STATS 202: Data Mining and Analysis

Instructor: Linh Tran

Logistic regression

Stanford University

Let $y \in \{0, 1\}$ and $x \in \mathbb{R}$. We can get an estimate of $\mathbb{E}_0[Y|X]$ using a logistic model, e.g.

$$\mathbb{E}[Y|X] = logit^{-1}(\beta_0 + \beta_1 x_1) : logit^{-1}(z) = \frac{1}{1 + e^{-z}} \tag{1}$$

Unlike linear regression, this model has no closed form solution, preventing us from directly calculating our model's coefficients $\beta = (\beta_0, \beta_1)$. Instead, a common approach used for estimating our coefficients is gradient descent. This process is outlined as follows:

1. Make an initial guess of β, denoted $\beta^{(0)}$.
2. Iteratively for $t = 1, 2, \ldots$ until convergence:
   a. Calculate the gradient (of our chosen loss function) with respect to our parameters (β).
   b. Update our $\beta^{(t)}$ via
   $$\beta^{(t+1)} = \beta^{(t)} - \gamma \nabla L(\beta) \tag{2}$$
   where γ is a learning rate that specifies how quickly we update our parameter estimates and $\nabla L(\beta)$ is the gradient of our chosen loss function that we would like to minimize.

Commmonly, in logistic regression, we will try to maximize the likelihood, i.e. the probability of observing our data under our model's parameters. Under our model, the likelihood is a concave function with a unique maximum point that will correspond to our parameter estimates $\hat{\beta}_n$. Figure 1 shows the log-likelihood under various values of β under the model $logit^{-1}(\beta x)$, along with the value that acheives the maximum of the likelihood function. Note that we typically take the log-likelihood (rather than simply the likelihood) as the resulting summations make the computation easier (than taking the cumulative products).

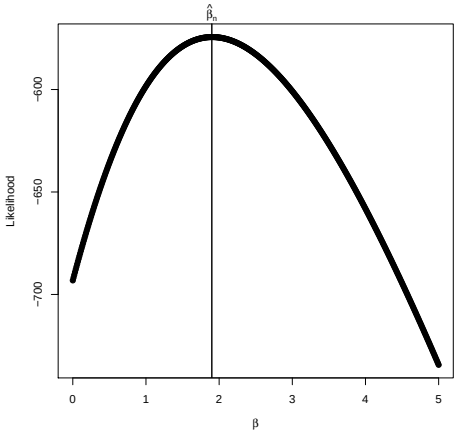

Figure 1: Log-likelihood under different values of x. $\hat{\beta}_n$ is the point at which the log-likelihood is maximized.

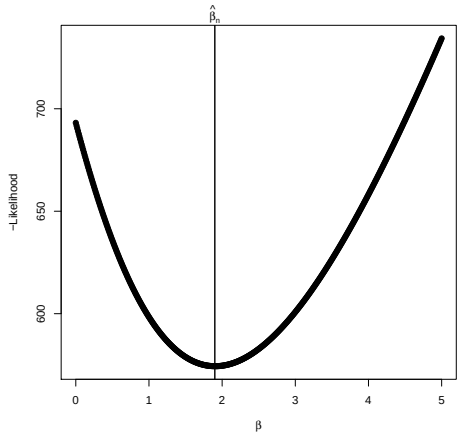

Figure 2: Negative log-likelihood under different values of x. $\hat{\beta}_n$ is the point at which the negative log-likelihood is minimized.

Note that, in general, gradient descent is used with loss functions that we wish to minimize (rather than maximize, as we are doing in the maximum likelihood estimation approach). To conform to this convention, we simply take the negative of the likelihood, as shown in Figure 2.

Let

$$L(\beta) = -\left[y \log(p) + (1-y)\log(1-p)\right], \text{ where} \tag{3}$$

$$p = \frac{1}{1+\exp(-Z)} \tag{4}$$

$$Z = \mathbf{X}\beta \tag{5}$$

$$\tag{6}$$

where, for notational simplicity, we define $\mathbf{X}_i \triangleq (1, X)$. We now derive $\nabla L(\beta)$. By the chain rule, we have that

$$\nabla L(\beta) = \frac{\partial L(\beta)}{\partial \beta} \tag{7}$$

$$= \frac{\partial L(\beta)}{\partial p}\frac{\partial p}{\partial Z}\frac{\partial Z}{\partial \beta} \tag{8}$$

$$\tag{9}$$

Thus, we can get $\nabla L(\beta)$ by calculating each of the three partial derivatives and taking the product. We derive each below.

$$
\begin{aligned}
\frac{\partial L(\beta)}{\partial p} &= \frac{\partial}{\partial p} - [y\log(p) + (1-y)\log(1-p)] && (10)\\
&= -\frac{y}{p} + \frac{(1-y)}{1-p} && (11)\\
\frac{\partial p}{\partial Z} &= \frac{\partial}{\partial Z}\frac{1}{1+\exp(-Z)} && (12)\\
&= -\frac{1}{(1+e^{-Z})^2} \times \frac{\partial p}{\partial Z}(1+e^{-Z}) && (13)\\
&= -\frac{1}{(1+e^{-Z})^2} \times (-e^{-Z}) && (14)\\
&= \frac{e^{-Z}}{(1+e^{-Z})^2} && (15)\\
&= \frac{1}{(1+e^{-Z})} \times \frac{e^{-Z}}{(1+e^{-Z})} && (16)\\
&= \frac{1}{(1+e^{-Z})} \times \frac{1+e^{-Z}-1}{(1+e^{-Z})} && (17)\\
&= \frac{1}{(1+e^{-Z})} \times \left[1 - \frac{1}{(1+e^{-Z})}\right] && (18)\\
&= p(1-p) && (19)\\
\frac{\partial Z}{\partial \beta} &= \frac{\partial}{\partial \beta}\mathbf{X}\beta && (20)\\
&= \mathbf{X} && (21)
\end{aligned}
$$

Combining these, we have

$$
\begin{aligned}
\nabla L(\beta) &= \frac{\partial L(\beta)}{\partial \beta} && (22)\\
&= \frac{\partial L(\beta)}{\partial p}\frac{\partial p}{\partial Z}\frac{\partial Z}{\partial \beta} && (23)\\
&= \left[-\frac{y}{p} + \frac{(1-y)}{1-p}\right][p(1-p)]\,[\mathbf{X}] && (24)\\
&= (p-y)\mathbf{X} && (25)
\end{aligned}
$$

We can apply this to our example to better understand how gradient descent works. Let $\gamma = 0.001$ and our initial estimate $\beta_1^{(0)} = 0$. Furthermore, (for ease of presentation) assume that the model has no intercept, i.e. $\beta_0 = 0$. Figure 3 shows the result of applying gradient descent.

```r
  gamma <- 0.001
  beta <- 0

  beta.diff = Inf
  beta.ests <- c()
  losses <- c()
  while(beta.diff>0.01) {
    p <- 1 / (1 + exp(-(x*beta)))
    nabla.L <- sum((p-y) * x)
    beta.update <- (gamma * nabla.L)
    beta <- beta - beta.update
    beta.diff <- sum(abs(beta.update))
    beta.ests <- append(beta.ests, beta)
    losses <- append(losses, -CalculateLikelihood(beta, y, x))
  }

  # Our final parameter estimate
  print (beta)
```

```
## [1] 1.730828
```

```r
  # Plot gradient descent results
  sfun0 <- stepfun(beta.ests[-1], losses, f = 0)
  plot(-likelihoods ~ betas, pch=19, ylab="-Likelihood", xlab=expression(beta))
  lines(sfun0, col=2)
```

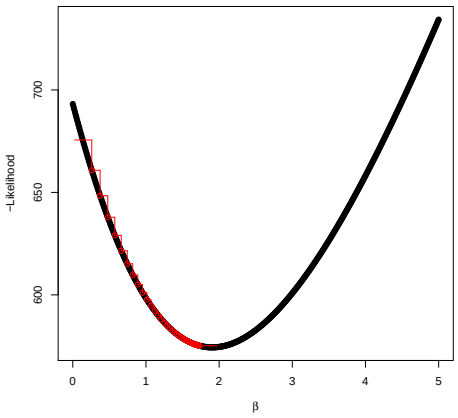

Figure 3: Negative log-likelihood under different values of x, along with estimates $\hat{\beta}_n$ from each iteration of gradient descent (in red).