UCI University of California, Irvine

CS 273A: Machine Learning

Fall 2021

Lecture 2: Nearest Neighbors

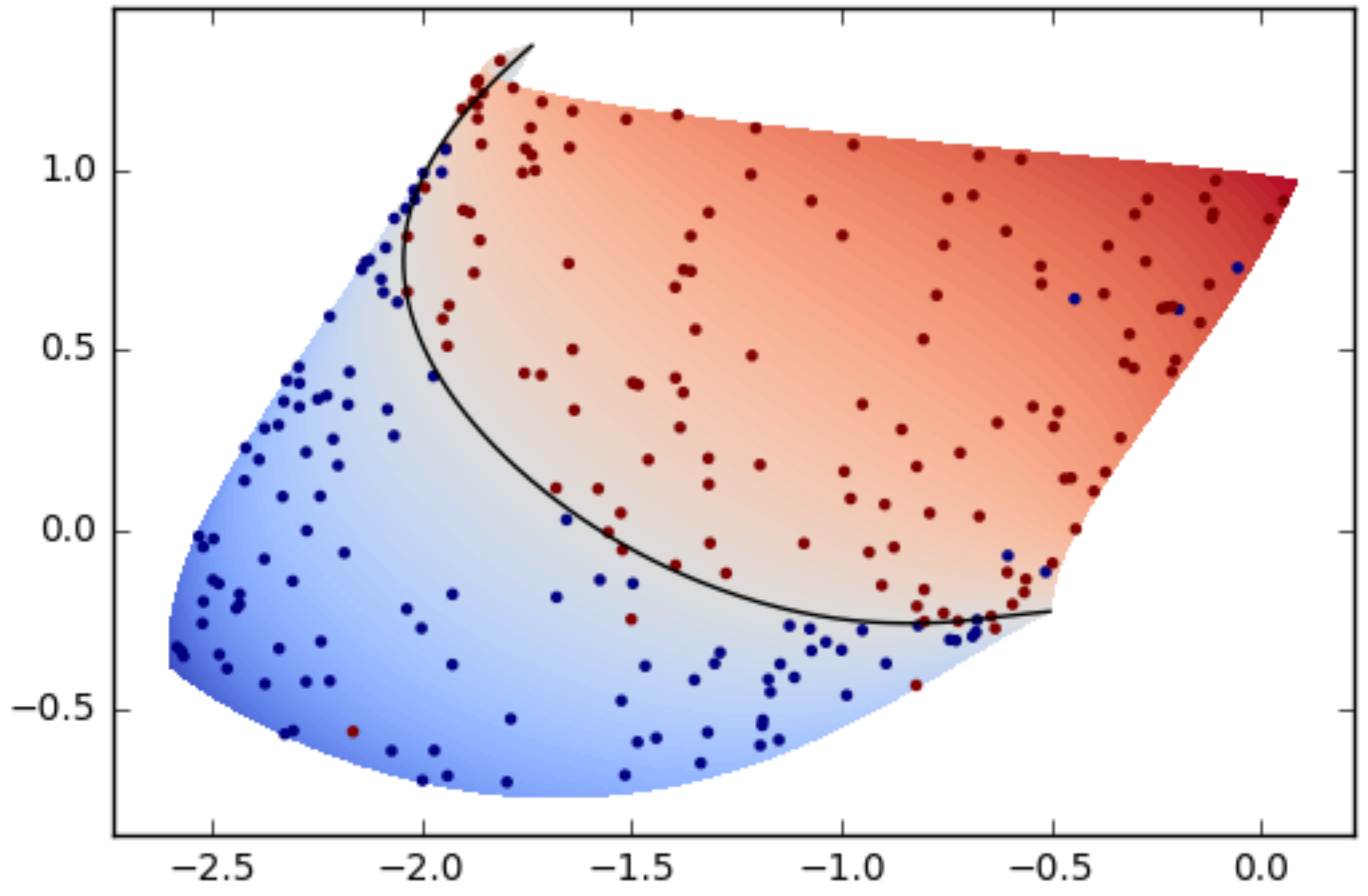
Roy Fox

Department of Computer Science

Bren School of Information and Computer Sciences

University of California, Irvine

All slides in this course adapted from Alex Ihler & Sameer Singh



Logistics

assignment 1

- Assignment 1 will be up soon
- Due: Tue, Oct 5 (Pacific)

project

- Project guidelines will resemble last year's

Today's lecture

Supervised learning

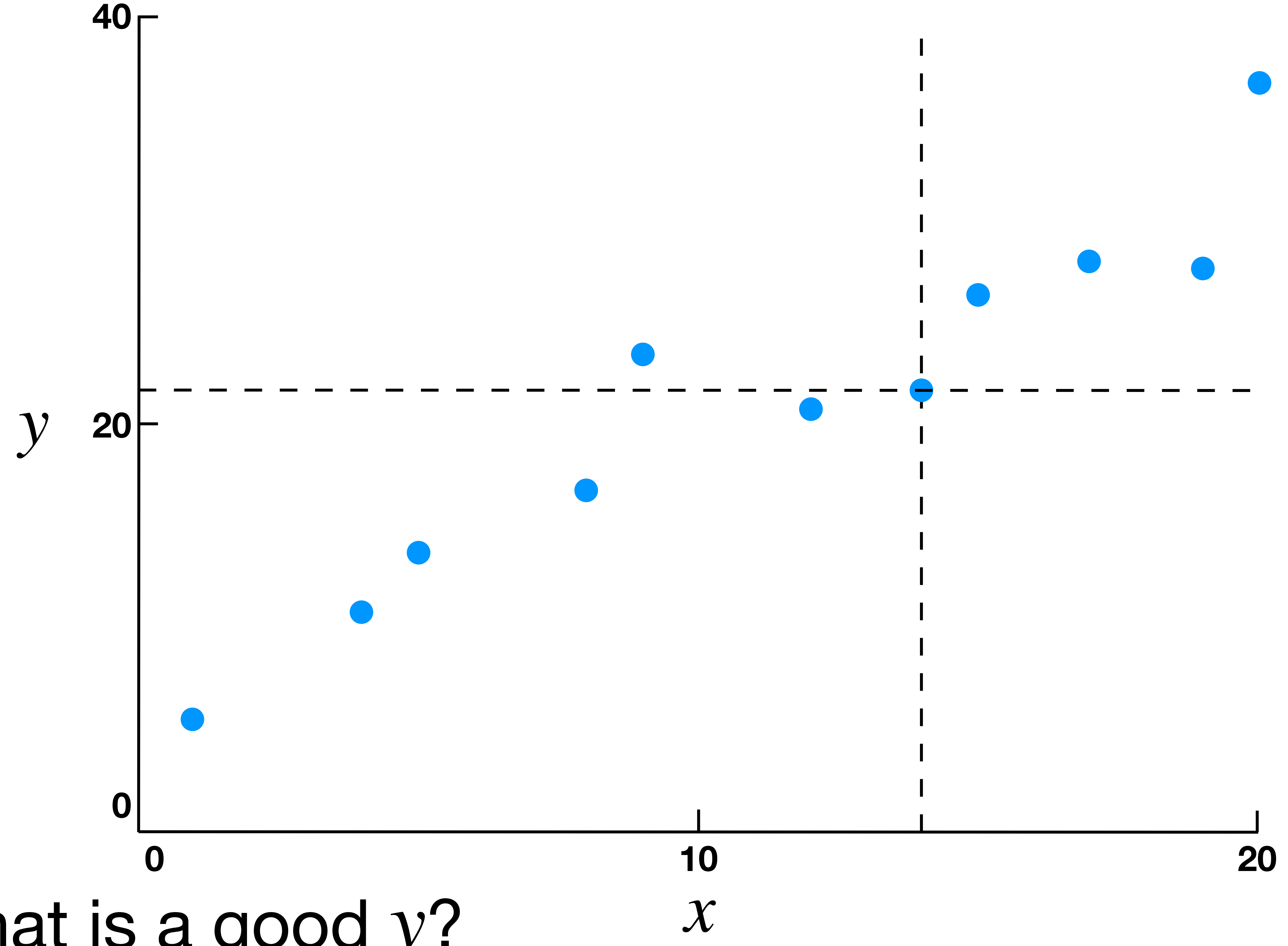
Nearest Neighbors

Overfitting and complexity

k-Nearest Neighbors

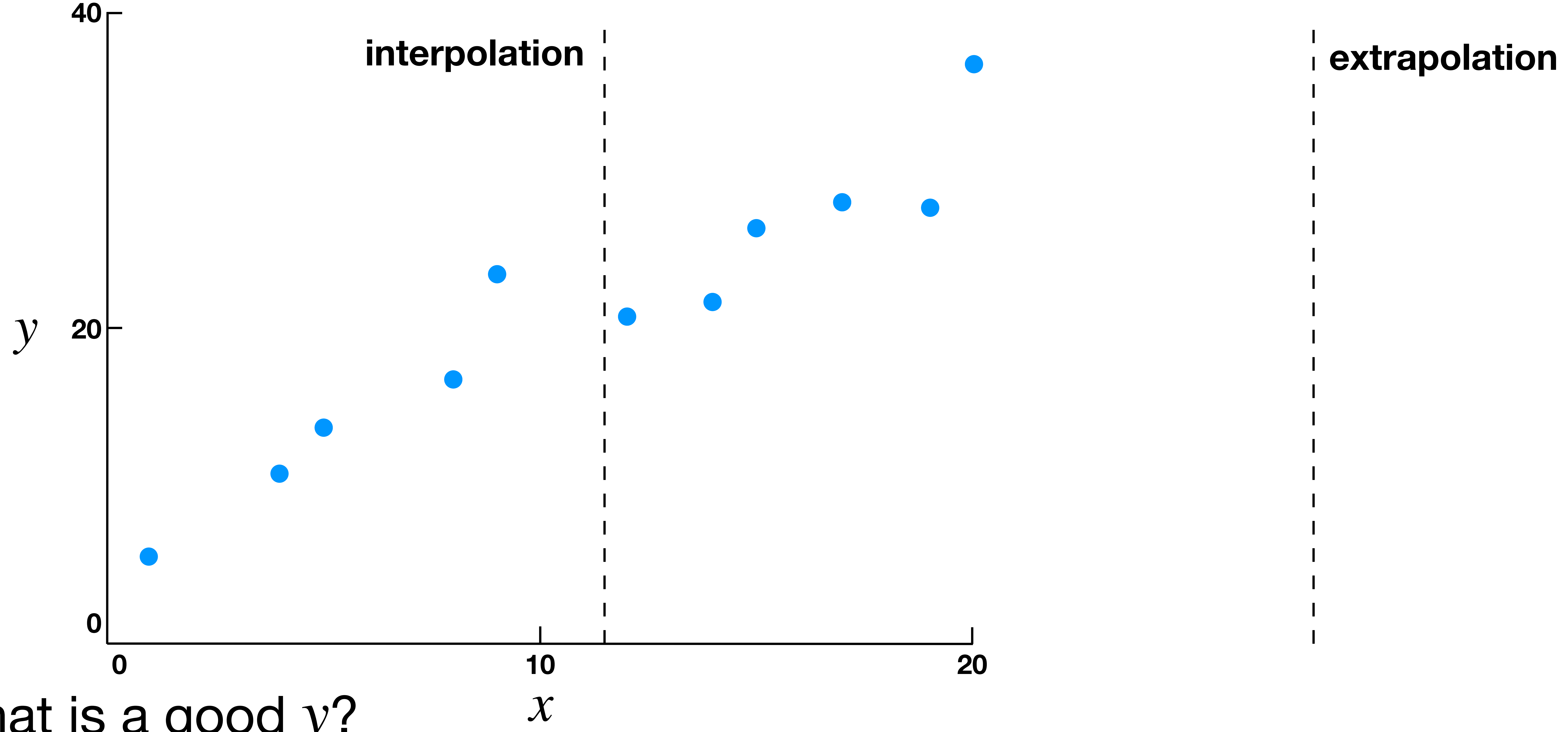
Supervised learning

- Data shows trend
- But also “noise”



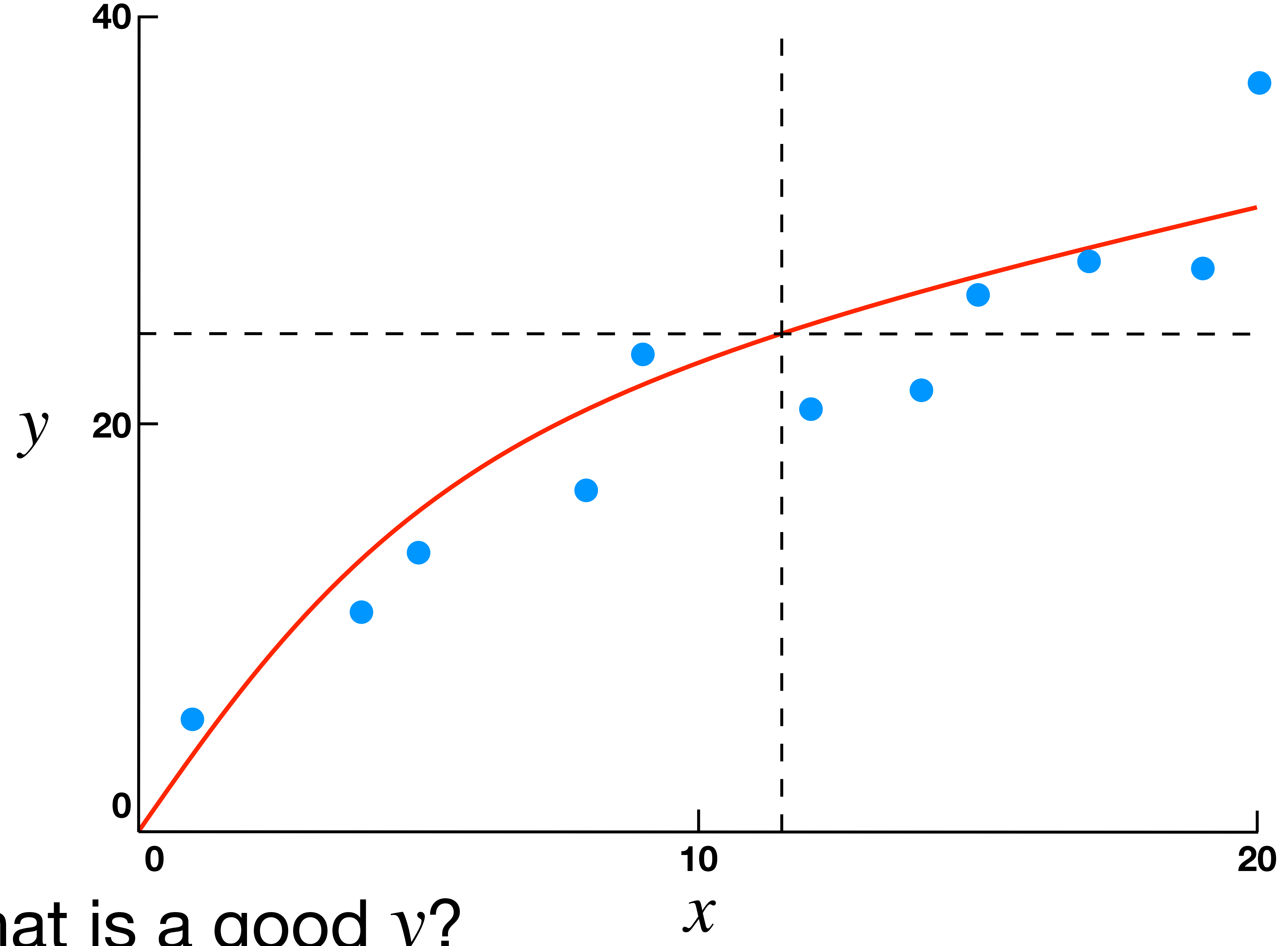
- Given some x, what is a good y?

Supervised learning



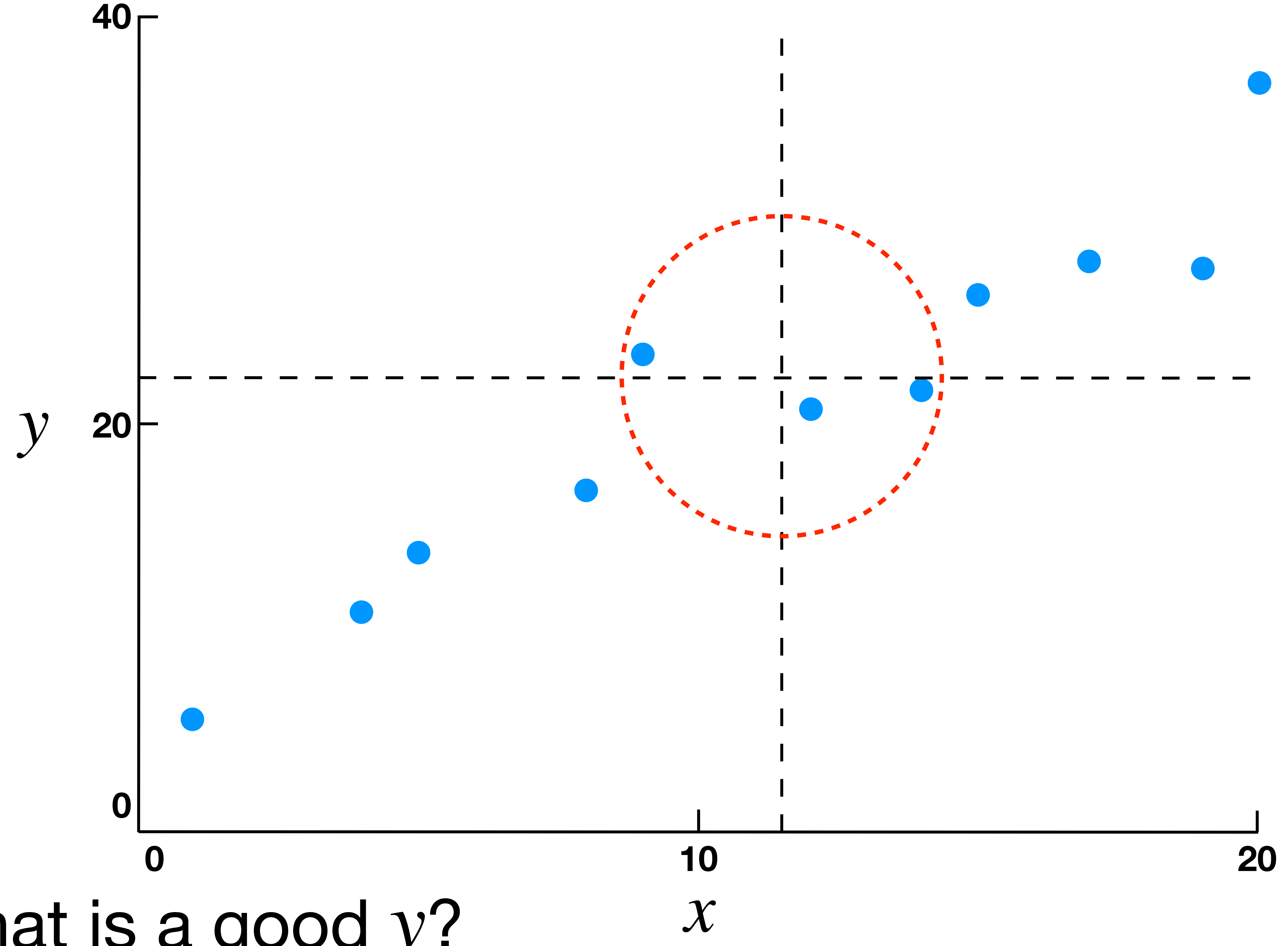
- Given some x, what is a good y?

Supervised learning



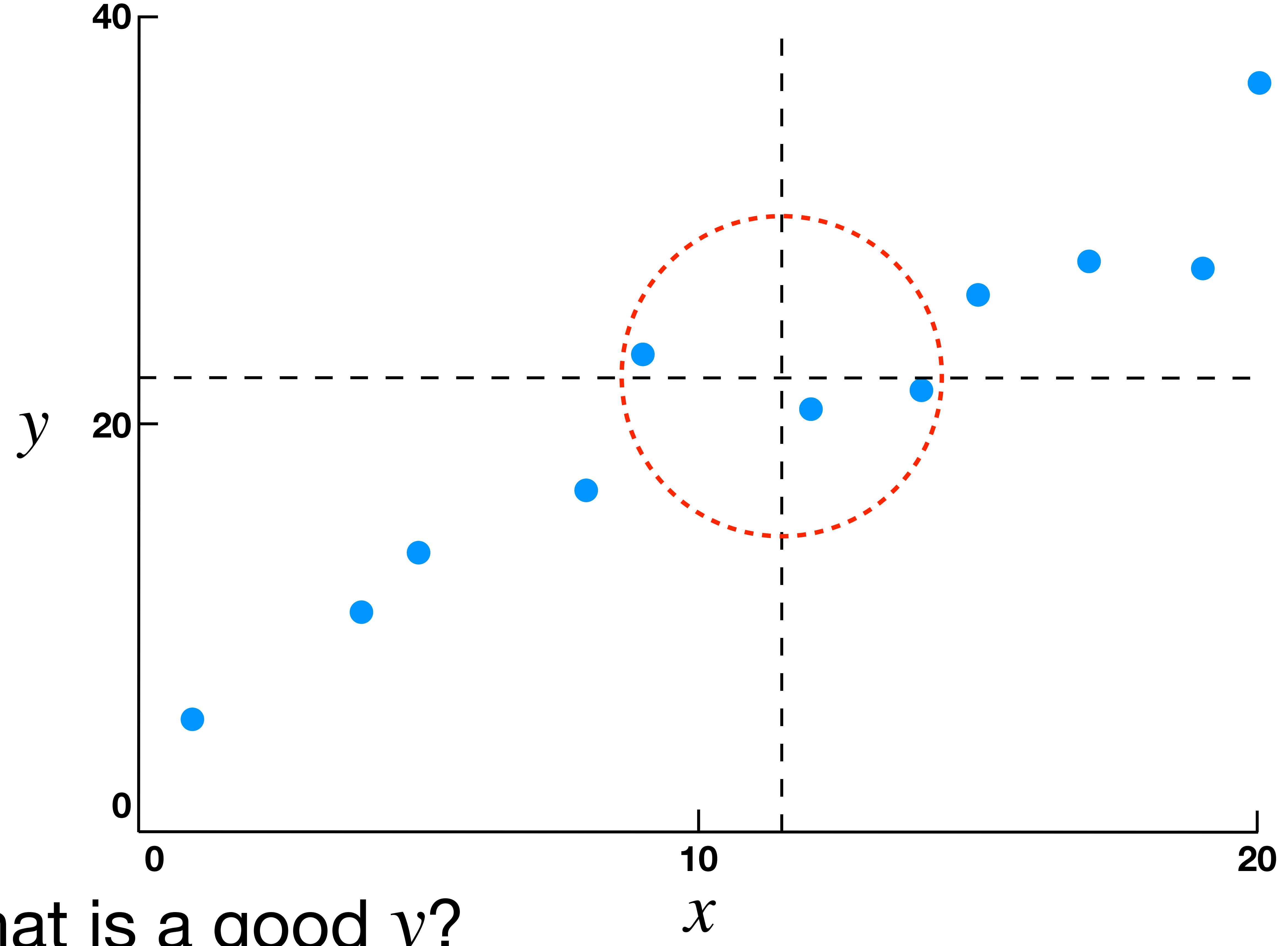
- Given some x, what is a good y?
  - Directly represent $f : x \mapsto y$

Supervised learning



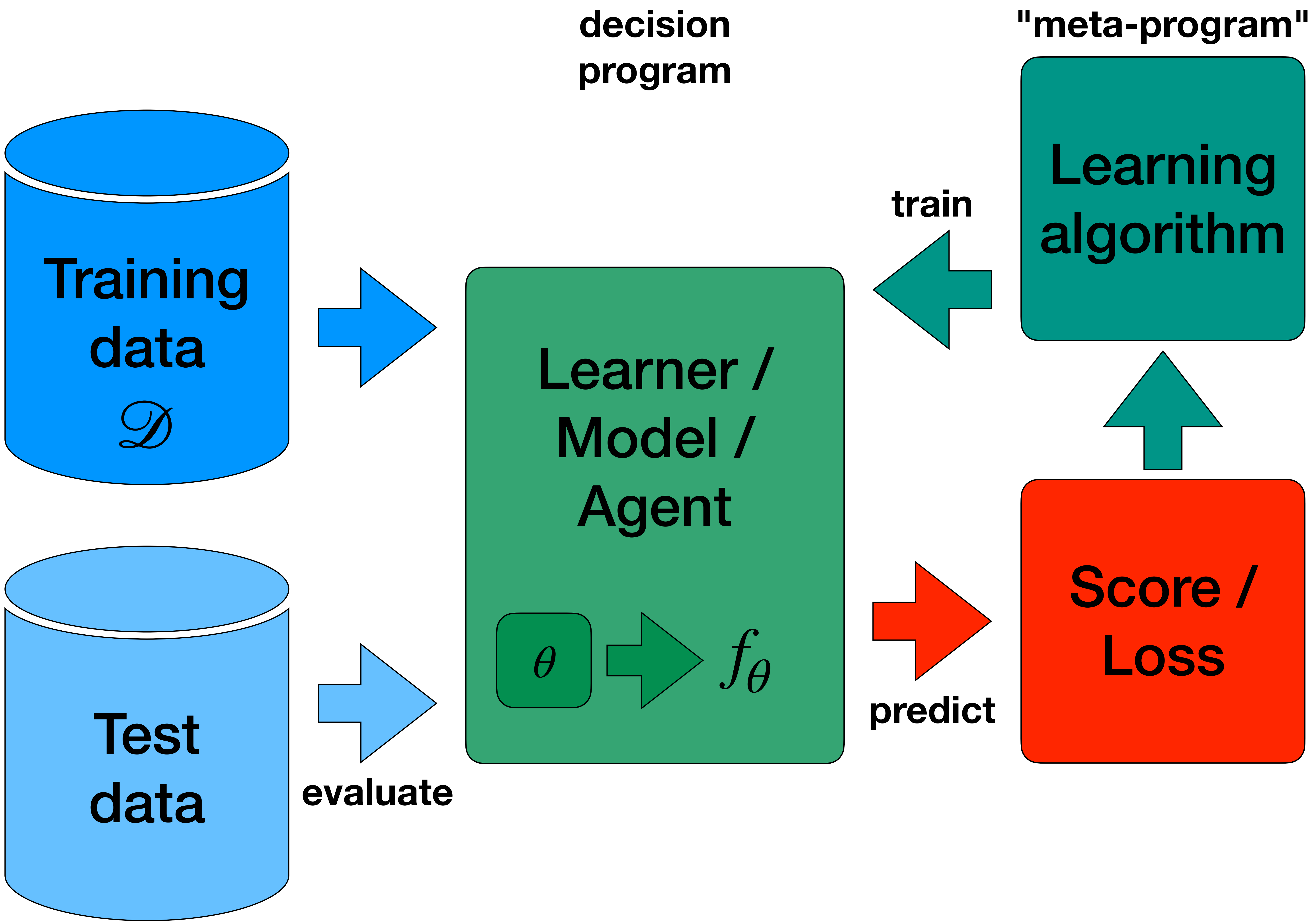
- Given some x, what is a good y?
  - Directly represent $f : x \mapsto y$
  - Average k nearest neighbors

Supervised learning



- Given some x, what is a good y?
  - Directly represent $f : x \mapsto y$
  - Average k nearest neighbors (k too large: missing trend; k too small: catching noise)

What is machine learning?



Today's lecture

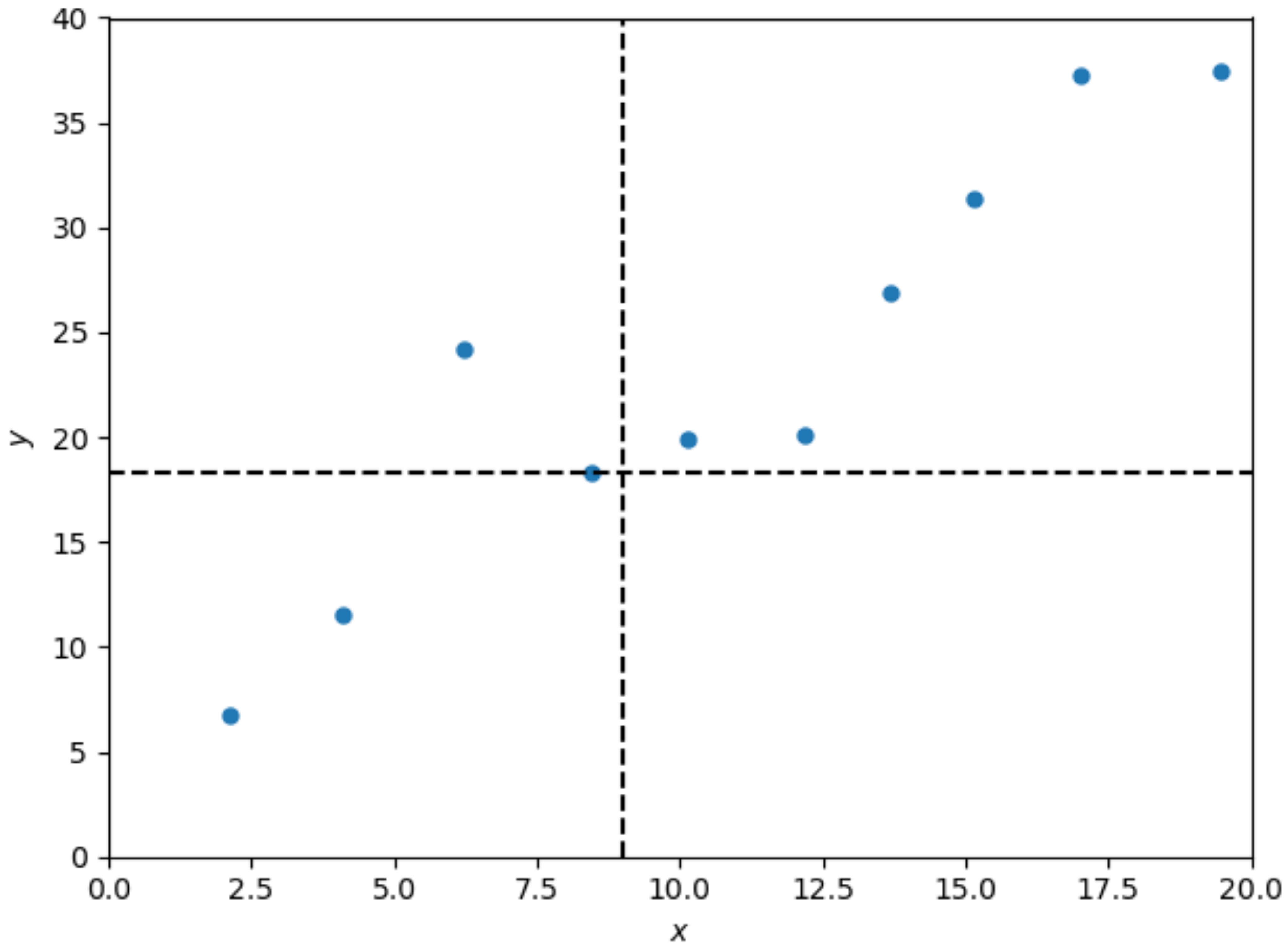
Supervised learning

Nearest Neighbors

Overfitting and complexity

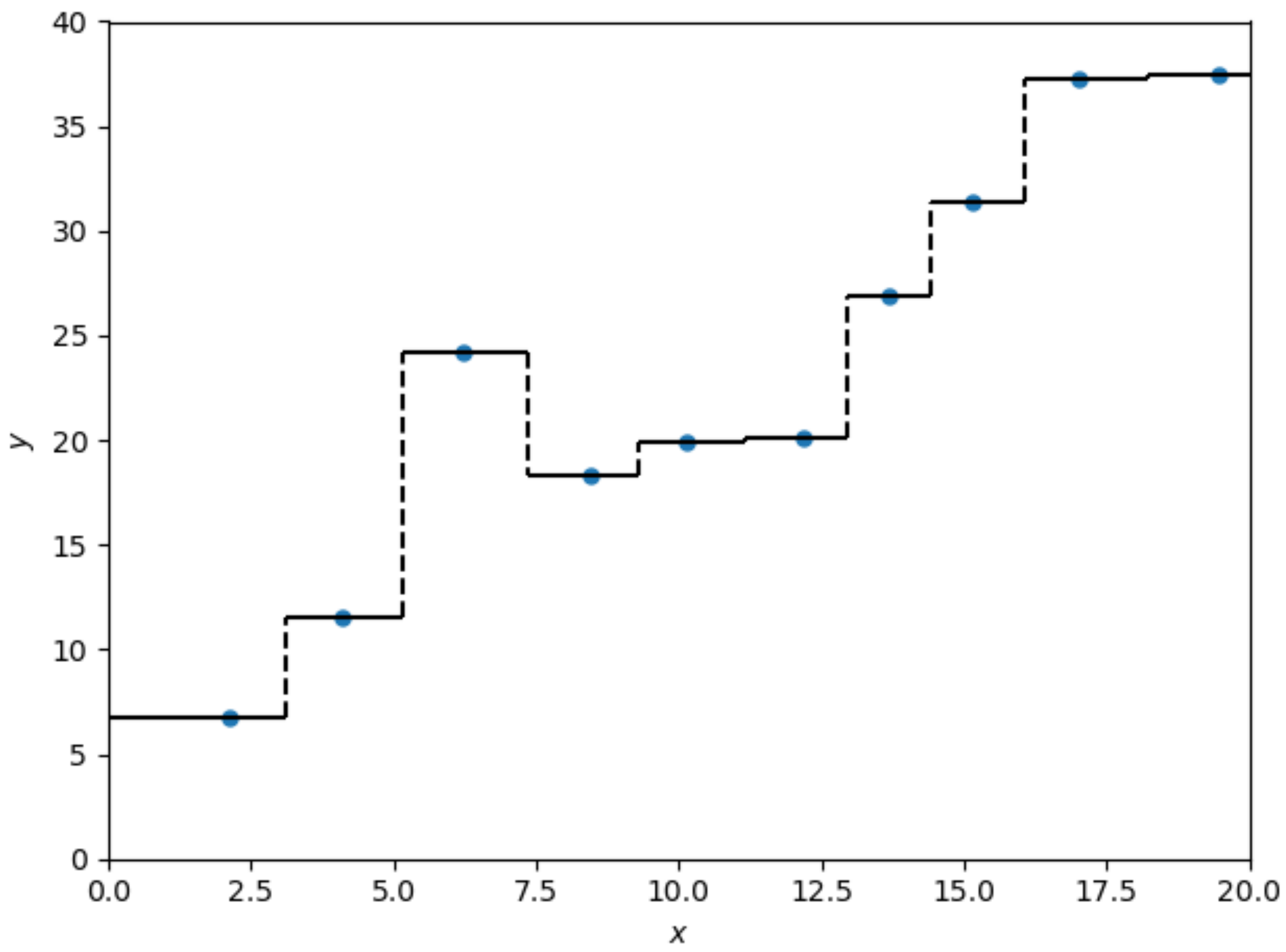
k-Nearest Neighbors

Nearest-Neighbor regression



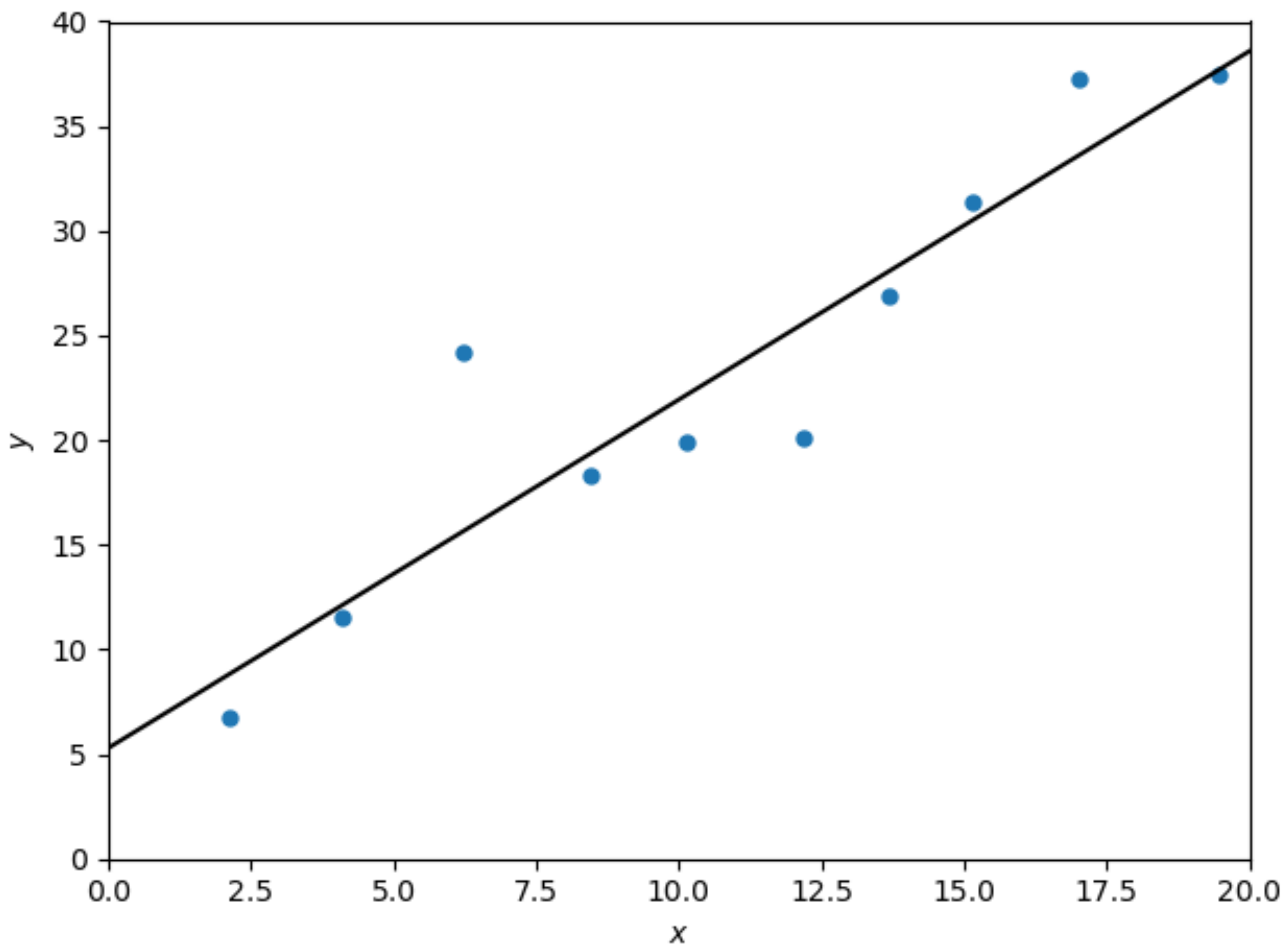
- $f(x) = y^{(i)}$, such that $x^{(i)} \in \mathscr{D}$ is the closest data point to x

Nearest-Neighbor regression



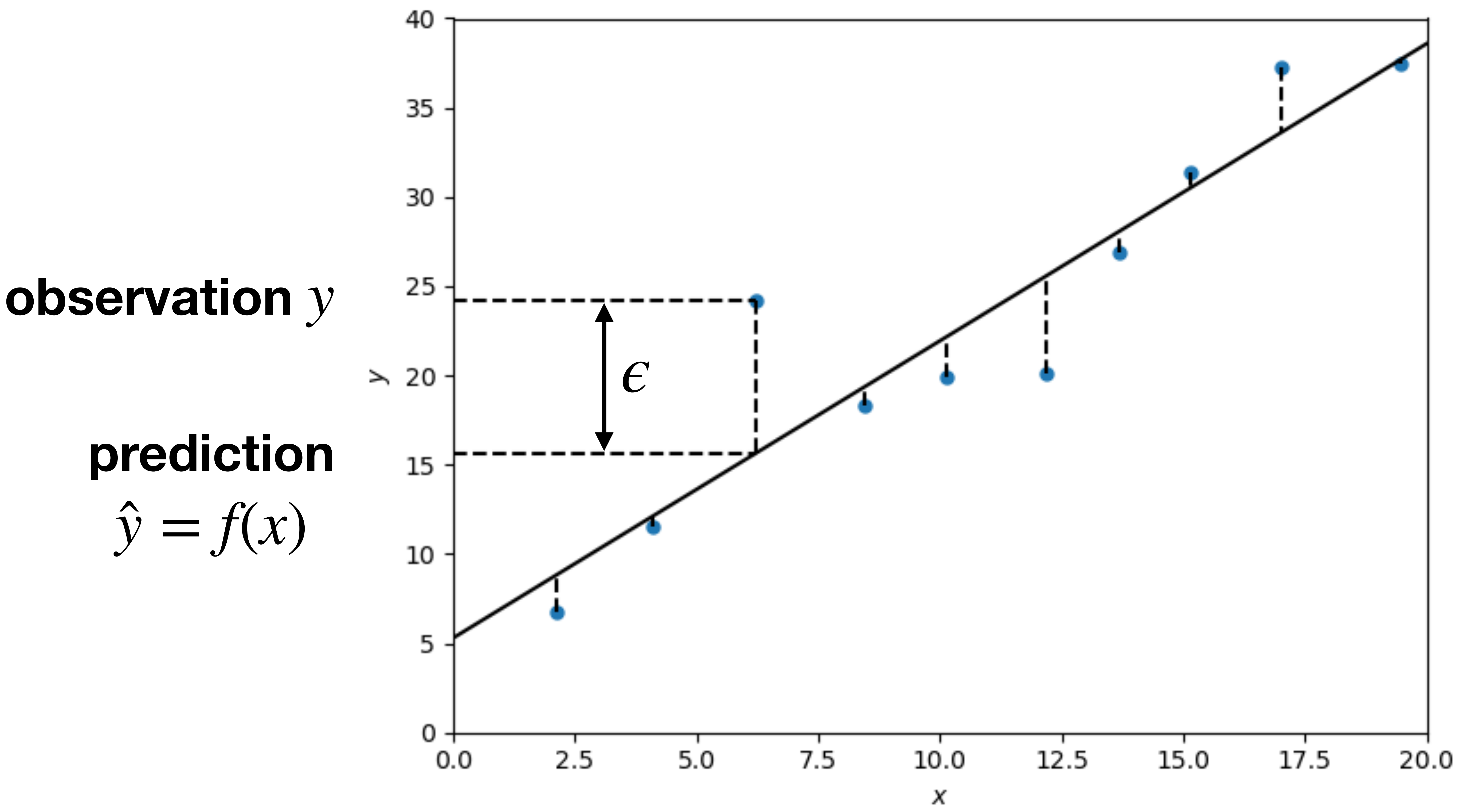
- Decision function $f : x \mapsto y$ is piecewise constant (for 1D x)
- Data induces f implicitly; f is never stored explicitly, but can be computed

Alternative: linear regression



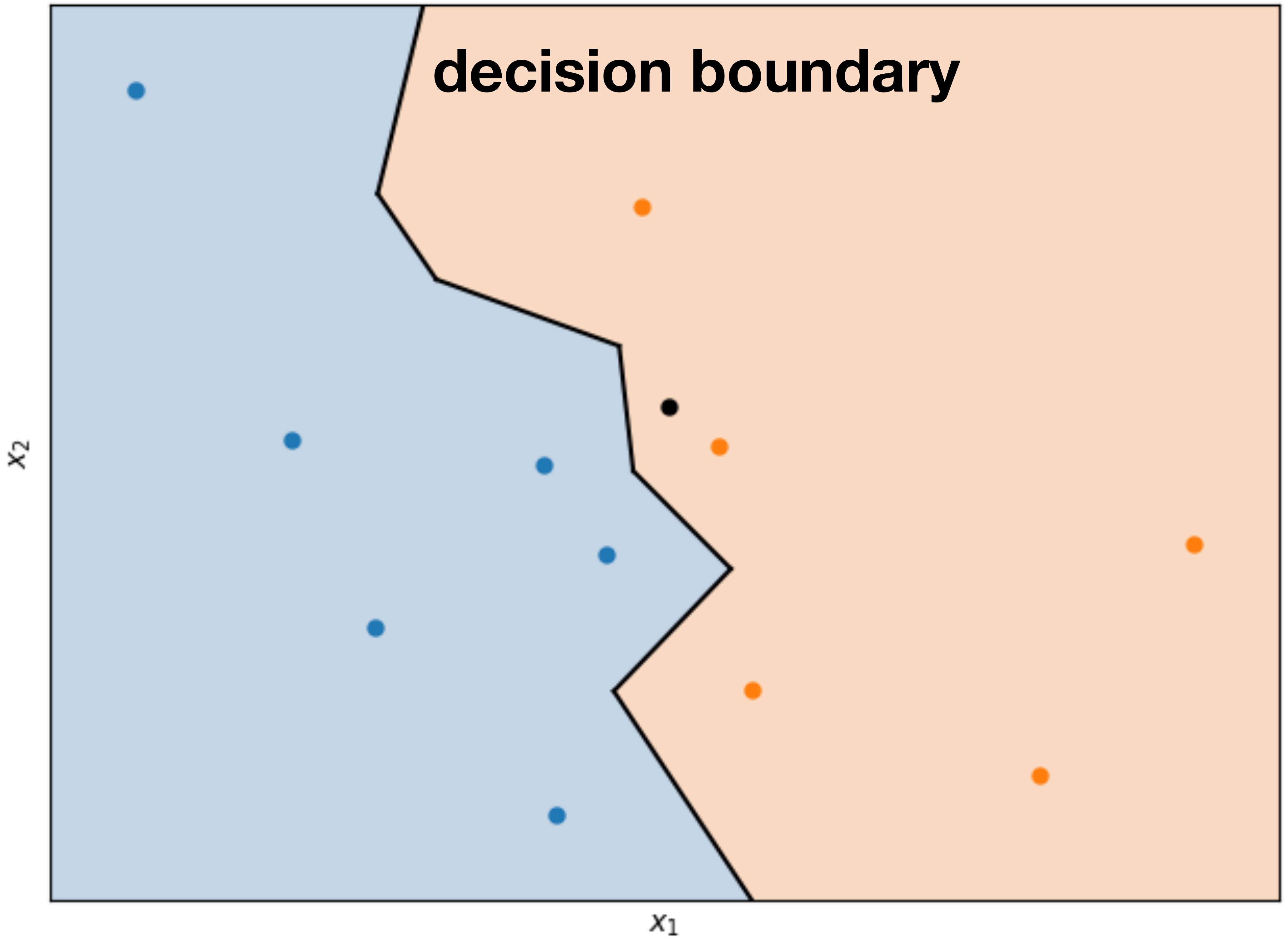
- Decision function $f: x \mapsto y$ is linear, $f(x) = \theta_0 + \theta_1 x$
- f is stored by its parameters $\theta = \begin{bmatrix} \theta_0 & \theta_1 \end{bmatrix}$

Measuring error



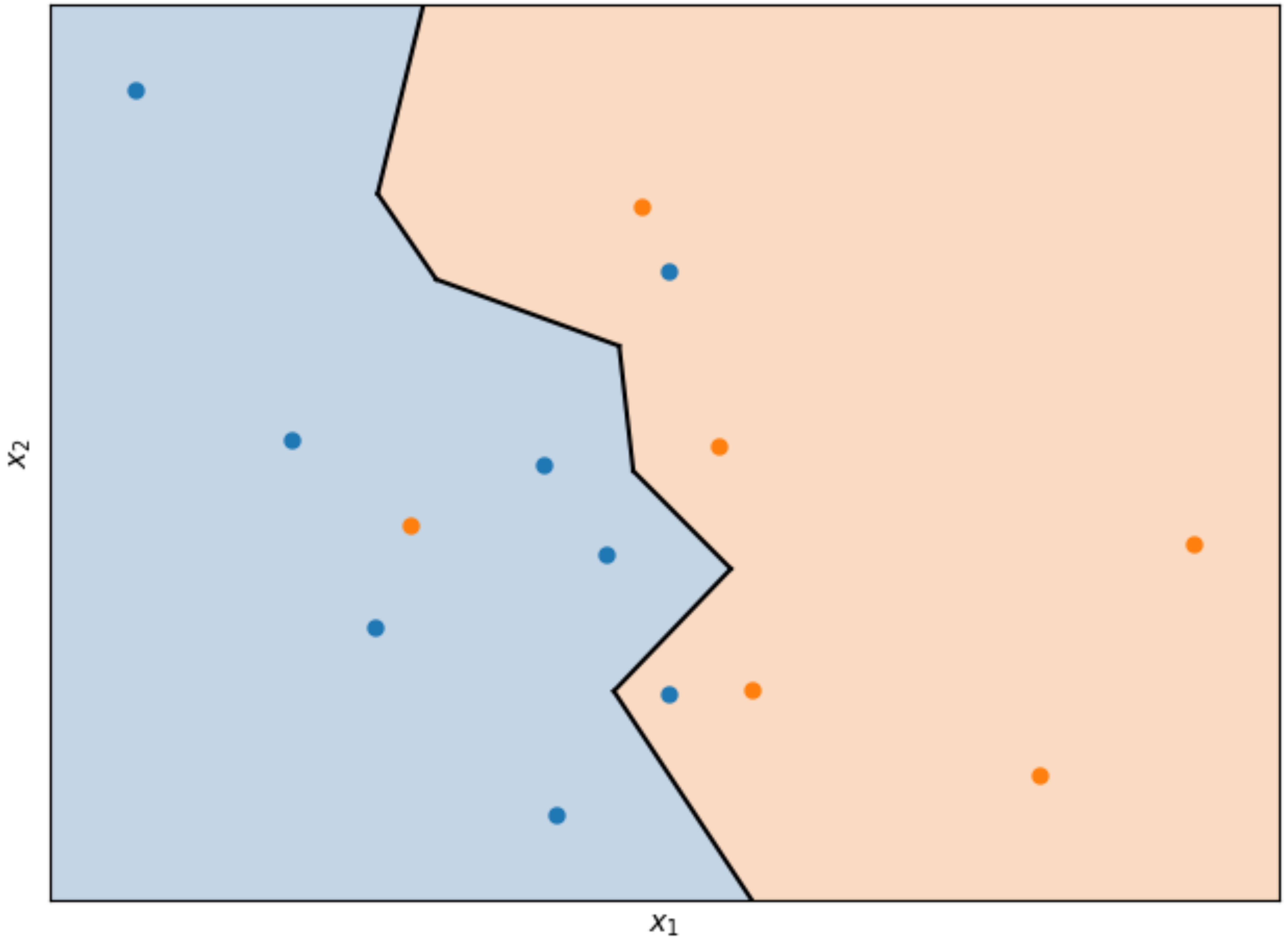
- Error / residual: $\epsilon = y - \hat{y}$
- Mean square error (MSE): $\frac{1}{m}\sum_i (\epsilon^{(i)})^2 = \frac{1}{m}\sum_i (y^{(i)} - \hat{y}^{(i)})^2$

Classification



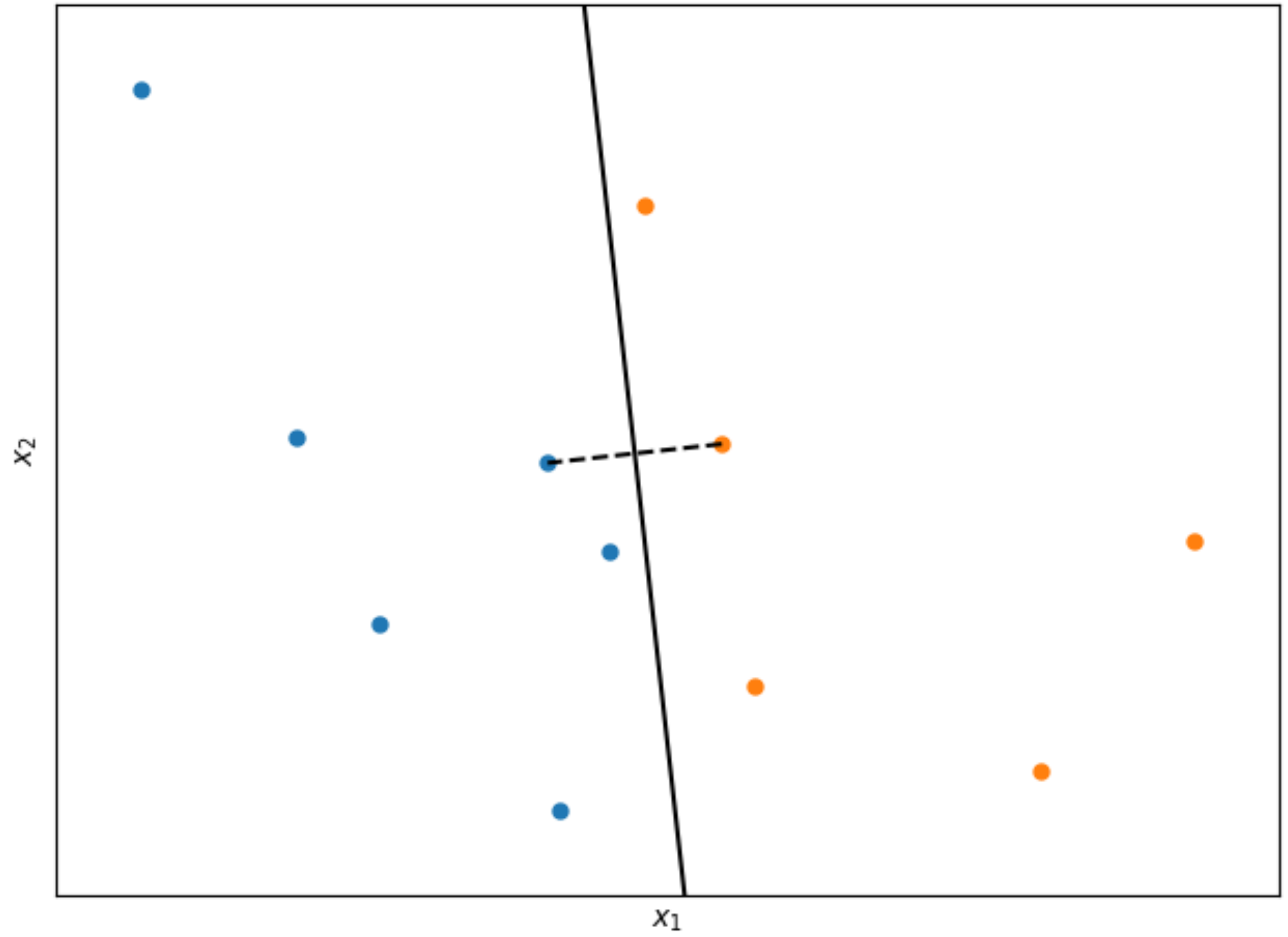
- Using colors as our "third dimension", we can visualize in 2D
- Particularly clear for classification, where y is discrete

Measuring error



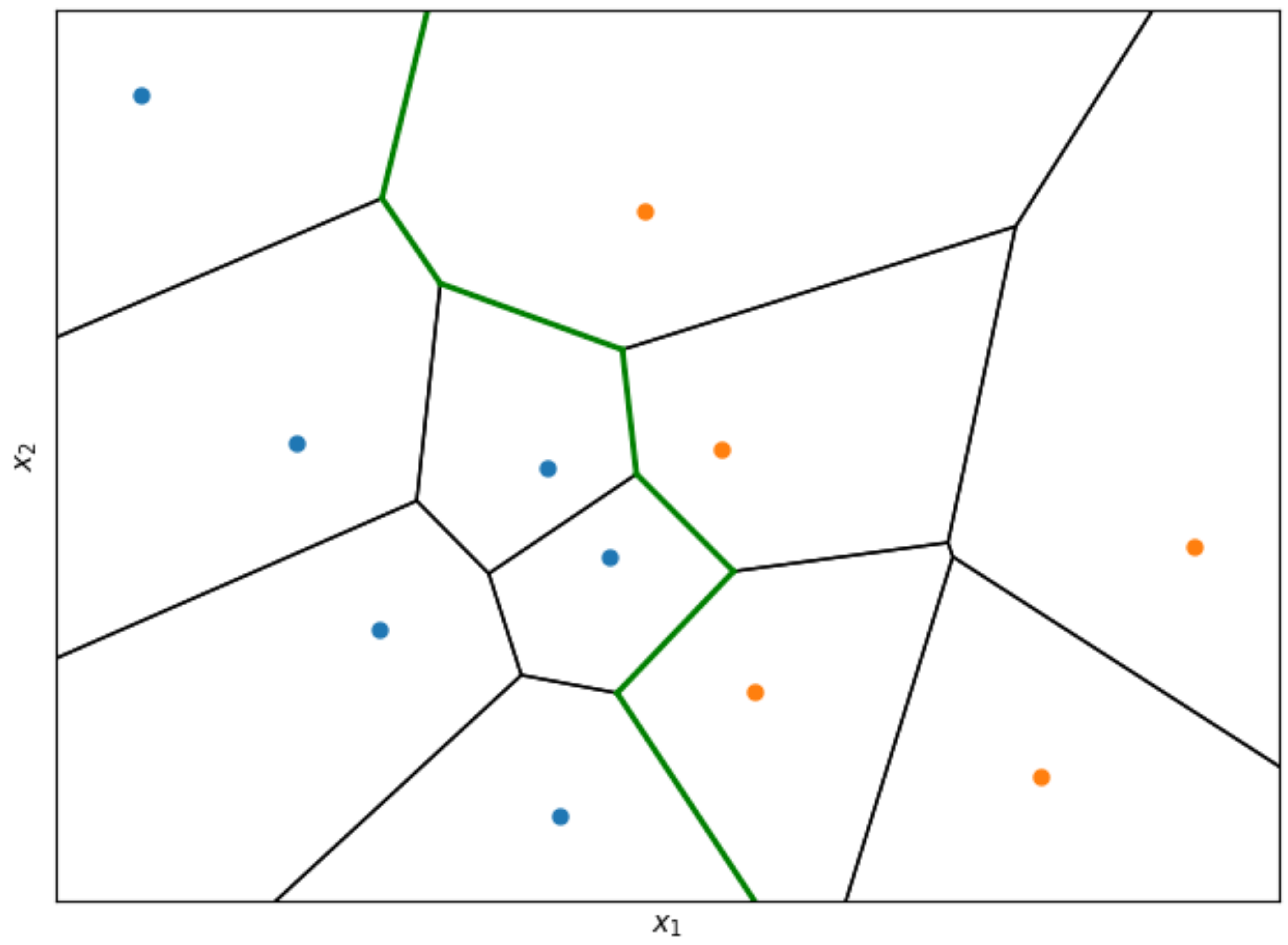
- Error rate: $\frac{1}{m}\sum_i \delta[y^{(i)} \neq \hat{y}^{(i)}]$

Decision boundary is piecewise linear



- For every two data points $x^{(i)}, x^{(j)}$ of different classes $y^{(i)} \neq y^{(j)}$
  - The hyperplane orthogonal to their midpoint is where $d(x, x^{(i)}) = d(x, x^{(j)})$
  - The decision boundary consists of some of these hyperplanes

Voronoi tessellation



- Each data point has a region in which it is the nearest neighbor
  - This region is a polygon
- The decision boundary consists of the edges that cross classes

Today's lecture

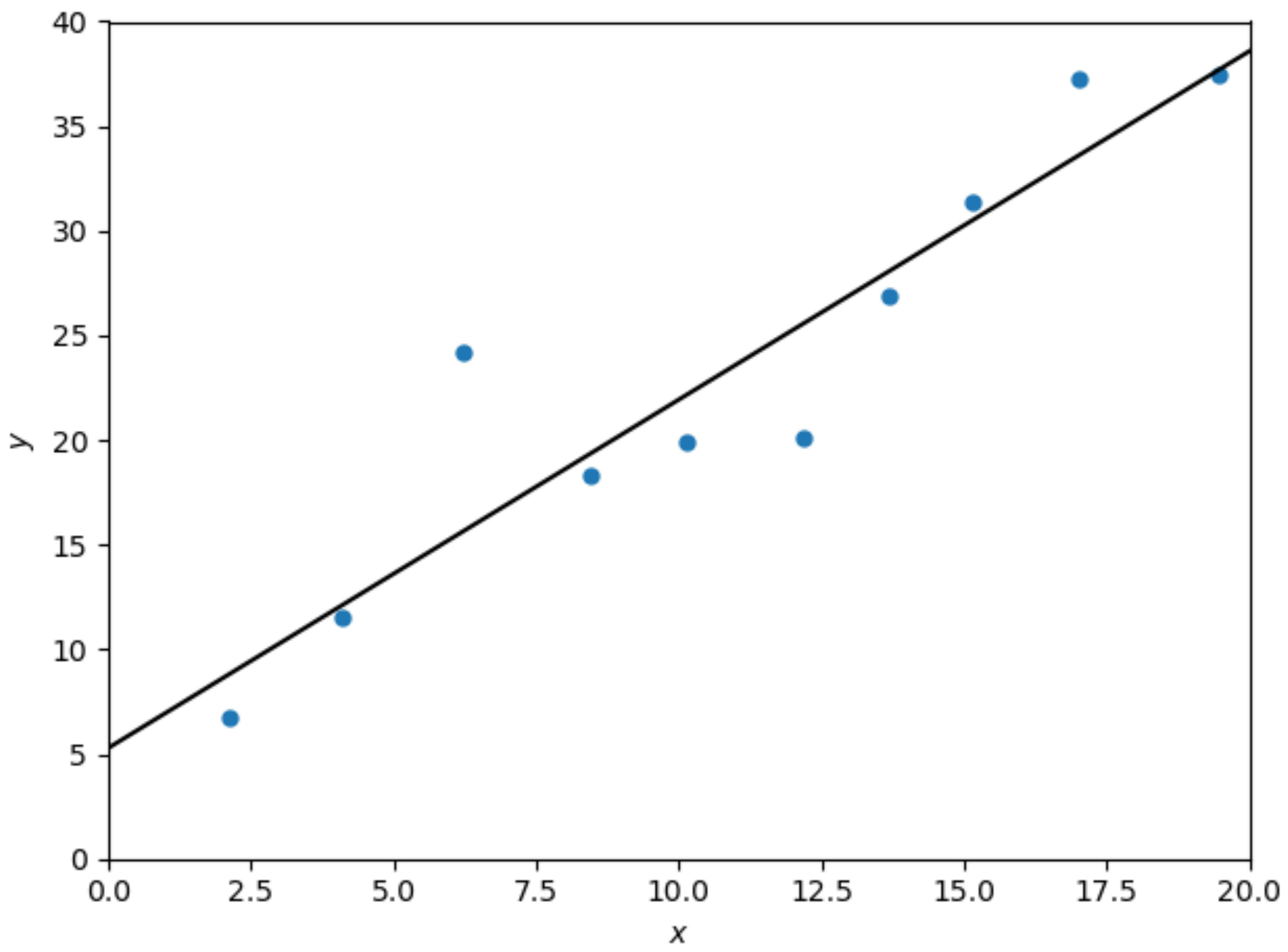
Supervised learning

Nearest Neighbors

Overfitting and complexity

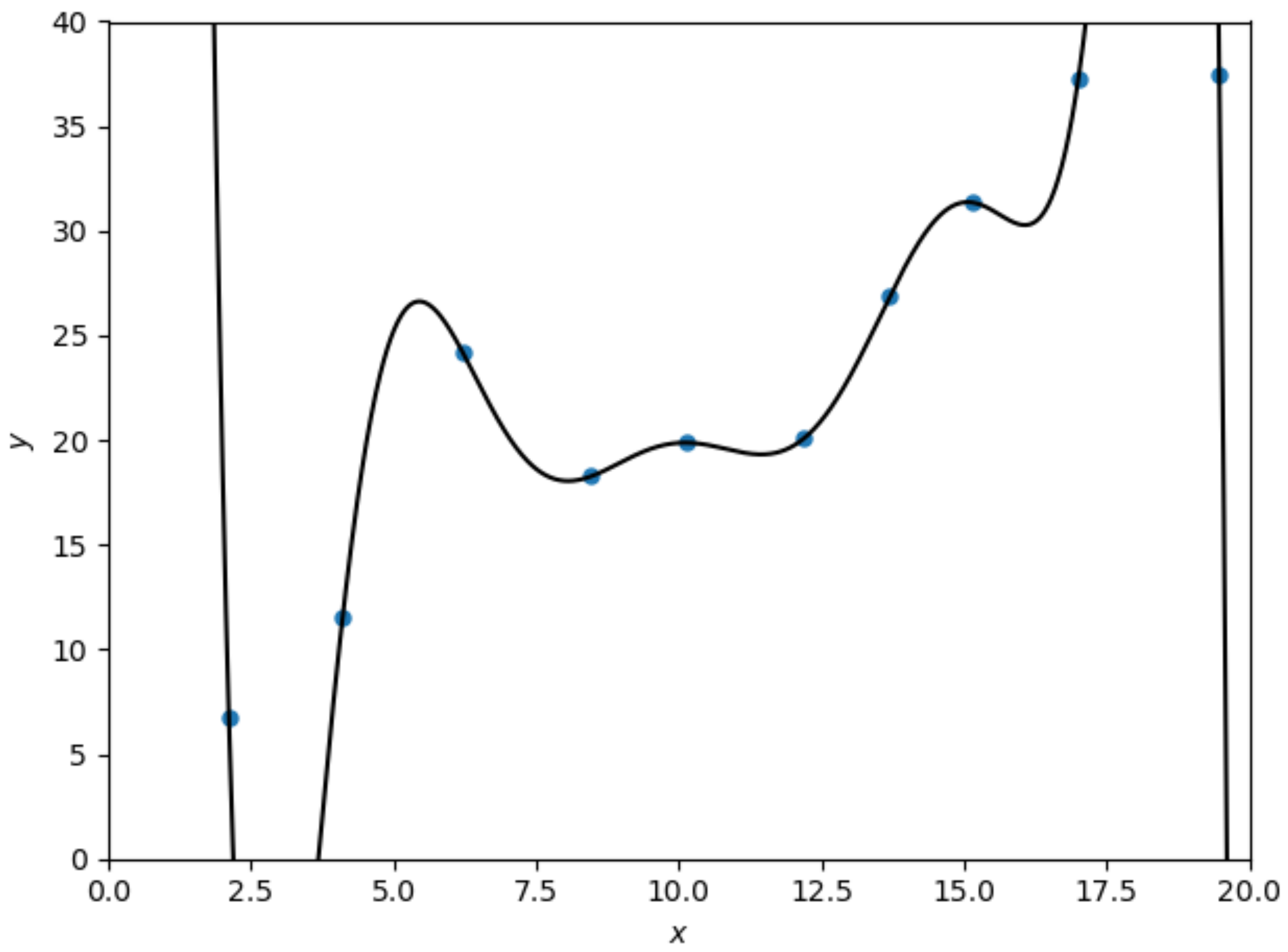
k-Nearest Neighbors

Overfitting and complexity



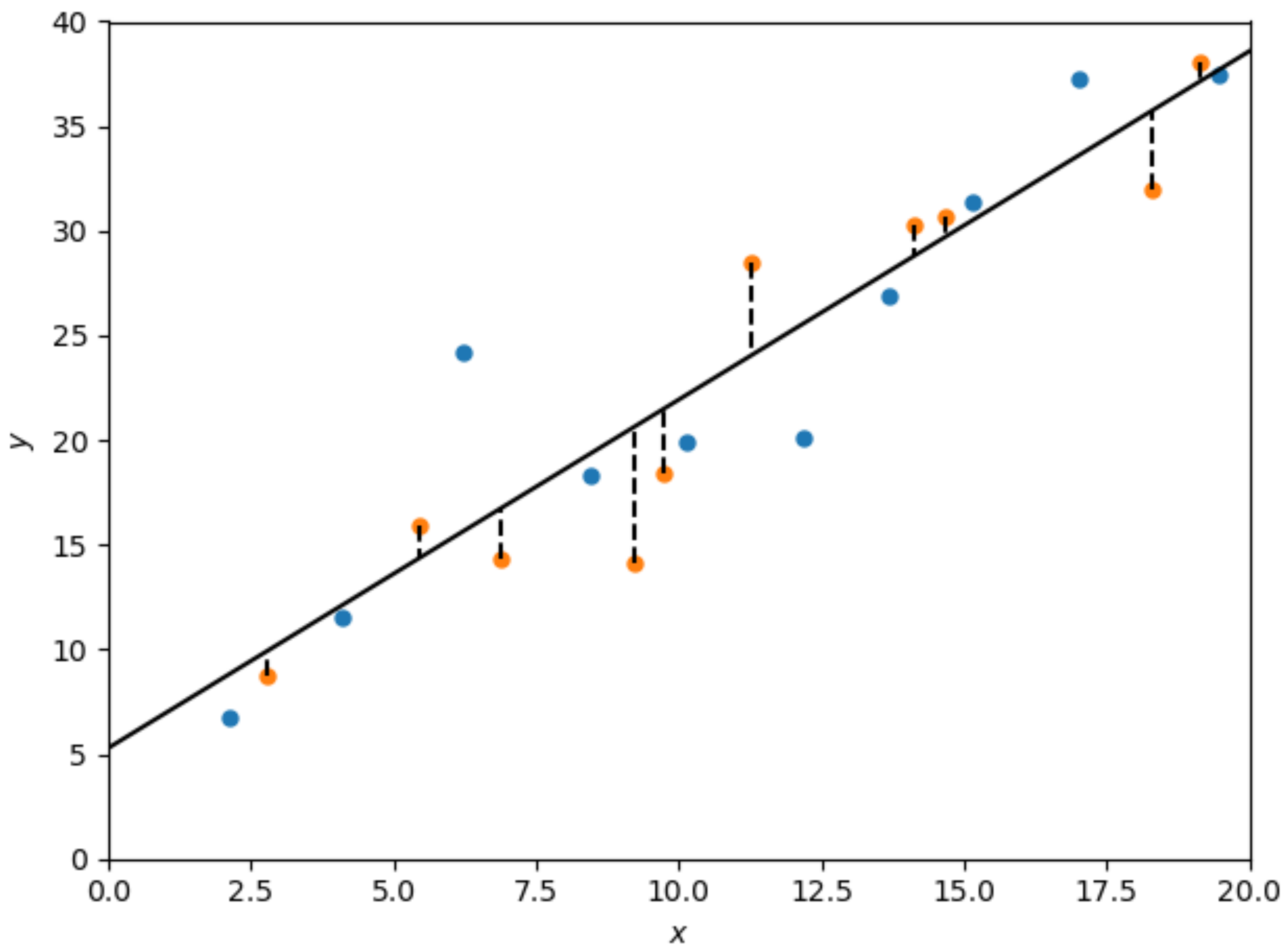
- Simple linear model
- Fits the training data, but with errors
- Interpolation seems reasonable

Overfitting and complexity



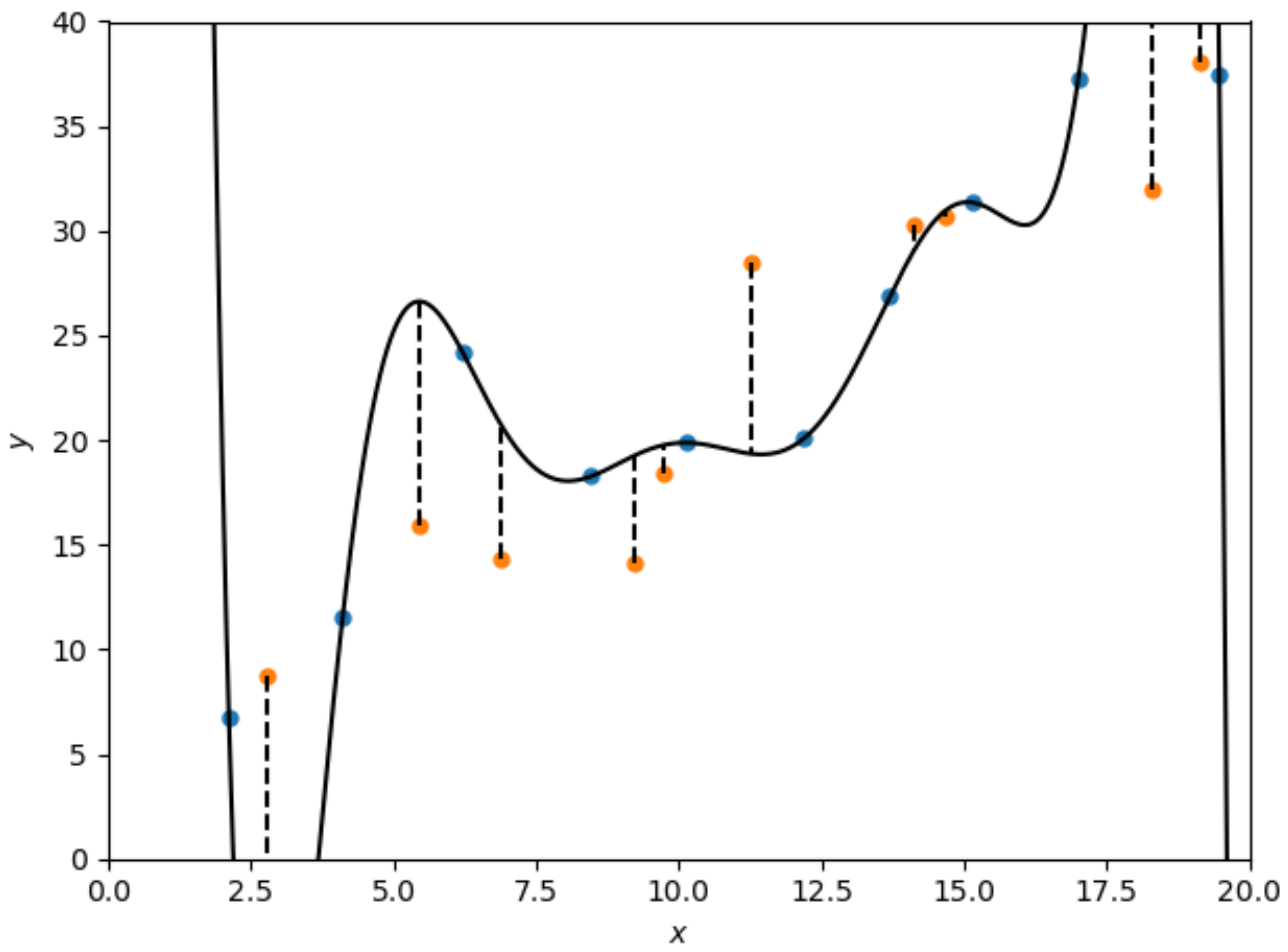
- High-order polynomial model
- Fits the training data perfectly
- Interpolation? more like confabulation, amirite?

Overfitting and complexity



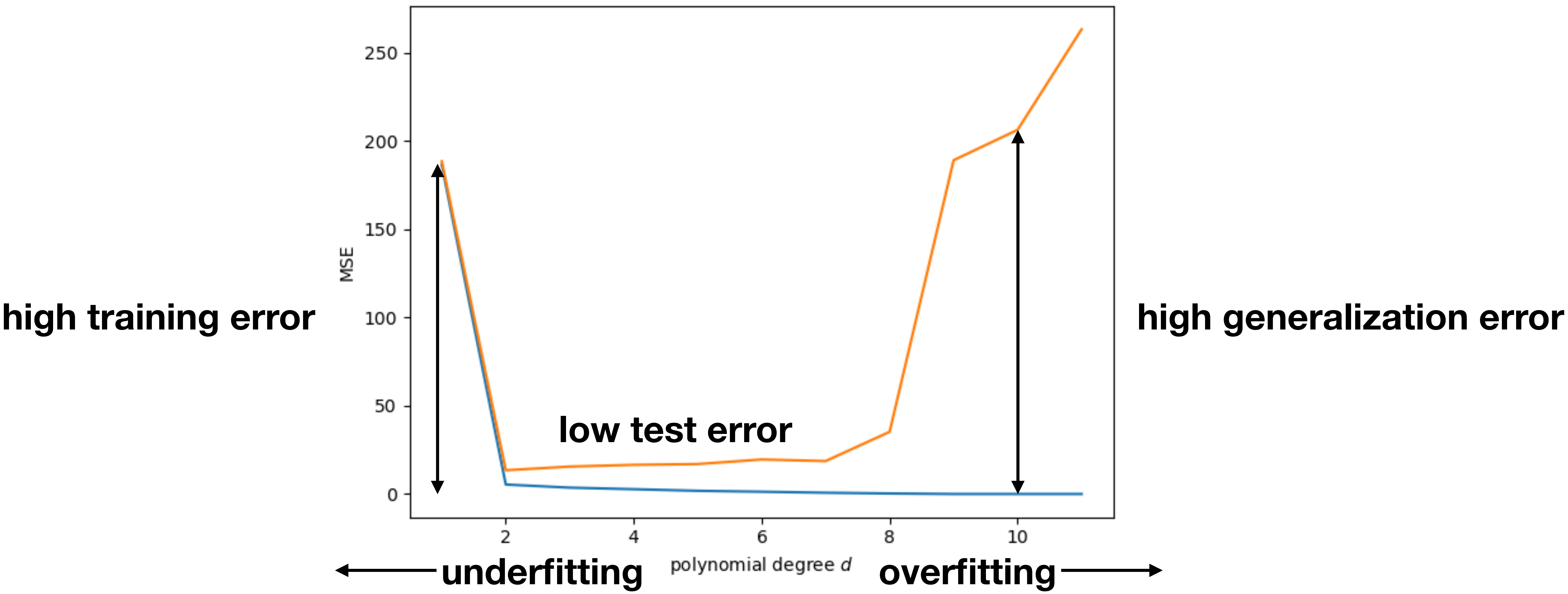
- New test data will also have prediction errors
- Good generalization = test errors will be similar to training errors

Overfitting and complexity



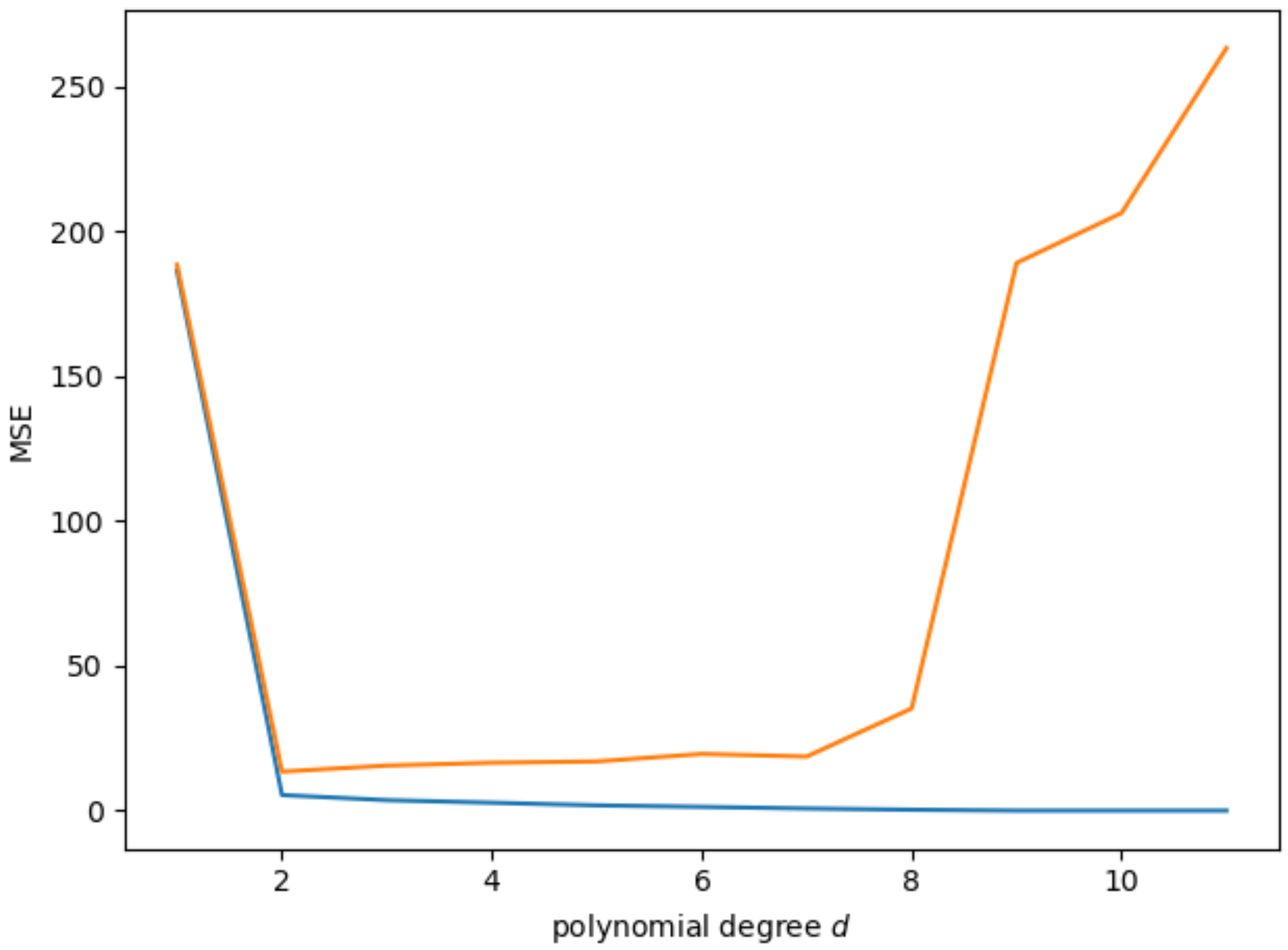
- A complex model may fit the training data well → low training error
- But it may generalize poorly to test data → high test error
- This is called overfitting the training data

How overfitting affects prediction error



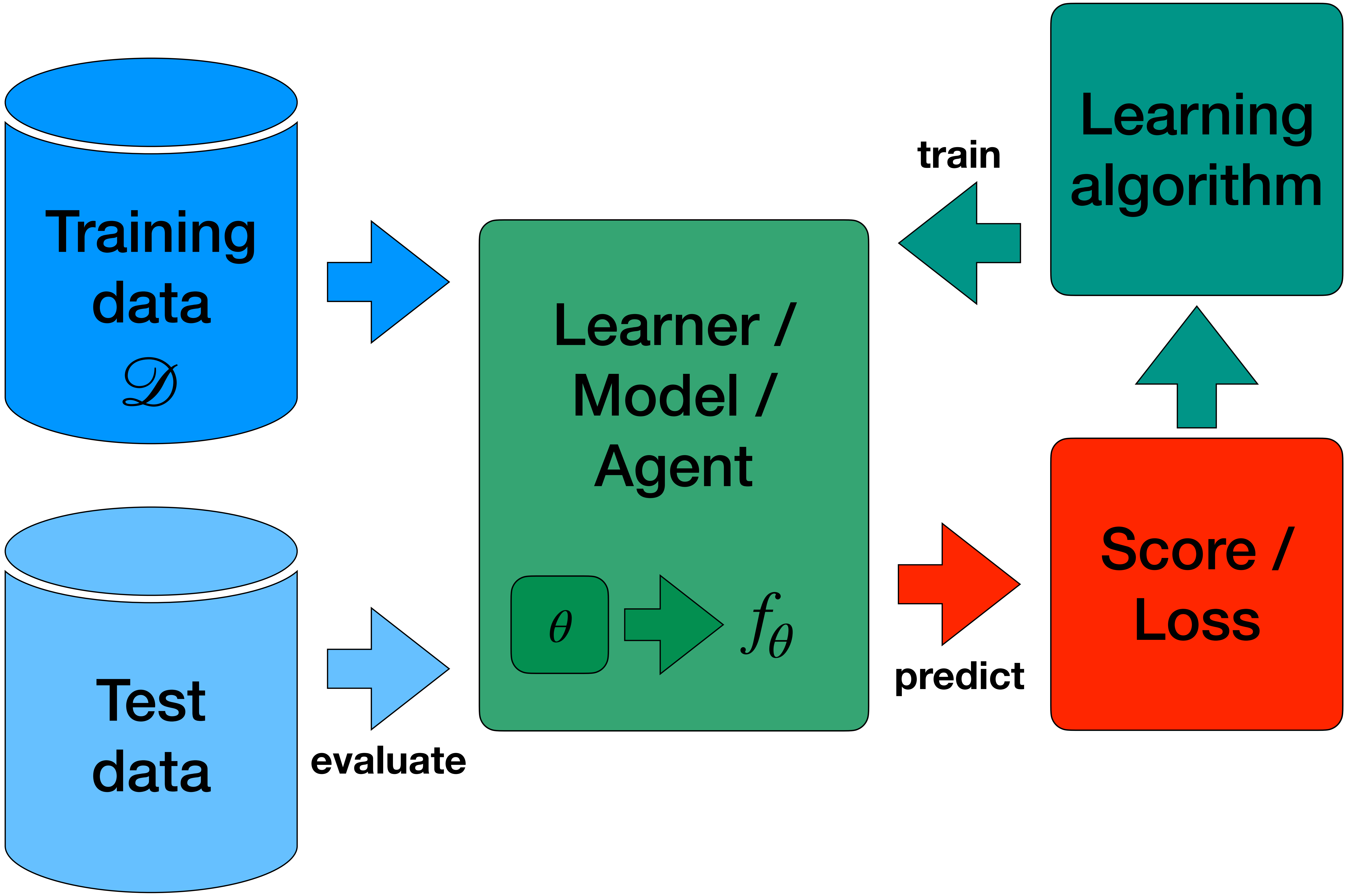
- Low model complexity → underfitting
  - High test error = high training error + low generalization error
- High model complexity → overfitting
  - High test error = low training error + high generalization error

Validation

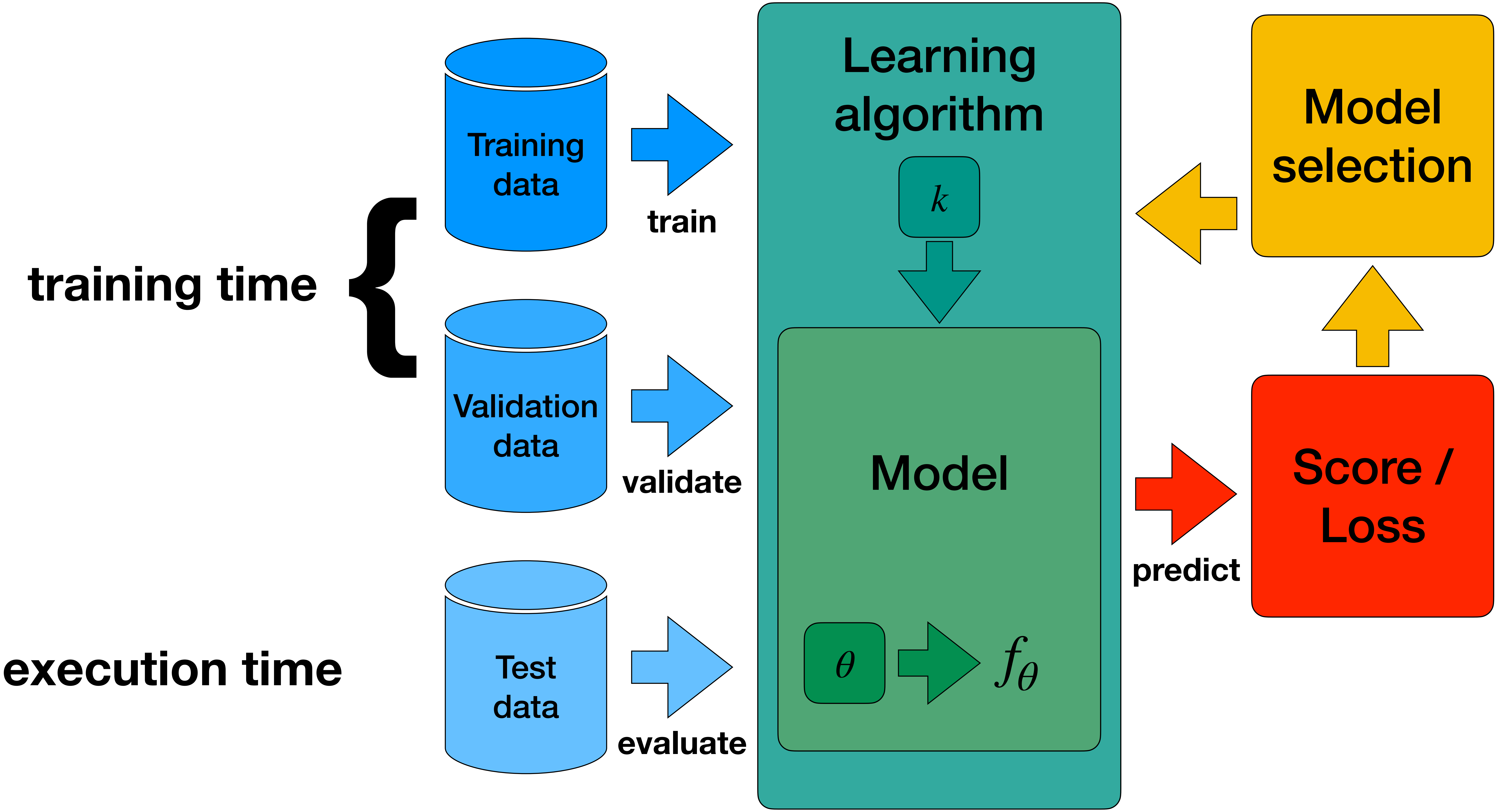


- How can we choose the model complexity? with learning!
  - Model selection = choose our model class
  - Score function: low test error = training error + generalization error

Model learning



Model selection



Recap: overfitting and complexity

- Test error = training error + generalization error
- Model complexity may lead to overfitting
  - Fit the training data very well, but generalize poorly
- Model simplicity may lead to underfitting
  - Do as poorly on the test data as on the training data

Today's lecture

Supervised learning

Nearest Neighbors

Overfitting and complexity

k-Nearest Neighbors

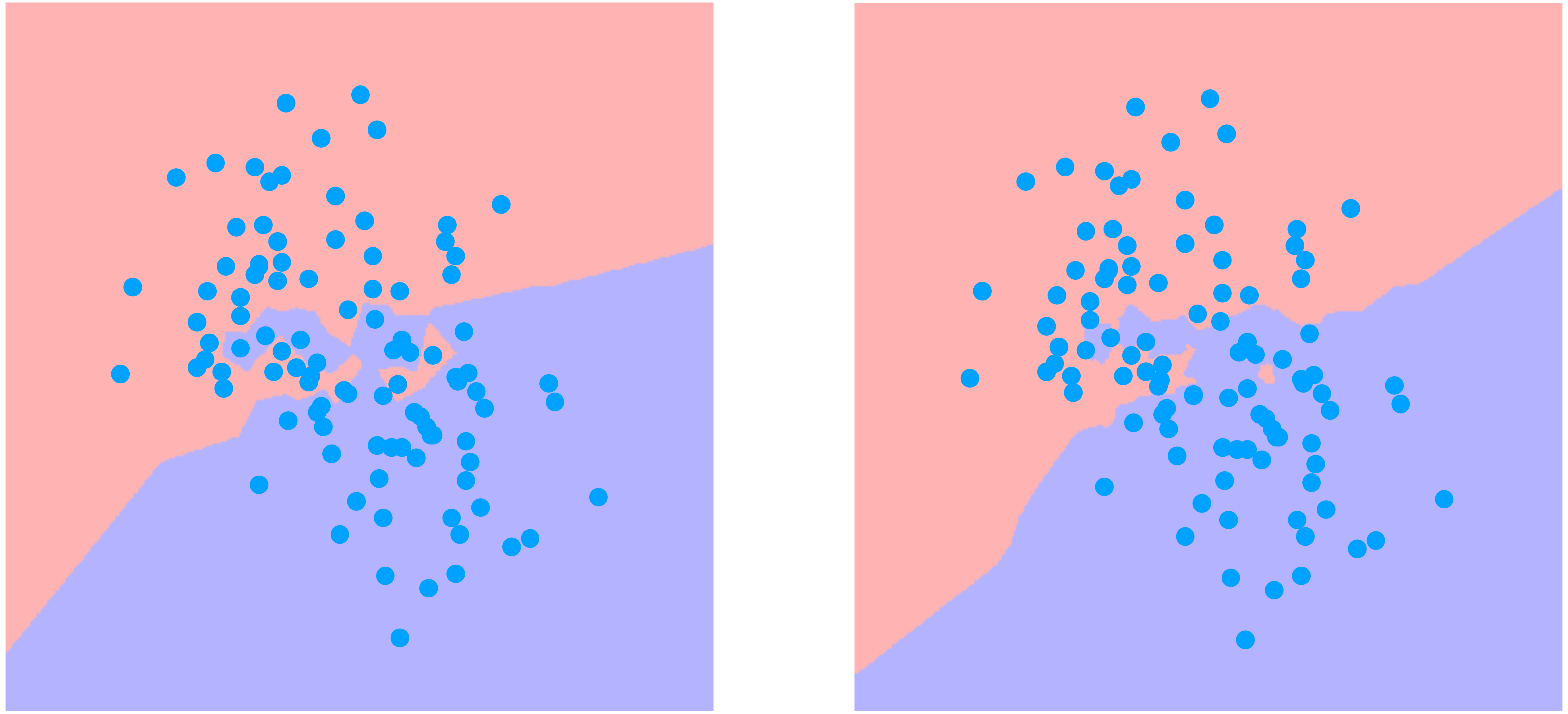
k-Nearest Neighbor (kNN)

- Find the k nearest neighbors to x in the dataset
  - Given x, rank the data points by their distance from x, $d(x, x^{(j)})$
    - Usually, Euclidean distance $d(x, x^{(j)}) = \sqrt{\sum_i (x_i - x_i^{(j)})^2}$
  - Select the k data points which are have smallest distance to x
- What is the prediction?
  - Regression: average $y^{(j)}$ for the k closest training examples
  - Classification: take a majority vote among $y^{(j)}$ for the k closest training examples
    - No ties in 2-class problems when k is odd

kNN decision boundary

- For classification, the decision boundary is piecewise linear
- Increasing k "simplifies" the decision boundary
  - Majority voting means less emphasis on individual points

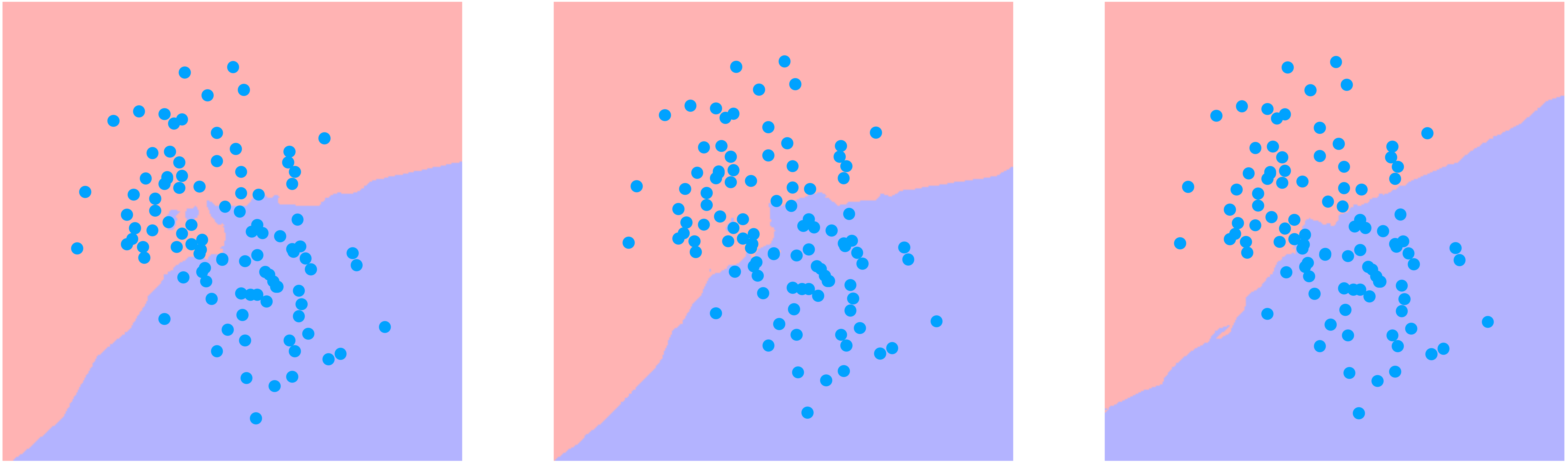
$k = 1$ $\qquad\qquad$ $k = 3$



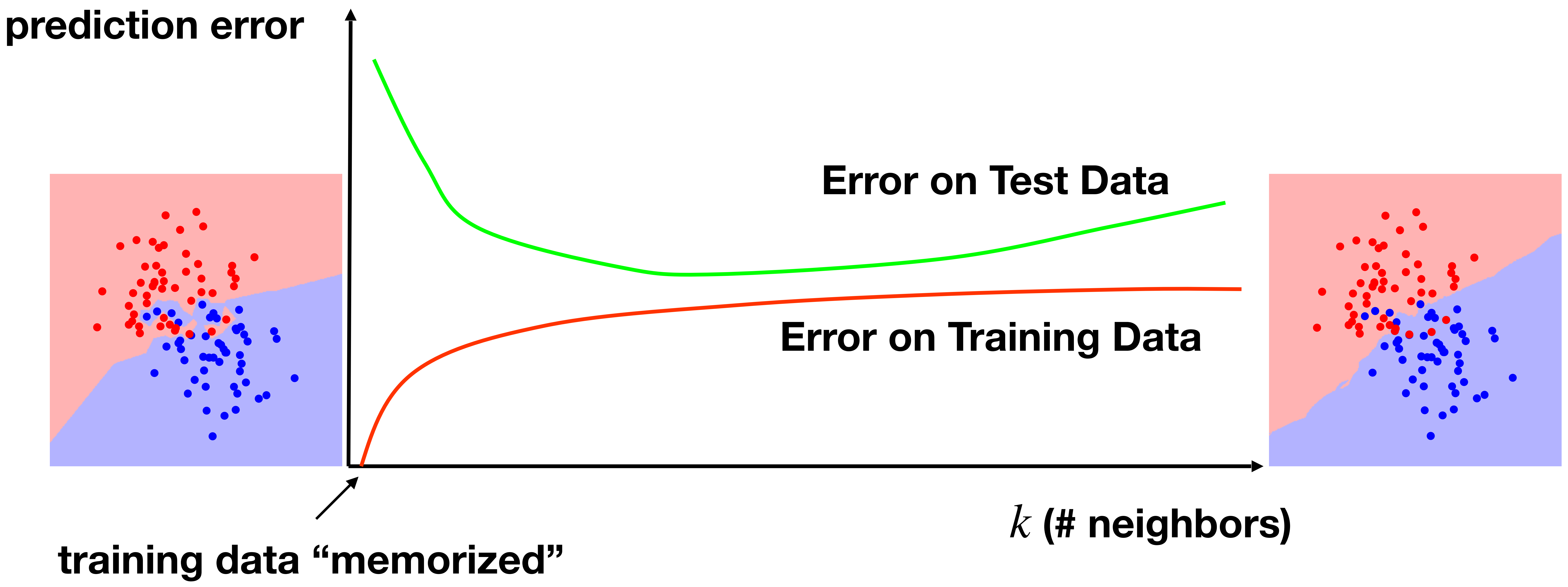
kNN decision boundary

- For classification, the decision boundary is piecewise linear
- Increasing k "simplifies" the decision boundary
  - Majority voting means less emphasis on individual points

$k = 5$ $k = 7$ $k = 25$



Error rates and k



- A complex model fits training data but generalizes poorly
- $k = 1$: perfect memorization of examples = complex
- $k = m$: predict majority class over entire dataset = simple
- We can select k with validation

kNN classifier: further considerations

- Decision boundary smoothness
  - Increases with k, as we average over more neighbors
  - Decreases with training size m, as more points support the boundary
  - Generally, optimal k should increase with m
- Extensions of k-Nearest Neighbors
  - Do features have the same scale? importance?
    - Weighted distance: $d(x, x') = \sqrt{\sum_i w_i(x_i - x'_i)^2}$
    - Non-Euclidean distances may be more appropriate for type of data
  - Fast search techniques (indexing) to find k closest points in high-dimensional space
  - Weighted average / voting based on distance: $\hat{y} = \sum_j w(d(x, x^{(j)}))y^{(j)}$

Recap: k-Nearest Neighbors

- Piecewise linear decision boundary
  - Just for analysis — the algorithm doesn't compute the boundary
- With $k > 1$:
  - Regression → (weighted) average
  - Classification → (weighted) vote
- Overfitting and complexity:
  - Model “complexity” goes down as k grows
  - Use validation data to estimate test error rates and select k

Logistics

assignment 1

- Assignment 1 will be up soon
- Due: Tue, Oct 5 (Pacific)

project

- Project guidelines will resemble last year's