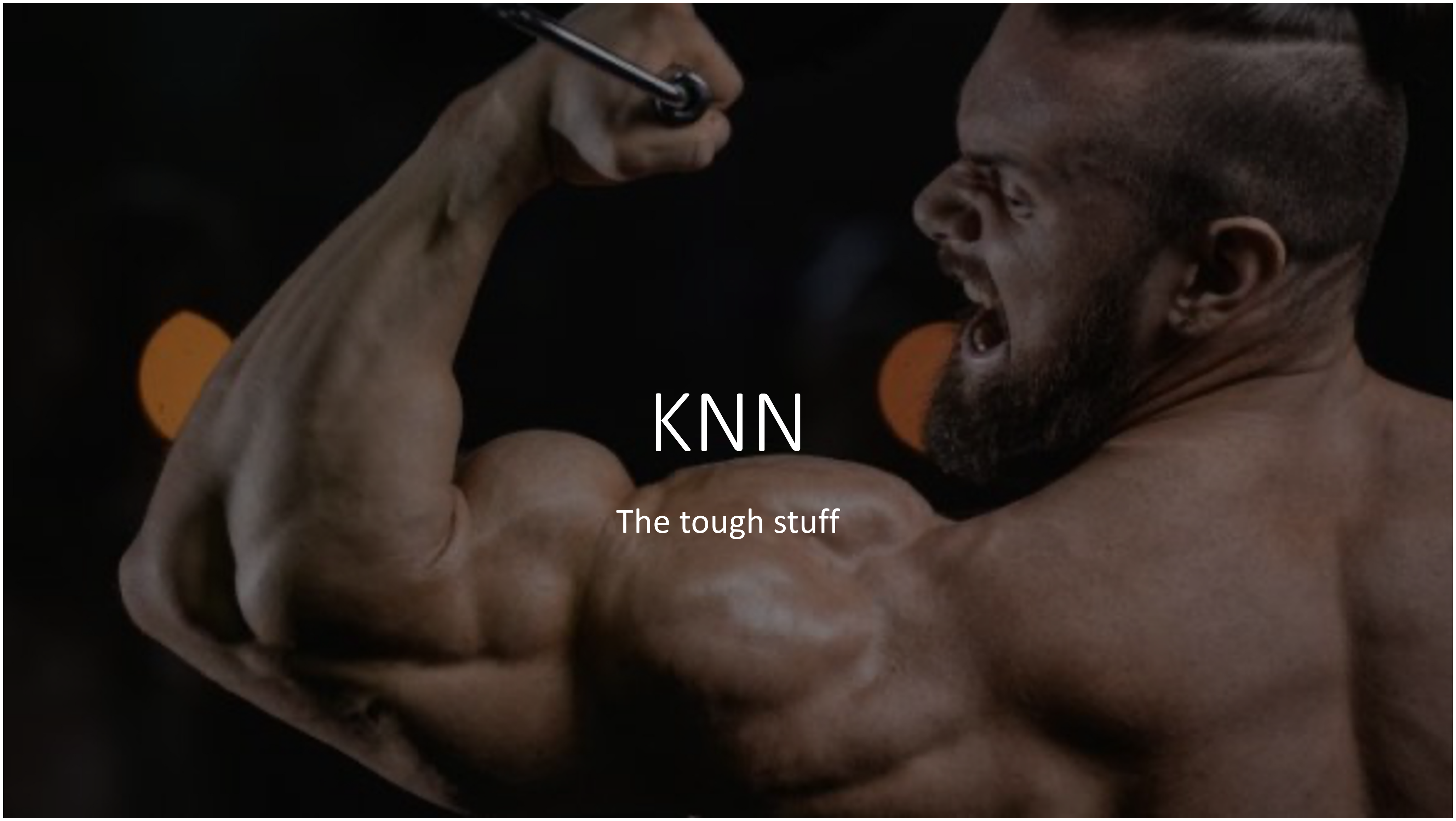
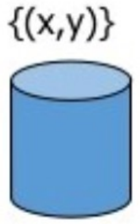


KNN

The tough stuff

Classification or Supervised Learning

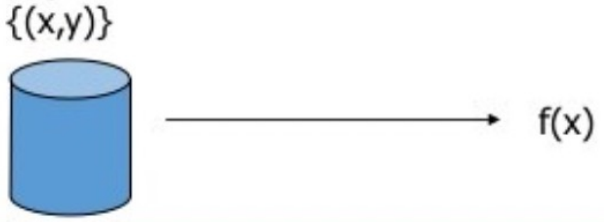
Supervised Learning:

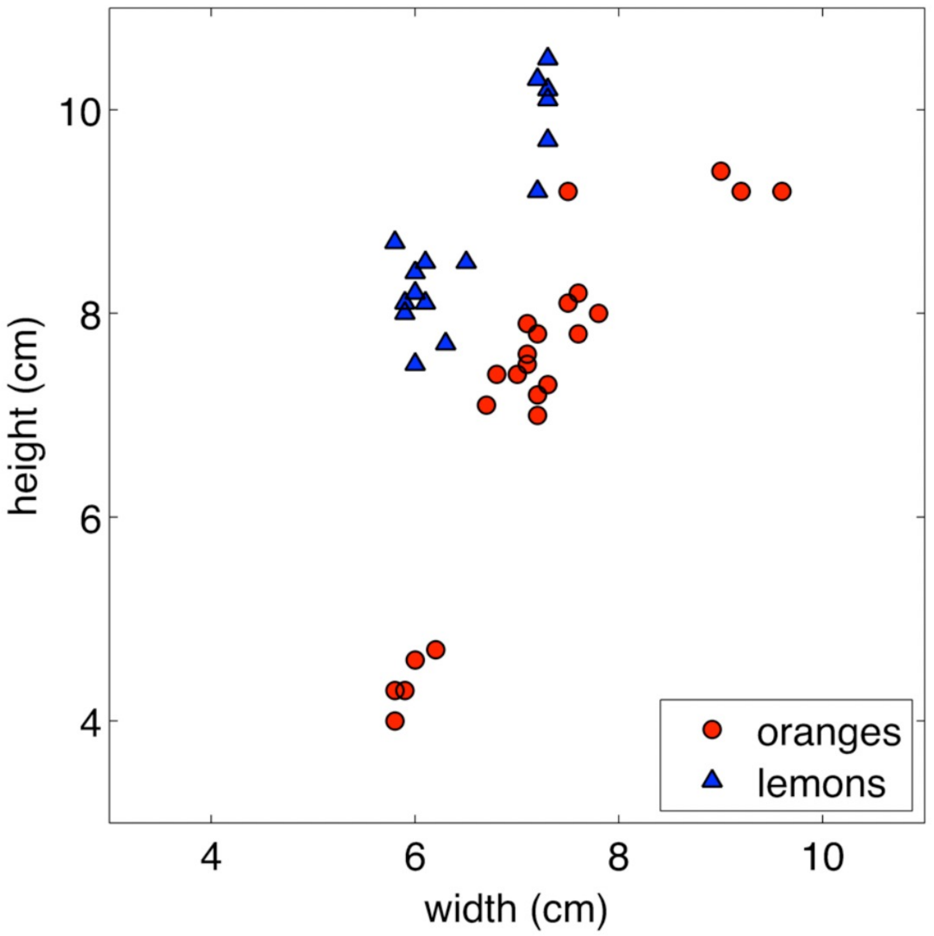


Training set **x** = $\{x_1, x_2, \ldots, x_N\}$

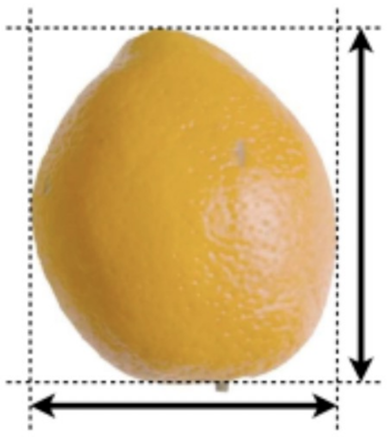
Class or target vector **y** = $\{y_1, y_2, \ldots, y_k\}$

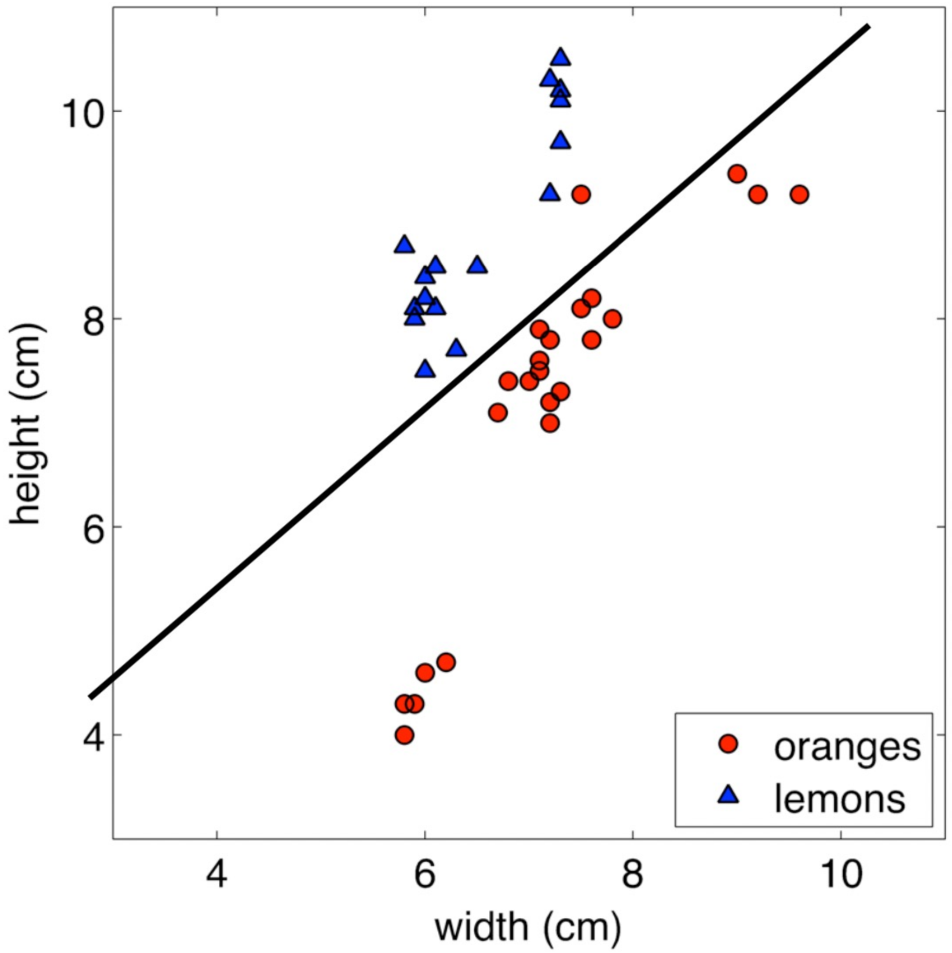
Find a function f(x) that takes a vector x and outputs a class y.





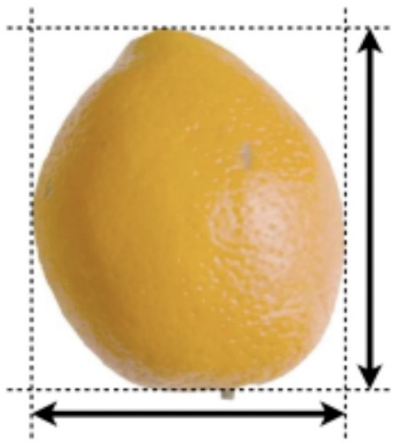
height (cm)
width (cm)
oranges
lemons
4
6
8
10





Can construct simple linear decision boundary:

$$y = \text{sign}(w_0 + w_1 x_1 + w_2 x_2)$$



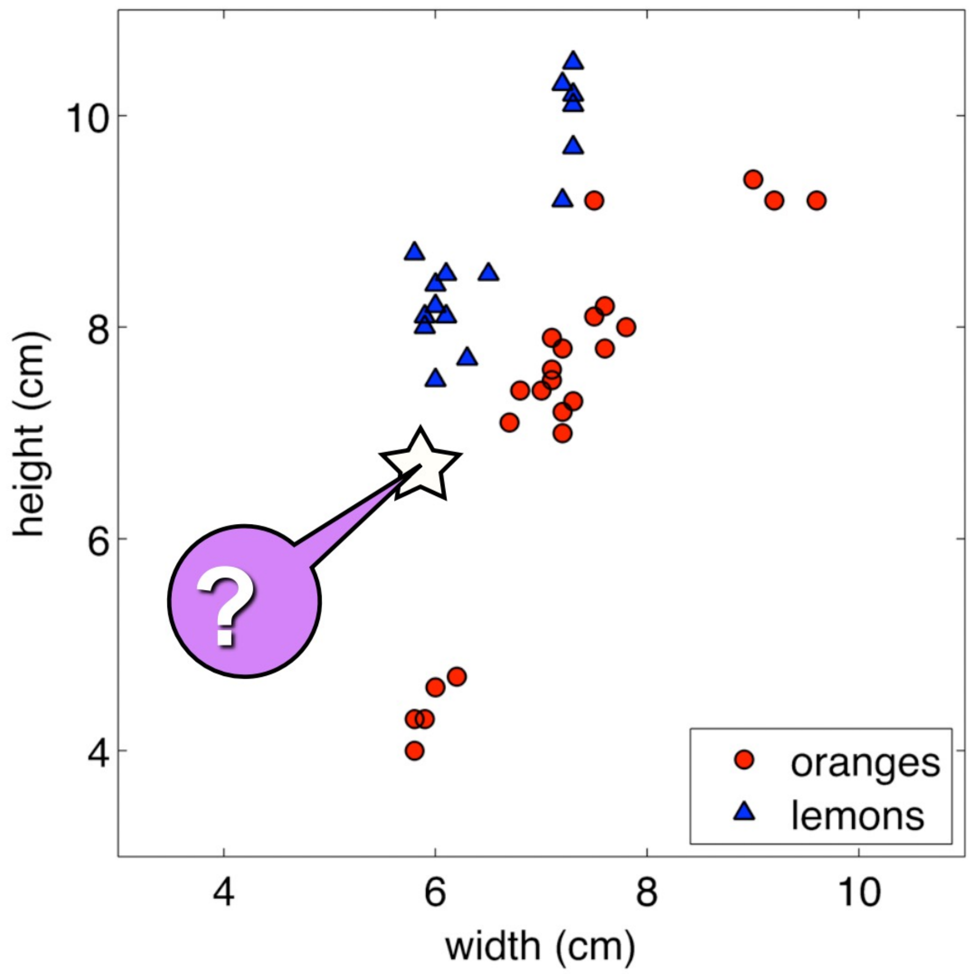
- Classification is intrinsically non-linear
  - It puts non-identical things in the same class, so a difference in the input vector sometimes causes zero change in the answer
- Linear classification means that the part that adapts is linear (just like linear regression)

  $$z(x) = \mathbf{w}^T \mathbf{x} + w_0$$

  with adaptive $\mathbf{w}, w_0$
- The adaptive part is followed by a non-linearity to make the decision

  $$y(\mathbf{x}) = f(z(\mathbf{x}))$$

- What functions $f()$ have we seen so far in class?


height (cm)
width (cm)
oranges
lemons
?

- Alternative to parametric models are non-parametric models
- These are typically simple methods for approximating discrete-valued or real-valued target functions (they work for classification or regression problems)
- Learning amounts to simply storing training data
- Test instances classified using similar training instances
- Embodies often sensible underlying assumptions:
  - Output varies smoothly with input
  - Data occupies sub-space of high-dimensional input space

Nearest Neighbors

- Training example in Euclidean space: $\mathbf{x} \in \Re^d$
- Idea: The value of the target function for a new query is estimated from the known value(s) of the nearest training example(s)
- Distance typically defined to be Euclidean:

$$||\mathbf{x}^{(a)} - \mathbf{x}^{(b)}||_2 = \sqrt{\sum_{j=1}^{d} (x_j^{(a)} - x_j^{(b)})^2}$$

Algorithm:

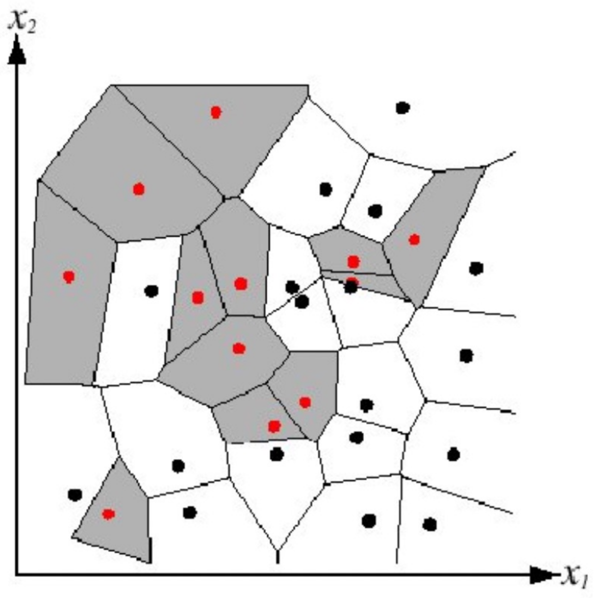
1. Find example $(\mathbf{x}^*, t^*)$ (from the stored training set) closest to the test instance $\mathbf{x}$. That is:

$$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{train. set}}{\text{argmin}} \ \text{distance}(\mathbf{x}^{(i)}, \mathbf{x})$$

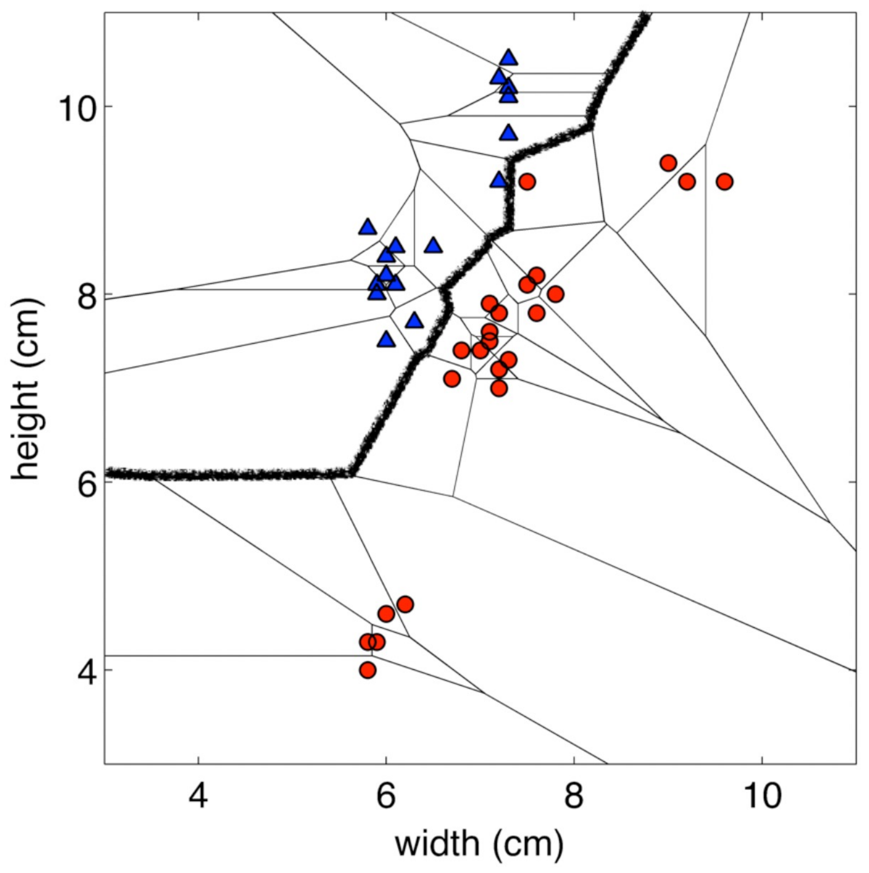
2. Output $y = t^*$

Nearest Neighbors: Decision Boundaries

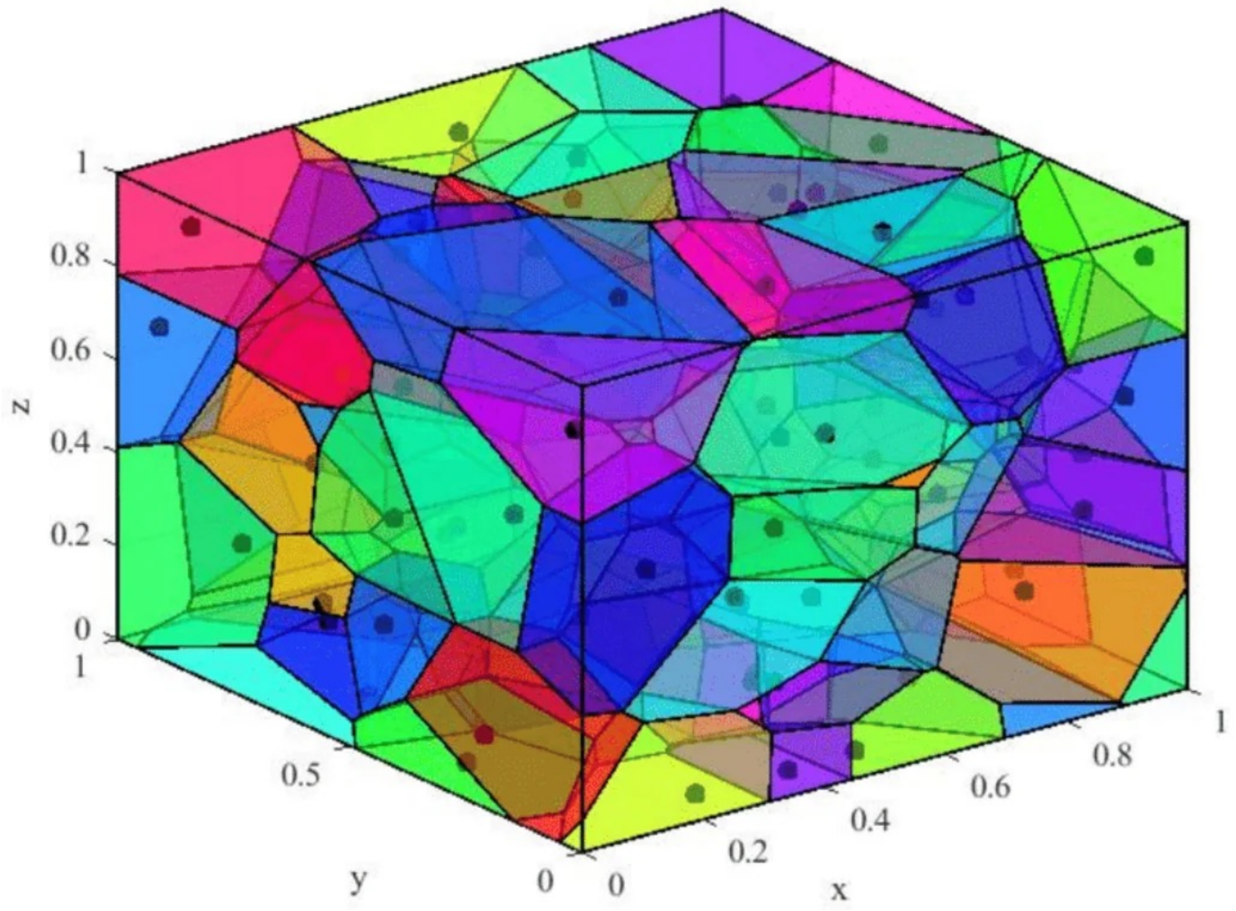
- Nearest neighbor algorithm does not explicitly compute decision boundaries, but these can be inferred
- Decision boundaries: Voronoi diagram visualization
  - show how input space divided into classes
  - each line segment is equidistant between two points of opposite classes



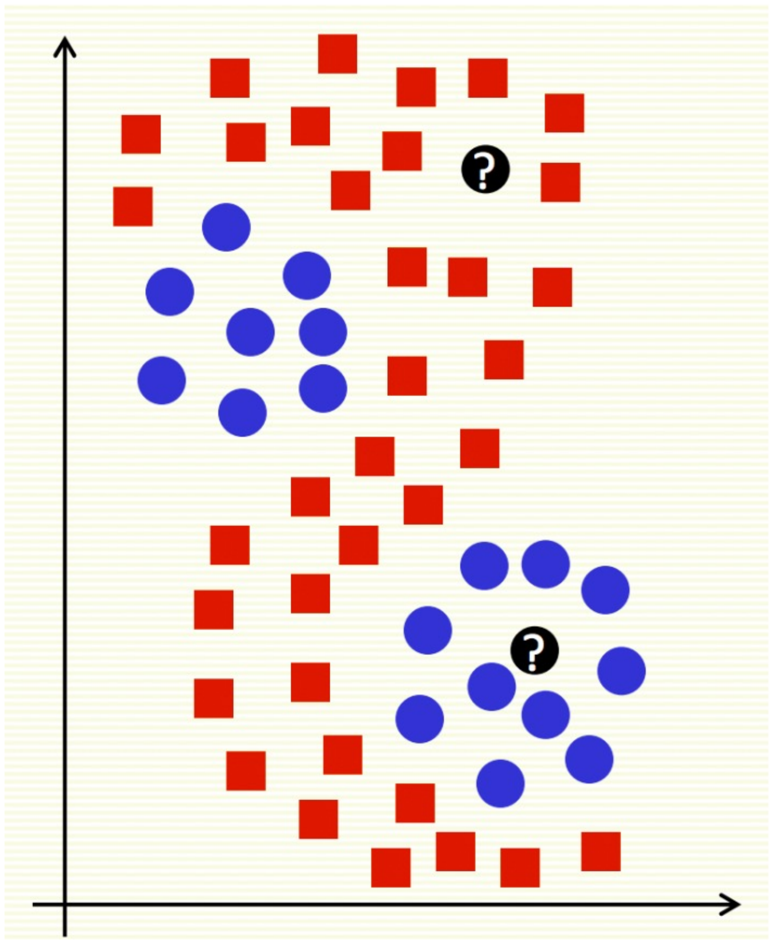
Decision Boundaries



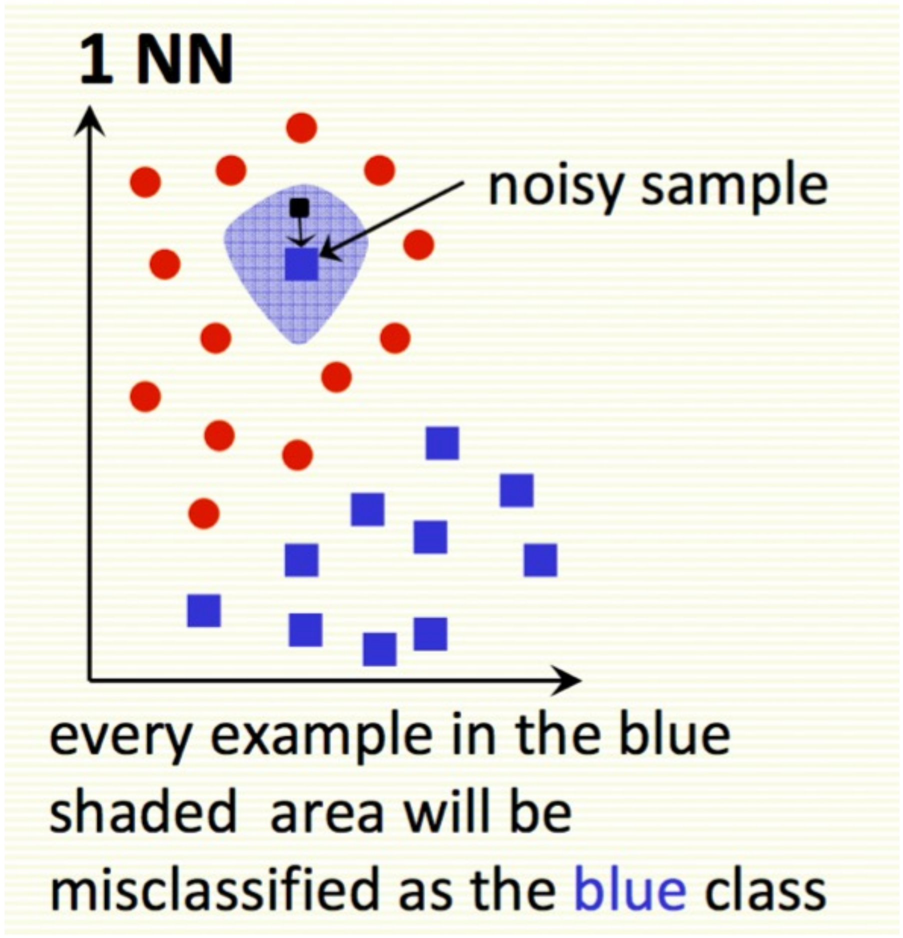
Decision Boundaries 3D



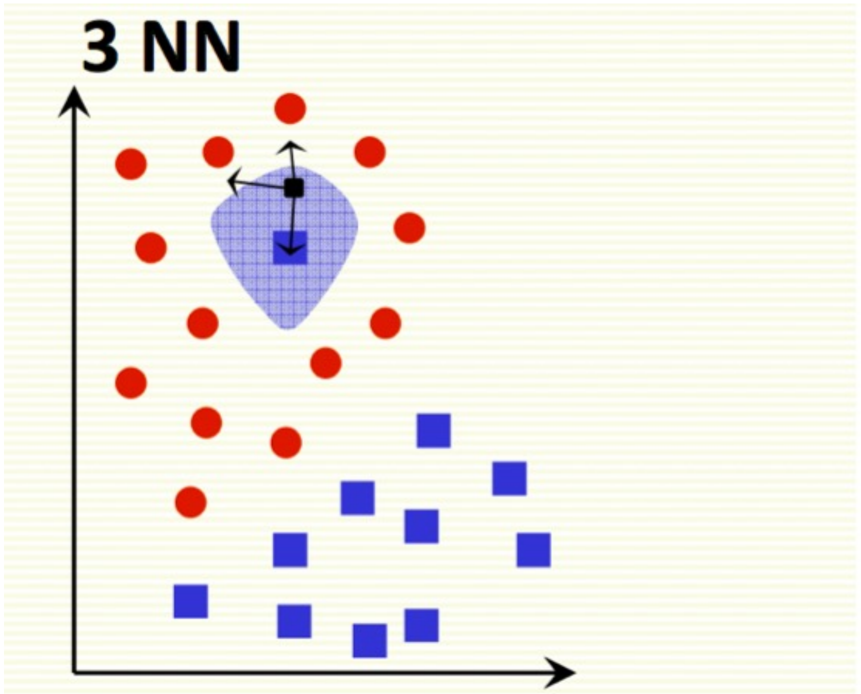
Multi-modal data



Nearest neighbor sensitive to mis-labeled data

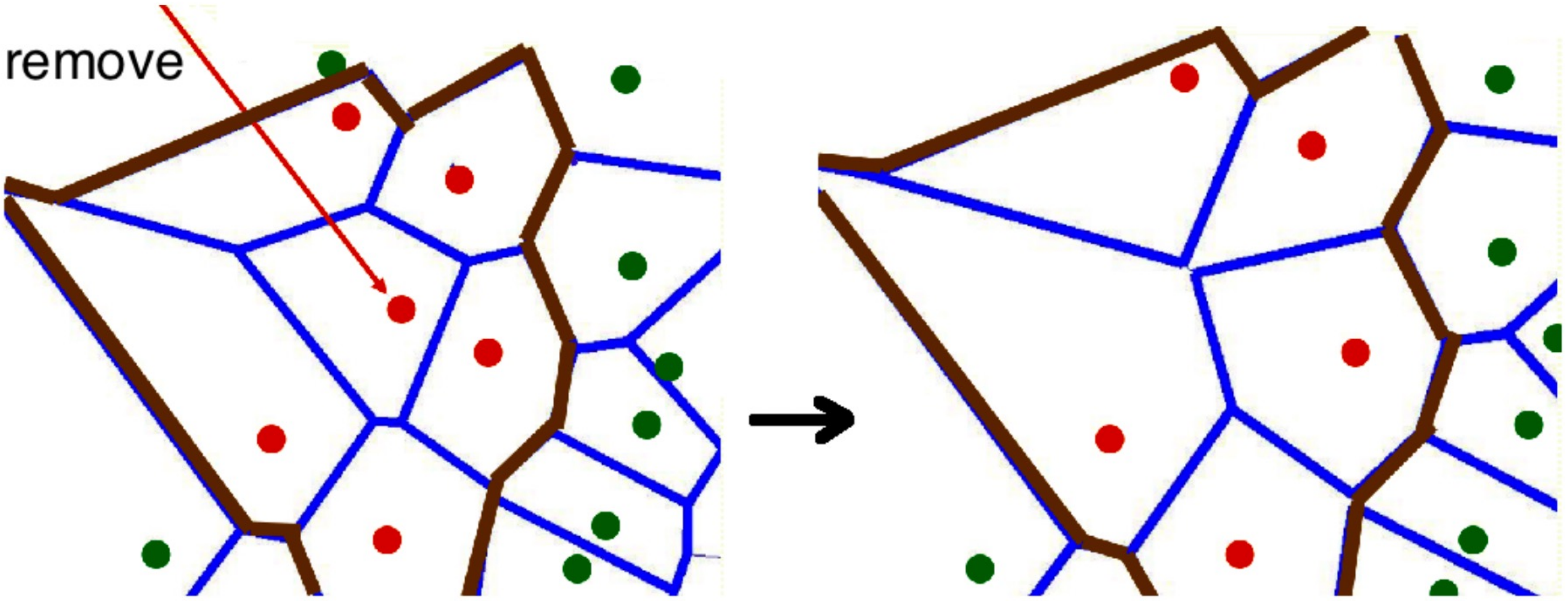

1 NN
noisy sample
every example in the blue shaded area will be misclassified as the blue class

Smooth by having a k voting scheme

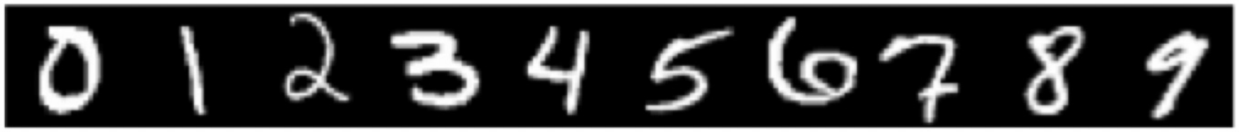


every example in the blue shaded area will be classified correctly as the red class

- If all Voronoi neighbors have the same class, a sample is useless, remove it



- Decent performance when lots of data



- Yann LeCunn – MNIST Digit Recognition
  - Handwritten digits
  - 28x28 pixel images: d = 784
  - 60,000 training samples
  - 10,000 test samples
- Nearest neighbour is competitive

	Test Error Rate (%)
Linear classifier (1-layer NN)	12.0
K-nearest-neighbors, Euclidean	5.0
K-nearest-neighbors, Euclidean, deskewed	2.4
K-NN, Tangent Distance, 16x16	1.1
K-NN, shape context matching	0.67
1000 RBF + linear classifier	3.6
SVM deg 4 polynomial	1.1
2-layer NN, 300 hidden units	4.7
2-layer NN, 300 HU, [deskewing]	1.6
LeNet-5, [distortions]	0.8
Boosted LeNet-4, [distortions]	0.7

KNN Input

A data set **D**.

A distance definition function **d**.

An integer **K**

The KNN Algorithm

- For a new observation **X** for which we want to predict its output variable **y:**

1. Calculate all the distances of this observation **X** with the other observations of the dataset **D**
2. Retain the **K** observations from the dataset **D** close to **X** using the distance calculation function **d**
3. Take the values of **y** from the **K** observations retained:
    1. If a regression problem, calculate the mean (or median) of **y** deductions
    2. If a classification problem, calculate the method of **y** deductions
4. Return the value calculated in step 3 as the value that was predicted by KNN for observation **X**.

Similarity measures

- Euclidean distance
- Taxicab geometry
- Minkowski distance
- Jaccard index
- Hamming distance

KNN

- The k-nearest neighbors algorithm (k-NN) is a **non-parametric classification method** first developed by Evelyn Fix and Joseph Hodges in 1951,[1] and later expanded by Thomas Cover.

- [1] Fix, Evelyn; Hodges, Joseph L. (1951). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties (PDF) (Report). USAF School of Aviation Medicine, Randolph Field, Texas.

- https://apps.dtic.mil/dtic/tr/fulltext/u2/a800276.pdf

What is a Parametric method

Select a form for the function.

Learn the coefficients for the function from the training data.

KNN is a non parametric method

- Algorithms that do not make strong assumptions about the form of the mapping function are called nonparametric machine learning algorithms. By not making assumptions, they are free to learn any functional form from the training data.
- An easy to understand nonparametric model is the k-nearest neighbors algorithm that makes predictions based on the k most similar training patterns for a new data instance. The method does not assume anything about the form of the mapping function other than patterns that are close are likely to have a similar output variable.

Benefits vs limitations of non parametric methods

- **Flexibility**: Capable of fitting a large number of functional forms.
- **Power**: No assumptions (or weak assumptions) about the underlying function.
- **Performance**: Can result in higher performance models for prediction.

- **More data**: Require a lot more training data to estimate the mapping function.
- **Slower**: A lot slower to train as they often have far more parameters to train.
- **Overfitting**: More of a risk to overfit the training data and it is harder to explain why specific predictions are made.