CHAPTER 5: Logistic Regression

Instructor: PhD. Nguyen Thi Quy

Group 5: 5.4 - 5.6.1

October 17, 2023

5.4 Learning in Logistic Regression

Learning in Logistic Regression

Supervised classification:

- We know the correct label y (either 0 or 1) for each x.
- But what the system produces is an estimate, $\hat{y}$.

Learning in Logistic Regression

Supervised classification:

- We know the correct label y (either 0 or 1) for each x.
- But what the system produces is an estimate, $\hat{y}$.

We want to set w and b to minimize the distance between our estimate y^i and the true $\hat{y}^i$.

- We need a distance estimator: **a loss function** or **a cost function**
- We need **an optimization algorithm** to update w and b to minimize the loss.

Learning in Logistic Regression

Learning components:

- A loss function: **cross-entropy loss**.
- An optimization algorithm: **stochastic gradient descent**.

5.5 The cross-entropy loss function

The cross-entropy loss function

The distance between $\hat{y}$ and y

We want to know how far is the classifier output:

$$\hat{y} = \sigma(w.x + b)$$

From the true output:

y [= either 0 or 1]

The cross-entropy loss function

The distance between $\hat{y}$ and y

We want to know how far is the classifier output:

$$\hat{y} = \sigma(w.x + b)$$

From the true output:

y [= either 0 or 1]

We'll call this difference:

=> L($\hat{y}$,y) = how much $\hat{y}$ differs from the true y

Cross-entropy loss

Conditional maximum likelihood estimation

We choose the parameters w,b that:

- **Maximize** the log probability of the **true y labels** in the training data given the observations x

The resulting loss function is the **negative log likelihood loss**, generally called the **cross-entropy loss**.

Cross-entropy loss

Goal: maximize probability of the correct label p(y|x)

Since there are only 2 discrete outcomes (0 or 1) we can express the probability $p(y|x)$ from our classifier (the thing we want to maximize) as:

$$p(y|x) = \hat{y}^y (1 - \hat{y})^{1-y} \qquad (1)$$

NOTE:

- if y=1, this simplifies to $\hat{y}$
- if y=0, this simplifies to $1 - \hat{y}$

The cross-entropy loss function

Goal: maximize probability of the correct label p(y|x)

Maximize:
$$p(y|x) = \hat{y}^y (1-\hat{y})^{1-y} \qquad (1)$$

Take the log of both sides:

Maximize:
$$\begin{aligned} \log p(y|x) &= \log \left[\hat{y}^y (1-\hat{y})^{1-y}\right] \\ &= \mathrm{y} \;\; \log \hat{y} + (1-y) \log (1-\hat{y}) \qquad (2) \end{aligned}$$

Cross-entropy loss

Goal: maximize probability of the correct label p(y|x)

- **Maximize**: $\log p(y|x) = y \ \log \hat{y} + (1-y)\log(1-\hat{y}) \quad (2)$
- Flip sign to turn this into a loss: something to **minimize**

  Minimize:

  $L_{CE}(\hat{y}, y) = -\log p(y|x) = -[y \log \hat{y} + (1-y)\log(1-\hat{y})] (3)$

Plugging in definition of y : $\hat{y} = \sigma(w.x + b)$

$$L_{CE}(\hat{y}, y) = -[y \log \sigma(w.x + b) + (1-y)\log(1-\sigma(w.x+b))] (4)$$

Cross-entropy loss

Plugging in definition of y : $\hat{y} = \sigma(w.x + b)$

$$L_{CE}(\hat{y}, y) = -[y \log \sigma(w.x + b) + (1 - y) \log (1 - \sigma(w.x + b))] \quad (4)$$

We want loss to be:

- **Smaller** if the model estimate is close to correct
- **Bigger** if model is confused

Let's see if this works for our sentiment example

Example: Fig 5.2

It's hokey. There are virtually no surprises, and the writing is second-rate. So why was it so enjoyable ? For one thing, the cast is great. Another nice touch is the music. I was overcome with the urge to get off the couch and start dancing. It sucked me in, and it'll do the same to you.

With:

$w = [2.5, 5.0, 1.2, 0.5, 2.0, 0.7], b = 0.1, x = [3, 2, 1, 3, 0, 4.19]$

Fig 5.2

Check:

$$L_{CE}(\hat{y}, y) = -[y \log \sigma(w.x + b) + (1 - y) \log (1 - \sigma(w.x + b))] \quad (4)$$

- With:

$$w = [2.5, 5.0, 1.2, 0.5, 2.0, 0.7], b = 0.1, x = [3, 2, 1, 3, 0, 4.19]$$

Let's first suppose the true label of this is y=1 (positive)

$$\begin{aligned} L_{CE}(\hat{y}, y) &= -y \log \sigma(w.x + b) \\ &= -\log \sigma(w.x + b) \\ &= -\log 0.70 \\ &= 0.36 \end{aligned}$$

Fig 5.2

Check:

$$L_{CE}(\hat{y}, y) = -[y \log \sigma(w.x + b) + (1 - y) \log (1 - \sigma(w.x + b))] (4)$$

- With:

$$w = [2.5, 5.0, 1.2, 0.5, 2.0, 0.7], b = 0.1, x = [3, 2, 1, 3, 0, 4.19]$$

Let's first suppose the true label of this is y=0 (negative)

$$\begin{aligned} L_{CE}(\hat{y}, y) &= -(1 - y) \log (1 - \sigma(w.x + b)) \\ &= -\log 1 - \sigma(w.x + b) \\ &= -\log 0.30 \\ &= 1.2 \end{aligned}$$

Let's see if this works for our sentiment example

- The loss when model was right (if true y=1):
$$L_{CE}(\hat{y}, y) = 0.36$$
- Is lower than the loss when model was wrong (if true y=0):
$$L_{CE}(\hat{y}, y) = 1.2$$

Sure enough, loss was bigger when model was wrong!

5.6: Gradient Descent

Gradient Descent

Our goal: minimize the loss.

Let's make explicit that the loss function is parameterized by weights $\theta = (w, b)$

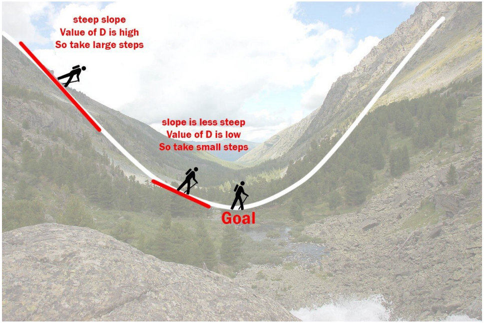
- And we'll represent $\hat{y}$ as $f(x;\theta)$ to make the dependence on θ more obvious.

We want the weights that minimize the loss, averaged over all examples:

$$\hat{\boldsymbol{\theta}} = \underset{\theta}{\operatorname{argmin}} \frac{\mathbf{1}}{\boldsymbol{m}} \sum_{i=1}^{m} \mathcal{L}_{\boldsymbol{CE}}(f(x^{(i)}; \boldsymbol{\theta}), y^{(i)})$$

Intuition of gradient descent

How do I get to the bottom of this mountain?

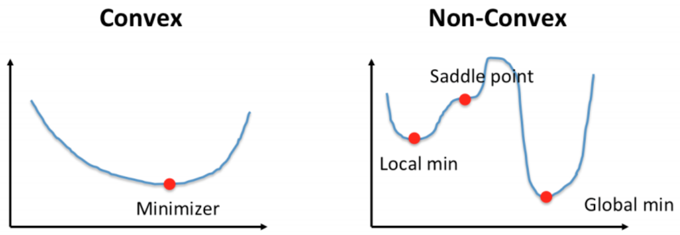


Look around me 360°
Find the direction of steepest slope down
Go that way

Our goal: minimize the loss

For logistic regression, loss function is convex

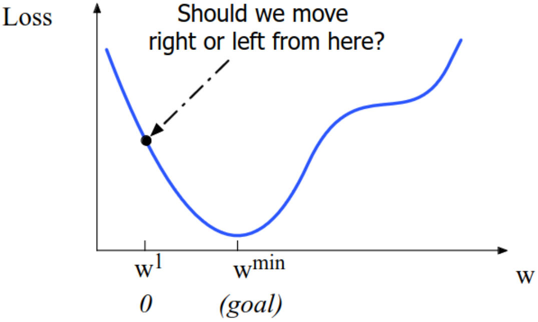
- A convex function has just one minimum
- Gradient descent starting from any point is guaranteed to find the minimum
  - (Loss for neural networks is non-convex)



Our goal: minimize the loss

Let's first visualize for a single scalar w

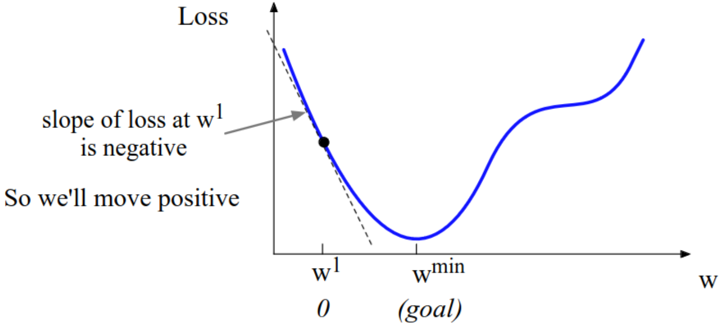
Q: Given current w, should we make it bigger or smaller?
A: Move w in the reverse direction from the slope of the functio



Our goal: minimize the loss

Let's first visualize for a single scalar w

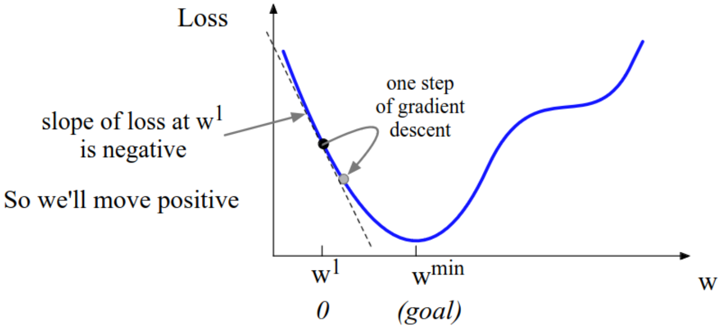
Q: Given current w, should we make it bigger or smaller?
A: Move w in the reverse direction from the slope of the functio



Our goal: minimize the loss

Let's first visualize for a single scalar w

Q: Given current w, should we make it bigger or smaller?
A: Move w in the reverse direction from the slope of the functio



Gradients

The **gradient** of a function of many variables is a vector pointing in the direction of the greatest increase in a function.

Gradient Descent: Find the gradient of the loss function at the current point and move in the **opposite** direction.

Gradient Descent Concepts

How much do we move in that direction ?

- The value of the gradient (slope in our example) $\frac{\mathrm{d}}{\mathrm{d}w}\mathcal{L}(f(x;w),y)$ weighted by a learning rate η
- Higher learning rate η means move w faster

$$w^{t+1} = w^t - \eta\frac{\mathrm{d}}{\mathrm{d}w}\mathcal{L}(f(x;w),y)$$

Reference

Speech and Language Processing (3rd ed. draft)
Dan Jurafsky and James H. Martin
Part I: Fundamental Algorithms, *Chapter 5: Logistic Regression*

Thanks for listening!

Q&A section