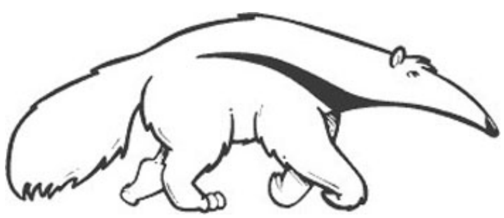
Machine Learning and Data Mining

Nearest neighbor methods

Prof. Alexander Ihler

Fall 2012

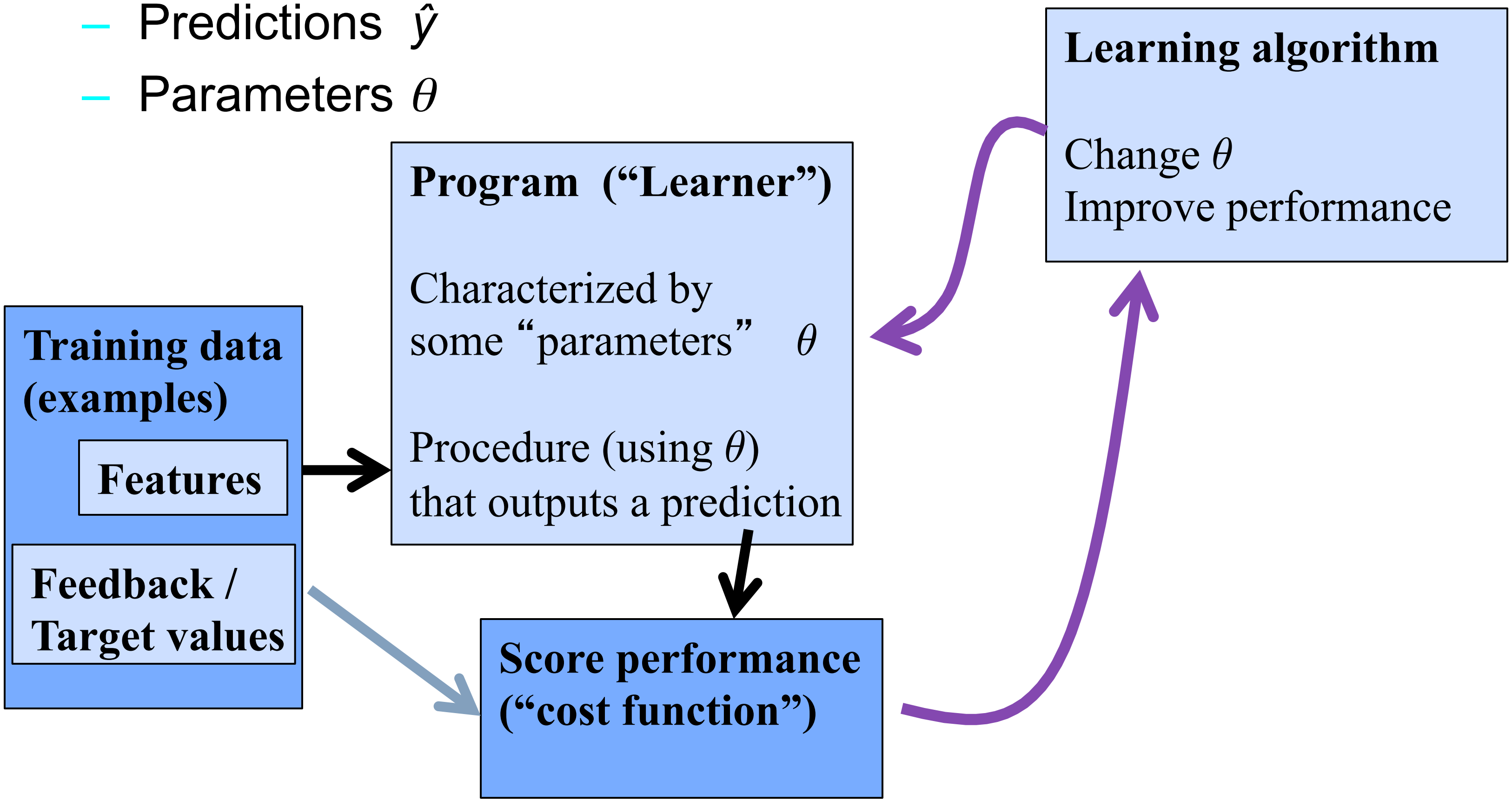


BREN:ICS
INFORMATION AND COMPUTER SCIENCES

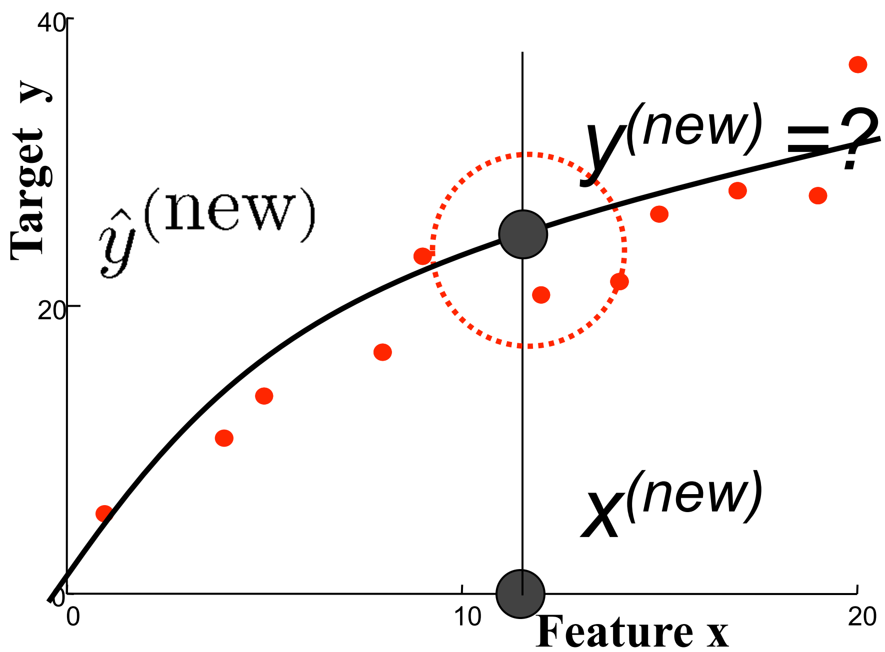
UNIVERSITY of CALIFORNIA IRVINE

Supervised learning

- Notation
  - Features x
  - Targets y
  - Predictions $\hat{y}$
  - Parameters θ

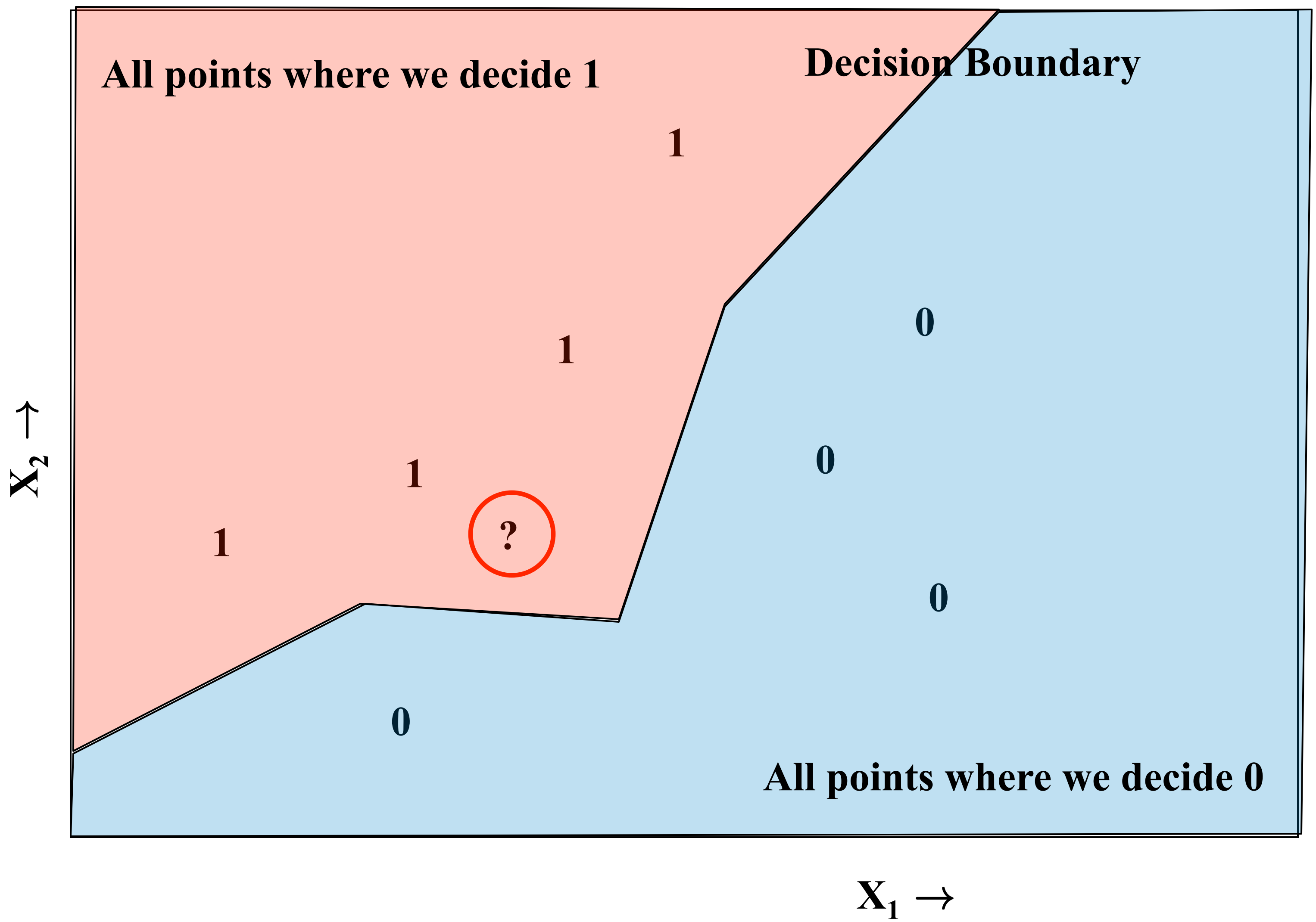


Regression; Scatter plots



- Suggests a relationship between x and y
- *Prediction*: new x, what is y?

Classification



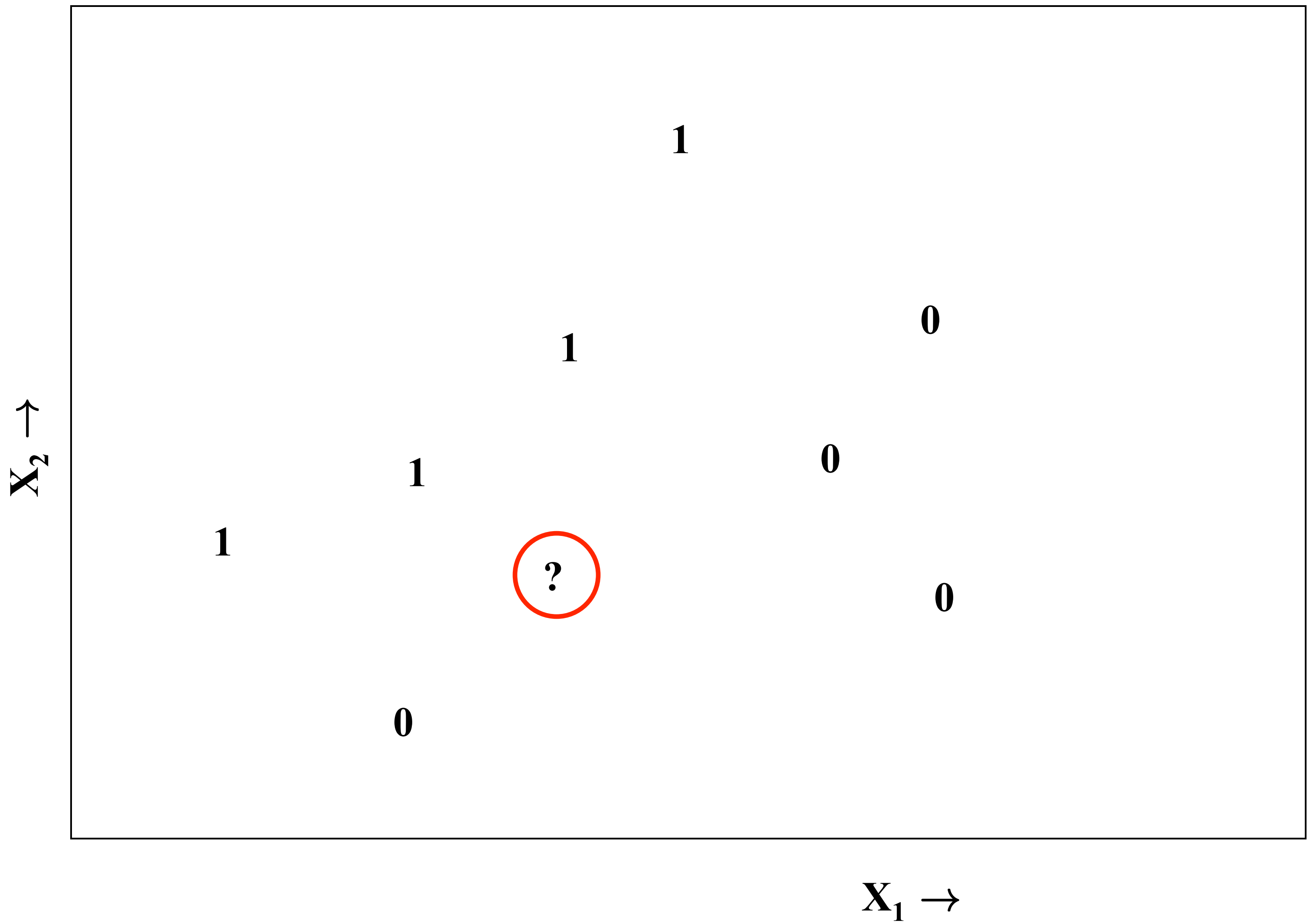
Nearest neighbor classifier

- $\underline{x}$ is a new feature vector whose class label is unknown
- Search training data for the closest feature vector to $\underline{x}$
  - Suppose the closest one is $\underline{x}^{(j)}$
- Classify $\underline{x}$ with the same label as $\underline{x}^{(j)}$, i.e.
  - Assign $\underline{x}$ the predicted label $\underline{y}^{(j)}$
- Interpretation as memorization

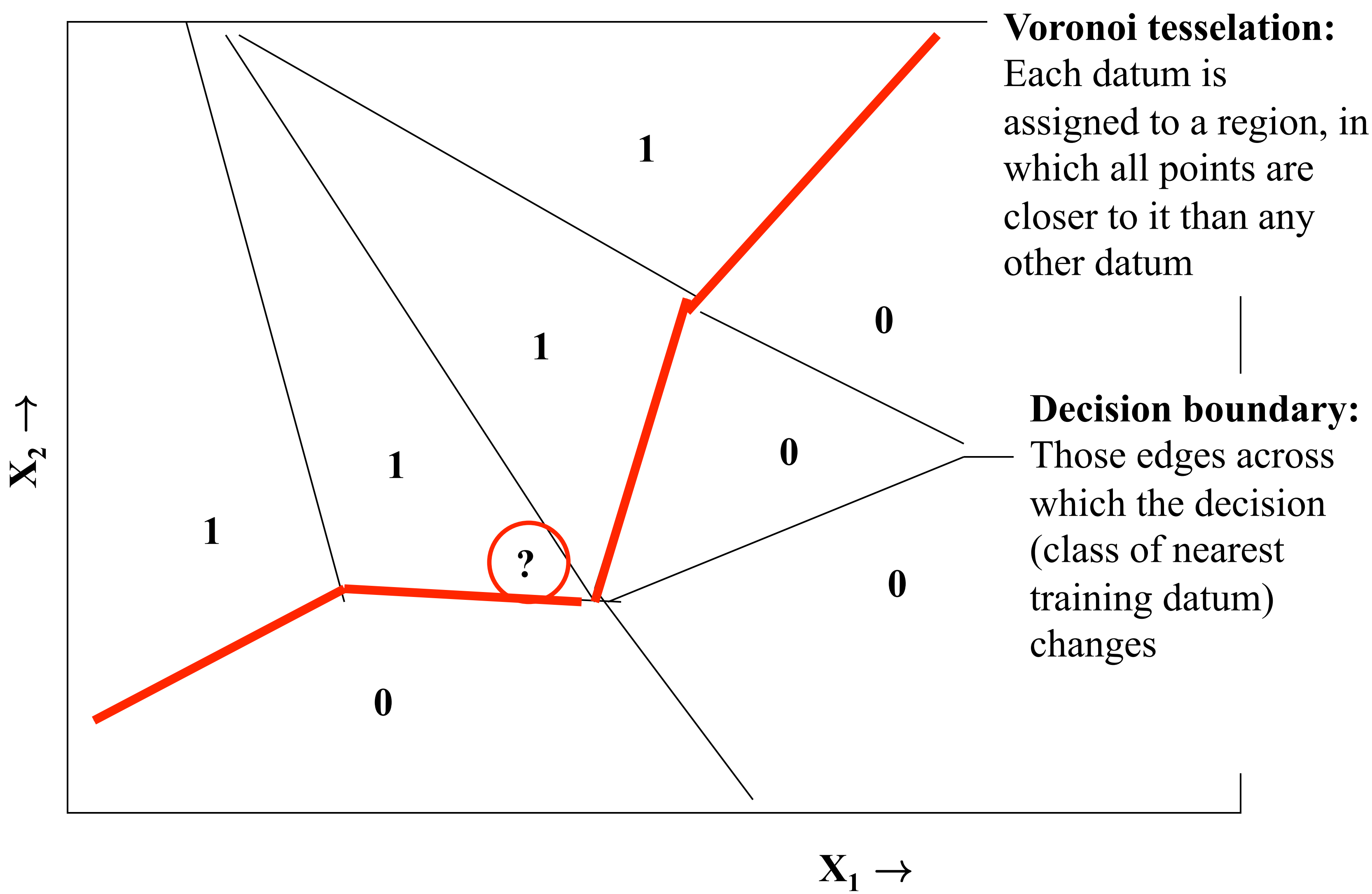
- How are "closest $\underline{x}$" vectors determined?
  - typically use minimum Euclidean distance

$$d(x, x') = \sqrt{\sum_i (x_i - x'_i)^2}$$

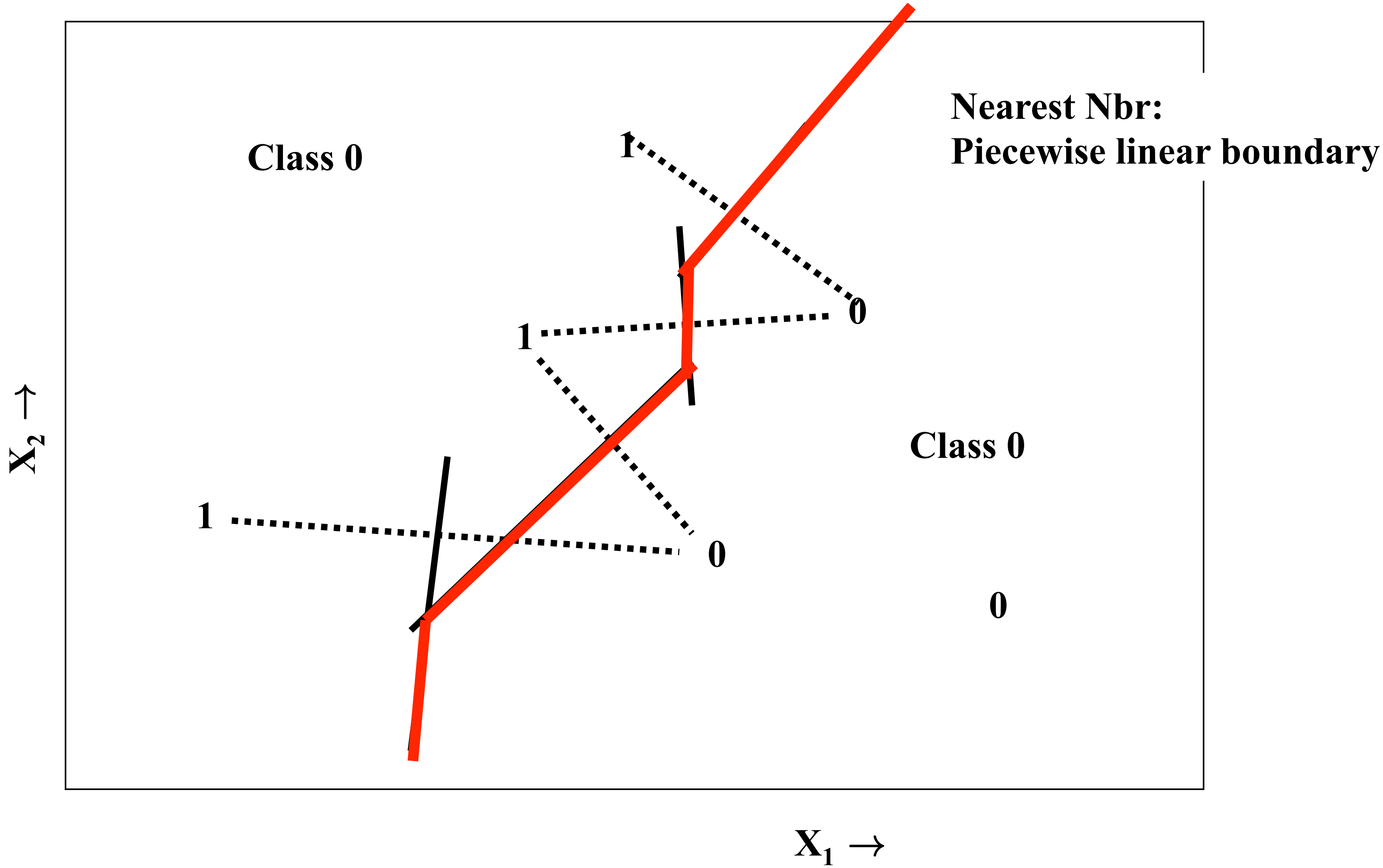
Nearest neighbor classifier



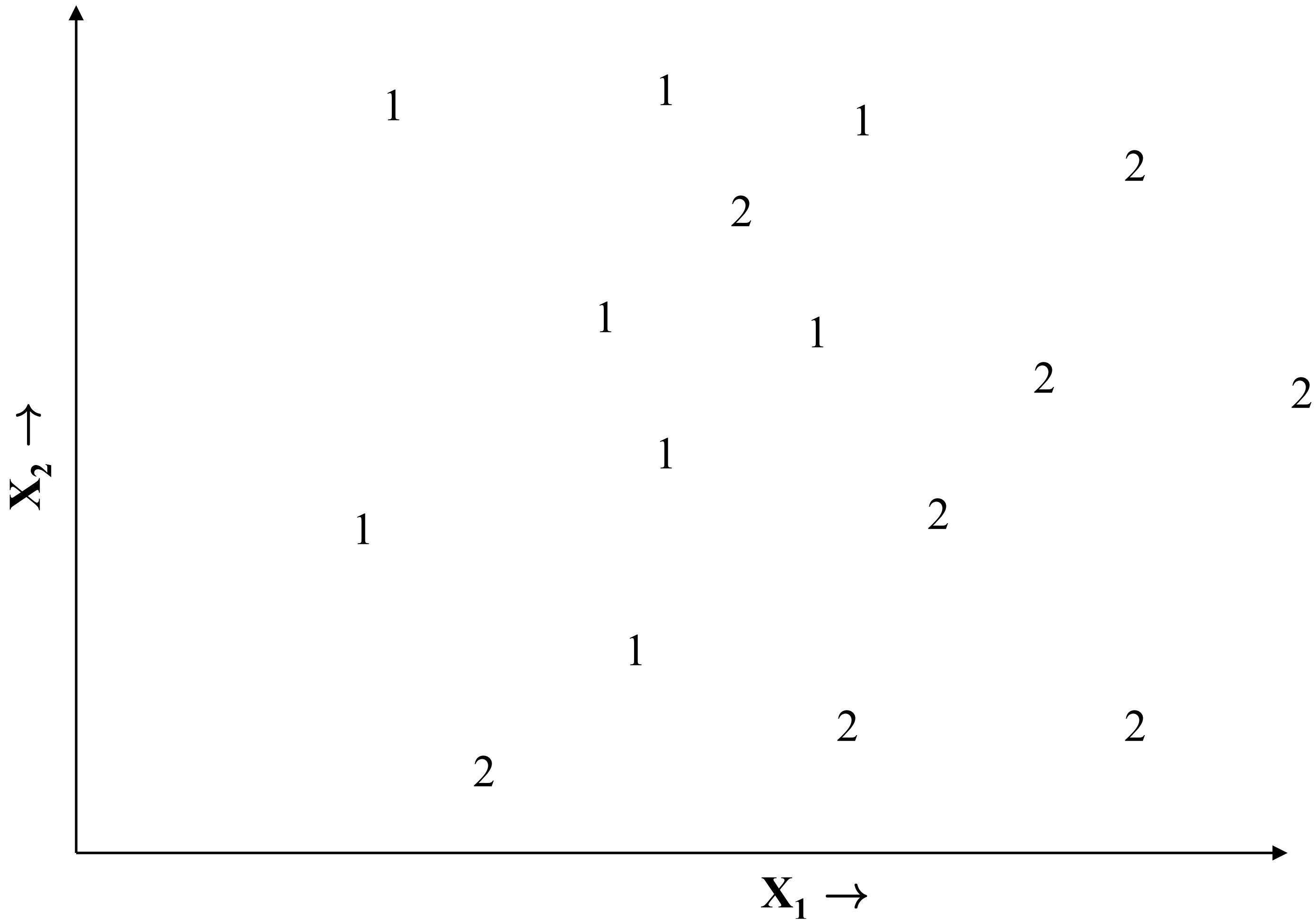
Nearest neighbor classifier



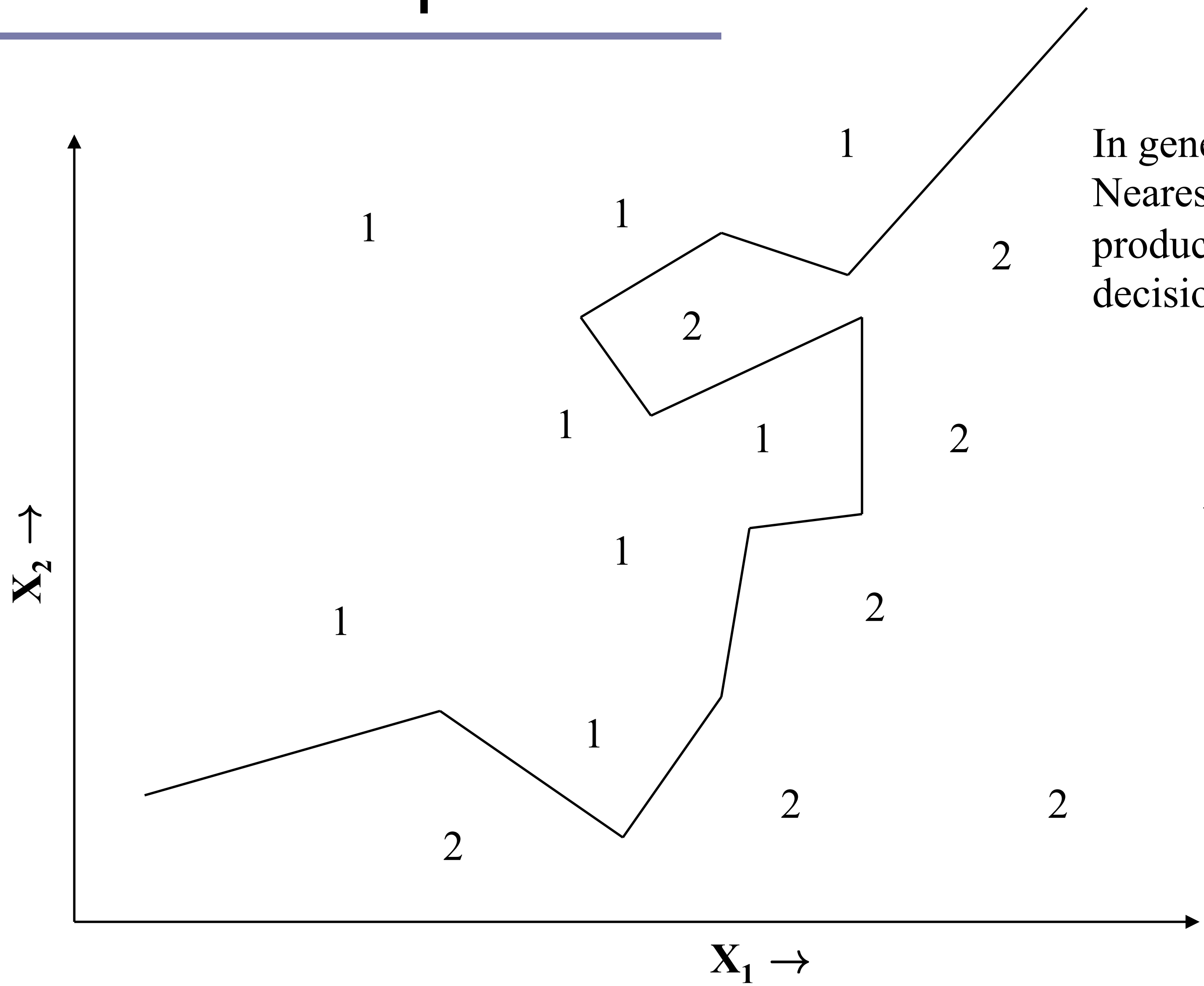
Nearest neighbor classifier



More Data Points



More Complex Decision Boundary



In general:
Nearest-neighbor classifier produces piecewise linear decision boundaries

K-Nearest Neighbor (kNN) Classifier

- Find the k-nearest neighbors to $\underline{x}$ in the data
  - i.e., rank the feature vectors according to Euclidean distance
  - select the k vectors which are have smallest distance to $\underline{x}$

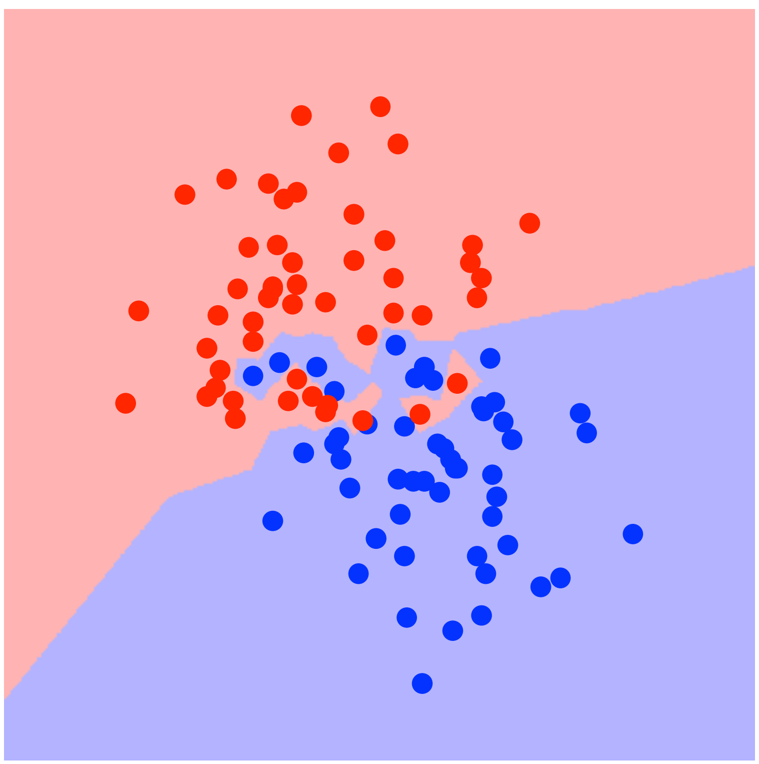
- Classification
  - ranking yields k feature vectors and a set of k class labels
  - pick the class label which is most common in this set ("vote")
  - classify $\underline{x}$ as belonging to this class

- Notes:
  - Nearest k feature vectors from training "vote" on a class label for $\underline{x}$
  - the single-nearest neighbor classifier is the special case of k=1
  - for two-class problems, if we choose k to be odd (i.e., k=1, 3, 5,…) then there will never be any "ties"
  - "training" is trivial for the kNN classifier, i.e., we just use training data as a "lookup table" and search to classify a new datum

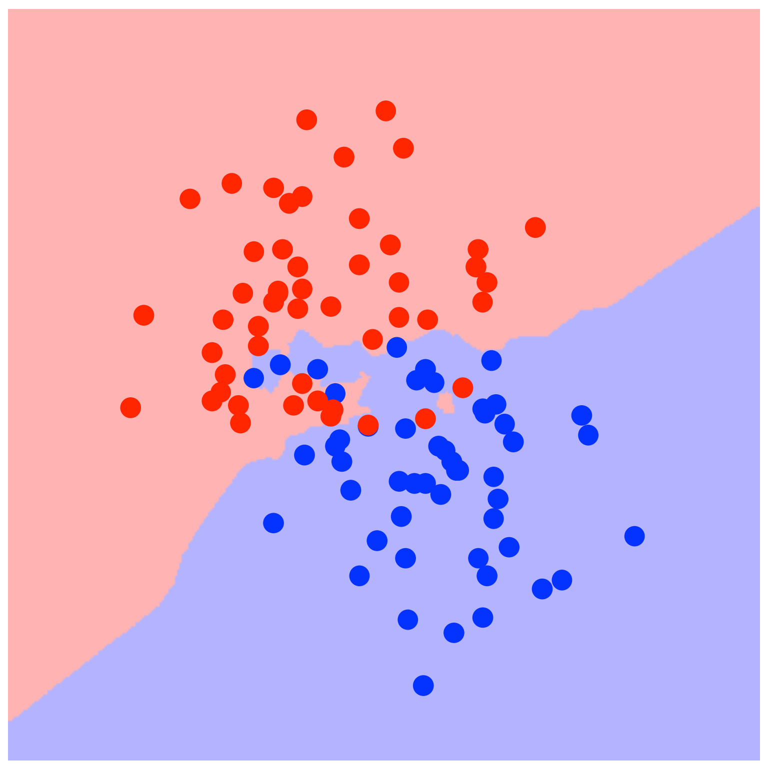
kNN Decision Boundary

- Piecewise linear decision boundary
- Increasing k “simplifies” decision boundary
  - Majority voting means less emphasis on individual points

K = 1

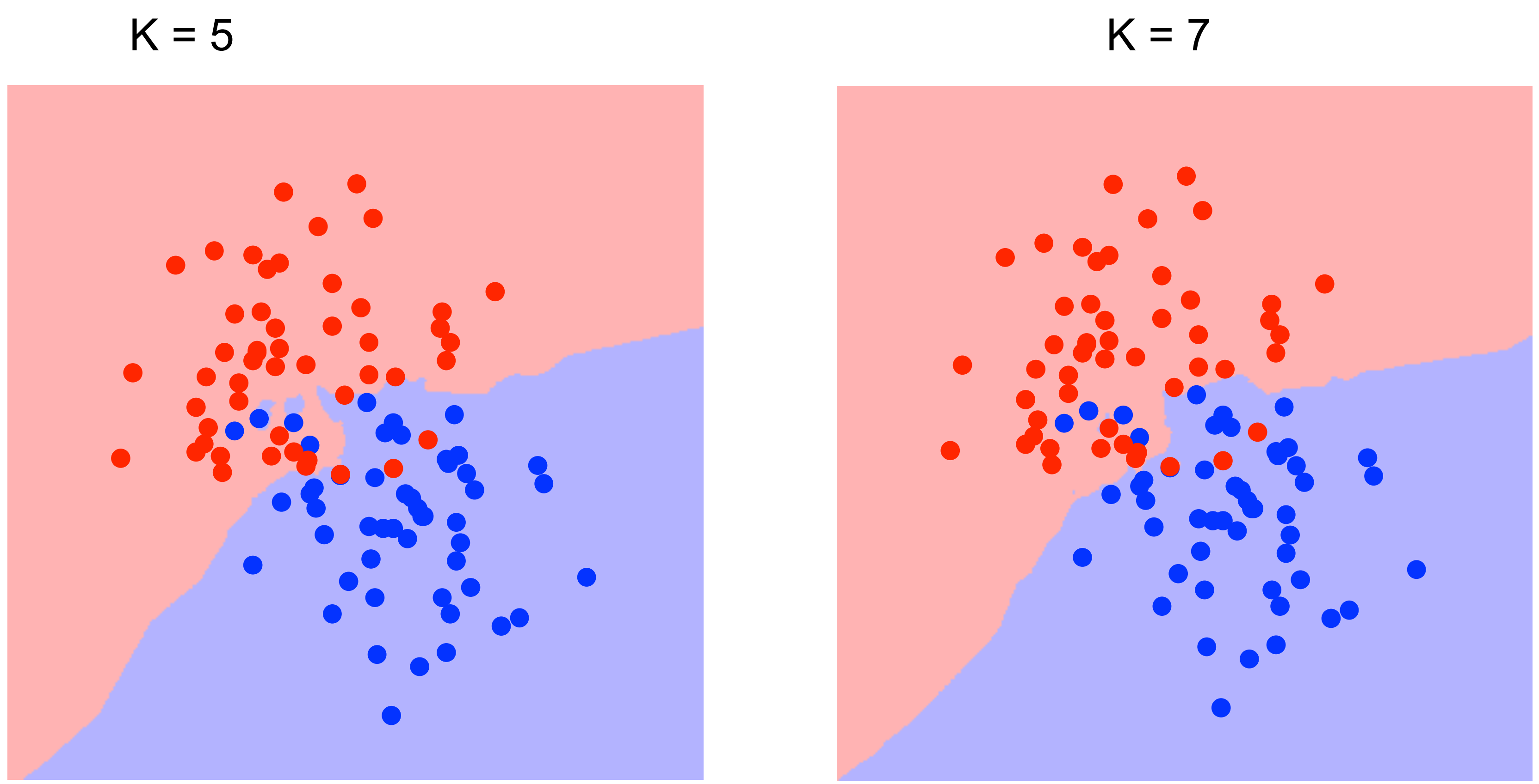


K = 3



kNN Decision Boundary

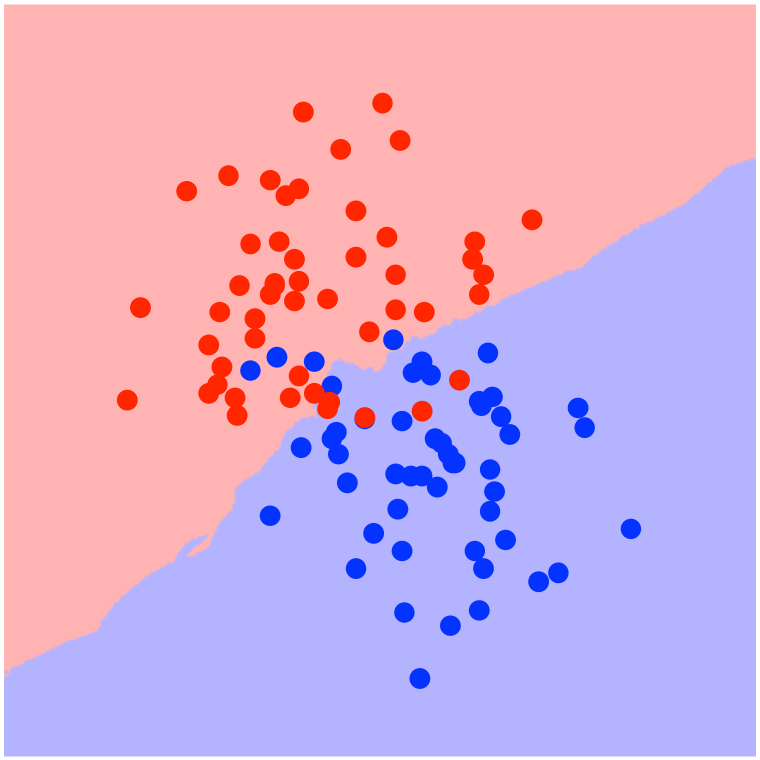
- Recall: piecewise linear decision boundary
- Increasing k “simplifies” decision boundary
  - Majority voting means less emphasis on individual points



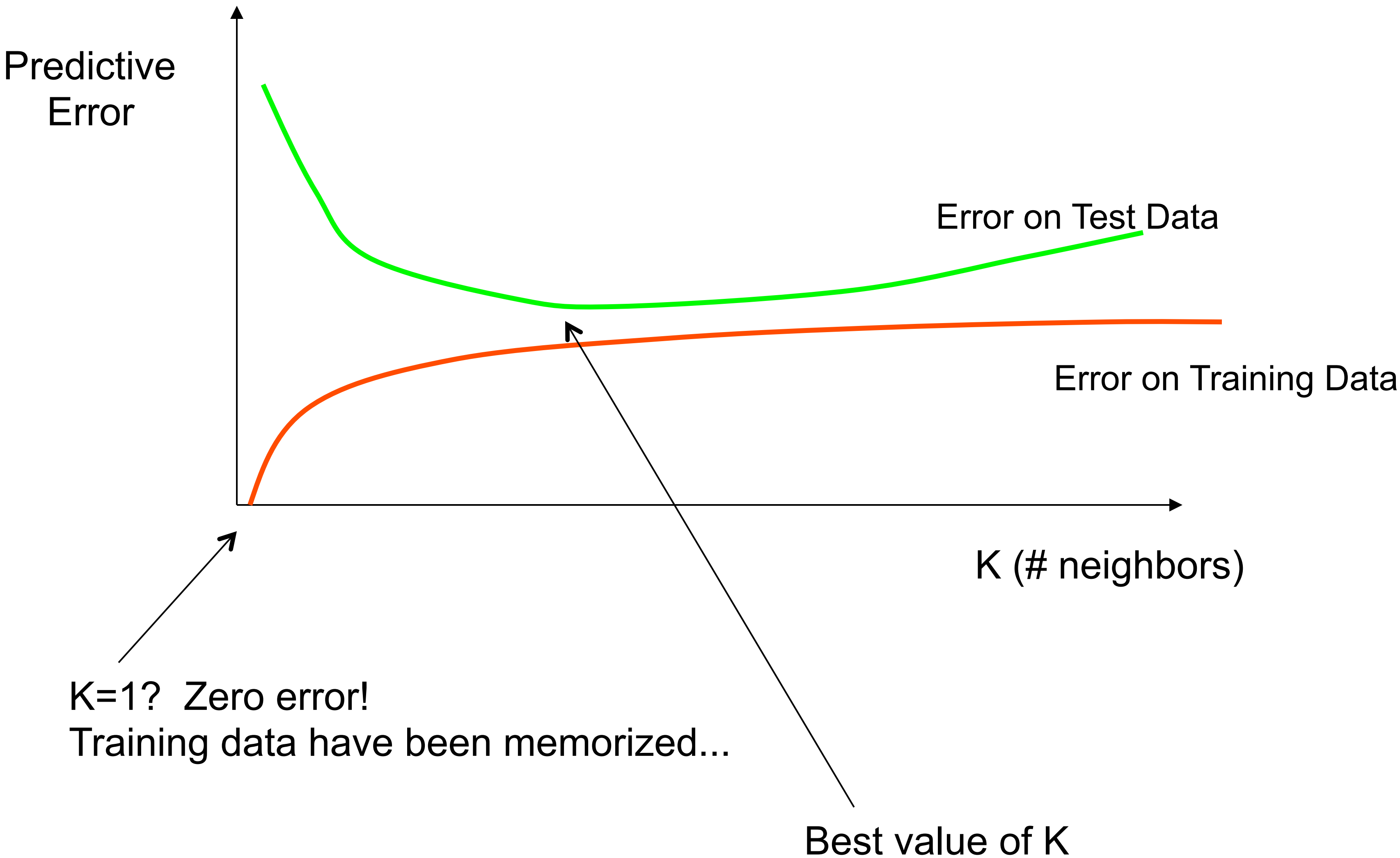
kNN Decision Boundary

- Recall: piecewise linear decision boundary
- Increasing k “simplifies” decision boundary
  - Majority voting means less emphasis on individual points

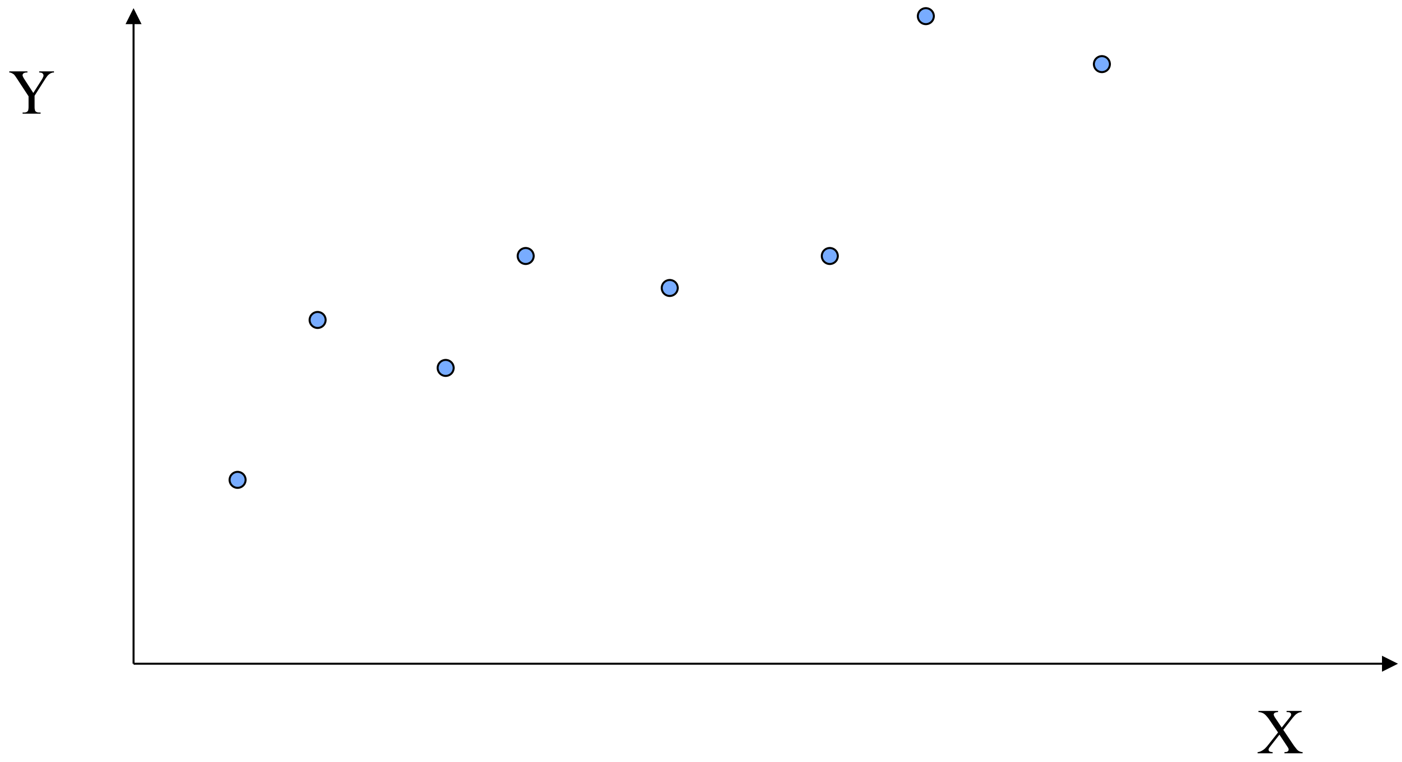
K = 25



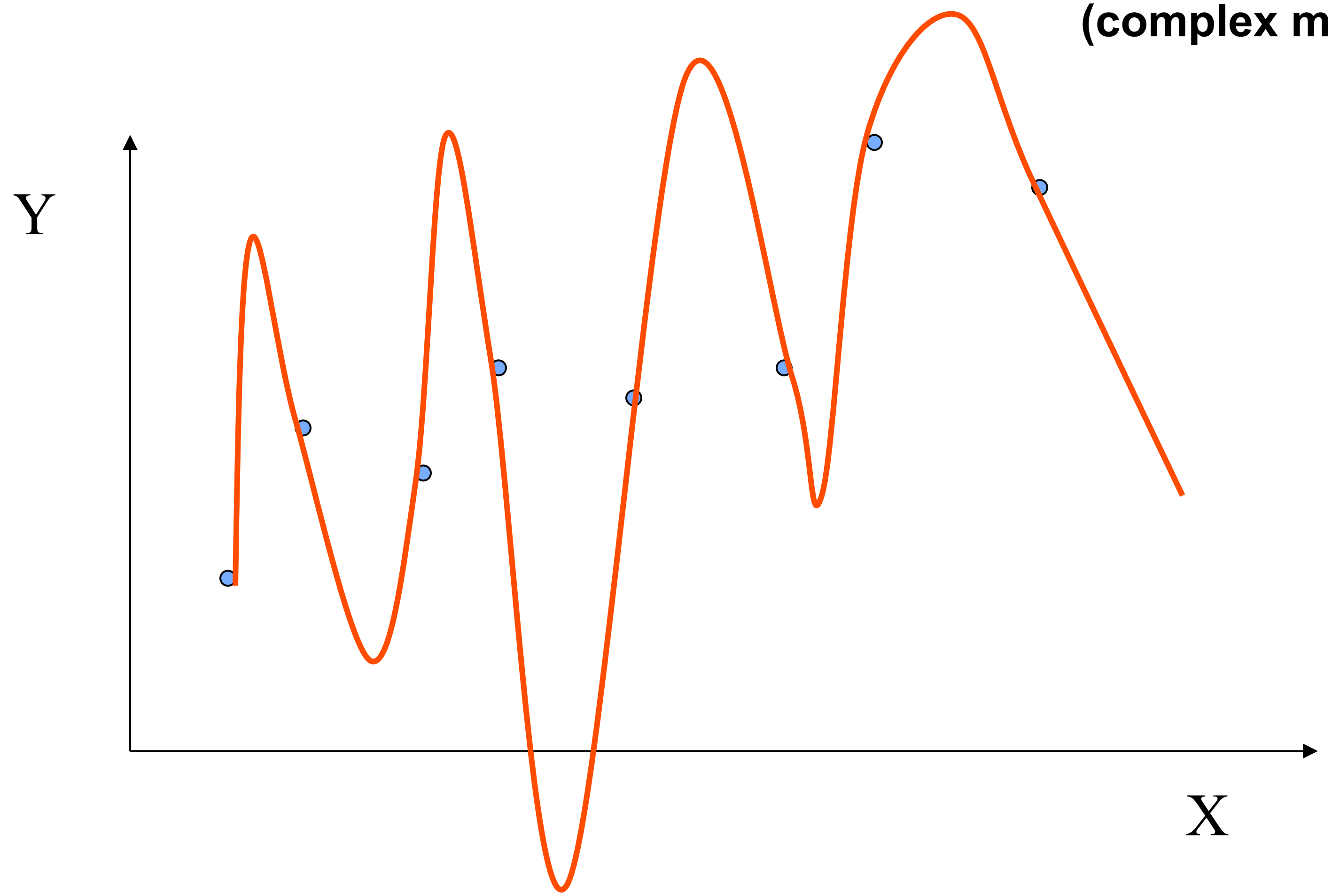
Error rates and K



Overfitting and complexity

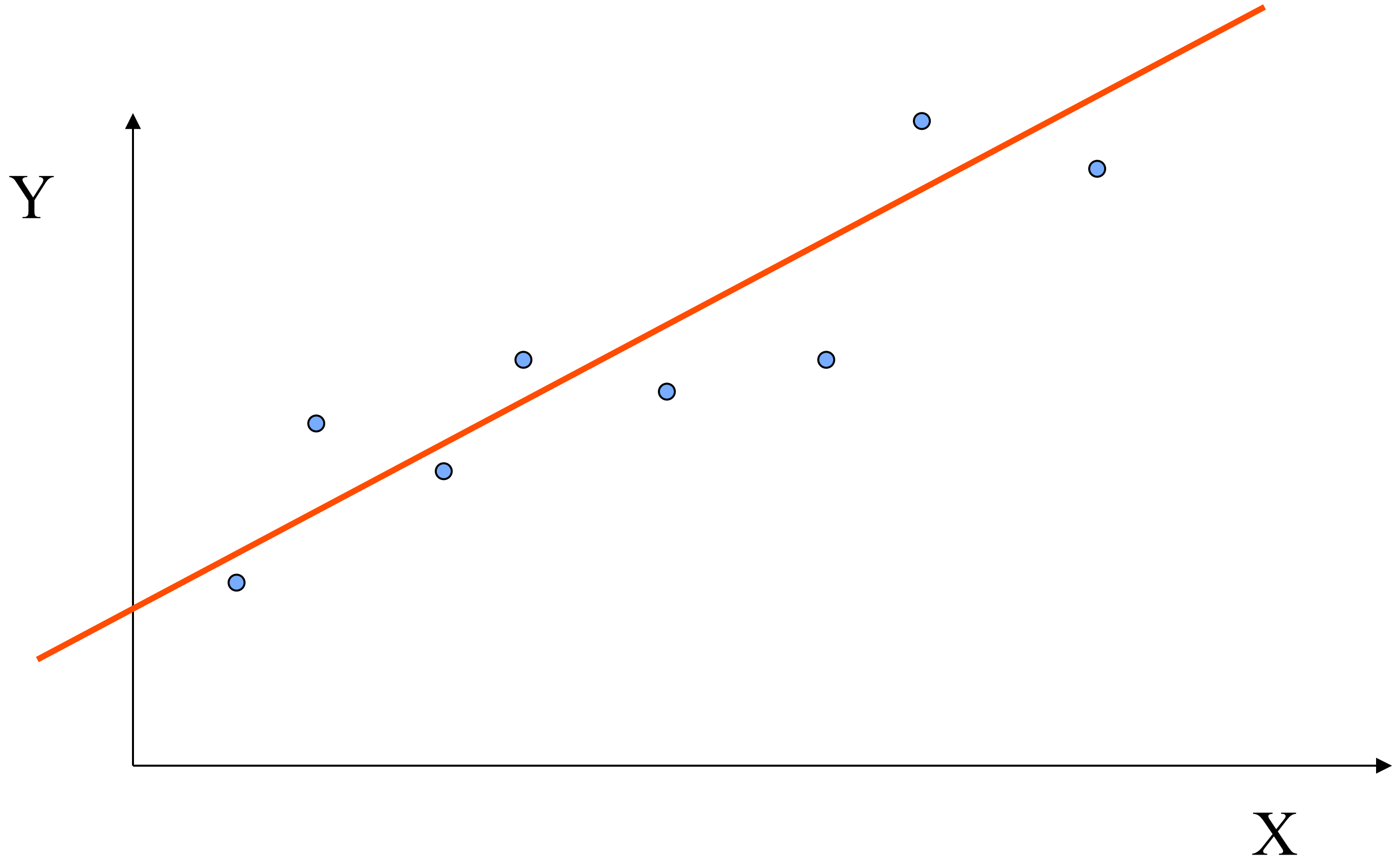


Overfitting and complexity



Overfitting and complexity

Simple model: Y= aX + b + e



Detecting overfitting

- Overfitting effect
  - Do better on training data than on future data
  - Need to choose the "right" complexity

- One solution: "Hold-out" or "validation" data
- Separate our data into two sets
  - Training
  - Test
- Learn only on training data
- Use test data to estimate generalization quality
  - Model selection

- All good competitions use this formulation
  - Often multiple splits: one by judges, then another by you

K-Nearest Neighbor (kNN) Classifier

- Theoretical Considerations
  - as k increases
    - we are averaging over more neighbors
    - the effective decision boundary is more "smooth"
  - as N increases, the optimal k value tends to increase
  - k=1, m increasing to infinity : error < 2x optimal

- Extensions of the Nearest Neighbor classifier
  - weighted distances
    - e.g., if some of the features are more important
    - e.g., if features are irrelevant

$$d(x, x') = \sqrt{\sum_i w_i (x_i - x'_i)^2}$$

  - fast search techniques (indexing) to find k-nearest neighbors in d-space

Summary

- K-nearest neighbor models
  - Classification (vote)
  - Regression (average or weighted avg)

- Piecewise linear decision boundary
  - How to calculate

- Test data and overfitting
  - Model “complexity” for knn
  - Validation data for test error rates