

# Upcoming Deadlines

- **Quiz 1 due tomorrow**
- HW 2 due in one week

# Announcements

- **Bug bounty**

- Bonus points for students who find “bugs” in the written/coding HW
- Score depends on severity of bugs

- **Clarification on late penalty**

- Only available for HWs, **not quizzes or project milestones**

- **HW Clarifications on EdStem**

- We will keep a list of clarifications pinned
- Please look through these before asking questions!

# Lecture 6: Logistic Regression

CIS 4190/5190

Spring 2023

# Loss Minimization View of ML

- **Three design decisions**

- **Model family:** What are the candidate models  $f$ ? (E.g., linear functions)
- **Loss function:** How to define “approximating”? (E.g., MSE loss)
- **Optimizer:** How do we optimize the loss? (E.g., gradient descent)

# Maximum Likelihood Estimation

- A **probabilistic** viewpoint on learning (from statistics)
- Given  $x_i$ , **suppose**  $y_i$  is drawn i.i.d. from distribution  $p_{Y|X}(Y = y | x; \beta)$  with parameters  $\beta$  (or density, if  $y_i$  is continuous):

$$y_i \sim p_{Y|X}(\cdot | x_i; \beta)$$

  $Y$  is random variable,  
not vector

- Typically write  $p_\beta(Y = y | x)$  or just  $p_\beta(y | x)$ 
  - Called a **model** (and  $\{p_\beta\}_\beta$  is the **model family**)
  - Will show up convert  $p_\beta$  to  $f_\beta$  later

# Maximum Likelihood Estimation

- **Two design decisions**
  - **Model family:** What are the candidate models  $f$ ? (E.g., linear functions)
  - **Loss function:** How to define “approximating”? (E.g., MSE loss)
  - **Optimizer:** How do we optimize the loss? (E.g., gradient descent)

# Maximum Likelihood Estimation

- **Two design decisions**

- **Likelihood:** What are the candidate likelihoods  $p$ ? (E.g., Gaussian)
- **Optimizer:** How do we optimize the loss? (E.g., gradient descent)

# Maximum Likelihood Estimation

- **Likelihood:** Given model  $p_{\beta}$ , the probability of dataset  $Z$  (replaces loss function in loss minimization view):

$$L(\beta; Z) = p_{\beta}(Y | X) = \prod_{i=1}^n p_{\beta}(y_i | x_i)$$

- **Negative Log-likelihood (NLL):** Computationally better behaved form:

$$\ell(\beta; Z) = -\log L(\beta; Z) = -\sum_{i=1}^n \log p_{\beta}(y_i | x_i)$$



# Example: Linear Regression

- Assume that the conditional density is

$$p_{\beta}(y_i | x_i) = N(y_i; \beta^{\top} x_i, 1) = \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{(\beta^{\top} x_i - y_i)^2}{2}}$$

- $N(y; \mu, \sigma^2)$  is the density of the normal (a.k.a. Gaussian) distribution with mean  $\mu$  and variance  $\sigma^2$

# Example: Linear Regression

- Then, the likelihood is

$$L(\beta; Z) = \prod_{i=1}^n p_{\beta}(y_i | x_i) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{(\beta^T x_i - y_i)^2}{2}}$$

- The NLL is

$$\ell(\beta; Z) = -\sum_{i=1}^n \log p_{\beta}(y_i | x_i) = \underbrace{\frac{n \log(2\pi)}{2}}_{\text{constant}} + \underbrace{\sum_{i=1}^n (\beta^T x_i - y_i)^2}_{\text{MSE!}}$$

# Example: Linear Regression

- Loss minimization for maximum likelihood estimation:

$$\hat{\beta}(Z) = \arg \min_{\beta} \ell(\beta; Z)$$

- **Note:** Called maximum likelihood estimation since maximizing the likelihood equivalent to minimizing the NLL

# Example: Linear Regression

- What about the model family?

$$\begin{aligned} f_{\beta}(x) &= \arg \max_y p_{\beta}(y | x) \\ &= \arg \max_y \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{\|\beta^{\top}x - y\|_2^2}{2}} \\ &= \beta^{\top}x \end{aligned}$$

- **Recovers linear functions!**

# Maximum Likelihood View of ML

- **Two design decisions**
  - **Likelihood:** Probability  $p_{\beta}(y | x)$  of data  $(x, y)$  given parameters  $\beta$
  - **Optimizer:** How do we optimize the NLL? (E.g., gradient descent)
- **Corresponding Loss Minimization View:**
  - **Model family:** Most likely label  $f_{\beta}(x) = \arg \max_y p_{\beta}(y | x)$
  - **Loss function:** Negative log likelihood (NLL)  $\ell(\beta; Z) = -\sum_{i=1}^n \log p_{\beta}(y_i | x_i)$
- Very powerful framework for designing cutting edge ML algorithms
  - Write down the “right” likelihood, form tractable approximation if needed
  - Especially useful for thinking about non-i.i.d. data

# What about classification?

Compare to linear regression:

$$p_{\beta}(y | x_i) \propto e^{-\frac{(\beta^T x_i - y)^2}{2}}$$

- Consider the following choice:

$$p_{\beta}(Y = 0 | x_i) \propto e^{-\frac{\beta^T x_i}{2}} \quad \text{and} \quad p_{\beta}(Y = 1 | x_i) \propto e^{\frac{\beta^T x_i}{2}}$$

- Then, we have

$$p_{\beta}(Y = 1 | x_i) = \frac{e^{\frac{\beta^T x_i}{2}}}{e^{\frac{\beta^T x_i}{2}} + e^{-\frac{\beta^T x_i}{2}}} = \frac{1}{1 + e^{-\beta^T x_i}}$$

**Sigmoid function**

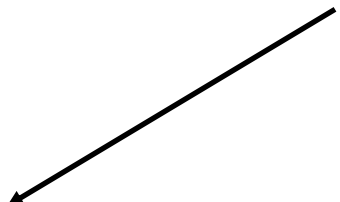
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

# What about classification?

Compare to linear regression:

- Consider the following choice:


$$p_{\beta}(Y = 0 | x_i) \propto e^{-\frac{\beta^T x_i}{2}} \quad \text{and} \quad p_{\beta}(Y = 1 | x_i) \propto e^{\frac{\beta^T x_i}{2}}$$

$$p_{\beta}(y | x_i) \propto e^{-\frac{(\beta^T x_i - y)^2}{2}}$$


- Then, we have

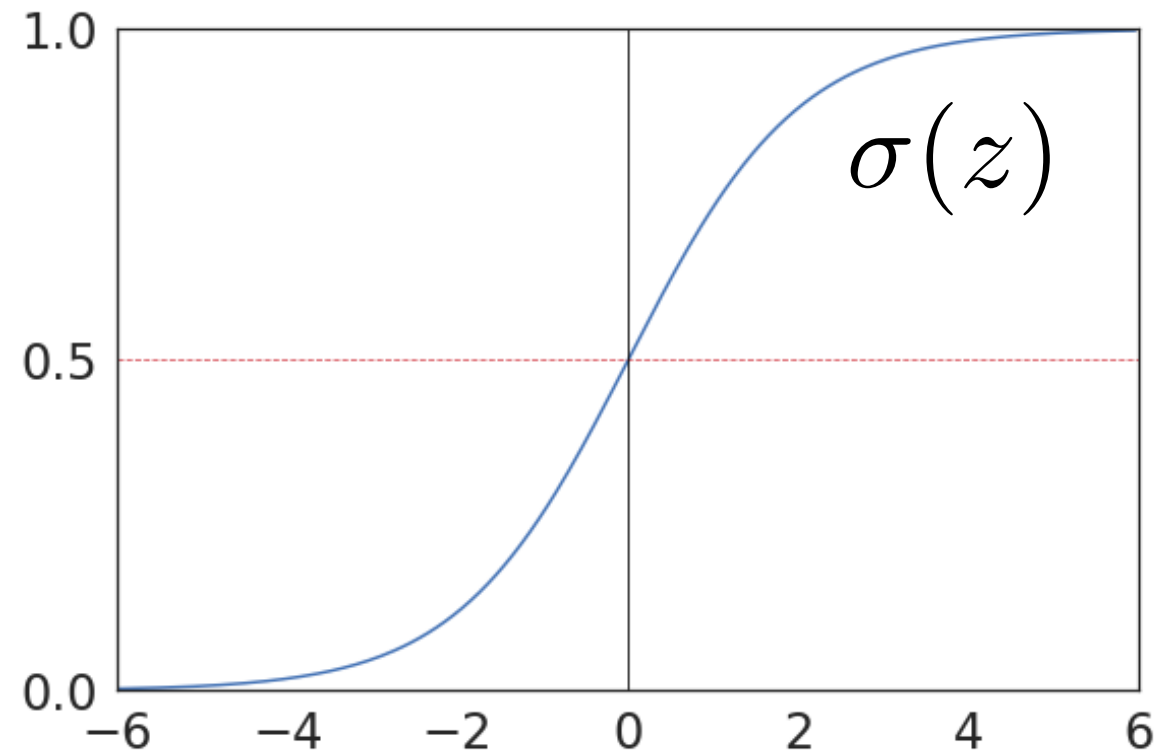
$$p_{\beta}(Y = 1 | x_i) = \frac{e^{\frac{\beta^T x_i}{2}}}{e^{\frac{\beta^T x_i}{2}} + e^{-\frac{\beta^T x_i}{2}}} = \sigma(\beta^T x_i)$$

**Sigmoid function**

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


- Furthermore,  $p_{\beta}(Y = 0 | x_i) = 1 - \sigma(\beta^T x_i)$

# Logistic/Sigmoid Function



$$p_{\beta}(Y = 1 \mid x_i) = \sigma(\beta^{\top} x_i)$$



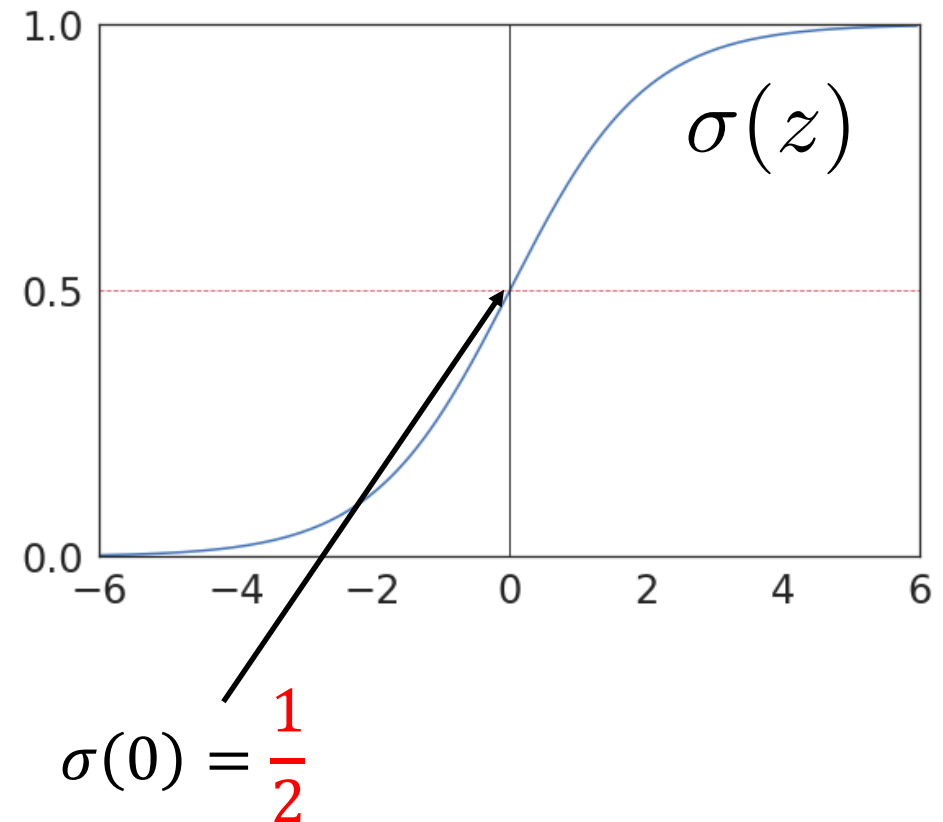
# Logistic Regression Model Family

$$\begin{aligned} f_{\beta}(x) &= \arg \max_y p_{\beta}(y | x) \\ &= \arg \max_y \begin{cases} \sigma(\beta^{\top} x) & \text{if } y = 1 \\ 1 - \sigma(\beta^{\top} x) & \text{if } y = 0 \end{cases} \\ &= \begin{cases} 1 & \text{if } \sigma(\beta^{\top} x) \geq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

# Logistic Regression Model Family

$$\begin{aligned} f_{\beta}(x) &= \arg \max_y p_{\beta}(y | x) \\ &= \arg \max_y \begin{cases} \sigma(\beta^{\top} x) & \text{if } y = 1 \\ 1 - \sigma(\beta^{\top} x) & \text{if } y = 0 \end{cases} \\ &= \begin{cases} 1 & \text{if } \sigma(\beta^{\top} x) \geq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \beta^{\top} x \geq 0 \\ 0 & \text{otherwise} \end{cases} \\ &= 1(\beta^{\top} x \geq 0) \end{aligned}$$

- Recovers linear classifiers!



# Logistic Regression Algorithm

- Then, we have the following NLL loss:

$$\begin{aligned}\ell(\beta; \mathbf{Z}) &= -\sum_{i=1}^n \log p_{\beta}(y_i | x_i) \\ &= -\sum_{i=1}^n 1(y_i = 1) \cdot \log(\sigma(\beta^{\top} x_i)) + 1(y_i = 0) \cdot \log(1 - \sigma(\beta^{\top} x_i)) \\ &= -\sum_{i=1}^n y_i \cdot \log(\sigma(\beta^{\top} x_i)) + (1 - y_i) \cdot \log(1 - \sigma(\beta^{\top} x_i))\end{aligned}$$

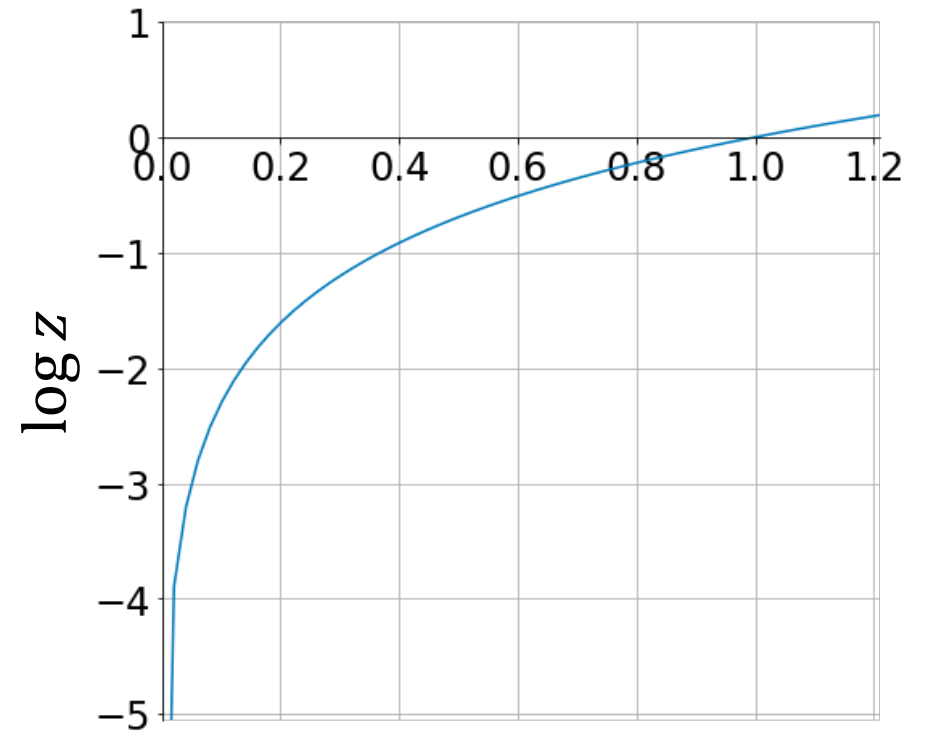
- Logistic regression minimizes this loss:

$$\hat{\beta}(\mathbf{Z}) = \arg \min_{\beta} \ell(\beta; \mathbf{Z})$$

# Intuition on the Objective

- Loss for example  $i$  is

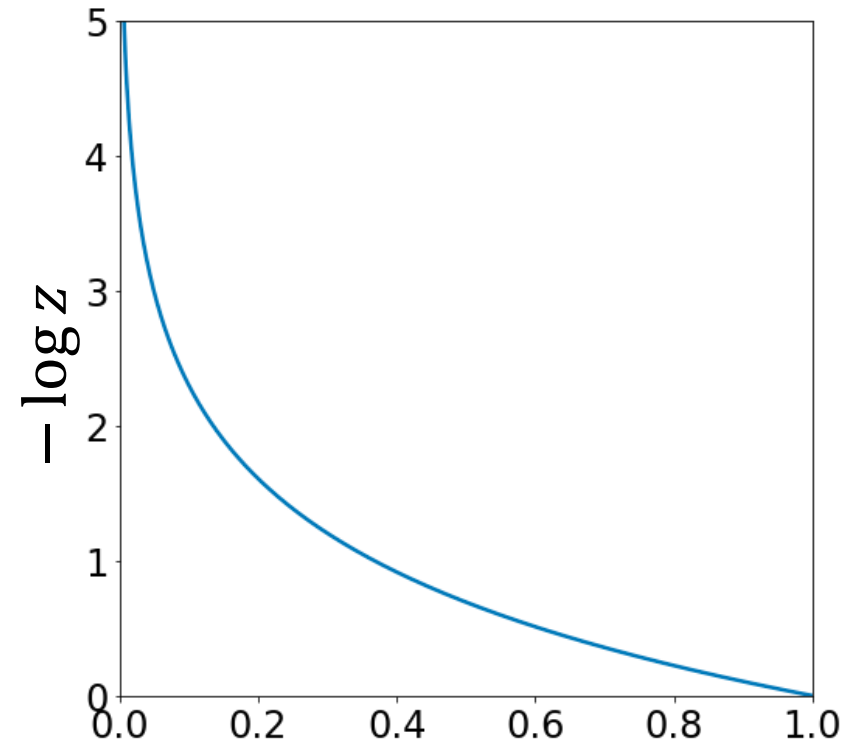
$$\begin{cases} -\log(\sigma(\beta^\top x_i)) & \text{if } y_i = 1 \\ -\log(1 - \sigma(\beta^\top x_i)) & \text{if } y_i = 0 \end{cases}$$



# Intuition on the Objective

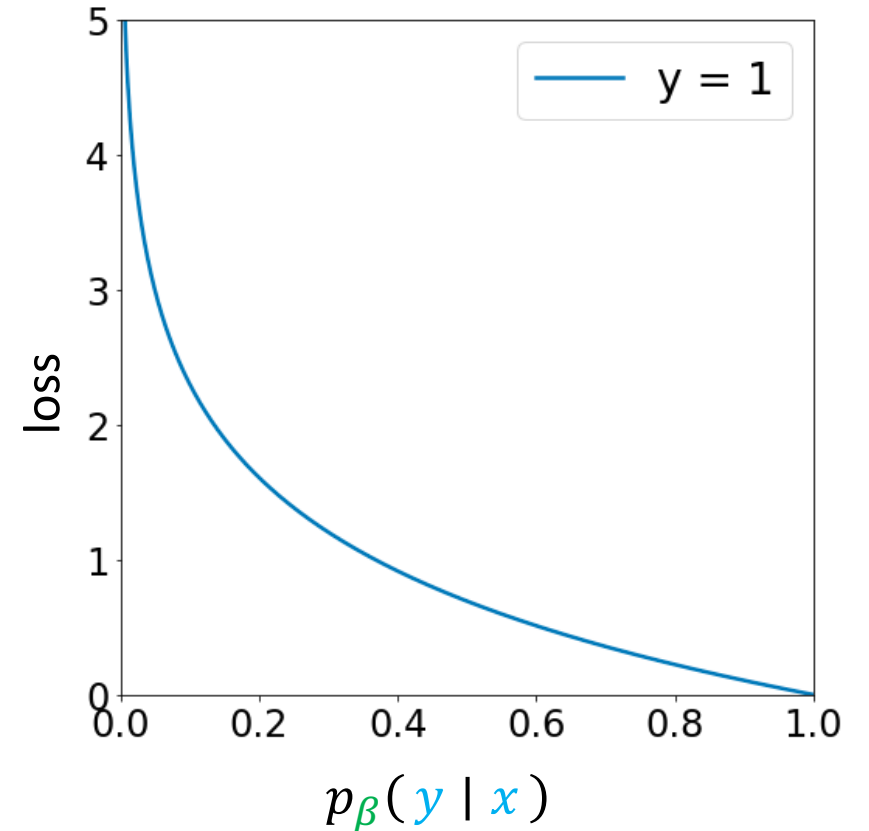
- Loss for example  $i$  is

$$\begin{cases} -\log(\sigma(\beta^\top x_i)) & \text{if } y_i = 1 \\ -\log(1 - \sigma(\beta^\top x_i)) & \text{if } y_i = 0 \end{cases}$$



# Intuition on the Objective

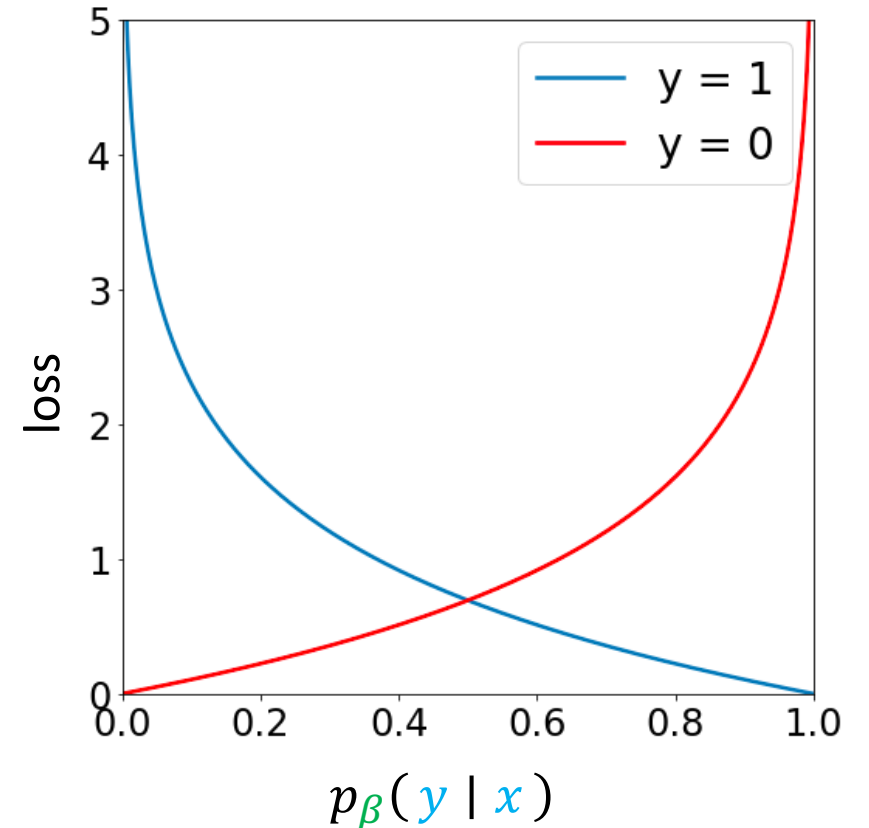
- If  $y_i = 1$ :
  - If  $p_\beta(Y = 1 | x_i) = 1$ , then loss = 0
  - As  $p_\beta(Y = 1 | x_i) \rightarrow 0$ , loss  $\rightarrow \infty$



$$-y_i \cdot \boxed{\log(\sigma(\beta^\top x_i))} - (1 - y_i) \cdot \log(1 - \sigma(\beta^\top x_i))$$

# Intuition on the Objective

- If  $y_i = 1$ :
  - If  $p_\beta(Y = 1 | x_i) = 1$ , then loss = 0
  - As  $p_\beta(Y = 1 | x_i) \rightarrow 0$ , loss  $\rightarrow \infty$
- If  $y_i = 0$ 
  - If  $p_\beta(Y = 0 | x_i) = 1$ , then loss = 0
  - As  $p_\beta(Y = 0 | x_i) \rightarrow 0$ , loss  $\rightarrow \infty$



$$-y_i \cdot \boxed{\log(\sigma(\beta^\top x_i))} - (1 - y_i) \cdot \boxed{\log(1 - \sigma(\beta^\top x_i))}$$

# Optimization for Logistic Regression

- To optimize the NLL loss, we need its gradient:

$$\nabla_{\beta} \ell(\beta; \mathbf{Z}) = -\sum_{i=1}^n y_i \cdot \nabla_{\beta} \log(\sigma(\beta^{\top} x_i)) + (1 - y_i) \cdot \nabla_{\beta} \log(1 - \sigma(\beta^{\top} x_i))$$

$$= -\sum_{i=1}^n y_i \cdot \frac{\nabla_{\beta} \sigma(\beta^{\top} x_i)}{\sigma(\beta^{\top} x_i)} - (1 - y_i) \cdot \frac{\nabla_{\beta} \sigma(\beta^{\top} x_i)}{1 - \sigma(\beta^{\top} x_i)}$$

$$\begin{array}{l} \sigma'(z) \\ = \sigma(z)(1 - \sigma(z)) \end{array} \xrightarrow{\hspace{10em}} = -\sum_{i=1}^n y_i \cdot \frac{\sigma(\beta^{\top} x_i)(1 - \sigma(\beta^{\top} x_i)) \cdot x_i}{\sigma(\beta^{\top} x_i)} - (1 - y_i) \cdot \frac{\sigma(\beta^{\top} x_i)(1 - \sigma(\beta^{\top} x_i)) \cdot x_i}{1 - \sigma(\beta^{\top} x_i)}$$

$$= -\sum_{i=1}^n y_i \cdot (1 - \sigma(\beta^{\top} x_i)) \cdot x_i - (1 - y_i) \cdot \sigma(\beta^{\top} x_i) \cdot x_i$$

$$= -\sum_{i=1}^n (y_i - \sigma(\beta^{\top} x_i)) \cdot x_i$$



# Optimization for Logistic Regression

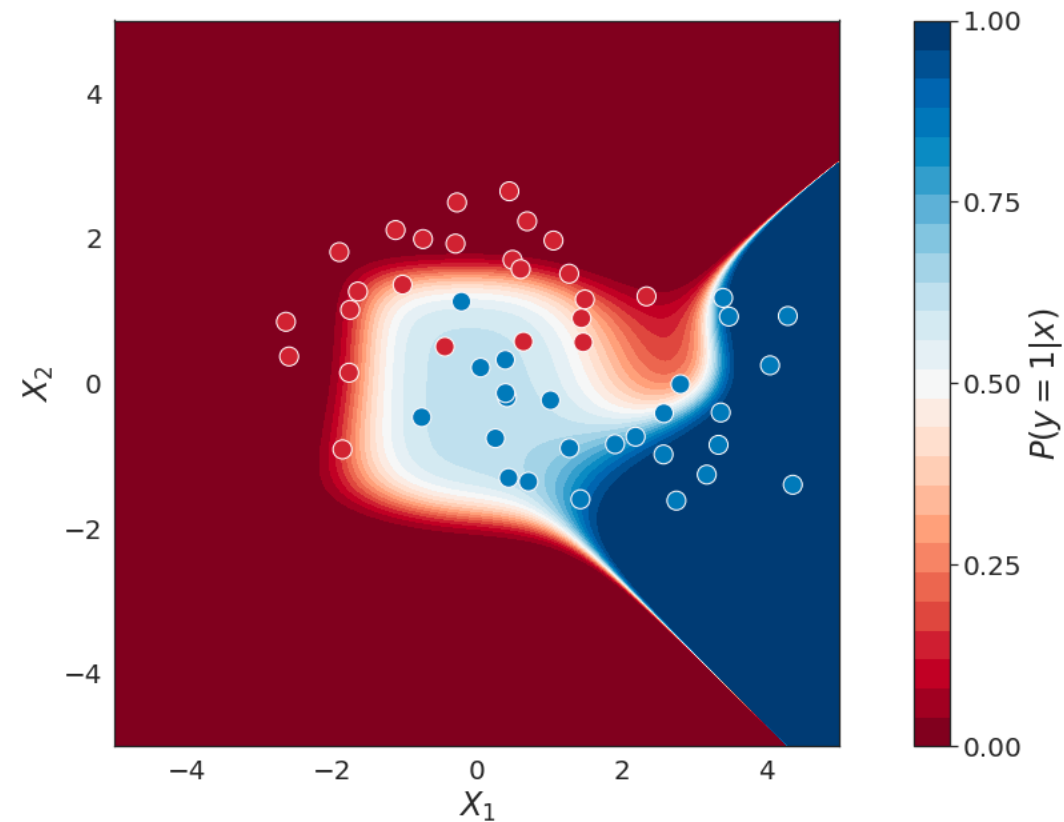
- Gradient of NLL:

$$\nabla_{\beta} \ell(\beta; \mathbf{Z}) = \sum_{i=1}^n (\sigma(\beta^{\top} x_i) - y_i) \cdot x_i$$

- Surprisingly similar to the gradient for linear regression!
  - Only difference is the  $\sigma$
- Gradient descent works as before
  - No closed-form solution for  $\hat{\beta}(\mathbf{Z})$

# Feature Maps

- Can use feature maps, just like linear regression



# Regularized Logistic Regression

- We can add  $L_1$  or  $L_2$  regularization to the NLL loss, e.g.:

$$\ell(\beta; \mathbf{Z}) = - \sum_{i=1}^n y_i \cdot \log(\sigma(\beta^\top x_i)) + (1 - y_i) \cdot \log(1 - \sigma(\beta^\top x_i)) + \lambda \cdot \|\beta\|_2^2$$

- Is there a more “natural” way to derive the regularized loss?

# Regularization as a Prior

- So far, we have not assumed any distribution over the parameters  $\beta$ 
  - What if we assume  $\beta \sim N(0, \sigma^2 I)$  (the  $d$  dimensional normal distribution)?
- Consider the modified likelihood

$$\begin{aligned} L(\beta; Z) &= p_{Y, \beta | X}(Y, \beta | X) \\ &= p_{Y | X, \beta}(Y | X, \beta) \cdot N(\beta; 0, \sigma^2 I) \\ &= \left( \prod_{i=1}^n p_{\beta}(y_i | x_i) \right) \cdot \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{\|\beta\|_2^2}{2\sigma^2}} \end{aligned}$$

# Regularization as a Prior

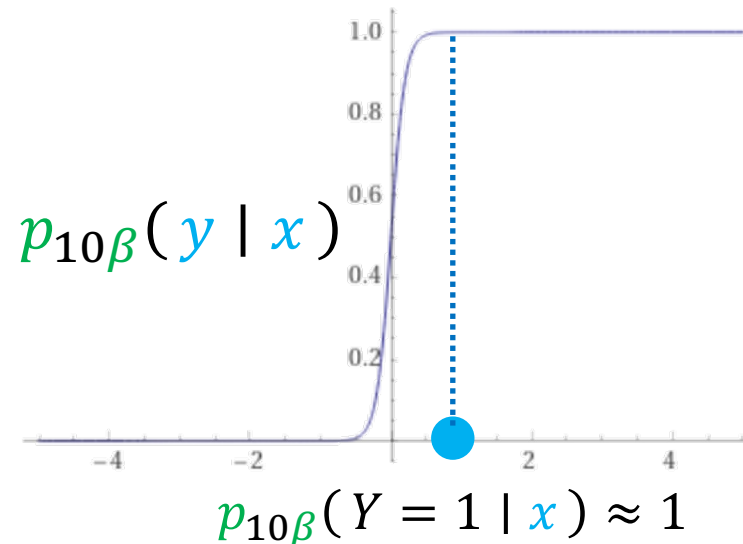
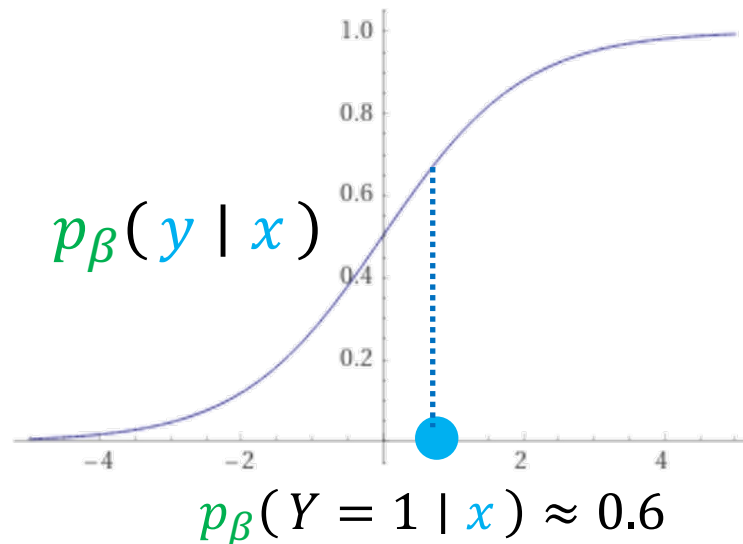
- So far, we have not assumed any distribution over the parameters  $\beta$ 
  - What if we assume  $\beta \sim N(0, \sigma^2 I)$  (the  $d$  dimensional normal distribution)?
- Consider the modified NLL

$$\ell(\beta; \mathbf{Z}) = -\sum_{i=1}^n \log p_{\beta}(y_i | x_i) + \underbrace{\log \sigma \sqrt{2\pi}}_{\text{constant}} + \underbrace{\frac{\|\beta\|_2^2}{2\sigma^2}}_{\text{regularization!}}$$

- Obtain  $L_2$  regularization on  $\beta$ !
  - With  $\lambda = \frac{1}{2\sigma^2}$
  - If  $\beta_i \sim \text{Laplace}(0, \sigma^2)$  for each  $i$ , obtain  $L_1$  regularization

# Additional Role of Regularization

- In  $p_\beta$ , if we replace  $\beta$  with  $c \cdot \beta$ , where  $c \gg 1$  (and  $c \in \mathbb{R}$ ), then:
  - The decision boundary does not change
  - The probabilities  $p_\beta(y | x)$  become more confident

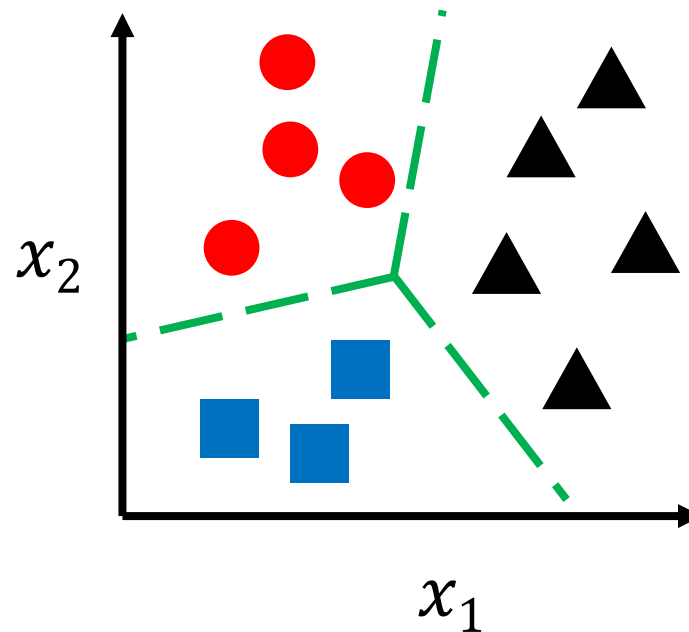


# Additional Role of Regularization

- Regularization ensures that  $\beta$  does not become too large
  - Prevents overconfidence
- Regularization can also be **necessary**
  - Without regularization (i.e.,  $\lambda = 0$ ) and data is linearly separable, then gradient descent diverges (i.e.,  $\beta \rightarrow \pm\infty$ )

# Multi-Class Classification

- What about more than two classes?
  - **Disease diagnosis:** healthy, cold, flu, pneumonia
  - **Object classification:** desk, chair, monitor, bookcase
  - In general, consider a finite space of labels  $\mathcal{Y}$

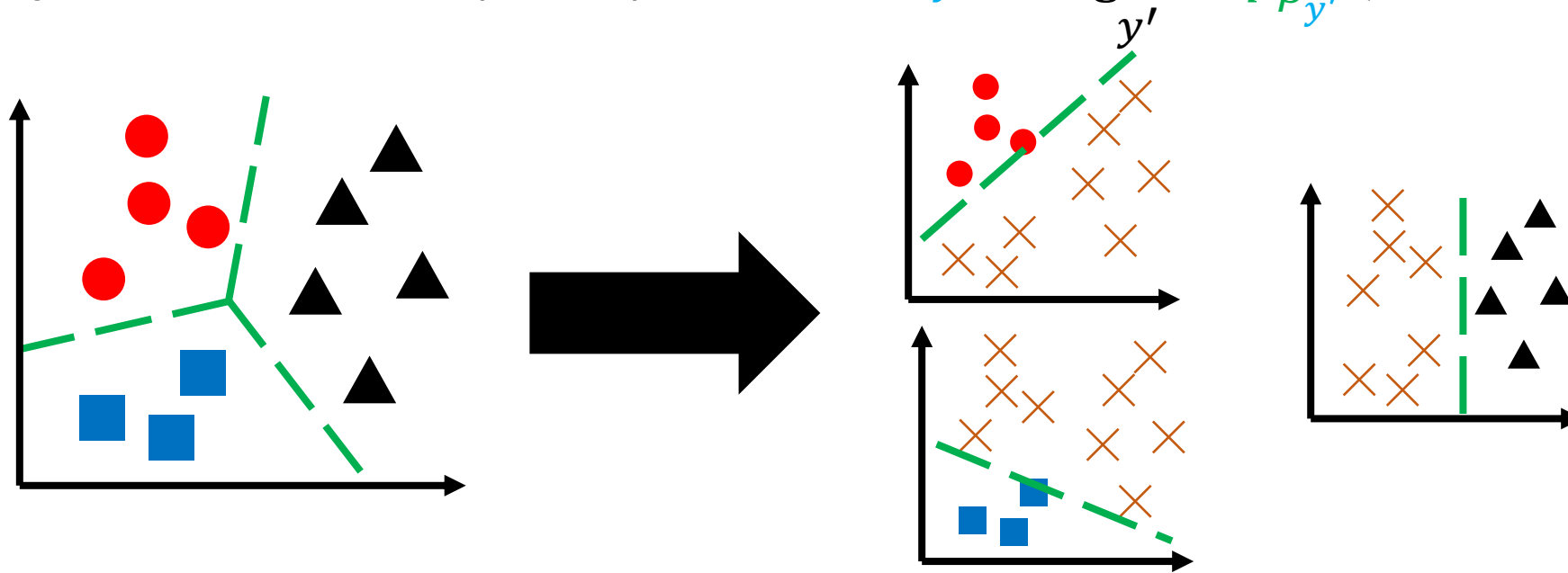




# Multi-Class Classification

- **Naïve Strategy: One-vs-rest classification**

- **Step 1:** Train  $|\mathcal{Y}|$  logistic regression models, where model  $p_{\beta_y}(Y = 1 | x)$  is interpreted as the probability that the label for  $x$  is  $y$
- **Step 2:** Given a new input  $x$ , predict label  $y = \arg \max p_{\beta_{y'}}(Y = 1 | x)$



# Multi-Class Logistic Regression

- **Strategy:** Include separate  $\beta_y$  for each label  $y \in \mathcal{Y} = \{1, \dots, k\}$
- Let  $p_\beta(y | x) \propto e^{\beta_y^\top x}$ , i.e.

$$p_\beta(y | x) = \frac{e^{\beta_y^\top x}}{\sum_{y' \in \mathcal{Y}} e^{\beta_{y'}^\top x}}$$

- We define  $\text{softmax}(z_1, \dots, z_k) = \left[ \frac{e^{z_1}}{\sum_{i=1}^k e^{z_i}} \quad \dots \quad \frac{e^{z_k}}{\sum_{i=1}^k e^{z_i}} \right]$
- Then,  $p_\beta(y | x) = \text{softmax}(\beta_1^\top x, \dots, \beta_k^\top x)_y$ 
  - Thus, sometimes called **softmax regression**

# Multi-Class Logistic Regression

- **Model family**

- $f_{\beta}(x) = \arg \max_y p_{\beta}(y | x) = \arg \max_y \frac{e^{\beta_y^T x}}{\sum_{y' \in \mathcal{Y}} e^{\beta_{y'}^T x}} = \arg \max_y \beta_y^T x$

- **Optimization**

- Gradient descent on NLL
  - Simultaneously update all parameters  $\{\beta_y\}_{y \in \mathcal{Y}}$