ELEC 292
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.

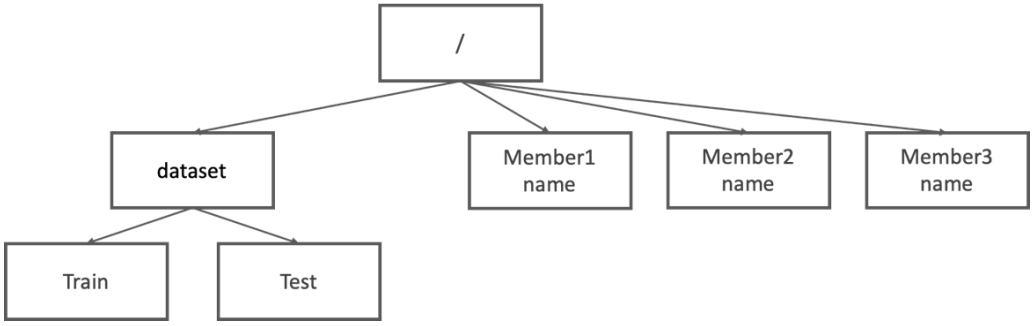
Description:
The project involves building a small and simple desktop application that accepts accelerometer data (*x*, *y*, and *z* axes) in CSV format, and writes the outputs into a separate CSV file. The output CSV file contains the labels ('walking' or 'jumping') for the corresponding input data. 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 storing
3. Visualization
4. Pre-processing
5. Feature extraction & Normalization
6. Training the model
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 phone 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, the phone should be placed in <u>different positions</u>. For example, you can place the phone in your front pocket, back pocket, pocket of a jacket, carry it in your hand, etc. (**c**) The duration of data collection by each member must <u>exceed 5 minutes</u>. Please note that it is important that you collect a roughly *balanced* dataset. In other words, the amount of time dedicated to each user, to each action ('walking' vs. 'jumping'), to each phone position, and others, should be roughly the same.

Step 2. Data storing
After transferring your dataset (all the subsets) to a computer and labeling them, 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 right side of this image. But in order to create a simple AI 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 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, the data will likely contain noise, which may be reduced by a moving average filter. Second, after feature extraction (next step), try to detect and remove the outliers in your collected data. Please note that if by removing outliers, the data becomes too imbalanced, remedy this. Finally, normalize the data so that it becomes suitable for logistic regression.

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. Creating 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. 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).

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?

Step 8. 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.

Step 9. Report
Write a report for the project. The project should contain:

- A title page containing the following:
  Course: ELEC292
  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, 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. It will also make grading and tracking easier.

- 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 though you are citing them. Text should be read, understood, and *paraphrased*, with proper citation of the original reference.

- Proper editing (grammar, typos, etc.) is expected for the reports.

- 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 it, transfer it to a PC, and what challenges did you deal with from during the data collection step. How did you overcome them? Mention all the hardware and software used.

- **2.** Data Storing: 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 may use a few plots here too). Why did you choose the parameters that you did (e.g., size of moving average)?

- **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.

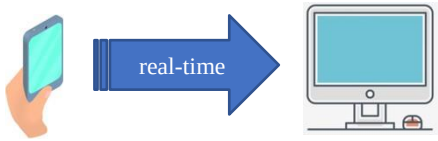
- **6.** Training the classifier: Provide a description of the way you trained the logistic regression model. This section must include the learning curves and your accuracy on the training and test sets. What parameters did you use here? Justify your answers.

- **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 for and contributed to each question. Please note that should someone not pull roughly 1/3 of the weight of the project, they may lose points.

Submission:
The following items will need to be submitted in OnQ:
- 1. Your project report in **PDF** format
- 2. Your saved HDF5 file in the mentioned format
- 3. The video as described earlier
- 4. Your clean and executable Python code, which contains the code for ranging from (1) visualization, (2) pre-processing, (3) feature extraction, (4) training and running the model

Bonus:
Part 1: This part of the final project is not mandatory and serves as a bonus deliverable, which can gain up to *10 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 a real-time and classify it immediately. As shown in the image above, 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. Then, you may 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.

Part 2: In this step, you will have to implement the SVM and Random Forest classifier from scratch without the use of any existing libraries for the model, such as, scikit-learn. You are free to use libraries for basic operations, such as, NumPy. Record the accuracy of your implemented models that you built from scratch on test set. Then, compare the performance of your own implemented models (without using libraries) with the models implemented using pre-existing libraries. This direct comparison will highlight the efficacy of your custom-built models versus standardized library models. Finally, provide overall insights on the comparison of models and explain the outcomes.

Deliverables for the bonus component:

1. The report should be extended by 3-5 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 trained classifier
    - A general description of how you implemented SVM and Random Forest from scratch
    - A table that shows the accuracy of models on test sets that you implemented and the models exist in standard libraries
2. The video should clearly show that a person is carrying a phone and the desktop app is classifying their actions in real-time
3. Your clean and executable Python code

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.

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 clear and easy to understand, complete
Good	3/4	Explanations are lacking a bit of clarity or completeness, but is 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 question is not answered at all

The following grading scheme will be used:

Task	**Grade**		**Weight**
1. Data collection	/ 4	Completeness/thoroughness, balance, diversity, good data collection principles	3
2. Data storing	/ 4	Proper data storage in the specified format, reasonable train-test splits, no data leakage	2
3. Visualization	/ 4	Several samples visualized, each class represented, meta-data visualized, additional creative plots, good visualization principles	2
4. Pre-processing	/ 4	Removal/reduction of outliers, removal/reduction of noise, discussion or remedy of imbalance, normalization, further visualization of data after pre-processing	2
5. Feature extraction	/ 4	Identification and extraction of a minimum of 10 different features, proper	2
6. Training the mode	/ 4	Proper construction and training of the model, reasonable results	2
7. Desktop app	/ 4	Nice/clean UI, functionality, testing of the system	2
8. Demo video	/ 4	Proper description and demo of the work, participation from everyone	3
9. Report	/ 4	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 on pages 3 and 4 of this document	7

The final grade for the project will be calculated out of 100. Up to 10 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.

Note: The use of generative AI such as ChatGPT is prohibited in Final Project submission and is considered a violation of the academic integrity principles of Queen's University. Please note that, we will check the assignments using the latest AI-content detectors on a random basis