CSE 546: Machine Learning **Lecture 8**

Convexity and Gradient Descent

Instructor: Sham Kakade

1 Introduction: learning and generalization

One of the standard settings is that we have a loss function $\ell(f(x), y)$ where x is an input, y is an output, and $f(x)$ is our model. Here, we have a distribution D over x, y pairs; we obtain a training set:

$$\{(x_1, y_1), \dots (x_n, y_n)\}$$

sampled identically and independent distributed (i.i.d.) according to D; and we are interested in finding a map which as low expected loss under L, where:

$$L(f) = \mathbb{E}_{(x,y)\sim D}[\ell(f(x), y)]$$

where the expectation is under a sample from x, y.

For example, in binary classification we could have: a input x which is a d-dimensional real valued vector in R^d (e.g., representing information of an email); the output Y is a binary-valued number (whether the email is a spam or not), assume the binary values are $\{0, 1\}$; and we our model f maps x to $\{0, 1\}$. We could make a prediction of y based on the sign of $f_w(x)$. The quality measure is $\ell(f(x), y) = I(f(x) \neq y)$, where $I(\cdot)$ is the 0-1 valued indicator function, so that $\ell(f(x), y) = 0$ if $f(x) = y$ (prediction is correct), or $\ell(f(x), y) = 1$ if $f(x) \neq y$ (prediction is incorrect). This loss is called classification error or the $0/1$-loss.

1.1 Empirical Risk Minimization

Suppose we have some *hypothesis class* $\{f(x)\}\mathcal{F}$, which is a set of our possible predictors. Let f_* in this class which minimizes the loss $L(f)$. Our goal is find some model $\hat{f}$ which has loss which comparable to f_*.

The *empirical risk minimization* algorithm is to find the f which minimizes the empirical loss. In particular, this procedure is to find:

$$\hat{f} \in \arg\min_{\mathcal{F}} \frac{1}{n} \sum_i \ell(f(x_i), y_i)$$

Note that this is an optimization problem.

We measure the quality of $\hat{f}$ by the regret, where the regret of $\hat{f}$ is defined as:

$$L(\hat{f}) - L(f_*)$$

Ideally, we would like a loss function which has the following properties: accurately reflects what we are interested in and is easy to optimize. For these reasons, convexity plays a central role in machine learning and statistics.

2 Binary (or multi-class) prediction

Suppose the input X is a p-dimensional real valued vector in R^p (e.g., representing information of an email). The output Y is a binary-valued number (whether the email is a spam or not), assume the binary values are $\{0, 1\}$. The function class C consists of linear functions, parameterized by linear weight (coefficient) vector $w \in R^p$. That is, $f_w \in C$ as a linear function $f_w(x) = w^\top x$.

2.1 The $0 - 1$ loss

We could make a prediction of y based on the sign of $f_w(x)$. The quality measure is $\ell(f(x), y) = I(f(x) \neq y)$, where $I(\cdot)$ is the 0-1 valued indicator function, so that $\ell(f(x), y) = 0$ if $f(x) = y$ (prediction is correct), or $\ell(f(x), y) = 1$ if $f(x) \neq y$ (prediction is incorrect). This loss is called classification error or the $0/1$-loss.

2.2 The log loss

Often times in practice, we not only want to make correct predictions, we also desire to understand the confidence in our predictions (e.g. when detecting spam, we might want to adjust the threshold for deciding if an email is spam or not). In these settings, it is more natural to consider to a probabilistic model.

The most natural model here is the *logistic regression* model where:

$$\Pr(Y = 1|X, w) = \exp(w^\top X)/(1 + \exp(w^\top X))$$
$$\Pr(Y = -1|X, w) = 1/(1 + \exp(w^\top X))$$

This is an example of a *generalized linear model.*

Given some training set $(X_1, Y_1), \ldots (X_n, Y_n)$, the negative log likelihood is:

$$L(w) = \frac{-1}{n} \sum_i \log \Pr(Y_i|X_i, w) = \frac{-1}{n} \sum_i Y_i(w^\top X_i) - \log(1 + \exp(w^\top X_i))$$

The maximum likelihood estimator is the $\hat{w}$ which minimizes this loss.

3 Convexity and optimization

Suppose we seek to minimize a function $G : \mathbb{R}^d \to \mathbb{R}$:

$$\min_w G(w)$$

Gradient is the simplest of algorithms:

$$w_{t+1} = w_t - \eta_t \nabla G(w_t)$$

Note that if we are at a 0 gradient point, then we do not move. For this reason, gradient descent tends to be somewhat robust in practice.

Of particular interest are functions G which are *convex*. A function $G : \mathbb{R}^d \to \mathbb{R}$ is convex is convex if and only if:

$$G((1 - \alpha)w + \alpha w') \leq (1 - \alpha)G(w) + \alpha G(w')$$

where $\alpha \in [0, 1]$. If G is differentiable, we have that:

$$G(w') \geq G(w) + \nabla G(w) \cdot (w' - w) \, .$$

Convex functions are typically easy to optimize (easy int he polynomial time sense) due to that local search algorithms will result in finding global optimizers.

Note that it is a sensible idea even when f is not convex. Let us examine the convex case.

3.1 Example: Logistic Regression

Let us define:

$$\hat{Y}_w(x) = Pr(Y = 1|X, W)$$

which is just the conditional expectation of Y given X under our model.

We have that:

$$\nabla L(w) = \frac{-1}{n} \sum_i \nabla \log \Pr(Y_i|X_i, w) = \frac{-1}{n} \sum_i (Y_i - \hat{Y}_w(X_i)) X_i$$