RP Phase 3: Advanced Evaluation of ML Classification

December 13, 2020

While the paper focuses more on binary classification when evaluating performance and interpreting classifier behavior, our problem is multilabel with over 200 possible output classes, where each input record can belong to multiple classes at once. For this, we believe that the closest analog we can do to what the paper suggests is to apply the paper's methodology (precision/recall curves, empirical risk curves, feature importance, mistake patterns, ...) for each output class separately.

So, after performing a train/test split on our data, we should train the different models we have on this same train dataset (to avoid data leakage when comparing performance on the exact same test data), predict the outputs of the test dataset, and focus on each output class separately when assessing performance, so that each class has its own set of precision/recall curves, its own empirical risk curves, its own mistake pattern analysis and feature importance sets, etc.

Since in the previous TP phase, not all our models were trained on the exact same train/test split, we cannot compare all of them against each other without having some data leakage, and re-training these models would take days. Hence for this phase we only focus on comparing SVM to Random Forests, for the first output class (similar work and plots can be repeated to the other 200 or so classes). The relevant code can be found in the included Python notebook.

However, we were surprised to find that our pre-saved pickle file for the trained random forest model takes a total of 4.5 GB of storage, and hence we were unable to load it into memory in order to reuse it for this phase. So, the only pickled model we have from Phase 3 that we can use is the SVM one. As most of the work involves comparing results across models to decide which ones are the most suited to our problem, then re-fitting models will be required. This takes days, and did not finish by the due date. Hence, the code pretty much represents what we would do when the training of the models on the same data finishes, but we do not have outputs to show for now.

Whenever training finishes, applying what we described above should give us

many insights that should be quite helpful for improving our ML results. These include: which models are performing better on which output classes (so we can shortlist the best performing models), which features are the most important for which classes (helps understanding a model's "logic", and can possibly guide a more intelligent feature selection process), and which data patterns and output classes the models are struggling on the most, so that we can possibly remedy to them. Most importantly, this work is assessing performance using more reliable and understandable metrics than traditional ones, which helps us communicate results with subject-matter experts in a easier way, and explain the reasons for choosing one model rather than another using simple visuals.