a bit of clarity or completeness, but are generally in good shape
Average	2/4	Several aspects are missing or incorrect. There is quite a bit of room for improvement
Poor	1/4	Most aspects are missing or incorrect
Not done	0/4	The section is not performed at all

The following grading scheme will be used:

Task	Description	Weight
1. Data collection	Completeness/thoroughness, balance, diversity, good data collection principles	3
2. Data storing	Proper data storage in the specified format, reasonable train-test splits, no data leakage	2
3. Visualization	Several samples visualized, each class represented, meta-data visualized, additional creative plots, good visualization principles	2
4. Pre-processing	Removal/reduction of noise, normalization, further visualization of data after pre-processing	2
5. Feature extraction	Proper identification and extraction of a minimum of 10 different features	2
6. Training the mode	Proper construction and training of the model, reasonable results	2
7. Desktop app	Nice/clean UI, functionality, testing of the system	2
8. Demo video	Proper description and demo of the work, participation from everyone	3
9. Report	Proper structure, detailed description, high quality images, writing and editing quality, references and citations, cover page, division of work statement, providing everything described under Step 9	7

The final grade for the project will be calculated out of 100. Up to 5 points for the bonus component will then be added to this grade (if available). The final score will be multiplied by 0.3 to obtain your project grade out of 30.