26: Logistic Regression

Jerry Cain

May 25, 2022

Table of Contents

Linear to Logical: Preparation

1. Weighted sum

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

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

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

weighted sum

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

dot product

1. Weighted sum

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

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

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

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

$$g(\boldsymbol{X}) = \theta_0 X_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_m X_m$$

Define $X_0 = 1$

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

New $\boldsymbol{X} = (1, X_1, X_2, \dots, X_m)$

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

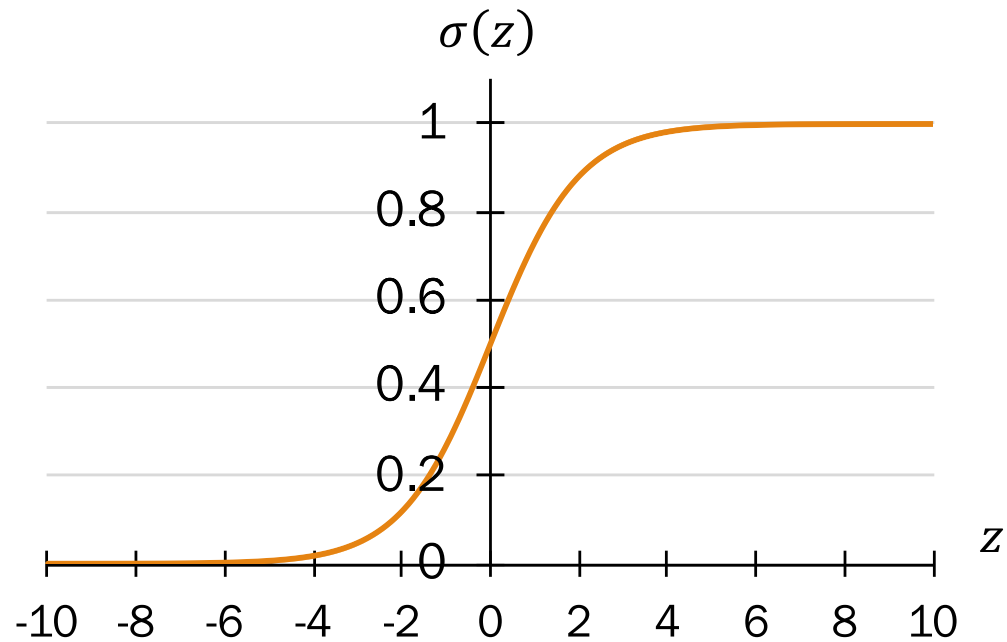
2. Sigmoid function $\sigma(z)$

- 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):

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

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

conditional likelihood of training data

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

log conditional likelihood

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

- 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)

$\boldsymbol{X}$ → $\boxed{\theta_0 + \sum_{j=1}^{m} \theta_j X_j}$ → $\hat{Y}$

$\hat{Y} = \theta_0 + \sum_{j=1}^{m} \theta_j X_j$

✅ $\boldsymbol{X}$ can be dependent

🤷‍♂️ Regression model ($\hat{Y} \in \mathbb{R}$, not discrete)

Naïve Bayes (Classification)

$\boldsymbol{X}$, Y → $\boxed{\hat{P}(\boldsymbol{X}|Y)\hat{P}(Y)}$ → $\hat{P}(\boldsymbol{X}, Y)$

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

✅ Tractable with NB assumption, but...

⚠️ Realistically, X_j features not always conditionally independent

🤷‍♂️ Actually models $P(\boldsymbol{X}, Y)$, not $P(Y|\boldsymbol{X})$?

Logistic Regression

$$\boldsymbol{X} \Rightarrow \boxed{\theta_0 + \sum_{j=1}^{m} \theta_j X_j} \xrightarrow{z} \boxed{\text{sigmoid function} \quad \sigma(z) = \frac{1}{1+e^{-z}}} \Rightarrow P(Y=1|\boldsymbol{X})$$

Logistic Regression Model:

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

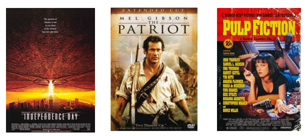
Predict $\hat{Y}$ as the more likely Y given our observation $\boldsymbol{X}=\boldsymbol{x}$:

$$\hat{Y} = \underset{y=\{0,1\}}{\arg\max}\, P(Y \mid \boldsymbol{X})$$

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

Logistic Regression

$\boldsymbol{x} = [0,1,1]$



θ parameter

0.81

$\boldsymbol{X}$

input features

$P(Y = 1|\boldsymbol{X} = \boldsymbol{x})$

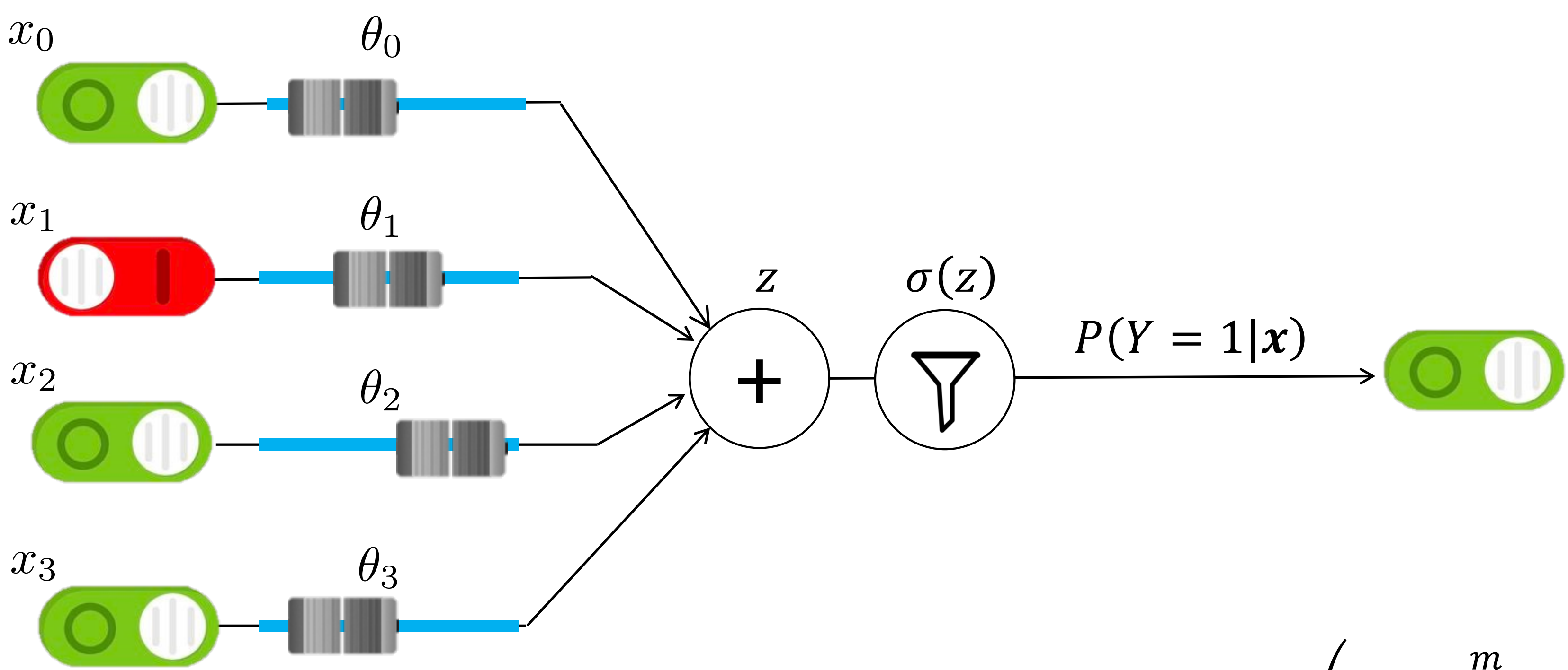
conditional likelihood

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

Logistic Regression: Key Metaphor

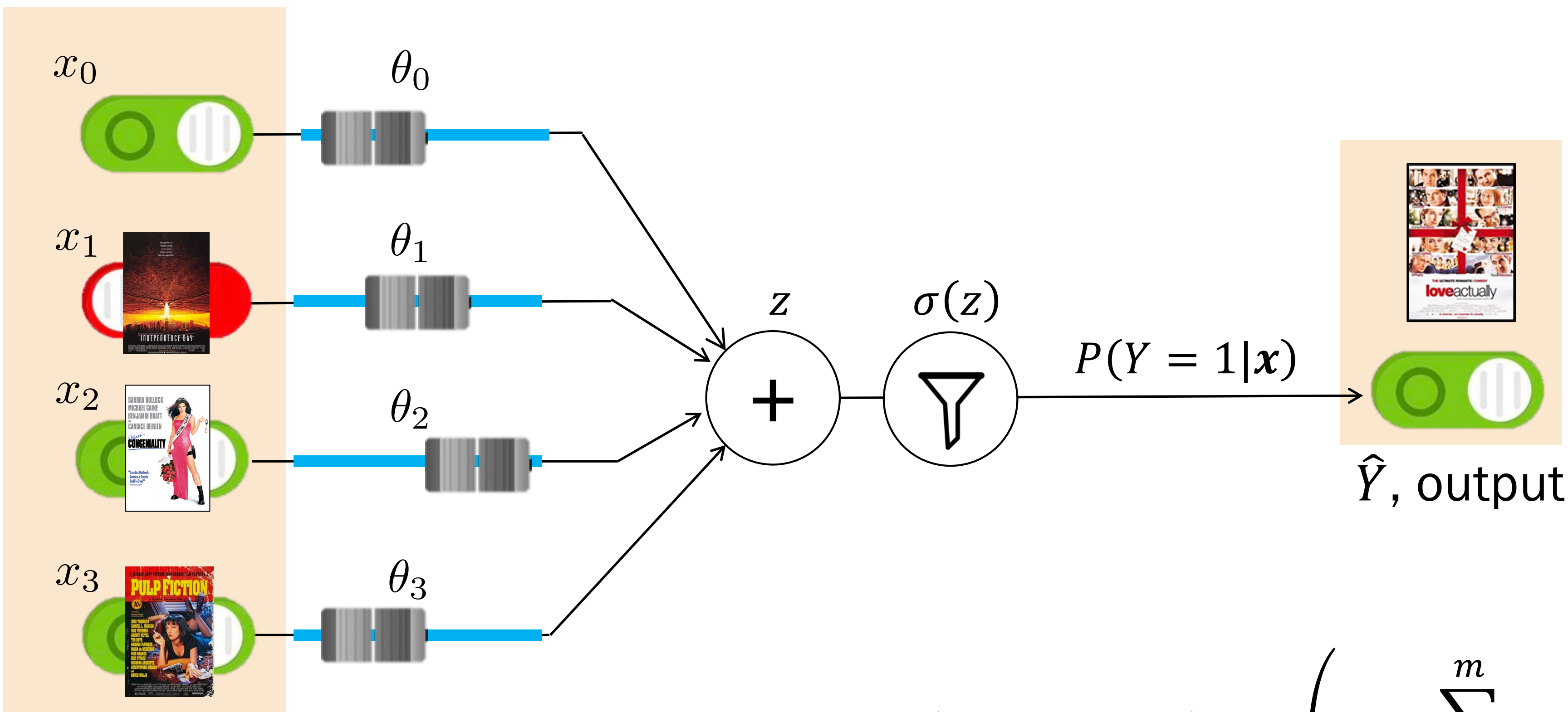
θ parameter

Logistic Regression: Key Metaphor



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

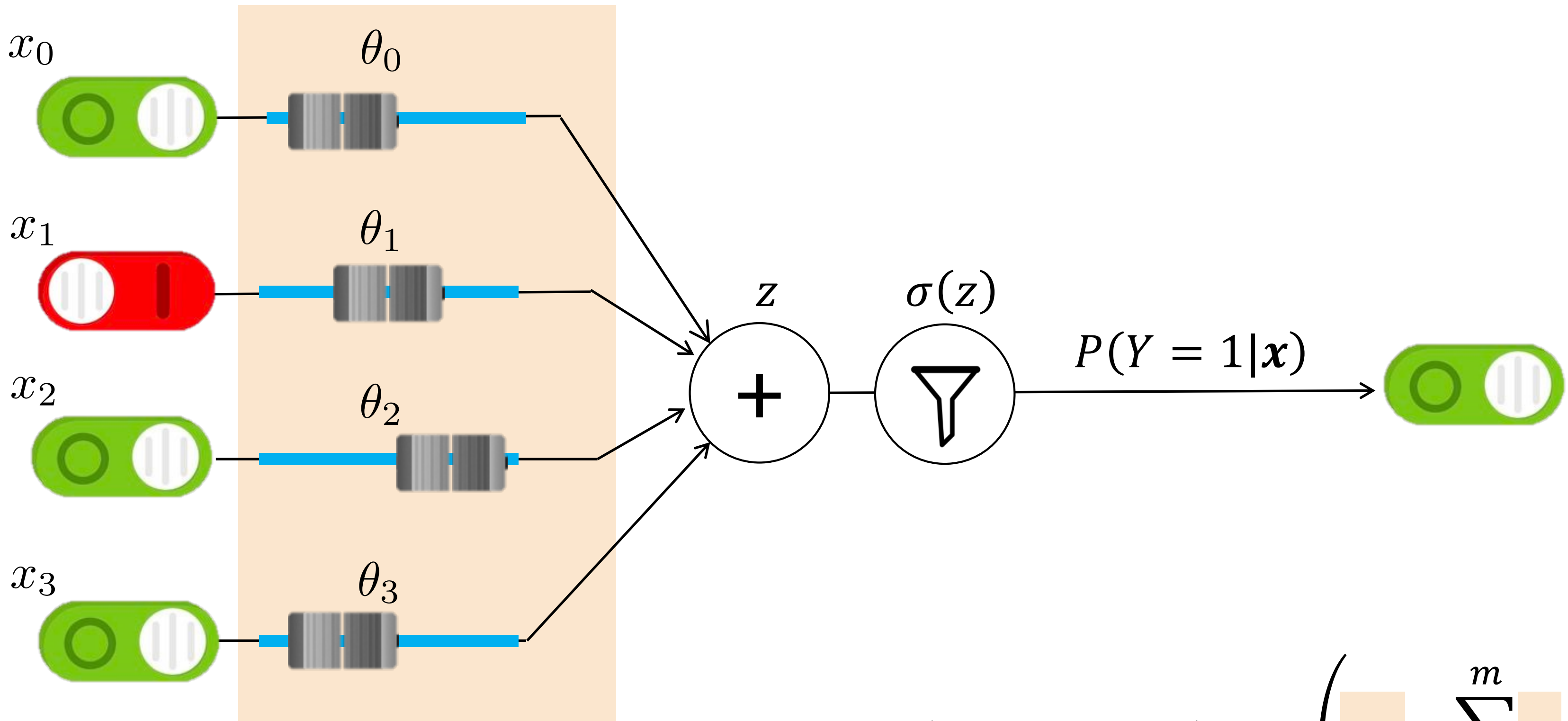
Logistic Regression: Key Metaphor



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

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

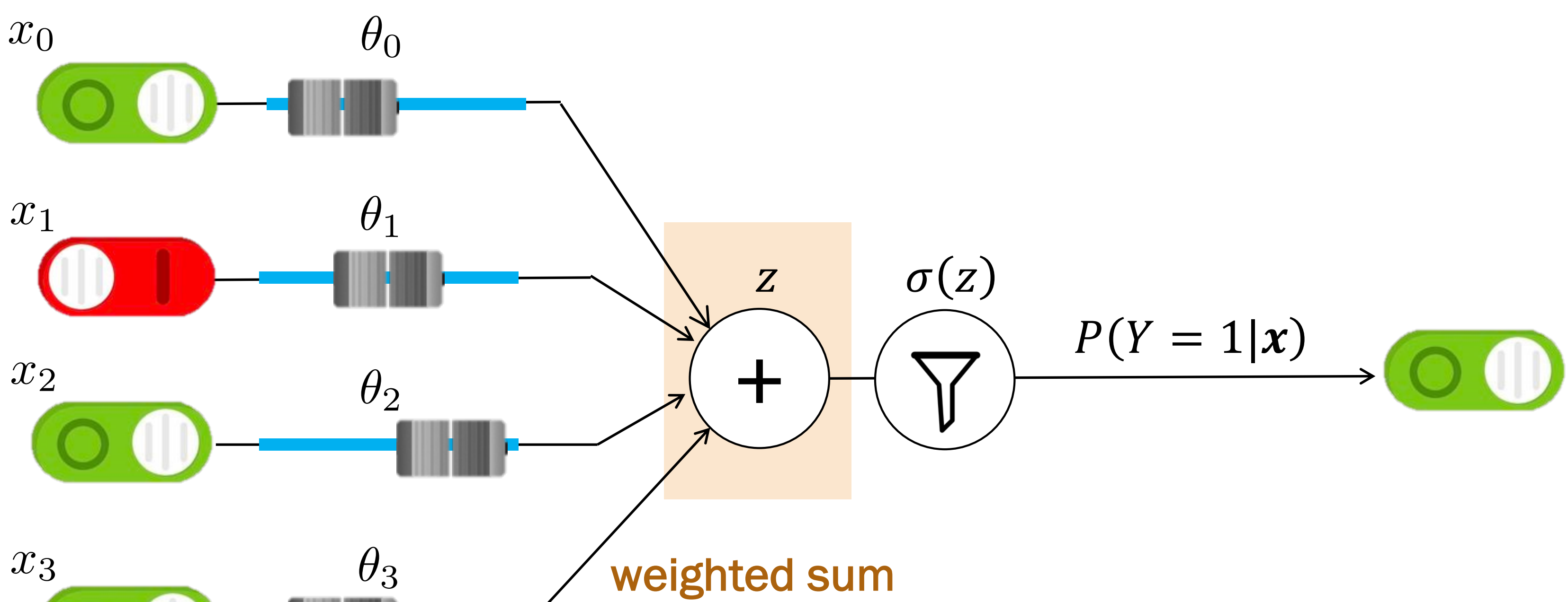
Components of Logistic Regression



θ weights
(aka parameters)

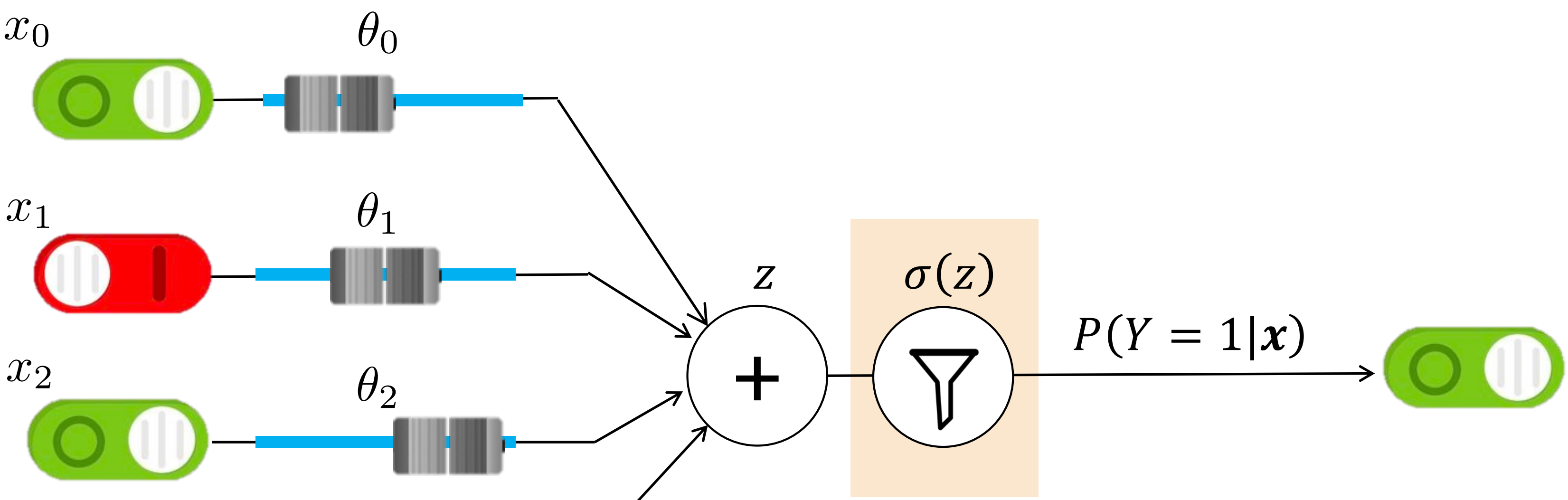
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

Components of Logistic Regression



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

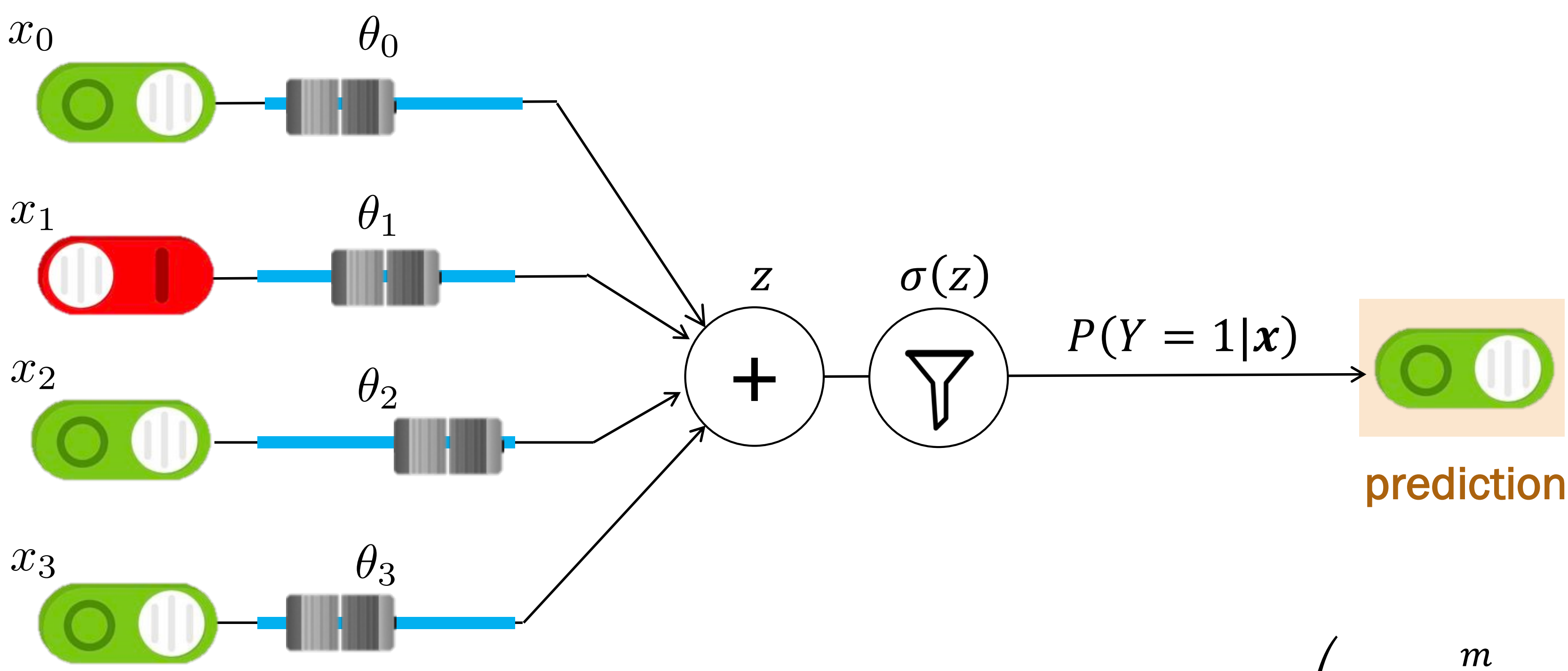
Components of Logistic Regression



squashing function
between 0 and 1

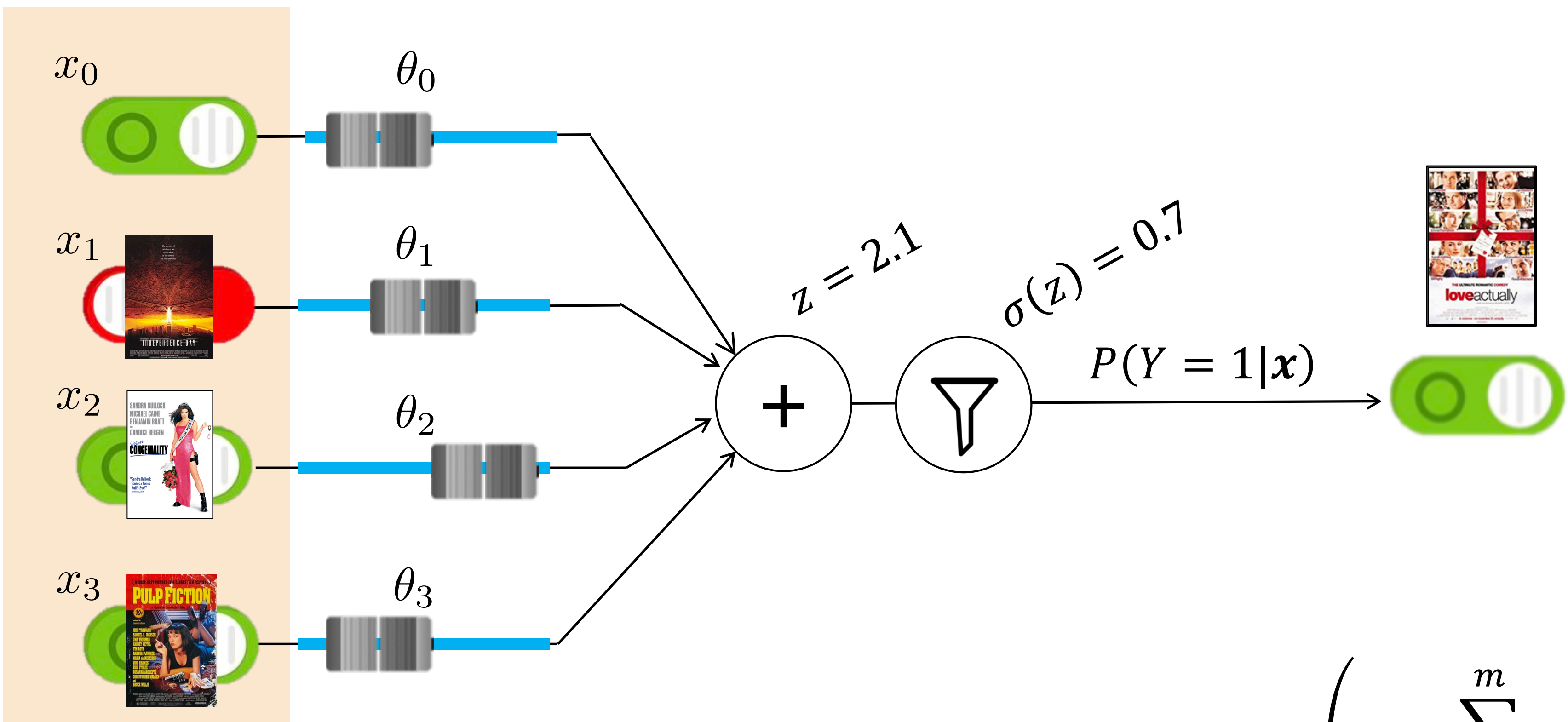
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

Components of Logistic Regression



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

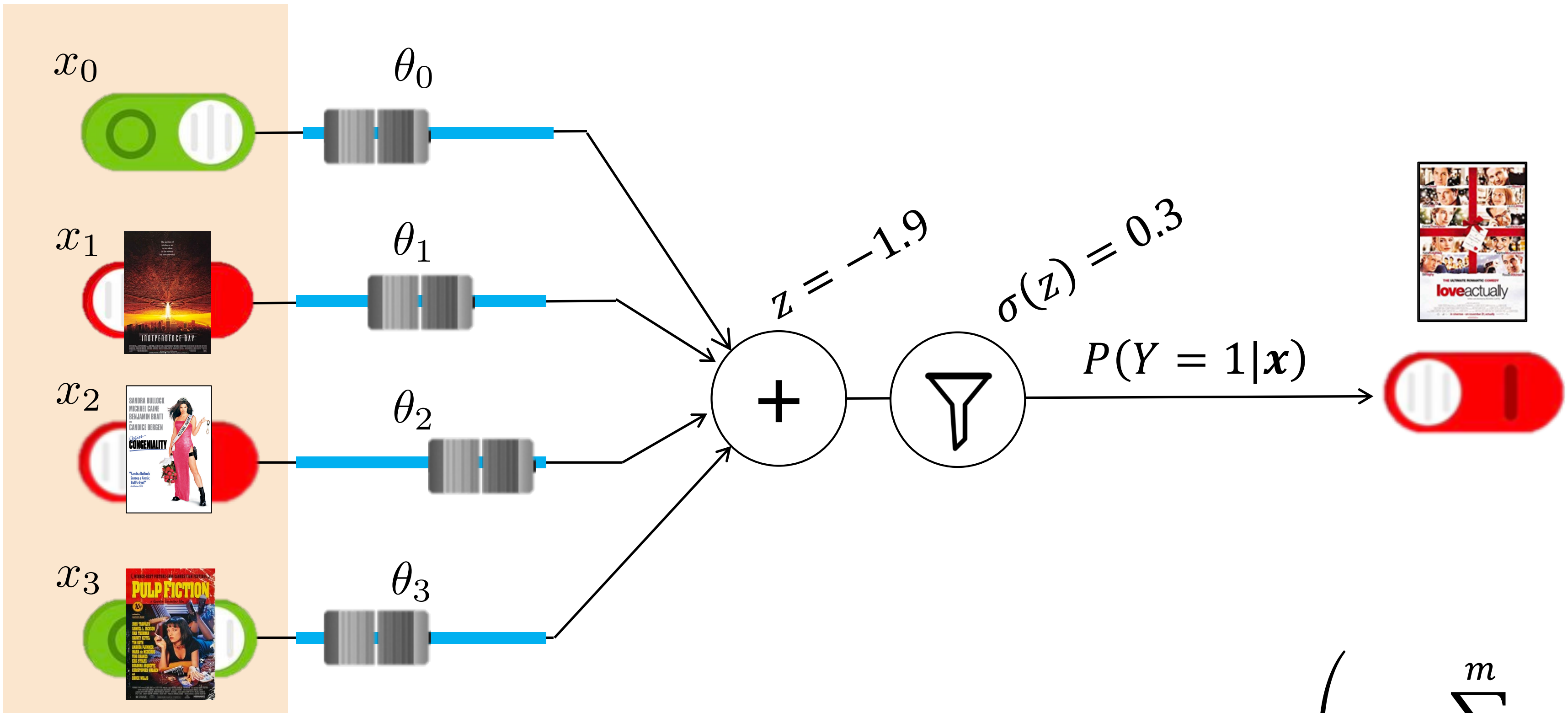
Different predictions for different inputs



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

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

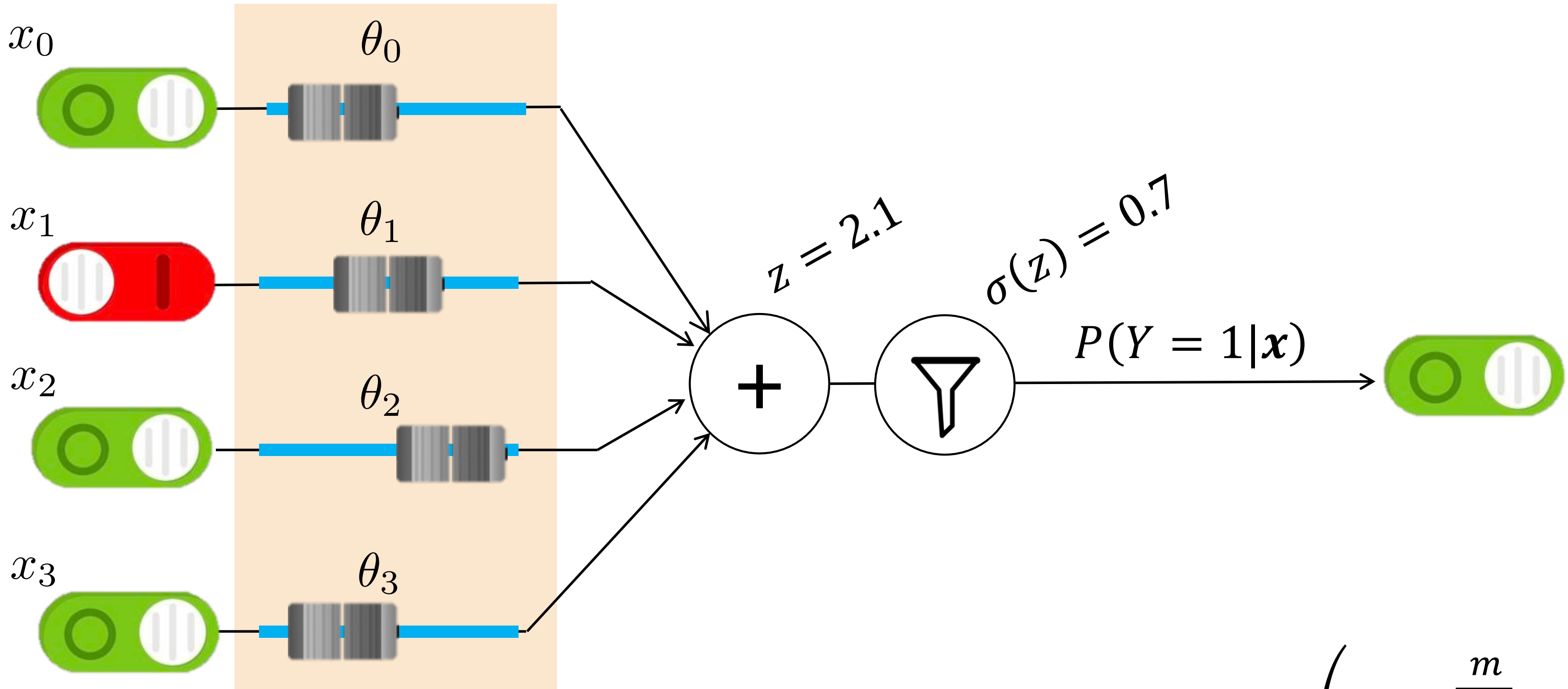
Different predictions for different inputs



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

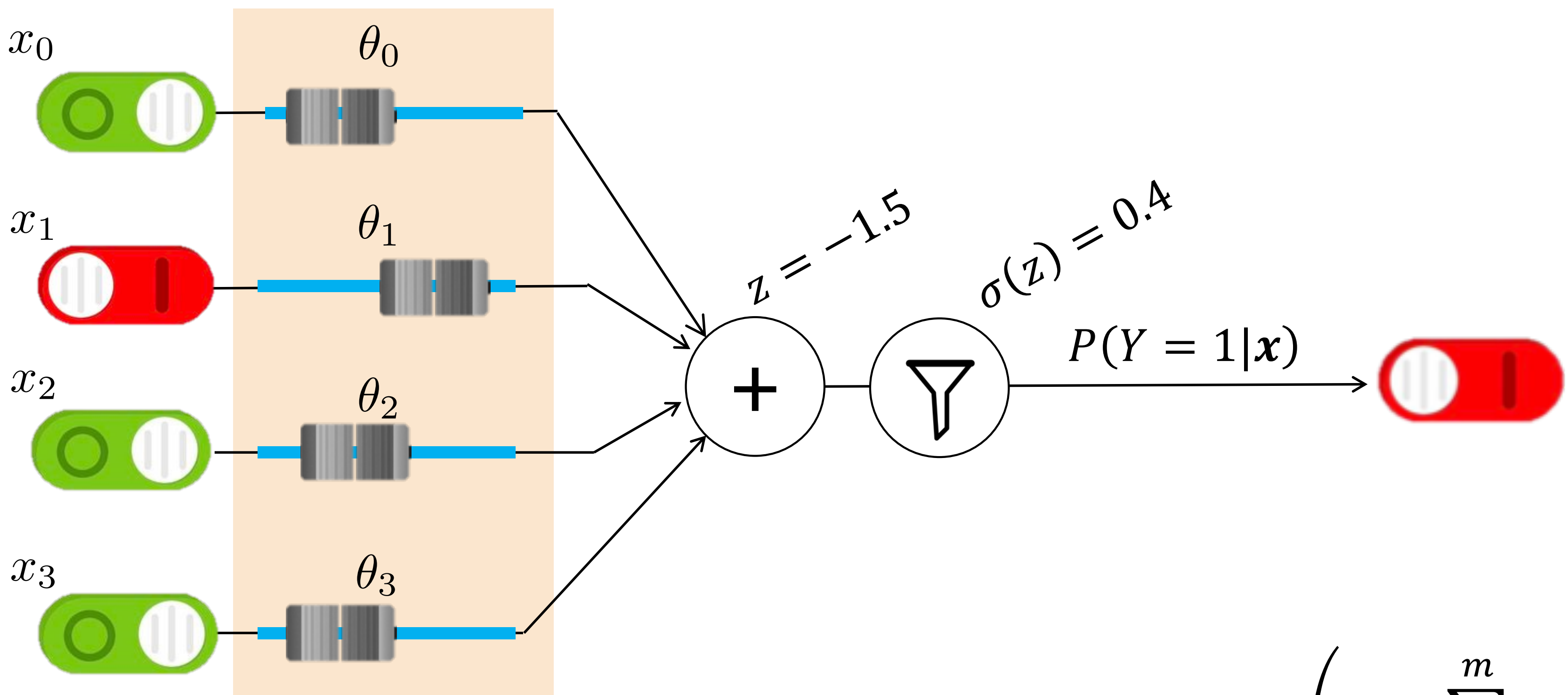
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

Parameters affect prediction



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

Parameters affect prediction



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

Parameters affect prediction

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

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

Logistic regression classifier

$$\hat{Y} = \underset{y=\{0,1\}}{\arg\max}\, P(Y|\boldsymbol{X})$$

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

Training

Estimate parameters from training data

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

Testing

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

$$\hat{Y} = \underset{y=\{0,1\}}{\arg\max}\, P(Y|\boldsymbol{X})$$

Training:
The big picture

Logistic regression classifier

$$\hat{Y} = \underset{y=\{0,1\}}{\arg\max}\, P(Y|\boldsymbol{X})$$

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

Training

Estimate parameters from training data

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

Choose θ that optimizes some objective:

1. Determine objective function
2. Find gradient with respect to θ
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

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right)$$

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

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 += η * gradient
```

1. Determine objective function

$$\theta_{MLE} = \boxed{\arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right)} = \boxed{\arg\max_{\theta} \; LL(\theta)}$$

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

First: Interpret conditional likelihood with Logistic Regression

Second: Write a differentiable expression for log conditional likelihood

1. Determine objective function (interpret)

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

Suppose you have $n = 2$ training datapoints: $\left(\boldsymbol{x}^{(1)}, 1\right), \left(\boldsymbol{x}^{(2)}, 0\right)$

Consider the following expressions for a given θ:

A. $\sigma\left(\theta^T \boldsymbol{x}^{(1)}\right) \sigma\left(\theta^T \boldsymbol{x}^{(2)}\right)$

B. $\left(1 - \sigma\left(\theta^T \boldsymbol{x}^{(1)}\right)\right) \sigma\left(\theta^T \boldsymbol{x}^{(2)}\right)$

C. $\sigma\left(\theta^T \boldsymbol{x}^{(1)}\right) \left(1 - \sigma\left(\theta^T \boldsymbol{x}^{(2)}\right)\right)$

D. $\left(1 - \sigma\left(\theta^T \boldsymbol{x}^{(1)}\right)\right) \left(1 - \sigma\left(\theta^T \boldsymbol{x}^{(2)}\right)\right)$

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)

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

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

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

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

$$LL(\theta) = \log \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right)$$

1. Determine objective function (write)

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \middle| \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

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

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

Recall
Bernoulli MLE!

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

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

1. Determine objective function (write)

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

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

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

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

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

2. Find gradient with respect to θ

Optimization problem:

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

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

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

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

(presented without proof)

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

2. Find gradient with respect to θ

Optimization problem:

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

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

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

(derived later)

scale by j-th feature

2. Find gradient with respect to θ

Optimization problem:

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

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

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \left[\underbrace{y^{(i)}}_{\text{1 or 0}} - \underbrace{\sigma\left(\theta^T \boldsymbol{x}^{(i)}\right)}_{P\left(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}^{(i)}\right)} \right] x_j^{(i)}$$

(presented without proof)

2. Find gradient with respect to θ

Optimization problem:

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

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

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

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

(derived later)

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

- If $\sigma\left(\theta^T \boldsymbol{x}^{(i)}\right) \geq 0.5$, correct
- If $\sigma\left(\theta^T \boldsymbol{x}^{(i)}\right) < 0.5$, incorrect → change θ_j more

3. Solve

1. Optimization problem:

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right) = \arg\max_{\theta} \; LL(\theta)$$

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

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

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

3. Solve using gradient ascent

Training: The details

Training: Gradient ascent step

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \left[y^{(i)} - \sigma\left(\theta^T \boldsymbol{x}^{(i)}\right) \right] x_j^{(i)} \qquad \text{for } j = 0, 1, \ldots, m$$

```
repeat many times:

    for all thetas:
```

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

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

What does this look like in code?

Training: Gradient Ascent

Gradient Ascent Step

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

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

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m

    // TODO: your code here

    // compute all gradient[j]'s
    // based on n training examples

    θ_j += η  * gradient[j] for all 0 ≤ j ≤ m
```

Training: Gradient Ascent

inner loop → for $j = 0, 1, \ldots, m$:

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

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example (x,y):
        for each 0 ≤ j ≤ m:
            // update gradient[j] for
            // current (x,y) example
    θ_j += η  * gradient[j] for all 0 ≤ j ≤ m
```

Training: Gradient Ascent

inner loop → for $j = 0, 1, \ldots, m$:

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

outer loop

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example (x,y):
        for each 0 ≤ j ≤ m:
            gradient[j] +=  [y − 1/(1 + e^(−θ^T x))] x_j
    θ_j += η * gradient[j] for all 0 ≤ j ≤ m
```

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

Some important details...

Training: Gradient Ascent

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

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example (x,y):
        for each 0 ≤ j ≤ m:
            gradient[j] += [y − 1/(1 + e^(−θ^T x))] x_j
    θ_j += η * gradient[j] for all 0 ≤ j ≤ m
```

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

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

Training: Gradient Ascent

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

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example (x,y):
        for each 0 ≤ j ≤ m:
            gradient[j] += [y − 1/(1 + e^(−θ^T x))] x_j
    θ_j += η * gradient[j] for all 0 ≤ j ≤ m
```

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

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

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

Training: Gradient Ascent

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

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example (x,y):
        for each 0 ≤ j ≤ m:
            gradient[j] += [y − 1/(1 + e^(−θ^T x))] x_j
    θ_j += η * gradient[j] for all 0 ≤ j ≤ m
```

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

- 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 $\boldsymbol{x} = (x_1, \ldots, x_m)$

Training: Gradient Ascent

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

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example (x,y):
        for each 0 ≤ j ≤ m:
            gradient[j] += [y − 1/(1 + e^(−θ^T x))] x_j
    θ_j += η * gradient[j] for all 0 ≤ j ≤ m
```

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

- 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 $\boldsymbol{x} = (x_1, \ldots, x_m)$
- Insert $x_0 = 1$ before training

Training: Gradient Ascent

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

initialize θ_j = 0 for 0 ≤ j ≤ m
repeat many times:
    gradient[j] = 0 for 0 ≤ j ≤ m
    for each training example $(\boldsymbol{x}, y)$:
        for each 0 ≤ j ≤ m:
            gradient[j] += $\left[y - \frac{1}{1 + e^{-\theta^T \boldsymbol{x}}} \right] x_j$
    θ_j += η * gradient[j] for all 0 ≤ j ≤ 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 $\boldsymbol{x} = (x_1, \dots, x_m)$
- Insert $x_0 = 1$ before training