Project Phase II

Instructions:

- This project aims to give you a hands-on with a real-life machine learning application.
- Use separate training and testing data as discussed in class.
- This is a group project, and the maximum number of members per group is 6. **Only one submission per group is required.**
- **Carefully read the submission instructions and plagiarism policy at the end of the project.**
- The deadline to submit phase II is **Tuesday, 14th December 2021.**

Problem:

This phase is a continuation of the previous phase. This time, you are given the data generated by the class. You will split the data into train-test sets and train your classifiers to classify given tests audios in the three classes: "en", "ur", and "ue". Your job is to try and maximize the classification accuracies and F1 scores. This phase will be relatively graded; Groups that perform better will be given more credit.

Feature Extraction:

In the feature extraction step, you will represent each WAVE file by 13-dimensional Mel-frequency Cepstral Coefficients (MFCCs). MFCCs are a widely used representation of human speech. A code snippet is provided here that shows how to install the required library and read & represent a sample WAVE file with an MFCC vector.

Part 1:

Use scikit-learn's implementation of **one** of the classifiers below to train and test on the provided dataset.

1) KNN
2) Decision Trees
3) Random Forest

Your decision can impact the scores for this part; Choose wisely!

Do 5-fold cross-validation and report the validation and training losses using graphs. Use scikit-learn's accuracy_score function to calculate the accuracy, classification_report to calculate macro-average (precision, recall, and F1), and confusion_matrix function to calculate confusion matrix on the test set.

Part 2:

Use scikit-learn's Kernelized Support Vector implementation to train and test the Kernelized SVM on the provided dataset. Do 5-fold cross-validation and report the validation and training losses using graphs. Use scikit-learn's accuracy_score function to calculate the accuracy, classification_report to calculate macro-average (precision, recall, and F1), and confusion_matrix function to calculate confusion matrix on the test set.

Part 3:

Use scikit-learn's Neural Network implementation to train and test the Neural Network on the provided dataset. Do 5-fold cross-validation and report the validation and training losses using graphs. Use scikit-learn's accuracy_score function to calculate the accuracy, classification_report to calculate macro-average (precision, recall, and F1), and confusion_matrix function to calculate confusion matrix on the test set.

Note:

Apart from these classifiers, you can also use the Ensemble method to classify your data. Ensemble methods usually yield the best results. If you are looking to score high on this phase of the project, we would strongly recommend trying this out.

Submission Instructions:

Submit your code both as a notebook file (.ipynb) and python script (.py) on LMS. The name of both files should be the roll numbers of group members and the group number, e.g., rollno1_rollno2_group1.ipynb. If you don't know how to save .ipynb as .py, see this. **Failing to submit any one of them will result in the reduction of marks**.

Plagiarism Policy:

The code MUST be done independently. Any plagiarism or cheating of work from others or the internet will be immediately referred to the DC. If you are confused about what constitutes plagiarism, it is YOUR responsibility to promptly consult with the instructor or the TA. No "after the fact" negotiations will be possible. The only way to guarantee that you do not lose marks is "DO NOT LOOK AT ANYONE ELSE'S CODE NOR DISCUSS IT WITH THEM".