Binary Classifiers and Logistic Regression

Omar Hijab*

Abstract

Hyperplane separability in a two-class dataset is related to the corresponding logistic regression problem. It is shown logistic regression, in its pure un-penalized form, is trainable exactly when the dataset is not separable.

Keywords: dataset, separability, logistic regression, maximum-margin hyperplane, binary classifier, decision boundary

1 Introduction

Linear classifiers of two-class datasets are widely used in machine learning. Two approaches to such classifiers are logistic regression and maximum-margin hyperplanes.

Maximum-margin hyperplanes [3] can be used when the classes are linearly separable. When this is not so, the classes may be linearly separated by embedding the dataset in a higher-dimensional space [1].

An alternate approach, the topic of this paper, is to use logistic regression to obtain a loss minimizing hyperplane. When this is feasible, we say logistic regression is *trainable*. Since this is not always feasible, trainability is often imposed by adding a penalty or regularization term to the loss function.

It turns out trainability of logistic regression, in the pure un-penalized setting, is intimately tied to the separability of the training dataset, in the sense *logistic regression is trainable exactly when the two classes are not linearly separable.*

As a consequence, gradient descent for logistic regression either *converges to the unique minimizing hyperplane*, or *diverges to infinity*, according to whether the dataset is inseparable or separable. These are the main results of the paper.

It is surprising that these basic results have not appeared in the literature. Also, because there is some terminology confusion in the literature, we discuss at the end the correct usage of some of the terms used here.

2 Background

Let x_1, x_2, ..., x_N be the samples of a two-class dataset in sample space, which we assume to be euclidean space $\mathbf{R}^d$ with d features, and let p_1, p_2, ..., p_N be the sequence of *bits* reflecting the class membership of the samples. Then the two classes correspond to $p = 0$ and $p = 1$ respectively. Because samples may be repeated, the two classes need not be disjoint.

*Temple University, Philadelphia, PA, USA, hijab@temple.edu.

Let m be a *nonzero* vector, b a scalar, and $m \cdot x$ the dot product. A *hyperplane* in sample space is specified by the equation

$$m \cdot x + b = 0.$$

When the samples x are scalars, a hyperplane is a point. When each sample has two features, a hyperplane is a line, and, with three features, a hyperplane is a plane.

Given a sample x, the *level* of x relative to a hyperplane is the scalar

$$y = m \cdot x + b.$$

Then the hyperplane consists of samples at level zero.

Let $y_k = m \cdot x_k + b$ be the level of the sample x_k relative to a hyperplane, $k = 1, 2, \dots, N$. The hyperplane is *separating* if

$$\begin{aligned} y_k &\le 0, &\quad \text{if } p_k = 0, \\ y_k &\ge 0, &\quad \text{if } p_k = 1, \end{aligned} \qquad k = 1, 2, \dots, N. \tag{1}$$

If the inequalities are strict, we say the hyperplane is *strictly separating.*

If there is a separating hyperplane, we say the dataset is *separable.* Otherwise, the dataset is *inseparable.* If there is a strictly separating hyperplane, we say the dataset is *strictly separable.*

If the dataset lies in a hyperplane, then that hyperplane is separating, so the separability question is only interesting when the dataset does not lie in a hyperplane.

If a dataset is separable, and neither class lies in the separating hyperplane (m, b), any samples in the separating hyperplane may be considered boundary cases.

In this case, we may select either class and reclassify these samples to belong to that class. This modification does not impact the separability, and the resulting dataset is strictly separable, as can be seen by shifting the bias b slightly. In this sense, separability and strict separability are almost the same.

If a dataset is strictly separable, then the convex hulls K_0 and K_1 of the two classes do not intersect, and there is a shortest line segment connecting them. A *maximum-margin hyperplane* is then the hyperplane orthogonal to such a shortest line segment and passing through its midpoint.

Let μ_0, μ_1 be the means of the two classes, and suppose μ_1 is in a separating hyperplane. Then, relative to the hyperplane, the level of μ_1 is zero, and the sample levels in the class $p = 1$ are nonnegative. Since the level of μ_1 is the average of the sample levels, these sample levels all vanish, hence the class lies in the hyperplane.

Since the same reasoning applies to μ_0, separability and equality of the means imply both means lie in the same separating hyperplane, hence the dataset lies in that separating hyperplane. Equivalently, if a separable dataset does not lie in a hyperplane, the means of the two classes are distinct.

Given a two-class dataset, a *binary classifier* is a procedure for classifying points in sample space into two classes, in a manner consistent with the dataset.

Given a separable dataset, we obtain a binary classifier by selecting a separating hyperplane and assigning points x to classes according to the sign of their level y.

More generally, given a hyperplane, we seek a procedure that computes a probability q, depending only on the level y of x, that x should be assigned to the class $p = 1$.

To be as consistent as possible with the dataset, we choose some *measure of discrepancy* $I(p, q)$ between probabilities p and q, and we select the hyperplane that minimizes the average of the discrepancies $I(p_k, q_k)$ between p_k and the probabilities q_k corresponding to the dataset samples x_k, $k = 1, 2, \ldots, N$.

Since y is a scalar and q is a probability, we use a *squashing function* $q = \sigma(y)$ to convert scalars to probabilities.

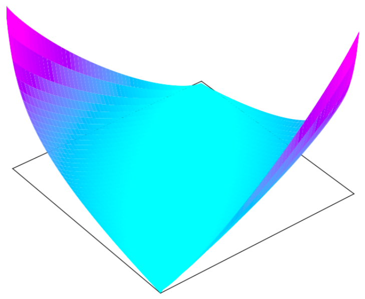

Figure 1: The graph of the relative information over the unit square.

The standard choices are the *relative information* (Figure 1)

$$I(p,q) = p\log\left(\frac{p}{q}\right) + (1-p)\log\left(\frac{1-p}{1-q}\right), \qquad 0 \le p \le 1, 0 < q < 1,$$

and the *sigmoid* activation function (Figure 4)

$$\sigma(y) = \frac{1}{1+e^{-y}}, \qquad -\infty < y < \infty.$$

Since $I \ge 0$ and $I = 0$ only when $p = q$, $I(p,q)$ is a measure of information discrepancy. Because $I(p,q)$ is not symmetric in (p,q), q is thought of as a base probability against which p is compared.

With these choices, *logistic regression* is the minimization of the *loss function*

$$J(m,b) = \frac{1}{N}\sum_{k=1}^{N} I(p_k,q_k), \qquad q_k = \sigma(y_k), \qquad y_k = m\cdot x_k + b,$$

over all m and b.

3 Results

A *minimizer* is a weight (m^*, b^*) satisfying $J(m^*, b^*) \le J(m,b)$ for all (m,b). We say logistic regression is *trainable* if the loss function has a minimizer (m^*, b^*).

Theorem 1. *Assume the dataset does not lie in a hyperplane. Then the loss function has at most one minimizer. Moreover the means of the classes agree iff the loss function has a minimizer with $m^* = 0$.*

Theorem 2. *Assume neither class lies in a hyperplane. Then logistic regression is trainable iff the dataset is inseparable.*

As a consequence of Theorems 1 and 2,

Theorem 3. *If*

1. *neither class lies in a hyperplane,*
2. *the means of the classes are distinct, and*
3. *the dataset is inseparable,*

there is a unique hyperplane (m^*, b^*) minimizing the loss function.

The minimizer (m^*, b^*) is the *LR hyperplane*. Let μ and Q be the mean and variance (matrix) of the dataset, and let

$$L = \frac{1}{4}\left(1 + |\mu|^2 + \mathrm{trace}(Q)\right). \tag{2}$$

A dataset is *standard* if each feature has mean zero and variance one. If the dataset is standard, $L = (1+d)/4$.

Theorem 4. *Assume neither class lies in a hyperplane, and let*

$$(m_1, b_1), \quad (m_2, b_2), \quad (m_3, b_3), \quad \dots$$

be a gradient descent sequence for the loss function, with learning rates equal to $1/L$. Then either the sequence converges to the unique minimizer of the loss function,

$$(m_n, b_n) \to (m^*, b^*), \qquad \text{as } n \to \infty,$$

or the sequence diverges to infinity,

$$|m_n|^2 + b_n^2 \to \infty, \qquad \text{as } n \to \infty,$$

according to whether the dataset is inseparable or separable.

The details and the proofs of these results are in §8.

4 An Example

A simple example of a two-class logistic regression problem, based on an example in [4], is as follows.

x	p	x	p	x	p	x	p	x	p
0.5	0	.75	0	1.0	0	1.25	0	1.5	0
1.75	0	1.75	1	2.0	0	2.25	1	2.5	0
2.75	1	3.0	0	3.25	1	3.5	0	4.0	1
4.25	1	4.5	1	4.75	1	5.0	1	5.5	1

Figure 2: Months trained and outcomes.

A group of hikers train to scale Mount Rainier. For each hiker, we know the number of months they train, as well as whether or not (p equal 1 or 0) they subsequently succeed at scaling the peak (Figure 2). We use logistic regression to provide a decision boundary or cut-off predicting success.

The samples here are scalars, and the dataset is one-dimensional. In Figure 3, $K_0 \cap K_1$ is the overlap between the two classes. The overlap plays a crucial role in the analysis.

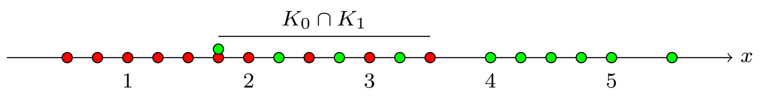

Figure 3: Hikers dataset samples.

Plotting the dataset on the (x,q) plane, the goal is to fit a curve

$$q = \sigma(mx + b) \tag{3}$$

as in Figure 4.

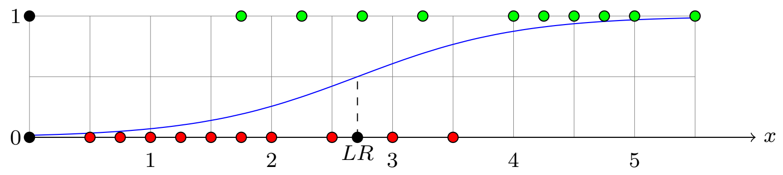

Figure 4: Fitted sigmoid curve for hikers dataset.

The dataset is one-dimensional, so a hyperplane is a point. Neither class lies in a hyperplane, and the dataset is inseparable (Figure 3). Hence logistic regression is trainable, and gradient descent is guaranteed to converge to the unique minimizing weight, which turns out to be

$$m^* = 1.50464542, \qquad b^* = -4.0777134.$$

The cut-off $x^* = -b^*/m^* = 2.71008$ is the LR hyperplane (Figure 4).

While this numerical result is in [4], the above framework guaranteeing the existence of (m^*, b^*) is not. Based on (2), (9), and Theorem 4, here is Python code for the gradient descent, with $(N,d) = (20,1)$ and $w = (m,b)$.

```
from numpy import *
from scipy.special import expit as sigma
from numpy.random import default_rng as rng

dataset = array([0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 1.75, 2.0,
    ↪ 2.25, 2.5, 2.75, 3.0, 3.25, 3.5, 4.0, 4.25, 4.5, 4.75,
    ↪ 5.0, 5.5])
N = len(dataset)
dataset = dataset.reshape(N,1)
labels = array([0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1,1,1,1])

def q(x,w):
   m, b = w[:-1], w[-1]
   return sigma(dot(m,x) + b)
```

```
def gradient(dataset,labels,w):
   G = array( [ (q(x,w)-p) * hstack([x,1]) for x,p in
    ↪ zip(dataset, labels) ])
   return mean(G, axis = 0)

def logreg(dataset,labels):
   Q = cov(dataset,bias = True)
   mu = mean(dataset,axis = 0)
   L = (1 + dot(mu,mu) + trace(Q))/4
   t = 1/L # short-step learning rate
   d = len(mu)
   w = rng().random(d+1) # initial m,b
   g = gradient(dataset,labels,w)
   num_iter = 0
   print("starting m,b: ",w)
   while not allclose(g,0):
      w -=  t * g
      g = gradient(dataset,labels,w)
      num_iter += 1
   print("minimizer m,b: ",w)
   print("gradient at minimizer: ",g)
   print("num iter: ", num_iter)

logreg(dataset,labels)
```

5 Properness

Let $|w|$ denote the absolute value of a scalar weight w or the length of a vector weight w, as the case may be. In our setting, weights are $w = (m, b)$. When $m \neq 0$, we say w is a *hyperplane weight*.

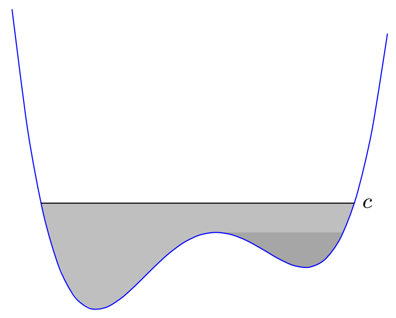

Figure 5: The graph of a proper continuous function.

Let $f(w)$ be a scalar function of weights w, defined on euclidean space. A *minimizer* is a weight w^* satisfying $f(w^*) \leq f(w)$ for all w.

We say $f(w)$ is *proper* if *every sublevel set is bounded:* for every level c, there is a bound C such that

$$f(w) \leq c \qquad \text{implies} \qquad |w| \leq C. \tag{4}$$

A proper function need not be convex everywhere (Figure 5). If the graph of $f(w)$ is the cross-section of a river, then properness means the river never floods its banks, no matter how much it rains.

Assume the gradient $\nabla f(w)$ is continuous. Then we have the following standard results:

- *If $f(w)$ is proper, then $f(w)$ has a minimizer* [2, §4.3].
- *If $f(w)$ is strictly convex and has a minimizer, then $f(w)$ is proper* [2, §4.5].

The parabola $f(w) = w^2$ is strictly convex on the real line, and is proper and has a minimizer. The exponential $f(w) = e^w$ is strictly convex on the real line, but is neither proper nor has a minimizer.

6 Separability

If $x_1, x_2, \dots, x_N$ is a dataset, a *convex combination* of samples is the weighed sum

$$x = t_1 x_1 + t_2 x_2 + \cdots + t_N x_N,$$

with t_k nonnegative and summing to one, $t_1 + t_2 + \cdots + t_N = 1$. The *convex hull* of a dataset is the set of all convex combinations of samples in the dataset.

A *ball* centered at a point μ with radius r is the set of points x whose distance from μ is no greater than r: $|x - \mu| \leq r$. Let K be any set of points in sample space. A point μ in K is in the *interior* of K if there is a ball centered at μ and wholly contained in K. With this understood, it is easy to check a hyperplane has no interior.

Let K_0 and K_1 be the convex hulls of the classes of a two-class dataset, and assume neither class lies in a hyperplane. Then we have the following standard result:

- *(Hyperplane separation theorem) The dataset is separable iff the intersection $K_0 \cap K_1$ has no interior.* [2, §4.5].

Any dataset may be "doubled" to a two-class dataset by taking each of the two classes to be the dataset itself. Then the doubled dataset is separable iff the original dataset lies in a hyperplane.

Applying the hyperplane separation theorem to the doubled dataset, we conclude *a dataset lies in a hyperplane iff its convex hull K has no interior.*

7 Gradient Descent

Let $f(w)$ be a scalar function of weights w, defined on euclidean space. In our setting, weights are $w = (m, b)$. A *descent sequence* is a sequence of weights $w_1, w_2, w_3, \dots$ satisfying

$$f(w_1) \geq f(w_2) \geq f(w_3) \geq \dots$$

If w is a weight in a sequence, let w^+ denote the next weight in the sequence. A *gradient descent sequence* is a sequence generated by

$$w^+ = w - t\nabla f(w).$$

The scalar $t > 0$ is the *learning rate*. In general, the learning rates may vary along the sequence.

If Q is a symmetric matrix and $u \cdot Qu \leq L$ for all unit vectors u, we write $Q \leq L$. For each w, the second derivative $D^2 f(w)$ is a symmetric matrix.

If the learning rates of a gradient descent sequence equal $1/L$ for some L satisfying

$$D^2 f(w) \leq L, \qquad \text{for all } w, \tag{5}$$

we say the gradient descent sequence is *short-step*.

When the gradient descent sequence is short-step,

$$f(w^+) \leq f(w) - \frac{t}{2}|\nabla f(w)|^2 \tag{6}$$

[2, §7.3], hence a short-step gradient descent sequence is a descent sequence.

We say a sequence *subconverges* to a limit if some subsequence converges to that limit. In this case, as a consequence of (6), if a short-step gradient descent sequence subconverges to w^*, then w^* is a critical point of $f(w)$. In particular, if $f(w)$ is convex, it follows w^* is a minimizer of $f(w)$.

8 Proofs of Results

Even though $m = 0$ does not correspond to a hyperplane, $J(m, b)$ is defined for all (m, b). We start by computing the derivatives of $J(m, b)$.

The *cumulant-generating function* of a fair coin, ignoring a constant term, is

$$Z(y) = \log(1 + e^y).$$

Then

$$Z'(y) = q, \qquad Z''(y) = q' = q(1-q), \qquad q = \sigma(y).$$

Let $I(p)$ be the *absolute information*,

$$I(p) = p \log p + (1-p)\log(1-p), \qquad 0 \leq p \leq 1.$$

Then

$$I(p, q) = I(p) - py + Z(y), \qquad q = \sigma(y), \tag{7}$$

This last identity, the *information error identity*, implies $I(p)$ and $Z(y)$ are dual convex functions [2, §4.1].

Since $I(p) = 0$ when $p = 0, 1$, (7) implies

$$I(p, q) = \left\{\begin{matrix} Z(y), & \text{if } p = 0, \\ Z(y) - y = Z(-y), & \text{if } p = 1, \end{matrix}\right\} \qquad q = \sigma(y). \tag{8}$$

Consequently, $J(m, b)$ is the standard "cross-entropy" loss function.

From (7), we have

$$\frac{d}{dy} I(p, q) = q - p, \qquad \frac{d^2}{dy^2} I(p, q) = q(1-q), \qquad q = \sigma(y).$$

By the chain rule,

$$\frac{d}{dt} I(p, \sigma(y + tv)) = (q - p)v, \qquad \frac{d^2}{dt^2} I(p, \sigma(y + tv)) = q(1-q)v^2.$$

Let v_0 be a vector and v_1 a scalar, and let $v = v_0 \cdot x + v_1$. Then

$$(m + tv_0) \cdot x + (b + tv_1) = (m \cdot x + b) + t(v_0 \cdot x + v_1) = y + tv.$$

With

$$q_k = \sigma(y_k) = \sigma(m \cdot x_k + b), \qquad k = 1, 2, \dots, N,$$

the first directional derivative of the loss function is

$$\left.\frac{d}{dt}\right|_{t=0} J(m + tv_0, b + tv_1) = \frac{1}{N} \sum_{k=1}^{N} (q_k - p_k)(v_0 \cdot x_k + v_1), \tag{9}$$

and the second directional derivative of the loss function is

$$\left.\frac{d^2}{dt^2}\right|_{t=0} J(m + tv_0, b + tv_1) = \frac{1}{N} \sum_{k=1}^{N} q_k(1 - q_k)(v_0 \cdot x_k + v_1)^2. \tag{10}$$

Since $0 < q_k < 1$, $k = 1, 2, \dots, N$, by (10), the second directional derivative is nonnegative, so the loss function is convex.

If the second directional derivative equals zero for some m, b, v_0, and v_1, by (10), the dataset satisfies $v_0 \cdot x_k + v_1 = 0$, $k = 1, 2, \dots, N$. Thus the dataset lies in a hyperplane, unless $v_0 = v_1 = 0$.

Under the assumptions of Theorem 1, the dataset does not lie in a hyperplane, hence $v_0 = v_1 = 0$. This establishes the strict convexity of the loss function. Since a strictly convex function has at most one minimizer, this establishes the first portion of Theorem 1.

Since (9) shows the gradient of the loss function is a continuous function of (m, b), we see trainability of logistic regression and properness of the loss function are equivalent, when the dataset does not lie in a hyperplane.

A point (m, b) is a *critical point* if the first directional derivative (9) vanishes in all directions (v_0, v_1). For any convex function, a point is critical iff it is a minimizer.

Let n be the number of samples in the class $p = 1$, and let μ_0, μ_1 be the means of the classes $p = 0$, $p = 1$. Then

$$\frac{n}{N}\mu_1 = \frac{1}{N} \sum_{p_k=1} x_k, \qquad \frac{N-n}{N}\mu_0 = \frac{1}{N} \sum_{p_k=0} x_k.$$

Given b, let $q = \sigma(b)$. If $(0, b)$ is a critical point, then the first directional derivative vanishes, and (9) implies

$$\frac{N-n}{N}\, q\, (\mu_0 \cdot v_0 + v_1) = \frac{n}{N}\, (1 - q)\, (\mu_1 \cdot v_0 + v_1) \tag{11}$$

for all v_0 and v_1. Taking $v_1 = 1$ and $v_0 = 0$ in (11) implies $q = n/N$. Using this, and taking $v_1 = 0$ and v_0 arbitrary in (11), we conclude $\mu_0 = \mu_1$.

Conversely, if $\mu_0 = \mu_1$, let $q = n/N$ be the proportion of samples satisfying $p_k = 1$, and let $b = \sigma^{-1}(q)$. Then (11) holds for all v_0 and v_1. It follows (9) vanishes with $(m, b) = (0, b)$, hence $(0, b)$ is a critical point. This establishes the second portion of Theorem 1.

By (8),

$$I(p, q) \le \log 2, \qquad q = \sigma(y), \qquad \begin{cases} \text{if } p = 0 \text{ and } y \le 0, \\ \text{if } p = 1 \text{ and } y \ge 0. \end{cases} \tag{12}$$

For Theorem 2, assume the dataset is separable. Then there is a separating hyperplane (m, b). By (12), $I(p_k, q_k) \leq \log 2$ for $k = 1, 2, \dots, N$, hence $J(m, b) \leq \log 2$. But (tm, tb) is the same hyperplane, so

$$J(tm, tb) \leq \log 2, \qquad t > 0.$$

Since this contradicts (4), the loss function is not proper, hence logistic regression is not trainable.

On the other hand, suppose the dataset is inseparable. Let K_0 and K_1 be the convex hulls of the two classes. If neither class lies in a hyperplane, by the hyperplane separation theorem, the intersection $K_0 \cap K_1$ has interior, so there is a ball B in $K_0 \cap K_1$.

Let μ and r be the center and radius of B. We establish properness by showing

$$J(m, b) \leq c \qquad \text{implies} \qquad |m| + |b| \leq \frac{cN}{r} \cdot (1 + r + |\mu|)\,. \tag{13}$$

Since $w = (m, b)$ implies

$$|w| = \sqrt{|m|^2 + b^2} \leq |m| + |b|,$$

(13) implies properness.

If $J(m, b) \leq c$, then $I(p_k, q_k) \leq cN$, $k = 1, 2, \dots, N$. Since $y < Z(y)$, by (8),

$$\begin{aligned} y_k < Z(y_k) = I(p_k, q_k) \leq cN, &\qquad \text{if } p_k = 0, \\ -y_k < Z(-y_k) = I(p_k, q_k) \leq cN, &\qquad \text{if } p_k = 1, \end{aligned} \qquad k = 1, 2, \dots, N.$$

By taking convex combinations,

$$\begin{aligned} y \leq cN, &\qquad \text{for } x \text{ in } K_0, \\ -y \leq cN, &\qquad \text{for } x \text{ in } K_1. \end{aligned}$$

From this,

$$|m \cdot x + b| \leq cN, \qquad \text{for } x \text{ in } K_0 \cap K_1.$$

If v is a unit vector, the points $x_\pm = \mu \pm rv$ are in B. Since

$$2rm \cdot v = (m \cdot x_+ + b) - (m \cdot x_- + b),$$

we have

$$2r|m \cdot v| \leq |m \cdot x_+ + b| + |m \cdot x_- + b| \leq 2cN.$$

Choosing $v = m/|m|$, we obtain

$$|m| \leq \frac{cN}{r}.$$

The point μ is in B. Since

$$b = (m \cdot \mu + b) - m \cdot \mu,$$

we have

$$|b| \leq |m \cdot \mu + b| + |m \cdot \mu| \leq cN + |m|\,|\mu|.$$

This leads to (13), establishing properness, hence trainability, of the loss function. This completes the proof of Theorem 2.

Theorem 3 is an immediate consequence of Theorems 1 and 2. To prove Theorem 4, let

$$\mu = \frac{1}{N}\sum_{k=1}^{N} x_k, \qquad Q = \frac{1}{N}\left(\sum_{k=1}^{N} x_k \otimes x_k\right) - \mu \otimes \mu$$

be the mean and variance of the dataset. Then, with L given by (2),

$$L = 1 + |\mu|^2 + \operatorname{trace}(Q) = \frac{1}{N}\sum_{k=1}^{N}\left(1 + |x_k|^2\right).$$

Since $q(1-q) \leq 1/4$ and

$$(v_0 \cdot x + v_1)^2 \leq \left(1 + |x|^2\right)\left(|v_0|^2 + v_1^2\right),$$

by (10), we conclude

$$D^2 J(m,b) \leq L, \qquad \text{for all } (m,b). \tag{14}$$

Let w_1, w_2, w_3, ... be a gradient descent sequence for the loss function, with L given by (2). Then, by (14), the sequence is short-step. Hence, by (6), the sequence is a descent sequence. It follows the sequence remains in a sublevel set. If the dataset is inseparable, then $J(m,b)$ is proper, hence sublevel sets are bounded. This shows the sequence does not diverge to infinity.

On the other hand, if the sequence does not diverge to infinity, then the sequence subconverges to some critical point, hence there is a minimizer. By Theorem 2, the dataset is inseparable. This completes the proof of Theorem 4.

Theorem 4 and its proof remain valid for non-constant learning rates t, as long as $\epsilon \leq t \leq 1/L$, for some fixed $\epsilon > 0$.

9 Discussion

The takeaway is we have two mutually exclusive cases. Either a dataset is separable or not. If a dataset is separable, it may be modified, as discussed in §2, to be strictly separable, and then a maximum-margin hyperplane provides an optimal decision boundary.

If a dataset is inseparable, logistic regression provides a unique minimizing hyperplane. To what extent this *LR hyperplane* mimicks the properties of a maximum-margin hyperplane is then a natural question.

Theorem 4 is an explicit algorithm for checking separability. Without explicit bounds on the rate of divergence, it remains more of theoretical value.

10 Terminology

1. In the literature, $I(p,q)$ is called the "Kullback-Liebler divergence" or "relative entropy". We prefer the term relative information because this terminology is more descriptive and consistent with the absolute information $I(p)$, and because $I(p,q)$ is convex.

2. In Python, as of this writing, `scipy.stats.entropy(p)` returns the *absolute entropy* $H(p) = -I(p)$, and `scipy.stats.entropy(p,q)` returns the relative information $I(p,q)$, not the *relative entropy* $H(p,q) = -I(p,q)$. Apart from being incorrect, this is inconsistent, even within Python.

3. In the literature, $Z(y) - p \cdot y$ is called the "cross-entropy". We prefer the term *cross-information* because this terminology is consistent with the terminology for $I(p)$ and $I(p,q)$, and because (see (7)) the cross-information equals $I(p,q) - I(p)$ when $q = \sigma(y)$.
4. To further support our terminology choices, we note entropy is the negative of information, entropy is concave, information is convex, and loss functions are minimized, not maximized. Of course, the issue here is terminology, not the choice of loss function: the loss function in the literature is identical with the loss function here.

11 Disclosure of Funding

This research was not supported by third-party funding. Moreover, the author reports there are no competing interests to declare.

References

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] O. Hijab. *Math for Data Science*. Springer Nature, 2025.

[3] I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, 2008.

[4] Wikipedia. *Logistic Regression*. URL: https://en.wikipedia.org/wiki/Logistic_regression.