Lecture 5 ML & Opt

HW I posted today
due 2 wks at 11:59pm

Agenda

- Bayes Law, MAP Estimation
- Regularization
- Risk Decomposition
- Framework for Optimization (Oracle Calls)
- Taylor Series & Local Models
- Convexity

MAP Estimation

Recall for MLE we have training data $\{(x_i, y_i)\}_{i=1}^n$ and we picked a model that we think can describe $y|x$, $P_\theta(y|x)$, then we learn the "best" θ by solving MLE

$$\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} -\log P_\theta(y_i|x_i)$$

Sometimes we have additional expectations about what the best θ should be, <u>independent</u> of our training data. To model these expectations, think of θ as a r.v.

$$\theta \sim p(\theta)$$

Think of maximizing the joint pdf of Θ and the training data

$$\hat{\Theta}_{MAP} = \underset{\Theta}{\operatorname{argmax}} \; p(\theta, y_1, \ldots, y_n | x_1, \ldots, x_n)$$

maximum a posteriori

$$= \underset{\Theta}{\operatorname{argmax}} \; p(\theta) p(y_1, \ldots, y_n | \theta, x_1, \ldots, x_n)$$

$$= \underset{\Theta}{\operatorname{argmax}} \; p(\theta) p_\theta(y_1, \ldots, y_n | x_1, \ldots, x_n)$$

$$= \underset{\Theta}{\operatorname{argmax}} \; p(\theta) \prod_{i=1}^{n} p_\theta(y_i | x_i)$$

$$= \underset{\Theta}{\operatorname{argmax}} \; p(\theta)^{1/n} \left[\prod_{i=1}^{n} p_\theta(y_i | x_i) \right]^{1/n}$$

$$= \underset{\theta}{\operatorname{argmax}} \; \frac{1}{n} \log p(\theta) + \frac{1}{n} \sum_{i=1}^{n} \log p_\theta(y_i | x_i)$$

$$= \underset{\theta}{\operatorname{argmin}} \; \underbrace{\frac{1}{n} \sum_{i=1}^{n} - \log p_\theta(y_i | x_i)}_{\text{data term}} \underbrace{- \frac{1}{n} \log p(\theta)}_{\text{regularization term}}$$

"model should explain data"

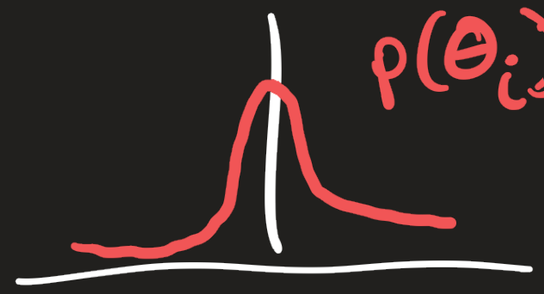
"model should look reasonable"

See as we have more training data, the influence of the regularization term goes down

Regularization

In ML we often don't have much knowledge of the "true" model for fitting our data, but some general expectations:

2-norm regularization — the model should assign little importance to any one feature (small weights)



$$p(\theta_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\theta_i)^2}{2\sigma^2}}$$

encourages θ_i to be close to 0

$$\Rightarrow p(\theta) = \frac{1}{(\sqrt{2\pi})^d \sigma^d} e^{-\frac{\sum_{i=1}^{d} \theta_i^2}{2\sigma^2}}$$

Recall $\sum_{i=1}^{d} \theta_i^2 = \theta^T\theta = \langle\theta,\theta\rangle = \|\theta\|_2^2$ ← 2-norm squared of θ

$$p(\theta) = \frac{1}{(\sqrt{2\pi}\sigma)^d} e^{-\frac{\|\theta\|_2^2}{2\sigma^2}}$$

$$\hat{\theta}_{MAP} = \underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i|x_i) - \frac{1}{n} \log p(\theta)$$

$$= \underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i|x_i)$$

$$- \frac{1}{n} \left[-d \log(\sqrt{2\pi}\sigma) - \frac{\|\theta\|_2^2}{2\sigma^2} \right]$$

$$= \underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i|x_i) + \underbrace{\frac{1}{2\sigma^2 n}}_{:= \lambda} \|\theta\|_2^2$$

$:= \lambda$, "regularization parameter"

$$= \underbrace{\underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i|x_i) + \lambda \|\theta\|_2^2}_{\text{A regularized ERM problem (RERM)}}$$

In practice: 1) fix a few values of λ to try

e.g. $\lambda \in \{0, 0.01, 0.1, 1, 10\}$

2) Fit $\hat{\theta}_{MAP,\lambda}$ for all these values

3) Evaluate the NLL of these $\hat{\theta}_{MAP,\lambda}$ on the validation data to determine which generalizes best

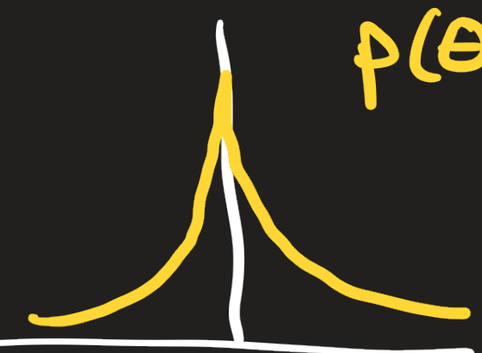
Regularization helps prevent overfitting to the training data set!

1-norm regularization (aka LASSO regularization) — Uses the regularizer $\|\theta\|_1$

encourages sparsity!

Corresponds to a sparsity prior on each coeff

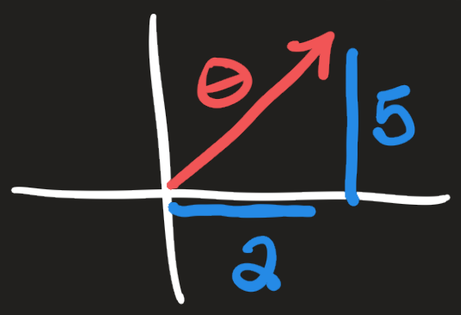
$$p(\theta_i) = \frac{1}{2\beta} e^{-\frac{|\theta_i|}{\beta}}$$



$$\Rightarrow$$

$$p(\theta) = \frac{1}{(2\beta)^d} e^{-\sum_{i=1}^{d} \frac{|\theta_i|}{\beta}}$$

Recall $\sum_{i=1}^{d} |\theta_i| \equiv \|\theta\|_1$ is the l_1-norm or 1-norm



$\|\theta\|_1 = 7$ $\|\theta\|_2 = \sqrt{2^2+5^2} = \sqrt{29}$

$$\hat{\theta}_{MAP} = \underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i | x_i) - \frac{1}{n} \log p(\theta)$$

$$\frac{1}{n} \log p(\theta) = \frac{1}{n} \log \left[\frac{1}{2b} e^{-\frac{\|\theta\|_1}{b}} \right]$$

$$= -\frac{1}{n} \log(2b) - \frac{\|\theta\|_1}{b}$$

$$= \underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i | x_i) + \underbrace{\frac{1}{bn}}_{:=\lambda} \|\theta\|_1$$

$$= \underset{\theta}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} -\log p_\theta(y_i | x_i) + \lambda \|\theta\|_1$$

Regularization is a powerful technique that is ubiquitous in ML. The general formulation as an optimization problem is

$$\hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \; R_n(f) + \lambda J(f) \qquad \text{(TERM)}$$

where $J : \mathcal{F} \to \mathbb{R}$ is the regularizer and $\lambda \geq 0$ is the regularization parameter

Risk decomposition for ML

- Bayes optimal model; minimizes the population risk

$$f^* = \underset{f}{\operatorname{argmin}} \; R(f) = \mathbb{E}_{(x,y)\sim p} \ell(f(x), y)$$

- Restrict our candidates to some plausible model class for a more tractable optim prob

$$f_{\mathcal{F}} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \; R(f)$$

- Use the empirical risk as a realizable estimator of the population risk

$$\hat{f}_{\mathcal{F}} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \; R_n(f)$$

- numerically solve the ERM using T iterations of some alg

$\hat{f}_T$

Ultimately we want to know how good our final model $\hat{f}_T$ is: how close is $R(\hat{f}_T)$ to $R(f^*)$

$$R(\hat{f}_T) = \underbrace{R(\hat{f}_T) - R(\hat{f})}_{\text{optimization error}} + \underbrace{R(\hat{f}) - R(f_{\mathcal{F}})}_{\text{generalization error}}$$

$$+ \underbrace{R(f_{\mathcal{F}}) - R(f^*)}_{\text{approximation error}} + R(f^*)$$

Why is the risk decomp relevant in this course?

1) **Approximation error** is controlled by choosing a good model class for the problem ;
address this in 2nd part of the course

2) **Optimization error** dictates how many iterations of my optim alg I need
(if max(**gen. error**, **approx. error**) = $\mathcal{O}(\epsilon)$
then it suffices to aim for an **optim error** = $\mathcal{O}(\epsilon)$)