Maximum Likelihood Estimation, Logistic Regression

Course: Data Mining

Professor: Dr. Tahaei

Herbod Pourali, Ilya Farhangfar

Subject: Logistic Regression Problem and Solution

December 2024

Problem 1: Logistic Regression. Given a dataset $\{(x^n, y^n), n = 1, \ldots, N\}$, where $y^n \in \{0, 1\}$, logistic regression uses the model $p(y^n = 1|x^n) = \sigma(w^\top x^n + b)$. Assuming that the data is drawn independently and identically, show that the derivative of the log likelihood $\mathcal{L}$ of the data is

$$\nabla_w \mathcal{L} = \sum_{n=1}^{N} \big(y^n - \sigma(w^\top x^n + b)\big)\, x^n.$$

HINT: show that

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z)).$$

Solution: $\mathcal{L}(w, b) = \sum_{n=1}^{N} y^n \log \sigma(b + w^\top x^n) + (1 - y^n) \log(1 - \sigma(b + w^\top x^n))$
Using $\nabla_w \sigma(y) = (1 - \sigma(y))\sigma(y)\nabla_w y$

$$\begin{aligned}
\nabla_w \mathcal{L}(w, b) &= \sum_{n=1}^{N} \frac{y^n \nabla_w \sigma(\cdot)}{\sigma(\cdot)} + \frac{\nabla_w (1 - \sigma(\cdot))}{1 - \sigma(\cdot)} - \frac{y^n \nabla_w (1 - \sigma(\cdot))}{1 - \sigma(\cdot)}. \\
&= \sum_{n=1}^{N} (y^n (1 - \sigma(\cdot)) x^n - \sigma(\cdot) x^n + y^n \sigma(\cdot) x^n) \\
&= \sum_{n=1}^{N} (y^n x^n - y^n \sigma(\cdot) x^n - \sigma(\cdot) x^n + y^n \sigma(\cdot) x^n) \\
&= \sum_{n=1}^{N} \big(y^n - \sigma(w^\top x^n + b)\big)\, x^n
\end{aligned}$$

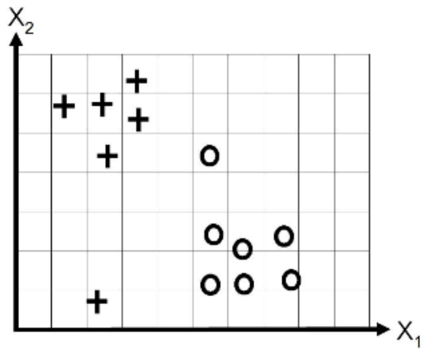
Problem 2. Consider a dataset $\{(x_n, y_n), n = 1, \ldots, N\}$, where $y_n \in \{0, 1\}$, and x_n is a D-dimensional vector.

(a) Data is linearly separable if the two classes can be completely separated by a hyperplane. Show that if the training data is linearly separable with the hyperplane $w^T x + b$, the data is also separable with the hyperplane $\tilde{w}^T x + \tilde{b}$, where $\tilde{w} = \lambda w$, $\tilde{b} = \lambda b$ for any scalar $\lambda > 0$.

(b) What consequence does the above result have for maximum likelihood training of logistic regression for linearly separable data?

Solution: The hyperplane $\tilde{b} + \tilde{w}^T x = \lambda b + \lambda w^T x \Rightarrow \lambda(b + w^T x) = 0$ is geometrically the same as $b + w^T x = 0$. If the data is linearly separable, the weights will continue to increase during the maximum likelihood training, and the classifications will become extreme (i.e., predictive probabilities of 0 or 1).

Problem 3. Consider the following data set



(a) Suppose that we fit a logistic regression model, i.e., $p(y = 1|x, w) = \sigma(w_0 + w_1 x_1 + w_2 x_2)$. Suppose we fit the model by maximum likelihood, i.e., we minimize

$$J(w) = -l(w, D_{\text{train}}),$$

where $-l$ is the logarithm of the likelihood above. Suppose we obtain the parameters $\hat{w}$. Sketch a possible decision boundary corresponding to $\hat{w}$.

Is your answer unique? How many classification errors does your method make on the training set?

(b) Now suppose that we regularize only the w_0 parameter, i.e., we minimize

$$J_0(w) = -l(w, D_{\text{train}}) + \lambda w_0^2.$$

Suppose λ is a very large number, so we regularize w_0 all the way to 0, but all other parameters are unregularized. Sketch a possible decision boundary. How many classification errors does your method make on the training set?

Hint: Consider the behavior of simple linear regression, $w_0 + w_1 x_1 + w_2 x_2$ *when* $x_1 = x_2 = 0$.

(c) Now suppose that we regularize only the w_1 parameter, i.e., we minimize

$$J_1(w) = -l(w, D_{\text{train}}) + \lambda w_1^2.$$

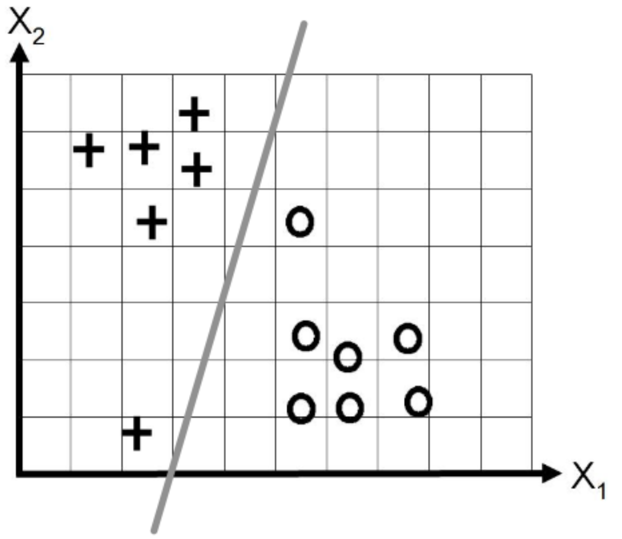
Again suppose λ is a very large number. Sketch a possible decision boundary. How many classification errors does your method make on the training set?

(d) Now suppose that we regularize only the w_2 parameter, i.e., we minimize

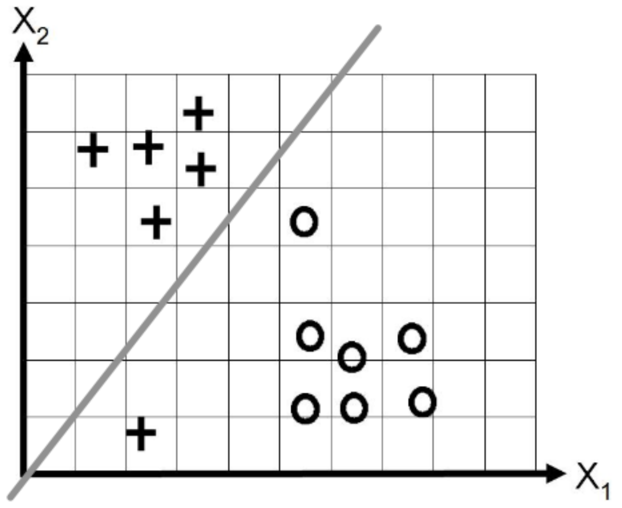
$$J_2(w) = -l(w, D_{\text{train}}) + \lambda w_2^2.$$

Again suppose λ is a very large number. Sketch a possible decision boundary. How many classification errors does your method make on the training set?

Solution: (a) As the data are linearly separable, logistic regression will find a line that fits the data perfectly. There will be no classification errors on the training set. The line is not unique (imagine wiggling it).

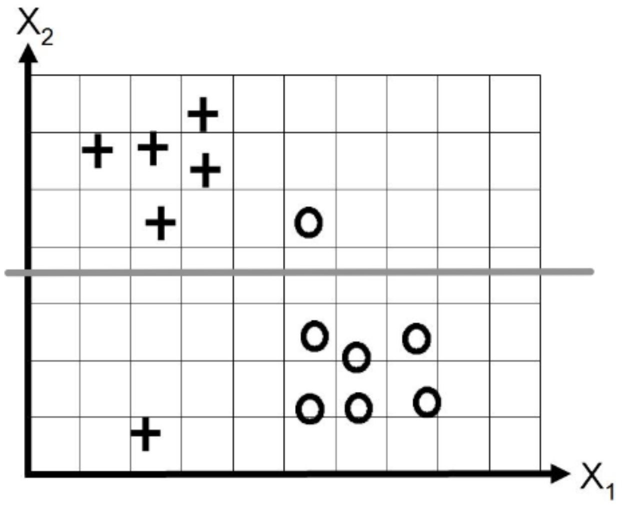


(b) Since $w_0 = 0$, this means that the point $(0, 0)$ must be on the decision boundary, because at that point $\sigma(w_0+w_1x_1+w_2x_2) = \sigma(0) = 0.5$. So, regularized logistic regression will find the best decision boundary that passes through $(0, 0)$. It will make one mistake on the training data.

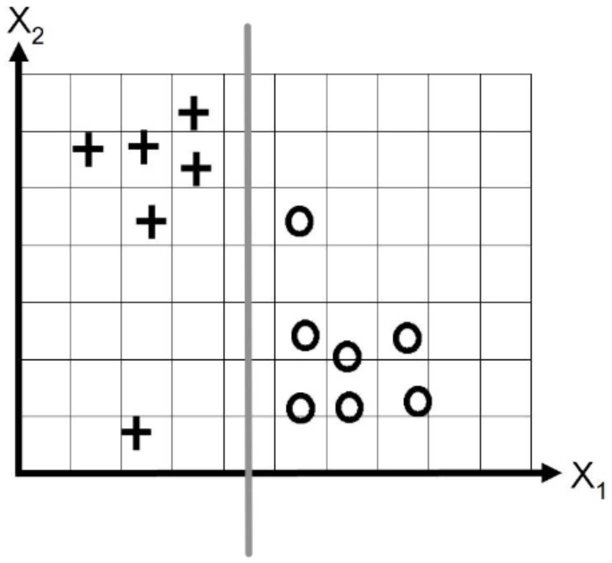


As an aside, it is for this reason that in regularized logistic or linear regression, we usually do not penalize the bias term (i.e., the weight that corresponds to the feature that is always 1).

(c) As the regularizer forces $w_1 = 0$, the decision boundary will be a horizontal line. There will be two classification errors.



(d) As the regularizer forces $w_2 = 0$, the decision boundary will be a vertical line. There will be zero classification errors.



Problem 4: In a binary classification problem, we use logistic regression with the cross-entropy loss function. The loss function for a single data point is defined as follows:

$$L = -\sum_{i=1}^{n} y_i \log(\hat{y}_i),$$

where y_i represents the true labels, and $\hat{y}_i$ represents the predicted probabilities obtained using the softmax function:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}.$$

Part (a): Calculate the derivative of the softmax function with respect to z_k for the cases $k = i$ and $k \neq i$.

Part (b): Calculate the derivative of the cross-entropy loss function with respect to z_k.

Solution:
Part (a): The derivative of the softmax function is calculated in two cases:
Case 1: When $i = k$, the goal is to compute $\frac{\partial \hat{y}_i}{\partial z_i}$:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Taking the derivative:

$$\frac{\partial \hat{y}_i}{\partial z_i} = \frac{\partial}{\partial z_i}\left(\frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}\right)$$

By applying the quotient rule:

$$\frac{\partial \hat{y}_i}{\partial z_i} = \frac{e^{z_i}\left(\sum_{j=1}^{n} e^{z_j}\right) - e^{z_i}e^{z_i}}{\left(\sum_{j=1}^{n} e^{z_j}\right)^2}$$

Simplifying:

$$\frac{\partial \hat{y}_i}{\partial z_i} = \hat{y}_i(1 - \hat{y}_i)$$

Case 2: When $i \neq k$, the goal is to compute $\frac{\partial \hat{y}_i}{\partial z_k}$:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Taking the derivative:

$$\frac{\partial \hat{y}_i}{\partial z_k} = \frac{e^{z_i}\left(\sum_{j=1}^{n} e^{z_j}\right) - e^{z_i}e^{z_k}}{\left(\sum_{j=1}^{n} e^{z_j}\right)^2}$$

Simplifying:

$$\frac{\partial \hat{y}_i}{\partial z_k} = -\hat{y}_i\hat{y}_k$$

Part (b): Using the chain rule, the derivative of the cross-entropy loss with respect to z_k is:

$$\frac{\partial L}{\partial z_k} = \sum_{i=1}^{n} \frac{\partial L}{\partial \hat{y}_i}\frac{\partial \hat{y}_i}{\partial z_k}$$

Given the loss function:

$$L = -\sum_{i=1}^{n} y_i \log(\hat{y}_i)$$

The derivative with respect to $\hat{y}_i$ is:

$$\frac{\partial L}{\partial \hat{y}_i} = -\frac{y_i}{\hat{y}_i}$$

Substituting this into the chain rule:

$$\frac{\partial L}{\partial z_k} = \sum_{i=1}^{n}\left(-\frac{y_i}{\hat{y}_i}\right)\frac{\partial \hat{y}_i}{\partial z_k}$$

From Part (a), we use the results of the softmax derivative:
For $i = k$:

$$\frac{\partial \hat{y}_i}{\partial z_k} = \hat{y}_k(1 - \hat{y}_k)$$

For $i \neq k$:

$$\frac{\partial \hat{y}_i}{\partial z_k} = -\hat{y}_i\hat{y}_k$$

Combining these:

$$\frac{\partial L}{\partial z_k} = -y_k(1 - \hat{y}_k) + \sum_{i \neq k}(\hat{y}_i\hat{y}_k)$$

Since $\sum_{i \neq k} y_i = 1 - y_k$:

$$\frac{\partial L}{\partial z_k} = -y_k(1 - \hat{y}_k) + \hat{y}_k(1 - y_k)$$

Simplifying further:

$$\frac{\partial L}{\partial z_k} = \hat{y}_k - y_k$$

Conclusion: This shows that the derivative of the cross-entropy loss is simply the difference between the predicted probability $\hat{y}_k$ and the true label y_k:

$$\frac{\partial L}{\partial z_k} = \hat{y}_k - y_k$$