Wind Prediction under Random Data Corruption (Student Abstract)

Conner Flansburg, Dimitrios I. Diochnos
University of Oklahoma
connerflansburg93@gmail.com, diochnos@ou.edu

Abstract

We study the robustness of ridge regression, lasso regression, and of a neural network, when the training set has been randomly corrupted and in response to this corruption the training-size is reduced in order to remove the corrupted data. While the neural network appears to be the most robust method among these three, nevertheless lasso regression appears to be the method of choice since it suffers less loss both when the full information is available to the learner, as well as when a significant amount of the original training set has been rendered useless because of random data corruption.

Introduction

Machine learning applications are prevalent in our daily lives. However, the data that is used for training a machine learning model may be corrupted or misleading due to adversarial situations that arise *naturally* (e.g., hard drive failures, transmission over unreliable mediums, etc), or *maliciously* (e.g., by attackers). Hence, studying the effects of adversarial situations that arise during the *training phase* of a learnt model, is an important research direction.

Training-time attacks can be divided into two categories: (i) *noisy settings* – see, e.g., (Valiant 1985) – where the training data are modified by *noise*, or (ii) *poisoning attacks* – see, e.g., (Barreno et al. 2006; Biggio, Nelson, and Laskov 2012) – where malicious adversaries may tamper with the training data and inject misleading information. Either way, the problem is that the machine learning model that is learnt may have larger *risk* and hence a fundamental issue is how *robust* can a model be in adversarial settings.

Framing our Work. We are interested in investigating the following question:

> *How much random data corruption can we sustain into our training sets while still producing machine learning models with acceptable error rate?*

This question can be asked both for the *average case*, as well as in a *worst-case* sense, along the lines of *certified robustness* (Steinhardt, Koh, and Liang 2017; Wong and Kolter 2018). More specifically, our machine learning models will be predicting *wind intensity* which is important in several facets of our daily lives; e.g., evacuating areas when tornadoes are about to emerge. We note that the vast majority of work in adversarial machine learning is concentrated to classification problems; regression problems are much less explored. Furthermore, we explore situations where the learner can tell if certain training examples are corrupted (e.g., through a checksum) and therefore *chooses to ignore these examples by discarding them from the training set*. This situation is similar to certain *indiscriminate clean-label* poisoning attacks (Mahloujifar, Diochnos, and Mahmoody 2020); in particular, when the adversary repeats legitimate examples, thus reducing the sample size of the learner.

Preliminaries

We use $\mathcal{X}$ to denote the set of instances and $\mathcal{Y}$ the set of possible labels. As we are predicting wind speed, the set $\mathcal{Y}$ is either $\mathbb{R}$ corresponding to real-valued predictions (in m/sec), or it is the set $\{0, 1, \ldots, 12\}$ corresponding to the different values of the Beaufort scale. We use c to denote the *ground truth* function $c\colon \mathcal{X} \mapsto \mathcal{Y}$ that assigns wind intensity to an instance x. We use $\mathcal{H}$ to denote the *hypothesis class* which contains all possible *hypotheses* (or *models*) that we can form. We denote with $h\colon \mathcal{X} \mapsto \mathcal{Y}$ a specific *hypothesis* that we select from $\mathcal{H}$. A loss function ℓ is defined as $\ell\colon \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$. That is, for a true label $y_1 = c(x_1)$, a prediction $y_1' = h(x_1)$ suffers loss $\ell\left(y_1', y_1\right) = \ell\left(h(x_1), c(x_1)\right)$. Given an underlying distribution D on $\mathcal{X}$, the *(true) risk* of a hypothesis h is $R_D\left(h, c\right) = \mathbf{E}_{x \sim D}\left[\ell\left(h(x), c(x)\right)\right]$. Given a sample $S = (x_1, \ldots, x_m)$, the *empirical risk* of a hypothesis h is $\widehat{R}_S\left(h, c\right) = \frac{1}{m}\sum_{i=1}^{m} \ell\left(h(x_i), c(x_i)\right)$. The idea is that when one selects the hypothesis that minimizes the empirical risk (w.r.t. a sufficiently large training set), then that hypothesis is close to the best solution that one could find in the entire $\mathcal{H}$ as far as the *true risk* is concerned.

Dataset. We are using a dataset with information for 16 cities (McGovern et al. 2021). Roughly, each city has 3,140-3,200 instances; each instance has 2,723 or 3,567 attributes. We perform an 80-20 split of the original dataset, so that 80% of the examples of a particular city is used for training, while the most recent chronologically 20% is used for testing the model that we learn so that we can estimate its predictive accuracy. With this idea in mind we created histograms for the training sets and test sets for all 16 cities and we decided to investigate the situation in 3 cities (kcys, kroa, and kdfw) where the distributions (in Beaufort scale) were skewed towards smaller or larger values, or there was a large discrepancy around the mode of the two distributions. The full version of the paper has all the details.

Methods and Experiments

Typical machine learning investigations aim at achieving low risk. However, here we care primarily on understanding better the rate by which our model improves as it has access to successively more and more data points (corresponding to a situation where, e.g., we have fewer and fewer hard drive failures). We tested *ridge regression*, *lasso regression*, and a *multi-layer neural network*. Linear models such as ridge and lasso regression are widely used in atmospheric sciences because they are more *interpretable*. We included a neural network as an additional comparison.

Error Calculation and Smoothing. We are using the squared loss function. Hence, in the test set (that has size t) the empirical risk is given by the mean squared error (MSE): $\frac{1}{t}\sum_{i=1}^{t}(h(x_i)-y_i)^2$. We also record the root mean squared error (RMSE). We ran every experiment $s = 10$ times so that we can smooth the results and understand in a better way the behavior of the learning process. Therefore, for example, when using RMSE for computing the error, we used the formula: $\frac{1}{s}\sum_{j=1}^{s}\left(\sqrt{\frac{1}{t}\sum_{i=1}^{t}(h_j(x_i)-y_i)^2}\right)$.

Random Data Corruption. By fixing a random permutation on the training examples at the beginning of each smoothing iteration we satisfy two purposes: *(a)* we are smoothing the results of the learning process, as we encounter the training examples in a random order, and *(b)* we are able to simulate random data corruption on our training set, where the (randomly selected) corrupted data are the training examples that belong to the batches that we have not used during a particular part of the learning process.

Experiments. Figure 1 presents the RMSE of the learnt model, as the learning process had access to successively more partitions of the training set for the city kcys. We can see that when the dataset for this city suffers random data corruption of about $1/3$ of the entire set and is thus *discarded* by the learner, then the RMSE may increase by at most 6% in the worst case. More information about all the tested cities is available in the full version of the paper, including non-smoothed worst-case guarantees.[1] Finally, for small values of random data corruption we may even observe improvements in our predictions (Min, Chen, and Karbasi 2020).

Conclusions

We studied the robustness of ridge regression, lasso regression and of a neural network under random data corruption. *Lasso* is the method of choice, even if the neural network is affected less by a reduced training-set size. One can perhaps also draw a parallel between this conclusion and the observation that, for linear models, robustness to adversarial examples (test-time attacks for classification models) can be achieved via an L_1-norm penalty on the weights within the loss function; (Goodfellow, Shlens, and Szegedy 2015; Wong and Kolter 2018).

[1]Source code: https://github.com/brom94/AAAI2022

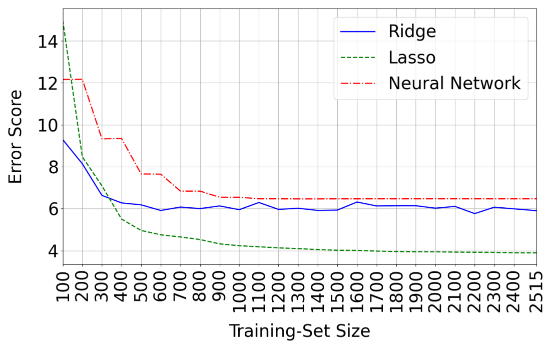

Figure 1: The smoothed (over $s = 10$ iterations) RMSE of the three models that we tested for the city *kcys*.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. AGS-1560419 and Grant No. ICER-2019758. This work is part of the NSF AI Institute for Research on Trustworthy AI in Weather, Climate, and Coastal Oceanography (AI2ES).

References

Barreno, M.; Nelson, B.; Sears, R.; Joseph, A. D.; and Tygar, J. D. 2006. Can machine learning be secure? In *ASIACCS*, 16–25. ACM.

Biggio, B.; Nelson, B.; and Laskov, P. 2012. Poisoning Attacks against Support Vector Machines. In *ICML*. icml.cc / Omnipress.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. In *ICLR*.

Mahloujifar, S.; Diochnos, D. I.; and Mahmoody, M. 2020. Learning under p-tampering poisoning attacks. *Annals of Mathematics and Artificial Intelligence*, 88(7): 759–792.

McGovern, A.; Burke, A.; Harrison, D.; and Lackmann, G. M. 2021. A Machine Learning Tutorial for Operational Forecasting: Part I. *Weather Forecasting*. In press.

Min, Y.; Chen, L.; and Karbasi, A. 2020. The Curious Case of Adversarially Robust Models: More Data Can Help, Double Descend, or Hurt Generalization. *CoRR*, abs/2002.11080.

Steinhardt, J.; Koh, P. W.; and Liang, P. 2017. Certified Defenses for Data Poisoning Attacks. In *NeurIPS*, 3517–3529.

Valiant, L. G. 1985. Learning Disjunction of Conjunctions. In *IJCAI*, 560–566. Morgan Kaufmann.

Wong, E.; and Kolter, J. Z. 2018. Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, 5283–5292. PMLR.