Complexity, Overfitting, and Cross-Validation

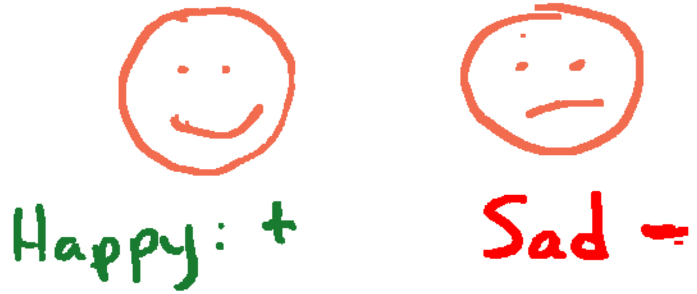
Objectives

- Create complex decision boundaries from measured data

- Examine complexity/performance tradeoffs
  - noise and errors
  - limited training data

- Introduce cross validation for performance evaluation

Add functions of data to generalize decision boundary



Features

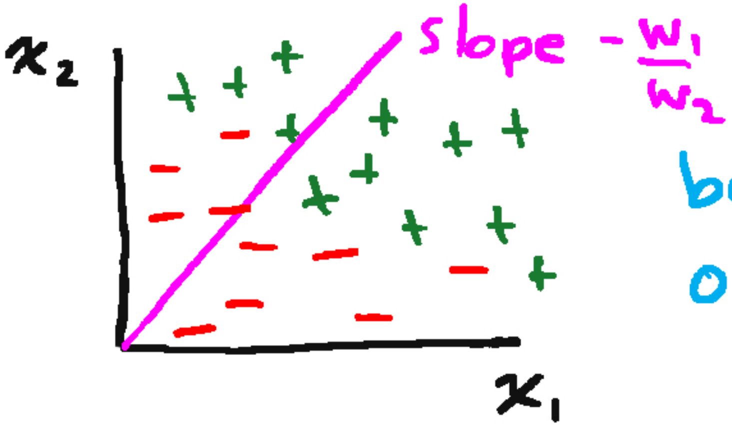
$y = \text{sign}(\underline{x}^T \underline{w})$

Boundary $\underline{x}^T \underline{w} = 0$

① $\underline{x}^T = [x_1 \;\; x_2]$

Boundary $x_1 w_1 + x_2 w_2 = 0$

$x_2 = -\frac{w_1}{w_2} x_1$

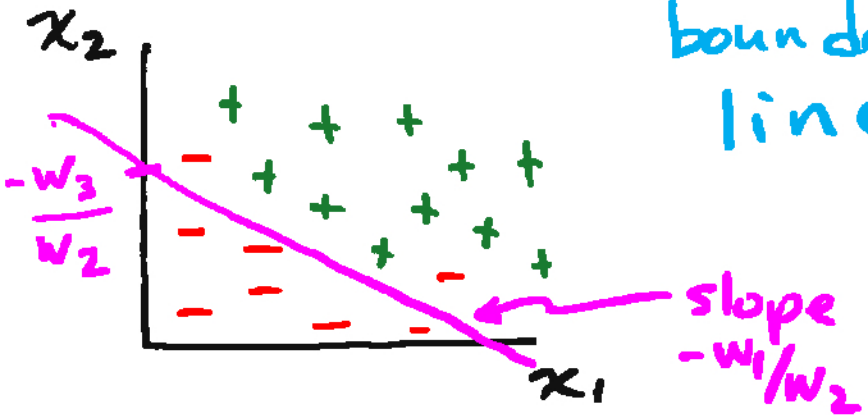


boundary through origin!

② $\underline{x}^T = [x_1 \;\; x_2 \;\; 1]$

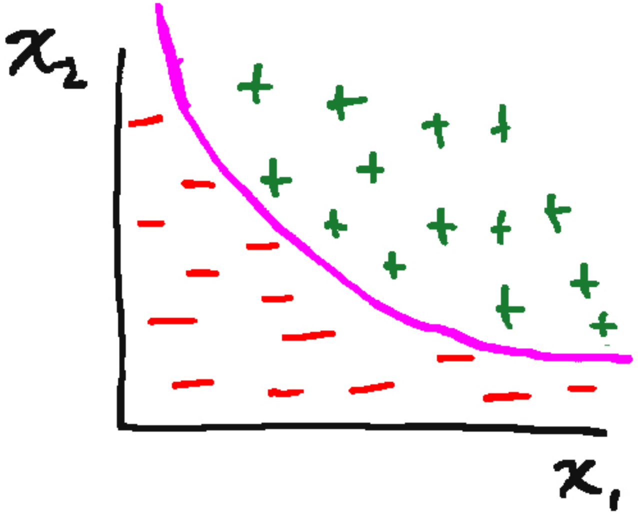
Boundary $x_1 w_1 + x_2 w_2 + w_3 = 0$

$x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_3}{w_2}$



boundary arbitrary line!

What if we want a nonlinear boundary?



Add functions of x_1, x_2 to $\underline{x}$

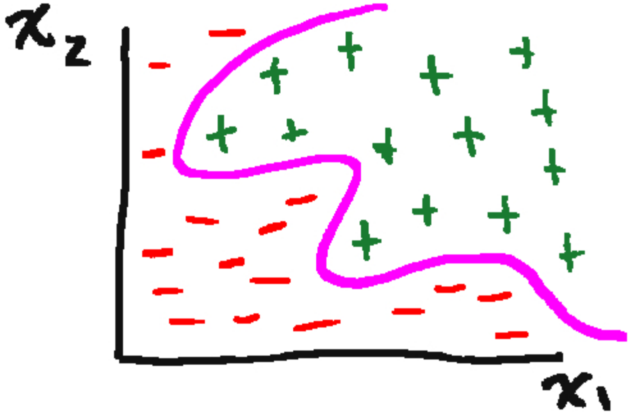
(3) $\underline{x}^T = [\, x_1 \;\; x_2 \;\; x_1^2 \;\; 1 \,] \Rightarrow \underline{x}^T \underline{w} = 0$

$$x_2 = \frac{-w_3}{w_2} x_1^2 - \frac{w_1}{w_2} x_1 - \frac{w_4}{w_2}$$

parabolic boundary

In general $\underline{x}^T = [\, x_1 \;\; x_2 \;\; 1 \;\; x_1 x_2 \;\; x_1^2 \;\; x_2^2 \;\; x_1^2 x_2 \; \cdots \,]$

arbitrarily complex boundary



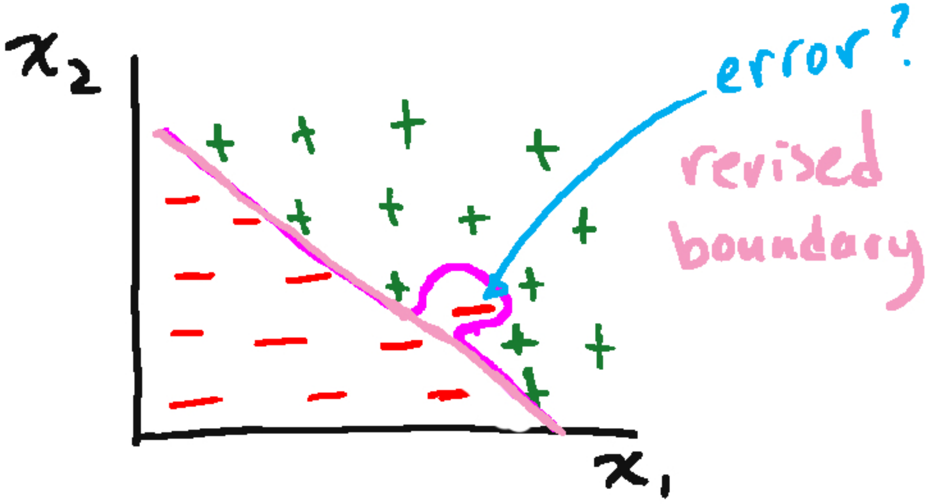
Includes simpler boundaries too!

Too good to be true?

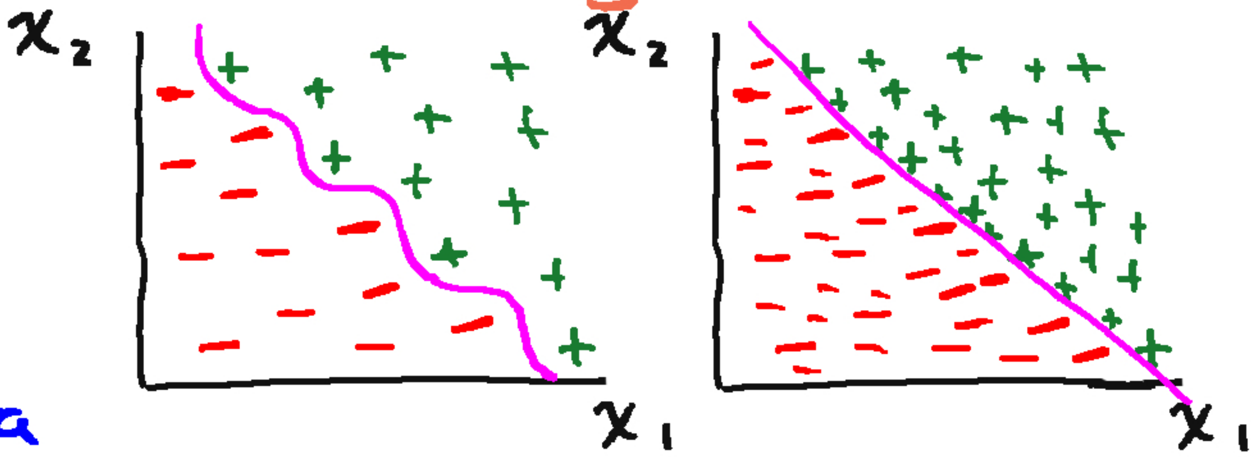
Number of model parameters ↑

Training data places limits on boundary complexity

1. Presence of noise / errors



2. Limited quantity of training data



Goal: Generalize to new data

Over fitting - using too complex a model for training data

$P = \dim(\underline{w}) \uparrow$, training error $\downarrow$ (☆ More complex weights → Less training error)

(But it may not generalize well to new data)

☆ In general, $P << N$

P: Number of parameters

N: Number of training samples

Cross validation measures generalization performance 5

- use subset of data to learn the model
- use held out data to evaluate performance

(☆ 每组里有 $\frac{N}{M}$ 个 $\{x, y\}$ 对)

Divide $\{\underline{x}_i, y_i, i=1,2,\cdots N\}$ into M groups $S_j = \{\underline{x}_i, y_i, i \in Q_j\}, j=1,2\cdots M$

Learn $\underline{w}_1$ from $S_2, S_3 \cdots S_M$. Compute $E_1 = \frac{1}{|Q_1|} \sum_{i \in Q_1} \frac{|y_i - \text{sign}(\underline{x}_i^T \underline{w})|}{2}$

Learn $\underline{w}_2$ from $S_1, S_3, \cdots S_M$. Compute E_2 using $\underline{w}_2, S_2$

$\vdots$ $\qquad\qquad$ $\vdots$

Learn $\underline{w}_M$ from $S_1, S_2, \cdots S_{M-1}$. Compute E_M using $\underline{w}_M, S_M$

$$E = \frac{1}{M} \sum_{i=1}^{M} E_i$$ Average CV error

M-fold cross validation

① $S_2 \cdots S_M$ for training, have a weight w_1

Validate using held-out S_1 for error E_1

② $S_1, S_3 \cdots S_M$ for training $\longrightarrow w_2$

Validate using held-out S_2 for E_2

☆ (The purpose is not to gain the final parameter w, but to evaluate models and the performance of hyperparameters)

↓

Evaluation tool, not a training tool

Cross validation for model comparison

- prefer model with smaller CV score

- manage model complexity for bias/variance tradeoff

- identical considerations apply to modeling of data. Squared error performance:

$$E_1 = \frac{1}{|Q_1|} \sum_{i \in Q_1} (y_i - \underline{x}_i^T \underline{w})^2$$