ExCAPE WP1. Integration of Conformal Prediction with ML Algorithms

Paolo Toccaceli*, Ilia Nouretdinov*, Zhiyuan Luo*
Vladimir Vovk*, Lars Carlsson** and Alex Gammerman*

*Royal Holloway, University of London
**AstraZeneca, Sweden

1 Introduction

The objective of this subpackage is to integrate Conformal Prediction (CP) [1] with the Machine Learning methods adopted in ExCAPE. Conformal Prediction was described in the first Report [2], with particular emphasis on Inductive and Class-conditional (Mondrian) forms. The deliverable is a Python module that implements Mondrian Inductive CP (MICP). The module is meant to be used as a stage downstream the ML algorithms in the overall pipeline that takes the EXCAPE DB data as input and produces predictions as final output. We understand that th The module offers a reference implementation. Partners in WP2 and WP3 are free to re-implement it

2 Conformal Prediction

We summarize very briefly those key points that need to be understood in order to use the implementation sensibly.

As detailed in [2], Conformal Predictors take NonConformity Measures (which express how non-conform a given example appears compared to those in the training set) and, given a significance level, produce Region Predictions, i.e. subsets of the label space (rather a single label). The prediction is said to be valid in the sense that errors, i.e. cases in which the Region Prediction does not contain the actual label, occur with a rate that is less than or equal to the chosen significance level (subject to statistical fluctuations). In fact, to compute the region predictions, CP first produces for every test object p-values for each of the possible labels. Then the region predictor outputs the following for every test object:

$$\Gamma_\epsilon = \{y \in Y : p_y > \epsilon\}$$

where $\epsilon \in [0, 1]$ is the chosen a significance level and Y is the set of the label values.

The NCMs are computed using a ML method, which is referred to as the underlying method in this context. The CP variant we recommend for ExCAPE is the Mondrian

Inductive one. In the Inductive Conformal Prediction, the underlying is trained on a proper training set. Then, NCMs are obtained for a calibration set. Finally, NCMs are calculated for each test object and for each possible value that the label of test object might take.

Once we have the NCMs for the calibration sets and the NCMs for the test objects with their hypothetical label assignments, we can apply the methods implemented in the Python module.

Finally, the Mondrian (i.e. class-conditional) qualification refers to the fact that the variant ensures validity for each class (i.e. label value) separately. This is essential for highly imbalanced data sets such as those in ExCAPE DB, where in several cases the Active class is 1% of the total. The Mondrian variant ensures that validity is guaranteed also for the minority class.

2.1 NCM

The NCM expresses how contrary to the hypothesis of randomness an example $(\mathbf{x}, y)$ appears to be. Informally, it is meant to have a value that is the greater, the "stranger" the example looks, compared to the examples in a set. For example, the NCM of $(\mathbf{x}_\ell, 0)$ should be large if the training set contains many examples in which the object similar to $\mathbf{x}_\ell$ but with label 1 and few or none with label 0. Note that the NCM depends on the object and the label.

The NCM can be calculated by means of a scoring classifier, i.e. a classifier which outputs a value that is directly related to the posterior probability of a given object to belong to a given class (in our case, the Active class).

To the best of the author's knowledge, this is the case for all the ML methods considered in ExCAPE. Under this assumption (i.e. a larger score means a higher posterior probability of the compound being Active), the NCM is computed easily by:

$$NCM(\mathbf{x}, y) = \begin{cases} -s(\mathbf{x}) & y = \text{Active} \\ +s(\mathbf{x}) & y = \text{Inactive} \end{cases}$$

where $s(\mathbf{x})$ is the score computed by the scoring classifier.

3 Implementation

The reference implementation was submitted, as per instructions, to the project git repository at `exapubgit@www.exascience.com:excape`, under `excape/code/wp1/MICPv2`. It is accompanied by a full example in the form of a Jupyter Notebook, so the reader is referred to that for further details.

References

[1] Vladimir Vovk, Alex Gammerman, and Glenn Shafer.
Algorithmic Learning in a Random World.
Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[2] Paolo Toccaceli, Ilia Nouretdinov, and Alex Gammerman.
ExCAPE WP1. Conformal Predictors.
http://www.clrc.rhul.ac.uk/projects/ExCAPE/Report_wp1_dec05a.pdf