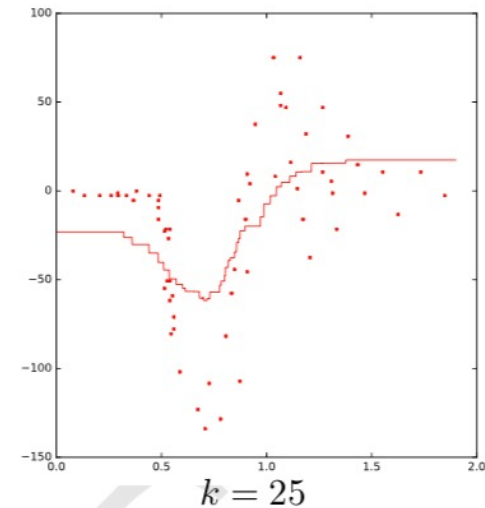
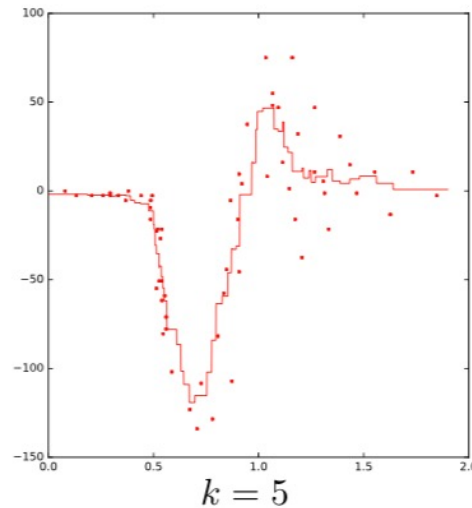
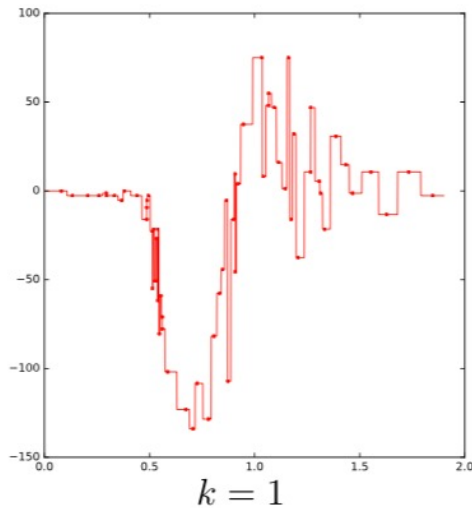


CS273A: Nearest Neighbor Methods



Prof. Alexander Ihler
Fall 2024

Machine Learning

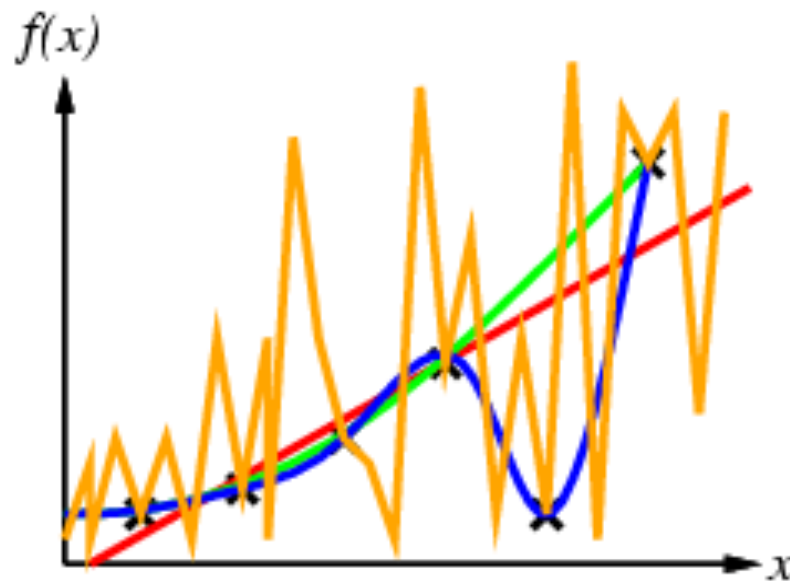
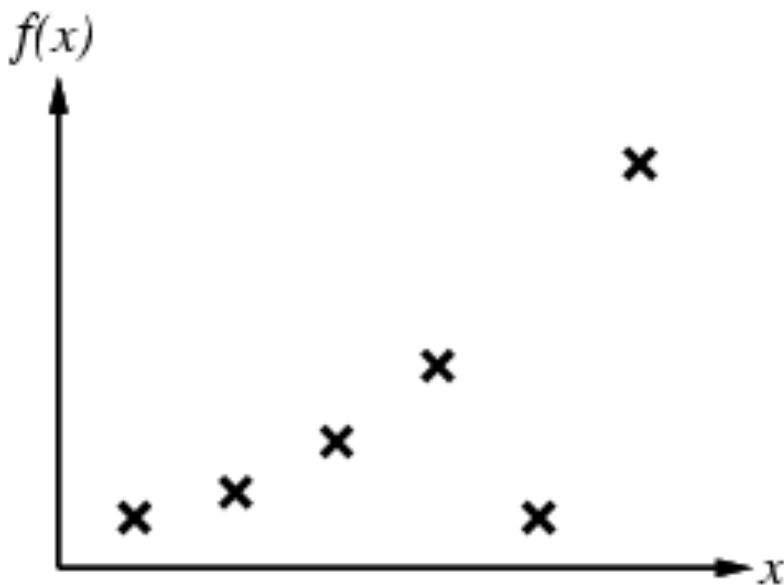
Complexity and Overfitting

Nearest Neighbors

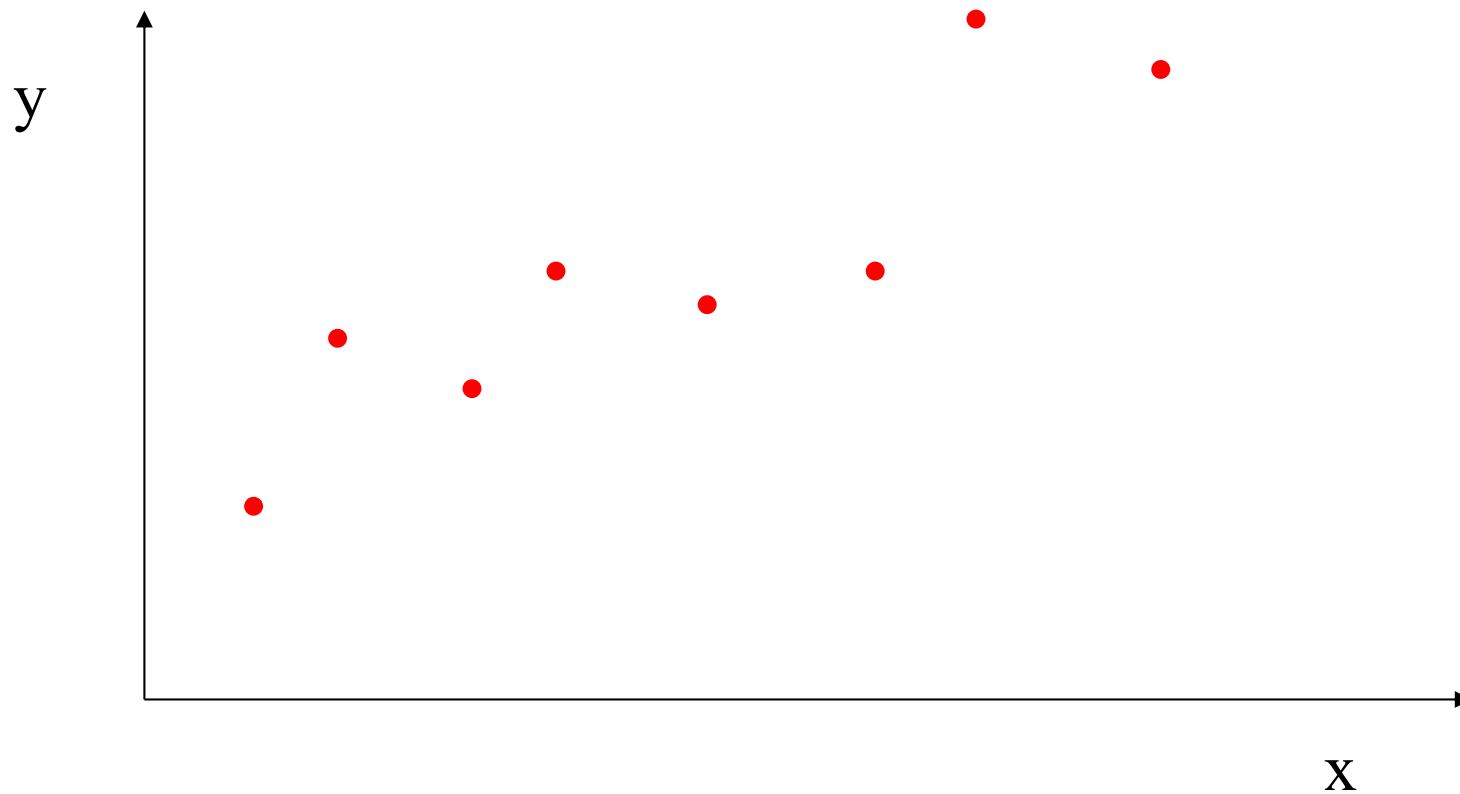
K-Nearest Neighbors

Inductive bias

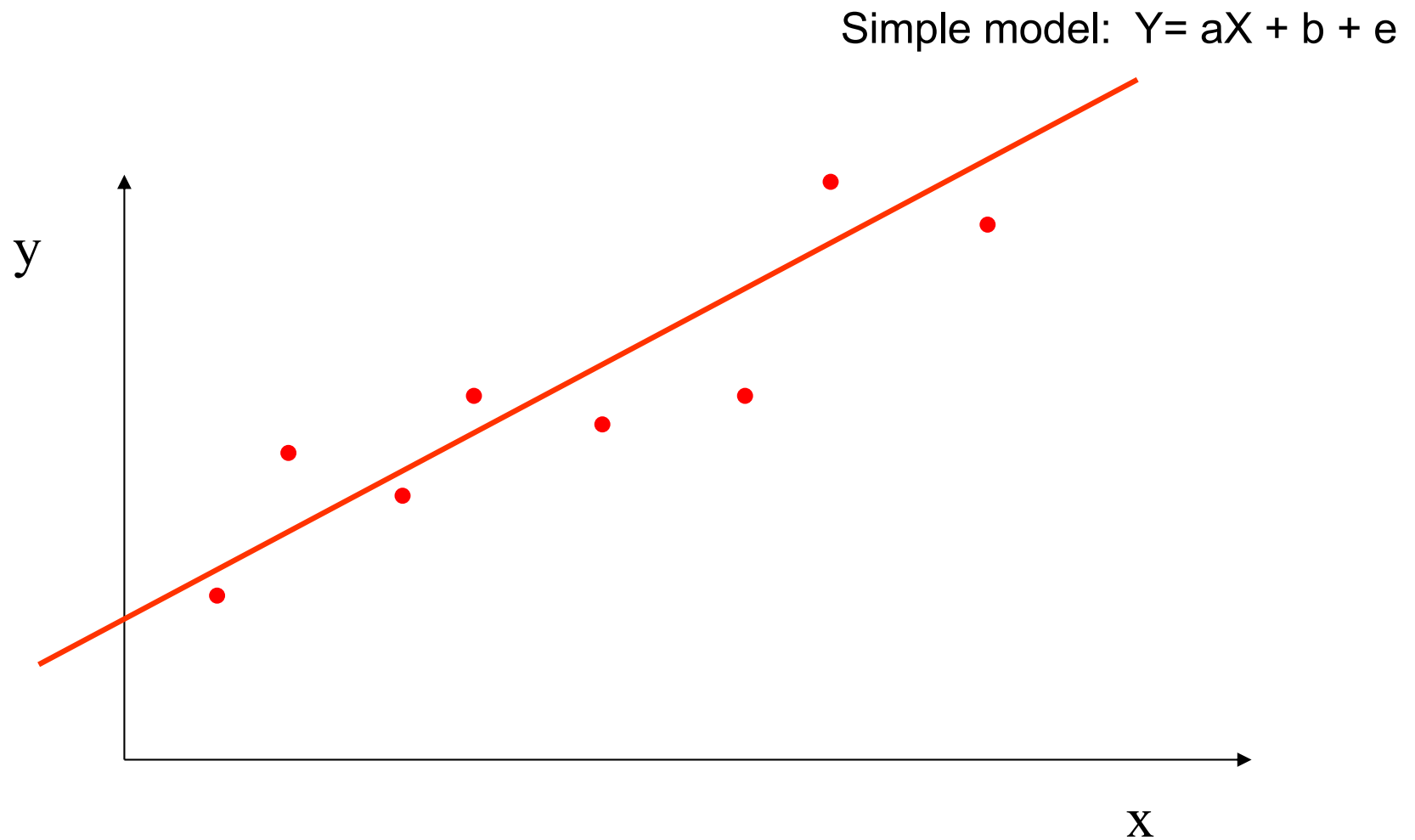
- “Extend” observed data to unobserved examples
 - “Interpolate” / “extrapolate”
- What kinds of functions to expect? Prefer these (“bias”)
 - Usually, let data pull us away from assumptions only with evidence!



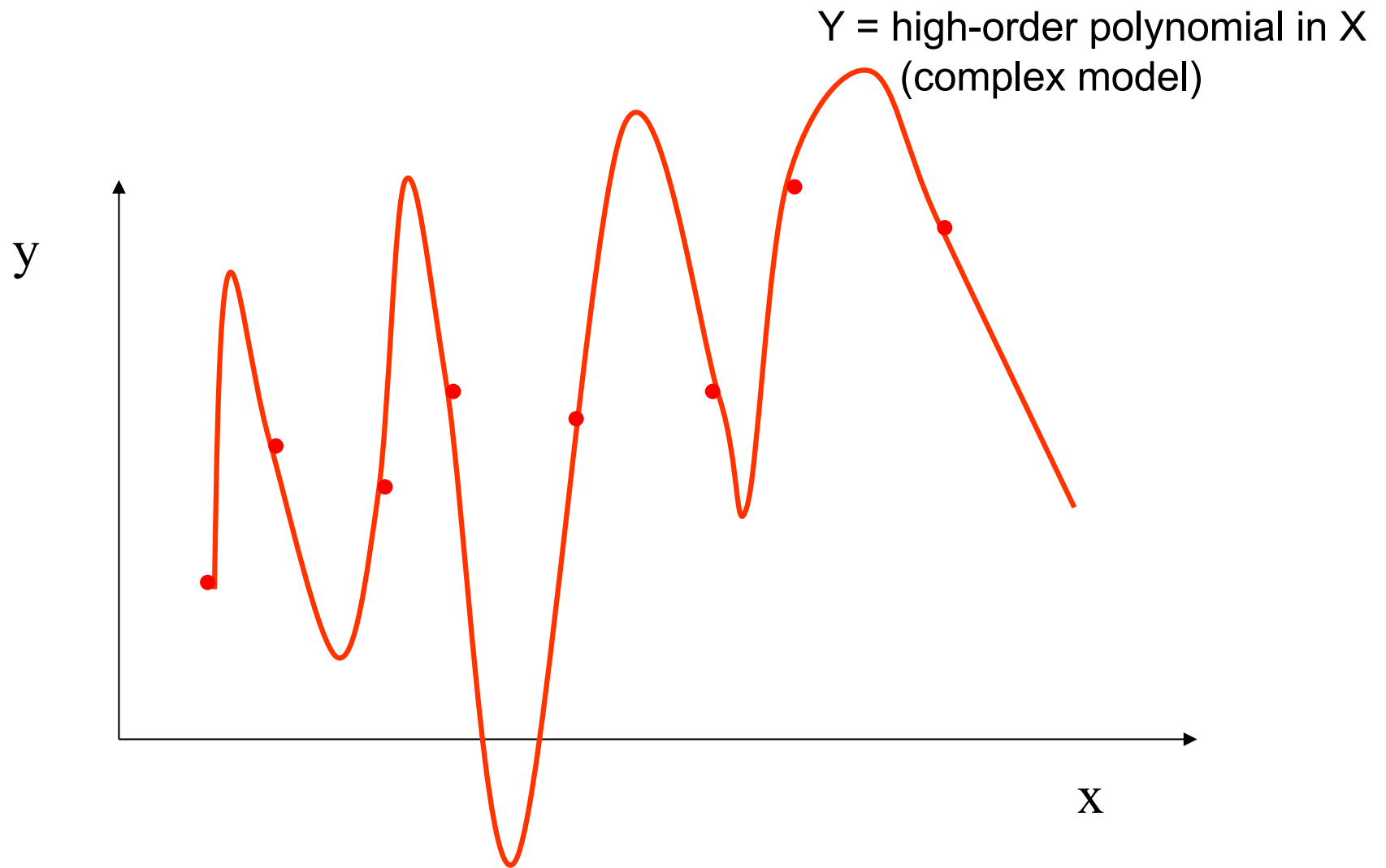
Overfitting and complexity



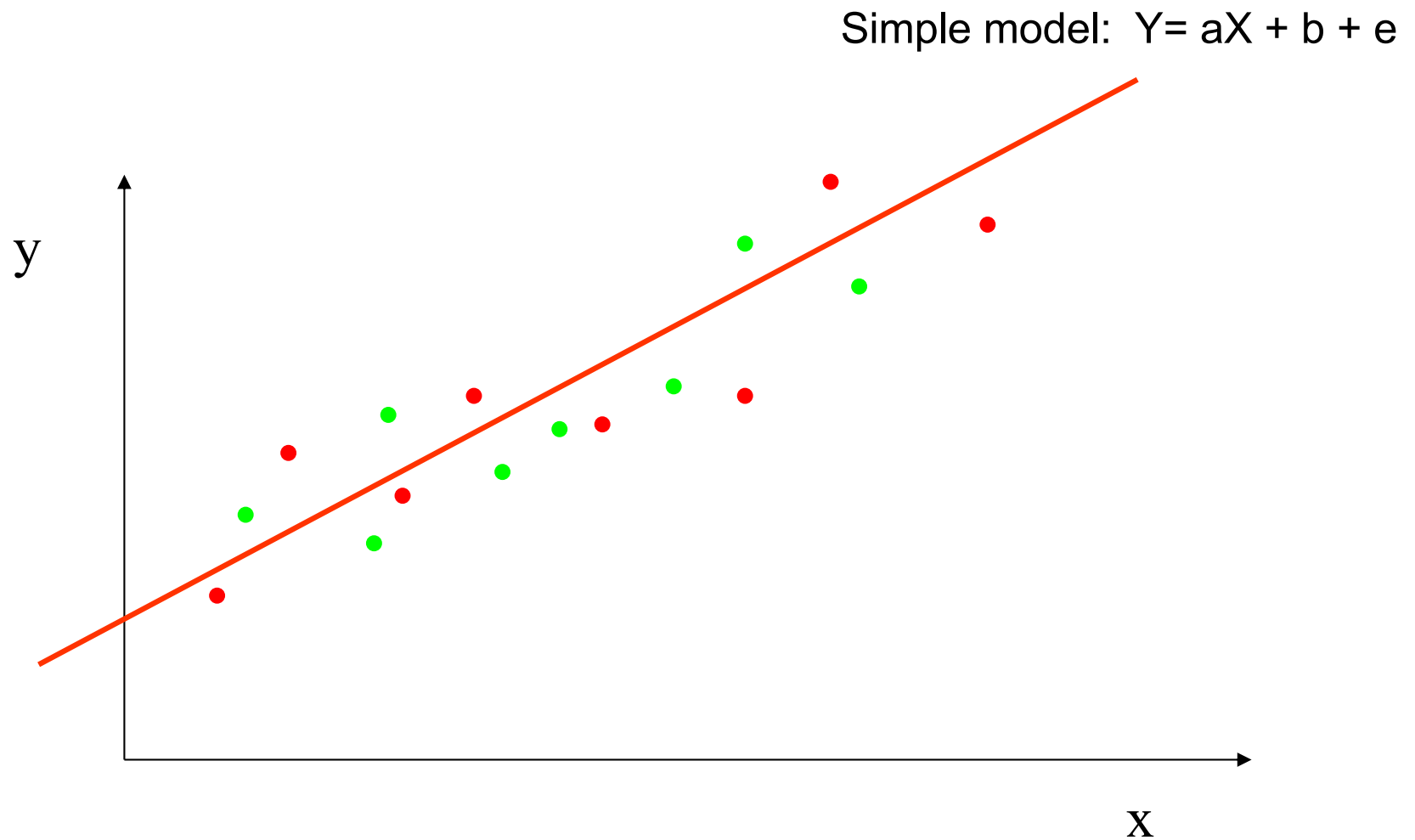
Overfitting and complexity



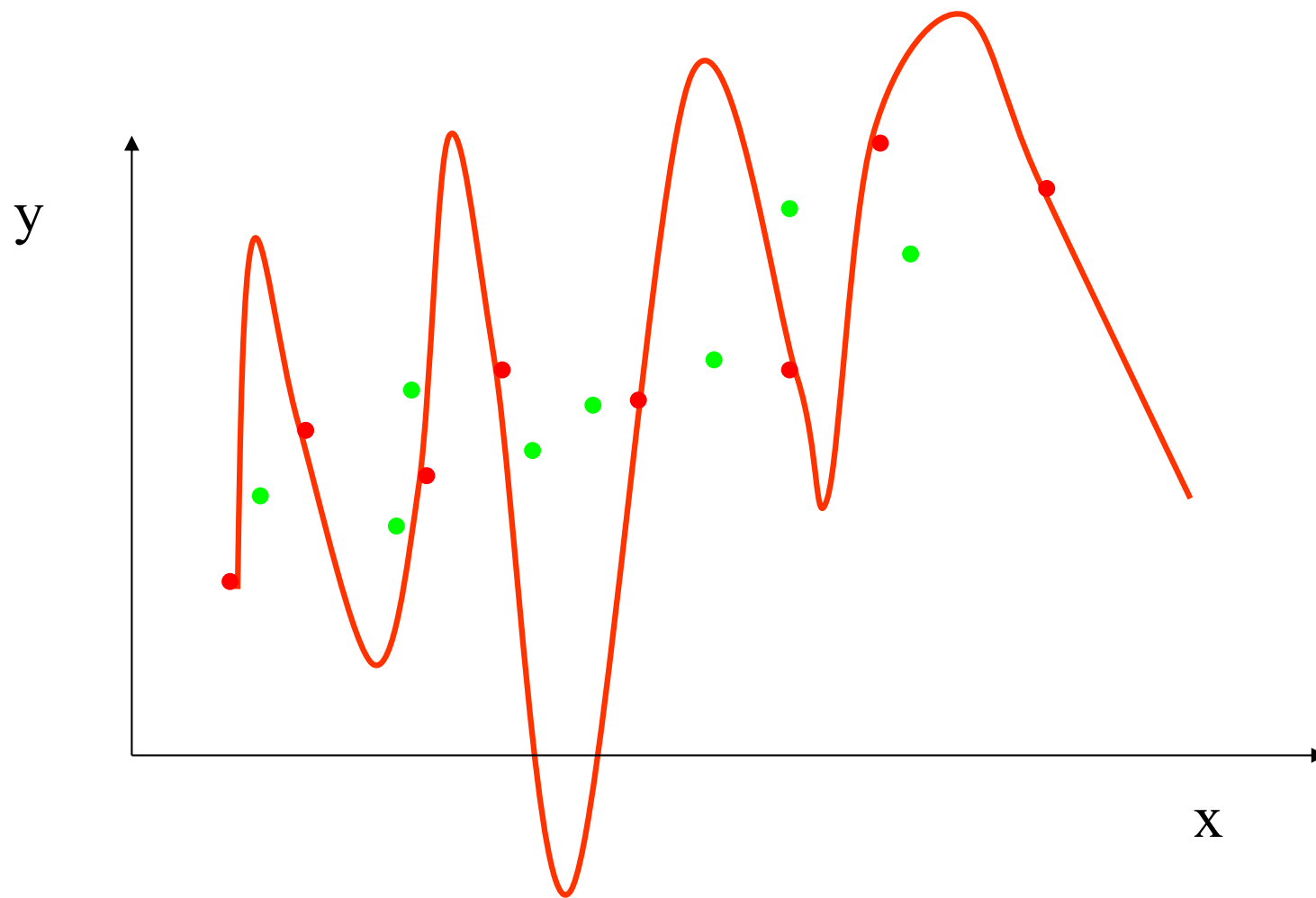
Overfitting and complexity



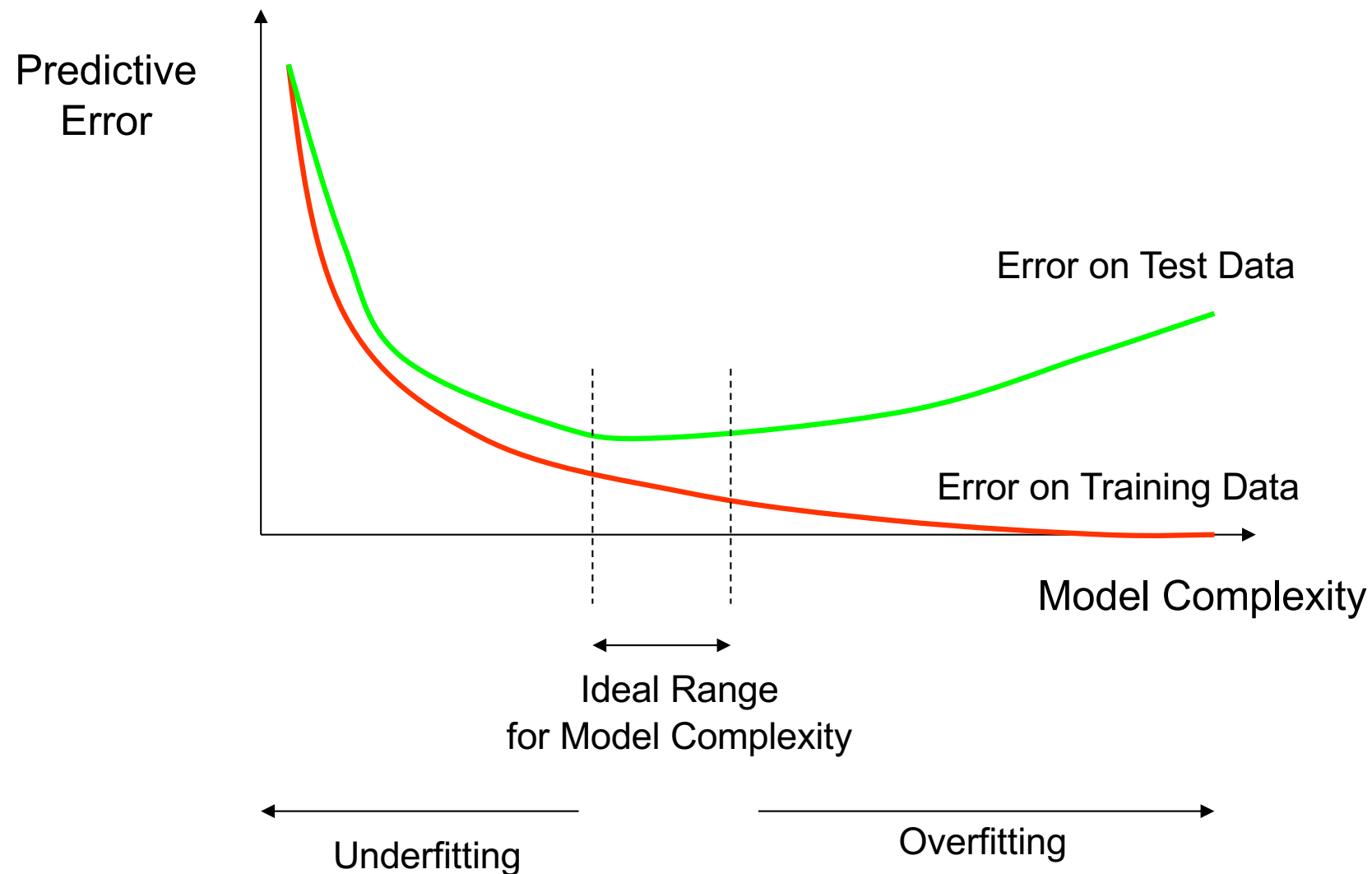
Overfitting and complexity



Overfitting and complexity



How Overfitting affects Prediction



Training and Test Data

Data

- Several candidate learning algorithms or models, each of which can be fit to data and used for prediction
- How can we decide which is best?

Approach 1: Split into train and test data

Training Data

Test Data

- Learn parameters of each model from training data
- Evaluate all models on test data, and pick best performer

Problem:

- Over-estimates test performance (“lucky” model)
- Learning algorithms should *never* have access to test data

Training, Validation, and Test Data

Data

- Several candidate learning algorithms or models, each of which can be fit to data and used for prediction
- How can we decide which is best?

Approach 2: Reserve some data for validation

Training Data

Validation

Test Data

- Learn parameters of each model from training data
- Evaluate models on validation data, pick best performer
- Reserve test data to benchmark chosen model

Problem:

- Wasteful of training data (learning can't use validation)
- May bias selection towards overly simple models

Competitions

- Training data
 - Used to build your model(s)
- Validation data
 - Used to assess, select among, or combine models
 - Personal validation; leaderboard; ...
- Test data
 - Used to estimate “real world” performance

#	Δ1w	Team Name <small>* in the money</small>	Score <small>?</small>	Entries	Last Submission UT
1	-	BrickMover <small>👤 *</small>	1.21251	40	Sat, 31 Aug 2013 23:
2	new	vsu <small>*</small>	1.21552	13	Sat, 31 Aug 2013 20:
3	↑2	Merlion	1.22724	29	Sat, 31 Aug 2013 23:
4	↓2	Sergey	1.22856	15	Sat, 31 Aug 2013 23:
5	new	liuyongqi	1.22980	13	Sat, 31 Aug 2013 13:

Machine Learning

Complexity and Overfitting

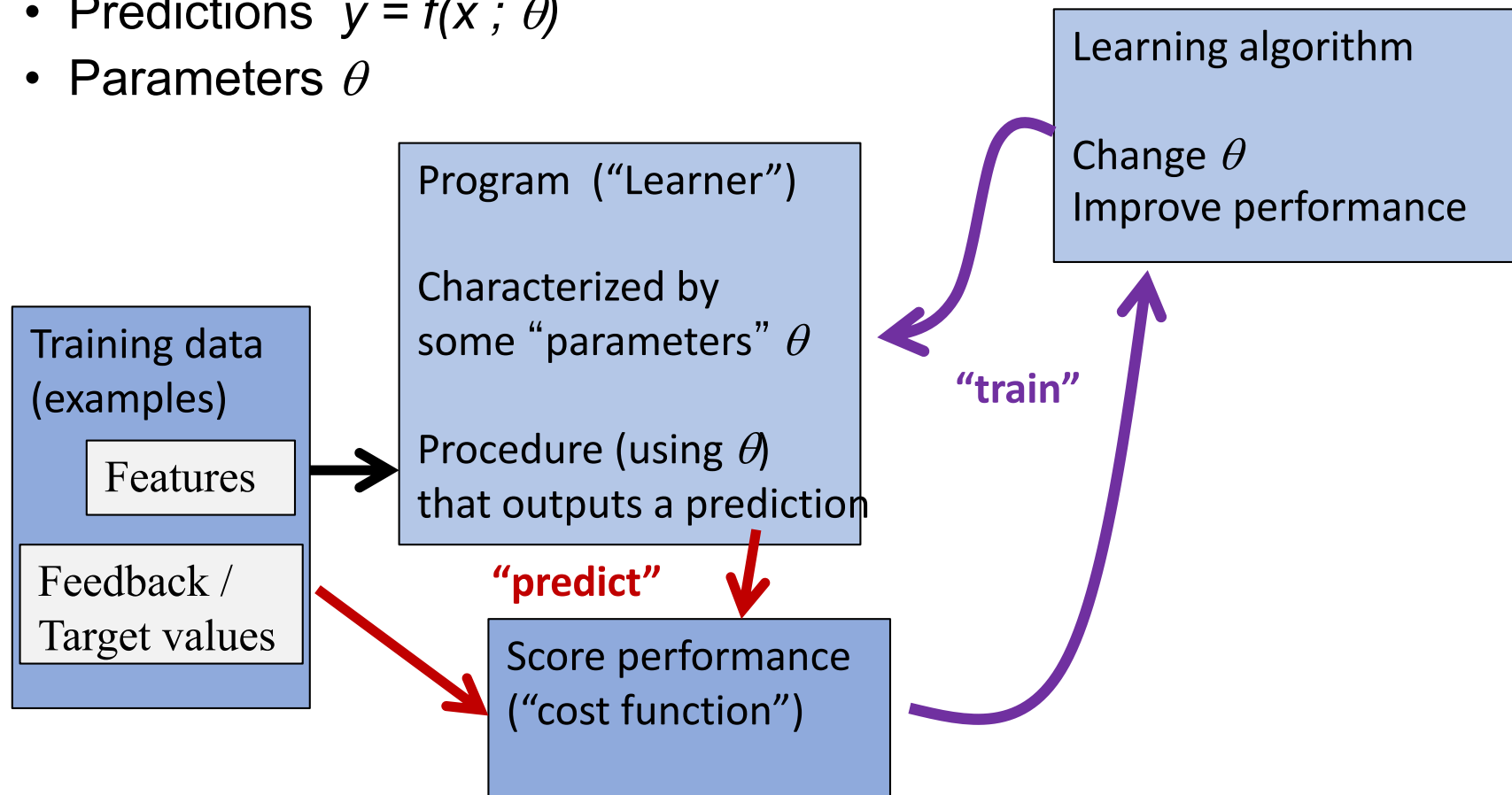
Nearest Neighbors

K-Nearest Neighbors

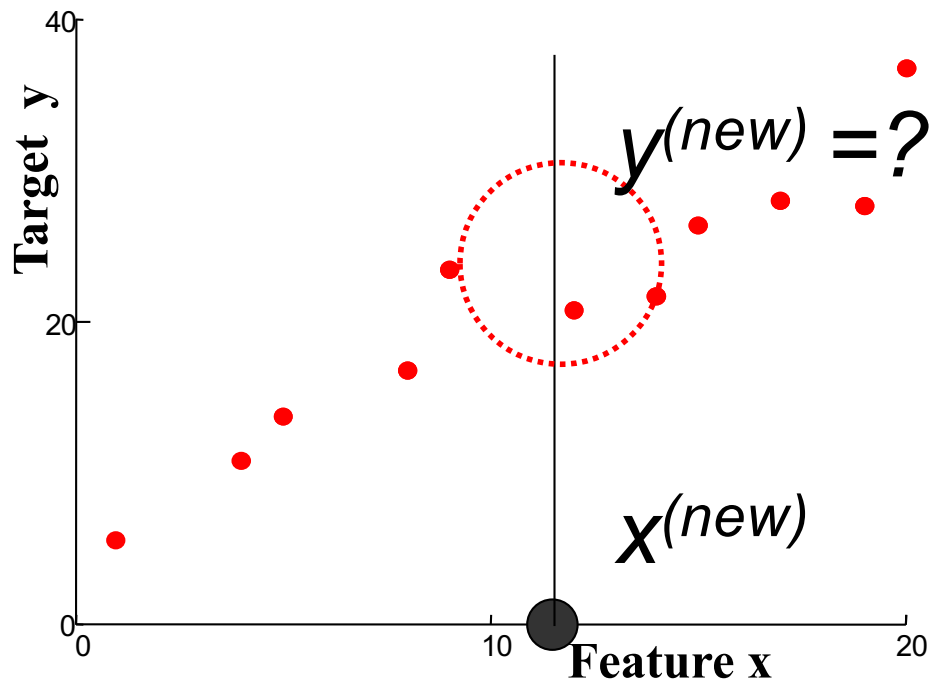
Supervised learning

- Notation

- Features x
- Targets y
- Predictions $\hat{y} = f(x; \theta)$
- Parameters θ

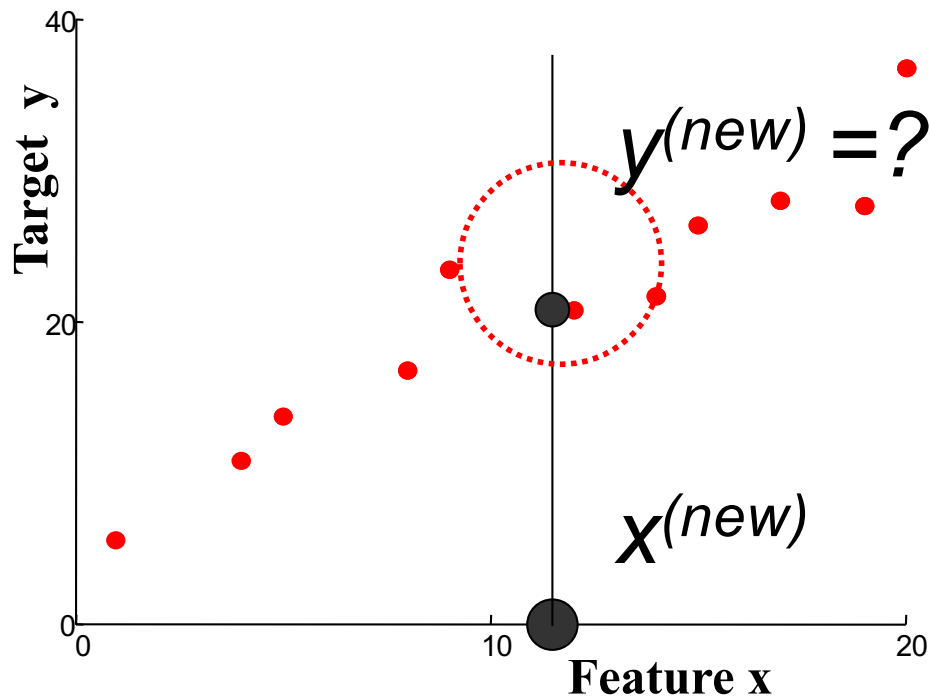


Regression; Scatter plots



- Suggests a relationship between x and y
- Regression: given new observed $x^{(new)}$, estimate $y^{(new)}$

Nearest neighbor regression

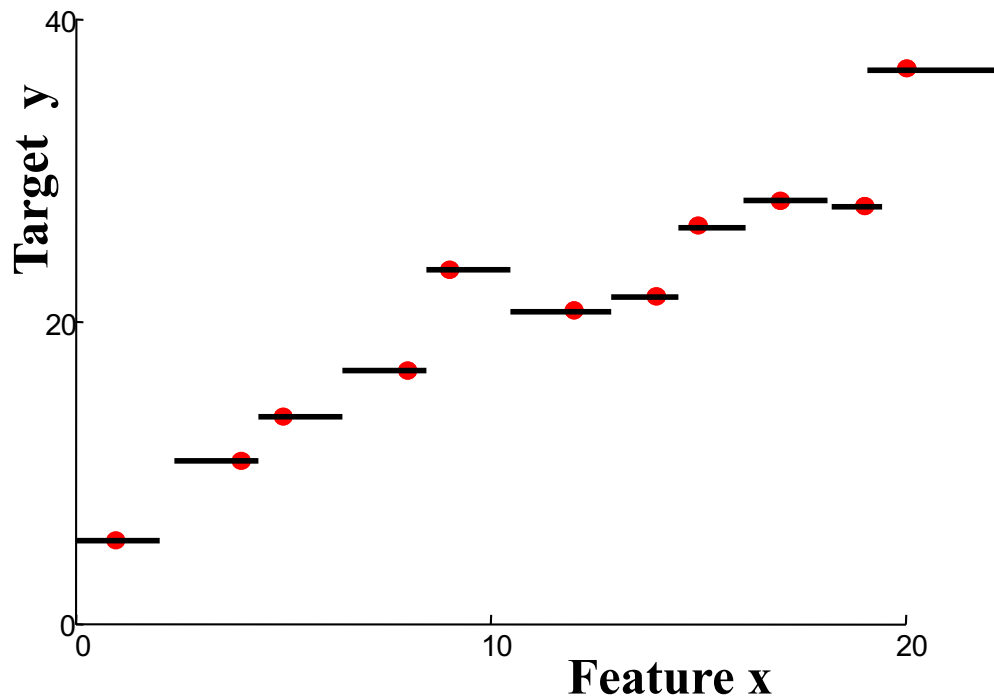


“Predictor”:

Given new features:
Find nearest example
Return its value

- Find training datum $x^{(i)}$ closest to $x^{(new)}$; predict $y^{(i)}$

Nearest neighbor regression

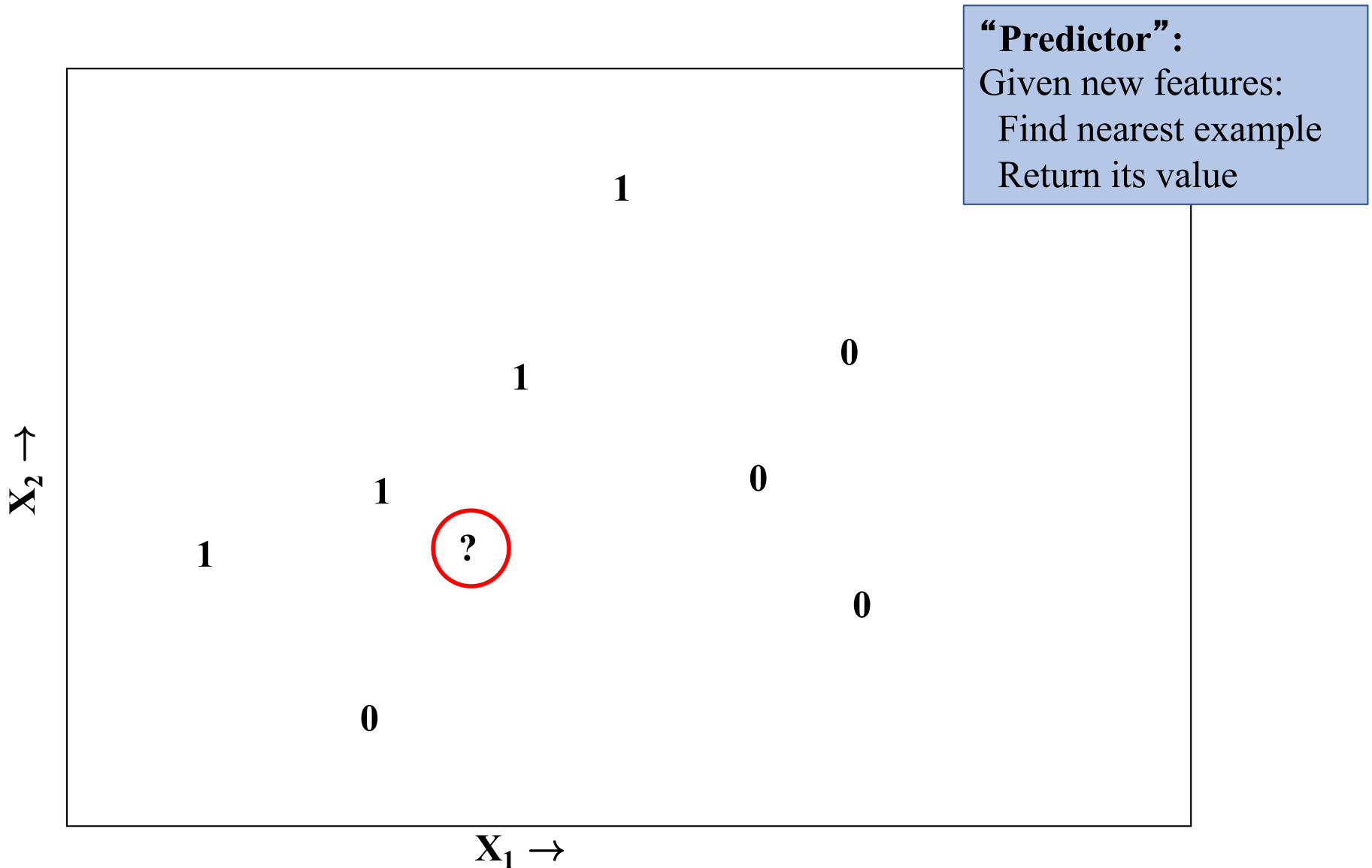


“Predictor”:

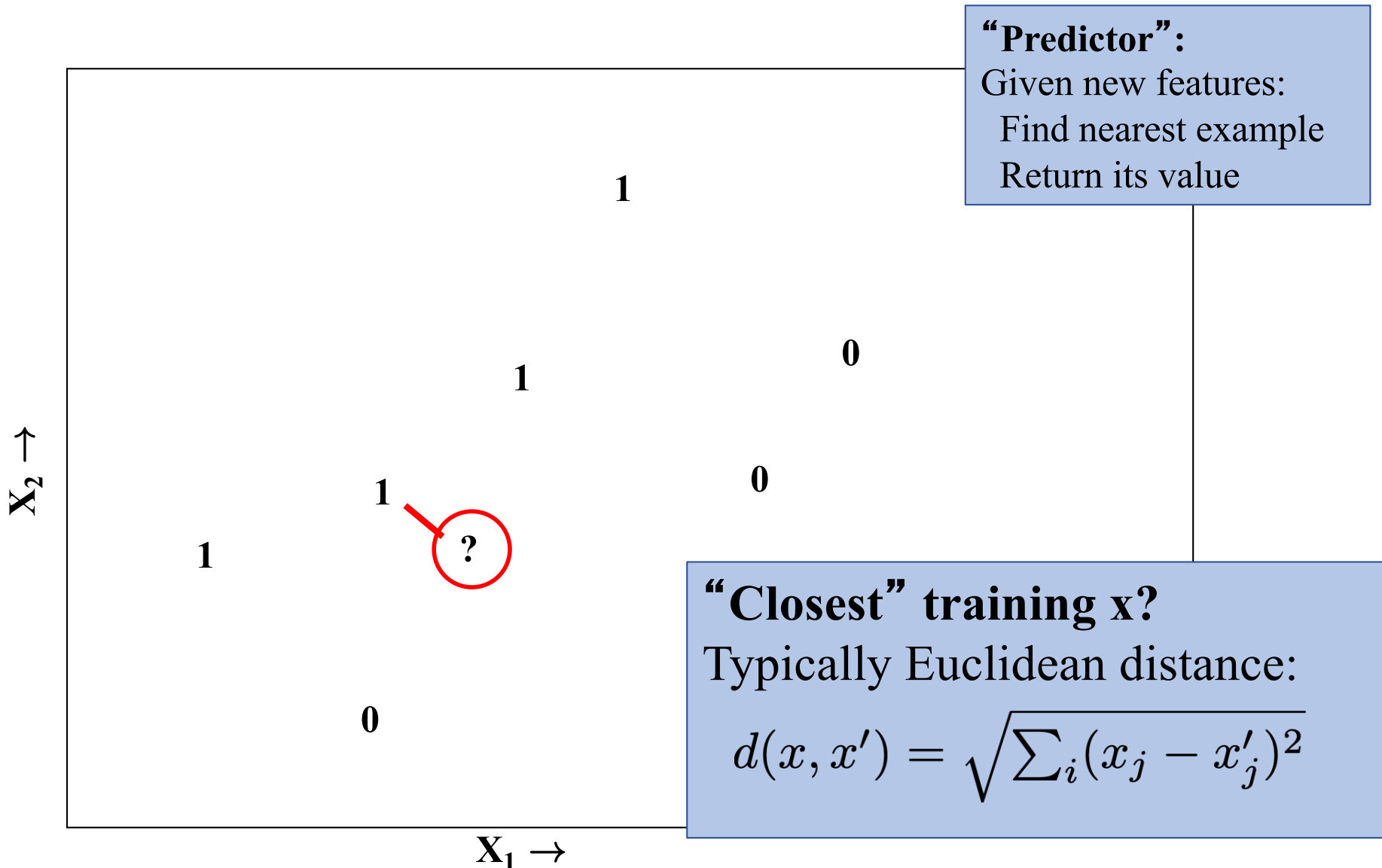
Given new features:
Find nearest example
Return its value

- Find training datum $x^{(i)}$ closest to $x^{(new)}$; predict $y^{(i)}$
- Defines an (implicit) function $f(x)$
- “Form” is piecewise constant

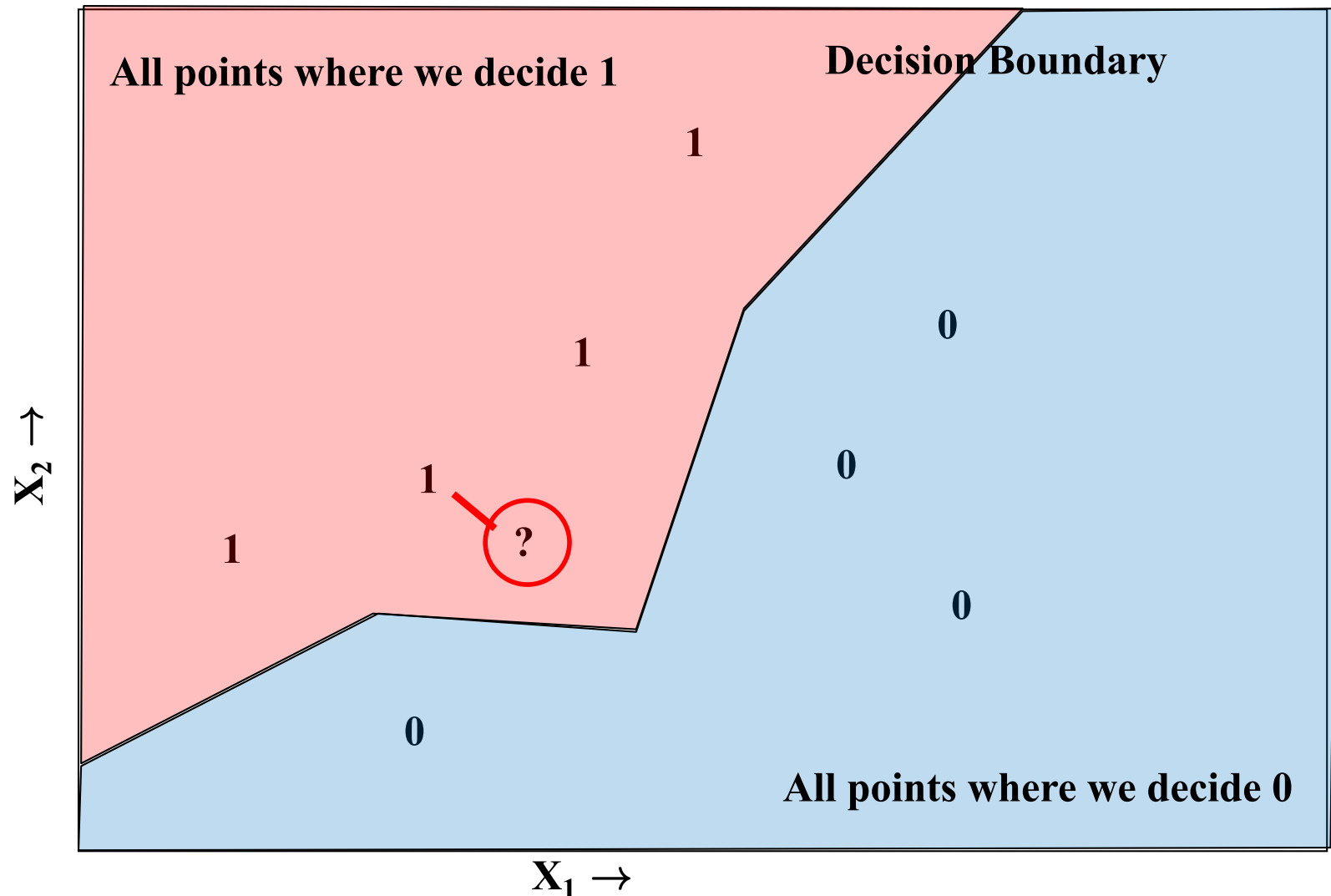
Nearest neighbor classifier



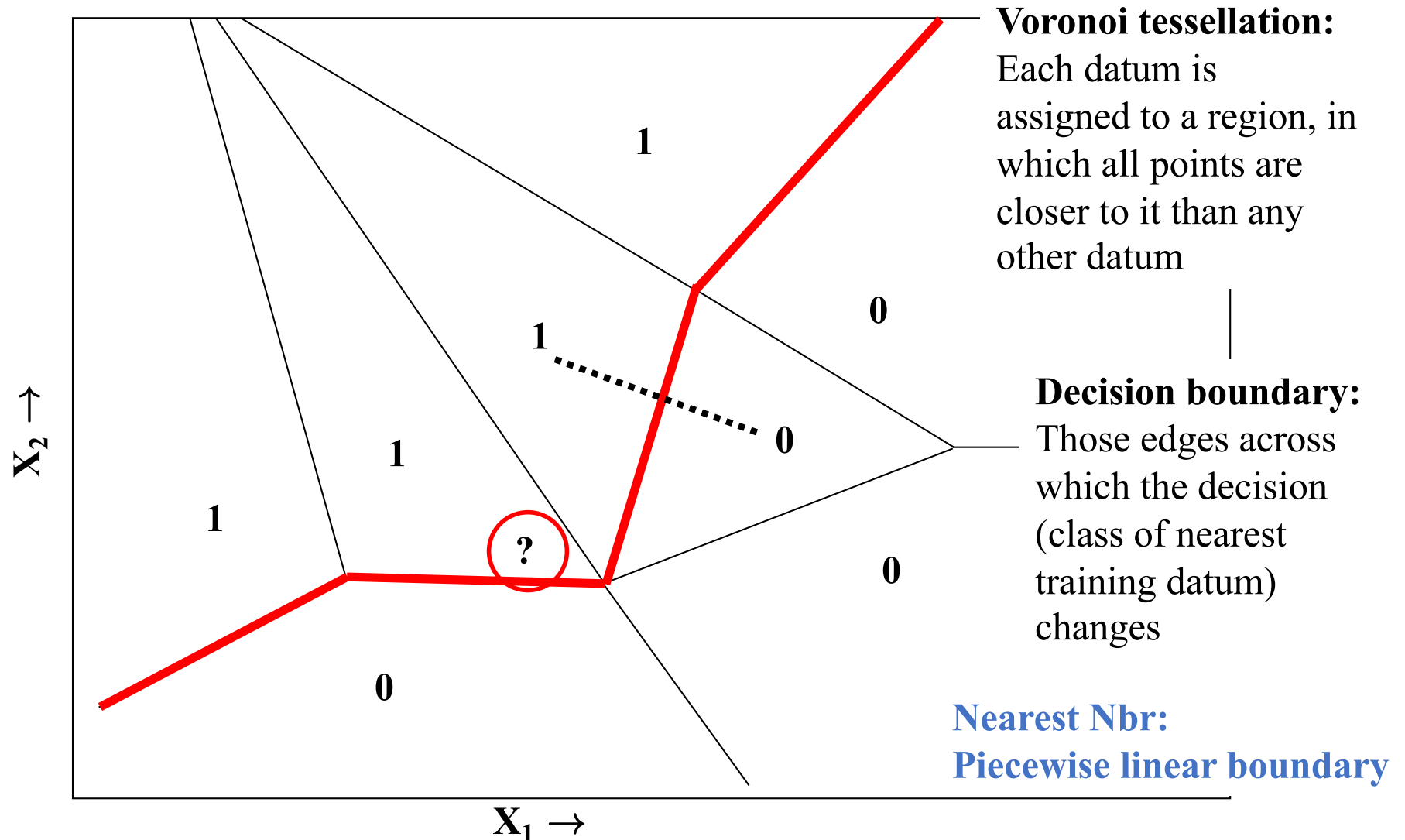
Nearest neighbor classifier



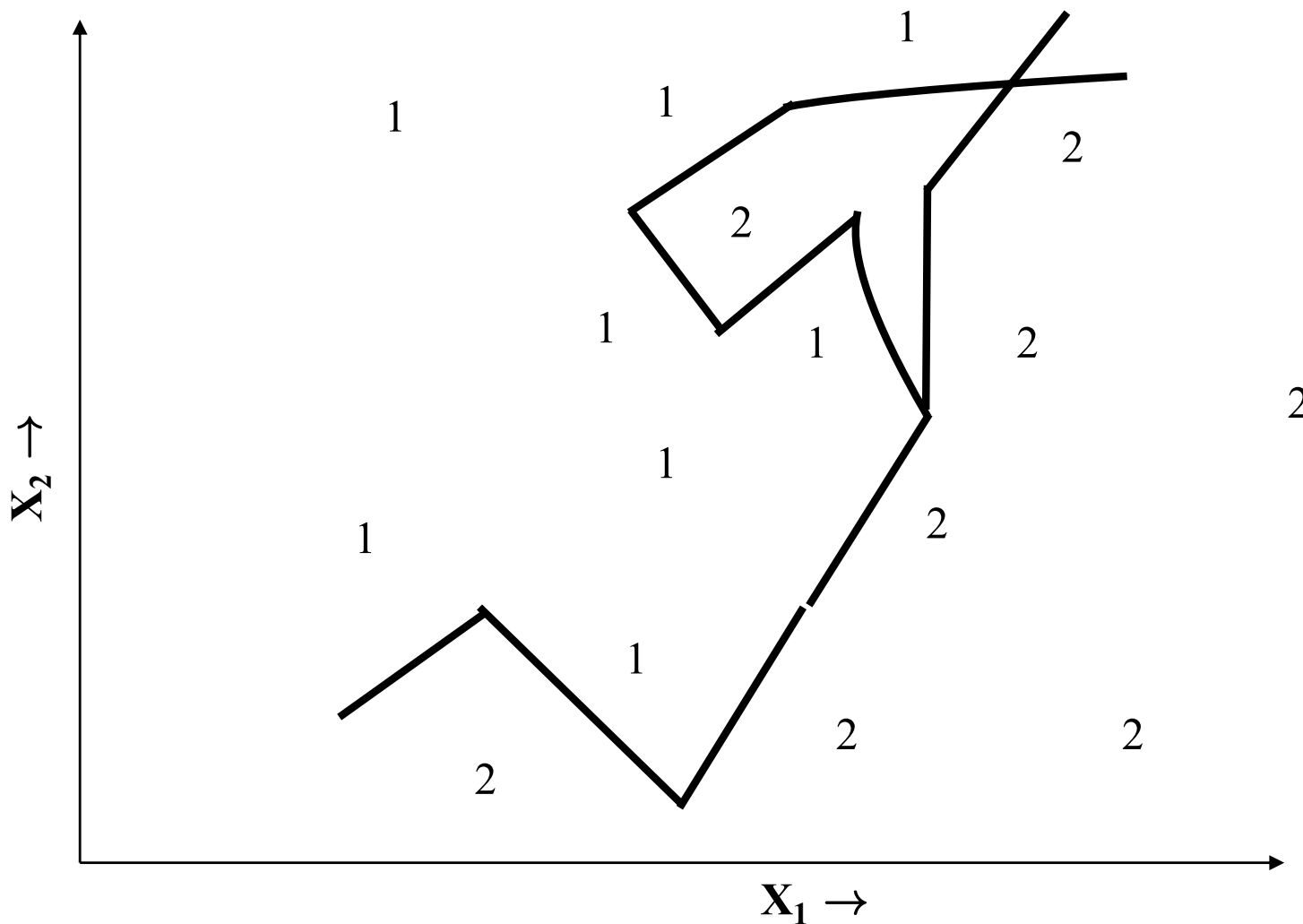
Nearest neighbor classifier



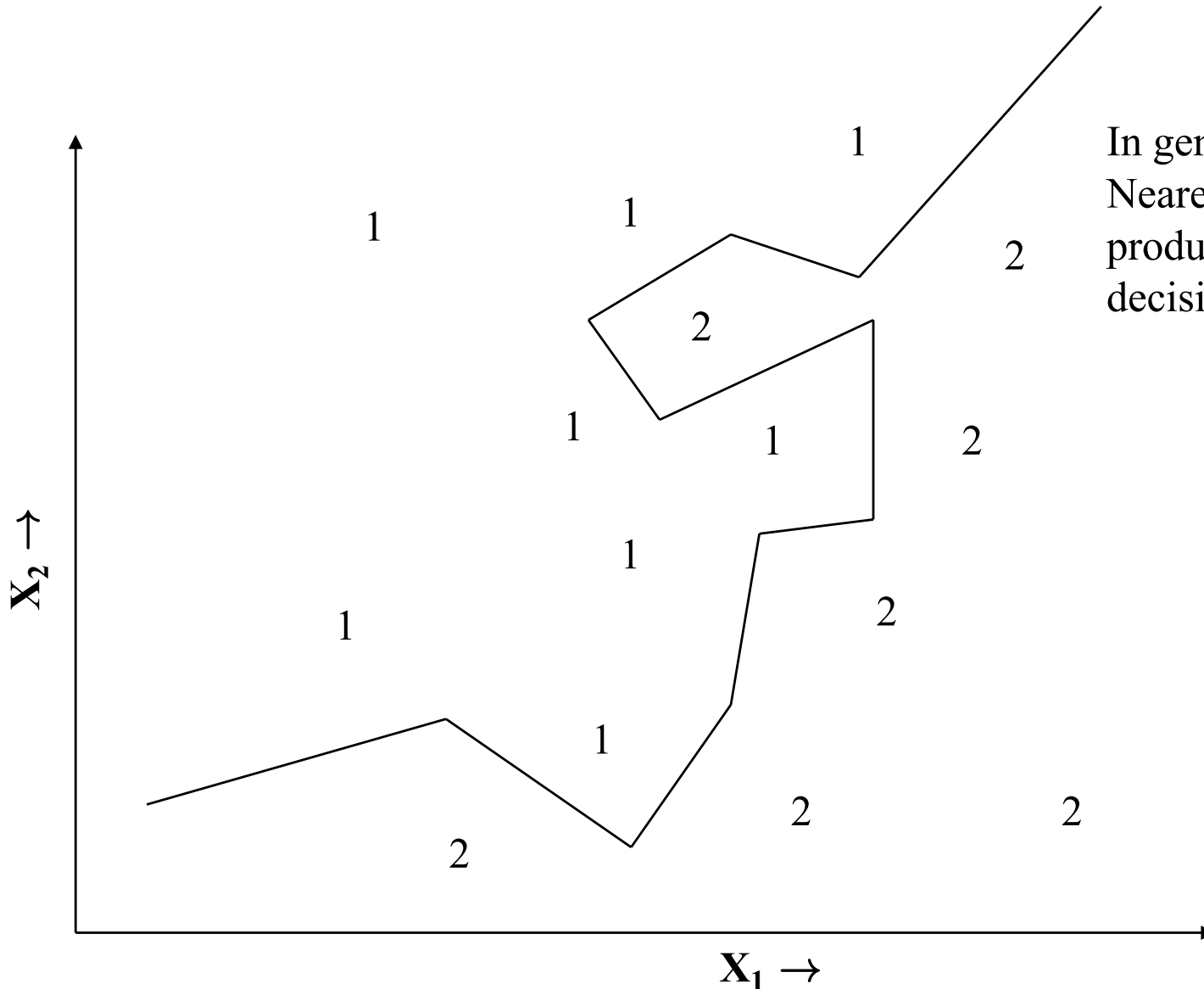
Nearest neighbor classifier



More Data Points



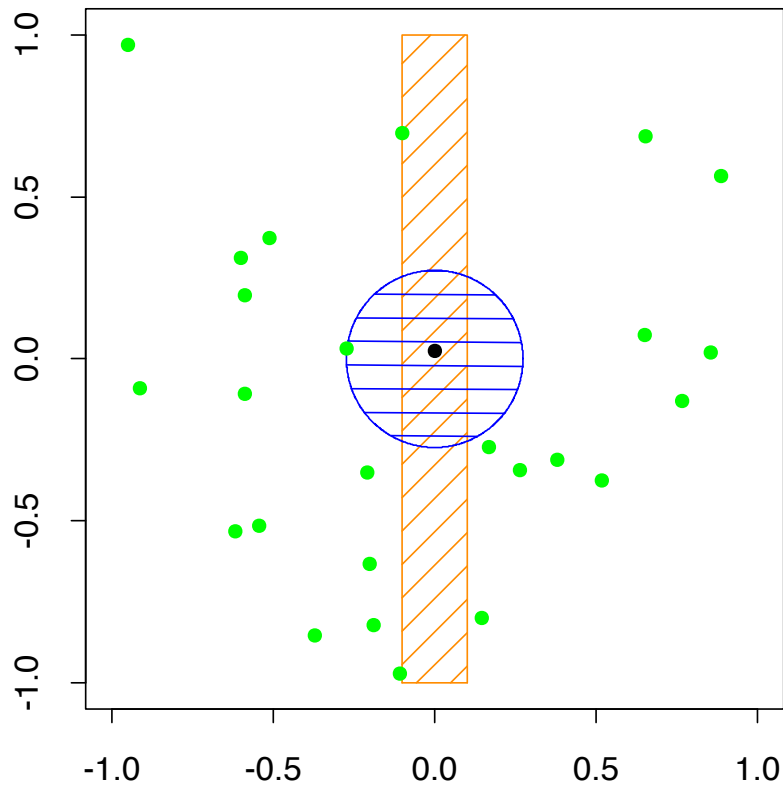
More Complex Decision Boundary



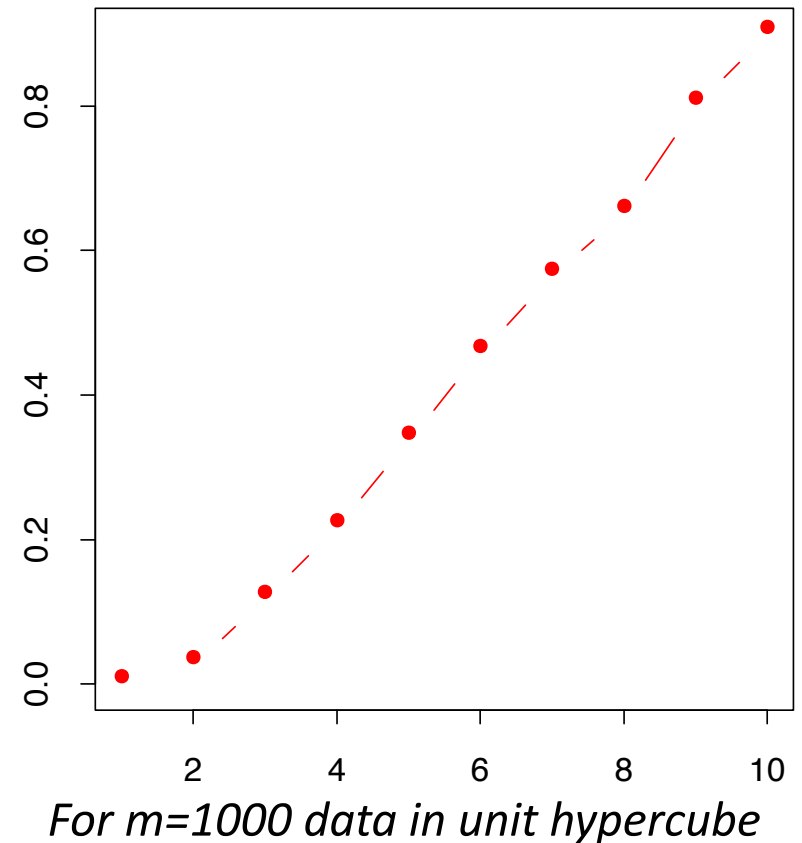
In general:
Nearest-neighbor classifier
produces piecewise linear
decision boundaries

Issue: Neighbor Distance & Dimension

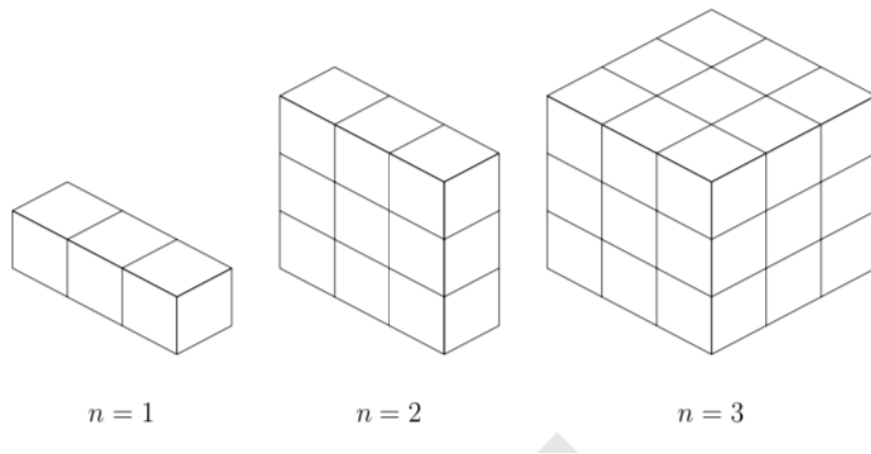
1-NN in One vs. Two Dimensions



Distance to 1-NN vs. Dimension

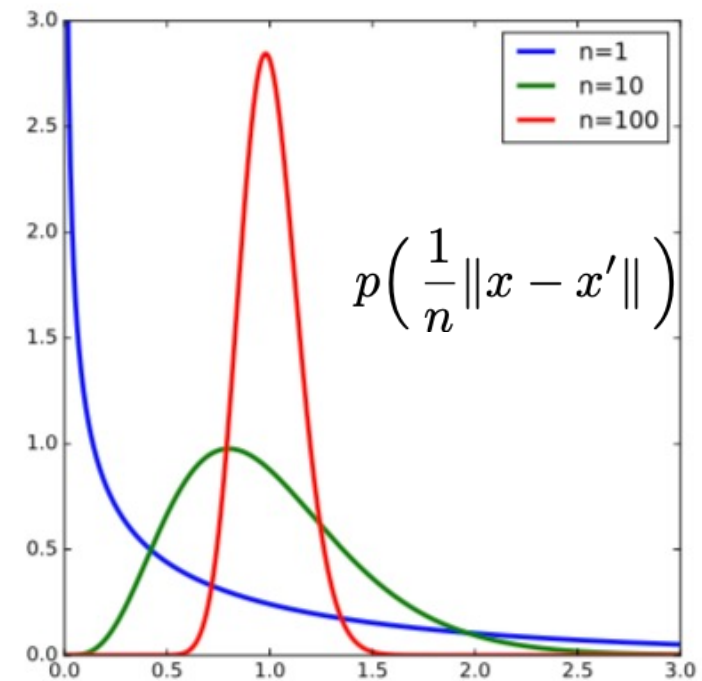


The “Curse of Dimensionality”



- Function is “smooth”?
 - Want a training example that is close by (epsilon)
 - How many data do we need?

- How far away are the other data points?
 - In higher dimension, “almost all” data are equally far away!



Machine Learning

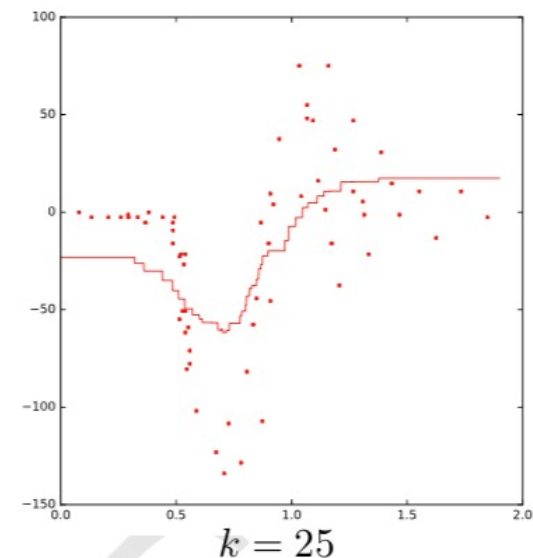
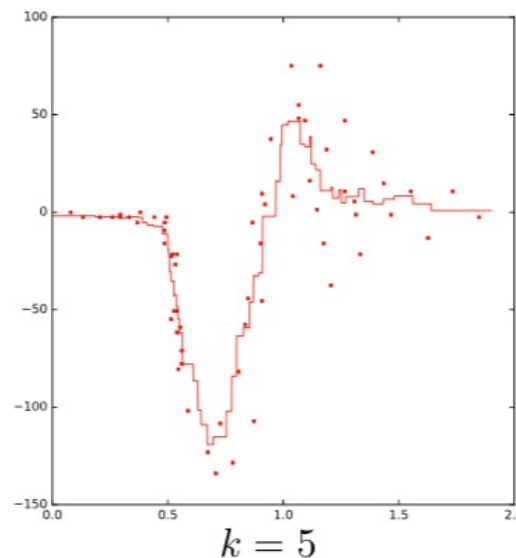
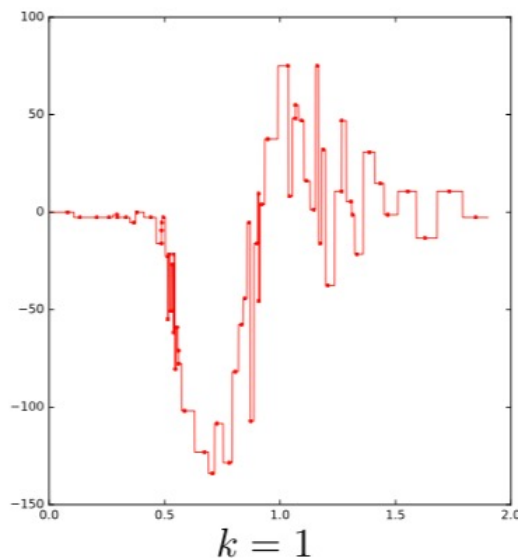
Complexity and Overfitting

Nearest Neighbors

K-Nearest Neighbors

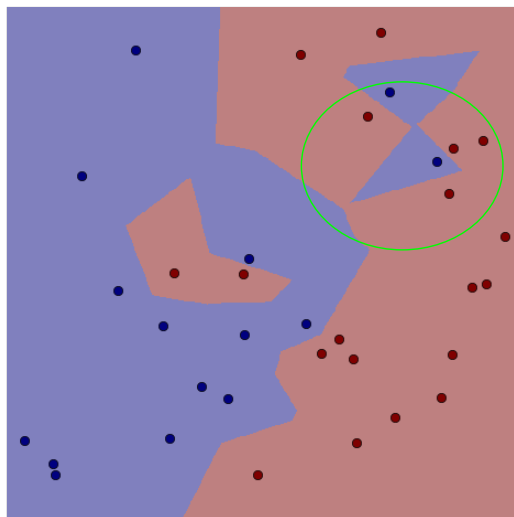
K-Nearest Neighbor (kNN) Predictor

- 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 have smallest distance to \underline{x}
- Regression
 - Ranking gives k closest examples and their target values “ y ”
 - Usually just average the y -values of the k closest training examples
 - Larger k = average over a larger area for prediction



K-Nearest Neighbor (kNN) Predictor

- 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 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”)
 - Note: for two-class problems, if k is odd ($k=1, 3, 5, \dots$) there will never be any “ties”; otherwise, just use (any) tie-breaking rule

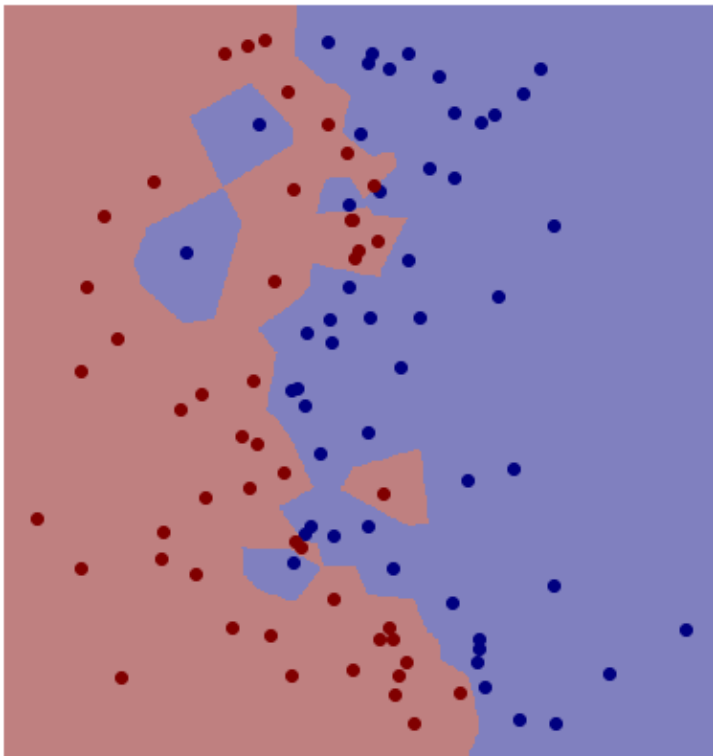


Predict blue?
Or just noise?

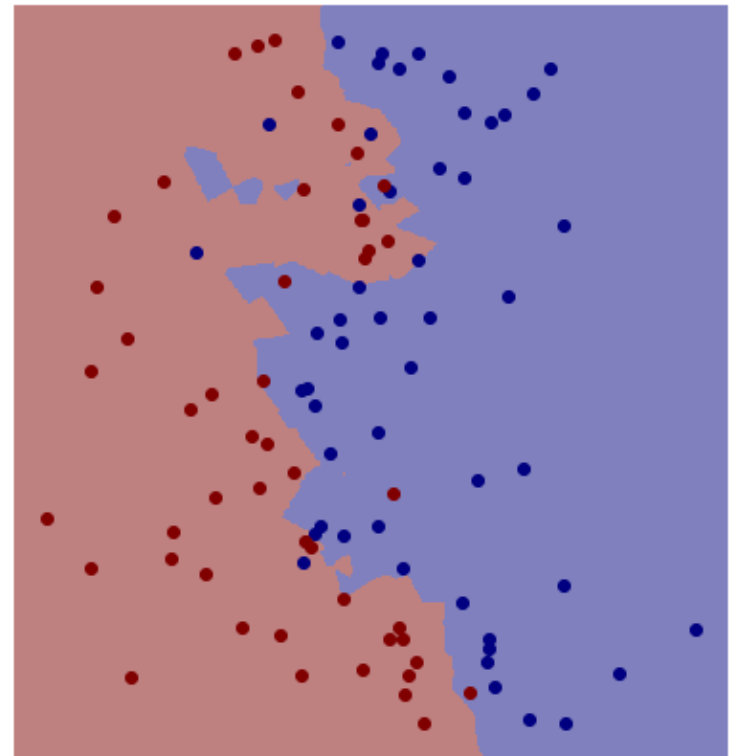
kNN Decision Boundary

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

$K = 1$



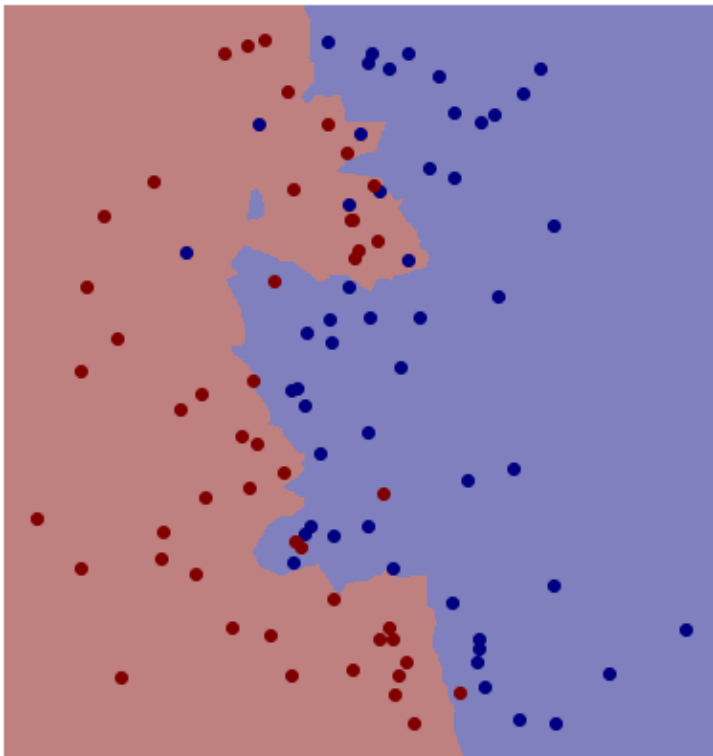
$K = 3$



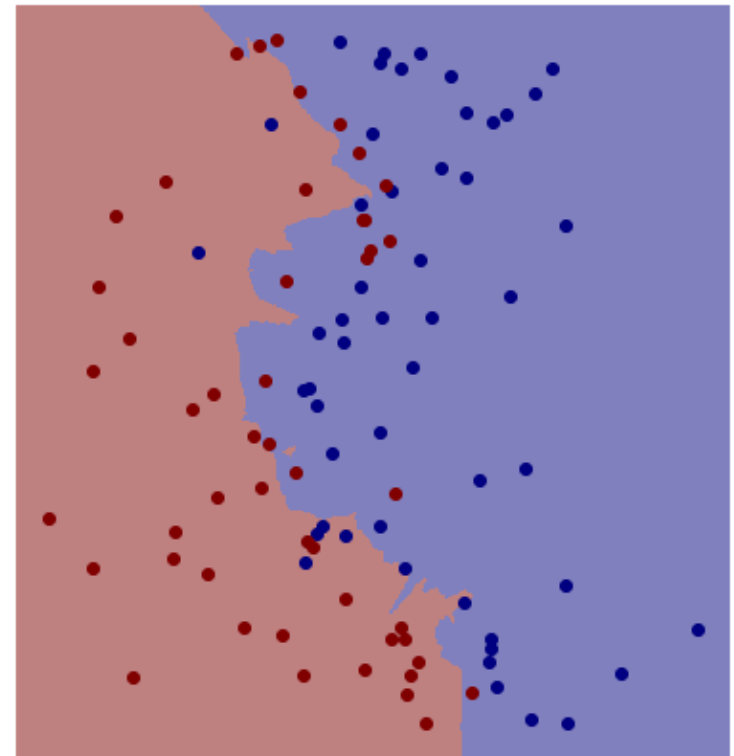
kNN Decision Boundary

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

$K = 5$



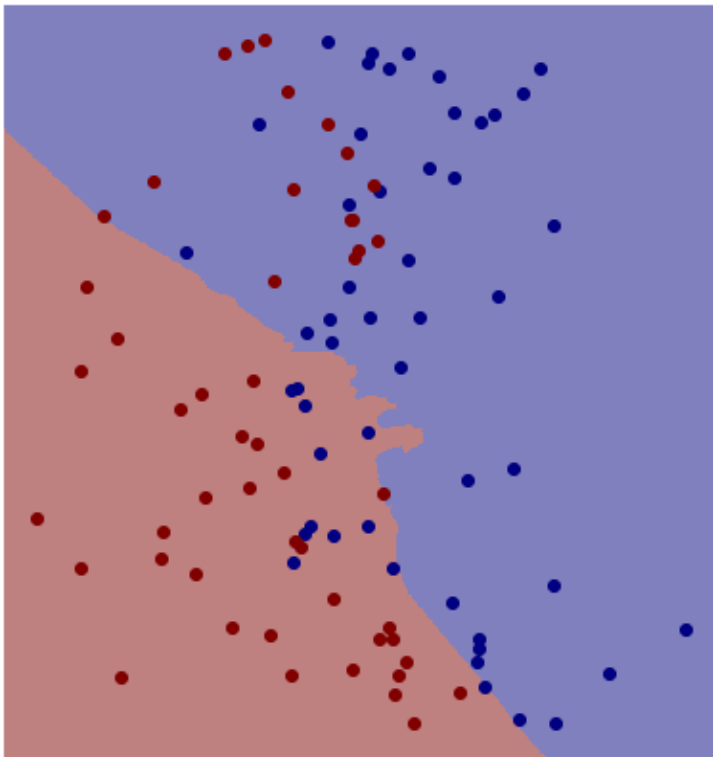
$K = 20$



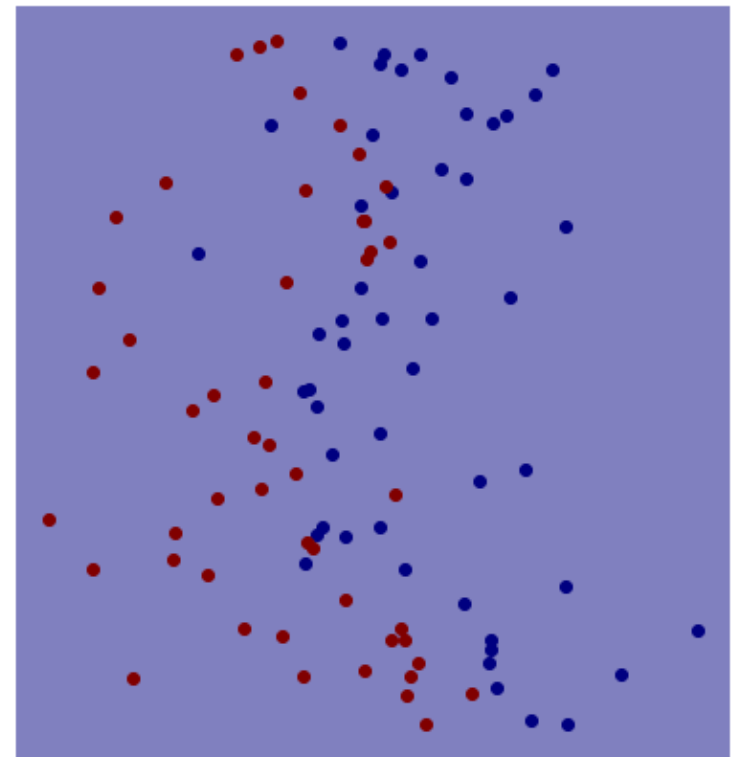
kNN Decision Boundary

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

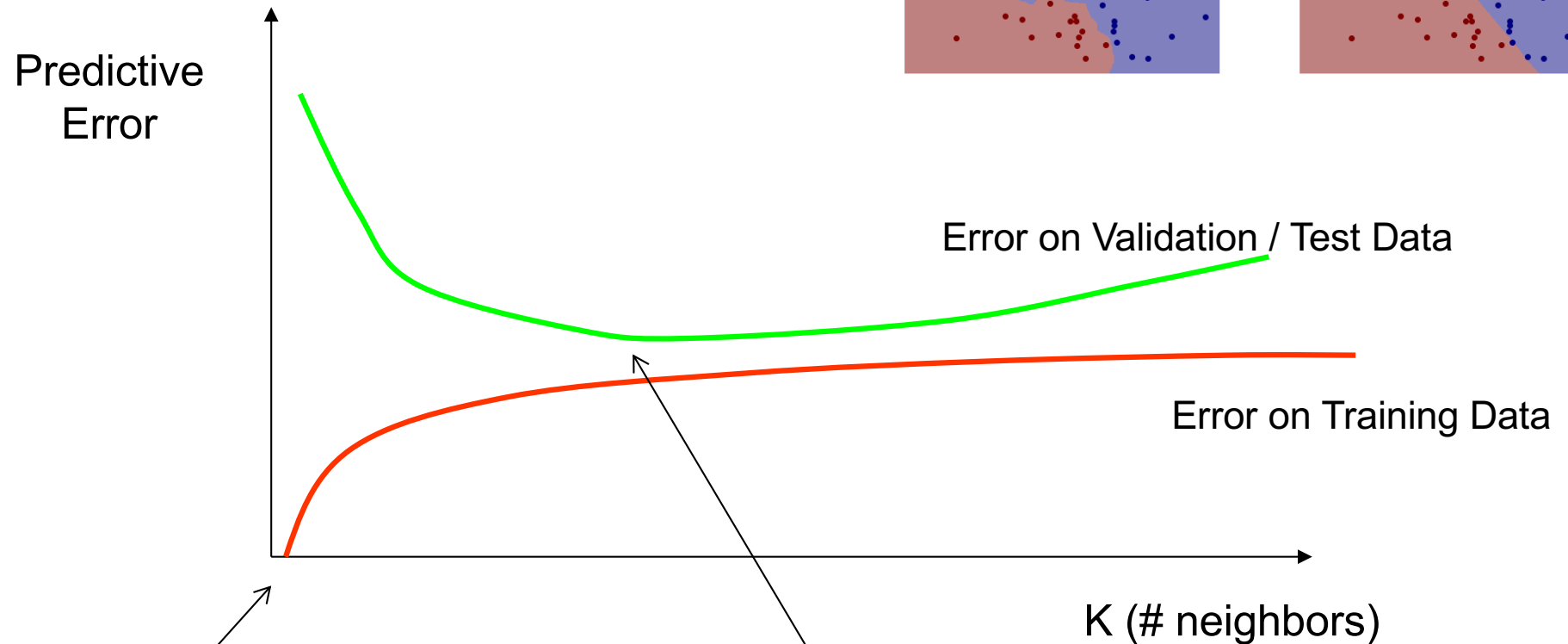
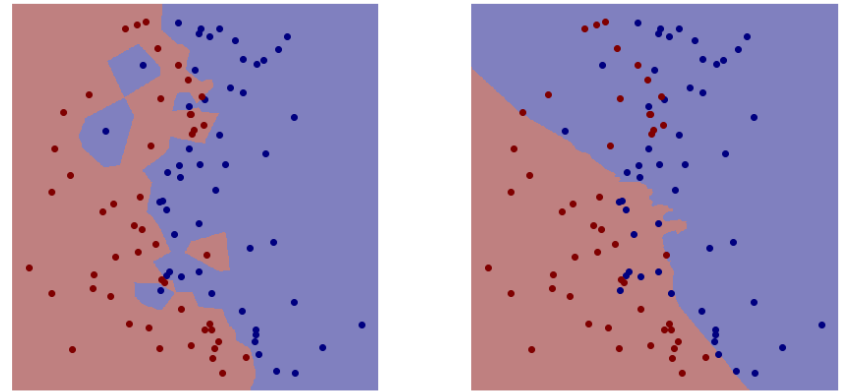
$K = 70$



$K = 100 (=m)$



Error rates and K

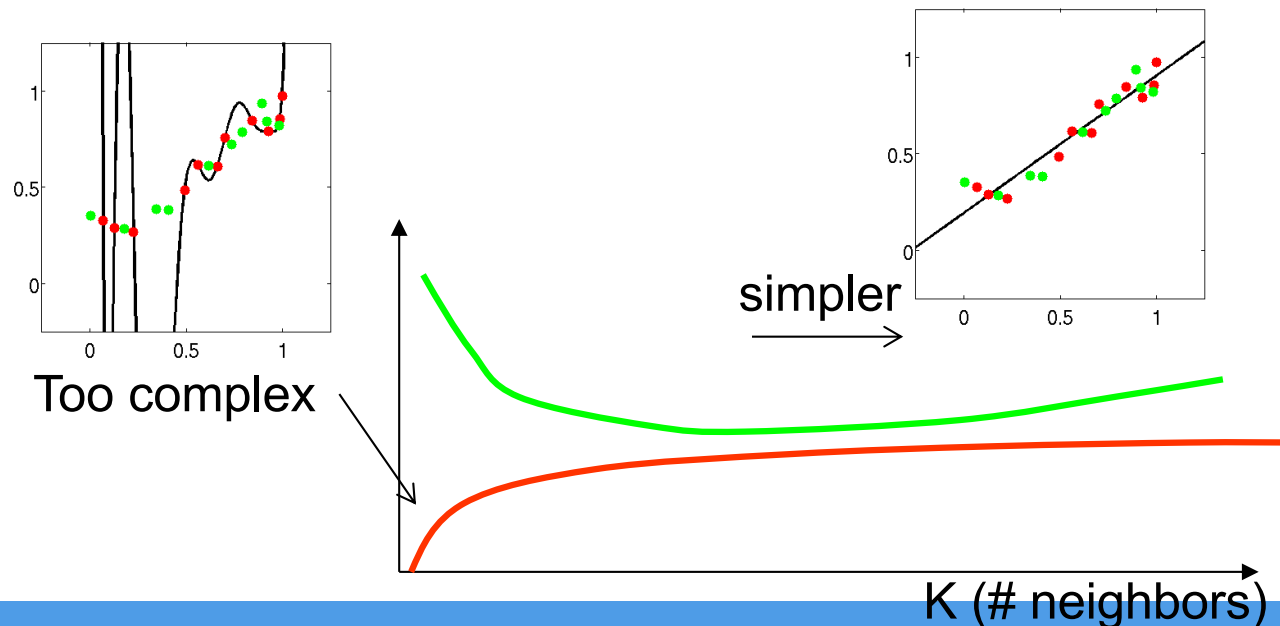


K=1? Zero error!
Training data have been memorized...

Best value of K

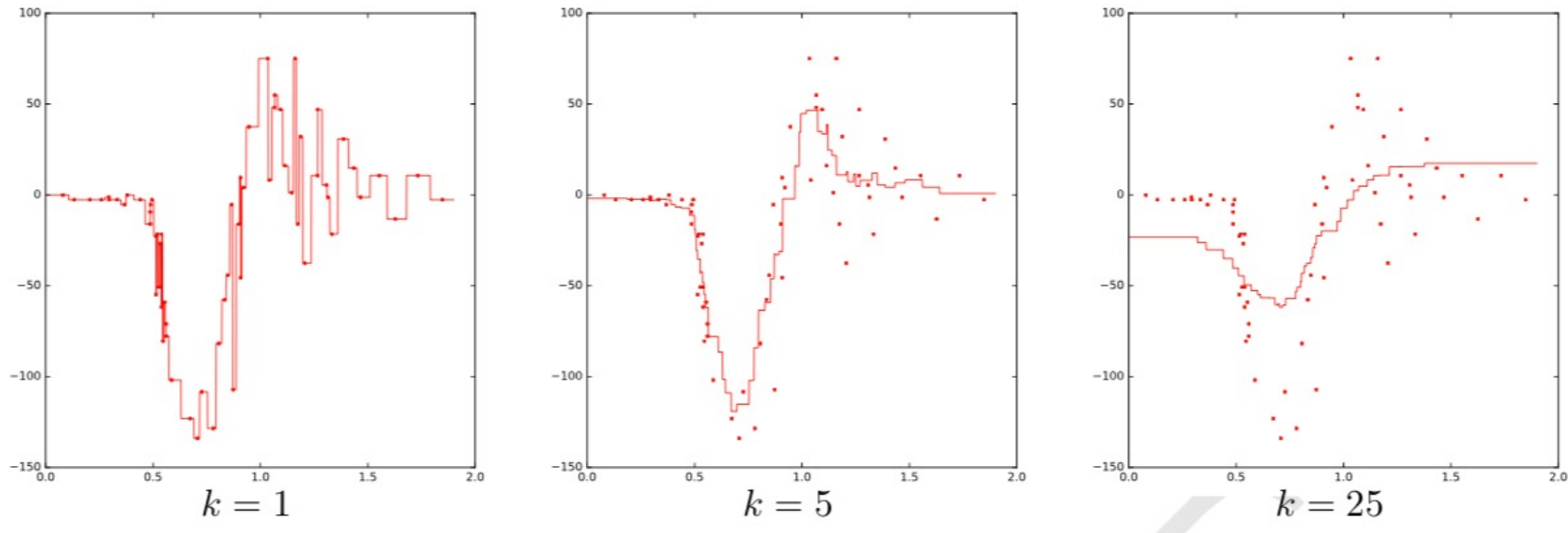
Complexity & Overfitting

- Complex model predicts all training points well
- Doesn't generalize to new data points
- $k = 1$: perfect memorization of examples (complex)
- $k = m$: always predict majority class in dataset (simple)
- Can select k using validation data, etc.



Regression example

- Similar behavior for regression predictions:



- K (& data density) defines a local area to average over
- K too small: follow “noise” in the data
- K too big: smooth over too large an area

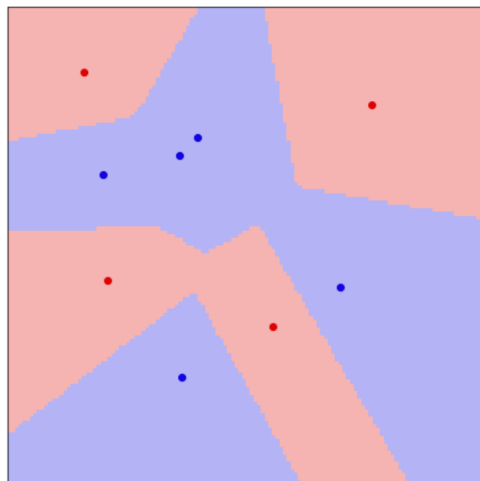
Live Demo

- Via Binder:

<https://mybinder.org/v2/gh/ihler/ml-demos/HEAD?filepath=notebooks%2FDemo-Live-KNN.ipynb>

(can be slow to start...)

- Ex:



Left click to add points, right click to remove nearest point.

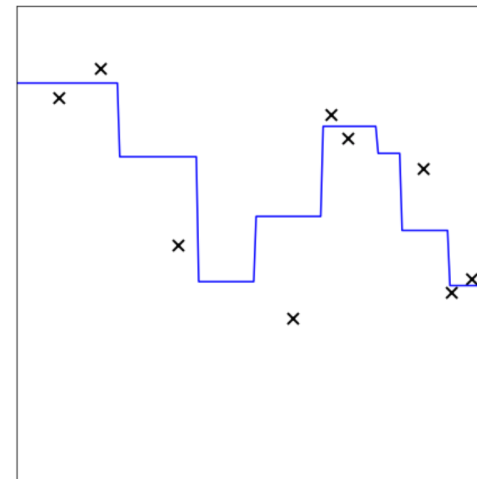
Use shift/alt/ctrl combinations to control class value when placing points.

Set Options:

K 1

☐ Weighted?
 $w^{(s)} \propto \exp(-\alpha \|x - x^{(s)}\|^2)$

alpha 0.500



Left click to add points, right click to remove nearest point.

Set Options:

K 2

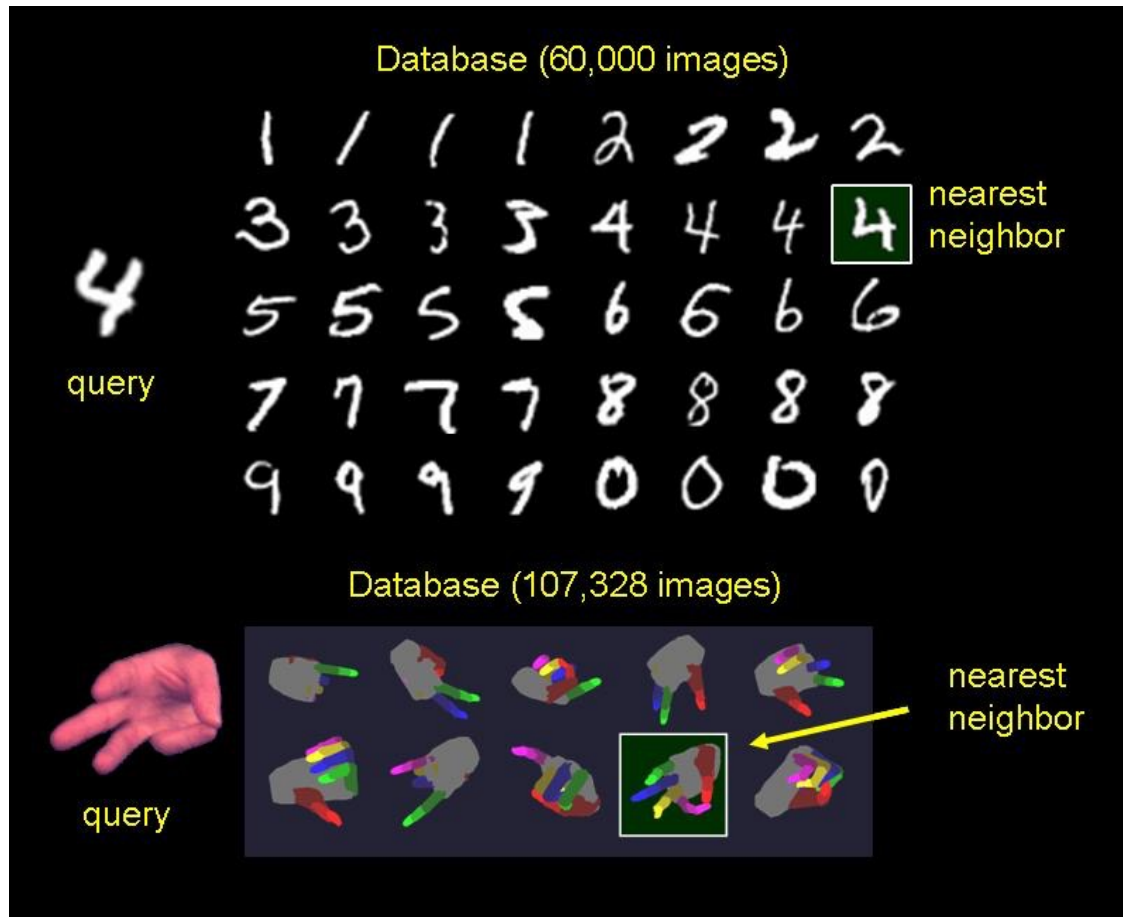
☐ Weighted?
 $w^{(s)} \propto \exp(-\alpha \|x - x^{(s)}\|^2)$

alpha 0.500

K-Nearest Neighbor (kNN) Classifier

- Theoretical Considerations
 - as k increases
 - we are averaging over more neighbors
 - the effective decision boundary is more “smooth”
 - as m increases, the optimal k value tends to increase
 - $k=1$, m increasing to infinity : error $< 2 \times$ optimal
- Extensions of the Nearest Neighbor classifier
 - Weighted distances $d(x, x') = \sqrt{\sum_i w_i (x_i - x'_i)^2}$
 - e.g., some features may be more important; others may be irrelevant
 - Fast search techniques (indexing) to find k -nearest points in d -space
 - Weighted average / voting based on distance

Digit & Hand Gesture Recognition



Athitsos et al., CVPR 2004 & PAMI 2008

Tricks to build efficient & accurate nearest-neighbor classifiers:

- Gather large training sets (possibly by generating synthetic data)
- Engineer clever distance functions that are invariant to aspects of the data unrelated to the class label
- Use algorithms to find (approximate) nearest neighbors in sub-linear time (locality sensitive hashing, class-sensitive embeddings, KD-trees, etc.)

Summary

- K-nearest neighbor models
 - Classification (vote)
 - Regression (average or weighted average)
- Piecewise linear decision boundary
 - How to calculate
- Test data and overfitting
 - Model “complexity” for knn
 - Use validation data to estimate test error rates & select k