

26: Logistic Regression

Jerry Cain

May 25, 2022

Table of Contents

2	Preparation
7	Logistic Regression
24	Training: Big Picture
37	Training: Details



Linear to Logical: Preparation

1. Weighted sum

If $\mathbf{X} = (X_1, X_2, \dots, X_m)$:

$$Z = \theta_1 X_1 + \theta_2 X_2 + \dots + \theta_m X_m$$

$$= \sum_{j=1}^m \theta_j X_j$$

weighted sum

$$= \theta^T \mathbf{X}$$

dot product

$\vec{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_n \end{bmatrix} \quad \vec{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_m \end{bmatrix}$

$= [\theta_1 \ \theta_2 \ \dots \ \theta_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$

1. Weighted sum

Dot product/
weighted sum $\theta^T \mathbf{X} = \sum_{j=1}^m \theta_j X_j$

Recall the linear regression model, where $\mathbf{X} = (X_1, X_2, \dots, X_m)$ and $Y \in \mathbb{R}$:

$$\hat{Y} = g(\mathbf{X}) = \theta_0 + \sum_{j=1}^m \theta_j X_j$$

$\hat{Y} = aX_1 + b$
 $\theta_1 \quad \theta_0$

How would you rewrite this expression as a single dot product?

$$g(\mathbf{X}) = \theta_0 X_0 + \theta_1 X_1 + \theta_2 X_2 + \dots + \theta_m X_m \quad \text{Define } X_0 = 1$$

$$= \theta^T \mathbf{X} \quad \text{New } \mathbf{X} = (1, X_1, X_2, \dots, X_m)$$

Prepending $X_0 = 1$ to each feature vector \mathbf{X} makes matrix operators more accessible.



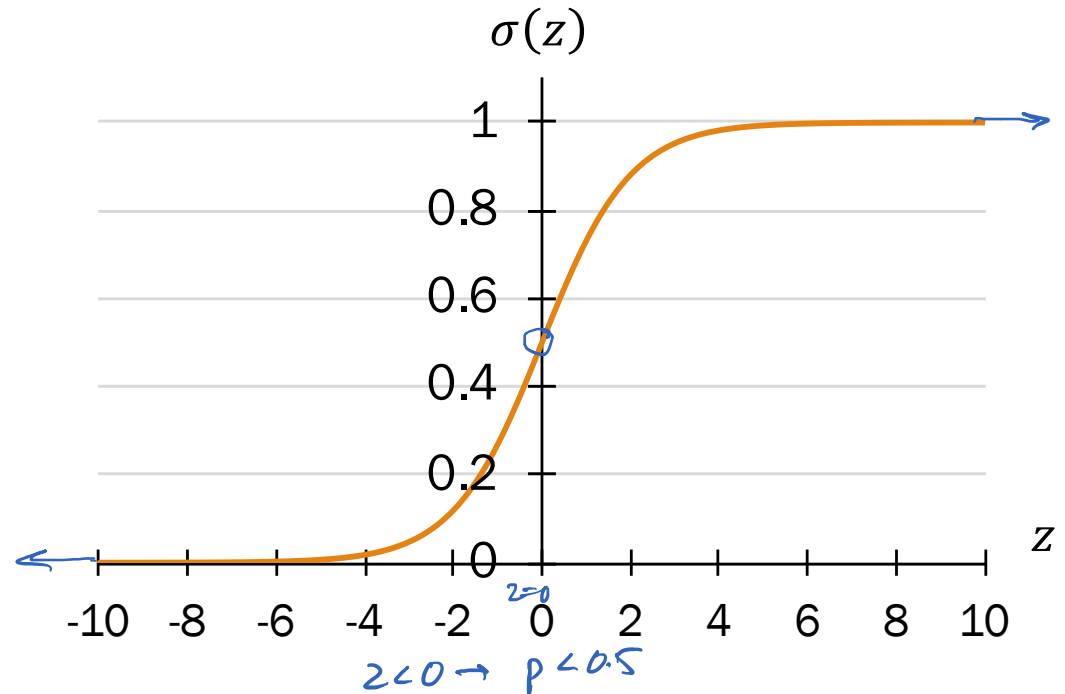
2. Sigmoid function $\sigma(z)$

$z > 0 \rightarrow p > 0.5$

- The sigmoid function:

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

- Sigmoid squashes z to a number between 0 and 1.



- Recall definition of probability:
A number between 0 and 1

$\sigma(z)$ can represent a probability.

3. Conditional likelihood function

Training data (n datapoints):

- $(\mathbf{x}^{(i)}, y^{(i)})$ drawn iid from a distribution $f(\mathbf{X} = \mathbf{x}^{(i)}, Y = y^{(i)} | \theta) = f(\mathbf{x}^{(i)}, y^{(i)} | \theta)$

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta)$$

$$= \arg \max_{\theta} \sum_{i=1}^n \log f(y^{(i)} | \mathbf{x}^{(i)}, \theta)$$

$$= \arg \max_{\theta} LL(\theta)$$

conditional likelihood
of training data

log conditional likelihood

- MLE here is estimator that maximizes **conditional likelihood**
- Confusingly, log conditional likelihood is also written as $LL(\theta)$



Logistic Regression

Prediction models so far

Linear Regression (Regression)

$$\mathbf{X} \rightarrow \left[\theta_0 + \sum_{j=1}^m \theta_j X_j \right] \rightarrow \hat{Y}$$
$$\hat{Y} = \theta_0 + \sum_{j=1}^m \theta_j X_j$$

- ✅ \mathbf{X} can be dependent
- 👤 Regression model ($\hat{Y} \in \mathbb{R}$, not discrete)

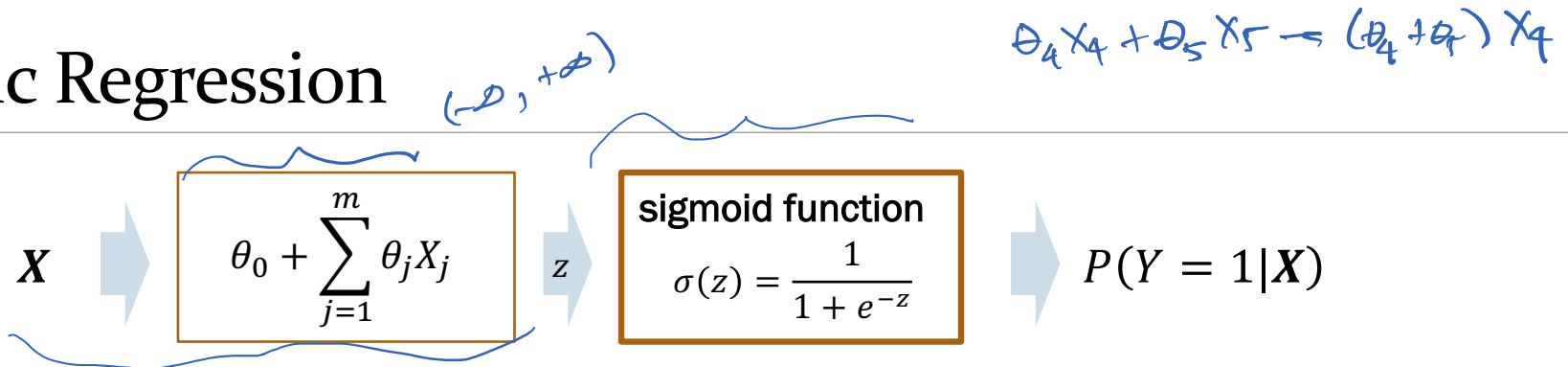
Naïve Bayes (Classification)

$$\begin{array}{l} \mathbf{X} \\ Y \end{array} \rightarrow \left[\hat{P}(\mathbf{X}|Y)\hat{P}(Y) \right] \rightarrow \hat{P}(\mathbf{X}, Y)$$

$$\begin{aligned} \hat{Y} &= \arg \max_{y=\{0,1\}} P(Y | \mathbf{X}) \\ &= \arg \max_{y=\{0,1\}} P(\mathbf{X}|Y)P(Y) \end{aligned}$$

- ✅ Tractable with NB assumption, but...
- ⚠️ Realistically, X_j features not always conditionally independent
- 👤 Actually models $P(\mathbf{X}, Y)$, not $P(Y|\mathbf{X})$?

Logistic Regression



Logistic Regression
Model:

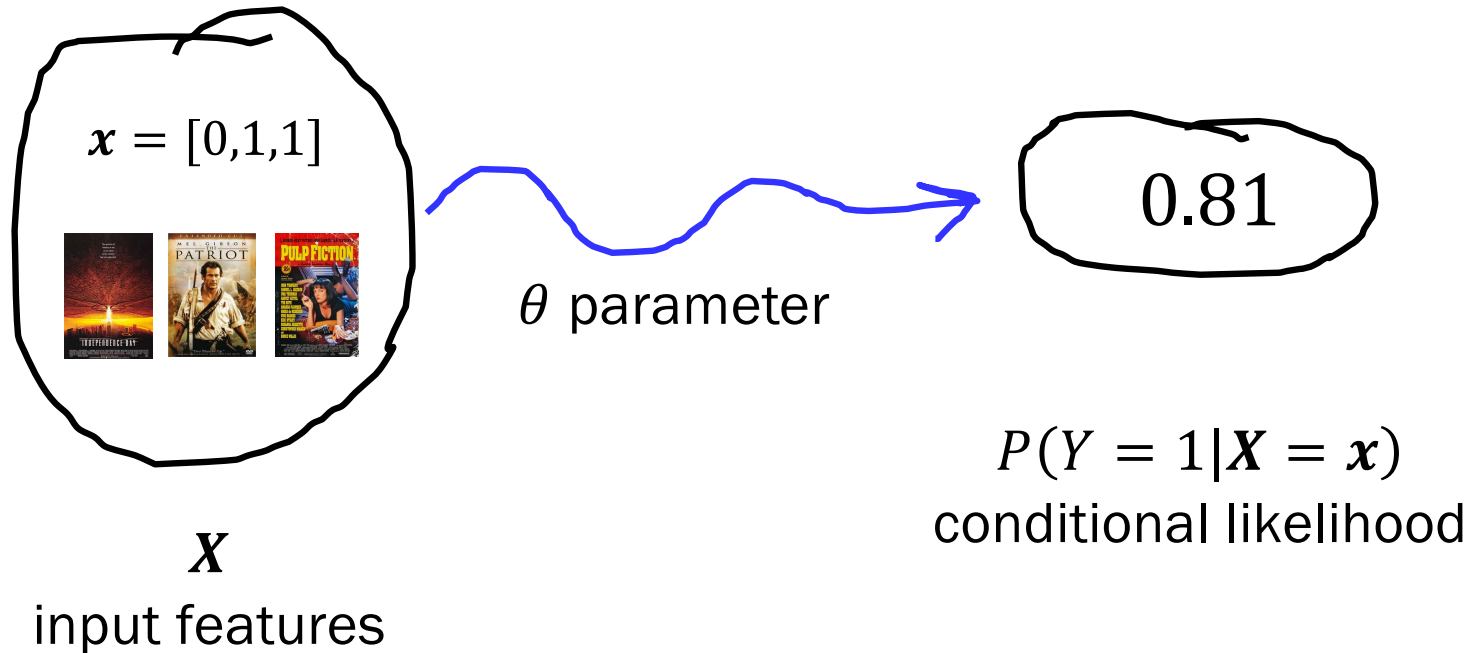
$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Predict \hat{Y} as the more likely Y
given our observation $\mathbf{X} = \mathbf{x}$:

$$\hat{Y} = \arg \max_{y \in \{0,1\}} P(Y | \mathbf{X})$$

- Since $Y \in \{0,1\}$, $P(Y = 0 | \mathbf{X} = \mathbf{x}) = 1 - \sigma(\theta_0 + \sum_{j=1}^m \theta_j x_j)$
- Sigmoid function also known as **logit function**

Logistic Regression



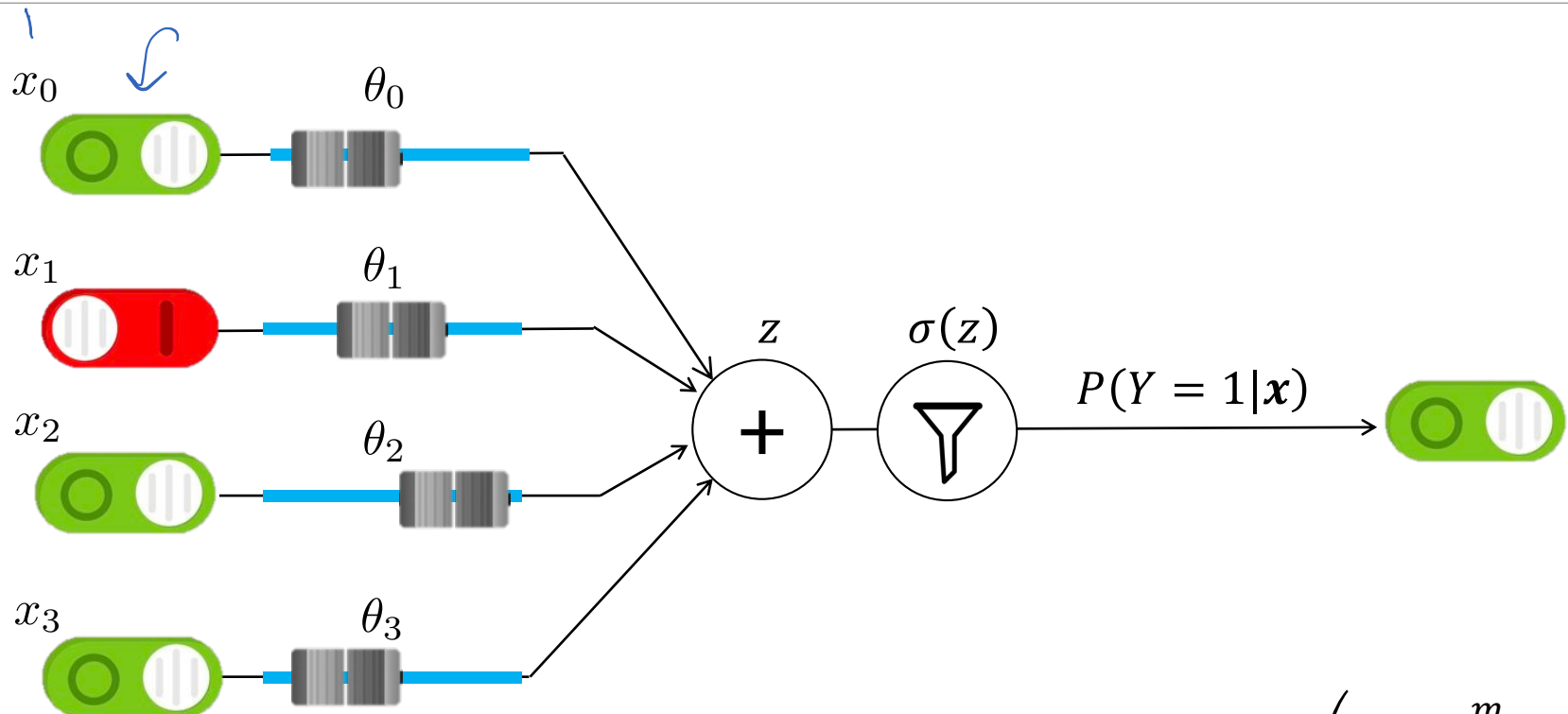
$$P(Y = 1 | X = x) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Logistic Regression: Key Metaphor



θ parameter

Logistic Regression: Key Metaphor

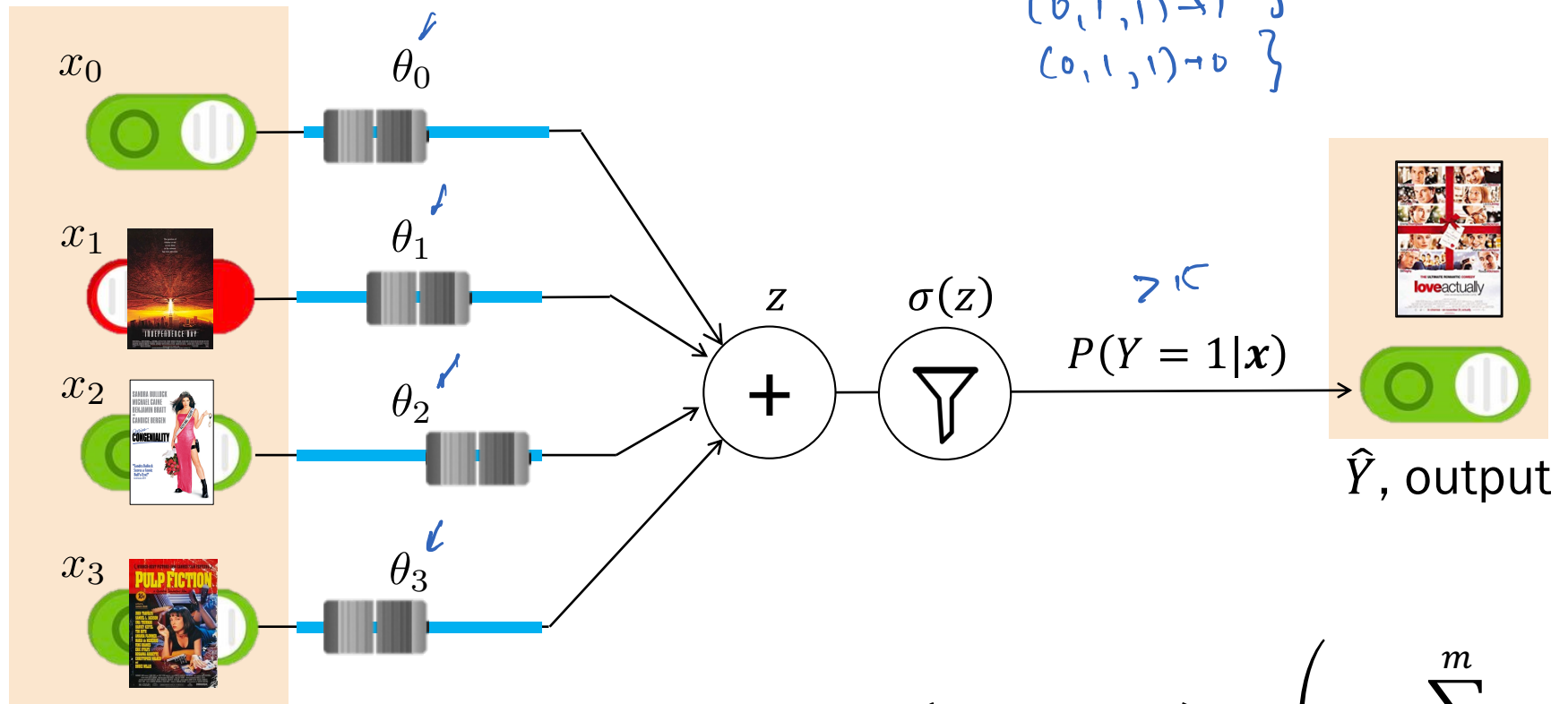


$$P(Y = 1|X = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Logistic Regression: Key Metaphor

$(0,1,1) \rightarrow 1$
 $(0,1,1) \rightarrow 1$
 $(0,1,1) \rightarrow 1$
 $(0,1,1) \rightarrow 0$

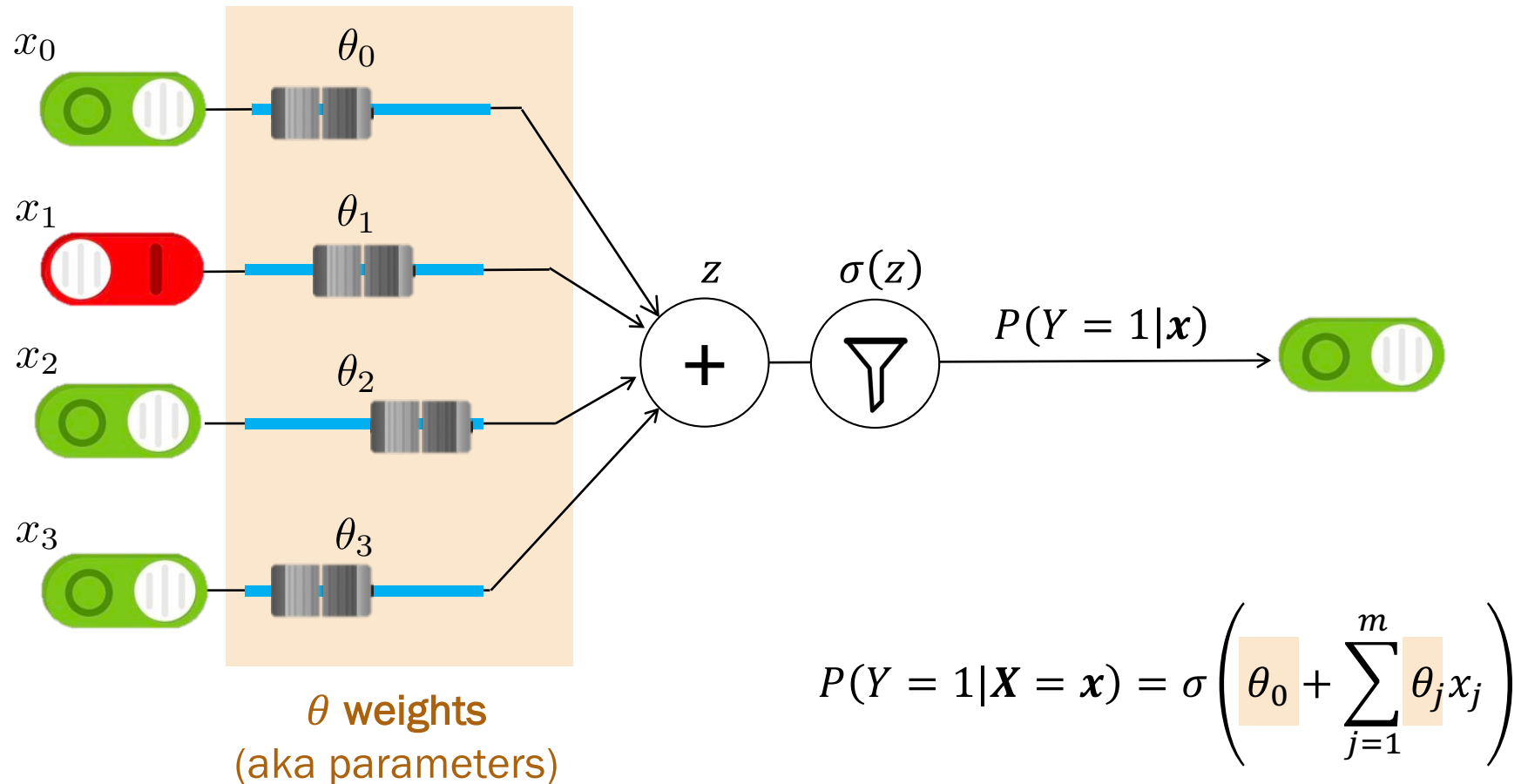
} training set



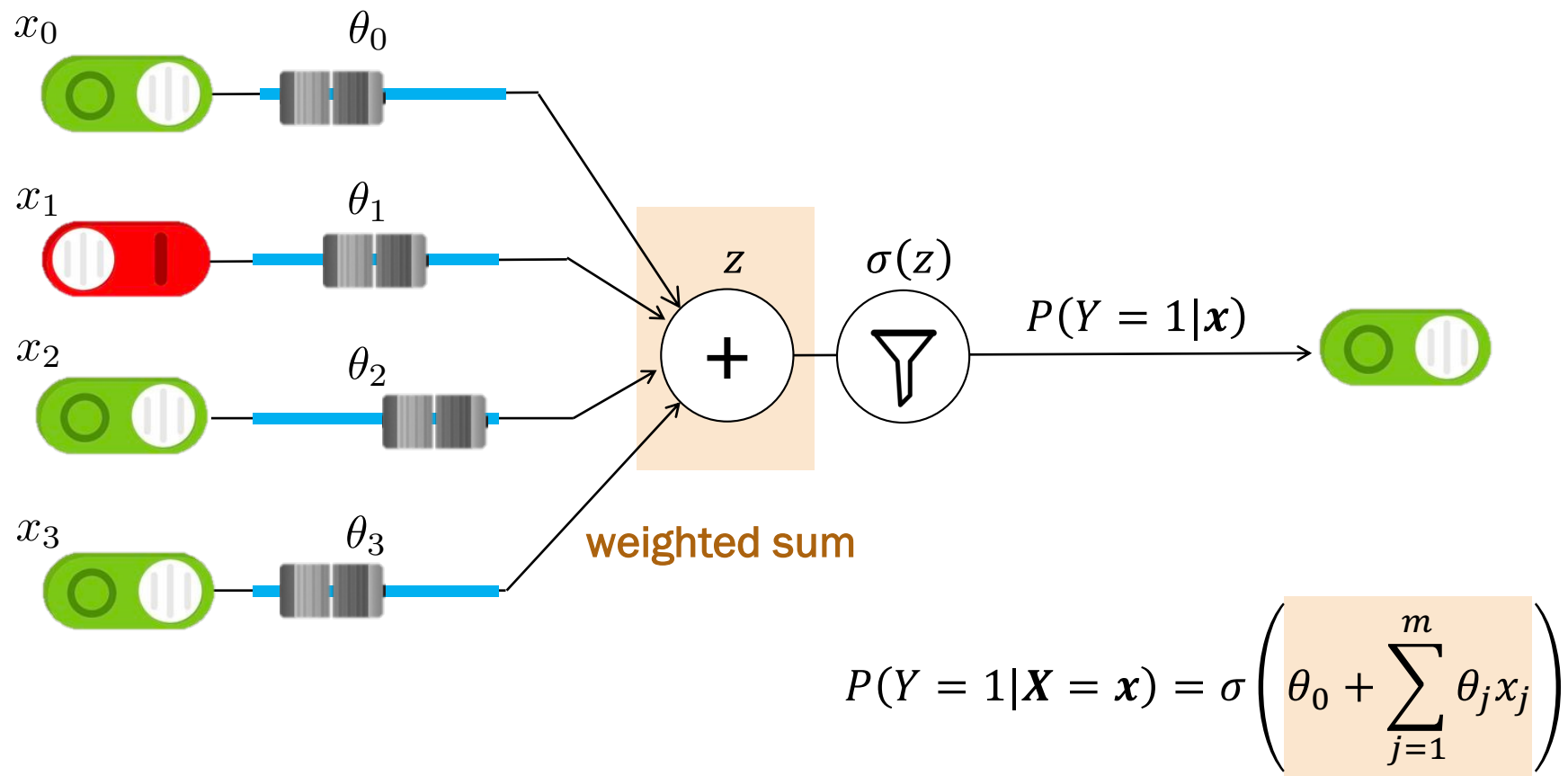
\mathbf{X} , input features
 $[0,1,1]$

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

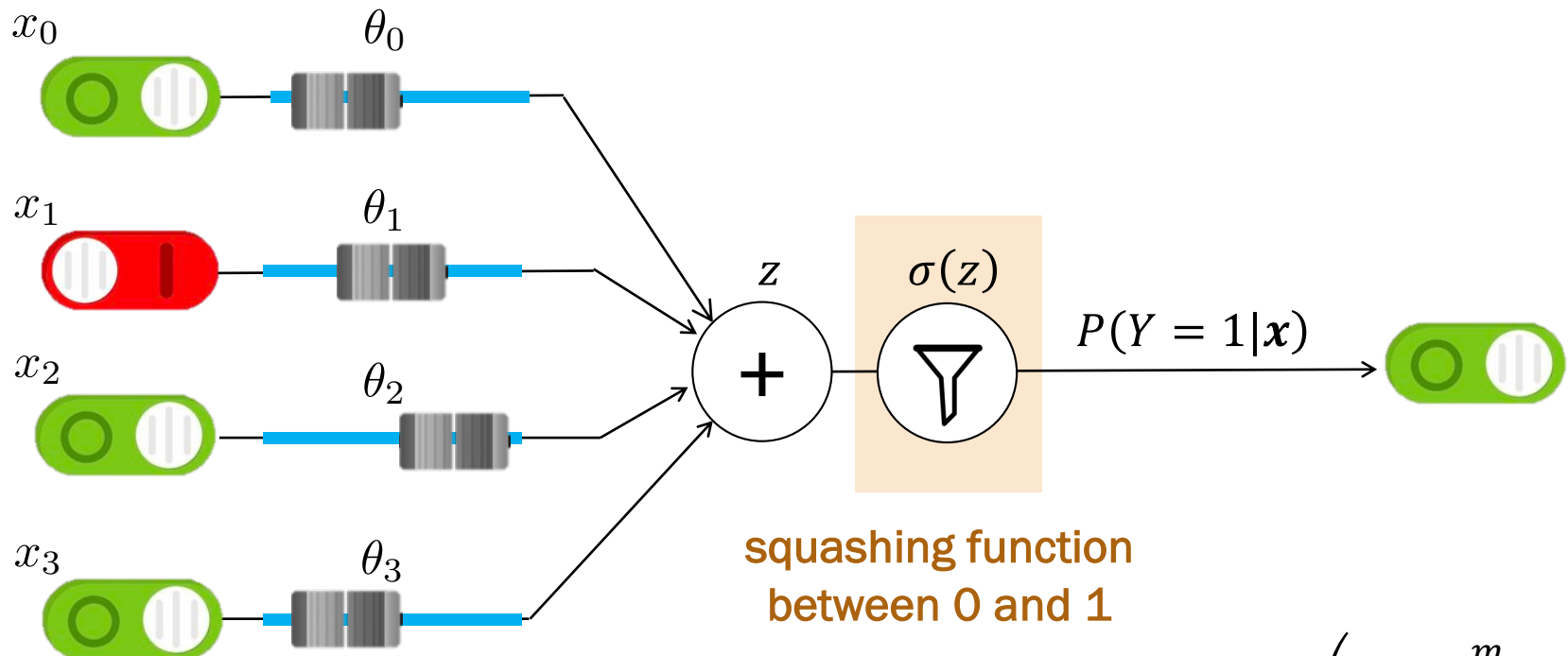
Components of Logistic Regression



Components of Logistic Regression



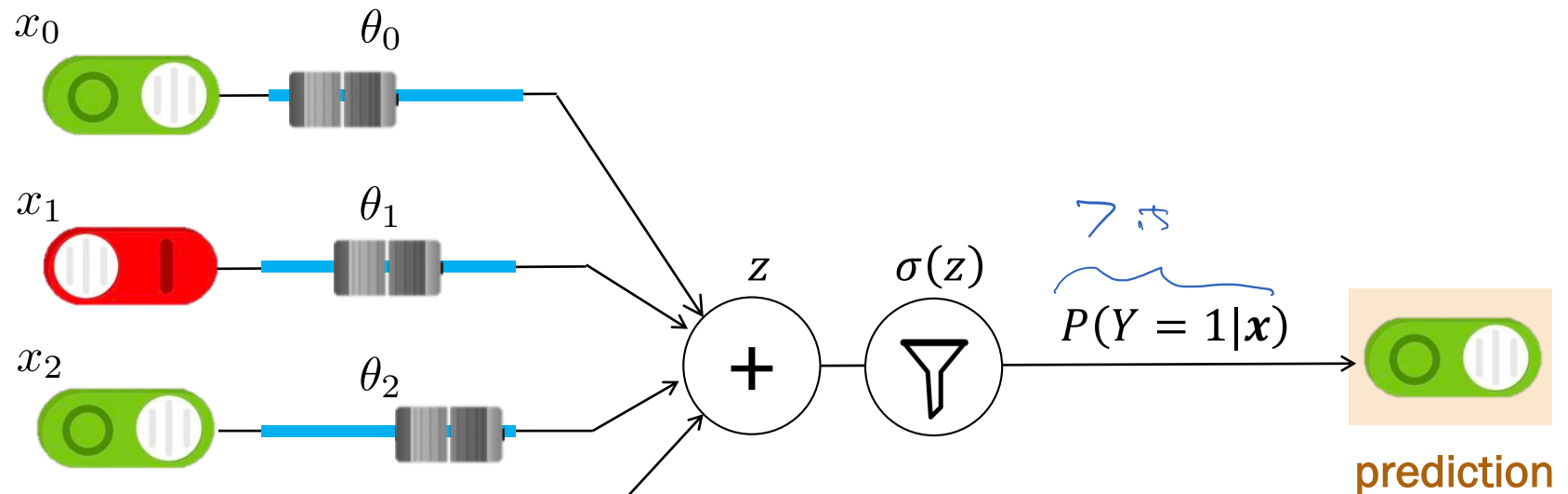
Components of Logistic Regression



squashing function
between 0 and 1

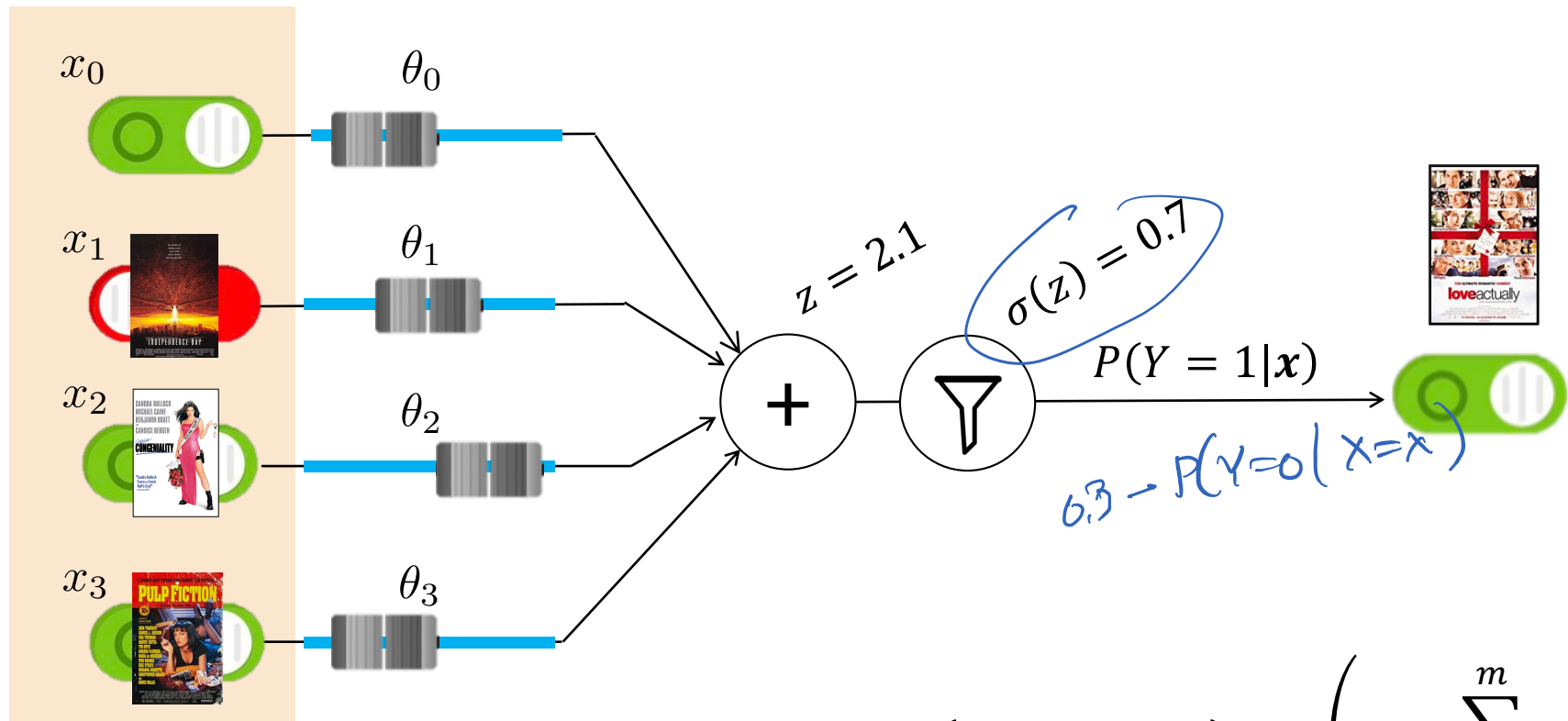
$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Components of Logistic Regression



$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

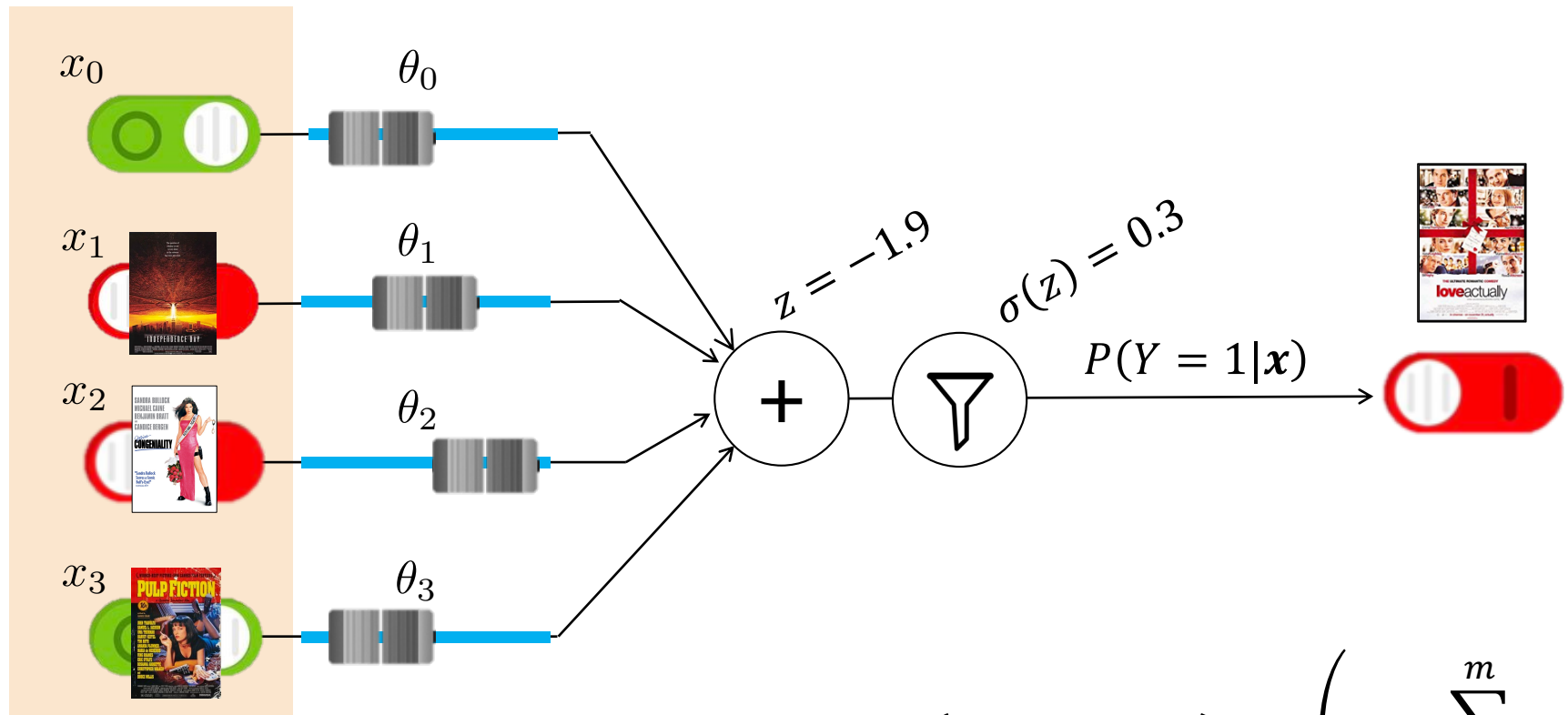
Different predictions for different inputs



\mathbf{X} , input features
[0,1,1]

$$P(Y = 1|\mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

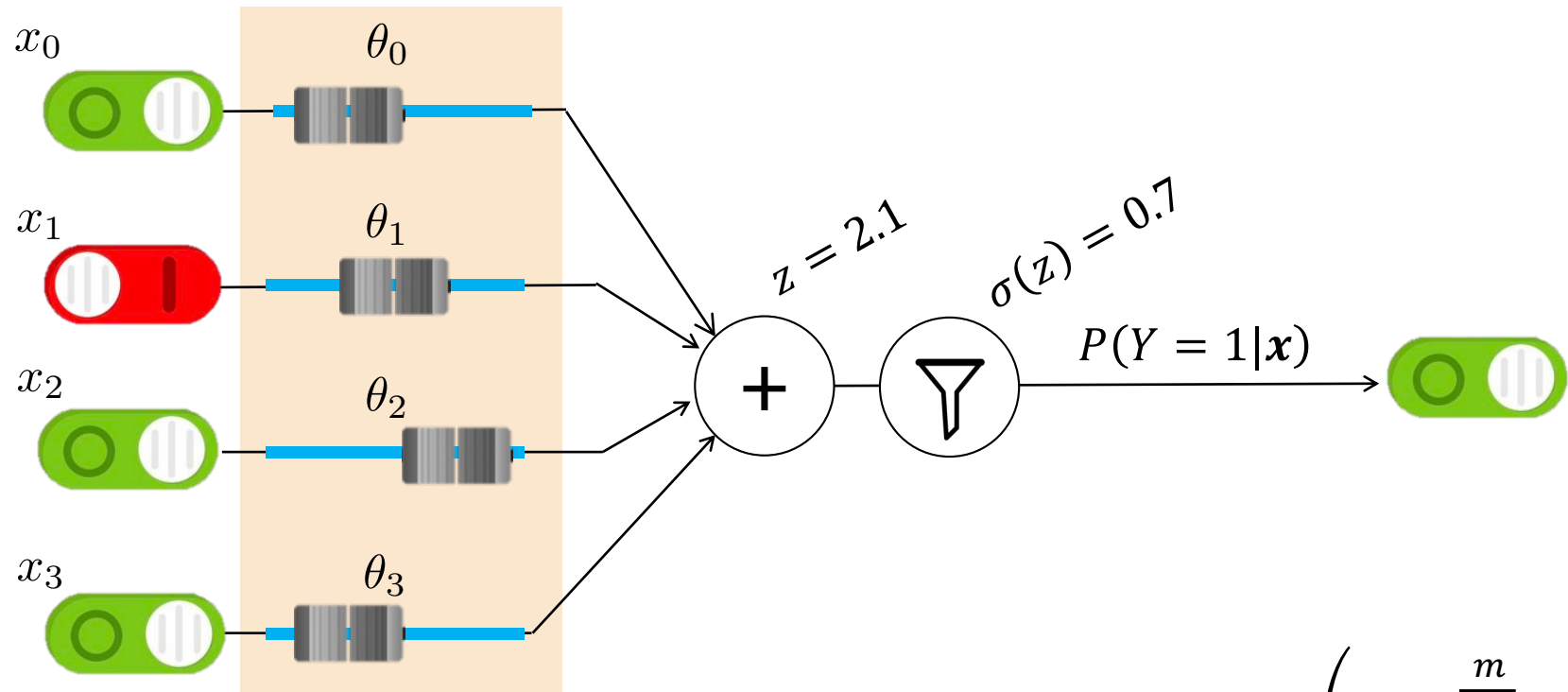
Different predictions for different inputs



\mathbf{X} , input features
[0,0,1]

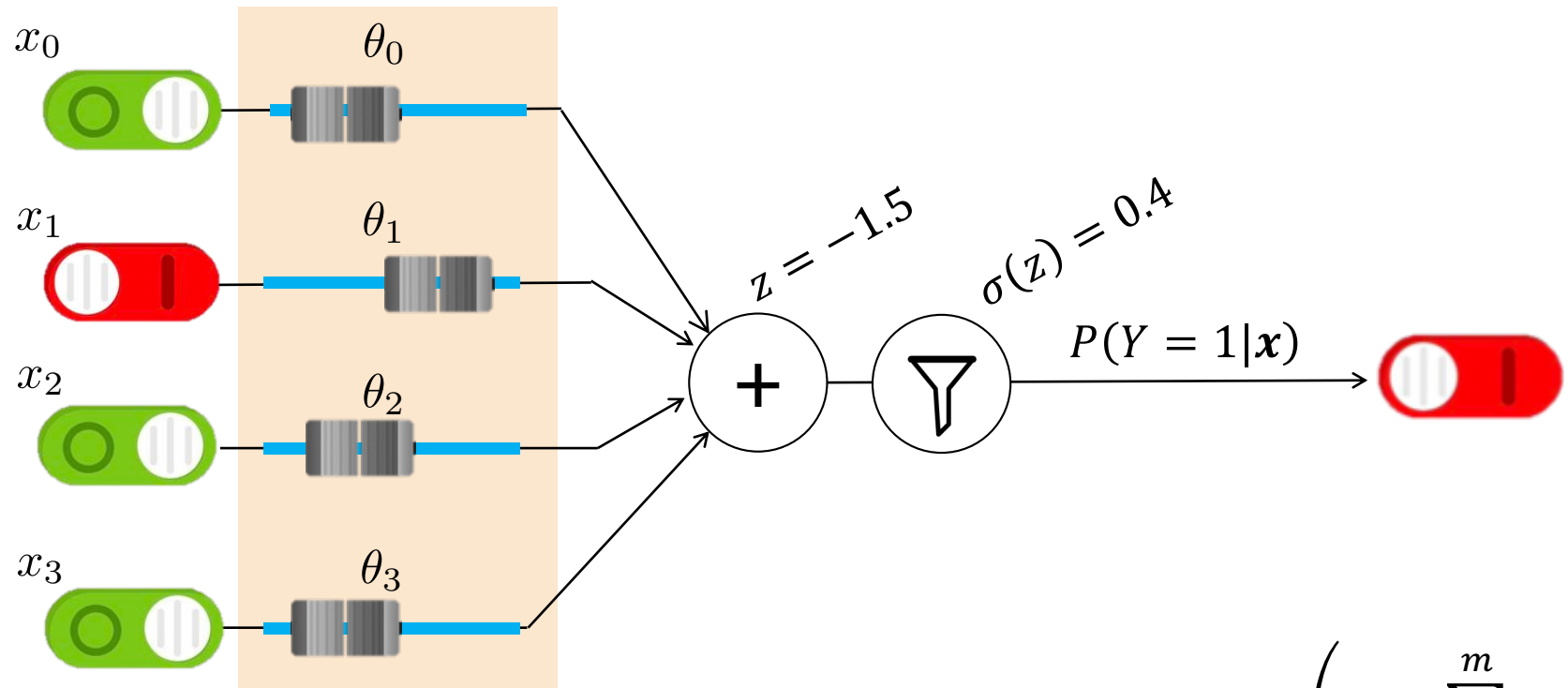
$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Parameters affect prediction



$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Parameters affect prediction



$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$

Parameters affect prediction

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\theta_0 + \sum_{j=1}^m \theta_j x_j \right)$$



$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma \left(\sum_{j=0}^m \theta_j x_j \right) = \sigma(\boldsymbol{\theta}^T \mathbf{x}) \quad \text{where } x_0 = 1$$

Logistic regression classifier

$$\hat{Y} = \arg \max_{y=\{0,1\}} P(Y|\mathbf{X})$$

$$P(Y = 1|\mathbf{X} = \mathbf{x}) = \sigma\left(\sum_{j=0}^m \theta_j x_j\right) = \sigma(\theta^T \mathbf{x})$$

Training

Estimate parameters
from training data

$$\theta = (\theta_0, \theta_1, \theta_2, \dots, \theta_m)$$

Testing

Given an observation $\mathbf{X} = (X_1, X_2, \dots, X_m)$, predict

$$\hat{Y} = \arg \max_{y=\{0,1\}} P(Y|\mathbf{X})$$



Training: The big picture

Logistic regression classifier

$$\hat{Y} = \arg \max_{y=\{0,1\}} P(Y|\mathbf{X})$$

$$P(Y = 1|\mathbf{X} = \mathbf{x}) = \sigma\left(\sum_{j=0}^m \theta_j x_j\right) = \sigma(\boldsymbol{\theta}^T \mathbf{x})$$

Training

Estimate parameters
from training data

$$\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \dots, \theta_m)$$

Choose $\boldsymbol{\theta}$ that optimizes some objective:

1. Determine objective function
2. Find gradient with respect to $\boldsymbol{\theta}$
3. Solve analytically by setting to 0, or solve computationally with gradient ascent

We are modeling $P(Y|X)$ directly, so we maximize the **conditional likelihood** of training data.

Estimating θ

1. Determine objective function

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \widehat{\mathbf{x}}^{(i)}, \theta)$$

2. Gradient w.r.t. θ_j , for $j = 0, 1, \dots, m$

3. Solve

- No analytical derivation of θ_{MLE} ...
- ...but can still compute θ_{MLE} with gradient ascent!

```
initialize x
repeat many times:
  compute gradient
  x +=  $\eta$  * gradient
```

1. Determine objective function

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma(\sum_{j=0}^m \theta_j x_j) \\ = \sigma(\theta^T \mathbf{x})$$

First: Interpret
conditional likelihood
with Logistic Regression

Second: Write a differentiable
expression for log conditional
likelihood

1. Determine objective function (interpret)

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma(\sum_{j=0}^m \theta_j x_j) = \sigma(\theta^T \mathbf{x})$$

Suppose you have $n = 2$ training datapoints:

$$(\mathbf{x}^{(1)}, 1), (\mathbf{x}^{(2)}, 0)$$

Consider the following expressions for a given θ :

A. $\sigma(\theta^T \mathbf{x}^{(1)}) \sigma(\theta^T \mathbf{x}^{(2)})$
 $P(Y=1 | X=\bar{x}^{(1)}) P(Y=1 | X=\bar{x}^{(2)})$

C. $\sigma(\theta^T \mathbf{x}^{(1)}) (1 - \sigma(\theta^T \mathbf{x}^{(2)}))$
 $P(Y=1 | X=\bar{x}^{(1)}) P(Y=0 | X=\bar{x}^{(2)})$

B. $(1 - \sigma(\theta^T \mathbf{x}^{(1)})) \sigma(\theta^T \mathbf{x}^{(2)})$
 $P(Y=0 | \dots) P(Y=1 | \dots)$

D. $(1 - \sigma(\theta^T \mathbf{x}^{(1)})) (1 - \sigma(\theta^T \mathbf{x}^{(2)}))$
 $P(Y=0 | \dots) P(Y=0 | \dots)$

1. Interpret the above expressions as probabilities.
2. If we let $\theta = \theta_{MLE}$, which probability should be the highest?

1. Determine objective function (write)

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma(\sum_{j=0}^m \theta_j x_j) \\ = \sigma(\theta^T \mathbf{x})$$

1. What is a differentiable expression for $P(Y = y | \mathbf{X} = \mathbf{x})$? $f(y^{(i)} | \mathbf{x}^{(i)}, \theta)$

$$P(Y = y | \mathbf{X} = \mathbf{x}) = \begin{cases} \sigma(\theta^T \mathbf{x}) & \text{if } y = 1 \\ 1 - \sigma(\theta^T \mathbf{x}) & \text{if } y = 0 \end{cases}$$

2. What is a differentiable expression for $LL(\theta)$, log conditional likelihood?

$$LL(\theta) = \log \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta)$$



1. Determine objective function (write)

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma(\sum_{j=0}^m \theta_j x_j) = \sigma(\theta^T \mathbf{x})$$

1. What is a differentiable expression for $P(Y = y | \mathbf{X} = \mathbf{x})$?

$B \sim \text{Ber}(p)$ $b=0,1$

Recall

Bernoulli MLE!

$$P(B=b) = p^b (1-p)^{1-b}$$

$$P(Y = y | \mathbf{X} = \mathbf{x}) = \begin{cases} \sigma(\theta^T \mathbf{x}) & \text{if } y = 1 \\ 1 - \sigma(\theta^T \mathbf{x}) & \text{if } y = 0 \end{cases}$$

$q = 1-p$

$$\left(\sigma(\theta^T \bar{\mathbf{x}})^y (1 - \sigma(\theta^T \bar{\mathbf{x}}))^{1-y} \right)$$

2. What is a differentiable expression for $LL(\theta)$, log conditional likelihood?

$$\sum_{i=1}^n \log \left(\sigma(\theta^T \bar{\mathbf{x}}^{(i)})^{y^{(i)}} (1 - \sigma(\theta^T \bar{\mathbf{x}}^{(i)}))^{1-y^{(i)}} \right)$$

$$LL(\theta) = \log \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta)$$

1. Determine objective function (write)

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$P(Y = 1 | \mathbf{X} = \mathbf{x}) = \sigma(\sum_{j=0}^m \theta_j x_j) \\ = \sigma(\theta^T \mathbf{x})$$

1. What is a differentiable expression for $P(Y = y | \mathbf{X} = \mathbf{x})$?

$$P(Y = y | \mathbf{X} = \mathbf{x}) = (\sigma(\theta^T \mathbf{x}))^y (1 - \sigma(\theta^T \mathbf{x}))^{1-y}$$

2. What is a differentiable expression for $LL(\theta)$, log conditional likelihood?

$$LL(\theta) = \sum_{i=1}^n y^{(i)} \log \sigma(\theta^T \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T \mathbf{x}^{(i)}))$$

linear regression a, b

2. Find gradient with respect to θ

$$\theta = (\theta_0, \theta_1, \theta_2, \theta_3, \dots, \theta_m) \quad \vec{x} = (x_1, x_2, x_3, \dots, x_m)$$

Optimization problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^n y^{(i)} \log \sigma(\theta^T \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T \mathbf{x}^{(i)}))$$

Gradient w.r.t. θ_j , for $j = 0, 1, \dots, m$:

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z) (1 - \sigma(z))$$

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^n [y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)})] x_j^{(i)}$$

(presented without proof)

$$\theta^T \mathbf{x}^{(i)} = \sum_{j=0}^m \theta_j x_j^{(i)}$$
$$\frac{\partial}{\partial \theta_j} = x_j^{(i)}$$

How do we interpret the gradient contribution of the i -th training datapoint?



2. Find gradient with respect to θ

Optimization
problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^n y^{(i)} \log \sigma(\theta^T \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T \mathbf{x}^{(i)}))$$

Gradient w.r.t. θ_j , for $j = 0, 1, \dots, m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^n [y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)})] x_j^{(i)} \quad (\text{presented without proof})$$

↑
scale by j-th feature

2. Find gradient with respect to θ

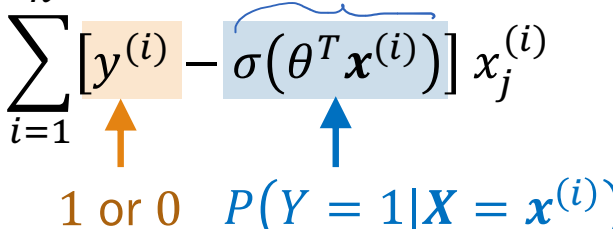
Optimization
problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^n y^{(i)} \log \sigma(\theta^T \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T \mathbf{x}^{(i)}))$$

Gradient w.r.t. θ_j , for $j = 0, 1, \dots, m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^n [y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)})] x_j^{(i)} \quad (\text{presented without proof})$$



2. Find gradient with respect to θ

Optimization
problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^n y^{(i)} \log \sigma(\theta^T \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T \mathbf{x}^{(i)}))$$

Gradient w.r.t. θ_j , for $j = 0, 1, \dots, m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^n \underbrace{[y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)})]}_{\text{(presented without proof)}} x_j^{(i)}$$

Suppose $y^{(i)} = 1$ (the true class label for i -th datapoint):

- If $\sigma(\theta^T \mathbf{x}^{(i)}) \geq 0.5$, correct
- If $\sigma(\theta^T \mathbf{x}^{(i)}) < 0.5$, incorrect \rightarrow change θ_j more

3. Solve

1. Optimization problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^n f(y^{(i)} | \mathbf{x}^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^n y^{(i)} \log \sigma(\theta^T \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T \mathbf{x}^{(i)}))$$

2. Gradient w.r.t. θ_j , for $j = 0, 1, \dots, m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^n [y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)})] x_j^{(i)}$$

3. Solve using gradient ascent



Training: The details

Training: Gradient ascent step

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^n [y^{(i)} - \sigma(\theta^T \mathbf{x}^{(i)})] x_j^{(i)} \quad \text{for } j = 0, 1, \dots, m$$

repeat many times:

for all thetas:

$$\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \frac{\partial LL(\theta^{\text{old}})}{\partial \theta_j^{\text{old}}}$$

Handwritten notes: $\theta_0, \theta_1, \theta_2, \dots, \theta_m$ (with θ_0 crossed out); $10, 100, 300, 1000 \rightarrow m$ change; σ (sigmoid (linear calibration)); η (learning rate)

$$= \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$$

What does this look like in code?

Training: Gradient Ascent

$$\text{for } j = 0, 1, \dots, m: \\ \text{Gradient Ascent Step } \theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$$

Linear Reg: $a, b = 0.0$

initialize $\theta_j = 0$ for $0 \leq j \leq m$ } $m+1$
repeat many times:

```
gradient[j] = 0 for  $0 \leq j \leq m$ 
```

```
// TODO: your code here
```

```
// compute all gradient[j]'s
```

```
// based on n training examples
```

```
 $\theta_j$  +=  $\eta$  * gradient[j] for all  $0 \leq j \leq m$ 
```

Training: Gradient Ascent

inner loop

for $j = 0, 1, \dots, m$:

Gradient Ascent Step

$$\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n \underbrace{[y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})]}_{\text{compute}} x_j^{(i)}$$

outer loop

```
initialize  $\theta_j = 0$  for  $0 \leq j \leq m$   
repeat many times:
```

```
  gradient[j] = 0 for  $0 \leq j \leq m$ 
```

```
  for each training example  $(x, y)$ : for i=1 to n
```

```
    for each  $0 \leq j \leq m$ :
```

```
      // update gradient[j] for  
      // current  $(x, y)$  example
```

```
     $\theta_j += \eta * \text{gradient}[j]$  for all  $0 \leq j \leq m$ 
```

Training: Gradient Ascent

inner loop

for $j = 0, 1, \dots, m$:

Gradient Ascent Step $\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n \underbrace{[y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})]}_{\text{compute}} x_j^{(i)}$



outer loop

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient[j] = 0 for $0 \leq j \leq m$

for each training example (\mathbf{x}, y) :

for each $0 \leq j \leq m$:

gradient[j] += $\left[y - \overbrace{\frac{1}{1 + e^{-\theta^T \mathbf{x}}}}^{\sigma(\theta^T \mathbf{x})} \right] x_j$

$\theta_j += \eta * \text{gradient}[j]$ for all $0 \leq j \leq m$

Some important details...

Training: Gradient Ascent

Gradient Ascent Step $\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient[j] = 0 for $0 \leq j \leq m$

for each training example (\mathbf{x}, y) :

for each $0 \leq j \leq m$:

$$\text{gradient}[j] += \left[y - \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \right] x_j$$

$\theta_j += \eta * \text{gradient}[j]$ for all $0 \leq j \leq m$

- Finish computing gradient with θ^{old} prior to any θ update

Training: Gradient Ascent

$$\text{Gradient Ascent Step } \theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$$

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient[j] = 0 for $0 \leq j \leq m$

for each training example (x, y) :

for each $0 \leq j \leq m$:

$$\text{gradient}[j] += \left[y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j$$

$\theta_j += \eta * \text{gradient}[j]$ for all $0 \leq j \leq m$

- Finish computing gradient with θ^{old} prior to any θ update
- Learning rate η is a constant you set before training

Training: Gradient Ascent


$$\text{Gradient Ascent Step } \theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$$

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

```
gradient[j] = 0 for  $0 \leq j \leq m$ 
```

```
for each training example  $(x, y)$ :
```

```
for each  $0 \leq j \leq m$ :
```

$$\text{gradient}[j] += \left[y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j$$


```
 $\theta_j += \eta * \text{gradient}[j]$  for all  $0 \leq j \leq m$ 
```

- Finish computing gradient with θ^{old} prior to any θ update
- Learning rate η is a constant you set before training
- x_j is j -th feature of input $\mathbf{x} = (x_1, \dots, x_m)$

Training: Gradient Ascent

$$\text{Gradient Ascent Step } \theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$$




initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

```
gradient[j] = 0 for  $0 \leq j \leq m$ 
```

```
for each training example  $(x, y)$ :
```

```
for each  $0 \leq j \leq m$ :
```

$$\text{gradient}[j] += \left[y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j$$


```
 $\theta_j += \eta * \text{gradient}[j]$  for all  $0 \leq j \leq m$ 
```

- Finish computing gradient with θ^{old} prior to any θ update
- Learning rate η is a constant you set before training
- x_j is j -th feature of input $\mathbf{x} = (x_1, \dots, x_m)$
- Insert $x_0 = 1$ before training

Training: Gradient Ascent

$$\text{Gradient Ascent Step } \theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^n [y^{(i)} - \sigma(\theta^{\text{old}T} \mathbf{x}^{(i)})] x_j^{(i)}$$

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

```
gradient[j] = 0 for  $0 \leq j \leq m$ 
```

```
for each training example  $(x, y)$ :
```

```
  for each  $0 \leq j \leq m$ :
```

```
    gradient[j] +=  $\left[ y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j$ 
```

```
 $\theta_j$  +=  $\eta$  * gradient[j] for all  $0 \leq j \leq m$ 
```

- Finish computing gradient with θ^{old} prior to any θ update
- Learning rate η is a constant you set before training
- x_j is j -th feature of input $\mathbf{x} = (x_1, \dots, x_m)$
- Insert $x_0 = 1$ before training

