Prediction problems

- Given: input-output pairs $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_m, \mathbf{y}_m)$, where the $\mathbf{x}_i \in \mathcal{X}$ and the $\mathbf{y}_i \in \mathcal{Y}$.

- Produce: a function $\hat{f} : \mathcal{X} \mapsto \mathcal{Y}$ that produces the "correct" $\mathbf{y}$ for any $\mathbf{x} \in \mathcal{X}$.

- This is called prediction, supervised learning, function approximation.
  Often, we imagine that there is a "true" $f : \mathcal{X} \mapsto \mathcal{Y}$, and $\mathbf{y}_i = f(\mathbf{x}_i)$.

Examples

input	output
DNA sequence	Does TF z bind there? (yes/no)
Tumor	Malignant/benign?
Tumor	Life expectancy of patient? (real value)
Expression of a gene under some conditions	Expression under other conditions? (real value)
Pair of proteins	Do they interact? (yes/no)
One protein	With what does it interact?

Steps to solving a supervised learning problem

1. Decide what your input-output pairs are.
2. Decide how to encode inputs and outputs.
   - This defines the input space $\mathcal{X}$, and the output space $\mathcal{Y}$.
   - Nearness in $\mathcal{X}$ should reflect nearness in $\mathcal{Y}$.
3. Choose a class of functions/representations $\mathcal{F}$ for approximating f.
   - Each possible $\hat{f} \in \mathcal{F}$ is a function from $\mathcal{X}$ to $\mathcal{Y}$.
4. Choose an error/cost function, $\mathcal{E}$, which measures how good each $\hat{f} \in \mathcal{F}$ is.
5. Apply a learning algorithm to find an $\hat{f}$. (Ideally, minimizing $\mathcal{E}$.)

Example

Wisconsin Breast Tumor data set from UC-Irvine Machine Learning repository.

- Thirty real-valued variables per tumor that can be used for prediction.
- Two variables that can be predicted:
  - Outcome (R=recurrence, N=non-recurrence)
  - Time (until recurrence, for R, time healthy, for N).

tumor size	texture	perimeter	...	outcome	time
18.02	27.6	117.5		N	31
17.99	10.38	122.8		N	61
20.29	14.34	135.1		R	27
...					

Terminology

tumor size	texture	perimeter	...	outcome	time
18.02	27.6	117.5		N	31
17.99	10.38	122.8		N	61
20.29	14.34	135.1		R	27
...					

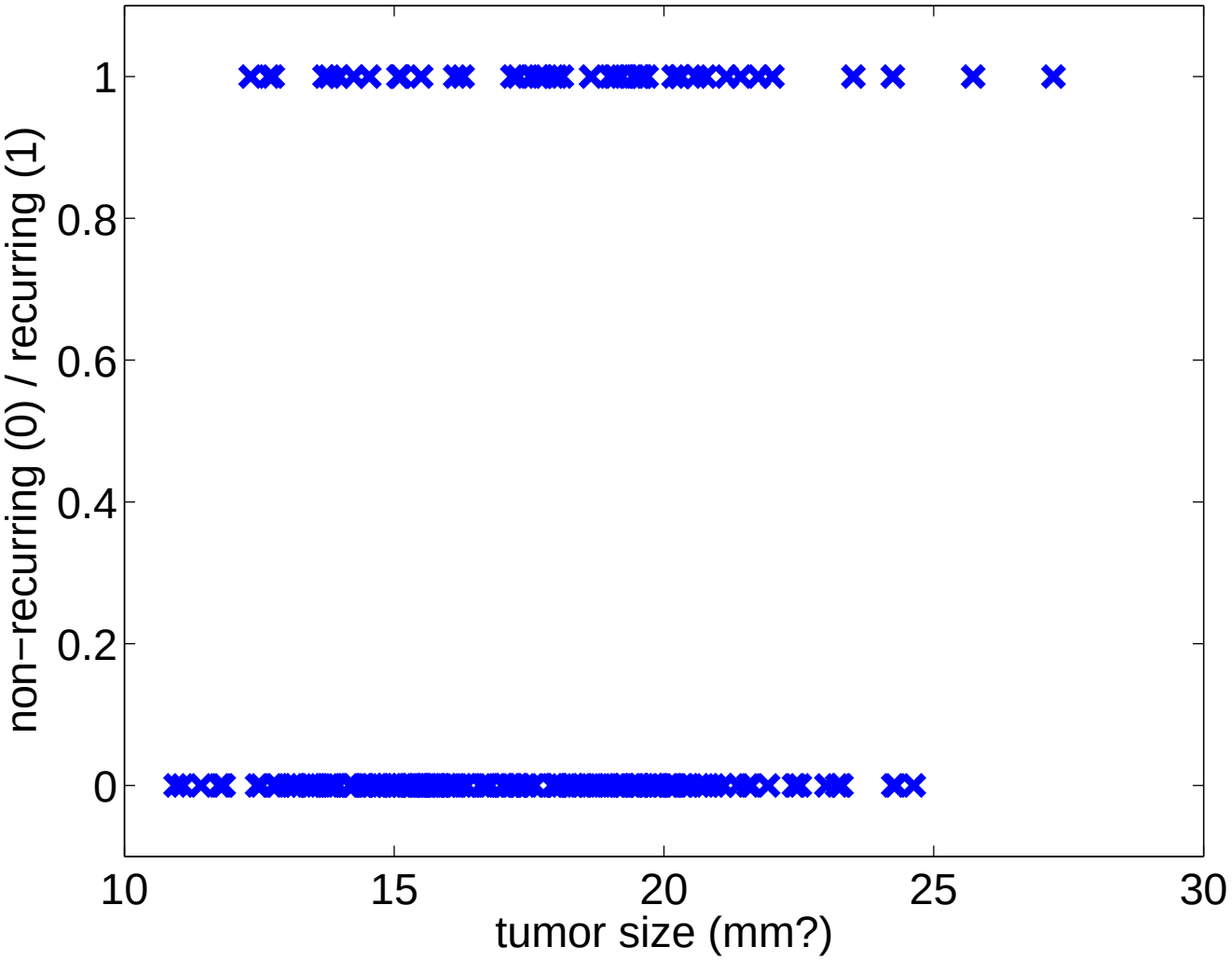
- Rows are [training] examples, samples.
- Columns (tumor size, texture, ...) are features, attributes, [independent] variables, inputs.
- Outcome and time are dependent variables, targets, outputs, target outputs.
- Predicting outcome is a *[binary] classification* problem.
- Predicting time is a *regression* problem.

Nearest-neighbor methods

with application to the Wisconsin Breast Cancer data

Problem formulation

1. Predict outcome based on tumor size.
2. Tumor size is taken as is providing a single, real-valued input. Outcomes are coded as N$= 0$, R$= 1$.
3. Any function $\hat{f} : \Re \mapsto \{0, 1\}$ is allowed.
4. Error function. . . to be discussed
5. Learning algorithm: variants of nearest neighbor.


non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)

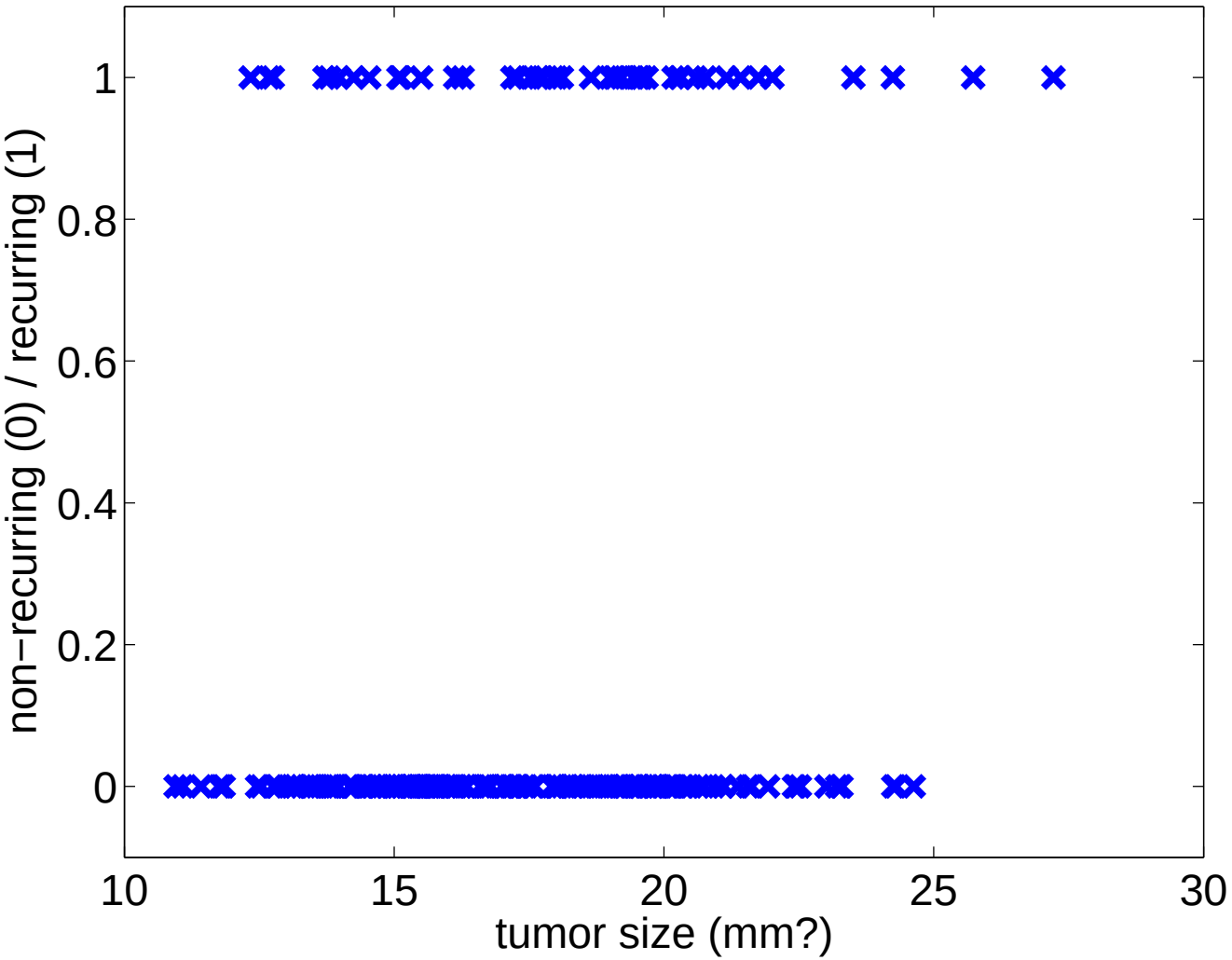
[Single] nearest neighbor

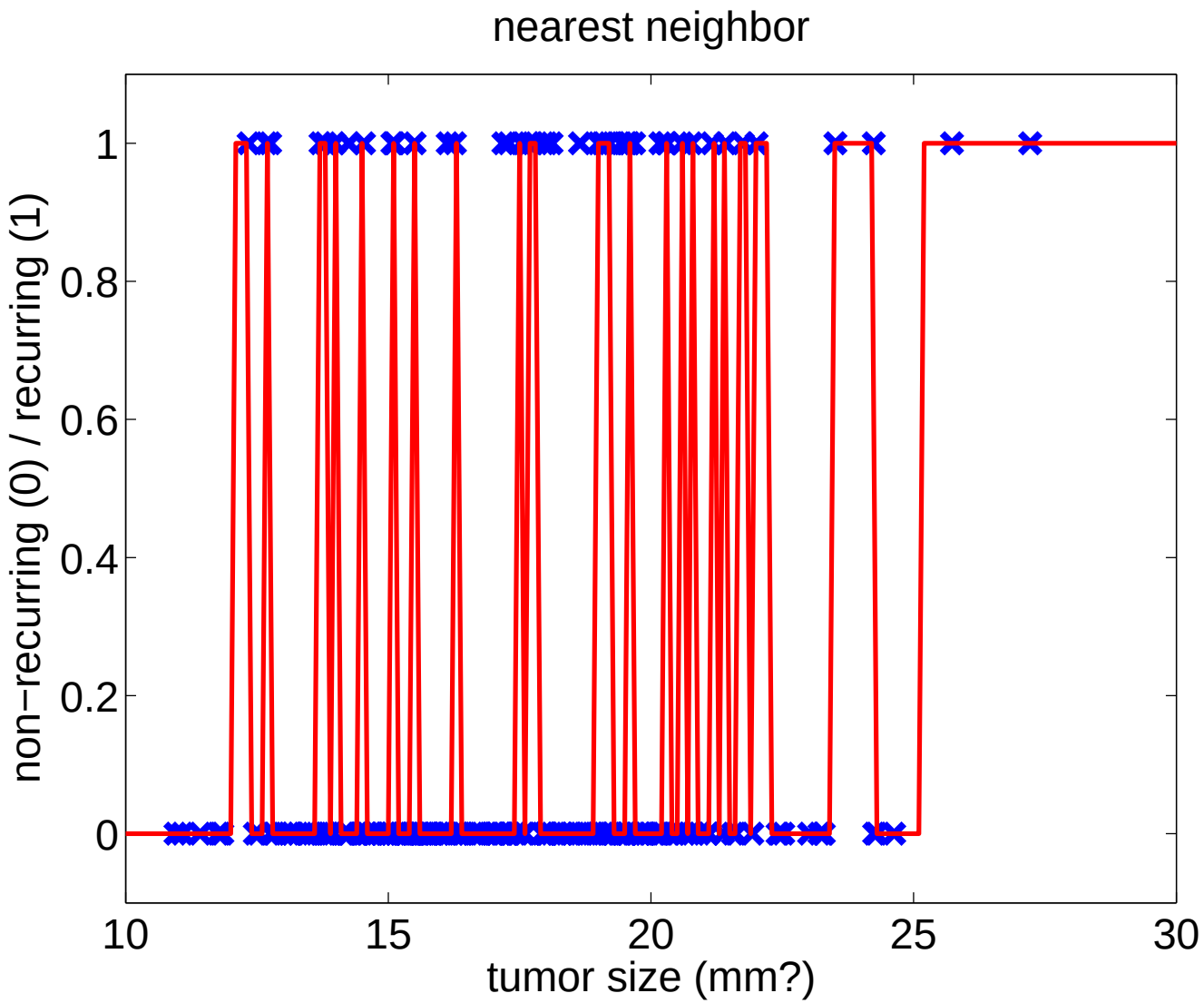
- Given: Training data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$, distance metric d on $\mathcal{X}$.
- Learning: Nothing to do!
- Prediction: for $\mathbf{x} \in \mathcal{X}$
  - Find nearest training sample to $\mathbf{x}$.

$$i \in \arg\min_i d(\mathbf{x}_i, \mathbf{x})$$

  - Predict $\mathbf{y} = \mathbf{y}_i$.

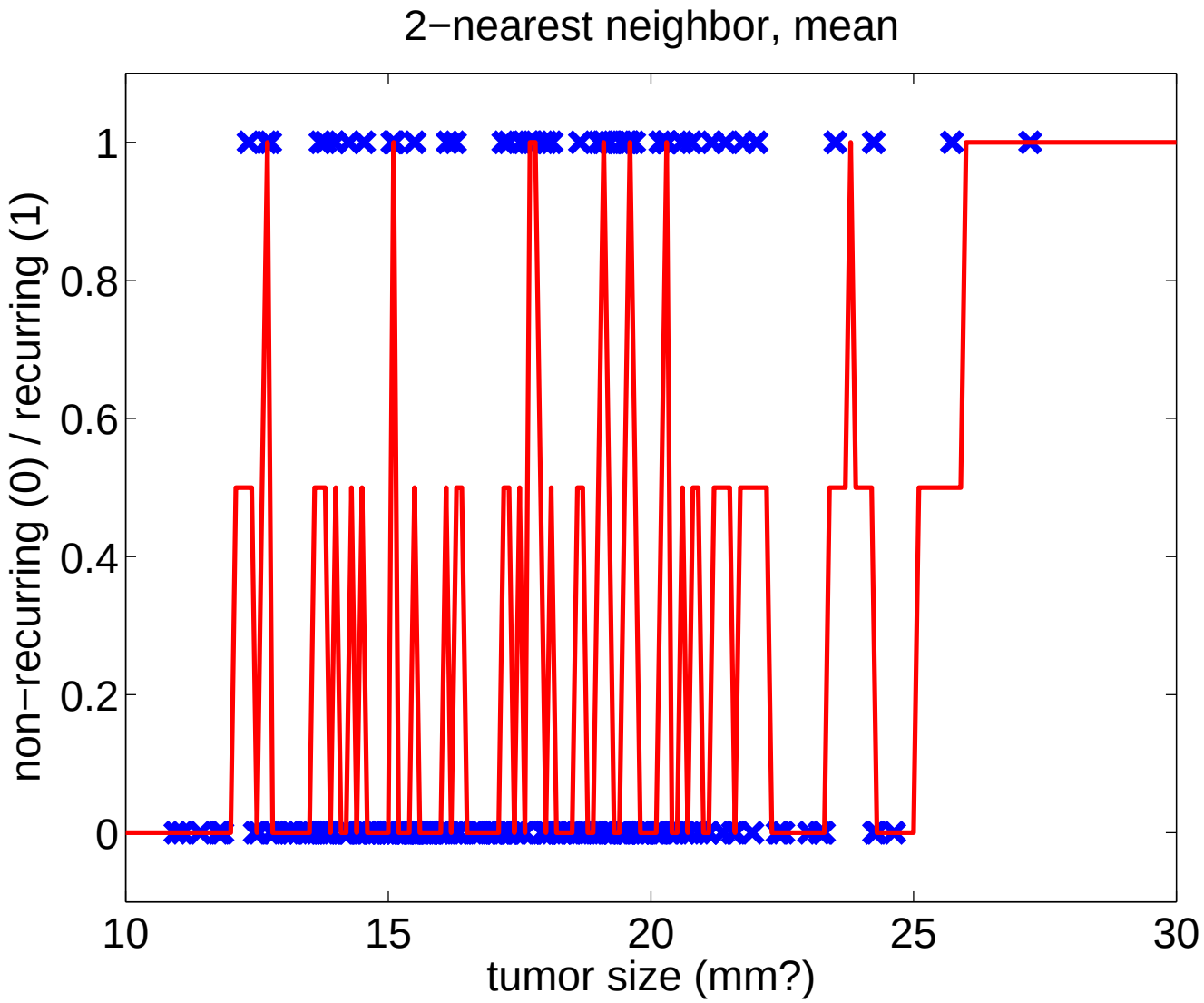
How will it look?



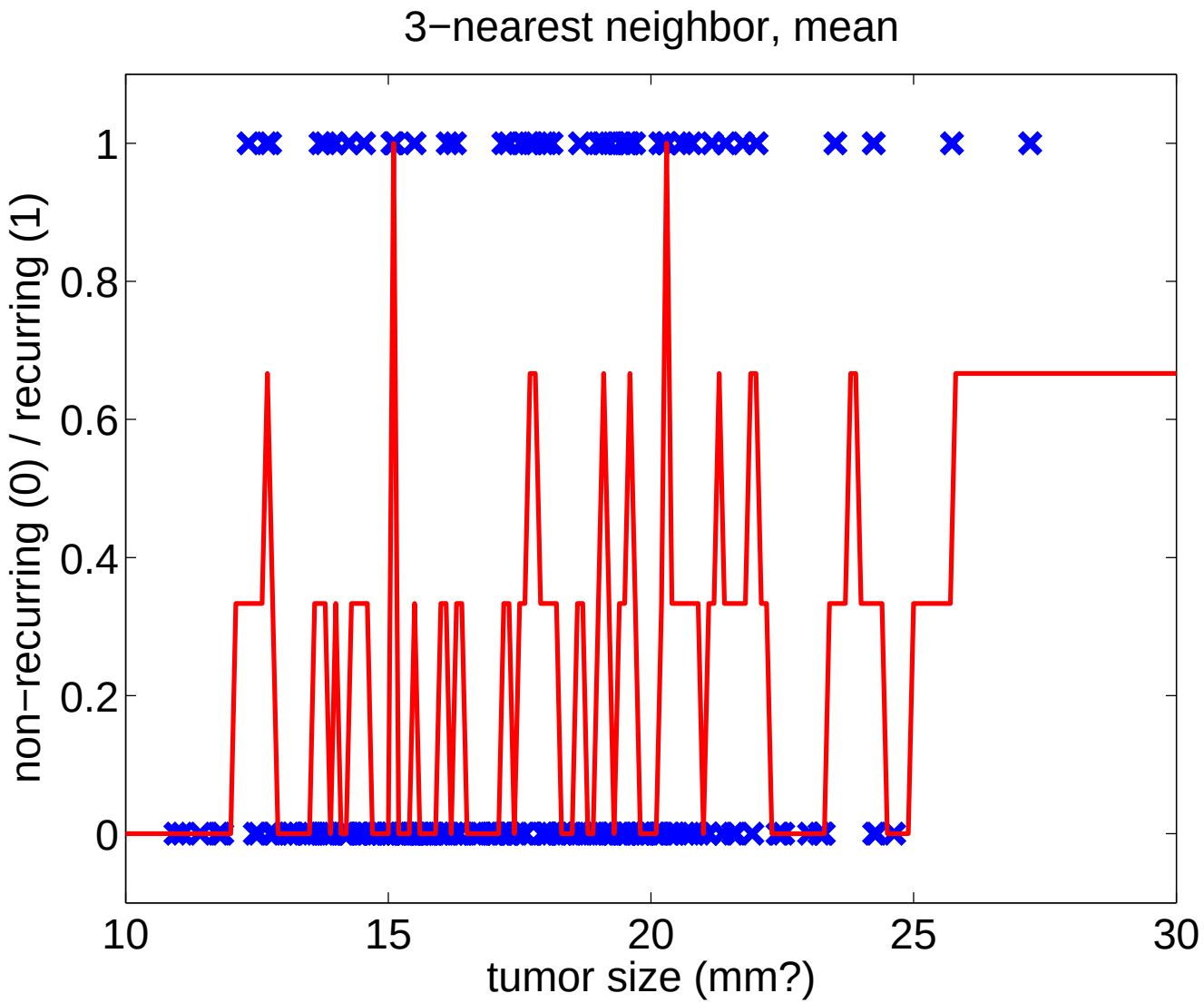

nearest neighbor
non−recurring (0) / recurring (1)
0
0.2
0.4
0.6
0.8
1
10
15
20
25
30
tumor size (mm?)

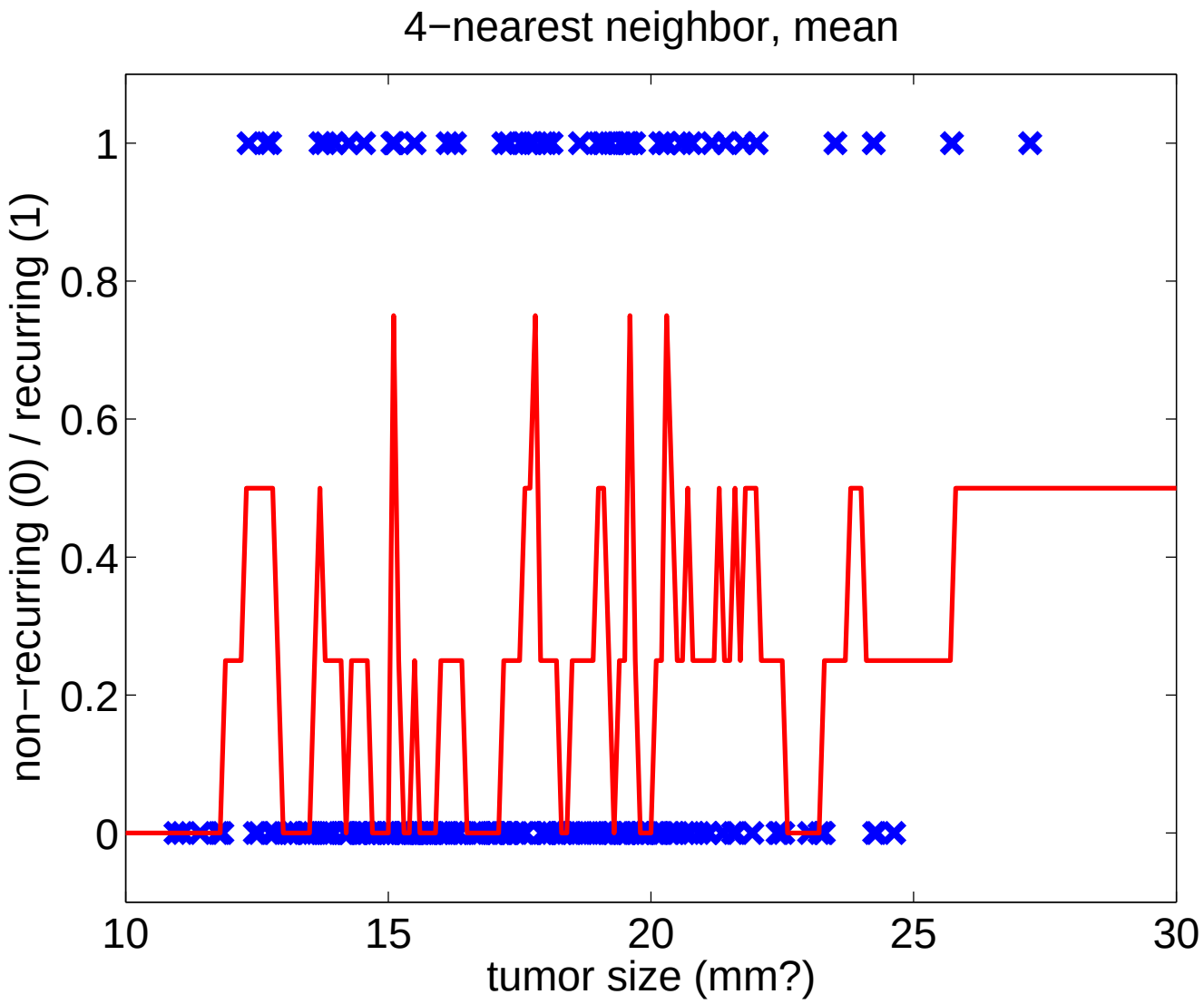
k-nearest neighbor

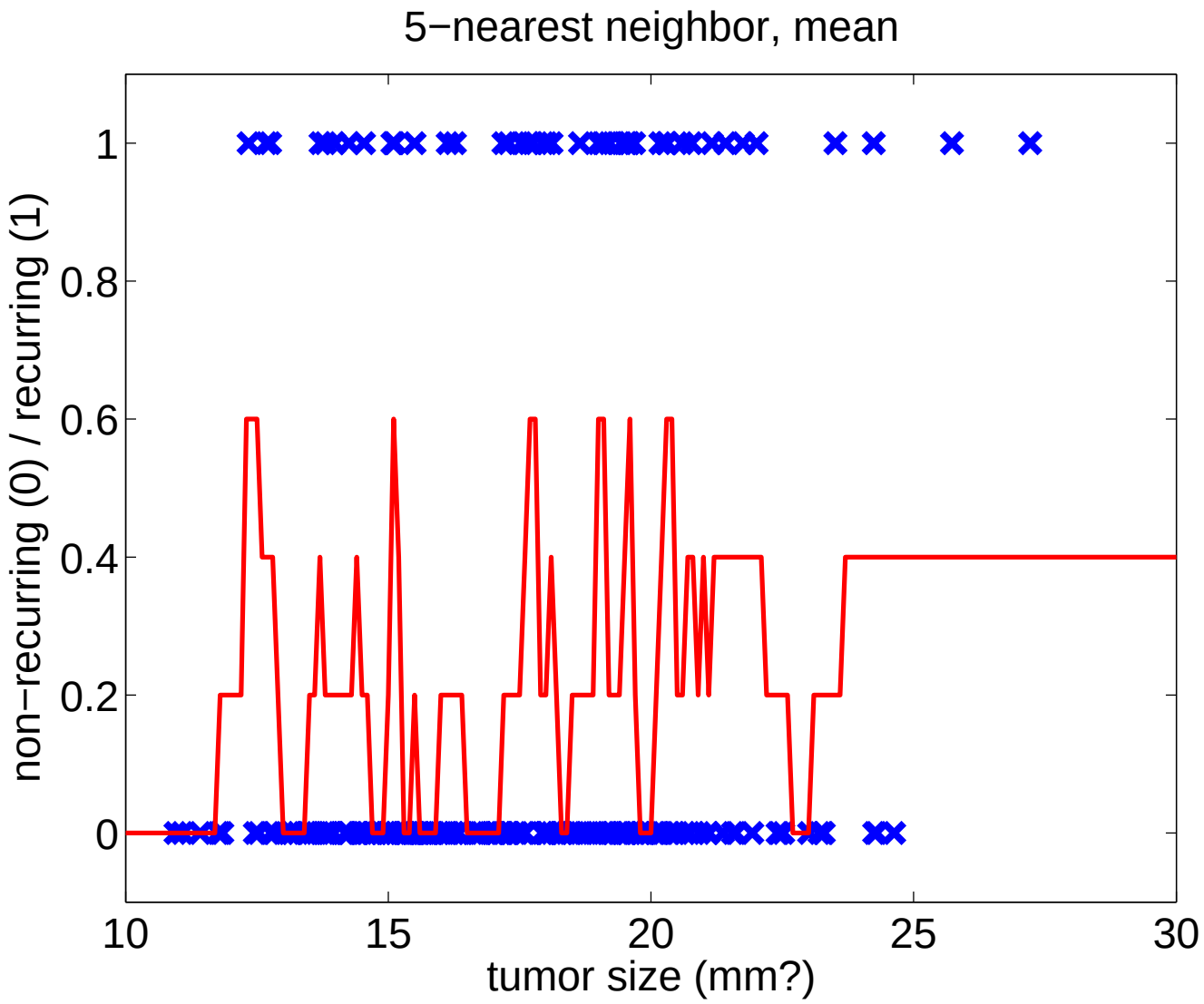
- Given: Training data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$, distance metric d on $\mathcal{X}$.
- Learning: Nothing to do!
- Prediction: for $\mathbf{x} \in \mathcal{X}$
  - Find the k nearest training samples to $\mathbf{x}$.
    Let their indeces be $i_1, i_2, \ldots, i_k$.
  - Predict $\mathbf{y}$ =mean/median/mode of $\{\mathbf{y}_{i_1}, \mathbf{y}_{i_2}, \ldots, \mathbf{y}_{i_k}\}$.

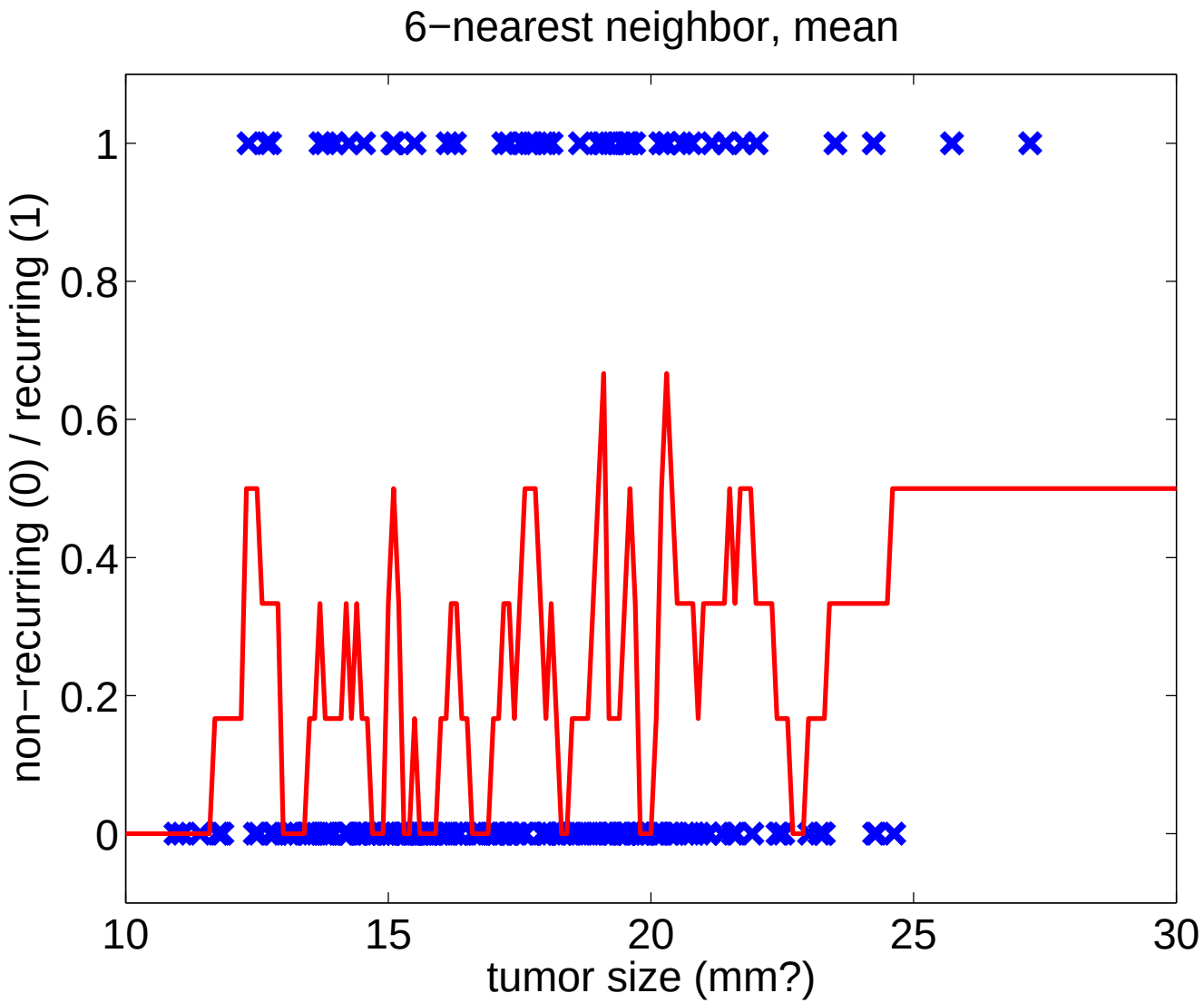


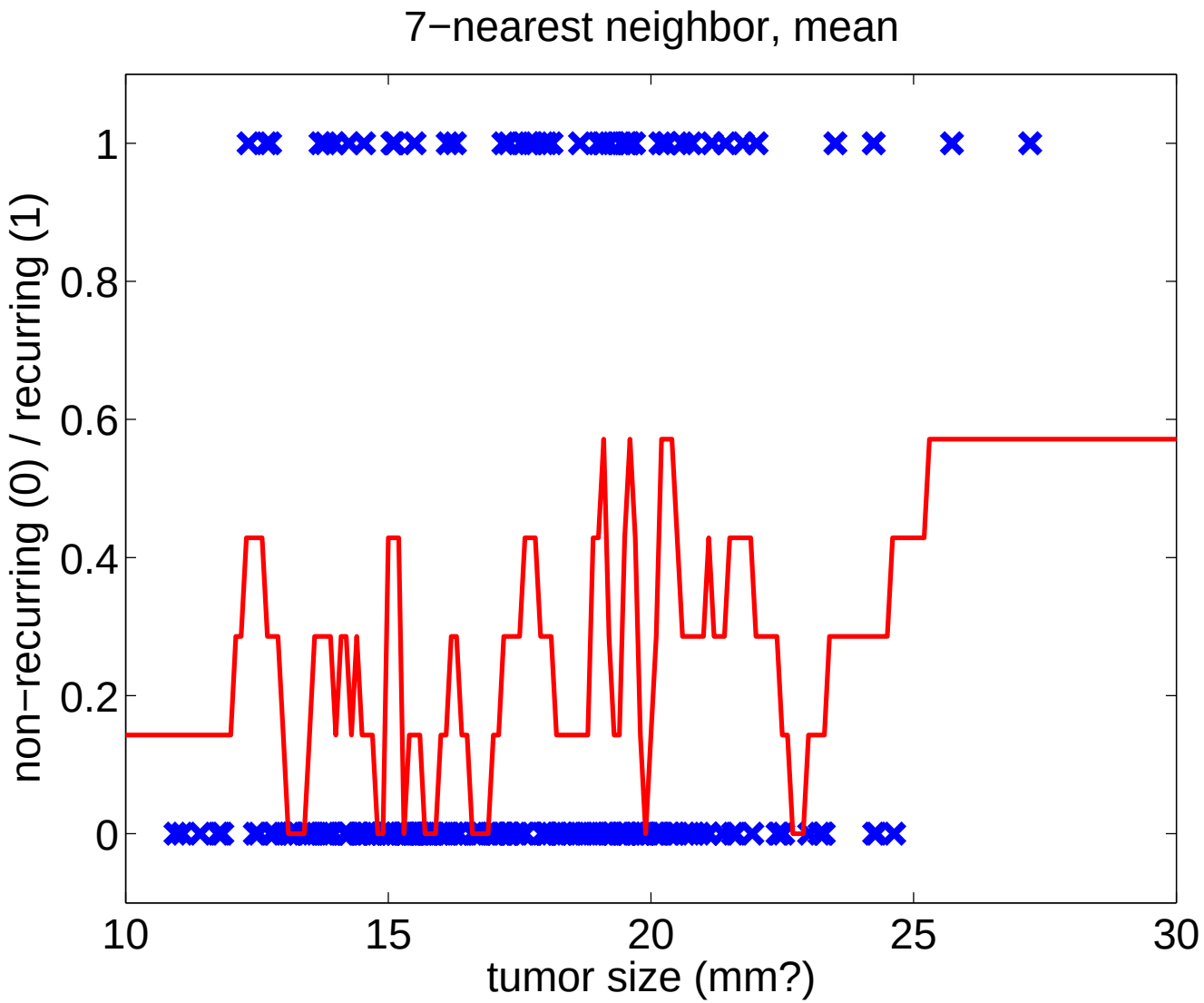
Smoother. . . but what does 0.5 mean?

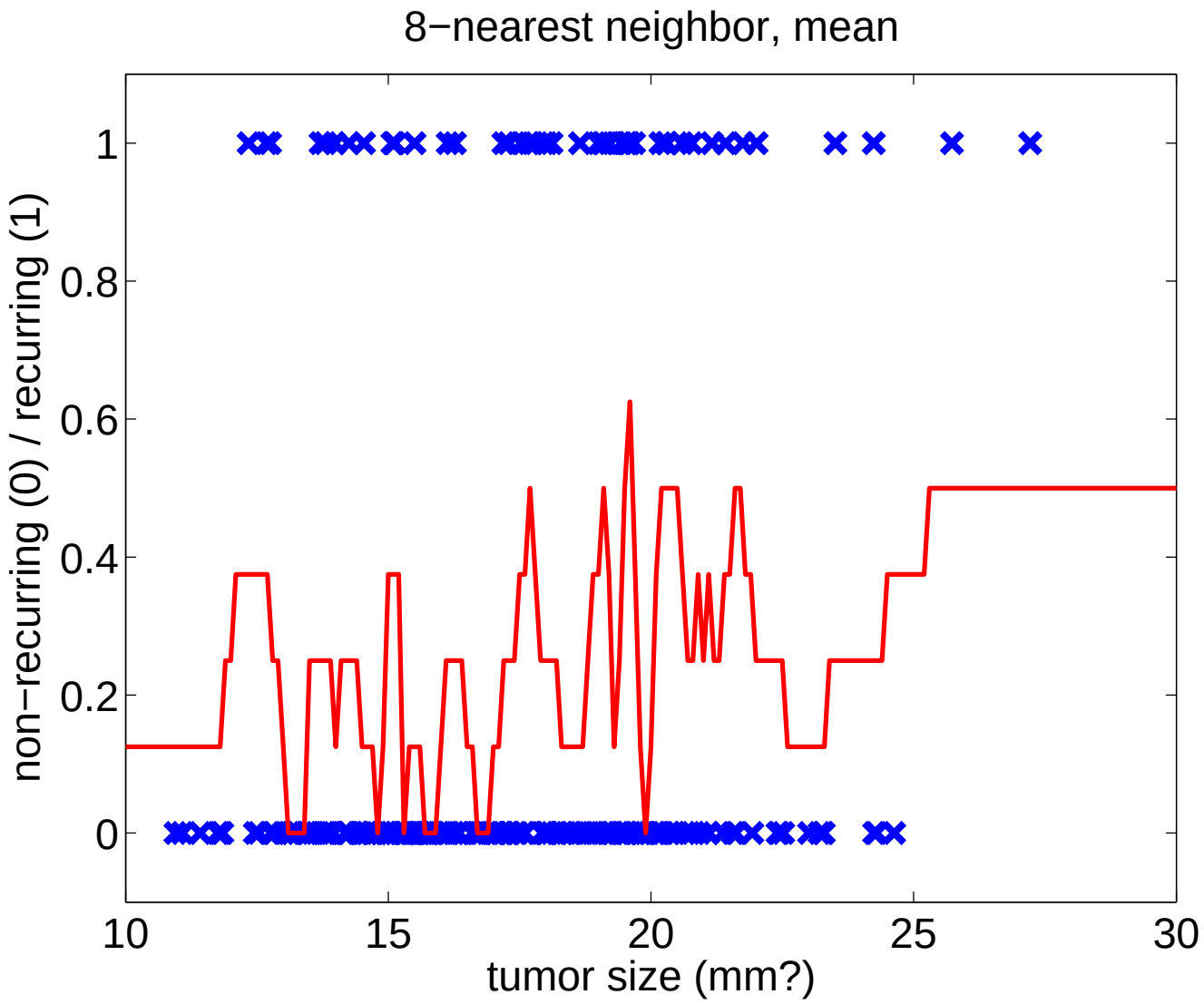

3−nearest neighbor, mean
non−recurring (0) / recurring (1)
tumor size (mm?)
0
0.2
0.4
0.6
0.8
1
10
15
20
25
30

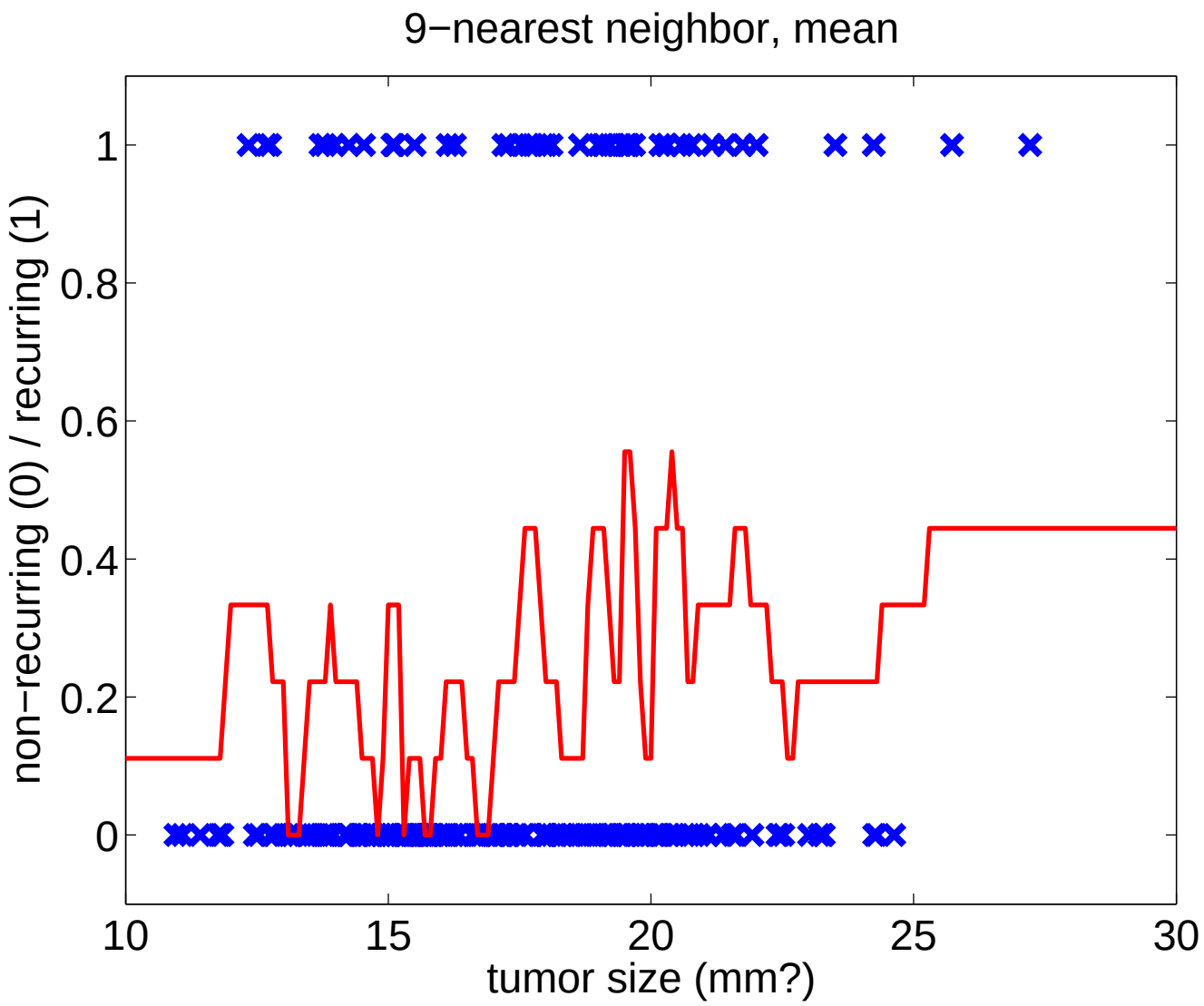

4−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)

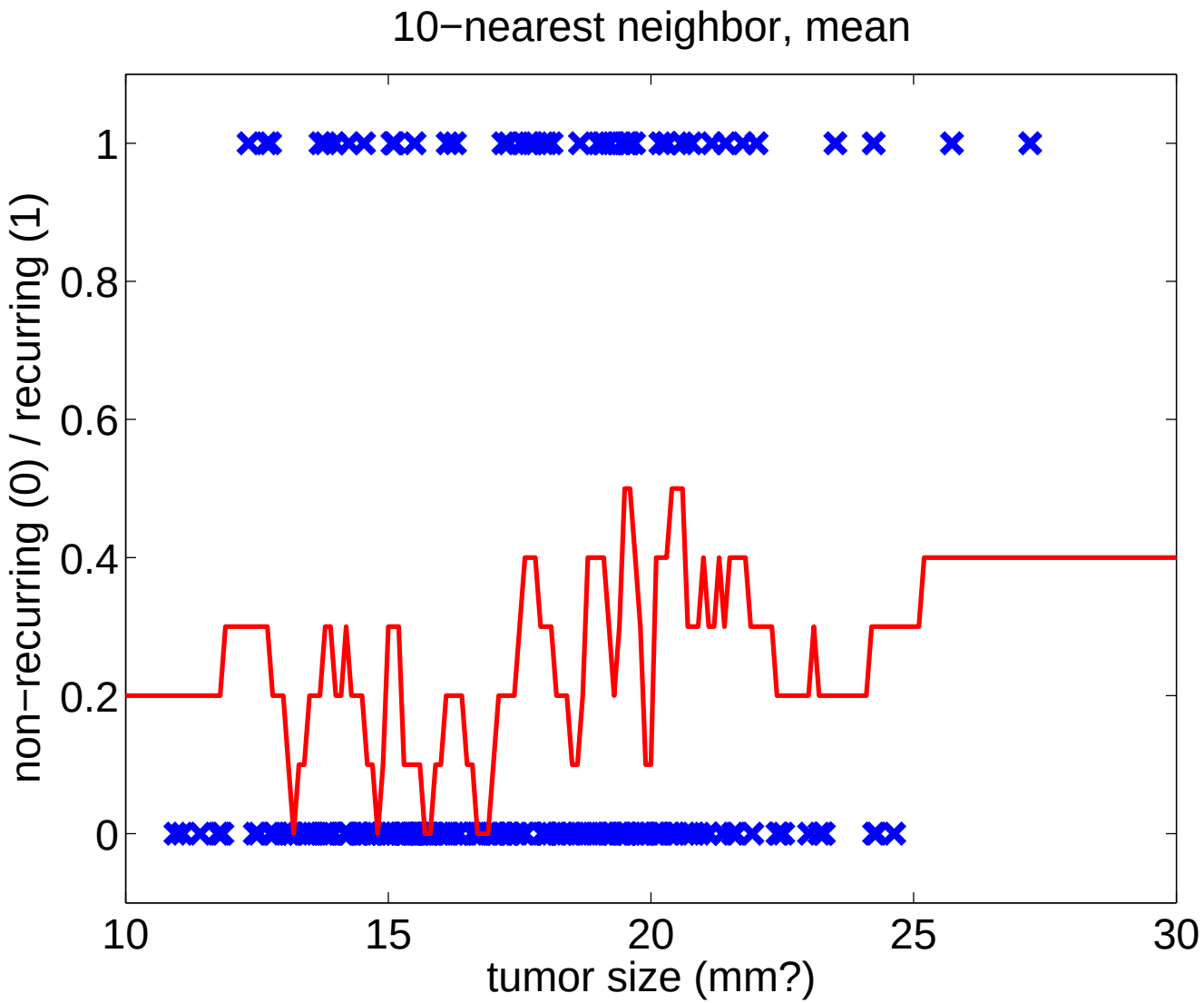

5−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)

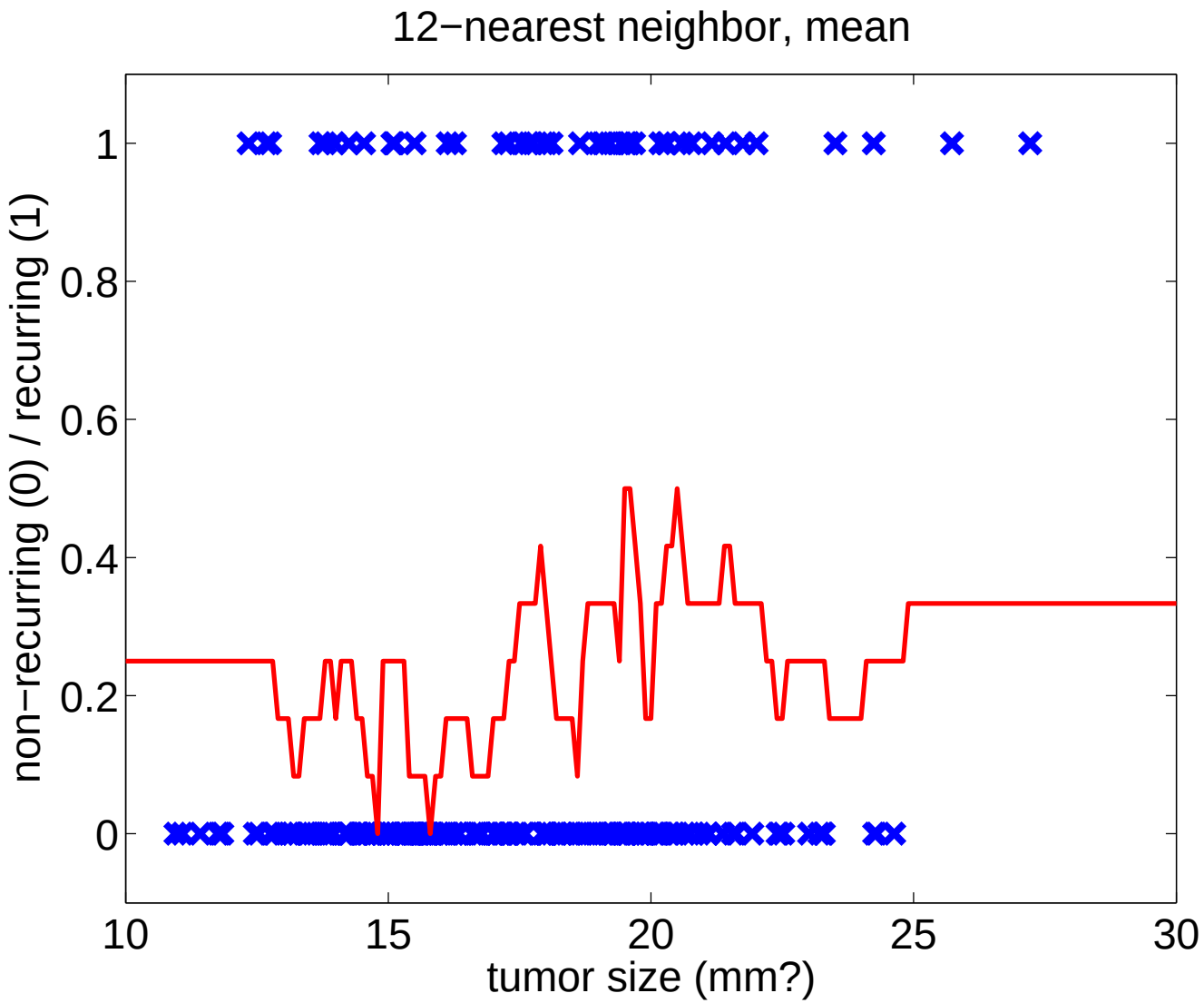

6−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)

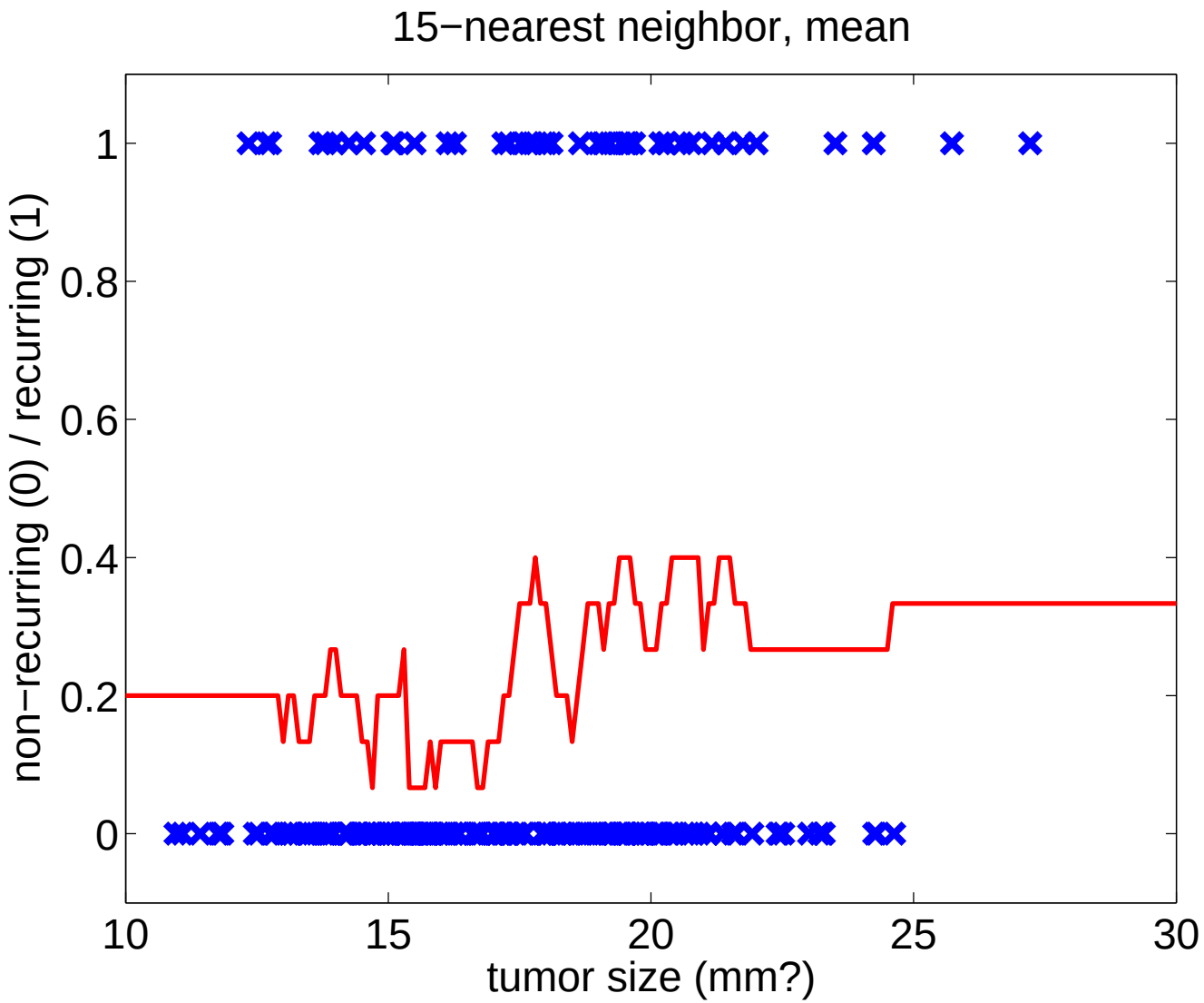

7−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)

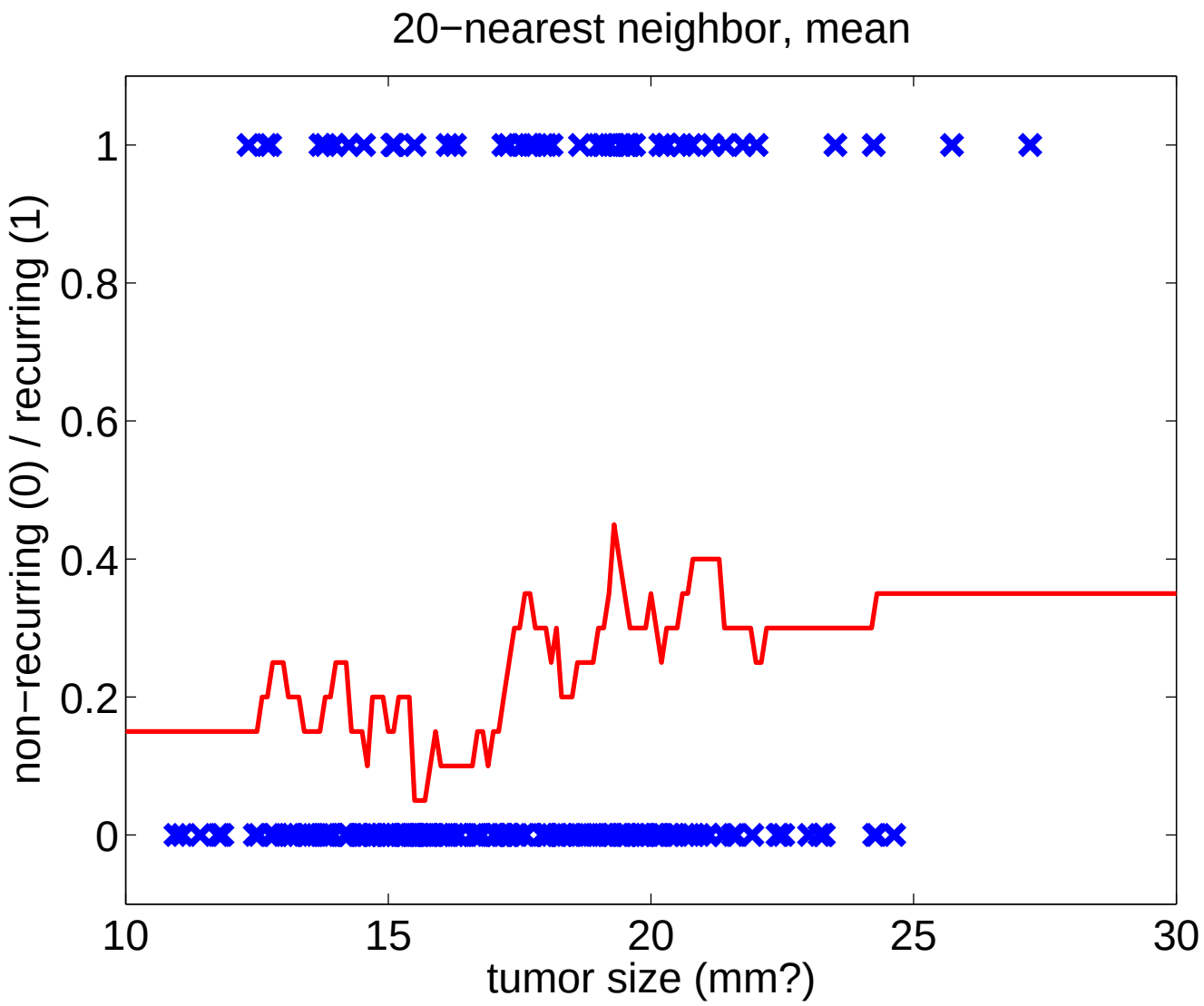

8−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)


9−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)


10−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)


12−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)


15−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)


20−nearest neighbor, mean
non−recurring (0) / recurring (1)
1
0.8
0.6
0.4
0.2
0
10
15
20
25
30
tumor size (mm?)

Questions

- What is the best choice of k?
- What are we optimizing? What should we be optimizing?

What we should be optimizing

- Suppose there is a true $f : \mathcal{X} \mapsto \mathcal{Y}$.
- For $\mathbf{x} \in \mathcal{X}$, $\mathbf{y}_1, \mathbf{y}_2 \in \mathcal{Y}$, let $\mathcal{E}_0(\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$ be the error/cost associated to predicting $\mathbf{y}_1$ for input $\mathbf{x}$ when the correct prediction is $\mathbf{y}_2$.
- Suppose sample inputs are drawn from a distribution P on $\mathcal{X}$.
- Then we can evaluate a candidate $\hat{f}$ by its expected prediction error:

$$\mathcal{E}(\hat{f}) = \int_{\mathbf{x}} \mathcal{E}_0(\mathbf{x}, \hat{f}(\mathbf{x}), f(\mathbf{x}))P(\mathbf{x})d\mathbf{x}$$

- Draw picture!
- Comments?

Unfortunately...

We cannot evaluate

$$\mathcal{E}(\hat{f}) = \int_{\mathbf{x}} \mathcal{E}_0(\mathbf{x}, \hat{f}(\mathbf{x}), f(\mathbf{x}))P(\mathbf{x})d\mathbf{x}$$

because

- We do not know f
- We do not know P
- We probably could not compute the integral even if we did know f and P.

Monte Carlo estimation of the integral

- Suppose our sample inputs $\mathbf{x}_i$ are drawn according to P.
- Suppose $\mathbf{y}_i = f(\mathbf{x}_i)$.
- Then for any $\hat{f}$ the *training error*

$$\mathcal{E}_{train} = \sum_{i=1}^{m} \mathcal{E}_0(\mathbf{x}_i, \hat{f}(\mathbf{x}_i), \mathbf{y}_i)$$

  is an unbiased estimate of the expected prediction error, $\mathcal{E}$ of $\hat{f}$.

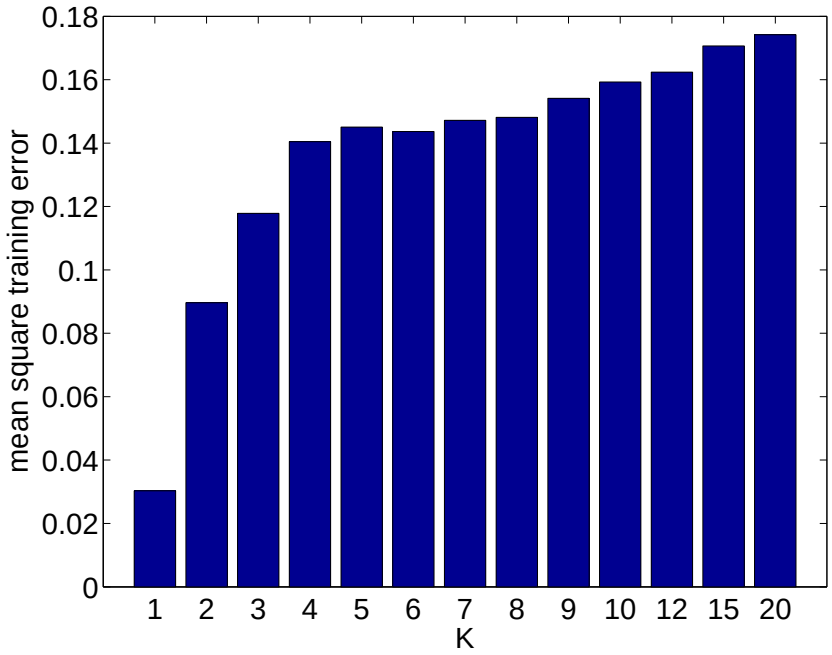
- Great! So we just choose $\hat{f}$ to minimize $\mathcal{E}_{train}$!

Application to choosing k

- Suppose we run k-nearest neighbor for $k = 1, 2, \ldots, m$, producing m candidate functions $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_m$.
- Compute $\mathcal{E}_{train}$ for each one.
- Which is the optimal k?

Application to choosing k

- Suppose we run k-nearest neighbor for $k = 1, 2, \ldots, m$, producing m candidate functions $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_m$.
- Compute $\mathcal{E}_{train}$ for each one.
- Which is the optimal k?



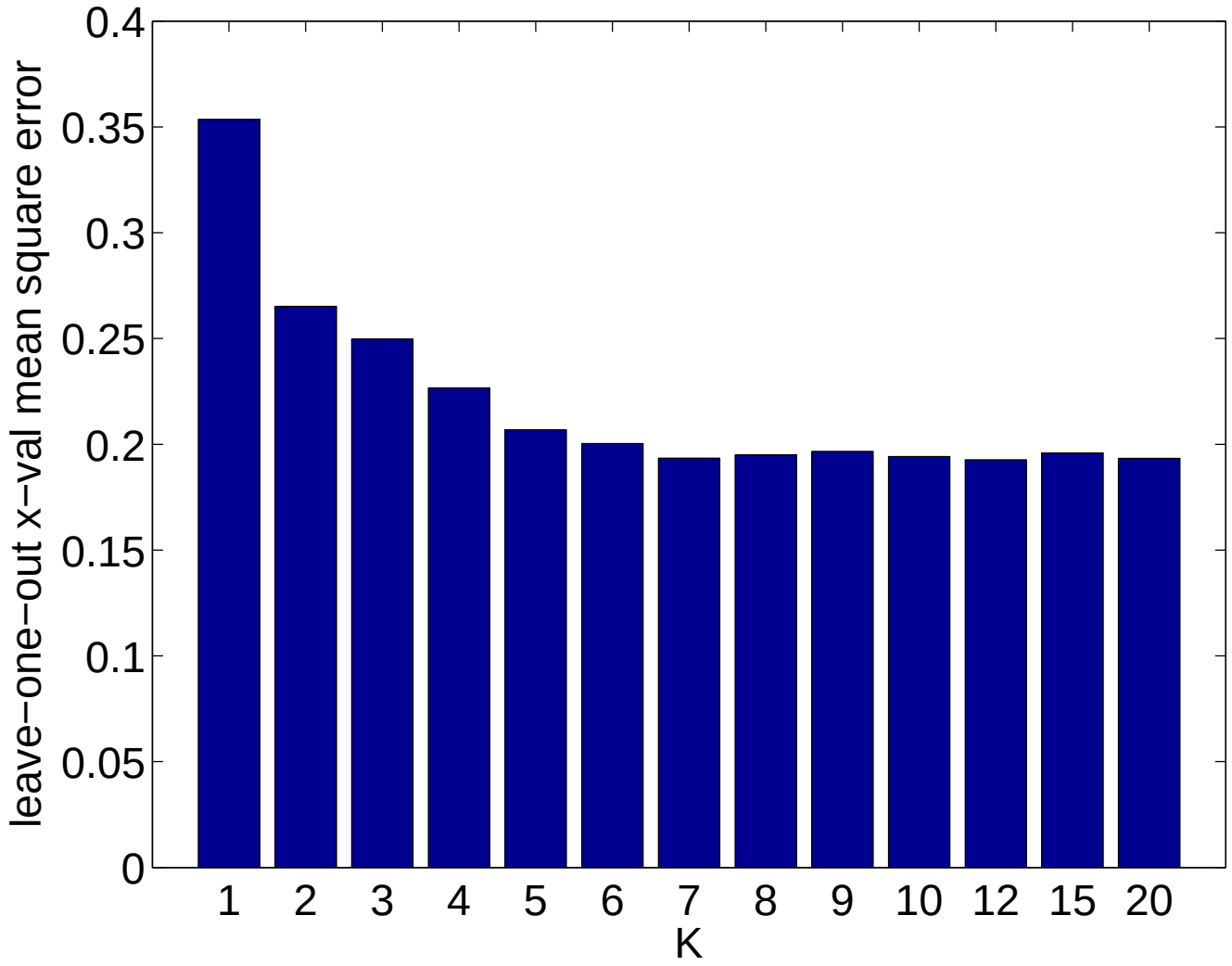
What is the problem?

What is the problem?

- There is no true f? (True f is stochastic/depends on other factors.)
- Noise in the $\mathbf{x}_i$ or $\mathbf{y}_y$?
- Noise in $\mathcal{E}_{train}$ due to small sample size?
- "Overfitting?" ($\mathcal{E}_{train}(\hat{f}_1) < \mathcal{E}_{train}(\hat{f}_2)$ but $\mathcal{E}(\hat{f}_1) > \mathcal{E}(\hat{f}_2)$.)

Validation sets

- Our goal is to build a predictor $\hat{f}$ with low error on new, unseen samples.
- We can simulate unseen samples by splitting our original data set into two: a (new, smaller) training set and a validation set.
- The new training set is used to learn $\hat{f}$, and the validation set is used to evaluate $\hat{f}$.
- Cross-validation. . .


leave−one−out x−val mean square error
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
1
2
3
4
5
6
7
8
9
10
12
15
20
K