The objective of this lab is for you to program in Matlab ensembles of learners (based on bagging, for regression or classification), apply them to some datasets and observe their behavior. The TA will first demonstrate the results of the algorithms on several datasets, and then you will program them, replicate those results, and further explore the datasets with the algorithms. You can use the textbook, lecture notes and your own notes.

# I Datasets

Construct your own toy datasets to visualize the result easily and be able to get the algorithm right. For regression, take the input instances $\{x_n\}_{n=1}^N$ in $\mathbb{R}$ and the output values $\{y_n\}_{n=1}^N$ in $\mathbb{R}$. Generate a noisy sample from a known function, e.g. $y_n = f(x_n) + \epsilon_n$ where $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$ and $f(x) = ax + b$ or $f(x) = \sin x$. For binary classification, take the input instances $\{\mathbf{x}_n\}_{n=1}^N$ in $\mathbb{R}$ or $\mathbb{R}^2$ and the labels $\{y_n\}_{n=1}^N$ in $\{-1, +1\}$.

# II Ensemble learning using bagging

In bagging, we generate $L$ bootstrap samples of the training set $\{(x_n, y_n)\}_{n=1}^N$, train a learner in each, and combine their outputs by averaging them (for regression) or by majority voting (for classification). So programming this in Matlab just requires a loop over the $l = 1, \dots, L$ learners. You can use any learner you want, we suggest the following:

- Regression: polynomials (degree 1 gives a linear regression); Gaussian RBF networks; decision trees.
- Classification: logistic regression; $k$-nearest-neighbor classifier; support vector machine; decision trees.

Use the provided script `lab11_regr.m` as driver for your experiments. It creates a dataset (training and validation) for regression, trains $L$ learners (specifically, polynomials of different degree, using the function `polytrain.m` from a previous lab), and plots the results. See more details below.

The point of this lab is to learn about the combination of learners in an ensemble (rather than about the specific learners themselves, which we have done in previous labs). Hence, *you are allowed to use as base learner any function*: your own functions from previous labs, or any function from Matlab or from the web, for regression or for classification. Make sure to provide such functions along with your suitably modified `lab11_regr.m` in your submission so we can test it. If you just use polynomials, which we provide, your lab grade will be lower.

Decision trees are among the most interesting learners to use in an ensemble. You can use the Matlab function `fitctree` to train them. Make sure you understand how to use it properly, in particular how to set any (default) parameters in may require.

**Exploration: regression** We discuss polynomials as an example. Consider an ensemble of $L$ polynomials of the same degree $k$ ($k = 1$ corresponds to linear regressors). To visualize the results, create the following plots:

- Plot the dataset ($y_n$ vs $x_n$), the true function $f(x)$ from which you generated the data, the $L$ learners' functions and the ensemble function.
- Plot the training and validation error as a function of the degree $k$ of the polynomials, and of the number of learners $L$.

Questions to consider:

- How does the ensemble regressor look, compared to the individual regressors?
- How does the validation error of the ensemble compare with the validation errors of the individual learners?
- How does the validation error depend on the degree $k$ (= complexity) of the polynomials?
- How does the validation error of the ensemble behave as $L$ increases?

**Exploration: classification** Define ensembles where the learners are of the same type, e.g. $k$-nearest-neighbor classifiers with fixed $k = 1$, and proceed as with regression. Questions to consider:

- Similar questions as for regression, suitably modified for your learners (e.g. effect of $k$ on the $k$-nearest-neighbor classifier, effect of the tree depth on decision trees).
- How do the following ensembles (of fixed size $L$) compare with each other: decision stumps; deep decision trees; $k$-nearest-neighbor classifiers; linear SVMs.

# III What you have to submit

We provide you with a script `lab11_regr.m` which sets up the problem (toy dataset for regression), trains $L$ learners (polynomials) and plots the figures mentioned earlier. You have to adapt it to a learner type of your choice and explore its behavior.

**Follow these instructions strictly.** Email the TA the following packed into a **single** file (`lab11.tar.gz` or `lab11.zip`) and with email subject `[CSE176] lab11`:

- Matlab code for your modified script `lab11_regr.m` and any functions it may depend on, so that the script works when we run it.
- A brief report (2 pages) in PDF format describing your experience with the algorithms. The more extensive and insighful your exploration, the higher the grade. Be concise. Don't include code or figures, we can recreate them by running your functions. Indicate the part that each member of the group did.