Chris Piech
CS109

Handout #40
May 20th, 2016

Logistic Regression

Before we get started I wanted to familiarize you with some notation:

$$\theta^T\mathbf{x} = \sum_{i=1}^{n} \theta_i x_i = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \qquad \text{weighted sum}$$

$$\sigma(z) = \frac{1}{1+e^{-z}} \qquad \text{sigmoid function}$$

Logistic Regression Overview

Classification is the task of choosing a value of y that maximizes $P(Y|X)$. Naïve Bayes worked by approximating that probability using the naïve assumption that each feature was independent given the class label.

For all classification algorithms you are given n I.I.D. training datapoints $(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots (\mathbf{x}^{(n)}, y^{(n)})$ where each "feature" vector $\mathbf{x}^{(i)}$ has $m = |\mathbf{x}^{(i)}|$ features.

Logistic Regression Assumption

Logistic Regression is a classification algorithm (I know, terrible name) that works by trying to learn a function that approximates $P(Y|X)$ as a function of the inputs. It makes the central assumption that the probability can be approximated as a sigmoid function applied to a linear combination of input features. Mathematically, for a single training datapoint $(\mathbf{x}, y)$ Logistic Regression assumes:

$$P(Y=1|\mathbf{X}=\mathbf{x}) = \sigma(z) \text{ where } z = \theta_0 + \sum_{i=1}^{m} \theta_i x_i$$

This assumption is often written as the following functions for $Y = 1$ and $Y = 0$:

$$P(Y=1|\mathbf{X}=\mathbf{x}) = \sigma(\theta^T\mathbf{x}) \qquad \text{where we always set } x_0 \text{ to be 1}$$

$$P(Y=0|\mathbf{X}=\mathbf{x}) = 1 - \sigma(\theta^T\mathbf{x}) \qquad \text{by total law of probability}$$

Using these equations for probability of $Y|X$ we can create an algorithm that select values of theta that maximize that probability for all data. I am first going to state the log probability function and partial derivatives with respect to theta. Then later we will (a) show an algorithm that can chose optimal values of theta and (b) show how the equations were derived.

Log Likelihood

Using the Logistic Regression Assumption, we can write an equation for the likelihood of all the data (under that assumption). If you take the log of that equation the result is:

$$LL(\theta) = \sum_{i=0}^{n} y^{(i)} \log \sigma(\theta^T\mathbf{x}^{(i)}) + (1-y^{(i)}) \log[1-\sigma(\theta^T\mathbf{x}^{(i)})]$$

We will show the intermediate steps later

Gradient of Log Likelihood

Now that we have a function for likelihood, we simply need to chose the values of theta that maximize it. Unlike it other questions, there is no closed form way to calculate theta. Instead we chose it using optimization. The first step is to calculate the partial derivative of log-likelihood with respect to each parameter:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=0}^{n} \left[y^{(i)} - \sigma(\theta^T\mathbf{x}^{(i)}) \right] x_j^{(i)}$$

Gradient Ascent Optimization

Typically once we have an equation for Log Likelihood, we chose the values for our parameters (θ) that maximize that function. In the case of logistic regression we can't solve this mathematically. Instead we use a computer to chose θ. To do so we employ an algorithm called gradient ascent. That algorithms claims that if you continuously take small steps in the direction of your gradient, you will eventually make it to a local maxima. In the case of Logistic Regression you can prove that the result will always be a global maxima.

The small step that we continually take given the training dataset can be calculated as:

$$\begin{aligned}
\theta_j^{\text{new}} &= \theta_j^{\text{old}} + \eta \cdot \frac{\partial LL(\theta^{\text{ old}})}{\partial \theta_j^{\text{old}}} \\
&= \theta_j^{\text{old}} + \eta \cdot \sum_{i=0}^{n} \left[y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)}) \right] x_j^{(i)}
\end{aligned}$$

Where η is the magnitude of the step size that we take. If you keep updating θ using the equation above you will converge on the best values of θ!

Derivations

In the following two sections we provide the mathematical derivations for the log-likelihood and the gradient. They are worth knowing because these ideas are heavily used in Neural Networks. To start, here is a super slick way of writing the probability of one datapoint:

$$P(Y = y|X = \mathbf{x}) = \sigma(\theta^T \mathbf{x})^y \cdot \left[1 - \sigma(\theta^T \mathbf{x})\right]^{(1-y)}$$

Since each datapoint is independent, the probability of all the data is:

$$\begin{aligned}
L(\theta) =& \prod_{i=1}^{n} P(Y = y^{(i)} | X = \mathbf{x}^{(i)}) \\
=& \prod_{i=1}^{n} \sigma(\theta^T \mathbf{x}^{(i)})^{y^{(i)}} \cdot \left[1 - \sigma(\theta^T \mathbf{x}^{(i)})\right]^{(1-y^{(i)})}
\end{aligned}$$

And if you take the log of this function, you get the reported Log Likelihood for Logistic Regression.

The next step is to calculate the derivative of the log likelihood with respect to each theta. To start, here is the definition for the derivative of sigma with respect to its inputs:

$$\frac{\partial}{\partial z}\sigma(z) = \sigma(z)[1 - \sigma(z)] \qquad \text{to get the derivative with respect to } \theta\text{, use the chain rule}$$

Derivative of gradient for one datapoint $(\mathbf{x}, y)$:

$$\begin{aligned}
\frac{\partial LL(\theta)}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} y \log \sigma(\theta^T \mathbf{x}) + \frac{\partial}{\partial \theta_j}(1-y)\log[1 - \sigma(\theta^T \mathbf{x}] && \text{derivative of sum of terms} \\
&= \left[\frac{y}{\sigma(\theta^T x)} - \frac{1-y}{1-\sigma(\theta^T x)} \right] \frac{\partial}{\partial \theta_j}\sigma(\theta^T x) && \text{derivative of log } f(x) \\
&= \left[\frac{y}{\sigma(\theta^T x)} - \frac{1-y}{1-\sigma(\theta^T x)} \right] \sigma(\theta^T x)[1 - \sigma(\theta^T x)]x_j && \text{chain rule + derivative of sigma} \\
&= \left[\frac{y - \sigma(\theta^T x)}{\sigma(\theta^T x)[1 - \sigma(\theta^T x)]} \right] \sigma(\theta^T x)[1 - \sigma(\theta^T x)]x_j && \text{algebraic manipulation} \\
&= \left[y - \sigma(\theta^T x) \right] x_j && \text{cancelling terms}
\end{aligned}$$

Because the derivative of sums is the sum of derivatives, the gradient of theta is simply the sum of this term for each training datapoint.