CS 189/289A Introduction to Machine Learning
Fall 2024 Jennifer Listgarten, Saeed Saremi DIS3

# 1 Logistic Regression

Assume that we have $n$ i.i.d. data points $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$, where each $y_i$ is a binary label in $\{0, 1\}$. We model the posterior probability as a Bernoulli distribution and the probability for each class is the sigmoid function, i.e., $p(y|\mathbf{x};\mathbf{w}) = q^y(1-q)^{1-y}$, where $q = s(\mathbf{w}^\top \mathbf{x})$ and $s(\zeta) = \frac{1}{1+e^{-\zeta}}$ is the sigmoid function.

(a) Show that for a given data point $\mathbf{x}$, the log ratio of the conditional probabilities, or *log odds*, is linear in $\mathbf{x}$. More specifically, show that

$$\log \frac{p(y=1 \mid \mathbf{x};\mathbf{w})}{p(y=0 \mid \mathbf{x};\mathbf{w})} = \mathbf{w}^\top \mathbf{x}.$$

**Solution:**

$$\begin{aligned}\log \frac{p(y=1 \mid \mathbf{x};\mathbf{w})}{p(y=0 \mid \mathbf{x};\mathbf{w})} &= \log \frac{q}{1-q} \\ &= \log \frac{\frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}}}}{\frac{e^{-\mathbf{w}^\top \mathbf{x}}}{1+e^{-\mathbf{w}^\top \mathbf{x}}}} \\ &= \log \frac{1}{e^{-\mathbf{w}^\top \mathbf{x}}} \\ &= \mathbf{w}^\top \mathbf{x}\end{aligned}$$

(b) Write out the likelihood and log likelihood functions for the $n$ data points.

**Solution:** The likelihood is:

$$L(\mathbf{w}) = \prod_{i=1}^{n} p(y = y_i|\mathbf{x}_i) = \prod_{i=1}^{n} q_i^{y_i}(1-q_i)^{1-y_i}.$$

The log likelihood is:

$$l(\mathbf{w}) = \sum_{i=1}^{n} y_i \log(q_i) + (1-y_i)\log(1-q_i)$$

(c) Show that finding maximum likelihood estimate of $\mathbf{w}$ is equivalent to the following optimization problem:

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left[ \sum_{i=1}^{n} (1 - y_i)\mathbf{w}^\top \mathbf{x}_i + \log\Big(1 + \exp\big\{-\mathbf{w}^\top \mathbf{x}_i\big\}\Big) \right]$$

**Solution:** Now, we step through minimizing the negative log likelihood of the training data as a function of the parameters $\mathbf{w}$:

$$\begin{aligned}
\hat{\mathbf{w}} &= \left[ \underset{\mathbf{w}}{\operatorname{argmin}} - \sum_{i=1}^{n} y_i \log(q_i) + (1 - y_i)\log(1 - q_i) \right] \\
&= \left[ \underset{\mathbf{w}}{\operatorname{argmin}} - \sum_{i=1}^{n} y_i \log\left( \frac{q_i}{1 - q_i} \right) + \log(1 - q_i) \right]
\end{aligned}$$

From the first question, we get that $y_i \log\left(\frac{q_i}{1-q_i}\right) = y_i \mathbf{w}^\top \mathbf{x}_i$. The second term of the sum can be simplified as follows:

$$\begin{aligned}
\log(1 - q_i) &= \log\left( \frac{1 + \exp(-\mathbf{w}^\top \mathbf{x}_i) - 1}{1 + \exp(-\mathbf{w}^\top \mathbf{x}_i)} \right) \\
&= \log\left( \frac{\exp(-\mathbf{w}^\top \mathbf{x}_i)}{1 + \exp(-\mathbf{w}^\top \mathbf{x}_i)} \right) \\
&= -\mathbf{w}^\top \mathbf{x}_i - \log\Big(1 + \exp\big(-\mathbf{w}^\top \mathbf{x}_i\big)\Big).
\end{aligned}$$

Plugging in, we get that

$$\begin{aligned}
\hat{\mathbf{w}} &= \underset{\mathbf{w}}{\operatorname{argmin}} \left[ -\sum_{i=1}^{n} y_i \mathbf{w}^\top \mathbf{x}_i - \mathbf{w}^\top \mathbf{x}_i - \log\Big(1 + \exp\{-\mathbf{w}^\top \mathbf{x}_i\}\Big) \right] \\
&= \underset{\mathbf{w}}{\operatorname{argmin}} \left[ \sum_{i=1}^{n} (1 - y_i)\mathbf{w}^\top \mathbf{x}_i + \log\Big(1 + \exp\big\{-\mathbf{w}^\top \mathbf{x}_i\big\}\Big) \right].
\end{aligned}$$

(d) Comment on whether it is possible to find a closed form maximum likelihood estimate of $\mathbf{w}$, and describe an alternate approach.

**Solution:** Let us denote $J(\mathbf{w}) = \sum_{i=1}^{n} (1 - y_i)\mathbf{w}^\top \mathbf{x}_i + \log\Big(1 + \exp\{-\mathbf{w}^\top \mathbf{x}_i\}\Big)$. Notice that $J(\mathbf{w})$ is convex in $\mathbf{w}$, so global minimum can be found. Note that $s'(\zeta) = s(\zeta)(1 - s(\zeta))$. Now let us take the gradient of $J(\mathbf{w})$ w.r.t $\mathbf{w}$:

$$\nabla_w J = \sum_{i=1}^{n} (1 - y_i)\mathbf{x}_i - \frac{\exp\{-\mathbf{w}^\top \mathbf{x}_i\}}{1 + \exp\{-\mathbf{w}^\top \mathbf{x}_i\}} \mathbf{x}_i = \sum_{i=1}^{n} (-1 + s(\mathbf{w}^\top \mathbf{x}_i) - y_i + 1)\mathbf{x}_i = \sum_{i=1}^{n} (s_i - y_i)\mathbf{x}_i = \mathbf{X}^\top (\mathbf{s} - \mathbf{y})$$

where, $s_i = s(\mathbf{w}^\top \mathbf{x}_i), \mathbf{s} = (s_1, \ldots, s_n)^\top, \mathbf{y} = (y_1, \ldots, y_n)^\top$ and $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix}$.

Unfortunately, we can't get a closed form estimate for $\mathbf{w}$ by setting the derivative to zero, given that the term $\mathbf{s}$ still contains $\mathbf{w}$, and further-order derivatives will continue to carry expressions over $\mathbf{w}$. However, the convexity of this problem allows for first-order optimization algorithms, such as gradient descent, to converge to a global minimum.

## 2 Gaussian Classification

Let $P(x \mid \omega_i) \sim \mathcal{N}(\mu_i, \sigma^2)$ for a two-category, one-dimensional classification problem with classes $\omega_1$ and $\omega_2$, $P(\omega_1) = P(\omega_2) = 1/2$, and $\mu_2 > \mu_1$.

A common procedure to classify a data point $x$ is to assign it to the class $\omega_i$ with the highest posterior probability $P(\omega_i \mid x)$. We will justify this choice later in class.

In this problem, the *decision boundary* is the line that separates the two classes, that is, the line where the posterior probabilities $P(\omega_1 \mid x)$ and $P(\omega_2 \mid x)$ are equal.

(a) Find the optimal decision boundary and the corresponding decision rule.

**Solution:**

$$\begin{array}{rcll} P(\omega_1 \mid x) & = & P(\omega_2 \mid x) & \Leftrightarrow \\ P(x \mid \omega_1)\frac{P(\omega_1)}{P(x)} & = & P(x \mid \omega_2)\frac{P(\omega_2)}{P(x)} & \Leftrightarrow \\ P(x \mid \omega_1) & = & P(x \mid \omega_2) & \Leftrightarrow \\ \mathcal{N}(\mu_1, \sigma^2) & = & \mathcal{N}(\mu_2, \sigma^2) & \Leftrightarrow \\ (x - \mu_1)^2 & = & (x - \mu_2)^2 & \end{array}$$

This yields the Bayes decision boundary: $x = \frac{\mu_1+\mu_2}{2}$.

The corresponding decision rule is, given a data point $x \in \mathbb{R}$:

- if $x < \frac{\mu_1+\mu_2}{2}$, then classify $x$ in class 1
- otherwise, classify $x$ in class 2

Note that this is the centroid method.

(b) The probability of misclassification (error rate) is:

$$P_e = P((\text{misclassified as } \omega_1) \mid \omega_2)\, P(\omega_2) + P((\text{misclassified as } \omega_2) \mid \omega_1)\, P(\omega_1).$$

Show that the probability of misclassification (error rate) associated with this decision rule is

$$P_e = \frac{1}{\sqrt{2\pi}} \int_a^\infty e^{-z^2/2} dz, \quad \text{where} \quad a = \frac{\mu_2 - \mu_1}{2\sigma}.$$

**Solution:** We use the change of variables $z = \frac{x-\mu_2}{\sigma}$, so that $dz = \frac{1}{\sigma}dx$.

$$\begin{aligned} P((\text{misclassified as } \omega_1) \mid \omega_2) &= \int_{-\infty}^{\frac{\mu_1+\mu_2}{2}} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu_2)^2}{2\sigma^2}} dx \\ &= \int_{-\infty}^{-(\mu_2-\mu_1)/2\sigma} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz \\ &= \frac{1}{\sqrt{2\pi}} \int_a^{+\infty} e^{-\frac{z^2}{2}} dz \\ &= P_e, \end{aligned}$$

Similarly, we use the change of variables $y = \frac{x-\mu_1}{\sigma}$ and $dy = \frac{1}{\sigma}dx$.

$$\begin{aligned} P((\text{misclassified as } \omega_2) \mid \omega_1) &= \int_{\frac{\mu_1+\mu_2}{2}}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu_1)^2}{2\sigma^2}} dx \\ &= \int_{(\mu_2-\mu_1)/2\sigma}^{+\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy \end{aligned}$$

Therefore:

$$P((\text{misclassified as } \omega_1) \mid \omega_2)P(\omega_2) + P((\text{misclassified as } \omega_2) \mid \omega_1)P(\omega_1) = P_e \cdot \frac{1}{2} + P_e \cdot \frac{1}{2} = P_e$$

(c) What is the limit of $P_e$ as $\sigma$ goes to 0?

**Solution:** As $\sigma$ goes to 0, $a$ goes to $\infty$, so the integral $P_e$ goes to 0.

## 3 Softmax Regression

Logistic regression directly models the probability of a point $x$ belonging to class C as $P(Y = \mathrm{C}|X = x) = s(w^\top x)$ where $s$ is the sigmoid function $s(\gamma) = 1/(1 + e^{-\gamma})$. (If you want a bias term $\alpha$, assume it is included as the last component of $w$, and each data point $x$ has a 1 appended.) It is limited to binary classification problems. While logistic regression can be applied to multi-class classification with many-to-one or one-to-one approaches, there is a more elegant method.

Rather than modeling only $P(Y = \mathrm{C}|X = x)$, softmax regression models the entire categorical distribution over $k$ classes, $P(Y = 1|X = x), P(Y = 2|X = x), \dots, P(Y = k|X = x)$. It does so by learning a linear model with weight vector $w_i$ for each of the $k$ classes and turning them into probabilities with the softmax function, $s_i(z) = e^{-z_i}/(\sum_{j=1}^{k} e^{-z_j})$, giving the predictions

$$P(Y = i|X = x) = \frac{e^{-w_i^\top x}}{\sum_{j=1}^{k} e^{-w_j^\top x}}.$$

The idea is that class $i$'s probability is proportional to $e^{-w_i^\top x}$. To make all the probabilities sum to 1, the denominator is the sum of the numerators.

**Note:** For future reference, the most general form of the softmax function is $s_i(z) = e^{\beta z_i}/(\sum_{j=1}^{k} e^{\beta z_j})$, where $\beta \in [-\infty, \infty]$ is a user-defined hyperparameter. If $\beta$ is negative, smaller inputs are mapped to larger outputs. If $\beta$ is positive, larger inputs are mapped to larger outputs. Here, we only consider $\beta = -1$.

(a) Show that where $k = 2$, the softmax regression function has the same form as the logistic regression function.

**Solution:**

$$P(Y = 1|X = x) = \frac{e^{-w_1^\top x}}{e^{-w_1^\top x} + e^{-w_2^\top x}} = \frac{1}{1 + e^{-(w_2 - w_1)^\top x}}.$$

(b) In the default form we gave above, softmax regression is overparameterized—there are more parameters than needed for the model. This should be evident in your answer to part (a). Reformulate softmax regression so it requires fewer parameters.

**Solution:** We can divide out by one of the classes to remove it from the equation.

$$P(Y = k|X = x) = \frac{e^{-w_k^\top x}}{\sum_{j=1}^{k} e^{-w_j^\top x}} = \frac{1}{1 + \sum_{j=1}^{k-1} e^{-(w_j - w_k)^\top x}} = \frac{1}{1 + \sum_{j=1}^{k-1} e^{-\alpha_j^\top x}},$$

where each $\alpha_j = w_j - w_k$ is a vector of weights for classes 1 through $k-1$. In this formulation, the weights in the vector $\alpha_j$ are implicitly comparing class $j$ to the last class, $k$. We don't need an $\alpha_k$ (which would be zero).

(c) Recall the logistic (aka binary cross-entropy) loss function (applied to a single training point),

$$L(\hat{y}, y) = -y \log \hat{y} - (1 - y) \log(1 - \hat{y}).$$

How would you design the analogous loss function $L(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_k, y_1, y_2, \ldots, y_k)$ for softmax regression? Assume that for each training point, the input includes $k$ class labels in $[0, 1]$ satisfying $\sum_{i=1}^{k} y_i = 1$, and the $k$ class predictions are $\hat{y}_i = P(Y = i|X = x)$ (which also sum to 1 because we forced them to sum to 1).

**Solution:** The generalization is called the cross-entropy loss,

$$L(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_k, y_1, y_2, \ldots, y_k) = \sum_{i=1}^{k} -y_i \log \hat{y}_i.$$