Machine Learning (CSE 446): Probabilistic Approaches

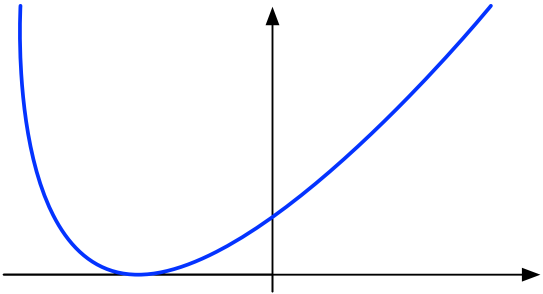
Sham M Kakade

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

Midterm Announcements

- Next Monday
- You may use a single side of a single sheet of handwritten notes that you prepared.

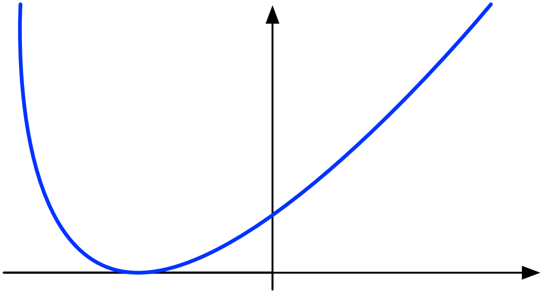
Remember: convexity



- A function $F(\cdot)$ is convex if for all $0 \leq t \leq 1$, w and w',

$$F((1-t)w+tw') \leq (1-t)F(w)+tF(w')$$

Gradient Descent



- Want to solve:

$$\min_{w} F(w)$$

- How should we update w?

Gradient Descent

Data: function $F : \mathbb{R}^d \to \mathbb{R}$, number of iterations K, step sizes $\eta^{(1)}, \ldots, \eta^{(K)}$
Result: $\mathbf{w} \in \mathbb{R}^d$
initialize: $\mathbf{w}^{(0)} = \mathbf{0}$;
for $k \in \{1, \ldots, K\}$ **do**
  $\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \eta^{(k)} \cdot \nabla F(\mathbf{w}^{(k-1)})$;
end
return $\mathbf{w}^{(K)}$;

Algorithm 1: GRADIENTDESCENT

Gradient Descent: Convergence

- Letting $\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} F(\mathbf{w})$ denote the global minimum
- Let $\mathbf{w}^{(k)}$ be our parameter after k updates.
- Thm: Suppose F is convex and "smooth". Using a **fixed step size** η (of appropriate length), we have:

$$F(\mathbf{w}^{(k)}) - F(\mathbf{w}^*) \leq O\left(\frac{1}{\cdot k}\right)$$

Gradient Descent: Simple example 1

- For $w \in \mathbb{R}$, $F(w) = \frac{1}{2}w^2$
- $w^* = \operatorname{argmin}_* F(w) = 0$
- $\frac{dF}{dw} = w$.
- The update:
$$w^{(k+1)} = w^{(k)} - \eta w^{(k)} = (1 - \eta)w^{(k)}$$
- Always use $\eta > 0$ (for GD)
- For $\eta \geq 2$, $w^{(k)}$ does not converge.
(diverges for η strictly above 2).
- For $|\eta| < 1$, $w^{(k)}$ converges to 0 (quickly!).
- For $|\eta| = 1$, $w^{(1)} = 0$.
This convergence in one step is 'lucky', due to being in 1dim.

Gradient Descent: Simple example 2

- For $w \in \mathbb{R}^2$, $F(w) = \frac{1}{2} w^\top \text{diag}(1,2) w = w_1^2 + 2w_2^2$
- $\mathbf{w}^* = \text{argmin}_{\mathbf{w}} F(\mathbf{w}) = 0$
- $\nabla F(w) = (w_1, 2w_2)^\top$.
- The update:
$$w^{(k+1)} = w^{(k)} - \eta w^{(k)}$$
- What happens here?

Gradient Descent: More formal statement

- Letting $\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} F(\mathbf{w})$ denote the global minimum
- Let $\mathbf{w}^{(k)}$ be our parameter after k updates.
- Thm: Suppose F is convex and "L-smooth". Using a **fixed step size** $\eta \leq \frac{1}{L}$, we have:

$$F(\mathbf{w}^{(k)}) - F(\mathbf{w}^*) \leq \frac{\|\mathbf{w}^{(0)} - \mathbf{w}^*\|^2}{\eta \cdot k}$$

- Smooth functions: for all w, w'

$$\|\nabla F(w) - \nabla F(w')\| \leq L\|w - w'\|$$

- Proof idea:
  1. If our gradient is large, we will make good progress decreasing our function value:
  2. If our gradient is small, we must have value near the optimal value:

Today

“Bayes Optimal” Decisions

- You have a task at hand. The Bayes Optimal decision rule is to do the best you possibly can given full knowledge of the true underlying probability distribution, $\mathcal{D}(x, y)$.
- The Bayes optimal classifier. $\mathcal{D}(x, y)$ is the true probability of (x, y).

$$f^{(\mathsf{BO})}(x) = \operatorname*{argmax}_{y} \mathcal{D}(y \mid x)$$

- Of course, we don't have $\mathcal{D}(y \mid x)$.

Probabilistic machine learning: **define a probabilistic model** relating random variables x to y and **estimate its parameters**.

Linear Regression as a Probabilistic Model

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

1. Observe the feature vector $\mathbf{x}$; transform it via the activation function:

$$\mu = \mathbf{w} \cdot \mathbf{x}$$

2. Let μ be the mean of a normal distribution and define the density:

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

3. Sample Y from $p_{\mathbf{w}}(Y \mid \mathbf{x})$.

Maximum Likelihood Estimation

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 $y_1, \ldots y_N$ given the inputs $x_1, \ldots x_N$.
- Note, by the i.i.d. assumption, for the $\mathcal{D}$ we have:

$$\mathcal{D}(y_1, \ldots y_N \mid \mathbf{x}_1, \ldots \mathbf{x}_N) = \prod_{i=1}^{N} \mathcal{D}(y_i | x_i)$$

- The Maximum Likelihood Estimator (the **'MLE'**) is:

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

Maximum Likelihood Estimation and the Log loss

- The 'MLE' is:

$$
\begin{aligned}
\hat{\mathbf{w}} &= \operatorname*{argmax}_{\mathbf{w}} \prod_{i=1}^{N} p_{\mathbf{w}}(y_i \mid \mathbf{x}_i) \\
&= \operatorname*{argmax}_{\mathbf{w}} \log \prod_{i=1}^{N} p_{\mathbf{w}}(y_i \mid \mathbf{x}_i) \\
&= \operatorname*{argmax}_{\mathbf{w}} \sum_{i=1}^{N} \log p_{\mathbf{w}}(y_i \mid \mathbf{x}_i) \\
&= \operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{N} -\log p_{\mathbf{w}}(y_i \mid \mathbf{x}_i)
\end{aligned}
$$

- This is referred to as the **log loss**.

Linear Regression-MLE is (Un-regularized) Squared Loss Minimization!

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

Where did the variance go?

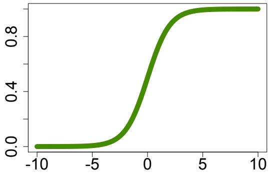
A Probabilistic Model for Binary Classification: Logistic Regression

- 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:

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



- If we learn $p_{\mathbf{w},b}(Y \mid \mathbf{x})$, we do more than just classification!