**Assignment 3 – Sensing and Machine Learning**
Release Date/Time: March 14th (Monday), 2022
Due Date/Time: March 29th (Tuesday), 2022, by midnight EST
*Last Updated: Mar 13th, 2022. 8:04pm EST*

In this assignment, you will use the accelerometer sensor on your phone to detect gestures. The aim of this assignment is to learn: (1) how to do feature engineering by analyzing your sensor data, (2) tuning the parameters of classification algorithms, and (3) training, validating and testing your machine learning models. You will use the smartphone's sensors to send data to a Python server, where you will featurize it, train it through a ML pipeline and then perform live inference. Starter code for the same will be provided.

Starter code: HW3GestureClassifier.zip

Note the starter code is slightly different from what was demoed in the class, instead of swiping for different classes, the app prompts you for which gesture training data to collect.

**Part 1: Gesture Detection using Accelerometer**

In this part, you will build a gesture detector using the smartphone accelerometer, and a Machine Learning model trained on a Python backend. Building on top of your assignment 2 for gestures, you are free to choose any four of the alphanumeric characters (lower or uppercase). There are two parts of the assignment:

- Make changes to the script server.py adding your featurization code to the featurize function and trying different ML classifiers in train_ml_classifier. This will be used to showcase the demo of your system working.
- Collect 10 trials per gesture using server.py and save the data for analysis using visualization.py ( or Jupyter - visualize.ipynb). Here you can test out your features and use the featurization code directly in server.py. Ideal goal is to make a pipeline that gets **average** of 90%+ accuracy in a Leave-One-Trial Out Cross Validation across gestures across 3 different classifiers(eg. SVC, KNeighborsClassifier,RandomForestClassifier). The reason we look at the average accuracy is to reduce overfitting. Accuracy and starter signal visualization code is included. The starter code flattens the raw values and uses them as features. In your final code, do not use the raw values(raw values will be prone to per-session overfitting) by themselves but rather extract features from them. The ideal goal is to use no more than 5 features.

- Which features work the best? Make a plot showcasing the feature ranking of the different features you used. Also note down how you did it.

**Part 2: Generalizability of your model.**

How generalizable is your model? Apart from training/testing on your chosen four gestures in part 1, we will ask you to train your model on one more alphabet gesture of our choice.

We will share the gesture activity during the demo and will evaluate if the features and algorithms you chose are able to handle an unknown class. Remember, you will still be able to train your model for this gesture/sound activity during the demo with 10 training examples. In your implementation to have an identifier for this extra class, just add a "bonus" class to your processing code for both parts.

**Note 1) Turn off auto-rotate on your android smartphone (this prevents respawning of the app on each tilt). 2) Go into "Sketch Permissions" and grant internet access for the android application on the Processing IDE.**

**Deliverables:**

- Python source code
- Processing source code
- Video capturing the phone screen while performing the gestures activity
- Write-up
    - Features used and your feature selection process
    - Feature importance plot of your final selected features
    - With your final selection of features: average accuracy across at least 3 ML model, accuracy of the ML model used in demo app

Demo of the whole system during the office hours

Grade distribution:

1. Accuracy of **averaged** gesture recognition in part 1: 40% (0.25 % point for each accuracy point. e.g., 35% grade for a system with 70% accuracy).
2. Smartphone implementation: 10% (Since its mostly starter code)
3. Python implementation and Feature Selection: 25% (1% point deduction for each additional feature, maximum deduction of 20%, i.e. if you have more than 25 features, you only lose 20%)
4. Writeup: 15% (5% for each sub-bullet point in the deliverable)
5. Stretch goal (instructor-selected gesture du ring demo for part 3): 10%