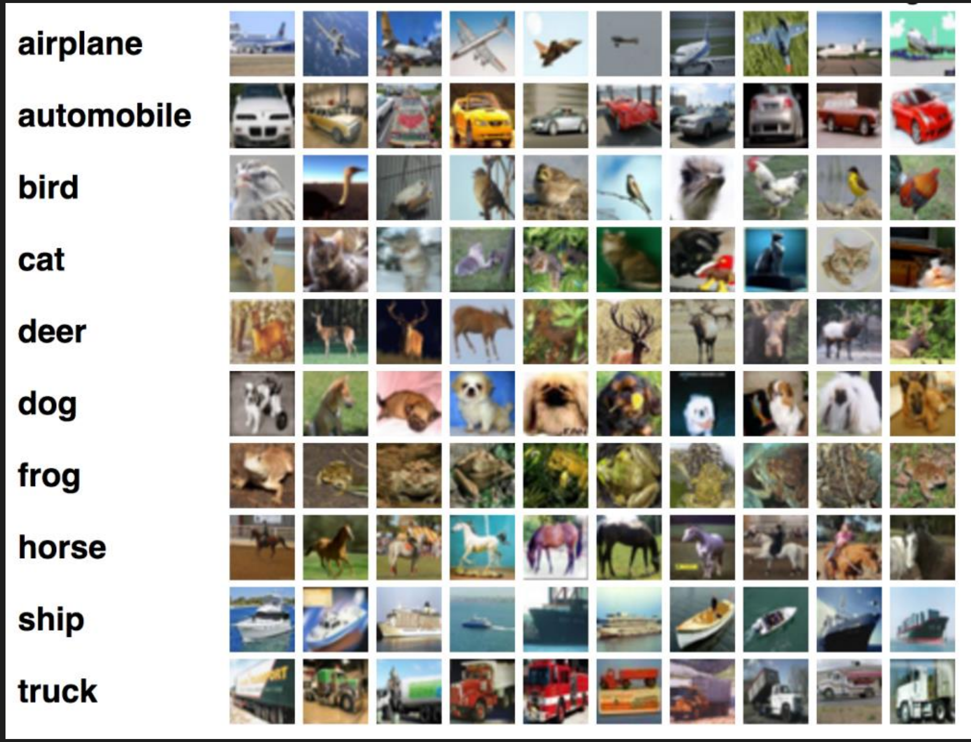
INTRODUCTION TO MACHINE LEARNING COMPSCI 4ML3

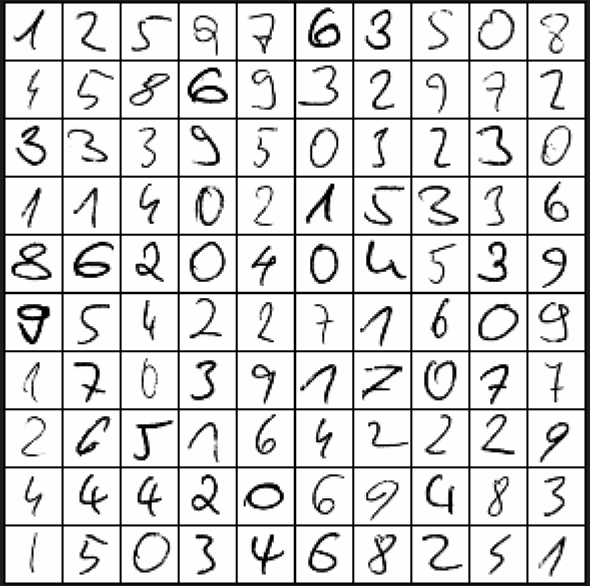
LECTURE 13

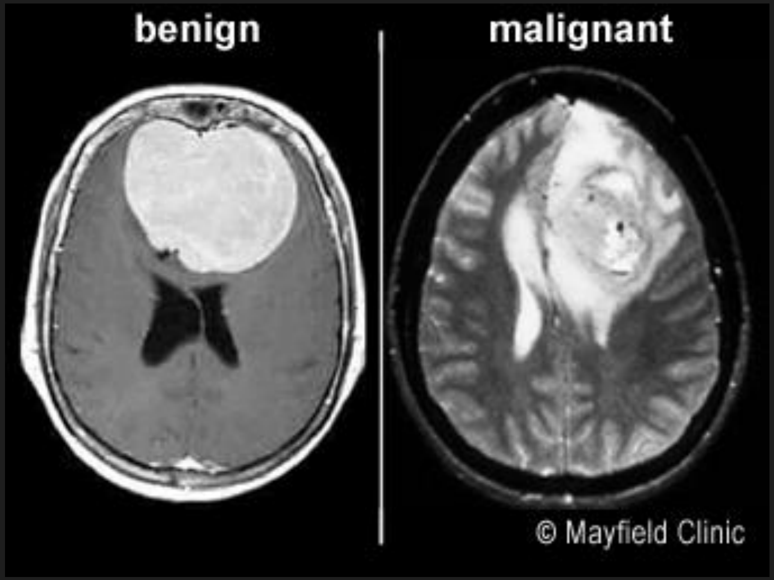
HASSAN ASHTIANI

CLASSIFICATION

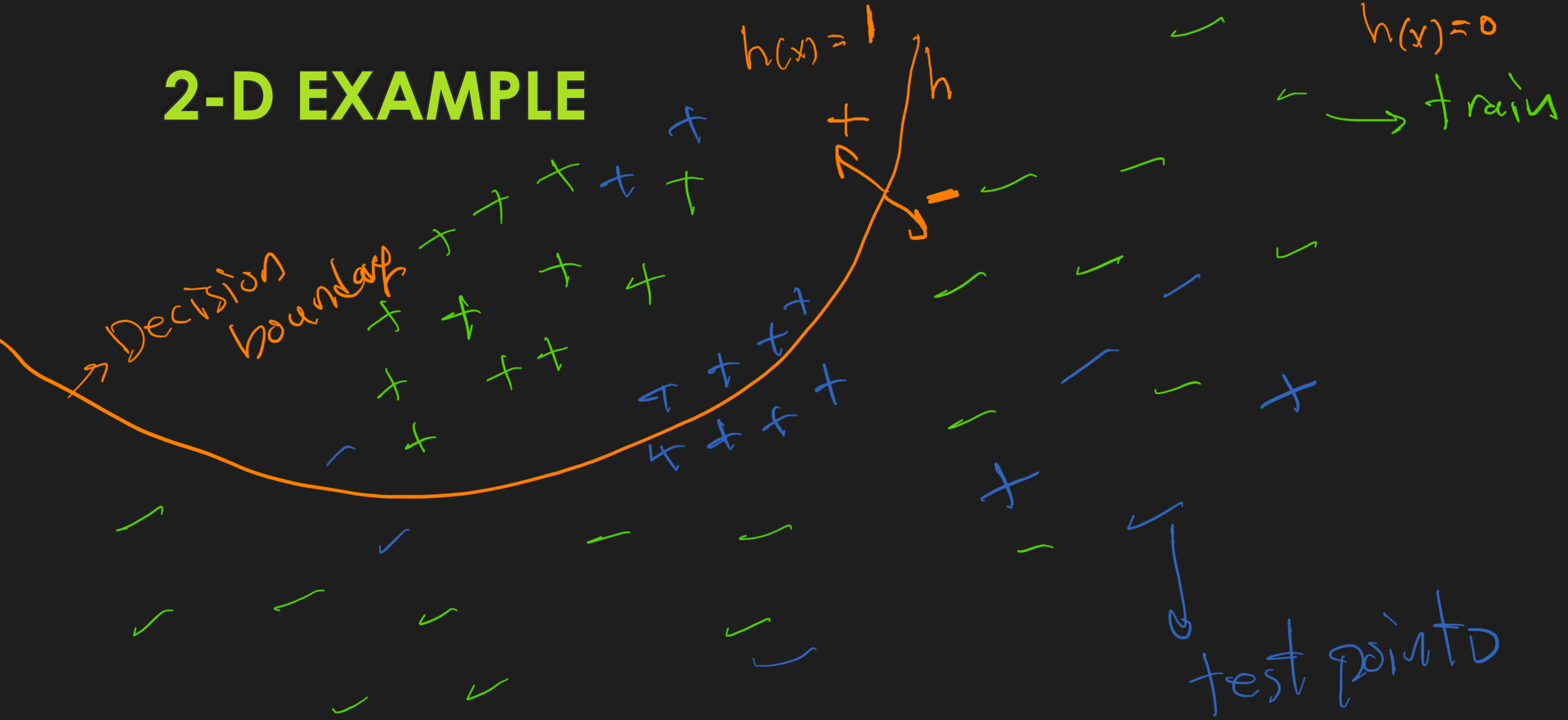
- PREDICT THE CATEGORY
- k-*CLASS CLASSIFICATION*
  - $y \in \{1,2,3,\ldots,k\}$







2-D EXAMPLE



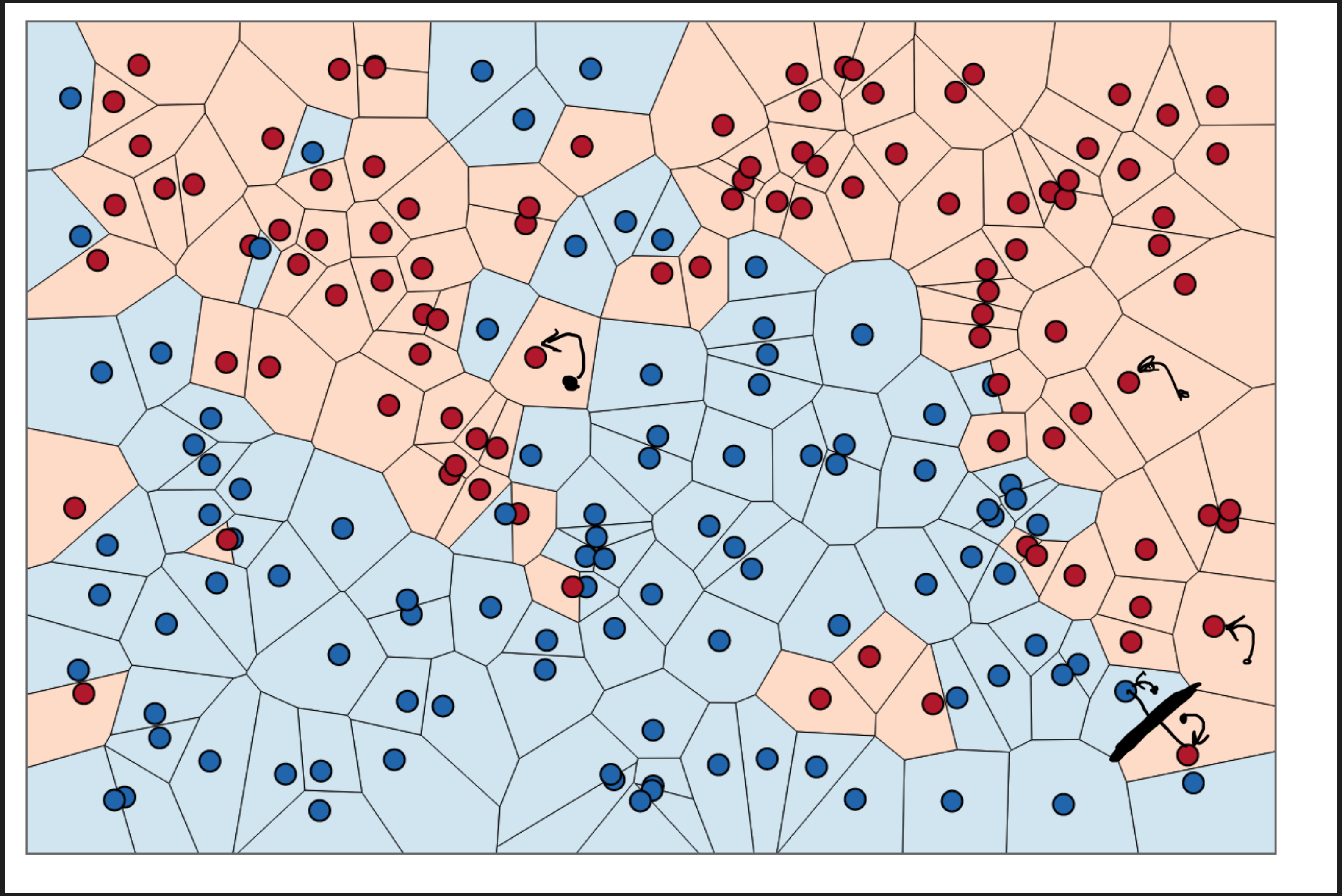
ACCURACY, 0-1 LOSS

- For regression we used the squared loss
  - $l(y, \hat{y}) = (y - \hat{y})^2$.
- Loss for classification?
  - The zero-one loss (error): $l^{0-1}(y, \hat{y}) = 1_{y \neq \hat{y}} = \begin{cases} 1 & y \neq \hat{y} \\ 0 & y = \hat{y} \end{cases}$
- Predictor/labeling-function: $h: X \rightarrow Y$
- Training set: $Z = ((x^1, y^1), \dots, (x^m, y^m))$
  - generated from D_{XY}, or D for short
- Empirical (training) error of h: $L_Z^{0-1}(h) = \frac{1}{m} \sum_i l^{0-1}(h(x^i), y^i)$
- Expected error of h: $L_D^{0-1} = \mathop{E}_{(x,y) \sim D} l^{0-1}(h(x), y)$

The Nearest Neighbor Classifier

- $\hat{y}(x; Z) =$
  - Find the closest x' to x in the data set
    - $\min_{x'} \|x - x'\|_2$
  - Output the label of x'
- Decision boundary?

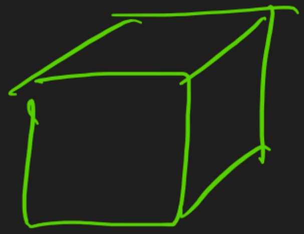
VORONOI DIAGRAM



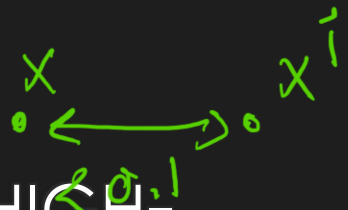
NEAREST NEIGHBOR: PROS AND CONST

- Pros
  - Basically no training is needed.
  - No parameter or hyper-parameter
  - Easy to implement
  - Powerful and flexible
    - non-parametric!
- Cons
  - High test-time computational complexity, memory intensive
  - Curse-of-dimensionality!
- Can we use nearest neighbor for regression?

CURSE OF DIMENSIONALITY FOR NN



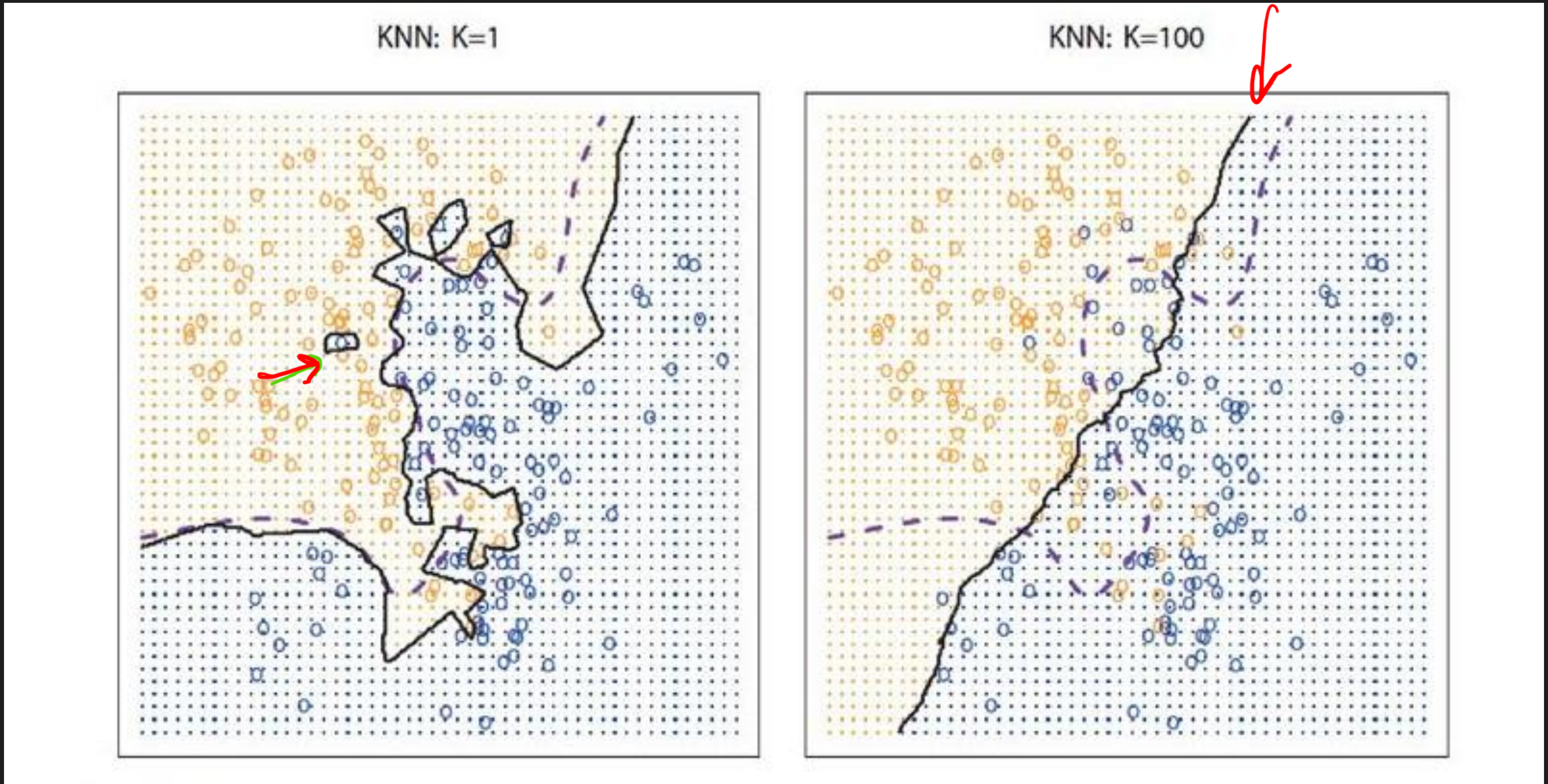
- ASSUME POINTS ARE IN THE d-DIMENSIONAL UNIT CUBE
  - HOW MANY TRAINING POINTS DO I NEED TO "COVER" THIS CUBE?
    - E.G., FOR ANY $x \in [0,1]^d$, I WANT TO HAVE AT LEAST ONE TRAINING POINT x^i SUCH THAT $\|x - x^i\|_2 < 0.1$
  - WE NEED 10^d TRAINING POINTS!



- NEAREST NEIGHBOR DOES NOT WORK WELL FOR HIGH-DIMENSIONAL DATA

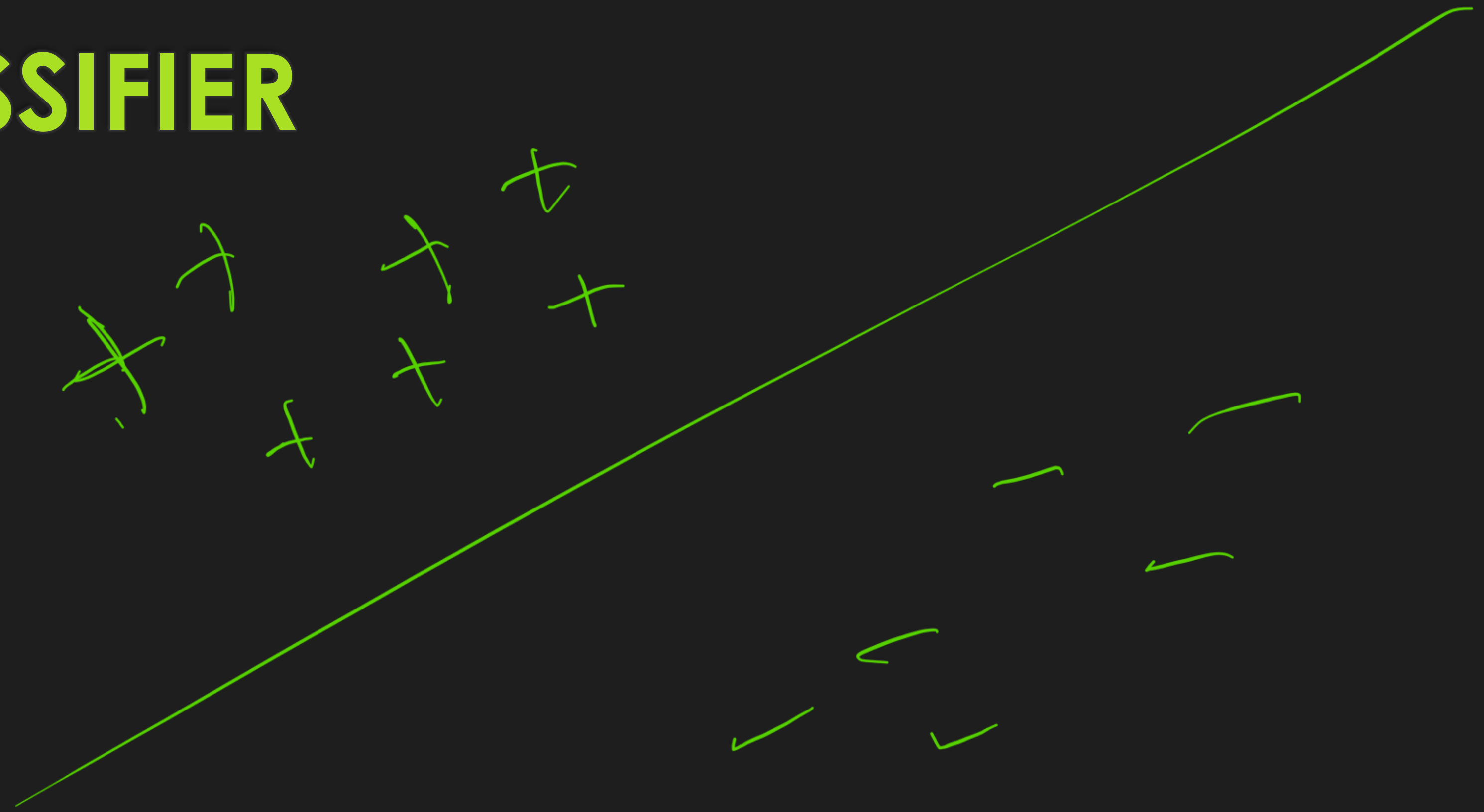
K-NEAREST NEIGHBOR

- $\hat{y}(x; Z) =$
  - FIND THE k CLOSEST POINTS TO x IN THE DATA SET
  - LET $y^1, y^2, \ldots, y^k$ BE THE LABELS OF THESE NEIGHBOR POINTS
  - OUTPUT THE **MAJORITY VOTE** AMONG y^iS



- MORE ROBUST, SMOOTHER DECISION BOUNDARY, HIGHER-TRAINING ERROR BUT LESS PRONE TO OVERFITTING
  - STILL CURSE OF DIMENSIONALITY

LINEAR CLASSIFIER



- $\hat{y}_W(x) = sign(W^T x) = 1_{W^T x \geq 0}$
- HOW TO FIND W GIVEN DATA SET Z?

$Y = \{0, 1\}$

$Y = \{-1, +1\}$

LINEAR CLASSIFIERS

- Empirical Risk Minimization
  - $\widehat{W} = arg\min_{w} L_Z^{0-1}(\hat{y}_w)$
- Computational complexity for $d = 2$?

  - Higher d?

HARDNESS OF LINEAR CLASSIFICATION

- In general, finding the linear separator with minimum classification error is NP-hard (with respect to d)
- Unless….
  - Data is linearly separable!
  - ..or optimize another loss function instead!
- Can't we just go into higher dimensions to make the data linearly separable?

LINEARLY SEPARABLE DATA

SURROGATE LOSS FUNCTIONS

- So far, we assumed the classifier returns a discrete value
  - E.g., $\hat{y}_w = sign(W^T x) \in \{0,1\}$
- What if the classifier's output is continuous
  - E.g., $\hat{y}_w = W^T x$
  - $\hat{y}$ will capture the "confidence" of the classifier too
- Other (continuous) loss functions
  - Margin loss, cross-entropy/negative-log-likelihood loss, …
- Potential benefits?
  - Easier to optimize
  - Mitigate Overfitting