# Final Project

Total Points = 40pts

**Due : December 19th, 2021**

In this project, we deal with a classification problem. You can download the dataset file from KLMS. Each data point is a vector $(\mathbf{x}, y)$ where $\mathbf{x} \in \mathcal{X} = \mathbb{R}^{400}$ and $y \in \{0, 1, 2\}$ (there are 3 classes). The file **train.npz** contains the training data and the file **test.npz** contains the test data. The size of the training dataset is 1200 and the size of the test dataset is 600. Each .npz file has two arrays as a dictionary data. In detail, if you load the training data with name LOAD_FILE, then LOAD_FILE['inputs'] is an $1200 \times 400$ array and LOAD_FILE['outputs'] is an 1200 vector. For example, (LOAD_FILE['inputs'][0], LOAD_FILE['outputs'][0]) is a data point. Our objective of this project is to construct a classifier $f : \mathcal{X} \to \{0, 1, 2\}$ such that $f(\cdot)$ predicts where the input belongs to among classes 0, 1, and 2.

1. (10pts) In this problem, make python codes to train the following models that we already learned in the class.

   - Decision Tree with a hyperparemeter max_depth = 4
   - Bagging Decision Tree with hyperparameters of Decision Tree: splitter = 'random', max_leaf_nodes = 16 and hyperparameters of Bagging: n_estimator = 500, max_samples = 500, max_features = 20, bootstrap = True, n_jobs = -1
   - Logistic Regression with hyperparameters multi_class = 'multinomial', solver = 'lbfgs', $C = 10$
   - Linear Support Vector Machine with hyperparameters $C = 1$, loss = 'hinge', max_iter = 1000

   For each algorithm, train the model using the training set, measure the training time and evaluate the accuracy of the trained model using the test set. To compare the performance of algorithms, repeat 5 times to train and evaluate (using different random_state) and average the results if there is a randomness in the algorithm. Report all of your results.

2. (20pts) Tune the hyperparameters or modify the above models to improve the performance. You don't need to report all of your tries. Report your best model. Your report should include the model performance, the training time, and the description of your model. The training time of your model should not be so long compared to the models in Problem 1. (e.g. $< 1s$) [Hint : You may use the following strategies.

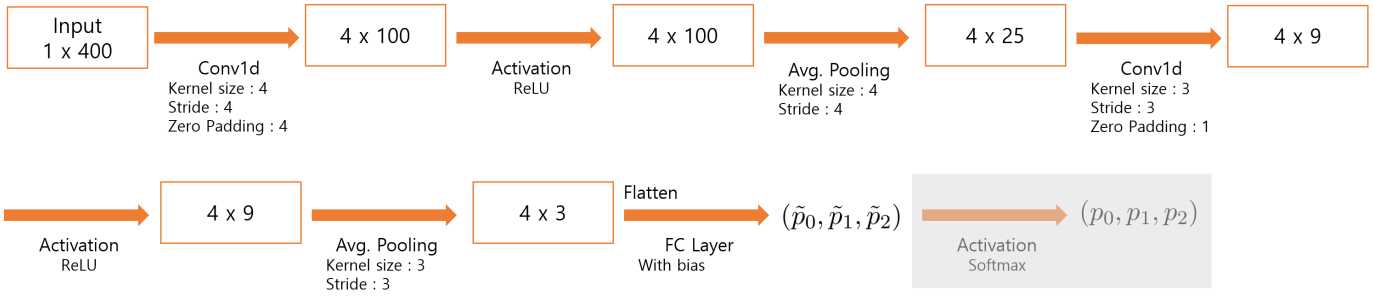
   (1) Preprocess the input vectors to find better representations of data inputs, e.g., design a transformation $\phi$ from $\mathcal{X}$ into another space $\widetilde{\mathcal{X}}$ and construct a classification model $f : \widetilde{\mathcal{X}} \to \{0, 1, 2\}$

   (2) Tune hyperparameters in the models]

[Next Page]

3. (10pts) Make a python code to train the convolutional neural network (CNN) given below. Train the model using the training set, measure the training time, and evaluate the accuracy of your trained model (when we assume we choose the class $\text{argmax}_{i\in\{0,1,2\}} p_i$) using the test set. To compare the performance of the CNN with other models, repeat 5 times to train and evaluate (using different random_state) and average the results. Report your five train results and the final result.

   - Use **torch.optim.Adam** with learning rate 0.005, **batch size** 30 and **the number of epochs** 20. The architecture is as follows. (Your architecture do not need to contain the gray box.)




[END]