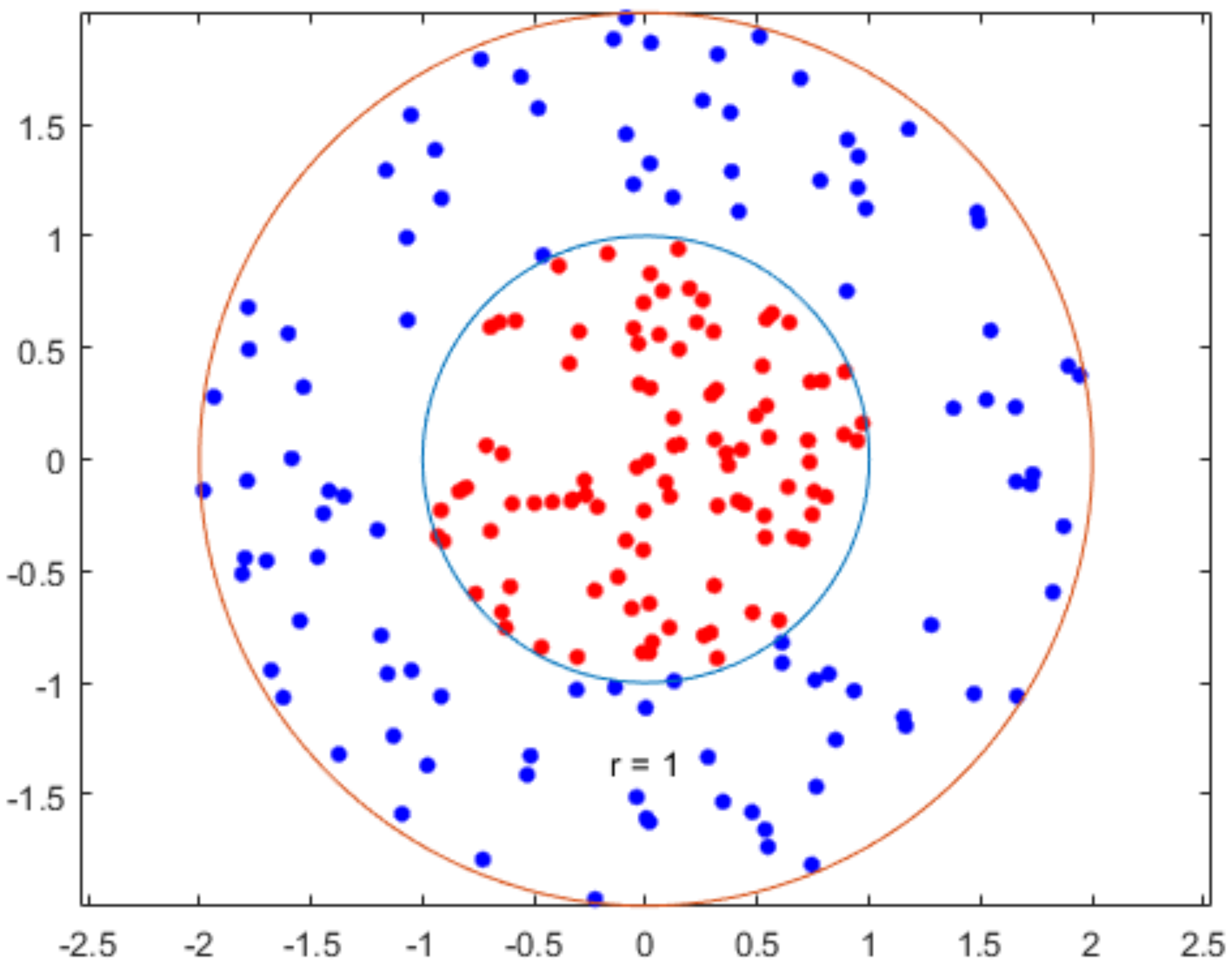
Data Representations

CMPUT 467: Machine Learning II

Projecting to Higher Dimensions Allows for Separability

- Consider this simple example where increasing from 2 to 3 dimensions (in a careful way) allows us to obtain linear separability



Brief Reminder: Linear Separability

- Logistic regression learns a hyperplane that attempts to separate points
  - Parameters $\mathbf{w}$ define a linear <u>decision boundary</u>
  - Observations on one side of decision boundary classified positive, other side negative
  - A dataset is <u>linearly separable</u> if there exists a linear decision boundary that perfectly classifies it

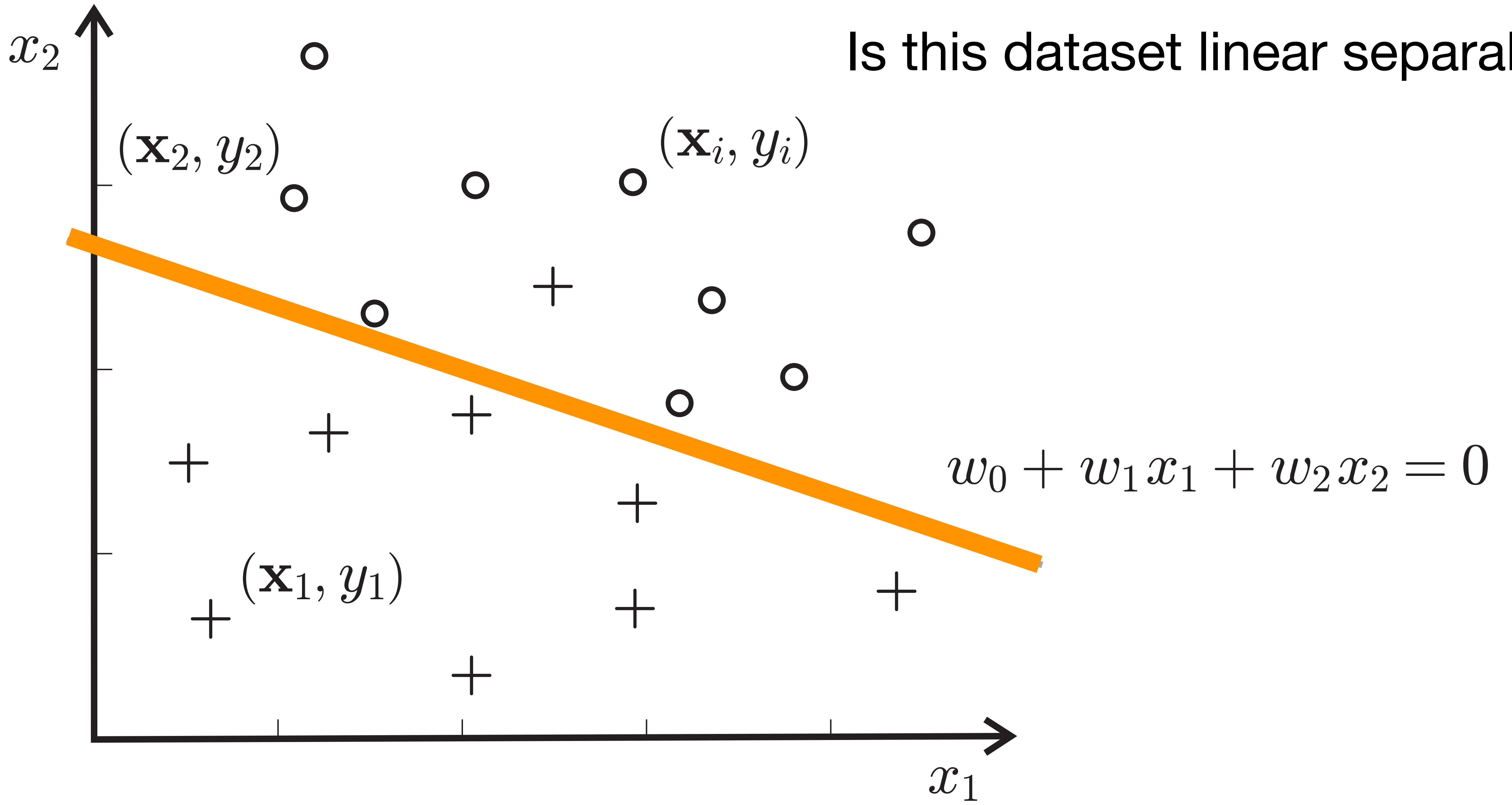
Is this dataset linear separable?

$$p(y=1\,|\,x) = \sigma(x^\top w) > 0.5 \text{ if } xw > 0$$

$$p(y=0\,|\,x) = 1 - \sigma(x^\top w) > 0.5$$

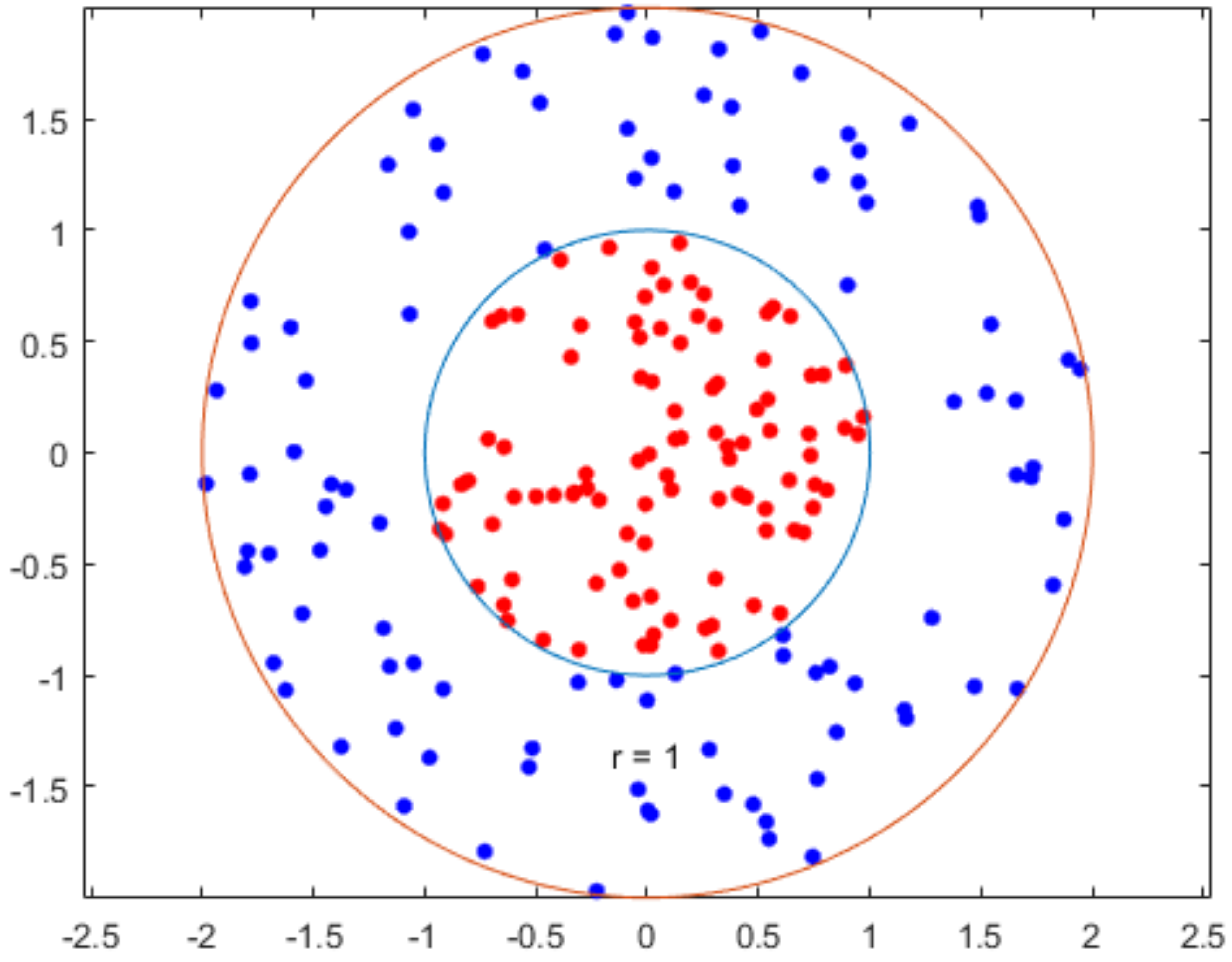
$$\text{if } \sigma(x^\top w) < 0.5$$

$$\text{if } x^\top w < 0$$



Back to Our Example

$$x_1^2 + x_2^2 = 1 \quad f(x) = x_1^2 + x_2^2 - 1$$



$$x_1 = x_2 = 0$$

$$\implies f(x) = -1 < 0$$

$$x_1 = 2, x_2 = -1$$

$$\implies f(x) = 4 + 1 - 1 = 4 > 0$$

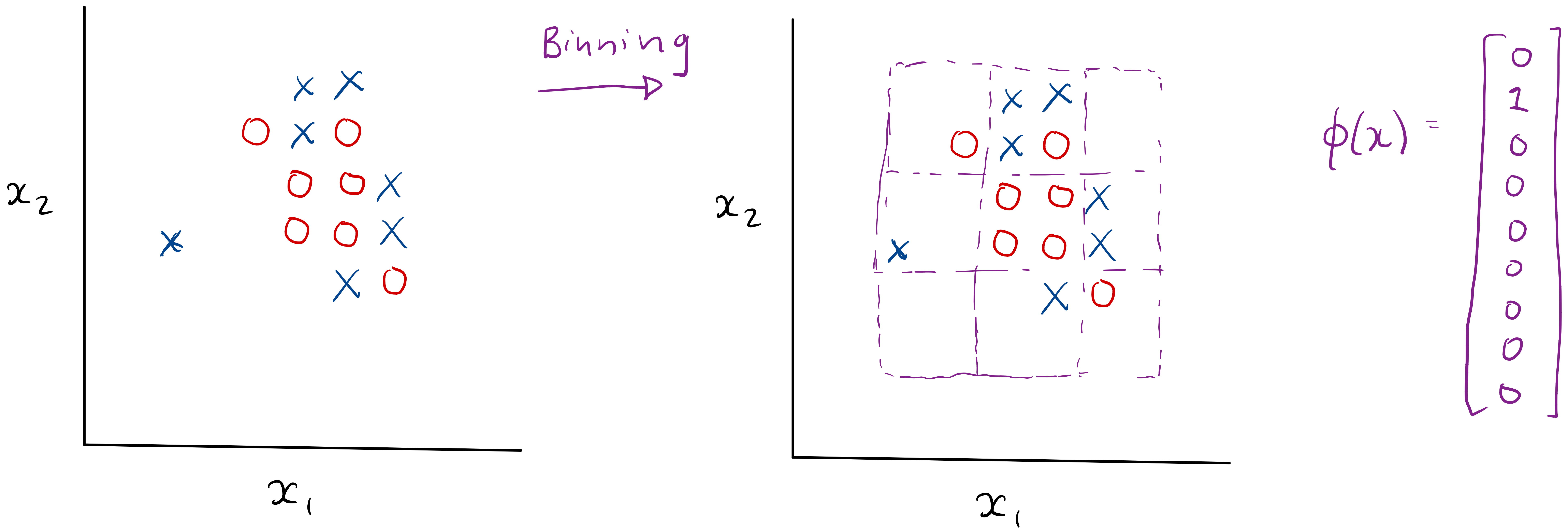
$$\boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ 1 \end{bmatrix} \qquad f(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \mathbf{w}$$

How to learn f(x) such that f(x) > 0 predicts positive and f(x) < 0 predicts negative?

May have to project higher

- **Cover's Theorem**: a dataset that is not linearly separable is highly likely to be separable by projecting to a higher-dimensional space with a nonlinear transformation

- One easy way to see this: consider a fine grained binning

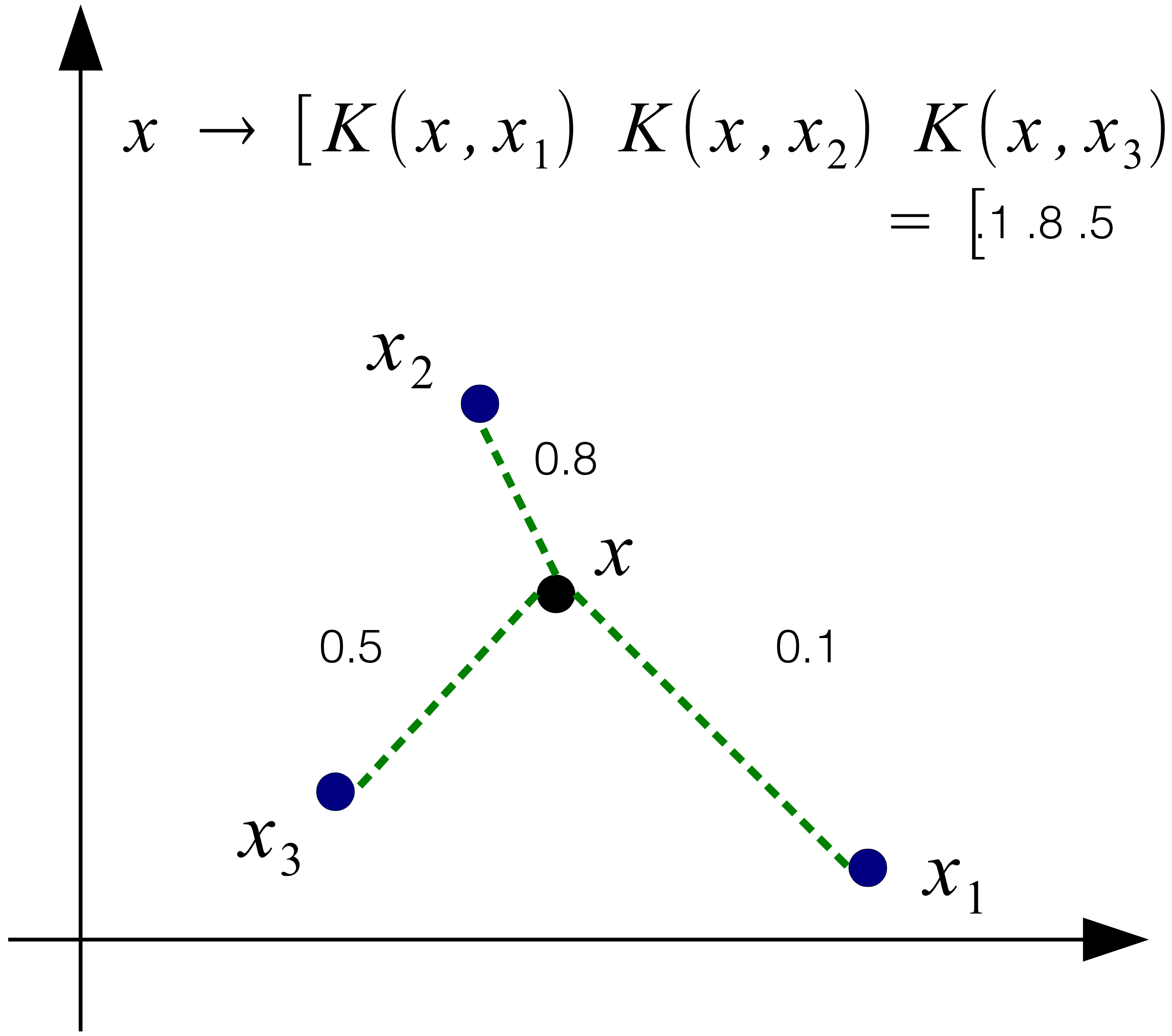


Exercise: What weights w are learned, assuming circle is the negative class?

This Theorem is One Motivation for Radial Basis Functions

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|_2^2}{\sigma^2}\right) \qquad f(\mathbf{x}) = \sum_{i=1}^{p} w_i k(\mathbf{x}, \mathbf{x}_i)$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_p) \end{bmatrix}$$

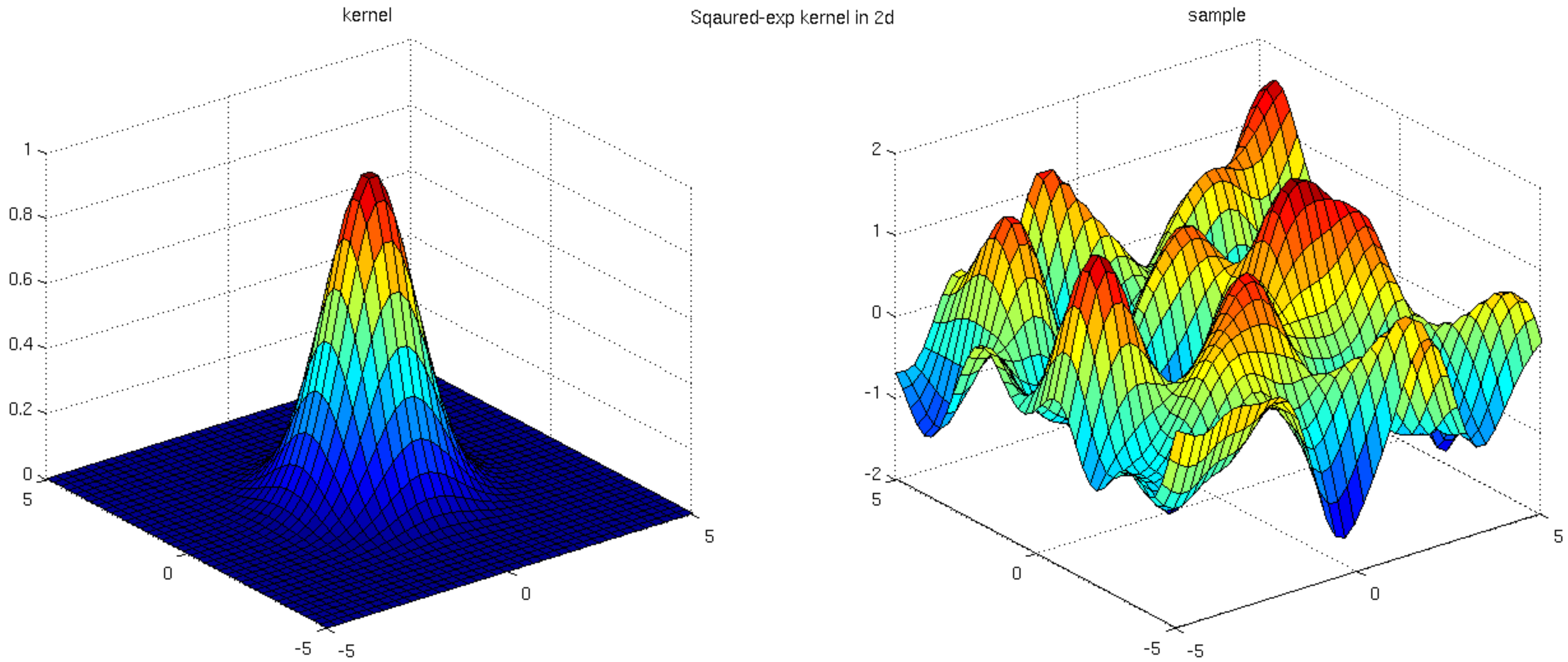


RBFs (continued)

$\mathbf{x}_i$ is a prototype or center

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|_2^2}{\sigma^2}\right) \qquad f(\mathbf{x}) = \sum_{i=1}^{p} w_i k(\mathbf{x}, \mathbf{x}_i)$$

Possible function f with several centers



Can learn a highly nonlinear function!

Other (similarity) transforms

- Linear kernel: $k(\mathbf{x}, \mathbf{c}) = \mathbf{x}^\top \mathbf{c}$

- Laplace kernel (Laplace distribution instead of Gaussian)

- Binning transformation $k(\mathbf{x}, \mathbf{c}) = \exp(-b\|\mathbf{x} - \mathbf{c}\|_1)$

$$s(\mathbf{x}, \mathbf{c}) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ in box around } \mathbf{c} \\ 0 & \text{else} \end{cases}$$

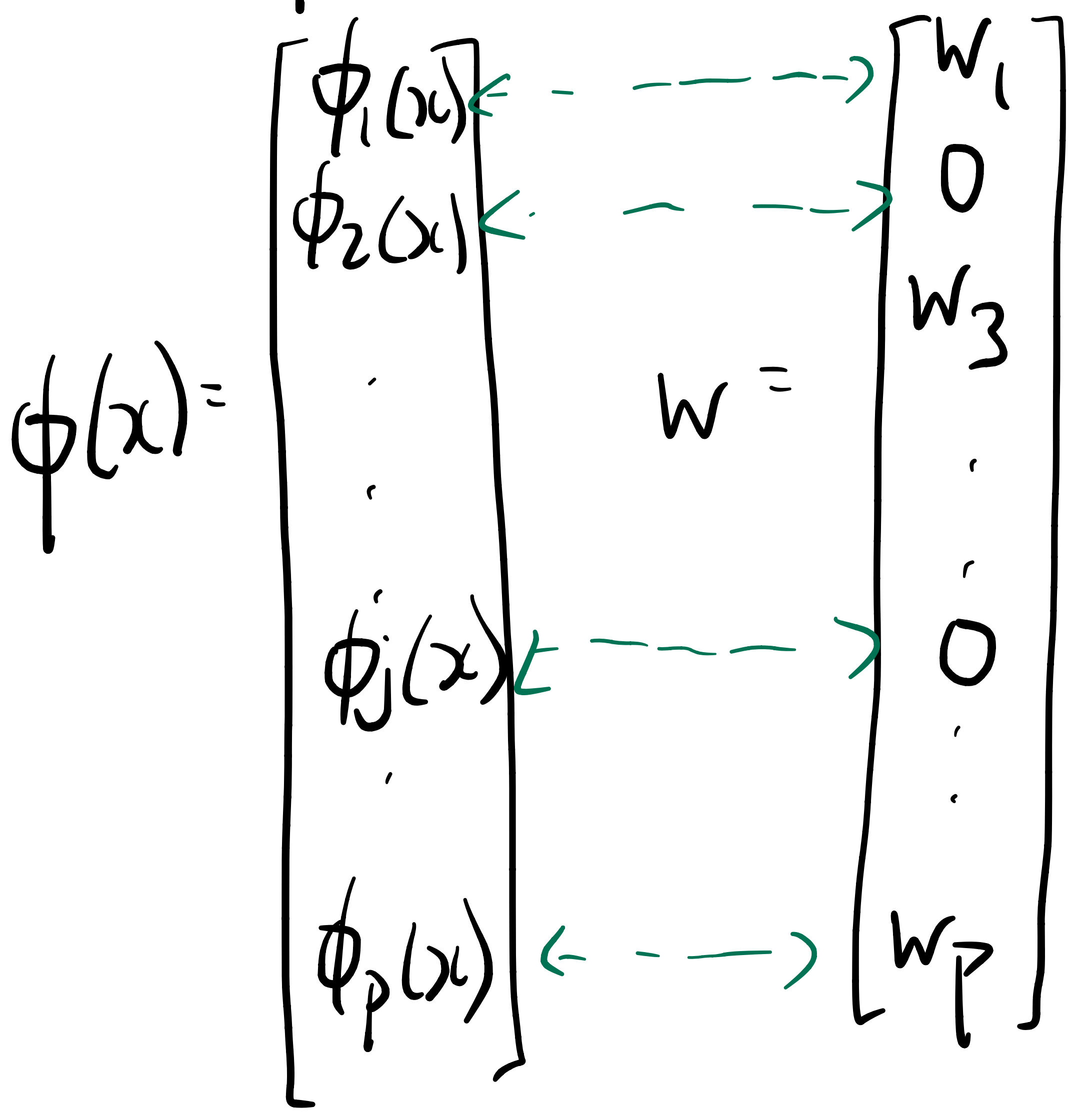
Picking prototypes

- The effectiveness of these methods depends heavily on how protoypes are picked
- One easy choice: use all of your data
  - Lots of features, really projected up!
- A more efficient choice: subselect a representative set of points
  - How? Many many algorithms, you will use l1 regularization

ℓ_1 regularization for feature selection

- Have feature vector $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{R}^p$
- When minimize $\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{\phi}(\mathbf{x}_i)\mathbf{w} - y_i)^2 + \lambda\|\mathbf{w}\|_1$, get back some weights that are zero. When a weight is zero, it is like a feature is removed

Dot product

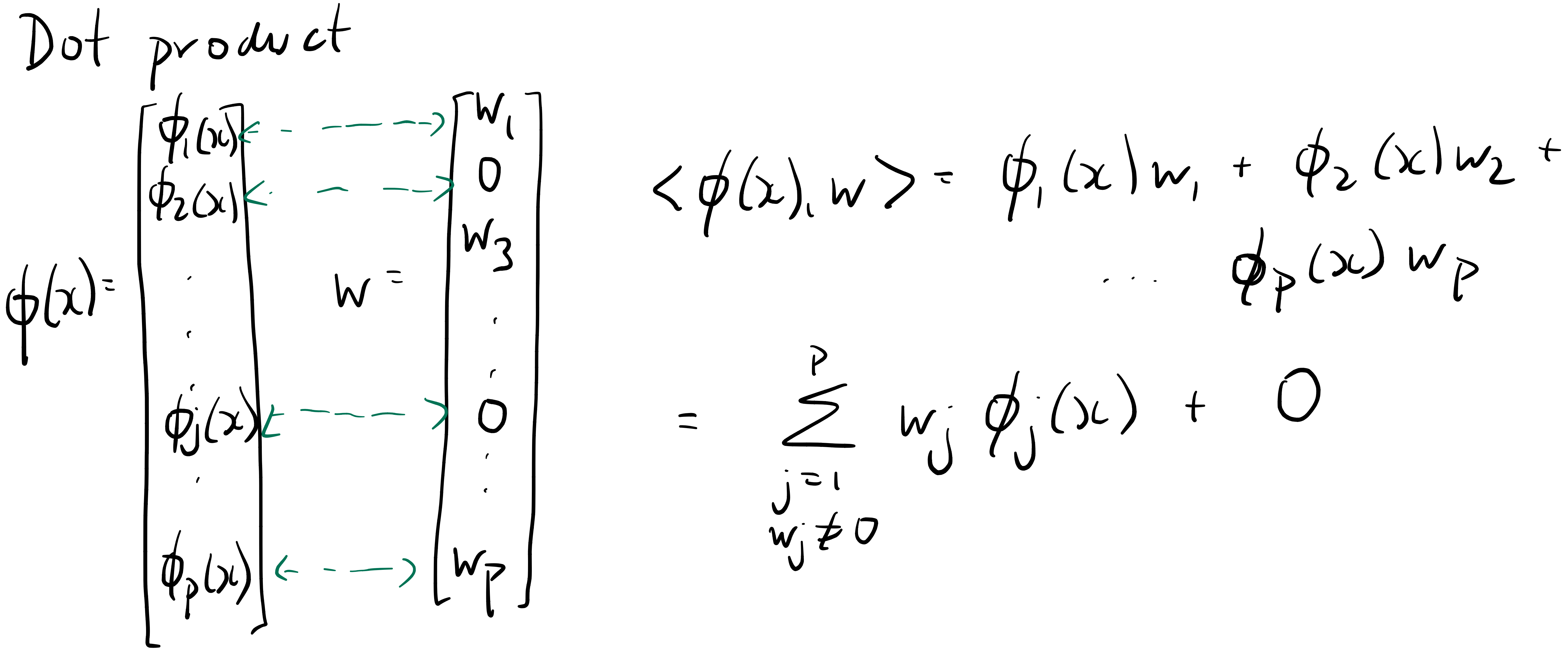


$\langle \phi(x), w \rangle = \phi_1(x)w_1 + \phi_2(x)w_2 + \dots \phi_p(x)w_p$

$= \sum_{j=1, w_j \neq 0}^{p} w_j \phi_j(x) + 0$

ℓ_1 regularization for prototype selection

- For prototype feature vectors $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{R}^p$, removing features is the same as removing a prototype
- The l1 regularization keeps only the most useful prototypes (selects a subset)



How do we control the number?

- The regularization parameter λ in $\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{\phi}(\mathbf{x}_i)\mathbf{w} - y_i)^2 + \lambda\|\mathbf{w}\|_1$ controls the level of sparsity but also shrinks the weights
- Larger λ will subselect more, but also bias the weights more
- We also might want to say: I want exactly 100 prototypes
- In your assignment, you will use this objective to find the most important weights, and then zero out the smallest weights to get exactly p prototypes