Please carefully read and follow the general instructions regarding computing assignments. Failing to meet the requirements might lead to penalties. [https://moodle.uef.fi/mod/page/view.php?id=1935632](https://moodle.uef.fi/mod/page/view.php?id=1935632)

If you suspect that something is wrong with some task instructions, please contact the lecturer.

If you face persistent issues while working on a task, do ask for help, e.g. during a course meeting or by contacting the lecturer via email.

Attached material.

- **iris-SV-sepal.csv** IrisSV, two-dimensional binary classification dataset, linearly separable
- **iris-VV-length.csv** IrisVV, two-dimensional binary classification dataset, not linearly separable
- **creditDE.csv** Credit, multi-dimensional binary classification dataset
- **classification_resources.py** Some potentially useful code snippets

! Imports of external libraries other than those that appear in the `classification_resources.py` file are not allowed.

**Task 1.** Fill in the gaps in the code (look for dots, i.e. “## ...”) to complete the implementation of the linear SVM algorithm with hard-margin and soft-margin variants. Apply them respectively to the IrisSV and IrisVV datasets.

That is, divide the IrisSV dataset into training and test subsets in proportions 4/5–1/5 at random, i.e. assign one fifth of instances, chosen at random, to the test dataset and the rest to the training dataset. Train a hard-margin SVM on the training subset, and apply the resulting model to the test subset. Compute and report the accuracy of the prediction. Give the equation of the separating hyperplane. Plot the separating hyperplane and highlight the support vectors.

Do the same with soft-margin linear SVM on the IrisVV dataset (setting $c = 2$, for example).

**Task 2.** Run an evaluation of the soft-margin SVM on the IrisVV dataset with cross-validation.

That is, run 10 rounds of cross-validation with 5 folds on the IrisVV dataset. Report the mean and variance of the classifier’s accuracy across the successive rounds.

**Task 3.** Implement the AdaBoost algorithm with instance weighting done via sampling. Apply it to the Credit dataset with linear SVM. Try using less aggressive weight updates, calculating the update factor as

$$
\alpha_t \leftarrow \beta \cdot \ln((1 - \epsilon_t)/\epsilon_t)/2
$$

where $0 < \beta < 1$. What happens if $\beta = 0$?

**Task 4.** Empirically compare the impact of boosting and bagging when combined to a linear SVM vs. to a SVM with a RBF kernel on the Credit dataset.