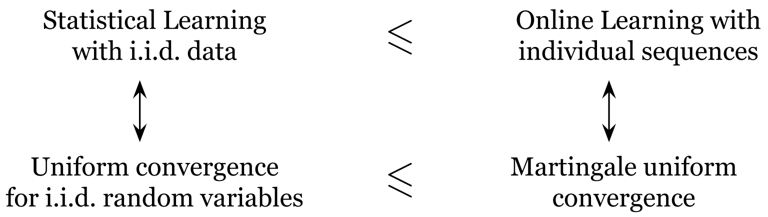
6.883: Online Methods in Machine Learning
Alexander Rakhlin

LECTURE 23

1. FINAL REMARKS

In the previous lecture, we sketched two results. First, if we have an online method that attains a non-trivial regret bound for all sequences, we may convert it into a batch method that has an i.i.d. guarantee in the style of Statistical Learning Theory. Second, we showed, for a particular example, that existence of a strategy that has non-trivial regret for all sequences is equivalent to probabilistic statements about martingales. These two results suggest a deeper connection between Online Learning and Statistical Learning. Indeed, such a connection has been recently found.


Statistical Learning with i.i.d. data
$\leqslant$
Online Learning with individual sequences
Uniform convergence for i.i.d. random variables
$\leqslant$
Martingale uniform convergence

The following discussion is admittedly vague, and misses the various assumptions under which the statements hold; however, it is hopefully useful as a rough guideline.

It is well-known that the problem of statistical learning with i.i.d. data is intimately related to questions of uniform convergence of means to expectations. This connection goes back to the beginning of machine learning – the work of Vapnik and Chervonenkis. It turns out that the problem of online learning is intimately connected (and in many cases equivalent) to the question of martingale uniform convergence. Martingale uniform convergence implies i.i.d. convergence, but not vice versa. Hence, one expects that online learning is, in general, harder than statistical learning. However, there are plenty of cases where uniform convergence for i.i.d. is no different than the corresponding statement for martingales. In such a case, one expects online and i.i.d. worlds to coincide. Essentially, this was the case in Lemma 2 in Lecture 18, where a martingale difference was replaced by an i.i.d. draw; we then ended up with classical Rademacher complexity even though the problem had no i.i.d. assumption on the sequence.

The discussion also connects to the first part of this course, and in particular Lecture 09. Recall that we had a surprise: uniform deviations needed for the analysis of SAA on the SA objective were, up to a constant, the same as the very rate of SA on the expected objective. We can now reason about the nature of this phenomenon. The rate of SA is given, as an online procedure, by martingale uniform convergence, which (in the case of Lecture 09) is identical to the i.i.d. variant.

Let us go back to the perceptron. To analyze the i.i.d. performance, one can either take the online mistake bound in terms of the margin ($1/\gamma^2$) and convert it into a Statistical Learning guarantee, or, alternatively, one may proceed with the classical Vapnik-Chervonenkis style analysis by observing that the VC dimension of the class of hyperplanes is d. In a 1968 paper (see [RS15] for the discussion), Vapnik and Chervonenkis appear to be surprised that the two distinct approaches exist. One is online, the other is offline. One is distribution-independent and based on VC theory, the other one requires the margin assumption and an online mistake bound. We now understand this connection, which involves the minimax theorem and an extension of VC theory to martingales. We refer to [RS15] for more details.

When contrasting the statistical learning theory with online learning, it is also worth emphasizing again the algorithmic potential of the online approach, as we are required to output one prediction at a time rather than the whole solution all at once.

Finally, the online paradigm of predict-test-update-predict-... allows us to test the performance on-the-go. We test on each data point and then incorporate it into the dataset. This is in contrast to the batch scenario where the only way to gauge the actual performance is to split the data into a separate test and training sets.

We finish the course with an interesting question of how to assess performance in machine learning competitions. This question is very much related to issues of uniform convergence.

1.1 Machine learning competitions

Recall our discussion earlier in the semester about splitting the data into training and test sets. The test set is supposed to be only used once to report the final error rate. However, it is nearly impossible to police this: nothing prevents the researcher from trying out a multitude of models until the test error is found to be small and reporting this final result in the paper.

In the last two decades, much of machine learning research has been driven by "competitions." Initially, the results were reported in papers as performance on a benchmark dataset (such as MNIST). Nowadays, the competitions are live and mediated by a third party (such as Kaggle). The teams sequentially adapt their methods to improve performance. Typically, a training set is made public, but the test set is locked by the organizers. Yet, to drive the competition, the teams must be allowed to check how well they are performing on this test set. A leaderboard maintains the best-performing teams.

How does one organize such a competition? This might seem like a simple question, but it turns out to be quite delicate. A typical approach is to limit the number of queries to the test set. Yet, it's not entirely clear how this helps avoid overfitting (there are zero-th order optimization methods that only require the value of the objective to perform descent). If overfitting occurs, then the leaderboard does not provide a good assessment of the relative performance of the proposed solutions.

A cute approach is the mechanism proposed in [BH15, DFH$^+$15], which we briefly describe here. Consider the case of only one team. The team queries the oracle (the keeper of the test set and the leaderboard) for the empirical value on the test set S. Let D be the population distribution from which S is obtained as an i.i.d. sample. The training set is immaterial here, and we allow the team to have an arbitrary (but deterministic) way to change the hypothesis upon obtaining new information from the oracle. The goal of the competition organizer is to report a nearly-correct error rate (with respect to D) of the leader at every time instant. Only the leader matters for this example, and not the whole leaderboard.

Let f_1 stand for the first query, f_2 for second, and so on, and let $\boldsymbol{\ell} \circ f$ be the loss function composed with the hypothesis. Of course, $\widehat{\mathbb{E}}\boldsymbol{\ell} \circ f_1 \approx \mathbb{E}\boldsymbol{\ell} \circ f_1$ by the central limit theorem, where $\widehat{\mathbb{E}}$ is the empirical measure supported on the test set S. If the oracle releases the empirical value $R_1 = \widehat{\mathbb{E}}\boldsymbol{\ell} \circ f_1$ on the test data, the next function f_2 is a function of R_1, and, hence, depends on the test data. The central limit theorem no longer applies at the second step, and, thus, there is no guarantee that the empirical value $\widehat{\mathbb{E}}\boldsymbol{\ell} \circ f_2 = \widehat{\mathbb{E}}[\boldsymbol{\ell} \circ f_2(R_1)]$ is close to the true expected value. Of course, if we know that the algorithm is only searching for hypotheses $f_1, \ldots,$ in the set $\mathcal{F}$ of a given complexity, we may invoke uniform convergence results. However, in the setting of online competitions, we cannot guarantee this. The proposal of [BH15, DFH+15] is to limit the number of functions that can be chosen by the algorithm, but the way this is enforced is by limiting the feedback given to the team. Here is their method: on round t, return $R_t = \lfloor \widehat{\mathbb{E}}\boldsymbol{\ell} \circ f_t \rceil_\eta$ (an η-discretized value of $\widehat{\mathbb{E}}\boldsymbol{\ell} \circ f_t$) if $\widehat{\mathbb{E}}\boldsymbol{\ell} \circ f_t < R_{t-1} - \eta$, and otherwise return $R_t = R_{t-1}$. If the learner is deterministic, $f_2[R_1]$ can take on $O(1/\eta)$ values according to the discretized value of R_1. In a similar manner, we can calculate the number of possible values of $f_t[R_1, \ldots, R_{t-1}]$. Suppose the loss function is bounded. Since any string $R_1, \ldots, R_t$ consists of non-increasing discretized values, it is completely described by the places where the value changes and by the jump in values. The number of possible strings is then easily bounded, which means a bound on the number of hypotheses produced by the learning method. A union bound gives the desired control of expected and empirical quantities.

Can we come up with a better way to evaluate algorithms in a competition? Can we design competitions based on the online predict-update-predict protocol?

References

[BH15] Avrim Blum and Moritz Hardt. The ladder: A reliable leaderboard for machine learning competitions. *arXiv preprint arXiv:1502.04585*, 2015.

[DFH+15] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toni Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In *Advances in Neural Information Processing Systems*, pages 2341–2349, 2015.

[RS15] Alexander Rakhlin and Karthik Sridharan. On martingale extensions of vapnik–chervonenkis theory with applications to online learning. In *Measures of Complexity*, pages 197–215. Springer, 2015.