Inf2B Coursework
Supplemental document for Task 1 (Ver. 0.9.1)

Instructions on the partitioning of data set for k-fold CV

Since the data set we use in the coursework is not very large, random partitioning could result in undesired partitions in which some classes have too few samples to run evaluation. To avoid the problem, we use class-wise partitioning - we split the data set into K partitions for each class so that each partition has a similar distribution to that of the data set. To that end, you should use the following algorithm for each class `c` $= 1, \ldots, C$, where C is the number of classes.

> Let `Nc` be the number of samples for class `c` in the data set. The number of the samples of class `c` assigned to the first partition is given by the following Matlab code:
>
> ```
> Mc = floor( double(Nc) / double(K) );
> ```
>
> Now, assign the first `Mc` samples of class `c` to Partition 1, and assign the next `Mc` samples to Partition 2, and so on up to Partition $K-1$. The last Partition K takes all the remaining samples, whose number should be equal to `Nc - Mc*(K-1)`.

Note that `PMap(i)` holds the partition number that `i`-th sample in `X` was assigned to, whereas `Y(i)` holds the class number of the `i`-th sample. This means that you should not shuffle the data set at all.

Instructions on k-fold CV

When you carry out k-fold CV with $k = K$, you repeat cross validation K times, selecting each partition as the validation (test) set and the remaining $K-1$ partitions as the training set. After you have run all the combinations to obtain K results, you calculate the average of results as an estimate with k-fold CV. To obtain an average confusion matrix, the confusion matrix for each run should be normalised by the number of samples in the test set before averaging.