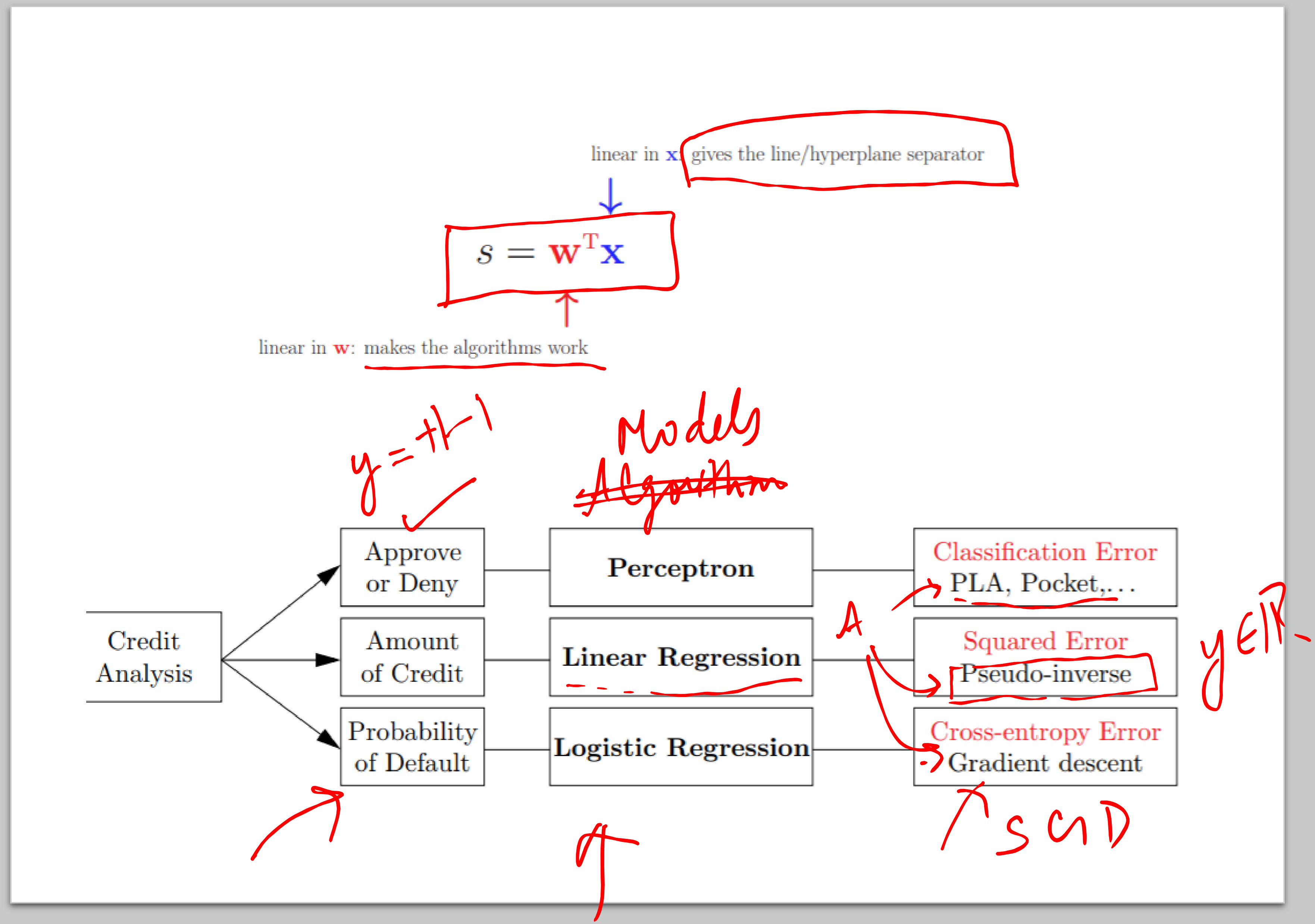
Machine Learning from Data

Lecture 10: Spring 2021

Today’s Lecture

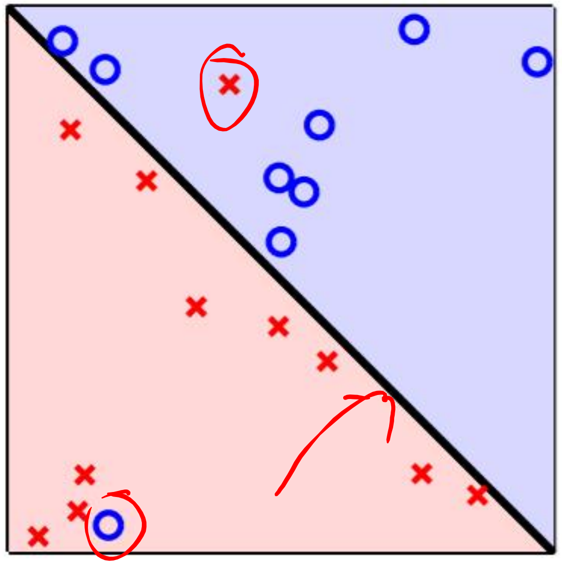
- Non-Linear Transforms
  - Z-Space
  - Polynomial Transforms

Linear Model (Recap)

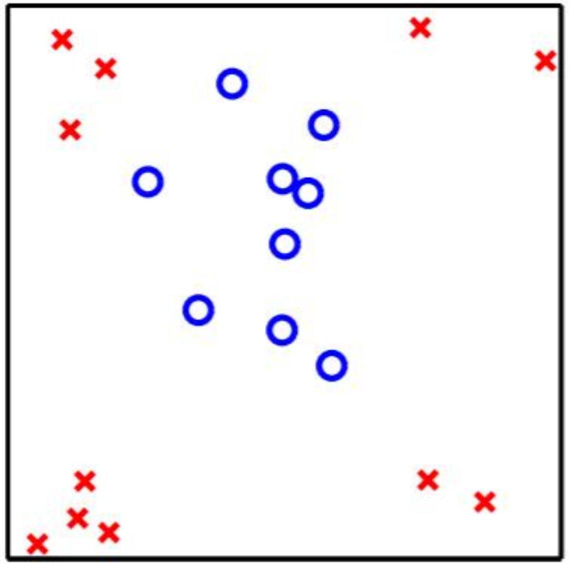


Limits of the Linear Model

PLA



(a) Linear with outliers

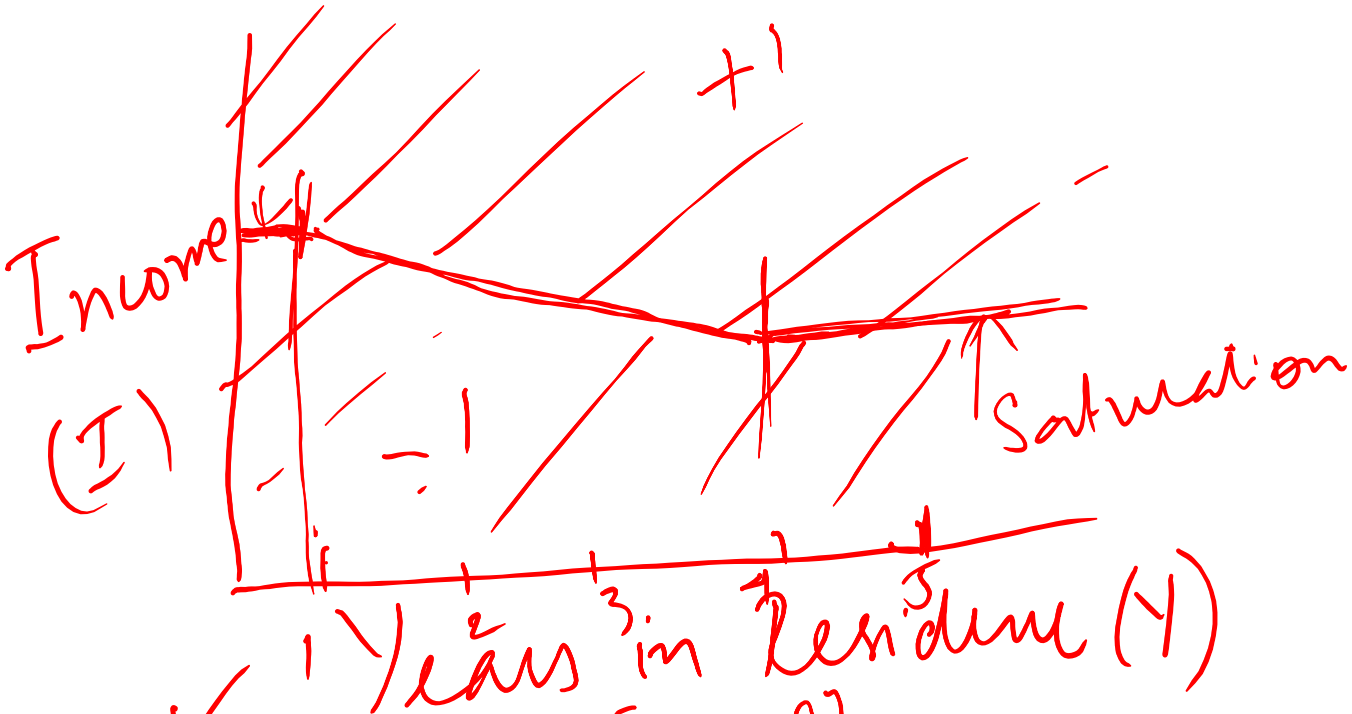


(b) Essentially nonlinear

To address (b) we need something more than linear.

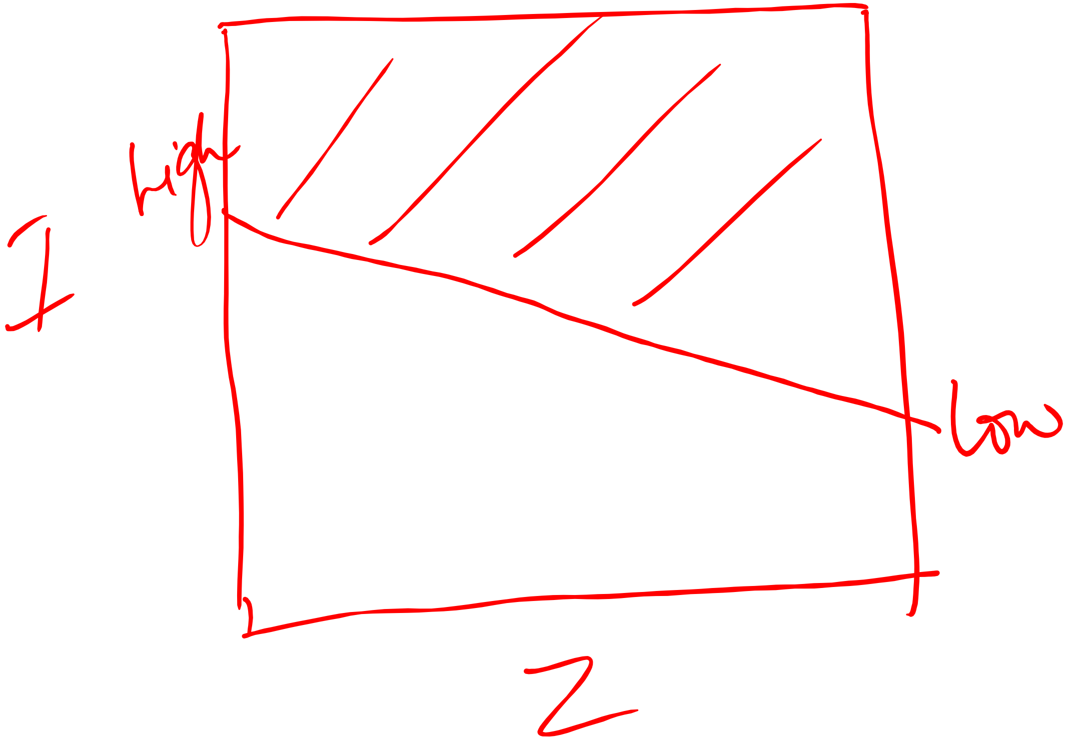
NON-LINEAR FEATURE TRANSFORM

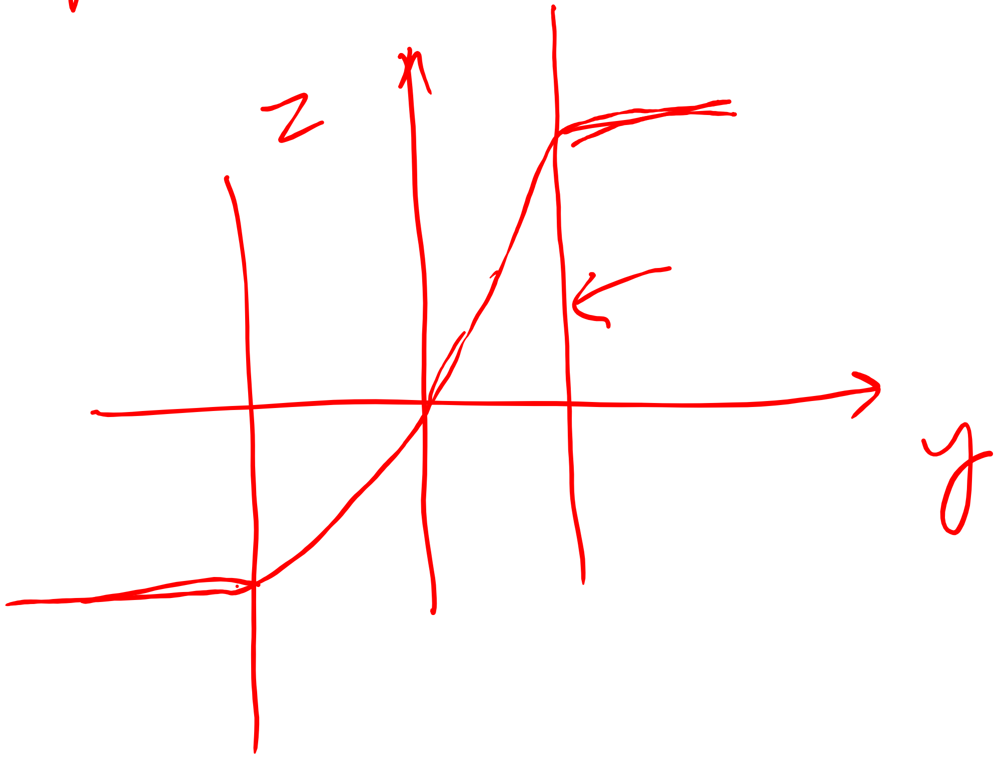
CREDIT LIMIT



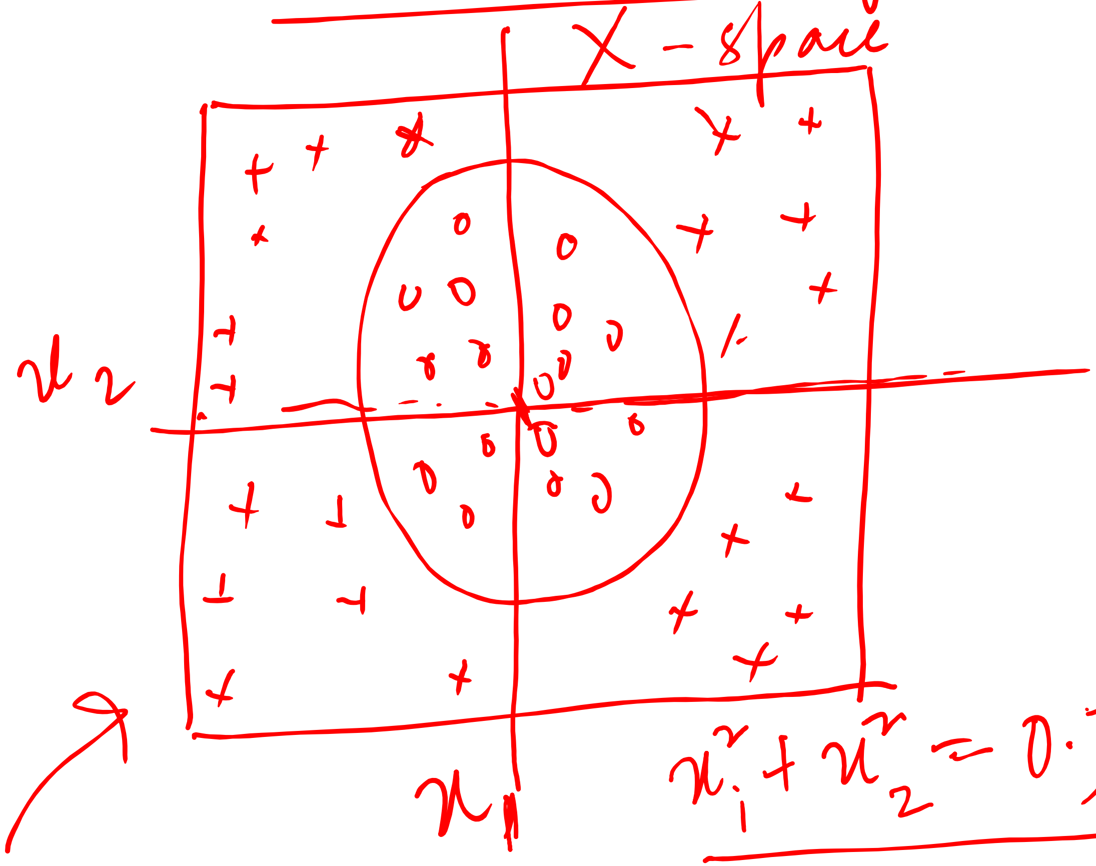
[[Y<1]] and [[Y>5]]

Take feature Y $\xrightarrow{\text{transform}}$ Z





Mechanics of Non-linear Transform



$x_1^2 + x_2^2 = 0.5$

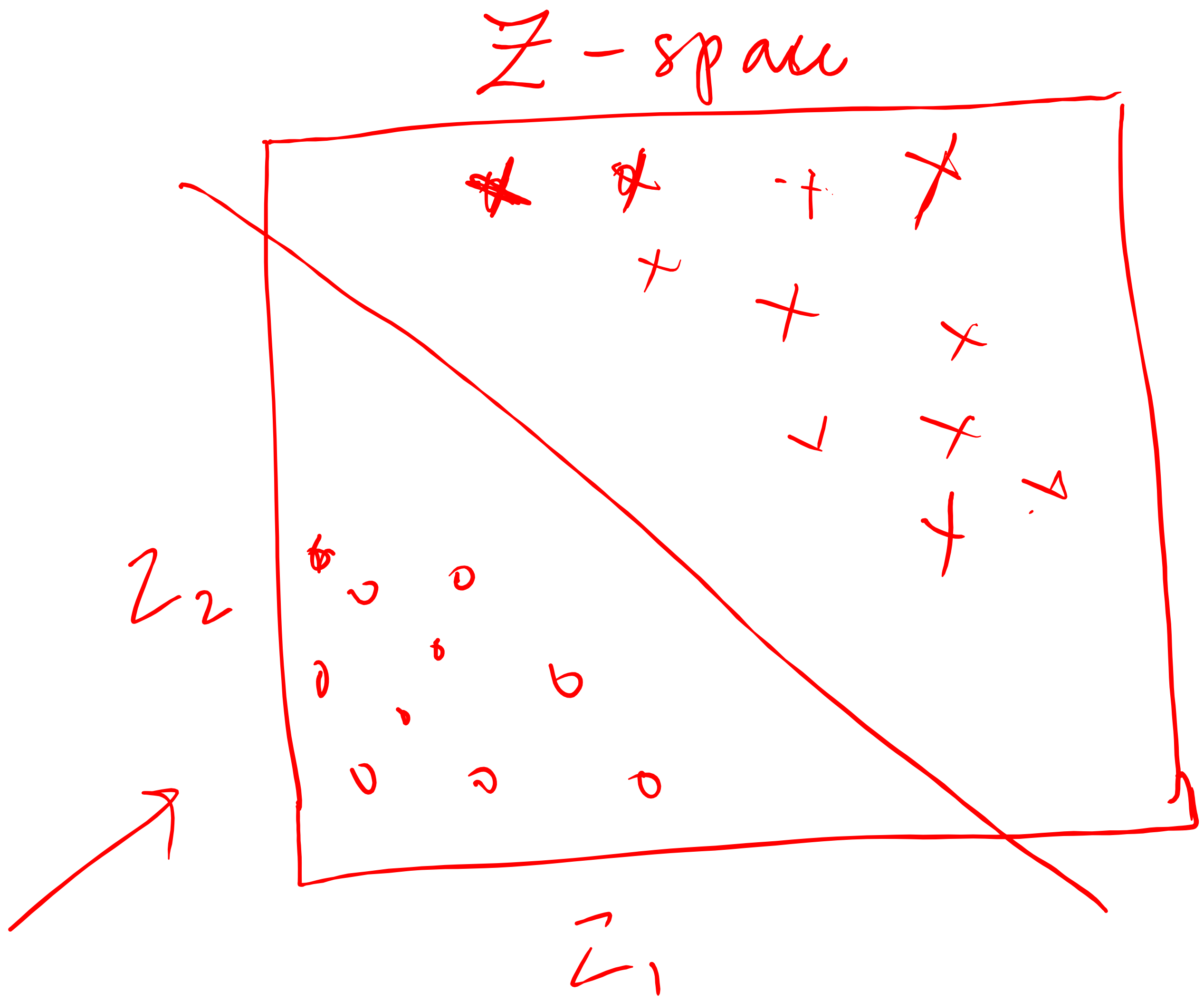
$$x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \xrightarrow[\text{'fi'}]{\Phi} z = \begin{bmatrix} 1 \\ z_1 \\ z_2 \end{bmatrix}$$

$$z_0 = 1$$
$$z_1 = x_1^2$$
$$z_2 = x_2^2$$

$$= \begin{bmatrix} 1 \\ \phi(x_1) \\ \phi(x_2) \end{bmatrix}$$

Problem itself.

$$\phi(x) = [1, x_1^2, x_2^2]$$



PLA

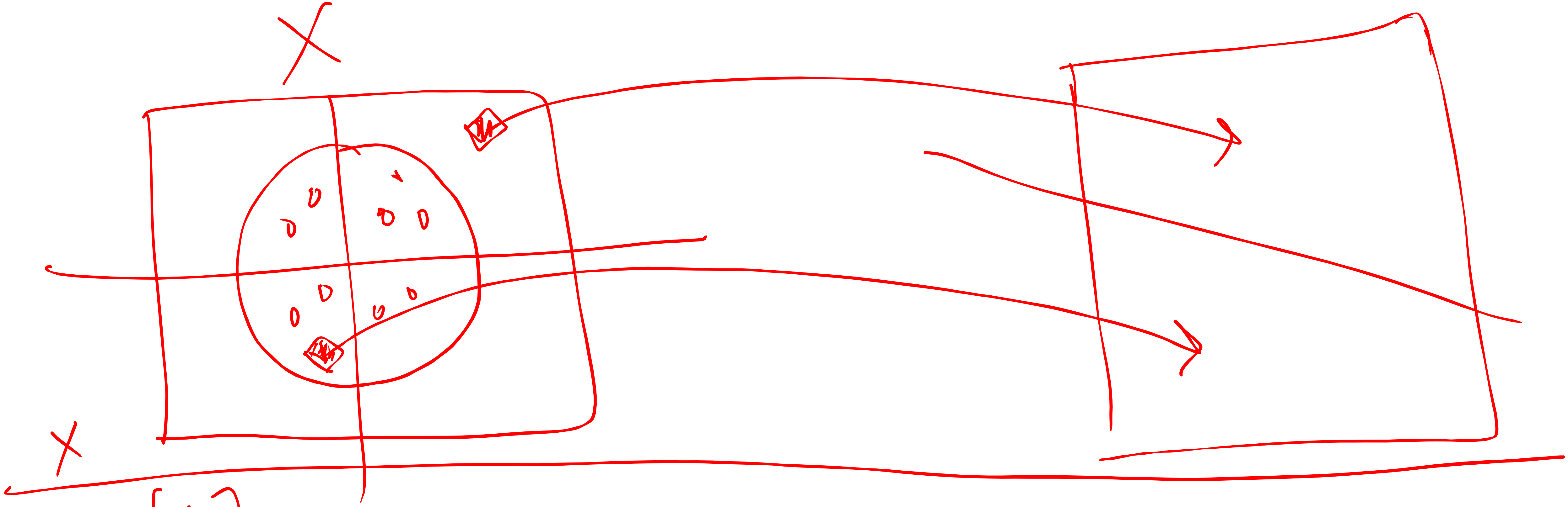
$\tilde{g}(z) = \text{sign}(\tilde{w}^T z)$

Take that test point

→ apply Φ

→ get value in Z-space

$\Phi(x_i)$



$$x = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$$

$$g(x) = \tilde{g}(\phi(x)) = \text{sign}(\tilde{w}^T \phi(x))$$

Input $= \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$

Data: $x_1 \; x_2 \; \cdots \; x_N$

$y_1 \; y_2 \; \cdots \; y_N$

No weights

$$g(x) = \tilde{g}(\phi(x))$$

$$= \text{sign}(\tilde{w}_{lin}^T \phi(x))$$

$$z = \phi(x)$$

$$= \begin{bmatrix} 1 \\ z_1 \\ \vdots \\ z_d \end{bmatrix} = \begin{bmatrix} 1 \\ \phi(x_1) \\ \vdots \\ \phi(x_d) \end{bmatrix}$$

$z_1 \; z_2 \; \cdots \; z_N$

$y_1 \; y_2 \; \cdots \; y_N$

$$h(z) = \text{sign}(\tilde{w}^T z)$$

$$\tilde{w}_{lin} = \begin{bmatrix} \tilde{w}_0 \\ \vdots \\ \tilde{w}_d \end{bmatrix}$$

$$\tilde{g}(z) = \text{sign}(\underline{\underline{\tilde{w}_{lin}^T z}})$$

Generalizations

X-space

$$X \in \{1\} \times \mathbb{R}^d$$

$$d_{vc} = d+1$$

$$E_{out}(g) \le E_{in}(g) + O\left(\sqrt{\frac{\tilde{d}\ln N}{N}}\right)$$

Z-space

$$Z \in 1 \times \mathbb{R}^{\tilde{d}}$$

$$\tilde{d}_{vc} = \tilde{d}+1$$

→ What should $\tilde{d}$ be?

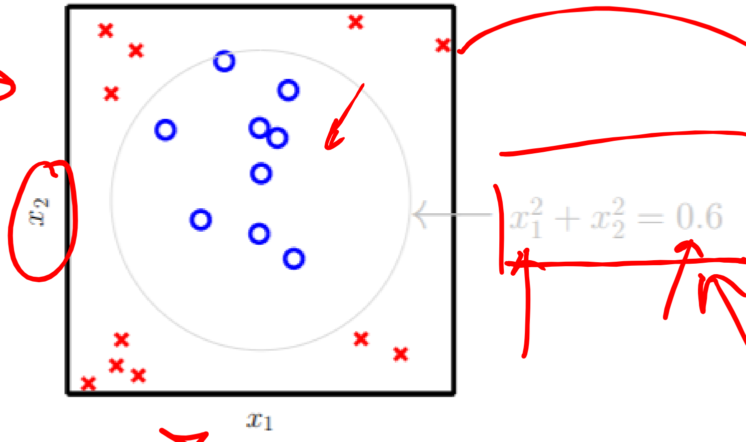
We want $\tilde{d}$ to be small → $E_{out} \simeq E_{in}$

Many Non-Linear Transforms May Work

Looked at Data

Issue?

X

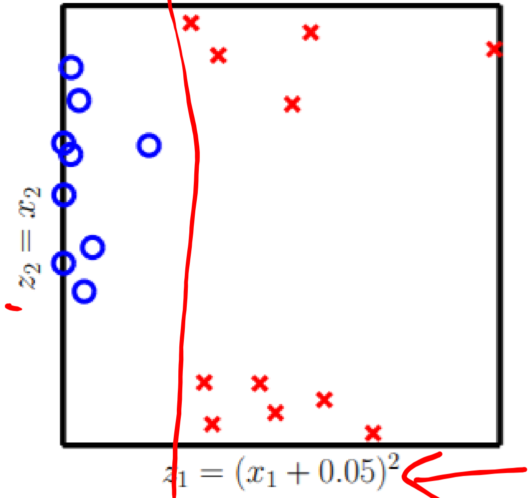


line

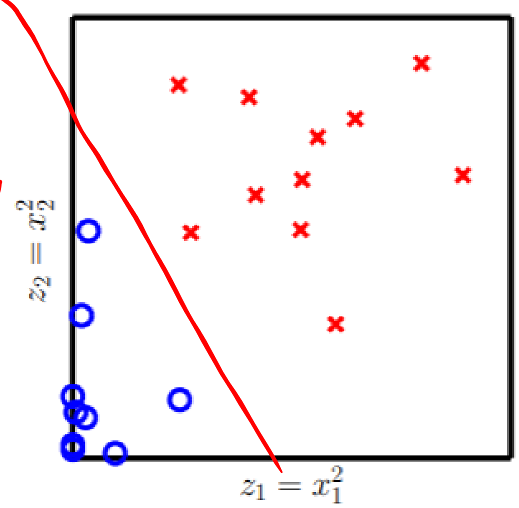
x X

2-d

Z

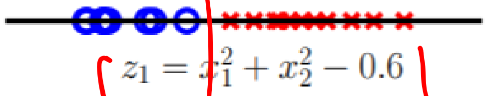


Z



Z

$d_{vc} = 2$



1-d

space

How to use feature transform in Practice?

→ Must choose the Non-linear feature transform before we see the data. (Domain, feature const. saturation)

POLYNOMIAL FEATURE TRANSFORM

2-d

$(1, x_1, x_2) \xrightarrow[\Phi_1]{\text{degree 1}} (1, x_1, x_2)$ $\quad \tilde{d}_{vc} = 3$

$(1, x_1, x_2) \xrightarrow[\Phi_2]{\text{degree 2}} (1, x_1, x_2, x_1^2, x_2^2, x_1x_2)$ $\quad \tilde{d}_{vc} = 6$

$(1, x_2, x_3) \xrightarrow[\Phi_3]{\text{degree 3}} \underbrace{(1, x_1, x_2, x_1^2, x_2^2, x_1x_2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3)}$ $\quad \tilde{d}_{vc} = 10$

$\vdots$ $\quad \tilde{d}_{vc} = 15$

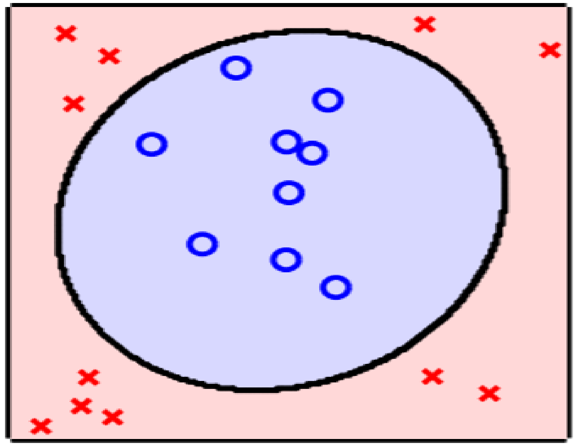
degree Q, input $= d$ $\qquad \underline{\tilde{d}_{vc} = \binom{Q+d}{Q}}$ $\qquad E_{out} \neq E_{in}$

Choose Transform before Looking at the data

After constructing features carefully, **before** seeing the data . . .

. . . if you think linear is not enough, try **the 2nd order polynomial transform**.

$$\begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} = \mathbf{x} \longrightarrow \boldsymbol{\Phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ \Phi_1(\mathbf{x}) \\ \Phi_2(\mathbf{x}) \\ \Phi_3(\mathbf{x}) \\ \Phi_4(\mathbf{x}) \\ \Phi_5(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}$$

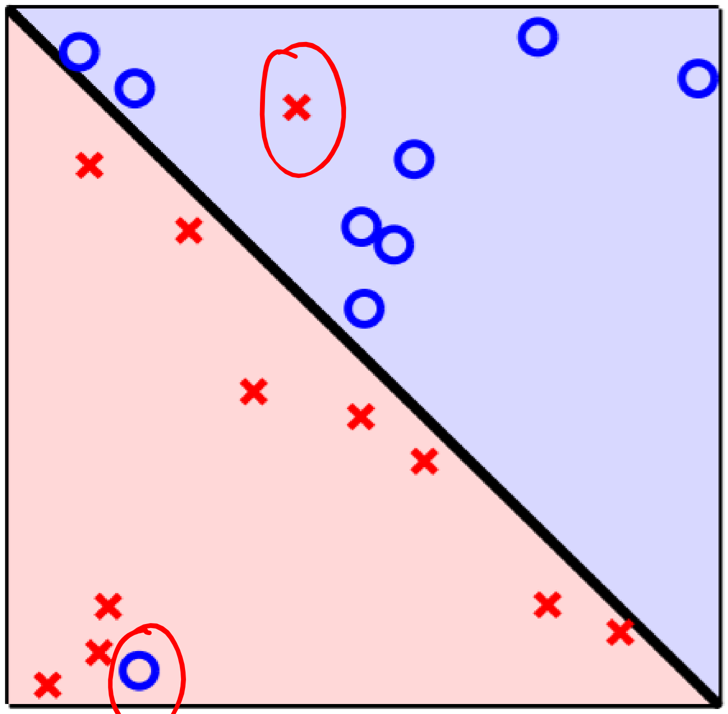


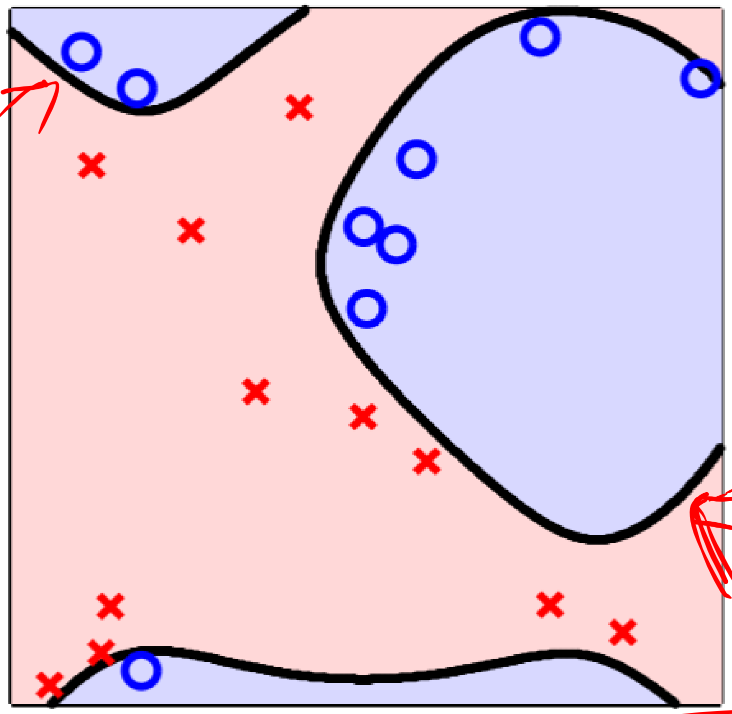
Be Careful with Feature Transforms

linear

4-th order polynomial

Data → $Q=4$





$E_{in} \simeq 0$

Overfitting

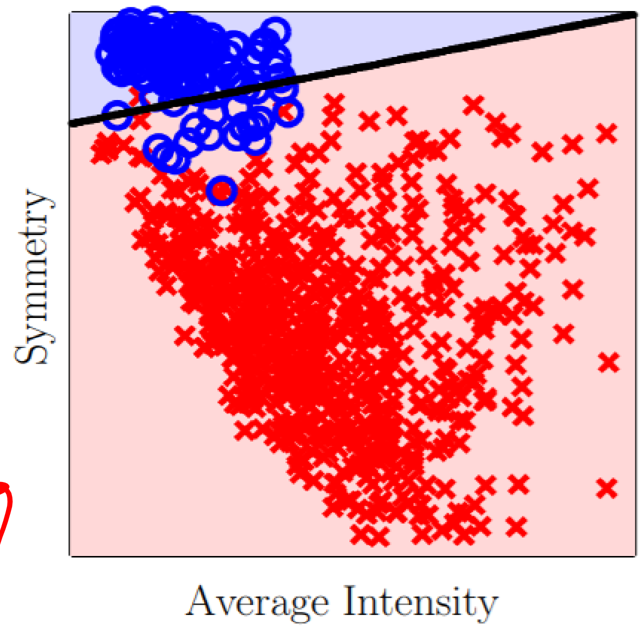
$\boxed{E_{in} \simeq E_{out}}$

Digits Data Again

(feature Construction)

fits better

1%

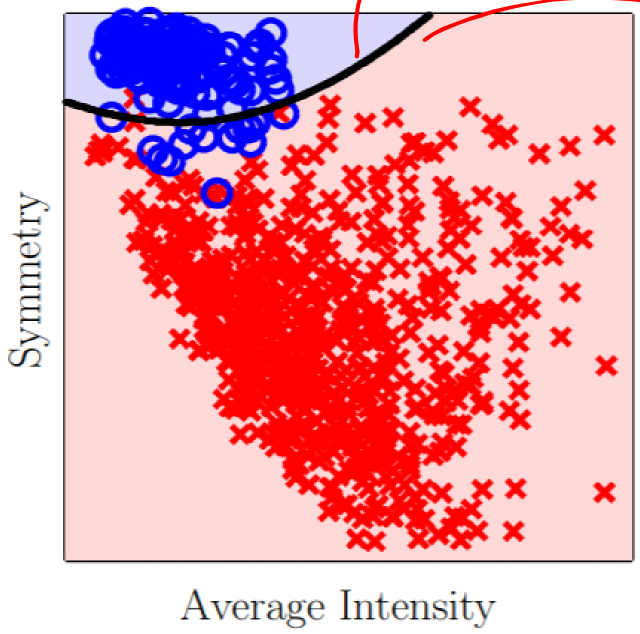


$E_{in} \sim E_{out}$

Linear model

$E_{\text{in}} = 2.13\%$

$E_{\text{out}} = 2.38\%$



3rd order polynomial model

$E_{\text{in}} = 1.75\%$

$E_{\text{out}} = 1.87\%$

N is large

Use the Linear Model!

Most of the cases.

- First try a linear model – simple, robust and works.

Pocket

- Algorithms can tolerate error plus you have nonlinear feature transforms.
- Choose a feature transform $\boxed{\textit{before}}$ seeing the data. Stay simple.

  Data snooping is hazardous to your E_{out}.
- Linear models are fundamental in their own right; they are also the building blocks of many more complex models like support vector machines. (SVM)
- Nonlinear transforms also apply to regression and logistic regression.

Thanks!