Models so far

We saw that a reasonable model for continuous outputs ($y \in \mathbb{R}$) is **linear regression**.

$$\textbf{Predict } y \in \textbf{ as} \begin{cases} y = \mathbf{x}^T\mathbf{w} & \text{(prediction function)} \\ p(y \mid \mathbf{x}, \mathbf{w}, \sigma^2) = \mathcal{N}(y \mid \mathbf{x}^T\mathbf{w}, \sigma^2) & \text{(probabilistic view)} \end{cases}$$

A reasonable model for *binary* outputs ($y \in \{0, 1\}$) is **logistic regression**:

$$\textbf{Predict } y \in \textbf{ as} \begin{cases} y = \mathbb{I}(\mathbf{x}^T\mathbf{w} > 0) & \text{(prediction function)} \\ p(y = 1 \mid \mathbf{x}, \mathbf{w}) = \sigma(\mathbf{x}^T\mathbf{w}) & \text{(probabilistic view)} \end{cases}$$

A reasonable model for *categorical* outputs ($y \in \{0, 1, \dots, C\}$) is **multinomial logistic regression**:

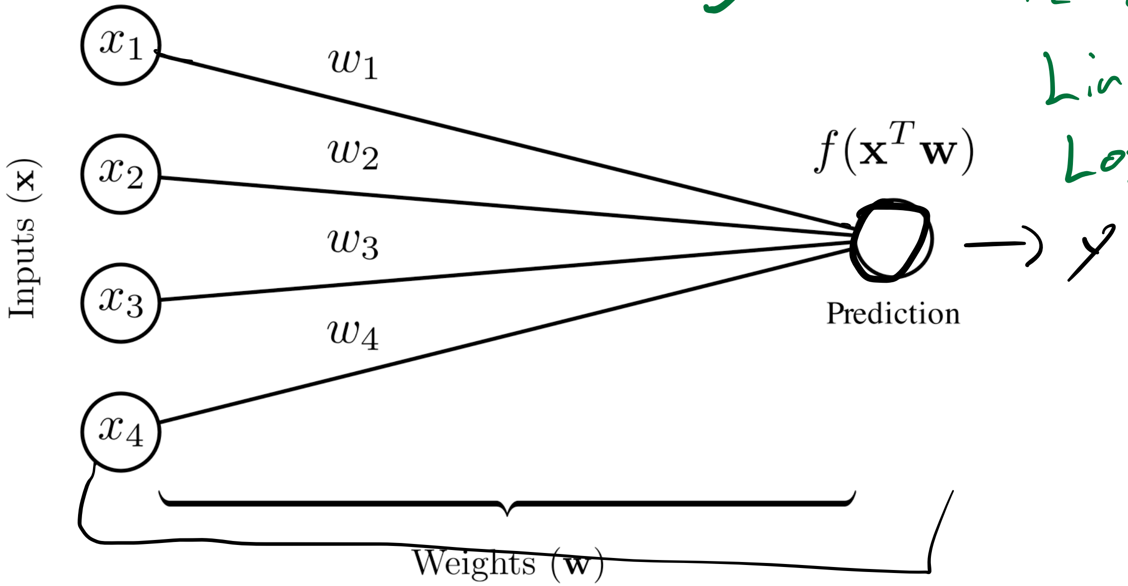
$$\textbf{Predict } y \in \textbf{ as} \begin{cases} y = \underset{c}{\operatorname{argmax}}\ \mathbf{x}^T\mathbf{w}_c & \text{(prediction function)} \\ p(y = c \mid \mathbf{x}, \mathbf{w}) = \operatorname{softmax}(\mathbf{x}^T\mathbf{W})_c & \text{(probabilistic view)} \end{cases}$$

$$\begin{bmatrix} W \end{bmatrix}^T$$

A new view

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \text{Weight} \\ \text{disp.} \\ \text{HP.} \\ \vdots \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \end{bmatrix}$$

weights



$$f(x^T w) = x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4$$

Linear $\quad y = x^T w \qquad f(a) = a$

Logistic $\quad y = \mathbb{I}(x^T w > 0) \qquad f(a) = \mathbb{I}(a > 0)$

Linear Predictions

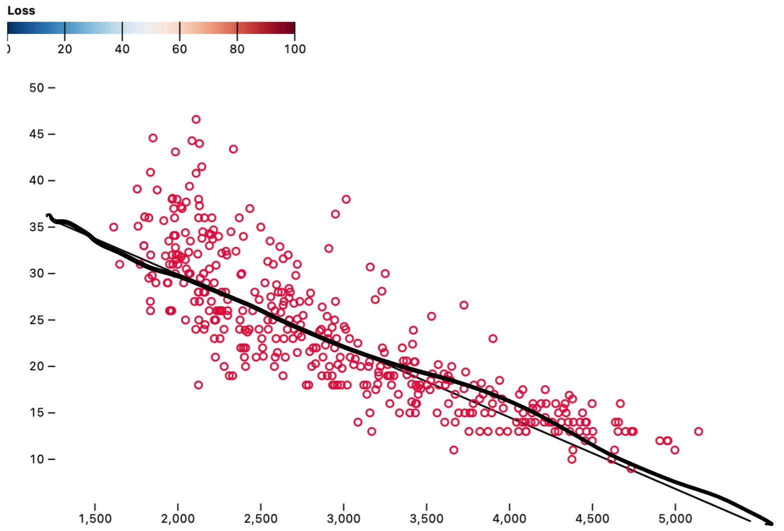
$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{w} = \sum_{i=1}^{n} x_i w_i$$

$$\begin{bmatrix} x \\ 1 \end{bmatrix} \begin{bmatrix} w \\ b \end{bmatrix}$$

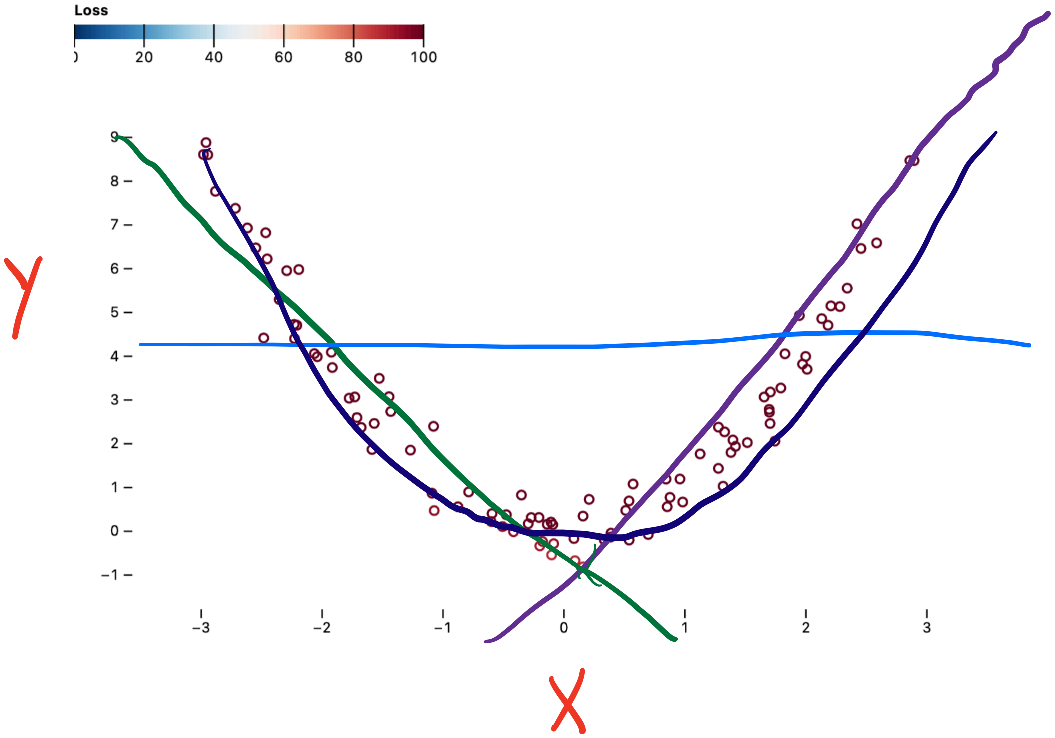
$$\text{Predicted MPG} = f(\mathbf{x}) =$$

$$(\text{weight})w_1 + (\text{horsepower})w_2 + (\text{displacement})w_3 + (\text{0-60mph})w_4 + b$$

$\cdot 1$



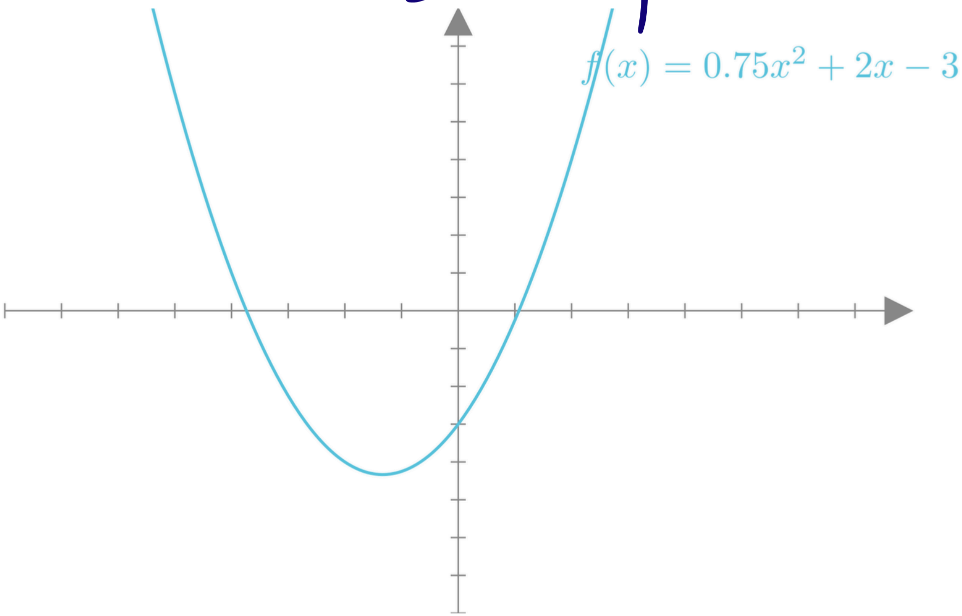
Non-Linear Data



Polynomial functions

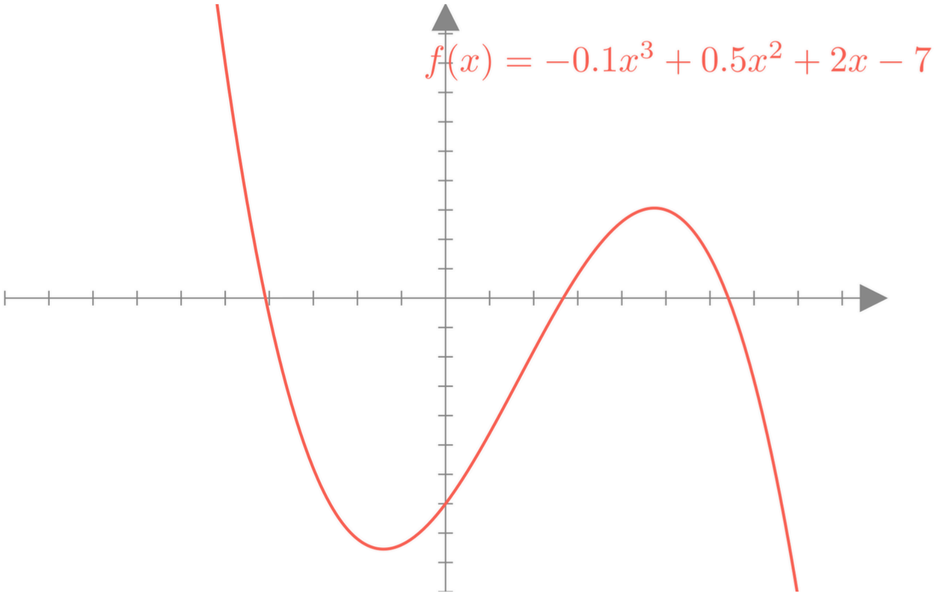
Quadratic functions

$$f(x) = w_2 x^2 + w_1 x + b$$



Cubic functions

$$f(x) = \overset{0}{\cancel{w_3}} x^3 + \overset{0}{\cancel{w_2}} x^2 + w_1 x + b$$



Polynomial functions of multiple inputs

$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

quadratic

$y = f(\mathbf{x}) = w_5 x_2^2 + w_4 x_1^2 + w_3 x_1 x_2 + w_2 x_2 + w_1 x_1 + b$

degree 4 :

$f(x) = w_1 x_1^4 + b$

$f(x) = w_1 x_1^2 x_2^2$

$w\, x_1^3 x_2$

degree 4

The ***degree*** of a polynomial is the largest exponent in any term of the polynomial

Polynomial functions as vector functions

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\underbrace{f(\mathbf{x}) = w_5 x_2^2 + w_4 x_1^2 + w_3 x_1 x_2 + w_2 x_2 + w_1 x_1 + b}_{W}$$

$$w_5 x_2^2 + w_4 x_1^2 + w_3 x_1 x_2 + w_2 x + w_1 y + b = \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}}_{\phi(x)} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ b \end{bmatrix}$$

$\phi(x)$: Feature ~~tr~~ transform of x

$$f(x) = \phi(x)^T W$$

Quadratic feature transforms

Let's consider the mapping from $\mathbf{x}$ to powers of the elements of $\mathbf{x}$. We'll call this mapping ϕ:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \xrightarrow[\phi]{} \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}$$

$$\phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w} = w_2 x_1^2 + w_1 x_1 + b, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_1^2 \\ 1 \end{bmatrix}$$

Fitting quadratic regression

Prediction function

$$f(\mathbf{x}) = \phi(\mathbf{x})^T\mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1x_2 \\ x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}$$

predict y

Probabilistic model

$$y_i \sim \mathcal{N}\left(\phi(\mathbf{x}_i)^T\mathbf{w}, \sigma^2\right)$$

$$\nabla_{\mathbf{w}}\mathbf{NLL}(\mathbf{w}, \mathbf{X}, \mathbf{y}) = \frac{1}{2\sigma^2}\sum_{i=1}^{N}(\phi(\mathbf{x}_i)^T\mathbf{w} - y_i)\phi(\mathbf{x}_i)$$

$w^{(i+1)} \leftarrow w^{(i)} - \alpha \nabla_w \text{loss}(x, y, w)$

Negative log-likelihood loss

$$\mathbf{Loss}(\mathbf{w}) = \mathbf{NLL}(\mathbf{w}, \mathbf{X}, \mathbf{y}) = -\sum_{i=1}^{N}\log p(y_i \mid \mathbf{x}_i, \mathbf{w})$$

$$= \frac{1}{2\sigma^2}\sum_{i=1}^{N}\left(y_i - \phi(\mathbf{x}_i)^T\boxed{\mathbf{w}}\right)^2 + N\log\sigma\sqrt{2\pi}$$

prediction func

$x^T w \longrightarrow \phi(x)^T w$

Fitting quadratic regression

Prediction function

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}$$

Probabilistic model

$$y_i \sim \mathcal{N}\left(\phi(\mathbf{x}_i)^T \mathbf{w}, \sigma^2\right)$$

$$\nabla_{\mathbf{w}} \mathbf{NLL}(\mathbf{w}, \mathbf{X}, \mathbf{y}) = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(\phi(\mathbf{x}_i)^T \mathbf{w} - y_i\right) \phi(\mathbf{x}_i)$$

Negative log-likelihood loss

$$\mathbf{Loss}(\mathbf{w}) = \mathbf{NLL}(\mathbf{w}, \mathbf{X}, \mathbf{y})$$

$$= \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \phi(\mathbf{x}_i)\right.$$

We see that the gradient doesn't change, it simply involves $\phi(\mathbf{x}_i)$ instead of $\mathbf{x}_i$.

$$\nabla_{\mathbf{w}} \mathbf{NLL}(\mathbf{w}, \mathbf{X}, \mathbf{y}) = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(\phi(\mathbf{x}_i)^T \mathbf{w} - y_i\right) \phi(\mathbf{x}_i)$$

Quadratic logistic regression

Prediction function

$$f(\mathbf{x}) = \mathbb{I}(\phi(\mathbf{x})^T\mathbf{w} \geq 0), \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1x_2 \\ x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}$$

$\mathbb{I}(x^T w \geq 0)$

Probabilistic model

$p(y=1|x,w)$

$$y_i \sim \mathbf{Bernoulli}(\sigma(\phi(\mathbf{x_i})^{\mathbf{T}}\mathbf{w})), \quad p(y_i = 1 \mid \mathbf{x}_i, \mathbf{w}) = \sigma(\phi(\mathbf{x}_i)^T\mathbf{w})$$

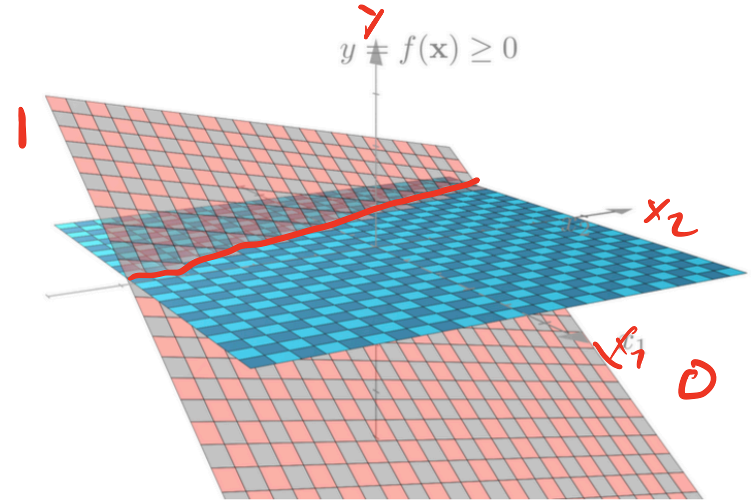
$f(x) = \mathbb{I}(x_1w_1 + x_2w_2 + x_1x_2w_3 + x_1^2w_4 + x_2^2w_5 + b \geq 0)$

Negative log-likelihood loss

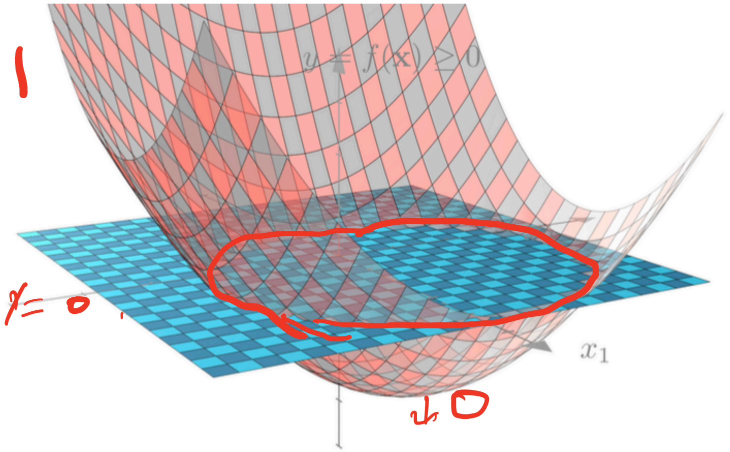
$$\mathbf{NLL}(\mathbf{w}, \mathbf{X}, \mathbf{y}) = -\sum_{i=1}^{N} \log \sigma((2y_i - 1)\boxed{\phi(\mathbf{x}_i)^T}\mathbf{w})$$

Quadratic decision boundary

Linear decision boundary



Quadratic decision boundary



$y = w_1 x_1^2 + w_2 x_2^2$

Cubic feature transform

$$[x_1] \xrightarrow[\phi]{} \begin{bmatrix} x_1 \\ x_1^2 \\ x_1^3 \\ 1 \end{bmatrix}$$

Prediction function (regression)

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w} = w_3 x_1^3 + w_2 x_1^2 + w_1 x_1 + b, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_1^2 \\ x_1^3 \\ 1 \end{bmatrix}$$

General feature transforms

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \\ x_1 x_2 \\ \sin(x_1) \\ \sin(x_2) \end{bmatrix}$$

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_6 \sin(x_1) + w_7 \sin(x_2)$$

General feature transforms

$$X = N \times d \quad \begin{bmatrix} x_{:1} & x_{:2} \\ x_{12} & x_{12} \end{bmatrix} x_1$$

$$\begin{bmatrix} x_{11}^2 & x_{12}^2 \\ x_{21}^2 & x_{22}^2 \end{bmatrix}$$

```python
squared_X = X ** 2                     # x^2
cross_X = X[:, :1] * X[:, 1:2]         # x_1 * x_2
sin_X = np.sin(X)                      # sin(x)

transformedX = np.concatenate([X, squared_X, cross_X, sin_X], axis=1)
```

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \\ x_1 x_2 \\ \sin(x_1) \\ \sin(x_2) \end{bmatrix}$$

$$\begin{bmatrix} x_{i1} & x_{i2} & x_{i1}^2 & x_{i2}^2 & x_{i1}x_{i2} & \sin x_{i1} \end{bmatrix}$$

$$\text{Phi}X = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix} \begin{bmatrix} x_{11}^2 & x_{12} \end{bmatrix} \begin{bmatrix} x_{11}x_{12} \end{bmatrix} \Big[$$

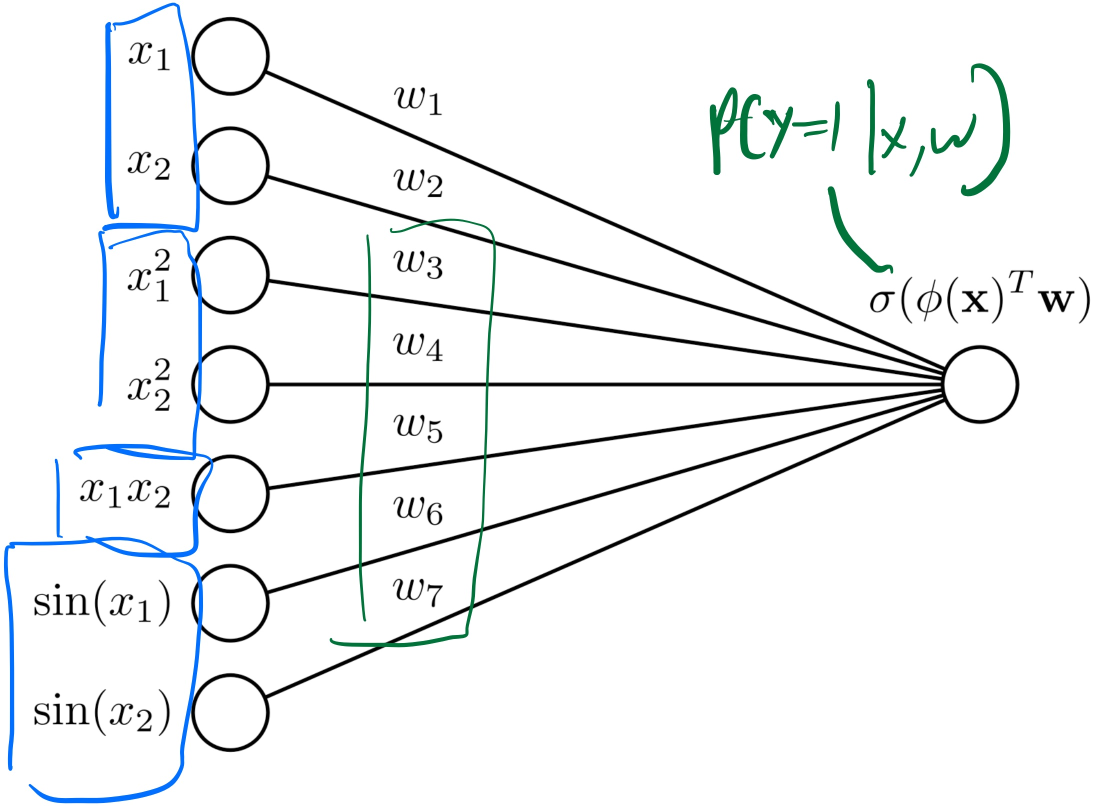
$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_6 \sin(x_1) + w_7 \sin(x_2)$$

$f(x^T w)$

Non-linear logistic regression

$$p(y=1 \mid \mathbf{x}, \mathbf{w}) = \sigma\left(\phi(\mathbf{x})^T \mathbf{w}\right)$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \\ x_1 x_2 \\ \sin(x_1) \\ \sin(x_2) \end{bmatrix}$$



Learning linear functions

We've already seen that we can *learn* a function by defining our function in terms of a set of *parameters* $\mathbf{w}$:

$$f(\mathbf{x}) = \mathbf{x}^T\boxed{\mathbf{w}}$$

and then minimizing a *loss* as a function of $\mathbf{w}$

$$\mathbf{w}^* = \underset{\mathbf{w}}{\text{argmin}}\ \mathbf{Loss}(\mathbf{w})$$

Which we can do with gradient descent:

$$\mathbf{w}^{(k+1)} \longleftarrow \mathbf{w}^{(k)} - \alpha \nabla_{\mathbf{w}} \mathbf{Loss}(\mathbf{w})$$

Can we learn the functions in our feature transform?

$g_1(x) = x^2$

$g_2(x) = x_1^2$

$g_3(x) = \sin(x)$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T\mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} g_1(x_1) \\ g_2(x_1) \\ g_3(x_1) \\ g_4(x_1) \end{bmatrix}$$

Can we learn the functions in our feature transform?

$$g_i(\mathbf{x}) = \sigma(\mathbf{x}^T \mathbf{w}_i)$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}, \quad \phi(\mathbf{x}) = \begin{bmatrix} \boxed{g_1(x_1)} \\ g_2(x_1) \\ g_3(x_1) \\ g_4(x_1) \end{bmatrix}$$

Logistic Regression

parameters of g_1

parameters of g_2

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}_0, \quad \phi(\mathbf{x}) = \begin{bmatrix} \sigma(\mathbf{x}^T \boxed{\mathbf{w}_1}) \\ \sigma(\mathbf{x}^T \boxed{\mathbf{w}_2}) \\ \sigma(\mathbf{x}^T \mathbf{w}_3) \\ \sigma(\mathbf{x}^T \mathbf{w}_4) \end{bmatrix}$$

Parameters of prediction func.

All parameters: w_0, w_1, w_2, w_3, w_4

A simple learned transform

$$w_1 = \begin{bmatrix} w_{11} \\ w_{12} \end{bmatrix}$$

Let's look at a very simple example:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \mathbf{w}_0 = \begin{bmatrix} w_{01} \\ w_{02} \\ w_{03} \end{bmatrix}$$

$$\underbrace{f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}_0}, \quad \phi(\mathbf{x}) = \begin{bmatrix} \sigma(\mathbf{x}^T\mathbf{w}_1) \\ \sigma(\mathbf{x}^T\mathbf{w}_2) \\ \sigma(\mathbf{x}^T\mathbf{w}_3) \end{bmatrix} = \begin{bmatrix} \boxed{\sigma(x_1 w_{11} + x_2 w_{12})} \\ \sigma(x_1 w_{21} + x_2 w_{22}) \\ \sigma(x_1 w_{31} + x_2 w_{32}) \end{bmatrix}$$

$\phi_1(x)$

In this case, we can write out our prediction function explicitly as:

$$f(x) = \phi(x)_1 w_{01} + \phi(x)_2 w_{02} + \phi(x)_3 w_{03}$$

$$= \sigma(x_1 w_{11} + x_2 w_{12}) w_{01} + \sigma(x_1 w_{21} + x_2 w_{22}) w_{02} + \sigma(x_1 w_{31} + x_2 w_{32}) w_{03}$$

A simple learned transform

Let's look at a very simple example:

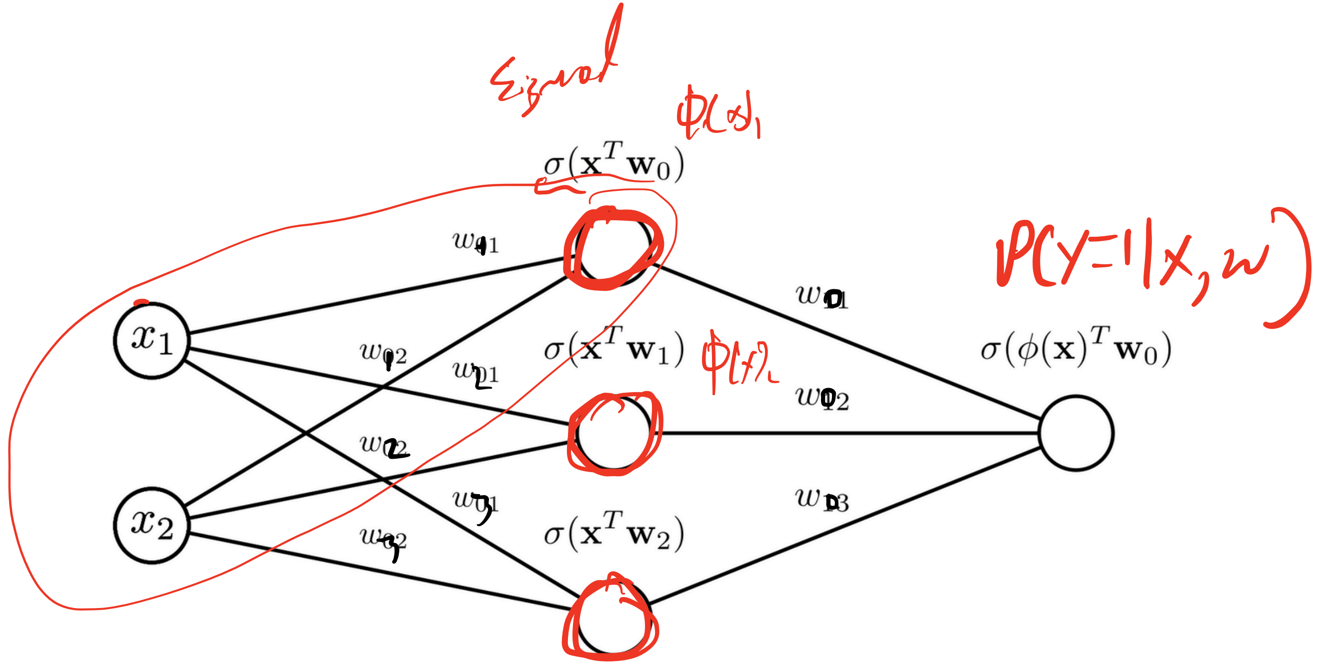
$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \mathbf{w}_0 = \begin{bmatrix} w_{01} \\ w_{02} \end{bmatrix}$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}_0, \quad \phi(\mathbf{x}) = \begin{bmatrix} \sigma(\mathbf{x}^T \mathbf{w}_1) \\ \sigma(\mathbf{x}^T \mathbf{w}_2) \\ \sigma(\mathbf{x}^T \mathbf{w}_3) \end{bmatrix} = \begin{bmatrix} \sigma(x_1 w_{11} + x_2 w_{12}) \\ \sigma(x_1 w_{21} + x_2 w_{22}) \\ \sigma(x_1 w_{31} + x_2 w_{32}) \end{bmatrix}$$

In this case, we can write out our prediction function explicitly as:

$$f(\mathbf{x}) = w_{01} \cdot \sigma(x_1 w_{11} + x_2 w_{12}) + w_{02} \cdot \sigma(x_1 w_{21} + x_2 w_{22}) + w_{03} \cdot \sigma(x_1 w_{31} + x_2 w_{32})$$

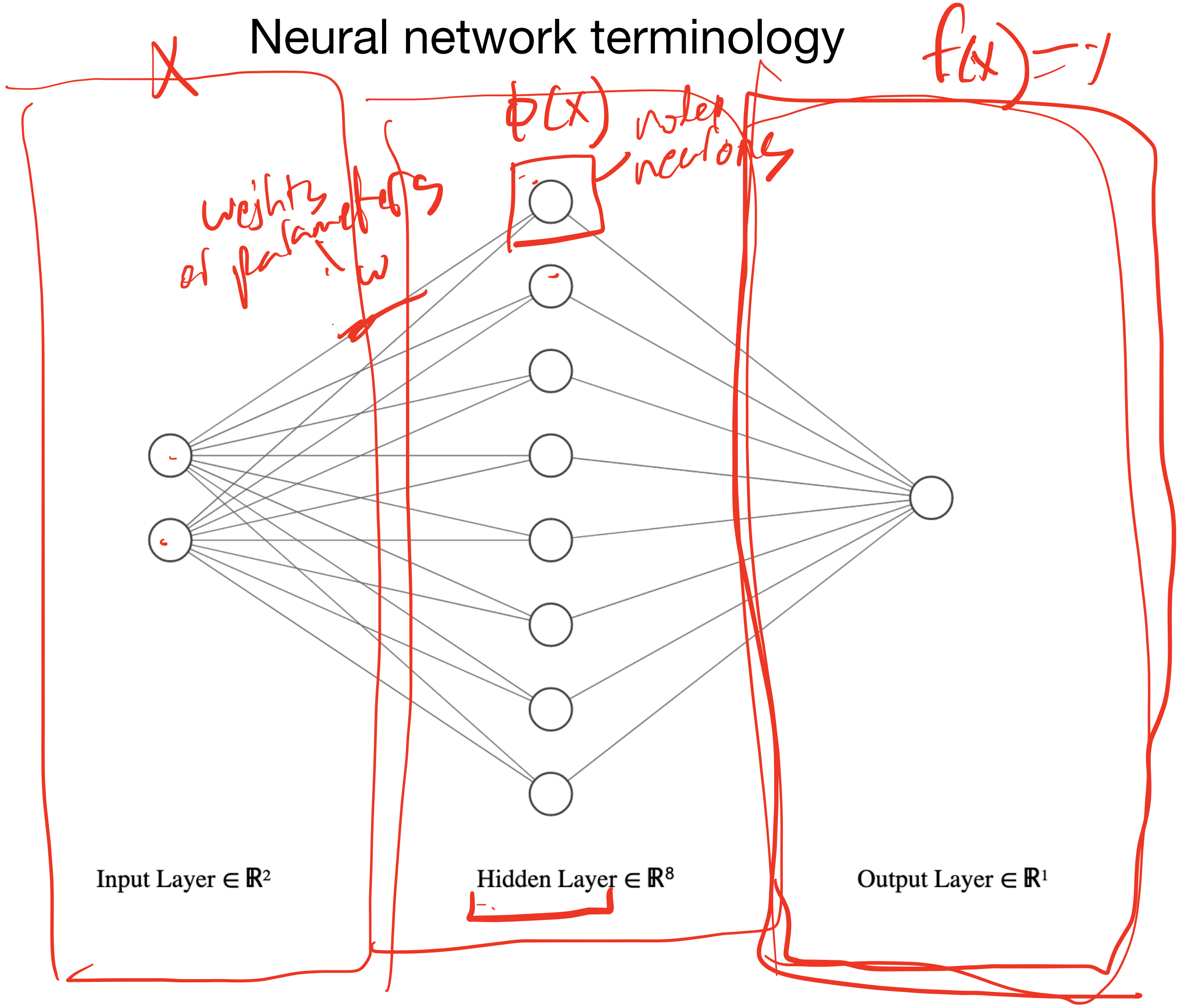
A neural network!



In this case, we can write out our prediction function explicitly as:

$$f(\mathbf{x}) = w_{01} \cdot \sigma(x_1 w_{11} + x_2 w_{12}) + w_{02} \cdot \sigma(x_1 w_{21} + x_2 w_{22}) + w_{03} \cdot \sigma(x_1 w_{31} + x_2 w_{32})$$

Neural network terminology



$$\mathbf{NLL}(\mathbf{W}, \mathbf{X}, \mathbf{y}) = -\sum_{i=1}^{N} \log\ \text{softmax}(\mathbf{x}_i^T \mathbf{W}^T)_{y_i} = -\sum_{i=1}^{N} \log \frac{e^{\mathbf{x}_i^T \mathbf{w}_{y_i}}}{\sum_{j=1}^{C} e^{\mathbf{x}_i^T \mathbf{w}_j}}$$

$$\mathbf{NLL}(\mathbf{W}, \mathbf{X}, \mathbf{y}) = -\sum_{i=1}^{N} \left(\mathbf{x}_i^T \mathbf{w}_{y_i} - \log \sum_{j=1}^{C} e^{\mathbf{x}_i^T \mathbf{w}_j} \right)$$