Machine Learning (CSE 446): Probabilistic Approaches

Sham M Kakade

University of Washington
cse446-staff@cs.washington.edu

Announcements

- Midterm was challenging.
- HW3 posted today/tomo.

Probabilistic machine learning:

Probabilistic machine learning:

- **define a probabilistic model** relating random variables x to y
- **estimate its parameters**.

Maximum Likelihood Estimation and the Log loss

The principle of maximum likelihood estimation is to choose our parameters to make our observed data as likely as possible (under our model).

- Mathematically: find $\hat{\mathbf{w}}$ that maximizes the probability of the labels $y_1, \dots y_N$ given the inputs $x_1, \dots x_N$.
- The Maximum Likelihood Estimator (the **'MLE'**) is:

we define a mode

solve min prob

$$\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} \prod_{i=1}^{N} p_{\mathbf{w}}(y_i \mid \mathbf{x}_i)$$

$$= \operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{N} -\log p_{\mathbf{w}}(y_i \mid \mathbf{x}_i)$$

Linear Regression-MLE is same as Squared Loss Minimization!

- Linear regression defines $p_{\mathbf{w}}(Y \mid X)$ as follows:

$$p_{\mathbf{w}}(Y \mid \mathbf{x}) = \frac{1}{\sigma\sqrt{2\pi}} \exp - \frac{(Y - \mathbf{w} \cdot \mathbf{x})^2}{2\sigma^2}$$

  this is a modeling assumption.

- the MLE is then:

$$\operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{N} - \log p_{\mathbf{w}}(y_i \mid \mathbf{x}_i) \equiv \operatorname*{argmin}_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \underbrace{(y_i - \mathbf{w} \cdot \mathbf{x}_i)^2}_{SquaredLoss_i(\mathbf{w},b)}$$

A Probabilistic Model for Binary Classification: Logistic Regression

- For $Y \in \{-1, 1\}$ define $p_{\mathbf{w},b}(Y \mid X)$ as:
  1. Transform feature vector $\mathbf{x}$ via the "activation" function:

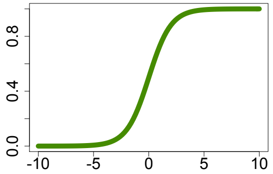
$$a = \mathbf{w} \cdot \mathbf{x} + b$$

  2. Transform a into a binomial probability by passing it through the logistic function:

$$p_{\mathbf{w},b}(Y = +1 \mid \mathbf{x}) = \frac{1}{1 + \exp[-a]} = \frac{1}{1 + \exp[-(\mathbf{w} \cdot \mathbf{x} + b)]}$$

$\exp(x) = e^{x}$

$P(Y=1|a)$



a

- If we learn $p_{\mathbf{w},b}(Y \mid \mathbf{x})$, we can (almost) do whatever we like!

The MLE for Logistic Regression

- the MLE for the logistic regression model:

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{N} -\log p_{\mathbf{w}}(y_i \mid \mathbf{x}_i) = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{N} \log\left(1 + \exp(-y_i \mathbf{w} \cdot \mathbf{x}_i)\right)$$

  for this expression we need $y \in \{-1, 1\}$

- This is the logistic loss function that we saw earlier.
- How do we compute the MLE?

Loss Minimization & Gradient Descent

$(y_i - w \cdot x_i)^2$ $\quad R(w) = \|w\|^2$

$\nabla_w \{ \}$

grad
w. respect to w

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \boxed{\frac{1}{N}\sum_{i=1}^{N} \underbrace{\ell(\mathbf{x}_i, y_i, \mathbf{w})}_{\ell_i(\mathbf{w})} + R(\mathbf{w})}$$

What is GD here?

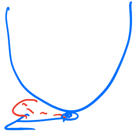
$$w \leftarrow w - \eta \left[\frac{1}{N} \sum_{i=1}^{N} \nabla_w \ell_i(w) \right]$$

use constant η

converges

What do we do if N is large?

Stochastic Gradient Descent (SGD): by example



$\nabla \ell_i(w)$

following something "noisy"

$$\underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2$$

- Gradient descent:

$$w \leftarrow w - \eta \left[\frac{-2}{N} \sum_{i=1}^{N} (y_i - w \cdot x_i) \vec{x}_i \right]$$

- Note we are computing an average. What is a crude way to estimate an average?
- Stochastic gradient descent: sample $i \sim$ uniformly $\{1, \cdots N\}$

$$w \leftarrow w - \eta \left[-2(y_i - w \cdot x_i) \vec{x}_i \right]$$

Will it converge?

Stochastic Gradient Descent (SGD): by example

$$\underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2$$

in expectation

we are moving in the correct direction

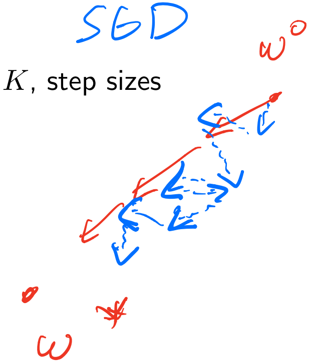
- Gradient descent:
- Note we are computing an average. What is a crude way to estimate an average?
- Stochastic gradient descent: $i \sim \mathrm{Uni}$

$$w \leftarrow w - \eta [-2(y_i - w \cdot x_i)] \vec{x}_i$$

Will it converge? **If the step size in SGD is a constant, we will not converge.**

we turn η down over time

Stochastic Gradient Descent (SGD) (without regularization) $\boxed{GD}$



Data: loss functions $\ell(\cdot)$, training data, number of iterations K, step sizes $\langle \eta^{(1)}, \ldots, \eta^{(K)} \rangle$
Result: parameters $\mathbf{w} \in \mathbb{R}^d$
initialize: $\mathbf{w}^{(0)} = \mathbf{0}$;
for $k \in \{1, \ldots, K\}$ **do**
  $i \sim \mathrm{Uniform}(\{1, \ldots, N\})$;
  $\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \eta^{(k)} \cdot \nabla_{\mathbf{w}} \ell_i(\mathbf{w}^{(k-1)})$;
end
return $\mathbf{w}^{(K)}$;

Algorithm 1: SGD

Stochastic Gradient Descent: Convergence

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \ell_i(\mathbf{w})$$

- $\mathbf{w}^{(k)}$: our parameter after k updates.
- Thm: Suppose $\ell(\cdot)$ is convex (and satisfies mild regularity conditions). There exists a way to decrease our step sizes $\eta^{(k)}$ over time so that our function value, $F(\mathbf{w}^{(k)})$ will converge to the minimal function value $F(\mathbf{w}^*)$.
- This Thm is different from GD in that **we need to turn down our step sizes over time!**

How to set learning rates:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \ell_i(\mathbf{w})$$

Theory:

Practice: How ~~do we turn η down~~?

- Initial η: start it "large"
  too large and things diverge (or are bad)
- Turning it down:
  1. sometimes we do not need to cut.
  2. "by hand": cut it down by some constant factor when we see the error doesn't drop any more.
  3. sometimes we tune the scheme by trying out different values.

"Early Stopping"

imposes some regularization

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{N}\sum_{i=1}^{N} \ell_i(\mathbf{w})$$

using $\lambda = 0$ is sometimes OK if you "stop early" [based on Dev set]

- How do we determine when to stop?
- Sometimes stopping early is itself a natural way to regularize.