Machine Learning from Data

Lecture 16: Spring 2021

Today’s Lecture

- Similarity
- Nearest Neighbor

FOUNDATIONS

Theory.

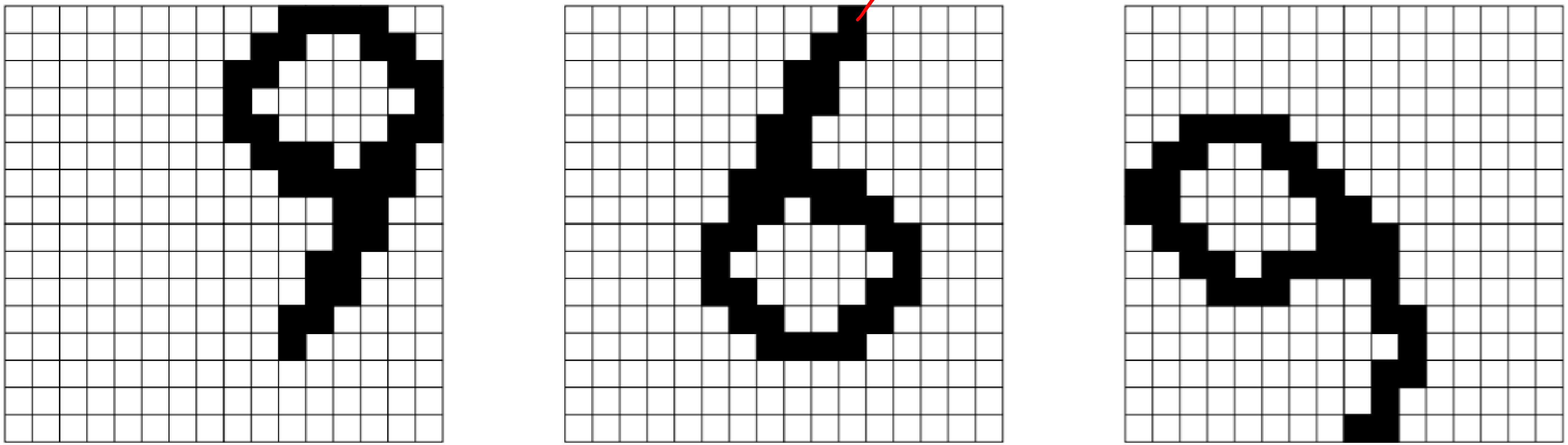
framework

Ask a 5-year-old what is this?

Measuring Similarity

NN rule

9 6 counting × 9



Capture essense of features w.r.t. the problem

Created / Constructed

$x \qquad x'$

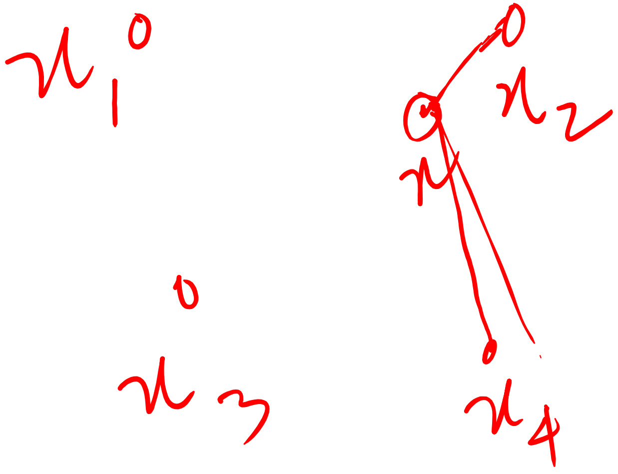
$$d(x, x') = ||x - x'||$$

Euclidean

Similarity measure

Test point $\rightarrow x$

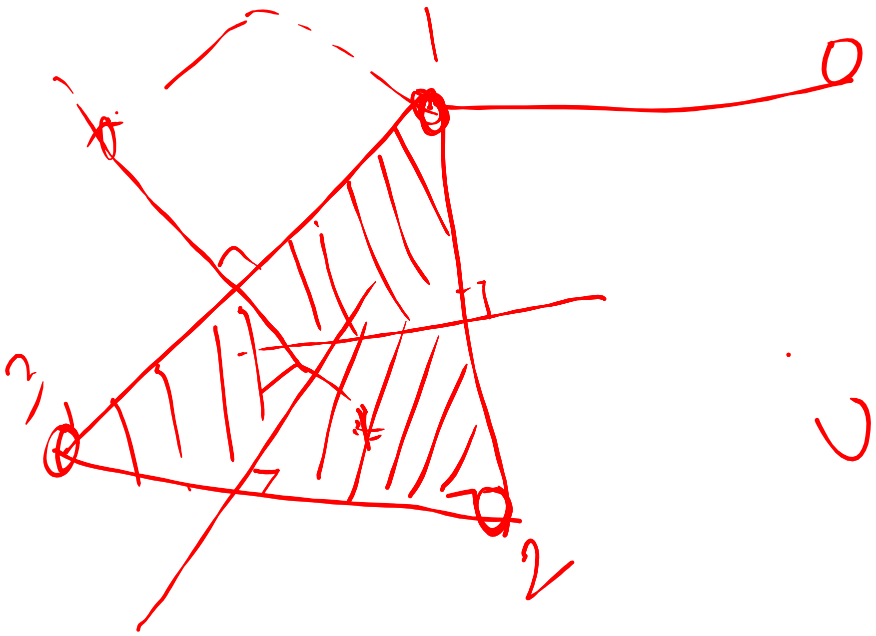
NN- Algorithm



$x_{[2]} \quad x_{[4]}$

$x_{[1]} \quad x_{[3]}$

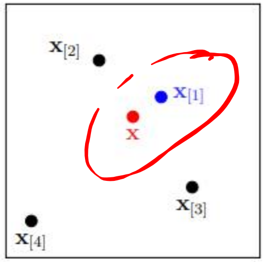
$$g(x) = \text{sign}[y_{[2]}]$$



Voronoi

Nearest Neighbor

Test 'x' is classified using its nearest neighbor.



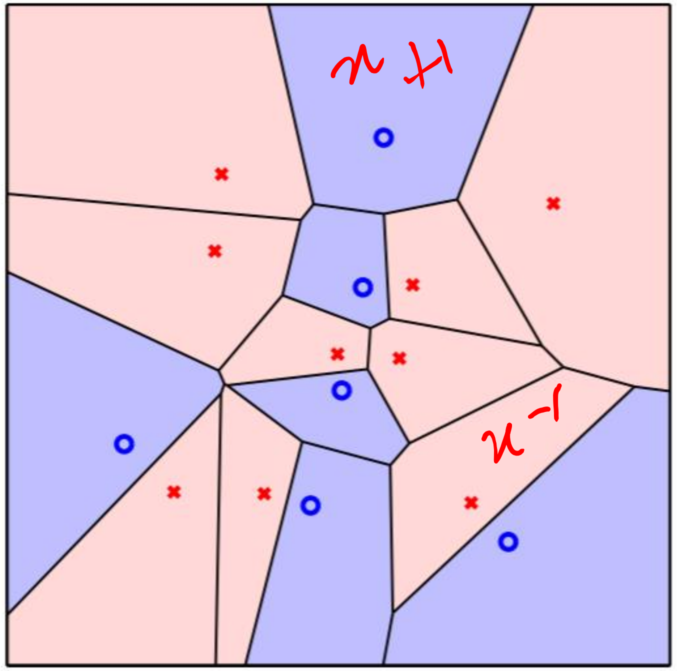
$$d(\mathbf{x}, \mathbf{x}_{[1]}) \leq d(\mathbf{x}, \mathbf{x}_{[2]}) \leq \cdots \leq d(\mathbf{x}, \mathbf{x}_{[N]})$$

$$g(\mathbf{x}) = y_{[1]}(\mathbf{x})$$

No training needed!

E_{in} vs E_{out}

$$E_{\text{in}} = 0$$



Nearest neighbor Voronoi tesselation

$\boxed{d_{vc} = \infty \qquad E_{in} = 0}$ $\quad E_{in}$ and E_{out}

$\rightarrow$ 5-year old.

Theorem : $\boxed{E_{out} \le 2 E^*_{out}}$ $\left\{ \begin{array}{l} \text{w.h.p} \\ \text{\& sufficiently} \\ \text{large } N \end{array} \right.$

$E^*_{out} \rightarrow$ Optimal $E_{out} \rightarrow$ The best possible out-of-sample error for a given problem.

How small can E_{out} get?

Proof : $f(x) \Rightarrow \underline{\pi(x)} = P(y=+1 \mid x)$

Optimal classifier:

$$\begin{cases} +1 & \text{if } \pi(x) \geq \frac{1}{2} \\ -1 & \text{if } \pi(x) < \frac{1}{2} \end{cases}$$

$$E^*_{out}(x) = \begin{cases} 1 - \pi(x), & \pi(x) \geq 1/2 \\ \pi(x), & \pi(x) < \frac{1}{2} \end{cases}$$

$$E^*_{out}(x) = \min\left(\pi(x), 1 - \pi(x)\right) = \eta(x)$$

$E_{out}^*(x)$

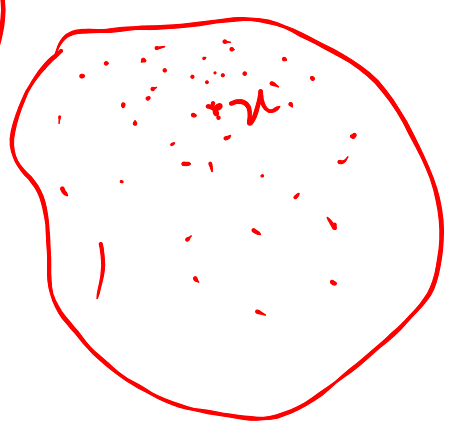
$= \int dx \, P(x) E^*_{out}(x)$

$\boxed{\text{Assumptions}}$

1) $\pi(x)$ is continuos

2) Data generation

$x_{[1]}$ $\quad N \to \infty \Rightarrow \underline{\underline{x_{[1]}}} \to x$



$$\text{Prob Error}[x] = P\left[y_{[1]} \neq y\right]$$

$y_{[1]}$ <

$$= P\left[y_{[1]} = +1, y = -1\right] + P\left[y_{[1]} = -1, y = +1\right]$$

$$= \pi(x_{[1]})\left(1 - \pi(x)\right) + \left(1 - \pi(x_{[1]})\right)\pi(x)$$

N is sufficiently large:

$x_{[1]} \longrightarrow x$

$\pi(x_{[1]}) \longrightarrow \pi(x)$

$$\text{ProbError}[x] = \pi(x)(1-\pi(x)) + (1-\pi(x))\pi(x)$$

$$= 2\,\underbrace{\pi(x)(1-\pi(x))}$$

$$\leq 2\,\eta(x) = 2E^*_{out}(x)$$

$$E_{out}(x) \leq 2E^*_{out}(x)$$

Proving $E_{\text{out}} \leq 2E^*_{\text{out}}$

$$\pi(\mathbf{x}) = \mathbb{P}[y = +1|\mathbf{x}].$$ ← the target in logistic regression

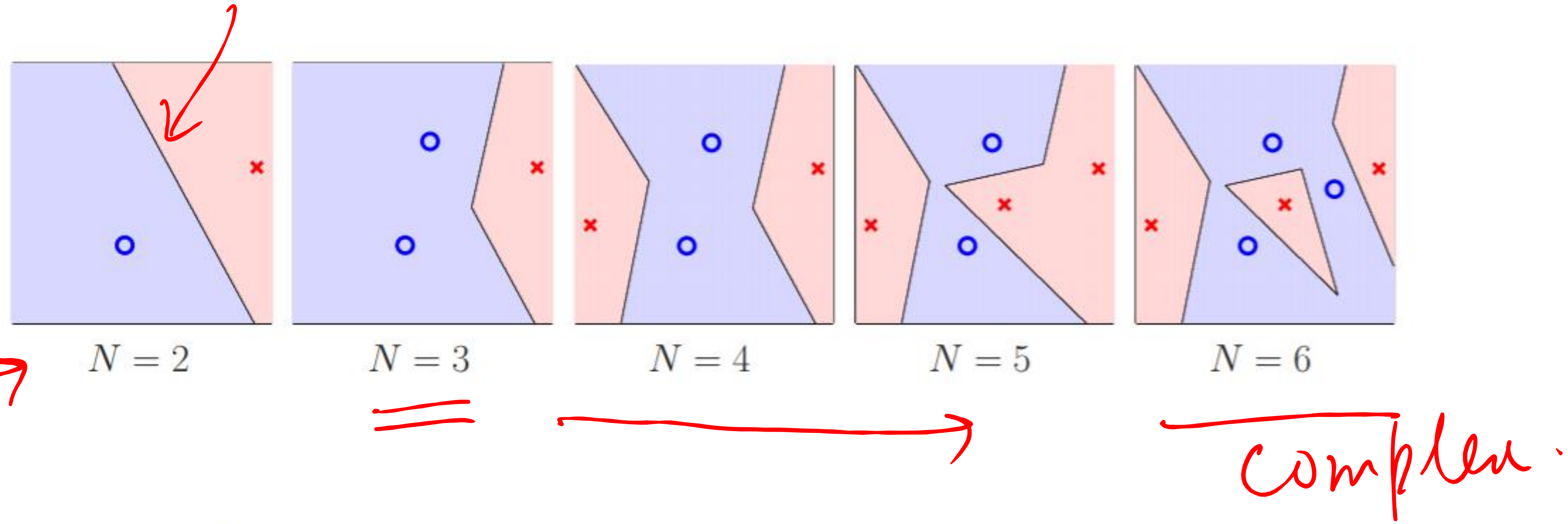
Assume $\pi(\mathbf{x})$ is continuous and $\mathbf{x}_{[1]} \xrightarrow{N\to\infty} \mathbf{x}$. Then $\pi(\mathbf{x}_{[1]}) \xrightarrow{N\to\infty} \pi(\mathbf{x})$.

$$\begin{aligned}
\mathbb{P}[g_N(\mathbf{x}) \neq y] &= \mathbb{P}[y = +1, y_{[1]} = -1] + \mathbb{P}[y = -1, y_{[1]} = +1], \\
&= \pi(\mathbf{x}) \cdot (1 - \pi(\mathbf{x}_{[1]})) + (1 - \pi(\mathbf{x})) \cdot \pi(\mathbf{x}_{[1]}), \\
&\to \pi(\mathbf{x}) \cdot (1 - \pi(\mathbf{x})) + (1 - \pi(\mathbf{x})) \cdot \pi(\mathbf{x}), \\
&= 2\pi(\mathbf{x}) \cdot (1 - \pi(\mathbf{x})), \\
&\boxed{\leq 2\min\{\pi(\mathbf{x}), 1 - \pi(\mathbf{x})\}.}
\end{aligned}$$

The best you can do is

$$E^*_{\text{out}}(\mathbf{x}) = \min\{\pi(\mathbf{x}), 1 - \pi(\mathbf{x})\}.$$

Nearest Neighbor 'Self-Regularizes'



A simple boundary is used with few data points.

A more complicated boundary is possible *only* when you have more data points.

regularization guides you to simpler hypotheses when data quality/quantity is lower.

k-Nearest Neighbor

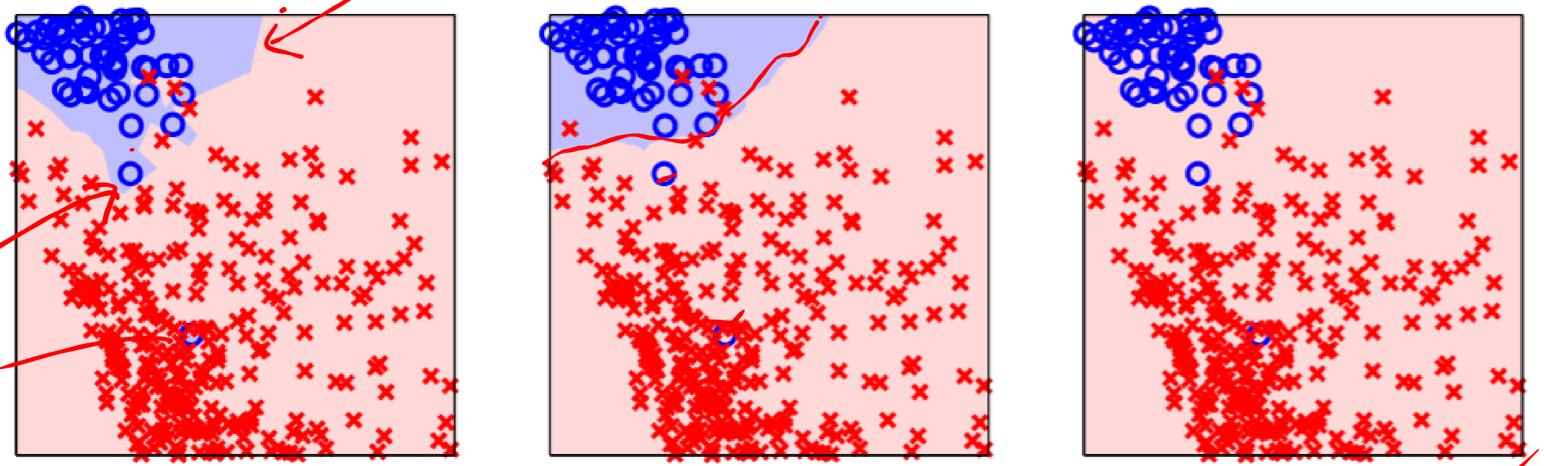
Majority

$$g(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{k} y_{[i]}(\mathbf{x})\right).$$

k-regularization

(k is odd and $y_n = \pm 1$).

$E_{in}=0$



1-NN rule 21-NN rule 127-NN rule

overfit Underfit

choose k

↳ CV

The Role of k

k determines the tradeoff between fitting the data and overfitting the data.

Theorem. For $N \to \infty$, if $k(N) \to \infty$ and $k(N)/N \to 0$ then,

$$E_{\text{in}}(g) \to E_{\text{out}}(g) \qquad \text{and} \qquad E_{\text{out}}(g) \to E^*_{\text{out}}.$$

convergy.

For example $k = \lceil \sqrt{N} \rceil$.

$\boxed{K=3}$

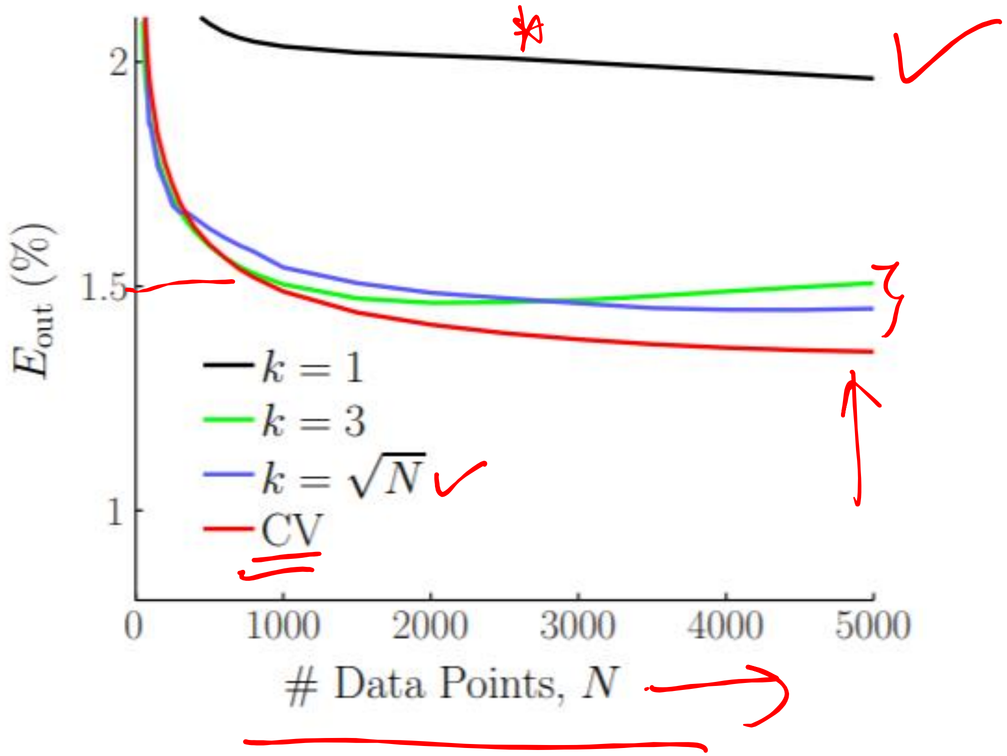
3 Ways To Choose k

1. $k = 3$.

2. $k = \left\lceil \sqrt{N} \right\rceil$.

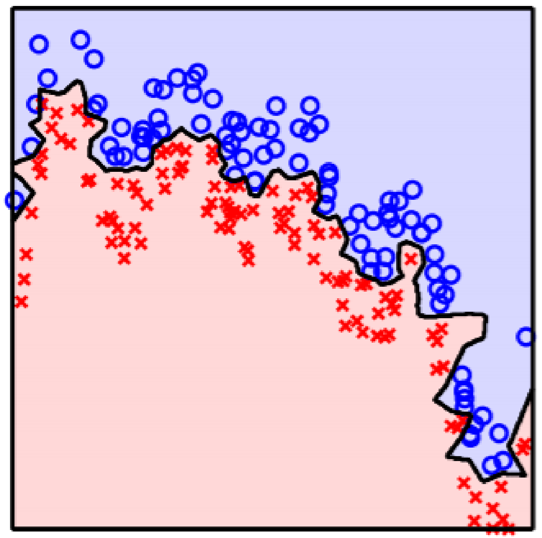
3. Validation or cross validation:

   k-NN rule hypotheses g_k^- constructed on training set, tested on validation set, and best k is picked.



Nearest Neighbor is Nonparametric

NN-rule



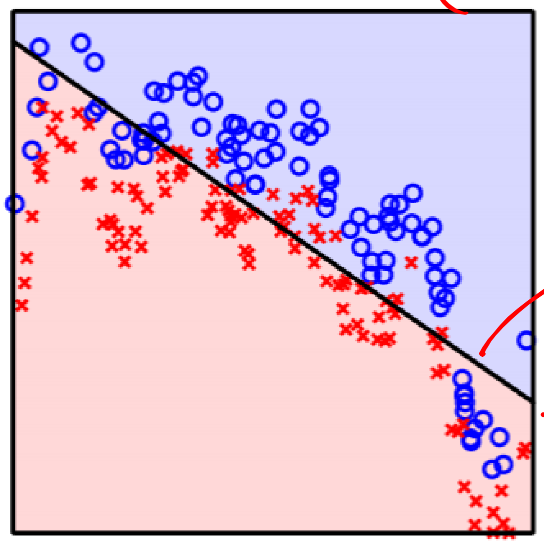
no parameters

expressive/flexible

$g(\mathbf{x})$ needs data

generic, can model anything

Linear Model (Parametric)



line

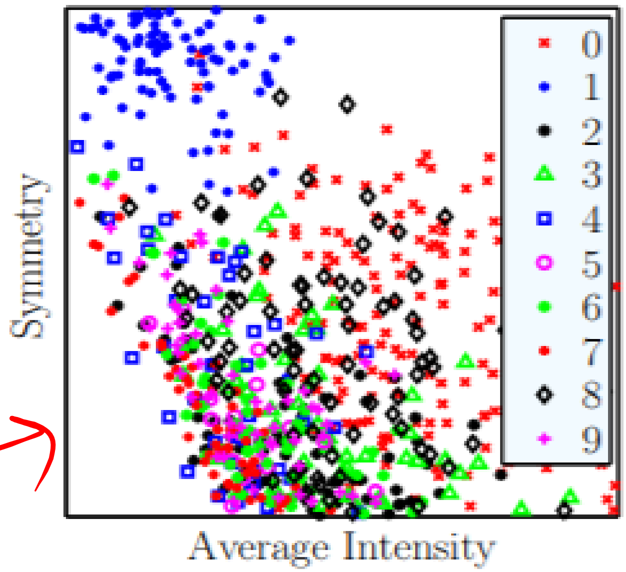
$(d+1)$ parameters

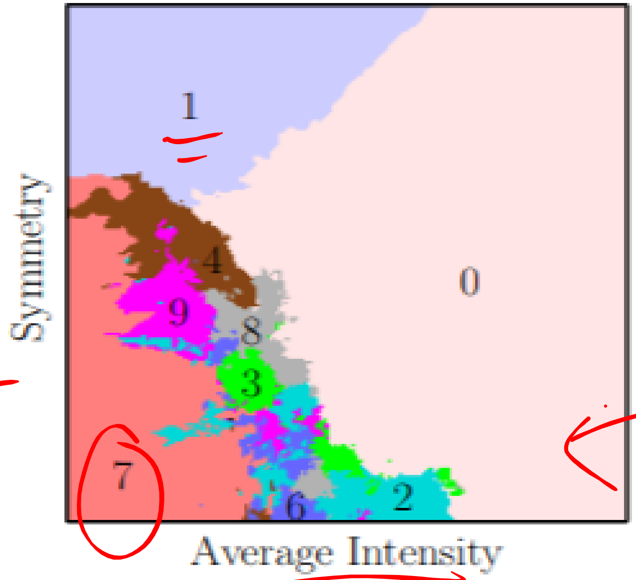
rigid, always linear

$g(\mathbf{x})$ needs only weights ← keep

specialized

Nearest Neighbor Easily Extends to Multiclass





Confusion Matrix

True	Predicted 0	1	2	3	4	5	6	7	8	9	
0	**13.5**	0.5	0.5	1	0	0.5	0	0	0.5	0	16.5
1	0.5	**13.5**	0	0	0	0	0	0	0	0	14
2	0.5	0	**3.5**	1	1	1.5	1	1	0	0.5	10
3	2.5	0	1.5	**2**	0.5	0.5	0.5	0.5	0.5	1	9.5
4	0.5	0	1	0.5	**1.5**	0.5	1	2	0	1.5	8.5
5	0.5	0	2.5	1	0.5	**1.5**	1	1	0	0.5	7.5
6	0.5	0	2	1	1	1	**1**	1	0	1	8.5
7	0	0	1.5	0.5	1.5	0.5	1	**3**	0	1	9
8	3.5	0	0.5	1	0.5	0.5	0.5	0	**0.5**	1	8
9	0.5	0	1	1	1	0.5	1	1	0.5	**2**	8.5
	22.5	14	14	9	7.5	7	7	9.5	2	8.5	100

41% accuracy!

Highlights of k-Nearest Neighbor

1. Simple.
2. No training.
3. Near optimal E_{out}. (N is sufficiently large)
4. Easy to justify classification to customer.

A **good!** method

5. Can easily do multi-class.
6. Can easily adapt to regression or logistic regression

$$g(\mathbf{x}) = \frac{1}{k}\sum_{i=1}^{k} y_{[i]}(\mathbf{x}) \qquad\qquad g(\mathbf{x}) = \frac{1}{k}\sum_{i=1}^{k} [\![y_{[i]}(\mathbf{x}) = +1]\!]$$

7. **Computationally demanding**. ⟵ we will address this next

Thanks!