# Machine Learning: Programming HW 1

G section

Due date: October 3, 2022, 8:45AM

**General notes:**

1. This assignment carries 3% weightage towards your final score in this course.
2. You will be doing this assignment in teams of two.
3. Please cite all the sources that you have used for solving this: even if it is your classmates, please acknowledge in writing that they helped you. **No penalty for honesty**.
4. I might randomly choose some of you for a Viva on your implementation.
5. The submission portal will automatically close at due time. That is, delay of 1 minute is also considered a late submission. No submission means a score of Zero.
6. Blatant plagiarism (copying) will give you a score of $-3$ (negative three). Note that Google Classroom allows automated test for plagiarism on 5 assignments per class and I will use it here.

**Problem statement** Build a $k$ nearest neighbor classifier in Python to classify the MNIST digit data. This is a multi-class classification problem with labels from 0 to 9.

1. **The value of k**: Different groups may have different values of $k$. The $k$ value for your group is the maximum of the last two digits of USNs of students in the group. For example, if your group has students with USNs: ENGXXAM0099 and ENGXXAM0084, then the $k$ value for your group is $\max\{99, 84\} = 99$.
2. **# data points**: The MNIST database has $60,000$ data points but we will not be using the whole data. Different groups may have different number of data points to work with: add the last two digits of USNs of students in the group and multiply it by hundred and add 1000. That will be the size of your data. For example, if your group has students with USNs: ENGXXAM0099 and ENGXXAM0084, then the $n$ value for your group is $(84+99)\times 100+1000 = 19300$. **Use the first $n$ points in the dataset as your data.**
3. **Testing process (cross-validation)**: Randomly pick 20% of your data for testing. Do the testing $k$-many times (each time picking the 20% test data randomly) where $k$ is determined as above. Report the average of scores: a confusion matrix with averaged entries.
4. **Implementation**: You will code the $k$NN algorithm on your own and generate an averaged multiclass confusion matrix for it (as discussed above). You will also use the python `scikit-learn` library and generate an averaged multiclass confusion matrix for it (as discussed above).

**Submission** Create and upload a document (no specific format) that must show:

1. Obviously, your names and USNs,
2. The $k$- and $n$-value calculations as per your USNs,
3. Your code,
4. The multiclass confusion matrices generated by your implementation of the algorithm and using the library implementation, and
5. A few lines of observations/inferences/comments on the matrices.

**Some sources which might be useful (clickable links)** Feel free to use any source you might want to, but please give them due credit. Some sources which might help you get started are:

1. How to Load and Plot the MNIST dataset in Python?
2. scikit-learn user guide for nearest neighbours
3. Nearest Neighbors Classification: Example implementation
4. Multilabel Confusion Matrix