Logistic Regression

CS 480
Intro to AI

Directly modeling class probability

Let's look at modeling the **probability** that a given **x** has class **y**. For now, restrict ourselves to binary classification, and the Bernoulli distribution.

$$\mathcal{Y} = \{0, 1\}, \quad \mathcal{X} = \mathbb{R}^D$$

$$p(y \mid \mathbf{x}; \theta) = h_\theta(\mathbf{x})^y (1 - h_\theta(\mathbf{x}))^{1-y}$$

$$h_\theta(\mathbf{x}) = p(y = 1 \mid \mathbf{x}; \theta)$$

So we need a hypothesis class that maps from R^D to [0,1]

$$h_\theta : \mathcal{X} \rightarrow [0, 1]$$

The logistic function (aka sigmoid)

Candidate function

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

Properties

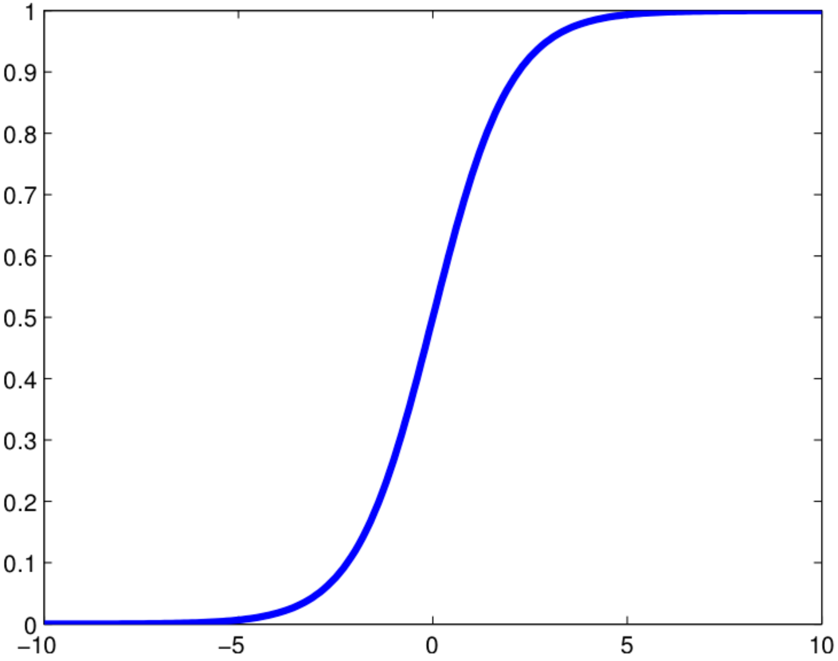
As $z \to -\infty$, $\sigma(z) \to 0$

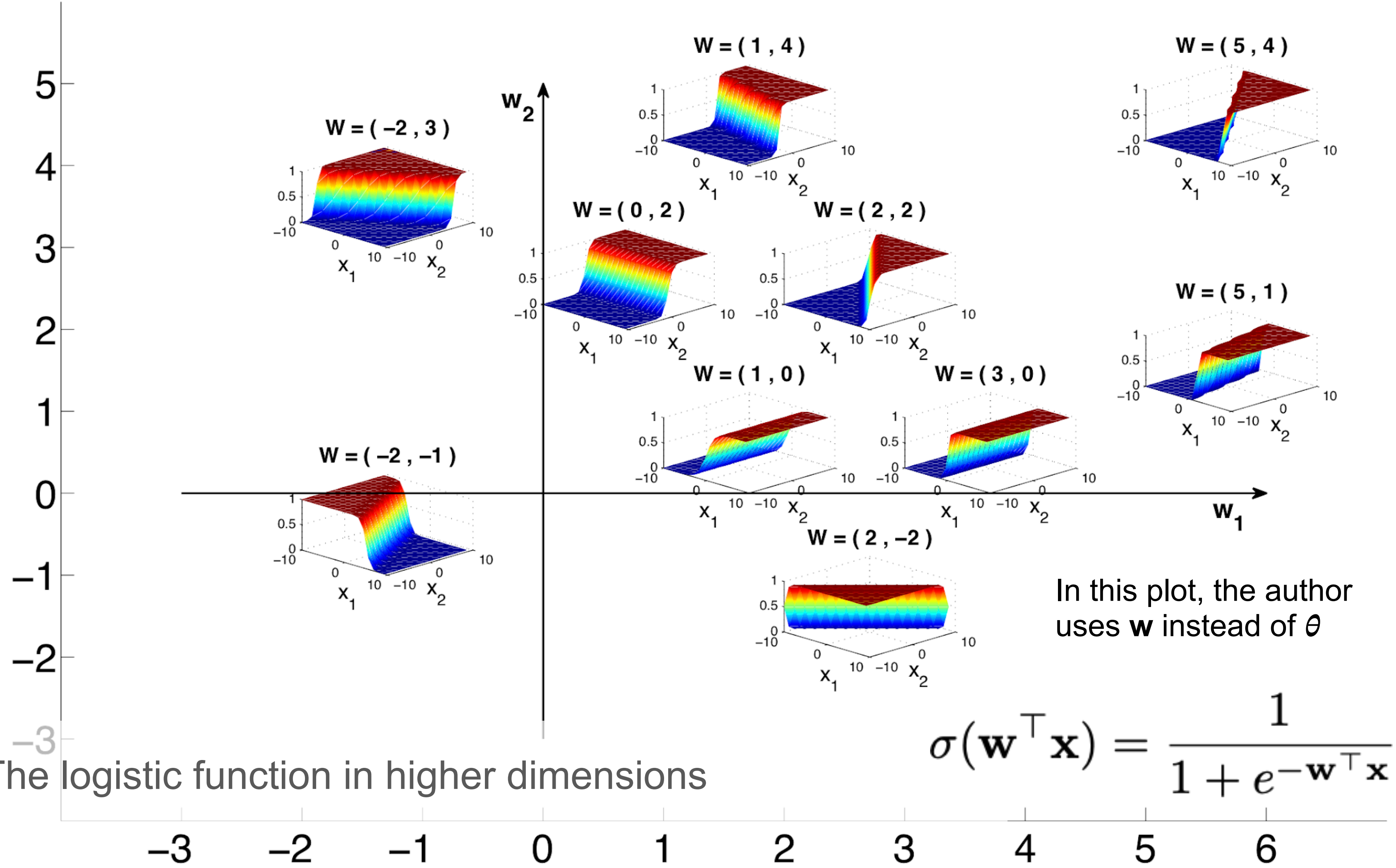
As $z \to \infty$, $\sigma(z) \to 1$

σ “squashes” it’s input to the range (0,1)

Derivative

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



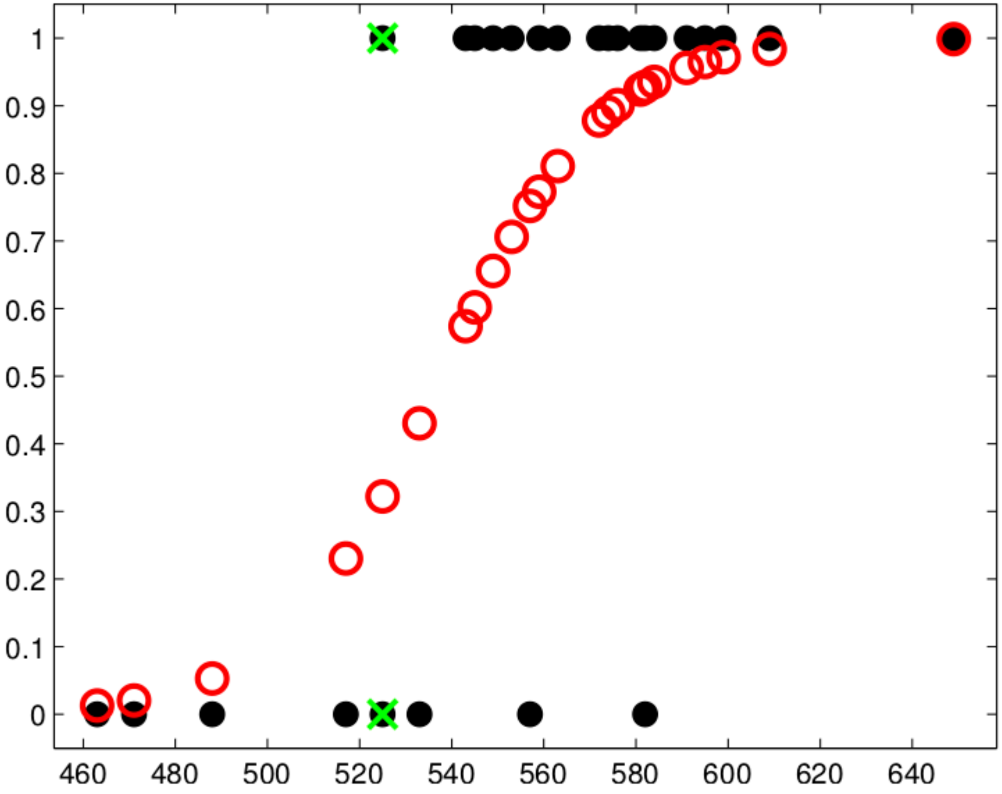


In this plot, the author uses **w** instead of θ

$$\sigma(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}}}$$

The logistic function in higher dimensions

The logistic function (1D example)



Source: Murphy, pg 21

Logistic Regression - MLE (1)

Plug in our definitions to get an objective to minimize

$$\mathcal{Y} = \{0, 1\}, \quad \mathcal{X} = \mathbb{R}^D$$
$$p(y \mid \mathbf{x}; \theta) = h_\theta(\mathbf{x})^y (1 - h_\theta(\mathbf{x}))^{1-y}$$
$$h_\theta(\mathbf{x}) = p(y = 1 \mid \mathbf{x}; \theta)$$

$$\log(a \cdot b) = \log a + \log b$$
$$\log(a^b) = b \log a$$

$$\mathcal{L}(h_\theta; S) = \prod_{i=1}^{N} p\left((\mathbf{x}^{(i)}, y^{(i)}) \mid h_\theta\right)$$

$$NLL(h_\theta; S) = -\log \mathcal{L}(h_\theta; S) = -\sum_{i=1}^{N} \log p\left((\mathbf{x}^{(i)}, y^{(i)}) \mid h_\theta\right)$$
$$= -\sum_{i=1}^{N} \log\left[h_\theta(\mathbf{x}^{(i)})^{y^{(i)}} (1 - h_\theta(\mathbf{x}^{(i)}))^{(1-y^{(i)})}\right]$$
$$= -\sum_{i=1}^{N} \left[y^{(i)} \log h_\theta(\mathbf{x}^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(\mathbf{x}^{(i)}))\right]$$

Logistic Regression - MLE (2)

Find the derivative so we can use gradient descent

$$\begin{aligned}
\nabla_\theta NLL(h_\theta; S) &= -\nabla_\theta \sum_{i=1}^{N} \left[y^{(i)} \log h_\theta(\mathbf{x}^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(\mathbf{x}^{(i)})) \right] \\
&= -\nabla_\theta \sum_{i=1}^{N} \left[y^{(i)} \log \sigma(\theta^\top \mathbf{x}^{(i)}) + (1 - y^{(i)}) \log(1 - \sigma(\theta^\top \mathbf{x}^{(i)}) \right] \\
&= -\sum_{i=1}^{N} \left[y^{(i)} \frac{1}{\sigma(\theta^\top \mathbf{x}^{(i)})} \nabla_\theta \sigma(\theta^\top \mathbf{x}^{(i)}) + (1 - y^{(i)}) \frac{1}{1 - \sigma(\theta^\top \mathbf{x}^{(i)})} \nabla_\theta (1 - \sigma(\theta^\top \mathbf{x}^{(i)})) \right]
\end{aligned}$$

Using: **chain rule**, derivative of **log**, and definition of **h**

Logistic Regression - MLE (3)

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

Use the derivative of the logistic function to get some terms to cancel

$$\begin{aligned}
\nabla_\theta NLL(h_\theta; S) &= -\sum_{i=1}^{N}\left[y^{(i)}\frac{1}{\sigma(\theta^\top \mathbf{x}^{(i)})}\nabla_\theta \sigma(\theta^\top \mathbf{x}^{(i)}) + (1-y^{(i)})\frac{1}{1-\sigma(\theta^\top \mathbf{x})}\nabla_\theta(1-\sigma(\theta^\top \mathbf{x}^{(i)}))\right] \\
&= -\sum_{i=1}^{N}\left[y^{(i)}\frac{\sigma(\theta^\top \mathbf{x}^{(i)})(1-\sigma(\theta^\top \mathbf{x}^{(i)}))\mathbf{x}^{(i)}}{\sigma(\theta^\top \mathbf{x}^{(i)})} + (1-y^{(i)})\frac{-\sigma(\theta^\top \mathbf{x}^{(i)})(1-\sigma(\theta^\top \mathbf{x}^{(i)}))\mathbf{x}^{(i)}}{1-\sigma(\theta^\top \mathbf{x}^{(i)})}\right] \\
&= -\sum_{i=1}^{N}\left[y^{(i)}(1-\sigma(\theta^\top \mathbf{x}^{(i)}))\mathbf{x}^{(i)} + (1-y^{(i)})(-\sigma(\theta^\top \mathbf{x}^{(i)}))\mathbf{x}^{(i)}\right] \\
&= -\sum_{i=1}^{N}\left[y^{(i)}(1-\sigma(\theta^\top \mathbf{x}^{(i)})) + (1-y^{(i)})(-\sigma(\theta^\top \mathbf{x}^{(i)}))\right]\mathbf{x}^{(i)}
\end{aligned}$$

Using: **chain rule**, derivative of **sigma**

Logistic Regression - MLE (4)

Do some rearranging to simplify the terms inside the brackets

$$\begin{aligned}\nabla_\theta NLL(h_\theta; S) &= -\sum_{i=1}^{N}\left[y^{(i)}(1-\sigma(\theta^\top \mathbf{x}^{(i)})) + (1-y^{(i)})(-\sigma(\theta^\top \mathbf{x}^{(i)}))\right]\mathbf{x}^{(i)} \\ &= -\sum_{i=1}^{N}\left[y^{(i)} - y^{(i)}\sigma(\theta^\top \mathbf{x}^{(i)}) - \sigma(\theta^\top \mathbf{x}^{(i)}) + y^{(i)}\sigma(\theta^\top \mathbf{x}^{(i)})\right]\mathbf{x}^{(i)} \\ &= \sum_{i=1}^{N}\left[\sigma(\theta^\top \mathbf{x}^{(i)}) - y^{(i)}\right]\mathbf{x}^{(i)} \\ &= \sum_{i=1}^{N}\left[h_\theta(\mathbf{x}^{(i)}) - y^{(i)}\right]\mathbf{x}^{(i)}\end{aligned}$$

Logistic Regression - MLE (5)

It turns out, we can use a linear algebra representation like we did with linear regression

$$Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}, \quad \mathbf{h}_\theta = \begin{bmatrix} h_\theta(\mathbf{x}^{(1)}) \\ h_\theta(\mathbf{x}^{(2)}) \\ \vdots \\ h_\theta(\mathbf{x}^{(N)}) \end{bmatrix}, \quad X = \begin{bmatrix} — & \mathbf{x}^{(1)} & — \\ — & \mathbf{x}^{(2)} & — \\ & \vdots & \\ — & \mathbf{x}^{(N)} & — \end{bmatrix}$$

Using these we can rewrite the gradient of the NLL as

$$\nabla_\theta NLL(h_\theta; S) = X^\top(\mathbf{h}_\theta - Y)$$

Logistic Regression vs Linear Regression

It turns out that this loss function is convex, just like linear regression! (proof hint: find the Hessian, show that it is pos. def.). **We can use Gradient Descent**.

Actually, the gradient for Linear Regression and Logistic Regression are quite similar

Logistic Regression

$$\nabla_\theta NLL(h_\theta; S) = X^\top(\mathbf{h}_\theta - Y)$$

Linear Regression

$$\begin{aligned}\nabla_\theta \mathcal{L}_S(h_\theta) &= \frac{1}{N}[X^\top X\theta - X^\top Y] \\ &= \frac{1}{N}X^\top[X\theta - Y] \\ &= \frac{1}{N}X^\top[\mathbf{h}_\theta - Y]\end{aligned}$$

Logistic Regression with Regularization

Just like with Basis Function Expansion, we can also apply regularization to Logistic Regression:

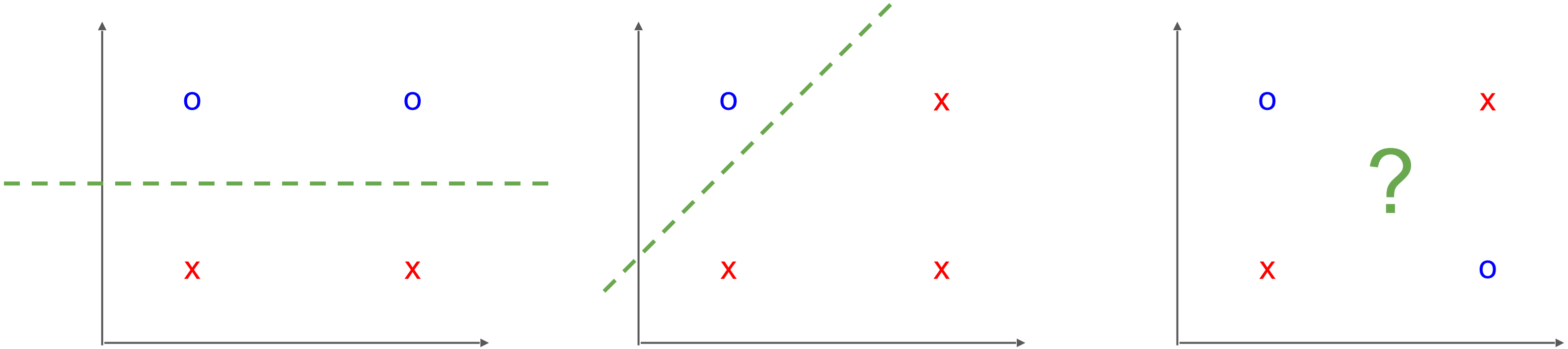
Regularized Logistic Regression

$$\nabla_\theta \left[NLL(h_\theta; S) + \lambda \|\theta\|^2 \right] = X^\top (\mathbf{h}_\theta - Y) + \lambda\theta$$

Regularization is important here, because otherwise gradient descent will "push" $||\theta|| \to \infty$ to make $p(y=y^{(i)} \mid h_\theta(x)) \to 1$ when the data is **linearly separable**.

Linear Separability

What kinds of data will Logistic Regression work well on?



Datasets where class can be separated by a straight line (hyperplane) are called **linearly separable**.

Composing multiple logistic functions

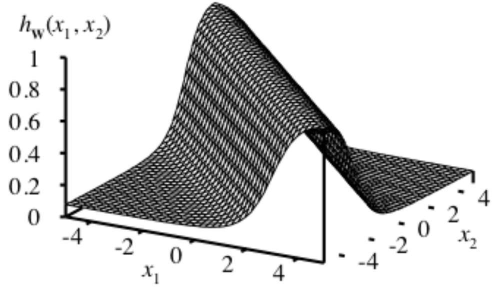
To handle non-linear datasets, we could use the Basis Function Expansion trick, or…

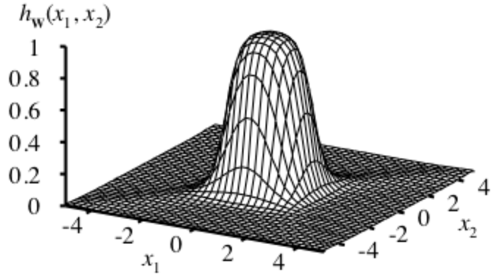
What happens if we **compose** a bunch of logistic functions?

$$h_{\theta_k}(\mathbf{x}) = \sigma(\theta_k^\top \mathbf{x})$$

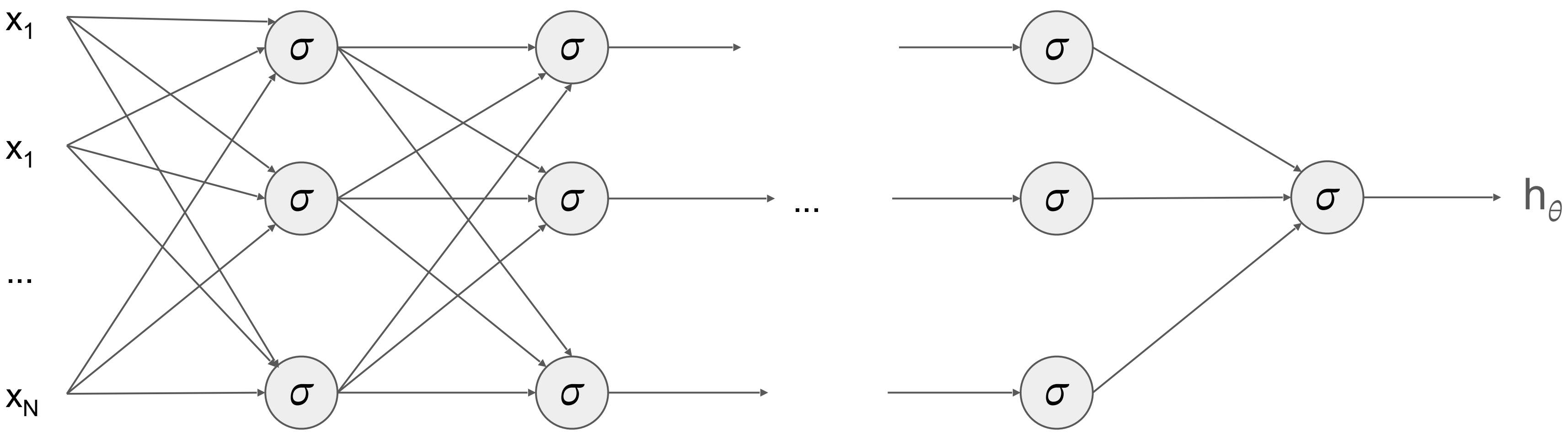
$$h_\psi(\mathbf{x}) = \sigma\left(\sum_{k=1}^{K} \psi_k h_{\theta_k}(\mathbf{x})\right)$$

$$= \sigma(\psi^\top \mathbf{h}_{\theta_k})$$





Feed-forward Neural Network preview



Summary and preview

Wrapping up

- Logistic Regression is a way of modeling the **probability** of the class label
- The MLE gives us a gradient that we can plug in to Gradient Descent to fit the model parameters
- Logistic Regression can fit **linearly separable** data well (to the point that we need to use **regularization** to prevent overfitting)

Next time

- Support Vector Machines