Classification II
Exercises

Exercise 1. (Nested CV)
For this task, use the digits dataset, that we already investigated at the beginning of this module.

1. Write a program, that optimizes and evaluates classifiers on that dataset using a nested cross validation setup.
   - For the number of repetitions in inner and outer cross validation and for the number of splits use 3.
   - (Note: Normally you will choose higher numbers, e.g. 10. We go with the smaller choice here to allow completing the task during the session.)
   - Hint: Look at `sklearn` components `GridSearchCV`, `StratifiedKFold`, `RepeatedStratifiedKFold`, and `cross_validate`.
   - Create a function that takes a specific estimator and parameters and returns the resulting evaluation.
2. Evaluate
   - k-NN with uniform- and distance-based voting and k running from 1 through 10,
   - decision trees with gini and entropy, and choices for the maximum tree depth as powers of 2: $2^0, 2^1, \ldots, 2^8$,
   - support vector machines with polynomial and radial basis function kernels with $C \in \{0.1, 1, 10, 100\}$ and degree $d \in \{1, 2, 3\}$ and $\gamma \in \{0.001, 0.01, 0.1, 1, 10\}$.
3. Extend your program using a pipeline that contains scaling and the preprocessing of the raw data.
4. Extend your program to report and compare various metrics:
   - mean, standard deviation, min, and max for accuracy and balanced accuracy
   - average learning time
   - average prediction time.
5. Decide which classifier you are going to use and create the final model.
6. Use the final model to predict the number for the array in the file `sample_digit.npy`.

Exercise 2.
Discuss the following aspects of the above analysis:

1. Which insights does the analysis generate?
2. Which further insights are desirable, but do not result from the above procedure?
3. Which increase in time do you expect when using larger numbers of repetitions and splits?
4. How representative is the reported average training and evaluation time?
5. In imbalanced class problems, one could use oversampling to improve the performance on the smaller classes. Where in the above setting would the oversampler have to be included?