

15.095: Machine Learning under a Modern Optimization Lens

Recitation 2

September 13, 2018

Outline

- 1 Julia and JuMP
- 2 Validation Set
- 3 Review of Lecture 2
- 4 Review of Lecture 3
- 5 Robust Optimization Applications

Norm in Julia and JuMP

Let $p \geq 1$ be a real number. The p -norm(also called ℓ_p -norm) of vector $\mathbf{x} = (x_1, x_2, \dots, x_n)$ is:

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^n |x_i|^p \right)^{1/p}. \quad (1)$$

How to use norm in JuMP?

- 1 Norm-2, you can use norm directly in JuMP.
- 2 Norm-1, Lecture 3 page 14.

Validation Set

A **validation set** is a set of data used to tune the hyperparameters (i.e. the architecture) of a model.

Lecture 2: Sparse Linear Regression

- 1 Traditional Method(lasso regression):

$$\min_{\beta} ||y - X\beta||_2^2 + \lambda \sum_i |\beta_i| \quad (2)$$

- 2 Things need to improve: No guarantee for the sparsity.
- 3 Optimization to solve the problem: MIO

$$\begin{aligned} \min_{\beta, z} & ||y - X\beta||_2^2 \\ \text{s.t.} & |\beta_i| \leq M_i \cdot z_i, i = 1, \dots, p \\ & \sum_{i=1}^p z_i \leq k, z_i \in \{0, 1\}, i = 1, \dots, p \end{aligned} \quad (3)$$

we used logical variables and big-M method we taught last time here.

Lecture 2: First order method

Algorithm 1

Input: $g(\beta)$, L , ϵ .

Output: A first order stationary solution β^* .

1. Initialize with $\beta_1 \in \mathbb{R}^p$ such that $\|\beta_1\|_0 \leq k$.

2. For $m \geq 1$

$$\beta_{m+1} \in H_k \left(\beta_m - \frac{1}{L} \nabla g(\beta_m) \right)$$

3. Repeat Step 2, until $g(\beta_m) - g(\beta_{m+1}) \leq \epsilon$.


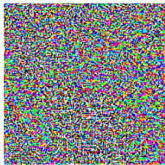

where $H_k(\beta_m - \frac{1}{L} \nabla g(\beta_m))$ retains the k largest elements of $\beta_m - \frac{1}{L} \nabla g(\beta_m)$ and sets the rest to zero.

Lecture 3: Robust Linear Regression

- 1 Traditional Method: Using lasso regression to add sparsity and avoid over-fit
- 2 Things need to improve/Optimization to solve the problem:
Equivalence between robustness and regularization. Provide theoretical evidence to explain how the regularization term works.

Problems in Neural Network

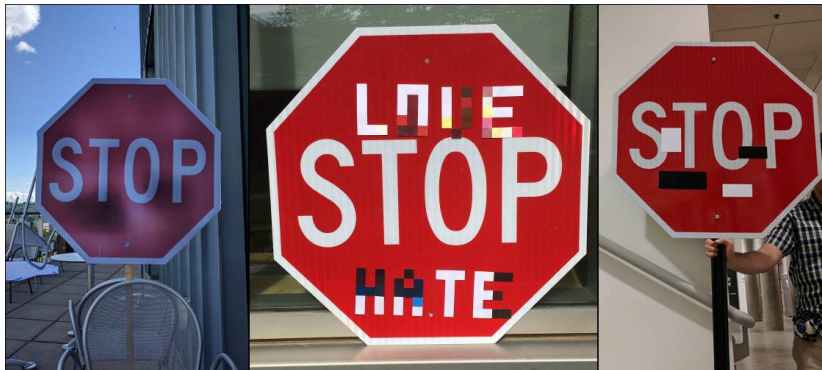
Neural Network is vulnerable to adversarial examples (examples with noise), which are examples that are crafted to cause a particular misclassification.

	$+ .007 \times$		$=$	
x		$\text{sign}(\nabla_x J(\theta, x, y))$		$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$
“panda”		“nematode”		“gibbon”
57.7% confidence		8.2% confidence		99.3 % confidence

Imagenet

Self-driving problem

We can easily fool the self-driving cars that a stop sign is a speed limit or others. (Eykholt et al. (2018))



That's why we need robustness for deep learning!

Fast gradient sign method(FGM): (Goodfellow et al.(2015))

$$Loss = \frac{1}{(\lambda + 1)k} \left(\sum_{i \in \text{Clean}} L(X_i|y_i) + \lambda \sum_{i \in \text{Adv}} L(X_i^{adv}|y_i) \right) \quad (4)$$

where k is the number of original data, $X^{adv} = X + \epsilon \text{sign}(\nabla_X L(X|y))$.

Robust optimization

The problem we want to solve is

$$\underset{\alpha, \beta, c}{\text{minimize}} \quad \sup_{P \in \mathbb{P}} \mathbb{E}_P[l(\theta; Z)] \quad (5)$$

where \mathbb{P} is a class of distributions around the data-generating distribution P_0 .

The expected loss

$$\mathbb{E}_{P_0}[l(\theta; Z)] \quad (6)$$

over a parameter $\theta \in \Theta$, where $Z \sim P_0$ is a distribution on a space \mathcal{Z} and l is a loss function.

Performs better than original method.

References

- ① https://en.wikipedia.org/wiki/Training,_test,_and_validation_sets
- ② [https://en.wikipedia.org/wiki/Norm_\(mathematics\)](https://en.wikipedia.org/wiki/Norm_(mathematics))
- ③ <https://en.wikipedia.org/wiki/ImageNet>
- ④ Goodfellow et al. "EXPLAINING AND HARNESSING ADVERSARIAL EXAMPLES" ICLR(2015)
- ⑤ Eykholt et al. "Robust Physical-World Attacks on Deep Learning Visual Classification" CVPR(2018)
- ⑥ Sinha et al. "Certifiable Distributional Robustness with Principled Adversarial Training"