25: Logistic Regression

Jerry Cain
May 29, 2024

Lecture Discussion on Ed

Linear to Logical: Preamble

1. Weighted sum

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

Recall the linear regression model, where $\boldsymbol{X} = (X_1, X_2, \ldots, X_m)$ 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?

$$g(\boldsymbol{X}) = \theta_0 X_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_m X_m \qquad \text{Define } X_0 = 1$$

$$= \theta^T \boldsymbol{X} \qquad \text{New } \boldsymbol{X} = (1, X_1, X_2, \ldots, X_m)$$

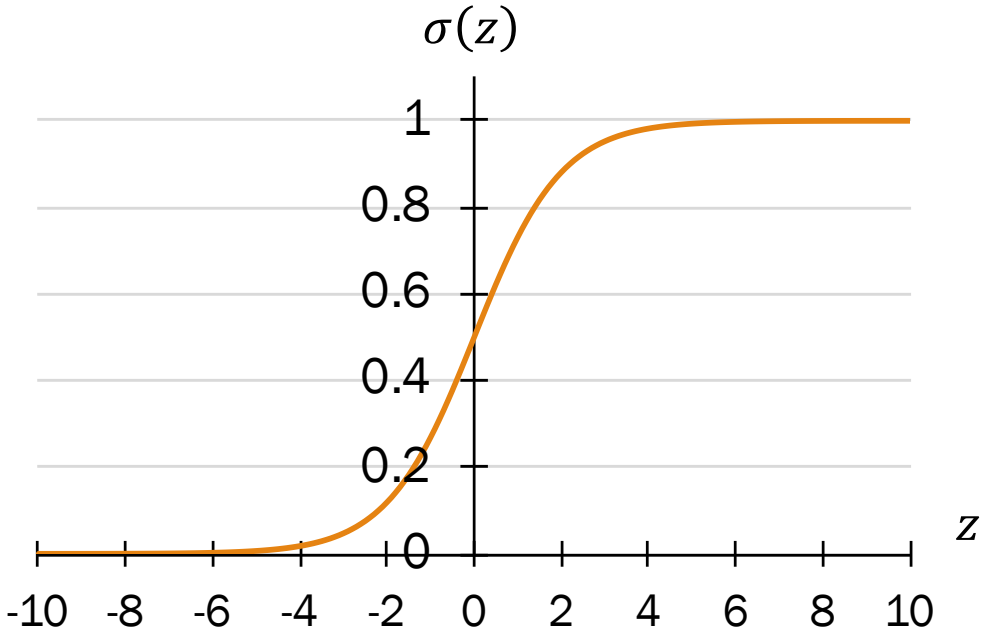
Prepending $X_0 = 1$ to each feature vector $\boldsymbol{X}$ simplifies matrix operators.

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 that expresses a belief that something is true.

$\sigma(z)$ 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
- log conditional likelihood is also written as $LL(\theta)$

Logistic Regression

Prediction models so far

Linear Regression (Regression)

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

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

✅ $\boldsymbol{X}$ can be dependent

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

Naïve Bayes (Classification)

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

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

✅ Tractable with NB assumption, but…

⚠️ Realistically, X_j features aren't always conditionally independent

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

Logistic Regression

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

Logistic Regression Model:

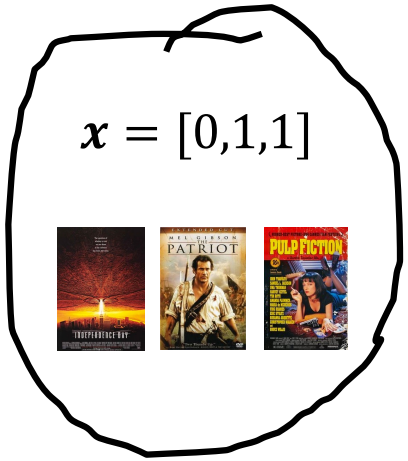
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

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

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

- 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



θ parameter

0.81

$\boldsymbol{X}$
input features

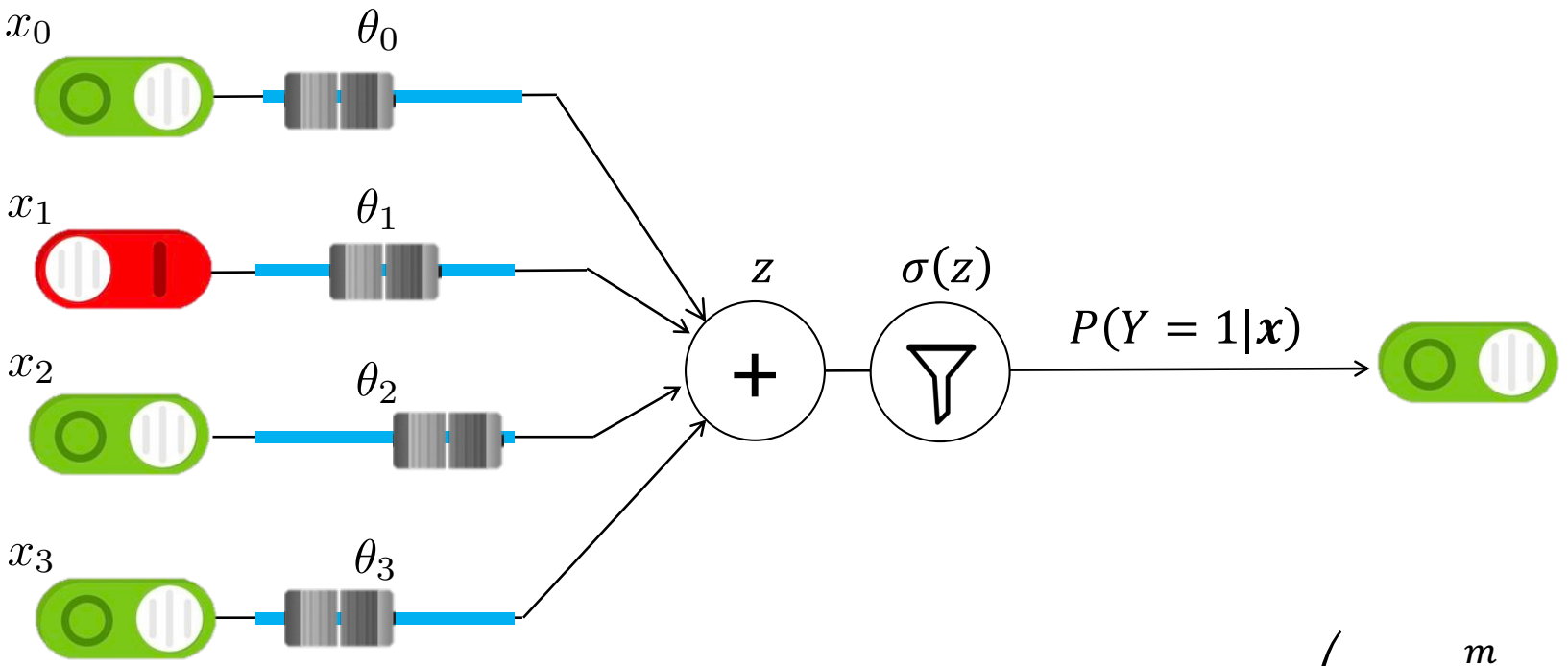
$P(Y = 1|\boldsymbol{X} = \boldsymbol{x})$
conditional likelihood

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

Logistic Regression: Key Metaphor

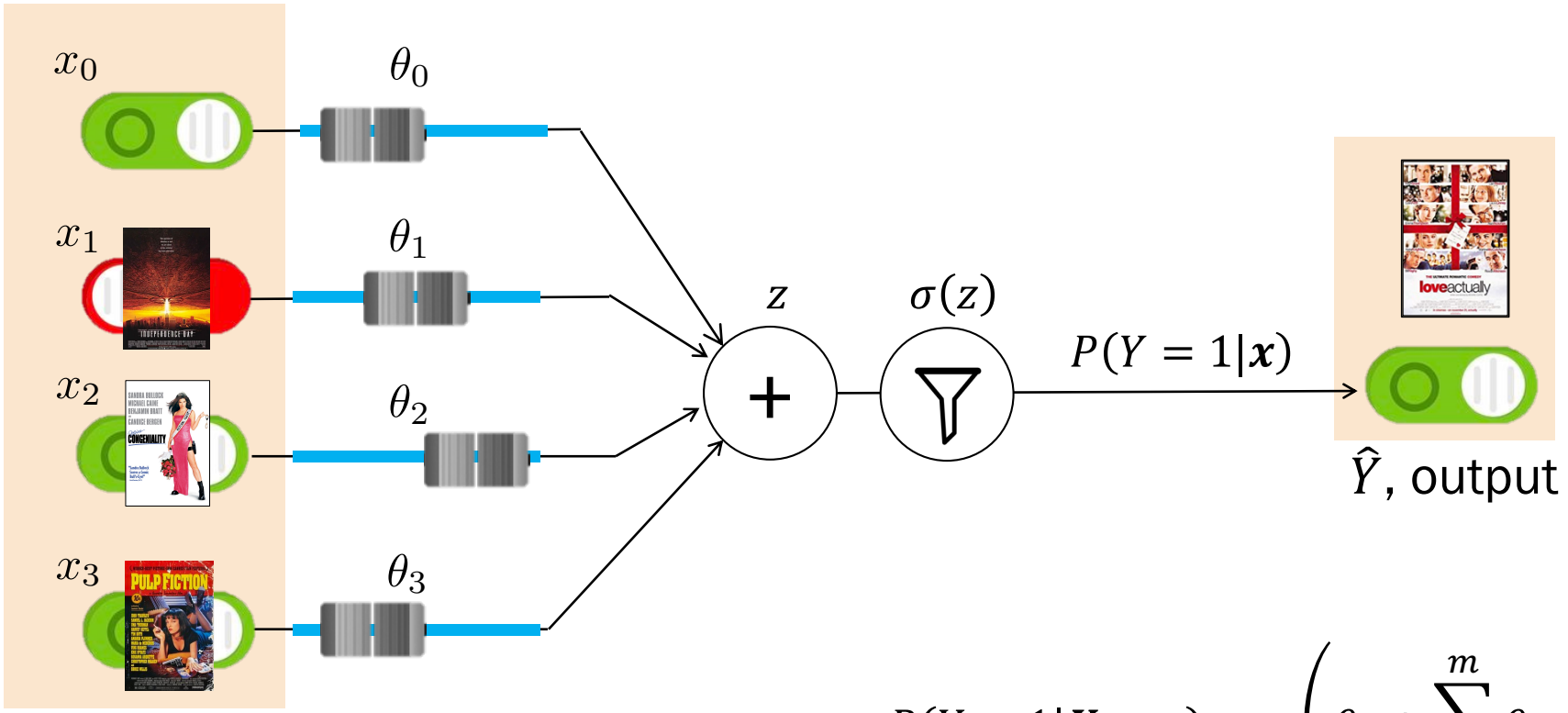
θ parameter

Logistic Regression: Key Metaphor



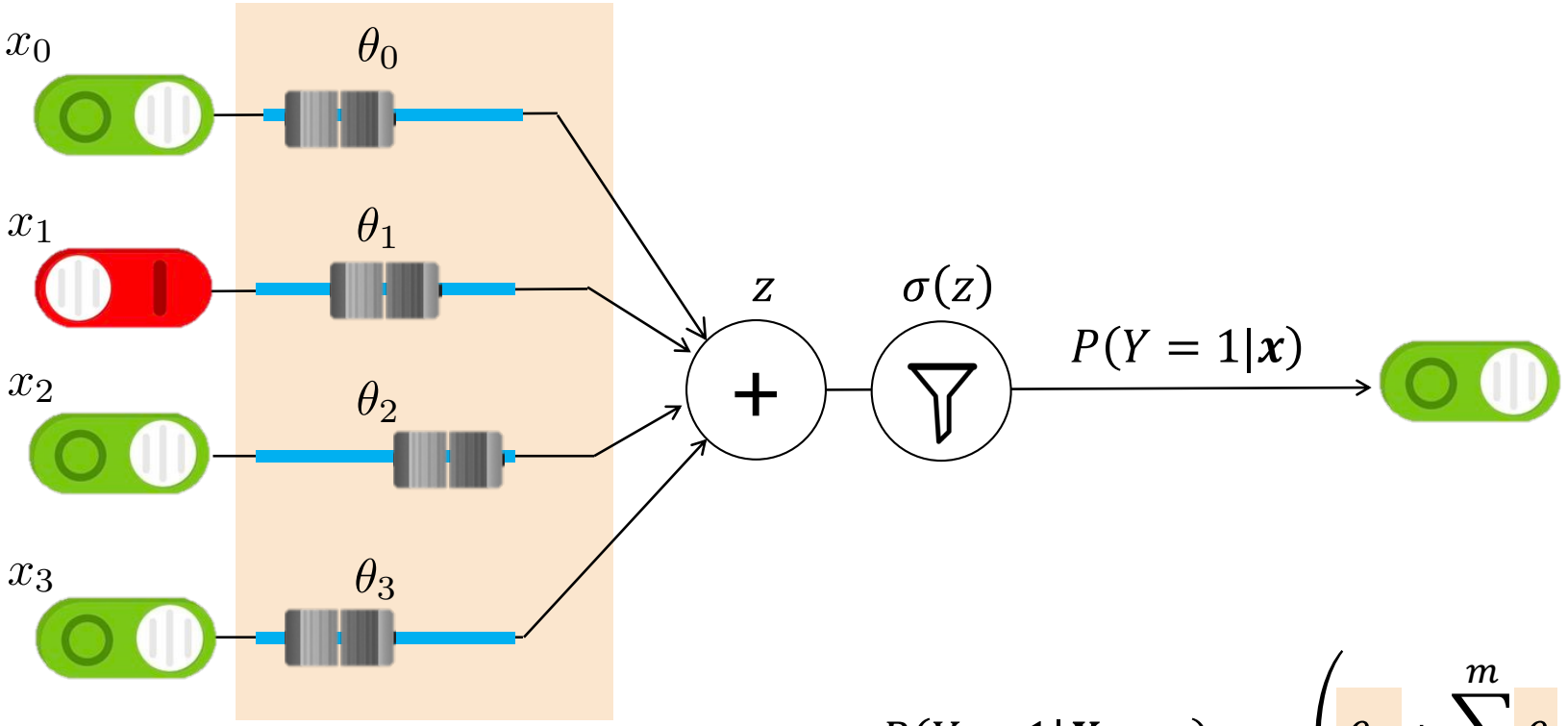
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

Logistic Regression: Key Metaphor



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

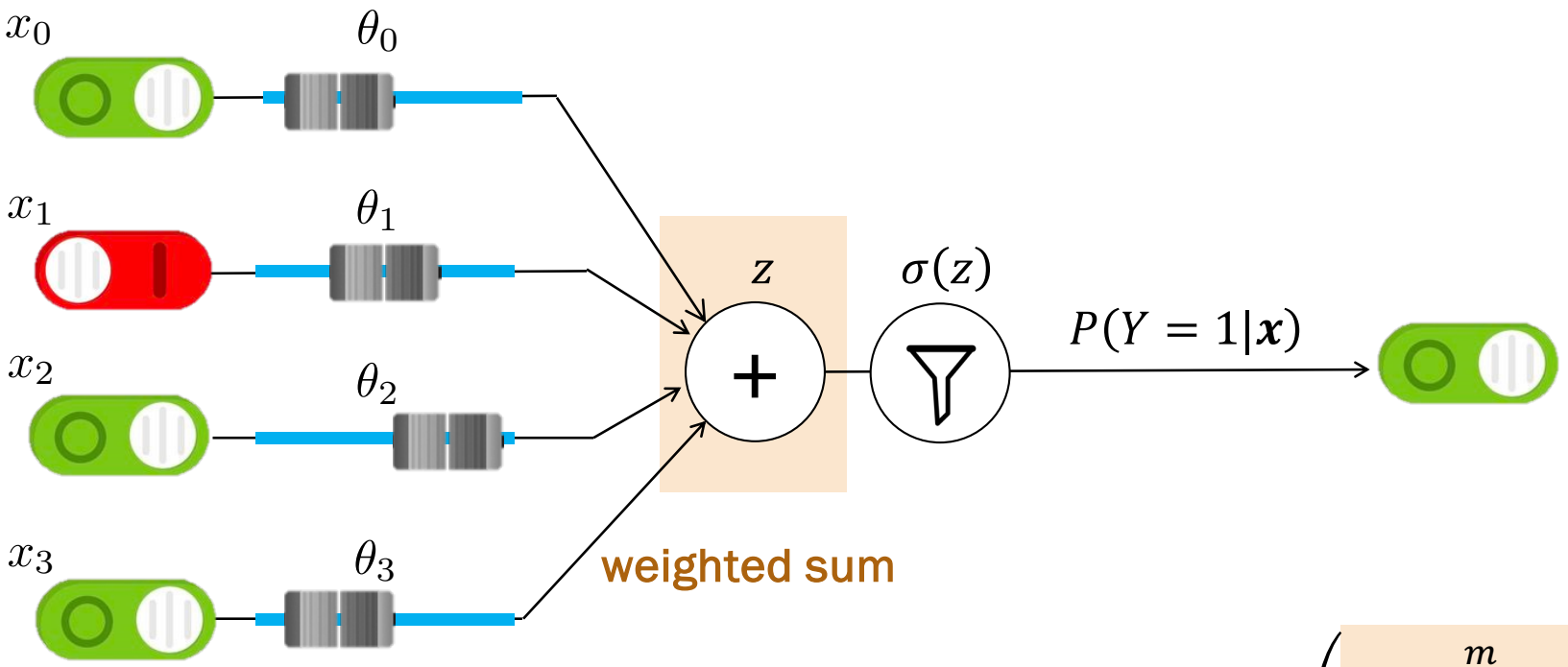
Components of Logistic Regression



θ weights
(aka parameters)

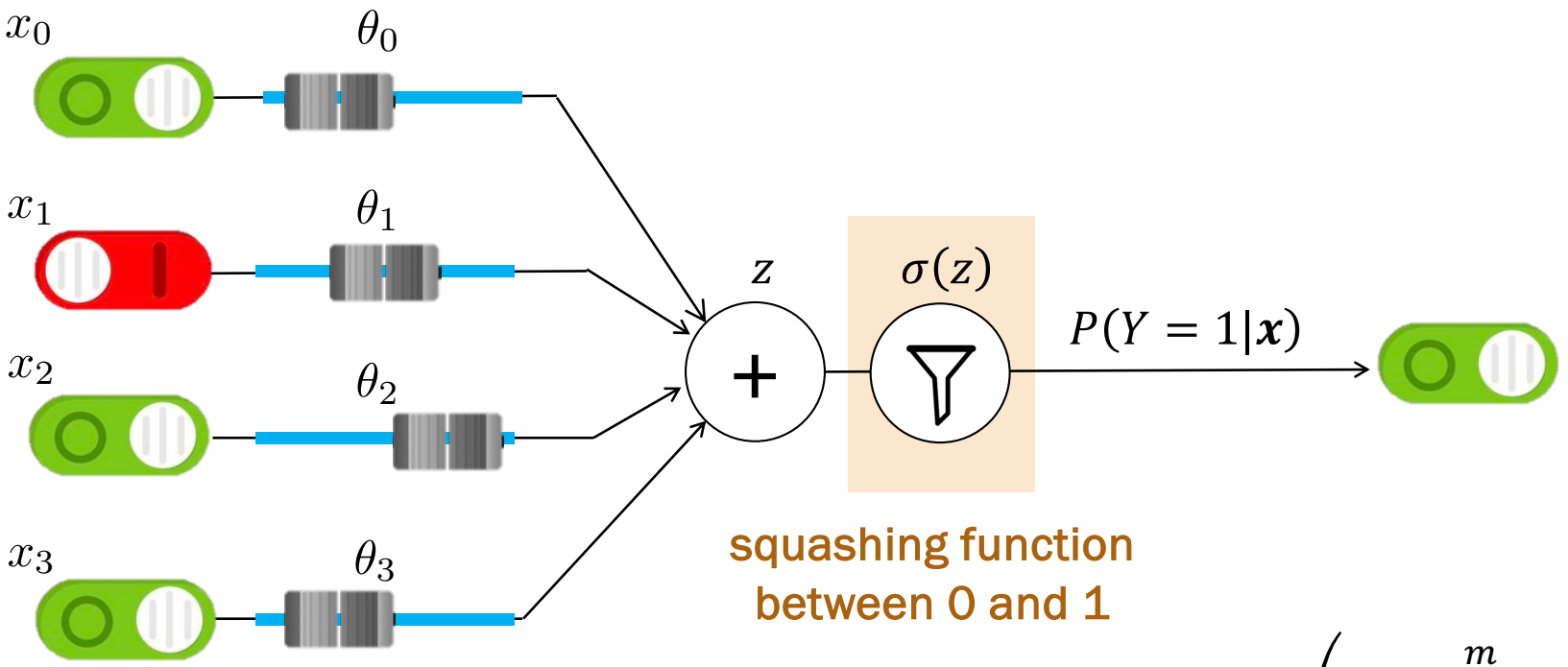
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

Components of Logistic Regression



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

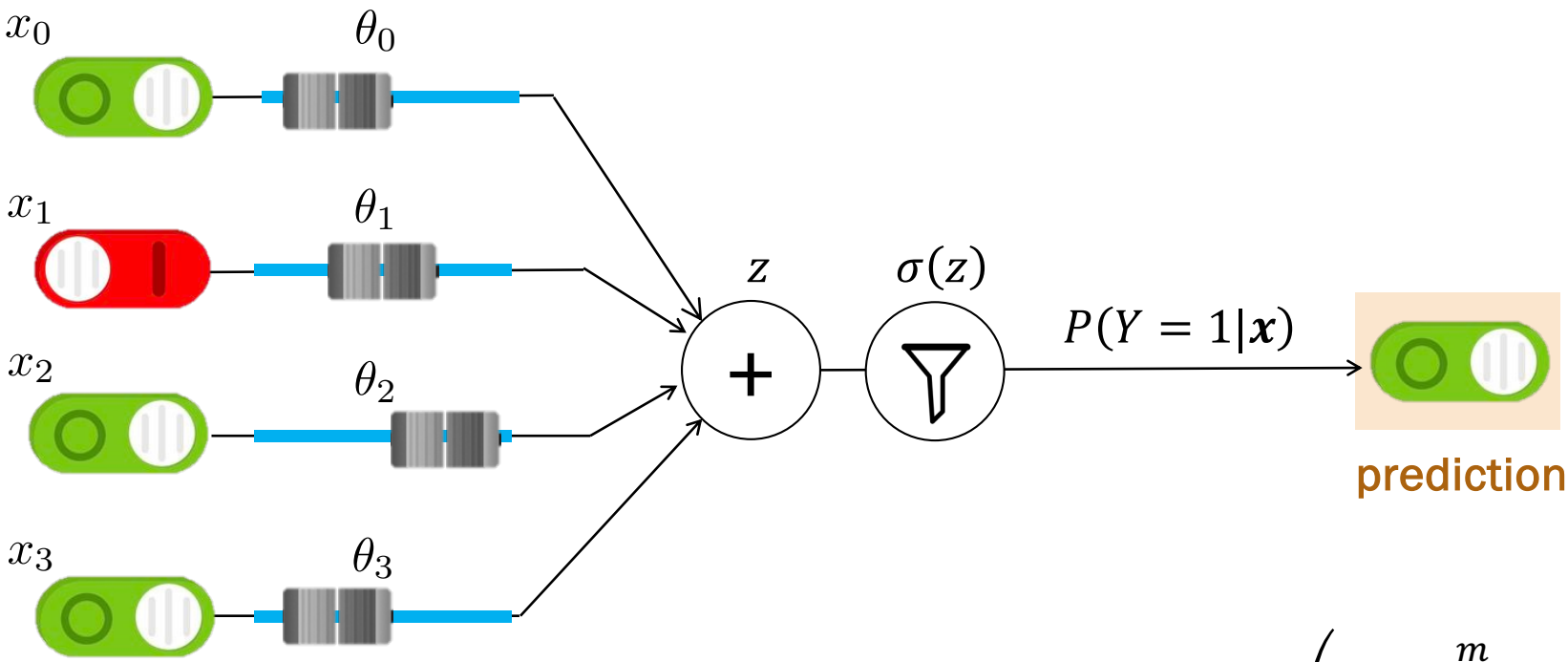
Components of Logistic Regression



squashing function between 0 and 1

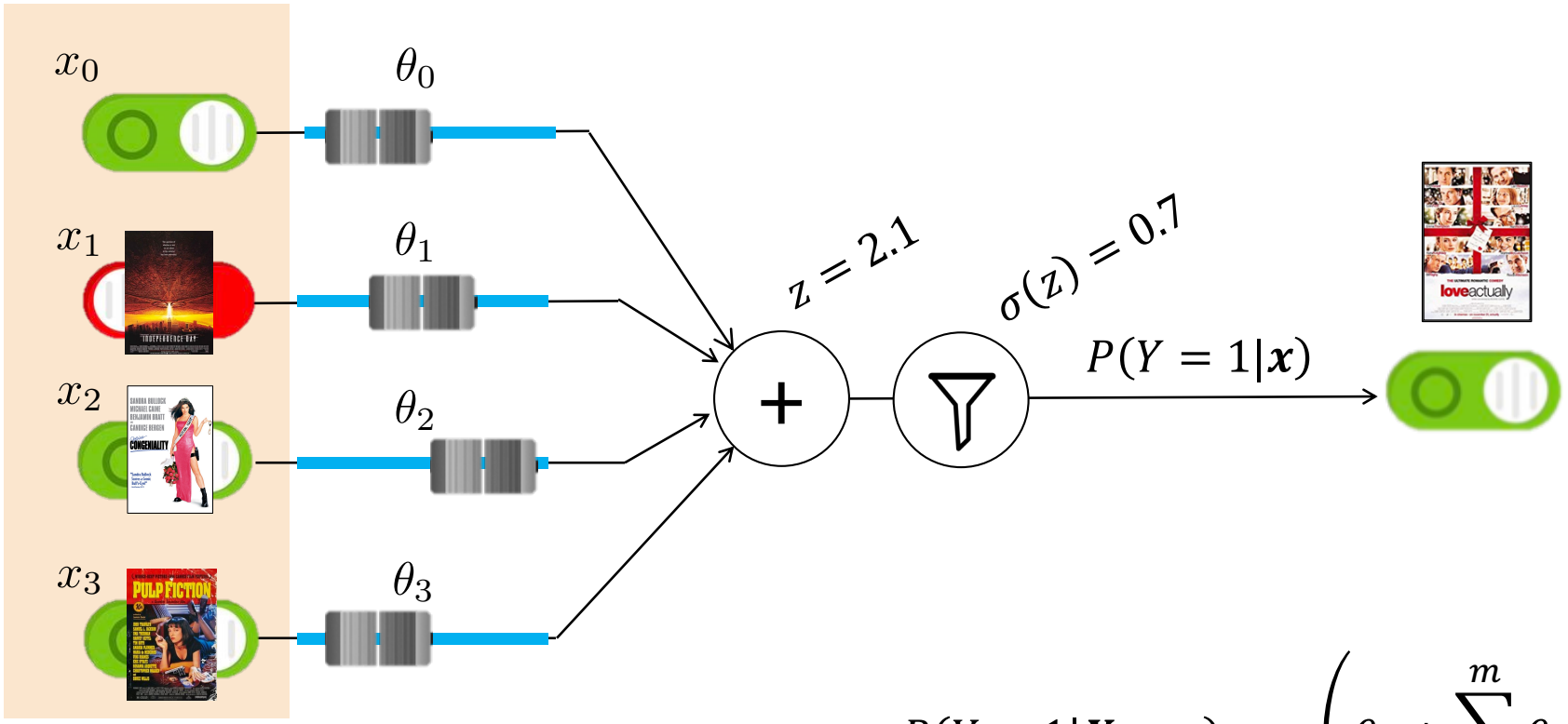
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

Components of Logistic Regression



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

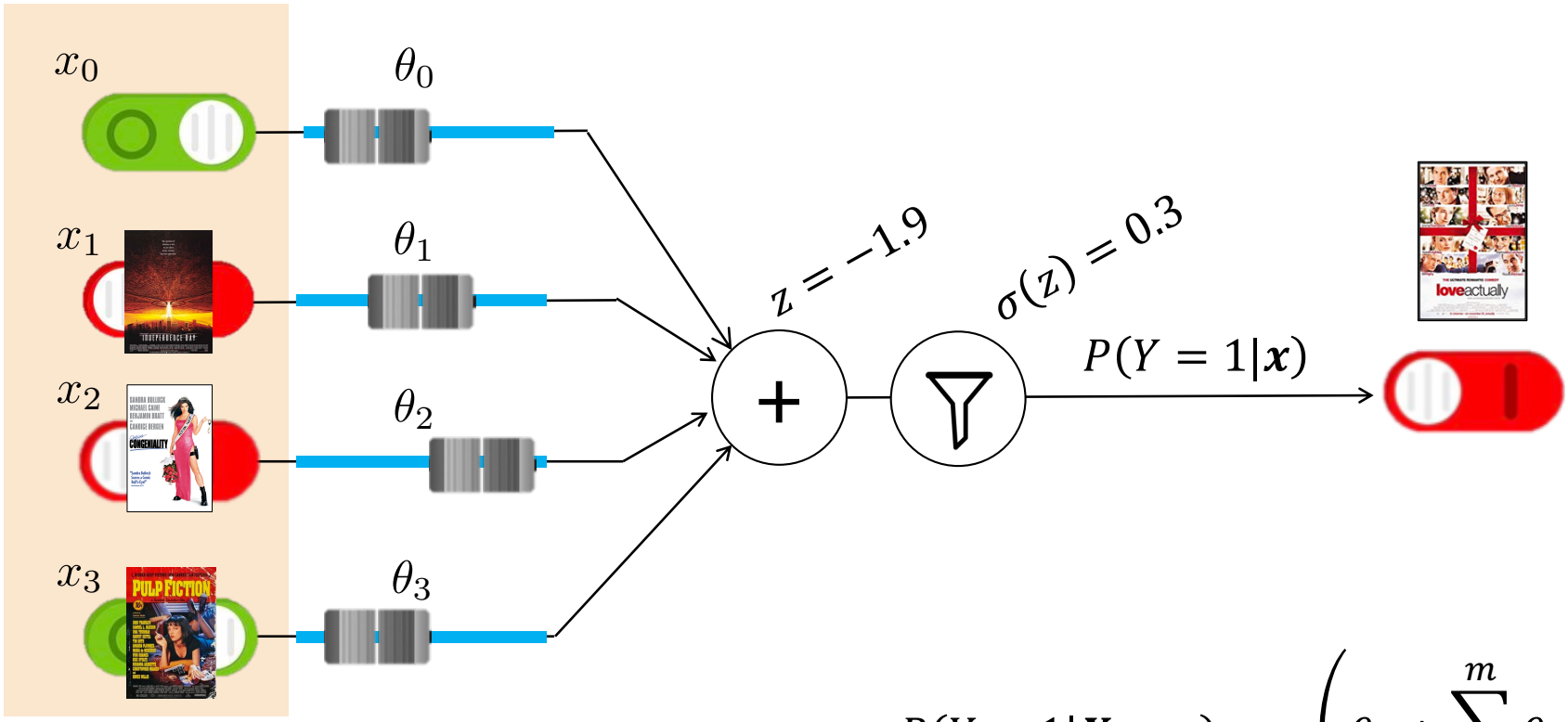
Different predictions for different inputs



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

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

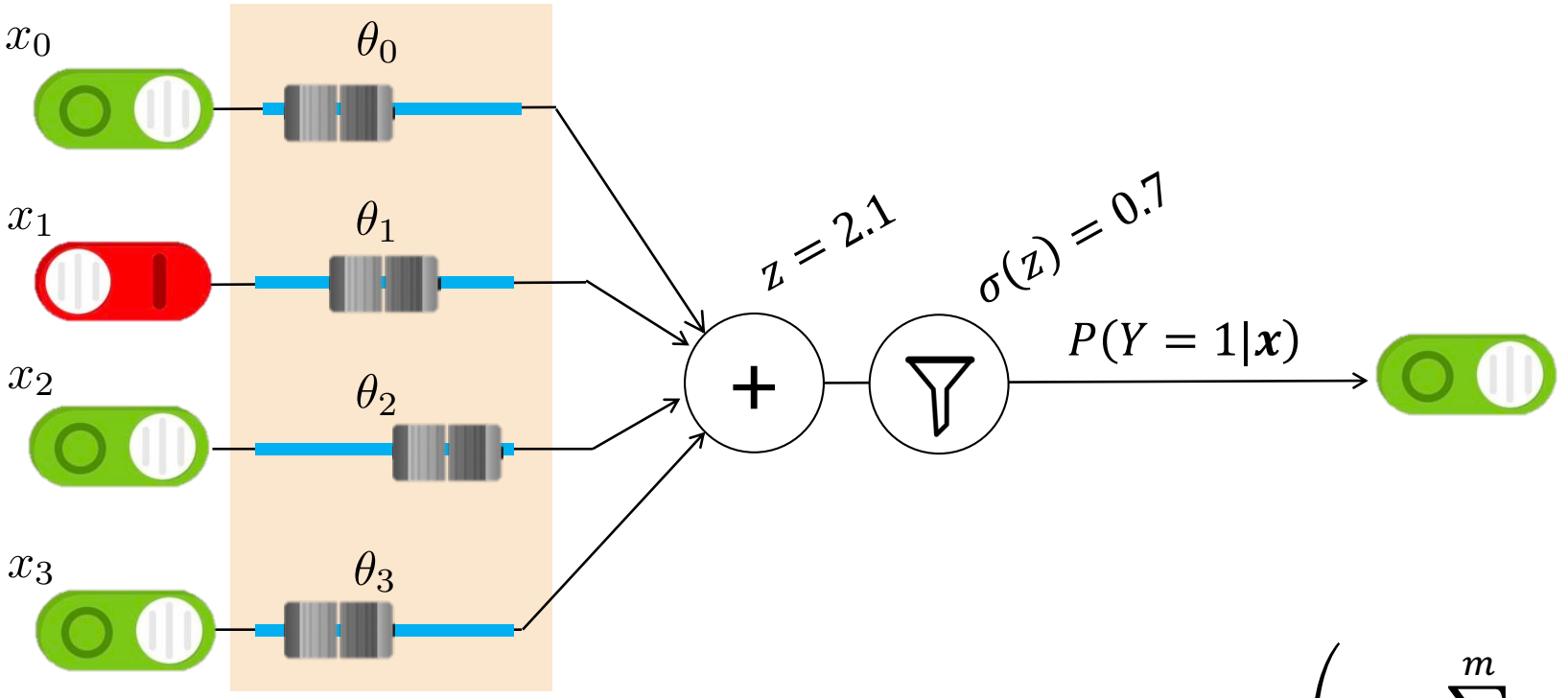
Different predictions for different inputs



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

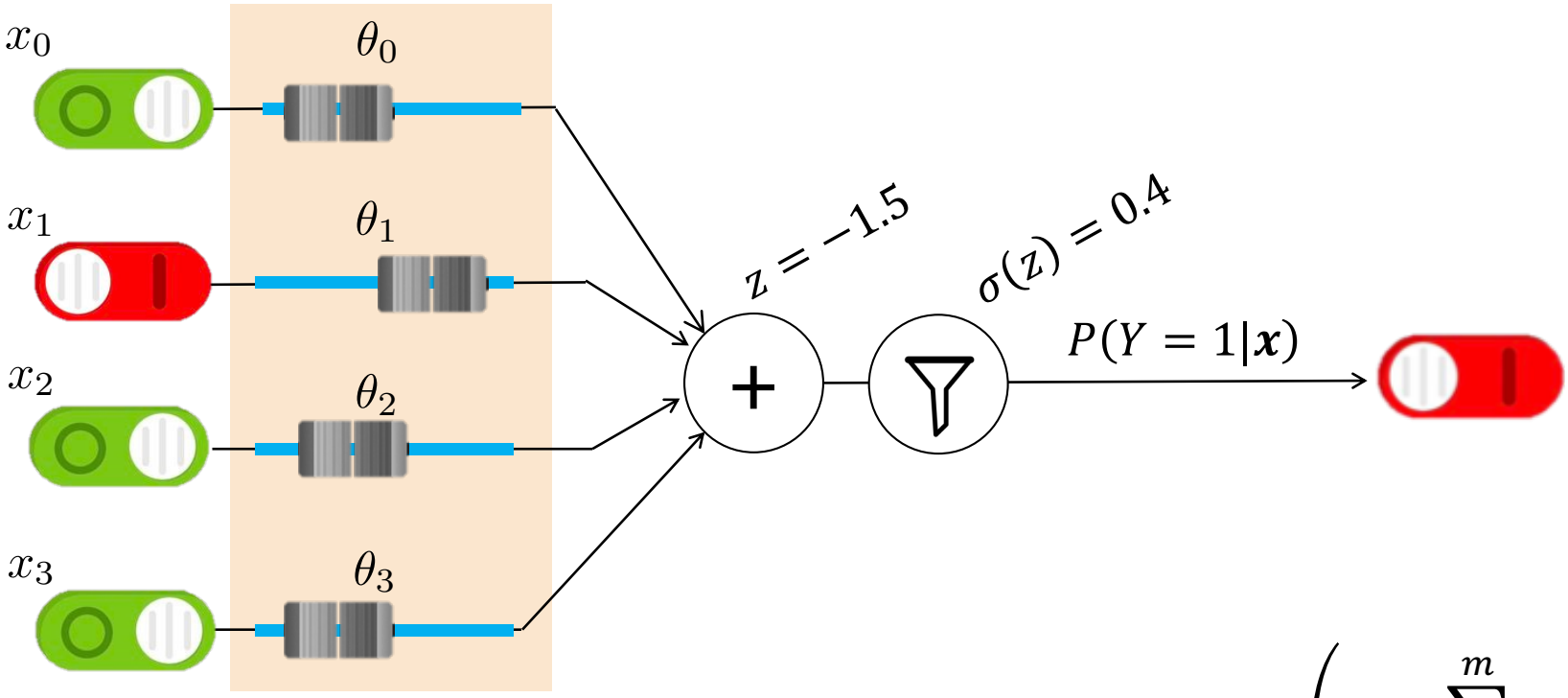
$$P(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = \sigma\left(\theta_0 + \sum_{j=1}^{m} \theta_j x_j\right)$$

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

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

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

Training

Estimate parameters from training data

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

Testing

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

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

Training: The big picture

Logistic regression classifier

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

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

Training

Estimate parameters from training data

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

Choose θ that optimizes some objective:

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

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

2. Gradient wrt θ_j, for $j = 0, 1, \ldots, m$

3. Solve
   - No analytical derivation of θ_{MLE}…
   - …but can still determine θ_{MLE} via gradient ascent!

```
initialize x
repeat many times:
  compute gradient
   x += η * gradient
```

1. Determine objective function

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

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

First: Interpret conditional likelihood with Logistic Regression

Second: Write a differentiable expression for log conditional likelihood

1. Determine objective function (interpret)

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

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

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)

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

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

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?

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

1. Determine objective function (write)

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

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

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 wrt θ_j, for $j = 0, 1, \ldots, m$:

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

🦄 presented here without proof, though it is 🦄 a generalization of the gradient of **a** in the $Y = aX + b + Z$ derivation from last lecture

How do we interpret the gradient contribution of the i^{th} training datapoint?

2. Find gradient with respect to θ

Optimization problem:

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

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

Gradient wrt θ_j, for $j = 0, 1, \ldots, m$:

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

scale by j-th feature

2. Find gradient with respect to θ

Optimization problem:

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

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

Gradient wrt θ_j, for $j = 0, 1, \ldots, m$:

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

1 or 0 $\quad P\left(Y = 1 | \boldsymbol{X} = \boldsymbol{x}^{(i)}\right)$

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 wrt θ_j, for $j = 0, 1, \ldots, m$:

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

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

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

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

2. Gradient wrt 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 until convergence:
    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

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

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 until convergence:
    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

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

compute

outer loop

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat until convergence:
    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

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

compute

outer loop

```
initialize θ_j = 0 for 0 ≤ j ≤ m
repeat until convergence:
    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
```

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 until convergence:
    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
```

- 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 $\theta_j = 0$ for $0 \leq j \leq m$
repeat until convergence:
    gradient[j] = 0 for $0 \leq j \leq m$
    for each training example $(\boldsymbol{x}, y)$:
        for each $0 \leq j \leq m$:
            gradient[j] += $\left[y - \frac{1}{1 + e^{-\theta^T \boldsymbol{x}}} \right] x_j$
    θ_j += η * 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

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 until convergence:
    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
```

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

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

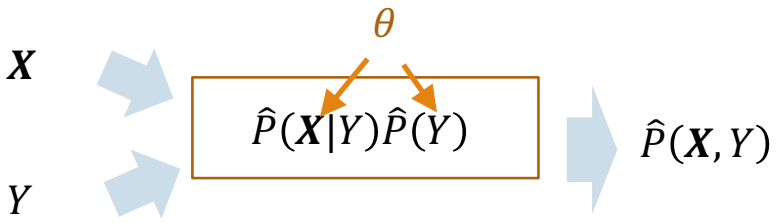
Training: Gradient Ascent

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

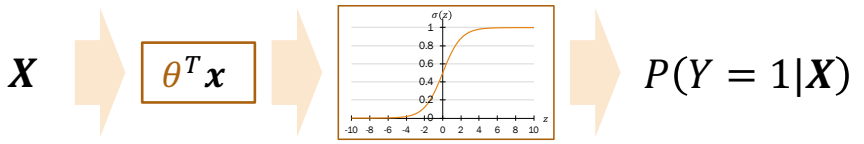
initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat until convergence:
    gradient[j] = 0 for $0 \leq j \leq m$
    for each training example $(\boldsymbol{x}, y)$:
        for each $0 \leq j \leq m$:
            gradient[j] += $\left[y - \frac{1}{1 + e^{-\theta^T \boldsymbol{x}}} \right] x_j$
    θ_j += η * 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 the j^{th} feature of input $\boldsymbol{x} = (x_1, \ldots, x_m)$
- Insert $x_0 = 1$ before training makes sigmoid of matrix compact!

Naïve Bayes vs Logistic Regression



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



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

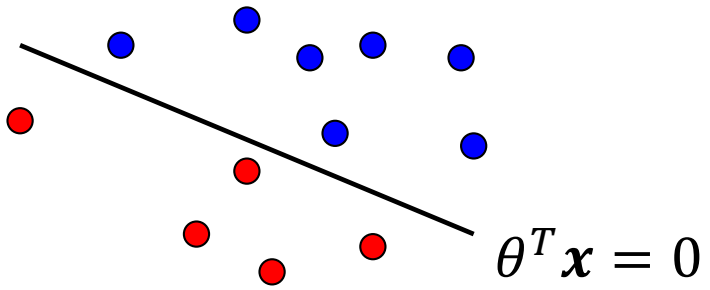
Compare/contrast:

1. What **distributions** are we modeling?
2. After learning our parameters, could we randomly **generate** a new datapoint $(\boldsymbol{x}, y)$?
3. Could we model a **continuous** X_j feature (e.g., $X_j \sim$Normal, or $X_j \sim$Unknown)?
4. Could we model a non-binary **discrete** X_j (e.g., $X_j \in \{1,2,\ldots,6\}$)?

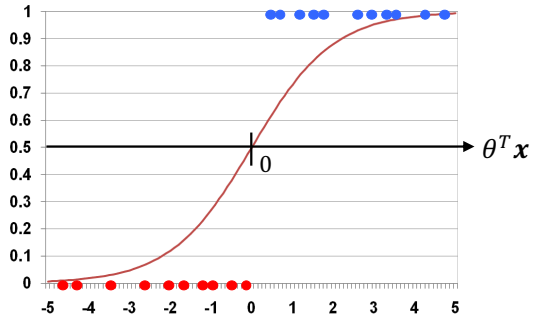
Tradeoffs:	Naïve Bayes	Logistic Regression
1. Modeling goal	$P(\boldsymbol{X}, Y)$	$P(Y \mid \boldsymbol{X})$
2. Generative or discriminative?	**Generative**: could use joint distribution to generate new points (⚠️ but you might not need this extra effort)	**Discriminative**: just tries to discriminate $y = 0$ vs $y = 1$ (❌ cannot generate new points b/c no $P(\boldsymbol{X}, Y)$)
3. Continuous input features	⚠️ Needs parametric form (e.g., Gaussian) or discretized buckets (for multinomial features)	✅ Yes, easily
4. Discrete input features	✅ Yes, multi-value discrete data = multinomial $P(X_i \mid Y)$	⚠️ Multi-valued discrete data hard (e.g., if $X_i \in \{A, B, C\}$, not necessarily good to encode as $\{1, 2, 3\}$

Linearly separable data

Logistic Regression is trying to find the **line** that separates data instances where $y = 1$ from those where $y = 0$:



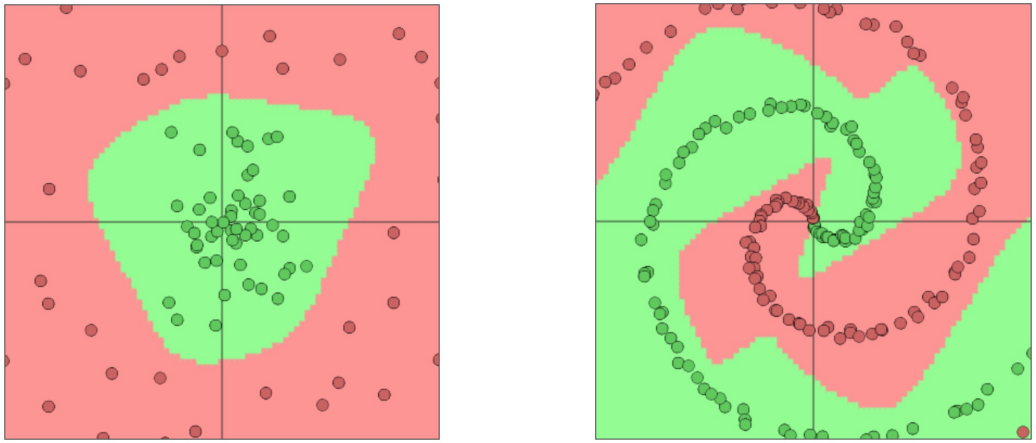
- We call such data (or functions generating that data) **linearly separable**.



- Naïve Bayes is linear too, because there is one parameter for each feature (and no parameters that involve multiple features).

$$\hat{P}(\boldsymbol{X}|Y) = \prod_{j=1}^{m} \hat{P}(X_j|Y)$$

Data is not always linearly separable



- Not possible to draw a line that **successfully** separates all the $y = 1$ points (green) from the $y = 0$ points (red)
- Despite this, Logistic Regression and Naive Bayes still often work well in practice

Extra: Gradient Derivation

Background: Calculus

Calculus refresher #1:
Derivative(sum) =
sum(derivative)

$$\frac{\partial}{\partial x}\sum_{i=1}^{n} f_i(x) = \sum_{i=1}^{n} \frac{\partial f_i(x)}{\partial x}$$

Calculus refresher #2:
Chain rule ⭐⭐⭐

$$\frac{\partial f(x)}{\partial x} = \frac{\partial f(z)}{\partial z}\frac{\partial z}{\partial x}$$

Calculus Chain Rule

$$f(x) = f\big(z(x)\big)$$

aka decomposition
of composed functions

Our goal

Find: $\frac{\partial LL(\theta)}{\partial \theta_j}$ where

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

log conditional likelihood

Two "pre-processing" steps to prepare for chain rule

1. Rewrite $LL(\theta)$ with $\hat{y}$
2. Compute gradient of $\hat{y}$

1. Rewriting $LL(\theta)$ with $\hat{y}$

Find: $\frac{\partial LL(\theta)}{\partial \theta_j}$ where

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

log conditional likelihood

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

Let $\hat{y}^{(i)} = \sigma\left(\theta^T \boldsymbol{x}^{(i)}\right)$

2. Compute gradient of $\hat{y} = \sigma(\theta^T \boldsymbol{x})$

Aside: sigmoid has a beautiful derivative!

Sigmoid function:

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

Derivative:

$$\frac{d}{dz}\sigma(z) = \sigma(z)[1 - \sigma(z)]$$

2. Compute gradient of $\hat{y} = \sigma(\theta^T \boldsymbol{x})$

Sigmoid function:

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

Derivative:

$$\frac{d}{dz}\sigma(z) = \sigma(z)[1 - \sigma(z)]$$

What is $\frac{\partial}{\partial \theta_j}\hat{y} = \frac{\partial}{\partial \theta_j}\sigma(\theta^T \boldsymbol{x})$?

A. $\sigma(x_j)[1 - \sigma(x_j)]x_j$

B. $\sigma(\theta^T \boldsymbol{x})[1 - \sigma(\theta^T \boldsymbol{x})]\boldsymbol{x}$

C. $\sigma(\theta^T \boldsymbol{x})[1 - \sigma(\theta^T \boldsymbol{x})]x_j$

D. $\sigma(\theta^T \boldsymbol{x})x_j[1 - \sigma(\theta^T \boldsymbol{x})x_j]$

E. None/other

2. Compute gradient of $\hat{y} = \sigma(\theta^T \boldsymbol{x})$

Sigmoid function:

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

Derivative:

$$\frac{d}{dz}\sigma(z) = \sigma(z)[1 - \sigma(z)]$$

What is $\frac{\partial}{\partial \theta_j}\sigma(\theta^T \boldsymbol{x})$?

A. $\sigma(x_j)[1 - \sigma(x_j)]x_j$

B. $\sigma(\theta^T \boldsymbol{x})[1 - \sigma(\theta^T \boldsymbol{x})]\boldsymbol{x}$

C. $\sigma(\theta^T \boldsymbol{x})[1 - \sigma(\theta^T \boldsymbol{x})]x_j$

D. $\sigma(\theta^T \boldsymbol{x})x_j[1 - \sigma(\theta^T \boldsymbol{x})x_j]$

E. None/other

Let $z = \theta^T \boldsymbol{x} = \sum_{k=0}^{m} \theta_k x_k$.

$$\frac{\partial}{\partial \theta_j}\sigma(\theta^T \boldsymbol{x}) = \frac{\partial}{\partial z}\sigma(z) \cdot \frac{\partial z}{\partial \theta_j} \quad \text{(Chain Rule)}$$

$$= \sigma(\theta^T \boldsymbol{x})[1 - \sigma(\theta^T \boldsymbol{x})]x_j$$

Compute gradient of log conditional likelihood

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \frac{\partial}{\partial \theta_j} \left[y^{(i)} \log\left(\hat{y}^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - \hat{y}^{(i)}\right) \right] \qquad \text{Let } \hat{y}^{(i)} = \sigma\left(\theta^T \boldsymbol{x}^{(i)}\right)$$

$$= \sum_{i=1}^{n} \frac{\partial}{\partial \hat{y}^{(i)}} \left[y^{(i)} \log\left(\hat{y}^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - \hat{y}^{(i)}\right) \right] \cdot \frac{\partial \hat{y}^{(i)}}{\partial \theta_j} \qquad \text{(Chain Rule)}$$

$$= \sum_{i=1}^{n} \left[y^{(i)} \frac{1}{\hat{y}^{(i)}} - \left(1 - y^{(i)}\right) \frac{1}{1 - \hat{y}^{(i)}} \right] \cdot \hat{y}^{(i)} \left(1 - \hat{y}^{(i)}\right) x_j^{(i)} \qquad \text{(calculus)}$$

$$= \sum_{i=1}^{n} \left[y^{(i)} - \hat{y}^{(i)} \right] x_j^{(i)} \qquad = \sum_{i=1}^{n} \left[y^{(i)} - \sigma\left(\theta^T \boldsymbol{x}^{(i)}\right) \right] x_j^{(i)} \qquad \text{(simplify)}$$

Compute gradient of log conditional likelihood

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \frac{\partial}{\partial \theta_j}\left[y^{(i)} \log\left(\hat{y}^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - \hat{y}^{(i)}\right)\right] \qquad \text{Let } \hat{y}^{(i)} = \sigma\left(\theta^T \boldsymbol{x}^{(i)}\right)$$

$$= \sum_{i=1}^{n} \frac{\partial}{\partial \hat{y}^{(i)}}\left[y^{(i)} \log\left(\hat{y}^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - \hat{y}^{(i)}\right)\right] \cdot \frac{\partial \hat{y}^{(i)}}{\partial \theta_j} \qquad \text{(Chain Rule)}$$

$$= \sum_{i=1}^{n} \left[y^{(i)} \frac{1}{\hat{y}^{(i)}} - \left(1 - y^{(i)}\right) \frac{1}{1 - \hat{y}^{(i)}}\right] \cdot \hat{y}^{(i)}\left(1 - \hat{y}^{(i)}\right) x_j^{(i)} \qquad \text{(calculus)}$$

$$= \sum_{i=1}^{n} \left[y^{(i)} - \hat{y}^{(i)}\right] x_j^{(i)} \qquad = \sum_{i=1}^{n} \left[y^{(i)} - \sigma\left(\theta^T \boldsymbol{x}^{(i)}\right)\right] x_j^{(i)} \qquad \text{(simplify)}$$