Logistic Regression

Chris Cornwell

Feb 27, 2025

Outline

Outline

Decision boundaries

Suppose function $h : \mathbb{R}^d \longrightarrow \{y_1, y_2, \ldots, y_m\}$ gives predictions for a classification task (data points $\mathbf{x} \in \mathbb{R}^d$, with m labels).

Decision boundaries

Suppose function $h : \mathbb{R}^d \longrightarrow \{y_1, y_2, \ldots, y_m\}$ gives predictions for a classification task (data points $\mathbf{x} \in \mathbb{R}^d$, with m labels).

For $i \in \{1, 2, \ldots, m\}$, define C_{y_i} to be the set of points with label y_i, i.e.,

$$C_{y_i} = h^{-1}(y_i) = \{\mathbf{x} \in \mathbb{R}^d \mid h(\mathbf{x}) = y_i\}.$$

Decision boundaries

Suppose function $h : \mathbb{R}^d \longrightarrow \{y_1, y_2, \ldots, y_m\}$ gives predictions for a classification task (data points $\mathbf{x} \in \mathbb{R}^d$, with m labels).

For $i \in \{1, 2, \ldots, m\}$, define C_{y_i} to be the set of points with label y_i, i.e.,

$$C_{y_i} = h^{-1}(y_i) = \{\mathbf{x} \in \mathbb{R}^d \mid h(\mathbf{x}) = y_i\}.$$

For $i \neq j$, a point is in **decision boundary for** h when it's in the boundary of both C_{y_i} and C_{y_j}. For half-space model (last lecture) determined by $\mathbf{w}$ and b, decision boundary is the hyperplane – i.e., points $\mathbf{x}$ such that $\mathbf{w} \cdot \mathbf{x} + b = 0$.

Decision boundaries

Suppose function $h : \mathbb{R}^d \to \{y_1, y_2, \dots, y_m\}$ gives predictions for a classification task (data points $\mathbf{x} \in \mathbb{R}^d$, with m labels).

For $i \in \{1, 2, \dots, m\}$, define C_{y_i} to be the set of points with label y_i, i.e.,

$$C_{y_i} = h^{-1}(y_i) = \{\mathbf{x} \in \mathbb{R}^d \mid h(\mathbf{x}) = y_i\}.$$

For $i \neq j$, a point is in **decision boundary for** h when it's in the boundary of both C_{y_i} and C_{y_j}. For half-space model (last lecture) determined by $\mathbf{w}$ and b, decision boundary is the hyperplane – i.e., points $\mathbf{x}$ such that $\mathbf{w} \cdot \mathbf{x} + b = 0$.

Perceptron algorithm might give model h with many data points that are *near* the decision boundary.

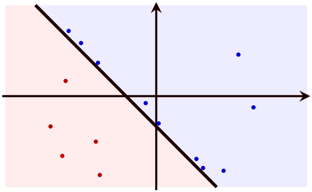


Figure: Many points near the decision boundary

But...data is messy

But…data is messy

Many points “close” to the decision boundary $\Longrightarrow$ likely for newly observed data to appear on “wrong side” of decision boundary. Perceptron algorithm has nothing to correct for this.

But...data is messy

Many points “close” to the decision boundary $\Rightarrow$ likely for newly observed data to appear on “wrong side” of decision boundary.

Perceptron algorithm has nothing to correct for this.

Some thoughts:

- When $\mathbf{x}$ close to decision boundary, we feel less *confident* in giving the label $h(\mathbf{x})$. In contrast, when $\mathbf{x}$ farther from boundary, and only one label is observed nearby, more confidence is warranted.

But...data is messy

Many points “close” to the decision boundary $\Rightarrow$ likely for newly observed data to appear on “wrong side” of decision boundary.
Perceptron algorithm has nothing to correct for this.
Some thoughts:

- When $\mathbf{x}$ close to decision boundary, we feel less *confident* in giving the label $h(\mathbf{x})$. In contrast, when $\mathbf{x}$ farther from boundary, and only one label is observed nearby, more confidence is warranted.
- Also, the immediate change of label across the boundary (a discontinuity in the model) ...perhaps not “natural”?

Outline

Reconsidering the Half-space Model

Logistic model

Incorporating a probability into half-space model

Don't only capture the sign of $\mathbf{w} \cdot \mathbf{x} + b$; instead, compose $\mathbf{w} \cdot \mathbf{x} + b$ with the **logistic function**.

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

Incorporating a probability into half-space model

Don't only capture the sign of $\mathbf{w} \cdot \mathbf{x} + b$; instead, compose $\mathbf{w} \cdot \mathbf{x} + b$ with the **logistic function**.

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

- $0 < \sigma(z) < 1$ for all $z \in \mathbb{R}$;

Incorporating a probability into half-space model

Don't only capture the sign of $\mathbf{w} \cdot \mathbf{x} + b$; instead, compose $\mathbf{w} \cdot \mathbf{x} + b$ with the **logistic function**.

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

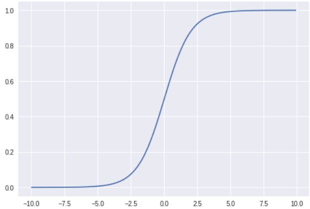
- $0 < \sigma(z) < 1$ for all $z \in \mathbb{R}$;
- $\lim_{z \to \infty} \sigma(z) = 1$ and $\lim_{z \to -\infty} \sigma(z) = 0$;
- $\sigma(0) = 1/2$.

Incorporating a probability into half-space model

Don't only capture the sign of $\mathbf{w} \cdot \mathbf{x} + b$; instead, compose $\mathbf{w} \cdot \mathbf{x} + b$ with the **logistic function**.

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

- $0 < \sigma(z) < 1$ for all $z \in \mathbb{R}$;
- $\lim_{z \to \infty} \sigma(z) = 1$ and $\lim_{z \to -\infty} \sigma(z) = 0$;
- $\sigma(0) = 1/2$.



Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

[1] Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

---

[1] Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

1. If $\mathbf{w} \cdot \mathbf{x} + b > 0$ is very large ($\mathbf{x}$ far away from H and on positive side), then $\sigma(z)$ is very close to 1.

[1]Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

1. If $\mathbf{w} \cdot \mathbf{x} + b > 0$ is very large ($\mathbf{x}$ far away from H and on positive side), then $\sigma(z)$ is very close to 1.
2. If $\mathbf{w} \cdot \mathbf{x} + b < 0$ has abs. value very large ($\mathbf{x}$ far away from H on negative side), then $\sigma(z)$ very close to 0.[1]

[1] Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

1. If $\mathbf{w} \cdot \mathbf{x} + b > 0$ is very large ($\mathbf{x}$ far away from H and on positive side), then $\sigma(z)$ is very close to 1.
2. If $\mathbf{w} \cdot \mathbf{x} + b < 0$ has abs. value very large ($\mathbf{x}$ far away from H on negative side), then $\sigma(z)$ very close to 0.[1]
3. If $\mathbf{x}$ is contained in H itself, $z = 0$ and $\sigma(z) = 0.5$.

[1] Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

1. If $\mathbf{w} \cdot \mathbf{x} + b > 0$ is very large ($\mathbf{x}$ far away from H and on positive side), then $\sigma(z)$ is very close to 1.
2. If $\mathbf{w} \cdot \mathbf{x} + b < 0$ has abs. value very large ($\mathbf{x}$ far away from H on negative side), then $\sigma(z)$ very close to 0.[1]
3. If $\mathbf{x}$ is contained in H itself, $z = 0$ and $\sigma(z) = 0.5$.

Logistic model (a binary classifier): $h(\mathbf{x}) = 1$ if $\sigma(\mathbf{w} \cdot \mathbf{x} + b) \geq 0.5$, and $h(\mathbf{x}) = -1$ otherwise.

- *Remember that probability (certainty) in the prediction.*

[1] Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.

Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

1. If $\mathbf{w} \cdot \mathbf{x} + b > 0$ is very large ($\mathbf{x}$ far away from H and on positive side), then $\sigma(z)$ is very close to 1.
2. If $\mathbf{w} \cdot \mathbf{x} + b < 0$ has abs. value very large ($\mathbf{x}$ far away from H on negative side), then $\sigma(z)$ very close to 0.[1]
3. If $\mathbf{x}$ is contained in H itself, $z = 0$ and $\sigma(z) = 0.5$.

Logistic model (a binary classifier): $h(\mathbf{x}) = 1$ if $\sigma(\mathbf{w} \cdot \mathbf{x} + b) \geq 0.5$, and $h(\mathbf{x}) = -1$ otherwise.

- *Remember that probability (certainty) in the prediction.*
- *Logistic model helps decision boundary not be near data* (more on this later).

---

[1] Since -1 is only other label, high probability of having label -1.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.
Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

Logistic model (a binary classifier): $h(\mathbf{x}) = 1$ if $\sigma(\mathbf{w} \cdot \mathbf{x} + b) \geq 0.5$, and $h(\mathbf{x}) = -1$ otherwise.

Logistic model, continued

"Logistic regression", used for binary classification, as follows.
Find hyperplane H, as determined by some $\mathbf{w}$ and b, that fits labeled data well; given new $\mathbf{x} \in \mathbb{R}^d$, find $z = \mathbf{w} \cdot \mathbf{x} + b$, then compute $\sigma(z)$.

- (Logistic regression) $\sigma(z)$, the *probability* that the label is $+1$.

Logistic model (a binary classifier): $h(\mathbf{x}) = 1$ if $\sigma(\mathbf{w} \cdot \mathbf{x} + b) \geq 0.5$, and $h(\mathbf{x}) = -1$ otherwise.
Some rationale for the use of the logistic function $\frac{1}{1+e^{-z}}$ in this process, comes from an inverse direction.

- If wanting to use *Maximum Likelihood Estimation* to get binary classification model, then some typical simplifying assumptions on conditional probability P, given model parameters, of observing $\mathbf{x_i}$ $\rightsquigarrow \log\left(\frac{P}{1-P}\right)$ being linear.

How to find $\mathbf{w}$ and b

Given $\{\pm 1\}$ labeled data, how do we go about finding $\mathbf{w}$ and b to use in the logistic (regression) model?

How to find $\mathbf{w}$ and b

Given $\{\pm 1\}$ labeled data, how do we go about finding $\mathbf{w}$ and b to use in the logistic (regression) model?

- Even if data is linearly separable, Perceptron algorithm does not try to make hyperplane be positioned “away from” data (Disadvantage).
- If data is not linearly separable, what should be done?

How to find $\mathbf{w}$ and b

Given $\{\pm 1\}$ labeled data, how do we go about finding $\mathbf{w}$ and b to use in the logistic (regression) model?

- Even if data is linearly separable, Perceptron algorithm does not try to make hyperplane be positioned “away from” data (Disadvantage).
- If data is not linearly separable, what should be done?

Future lectures: Will discuss using <u>optimization</u> (calculus-based) to find best parameters $w_1, w_2, \ldots, w_d, b$; process called Gradient Descent.

How to find $\mathbf{w}$ and b

Given $\{\pm 1\}$ labeled data, how do we go about finding $\mathbf{w}$ and b to use in the logistic (regression) model?

- Even if data is linearly separable, Perceptron algorithm does not try to make hyperplane be positioned "away from" data (Disadvantage).
- If data is not linearly separable, what should be done?

Future lectures: Will discuss using optimization (calculus-based) to find best parameters $w_1, w_2, \ldots, w_d, b$; process called Gradient Descent.

- Relevant: relationship between gradient of a function and its directional derivative (discussed in Calc III).

More messy versus less messy data

Could introduce additional parameter (more flexibility).

For $k > 0$, define

$$\sigma_k(z) = \frac{1}{1 + e^{-kz}}.$$

[2] I first saw the idea to introduce k, and interpret the model this way, from Eli Grigsby at Boston College.

More messy versus less messy data

Could introduce additional parameter (more flexibility).

For $k > 0$, define

$$\sigma_k(z) = \frac{1}{1 + e^{-kz}}.$$

$0 < k < 1$: values of $\sigma_k(z)$ transition from 0 to 1 more slowly.

$k > 1$: values of $\sigma_k(z)$ transition from 0 to 1 more quickly. (think about derivative)

---

[2] I first saw the idea to introduce k, and interpret the model this way, from Eli Grigsby at Boston College.

More messy versus less messy data

Could introduce additional parameter (more flexibility).

For $k > 0$, define

$$\sigma_k(z) = \frac{1}{1 + e^{-kz}}.$$

$0 < k < 1$: values of $\sigma_k(z)$ transition from 0 to 1 more slowly.

$k > 1$: values of $\sigma_k(z)$ transition from 0 to 1 more quickly. (think about derivative)

Hence, if know data has more noise, might use $0 < k < 1$ to decrease measure of confidence in prediction. In contrast, very “clean” data, interpretation of the model might benefit from $k > 1$.[2]

[2] I first saw the idea to introduce k, and interpret the model this way, from Eli Grigsby at Boston College.

More messy versus less messy data

Could introduce additional parameter (more flexibility).

For $k > 0$, define

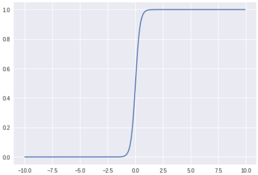
$$\sigma_k(z) = \frac{1}{1 + e^{-kz}}.$$

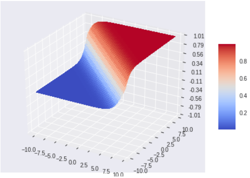
$0 < k < 1$: values of $\sigma_k(z)$ transition from 0 to 1 more slowly.

$k > 1$: values of $\sigma_k(z)$ transition from 0 to 1 more quickly. (think about derivative)

Hence, if know data has more noise, might use $0 < k < 1$ to decrease measure of confidence in prediction. In contrast, very "clean" data, interpretation of the model might benefit from $k > 1$.[2]

(**Left:** graph with $k = 5$; **Right:** applied to points in $\mathbb{R}^2$)





[2] I first saw the idea to introduce k, and interpret the model this way, from Eli Grigsby at Boston College.