

Conformal prediction: SoTA course JTH 2025

To pass, you need to complete the first programming assignment (Task 1) below.

The other three tasks are **not** mandatory but suggested for students who want to delve deeper into conformal prediction. Note that tasks 2-4 are independent, e.g., you don't have to solve Task 2 to solve Task 4; so just pick one task that you find interesting.

You should work in pairs. Solutions should be emailed to ulf.johansson@ju.se on Monday December 1, at the latest. Please see the exact format of the solution below.

Task 1: Empirical investigation of conformal classification (Mandatory. Easy)

In the lecture, we covered conformal regression. Conformal prediction was, however, first introduced for classification. To complete this task, you should implement inductive (a.k.a. split) conformal classification and perform an empirical evaluation looking at error rates and efficiency. More specifically, you should report the results from an experiment using the following setup:

- 10-fold cross validation
- At least ten of the provided two-class datasets
- At least three different predictive models (e.g., random forest, decision trees, XGB, etc.)

Report, in addition to predictive results like accuracy and/or AUC, empirical error rates for at least three significance levels, and some metric covering efficiency¹, i.e., informativeness of the conformal predictions.

The submission should be a Jupyter notebook, where you in the first (text) cell give your own (i.e., **not** AI-generated) description of conformal classification. Your description should cover:

- The procedure used
- The guarantees obtained
- An **informal** reasoning about why the procedure produces the guarantees

After that, you should have one or more well-commented (code) cells solving the task and reporting the results in a good way.

IMPORTANT: When you send the notebook, it should include the results, i.e., showing the output from the cells, but it should also be possible to run the cells from a folder containing the notebook and all datasets in a folder called `twoclass`.

Finally, add one (text) cell where you comment on the results.

For this task, **you are allowed to do vibe coding**, but you must make sure that you understand the generated code. If this is not clear from your description in the notebook, I might follow up with an oral examination before passing the assignment. Your solution may (but doesn't have to) utilize the `Crepes` package.

¹ Note that this is efficiency as used in conformal prediction, not algorithmic complexity or running time

Task 2: Class-conditional conformal prediction (Not mandatory. Easy)

The guarantees from standard conformal classification are global, which could (and often does) lead to much higher error rates for one class. In class-conditional conformal prediction (CCCP – yes it really is the abbreviation used 😊) a so-called Mondrian setup is used to provide independent guarantees for each class.

Implement class-conditional conformal prediction and redo the evaluation from Task 1, but report class-specific error rates when using and not using CCCP.

Task 3: “Normalized” conformal classification (Not mandatory. Medium)

In conformal regression, adding a difficulty component to the non-conformity function results in instance-specific predictions while keeping the guarantees. More specifically, harder instances get larger prediction intervals. While there is no obvious equivalent procedure for classification, a Mondrian approach where the categories are determined from difficulty estimations will result in more specific predictions.

Implement this and evaluate it in a reasonable way. Please reason about how this could be used in a real-world setting, and why it would be beneficial.

Task 4: Interpretable Conformal Classifiers (Not mandatory. Hard)

(This task is intentionally open-ended.)

Combining uncertainty quantification and interpretability / explainability is important when striving for trustworthy AI-models, e.g., in decision support systems. Implement, in a clever way, conformal classification trees, i.e., decision trees where each leaf returns a conformal prediction set. Depending on the calibration (Mondrian or not), you would get global or local guarantees.

Think carefully about how deep the trees should be; more shallow trees are of course easier to comprehend, but fewer leaves also makes it possible to use a Mondrian approach, where each leaf is its own category.

Implement and evaluate this as a proof-of-concept. A key component is of course the ability to provide visualizations of the trees.