PSTAT 131/231: Introduction to Statistical Machine Learning

Guo Yu

Lecture 3
Bias-Variance Tradeoff cont'd, k-Nearest Neighbors

ISL Chapter 2, 2.3 Introduction to R

ESL (for 231 students) Chapter 2, Chapter 7.1 - 7.3

Homework 1 out

Due **Monday, Oct 11,** 2021 at **23:59**

Data processing / analysis in R

Start early

Before we start…

Cloud based Rstudio Service https://pstat131.lsit.ucsb.edu/

Log in with your UCSB NetID

After several hours of inactivity, you will be logged out automatically

Save your work if you are not going to work for a while

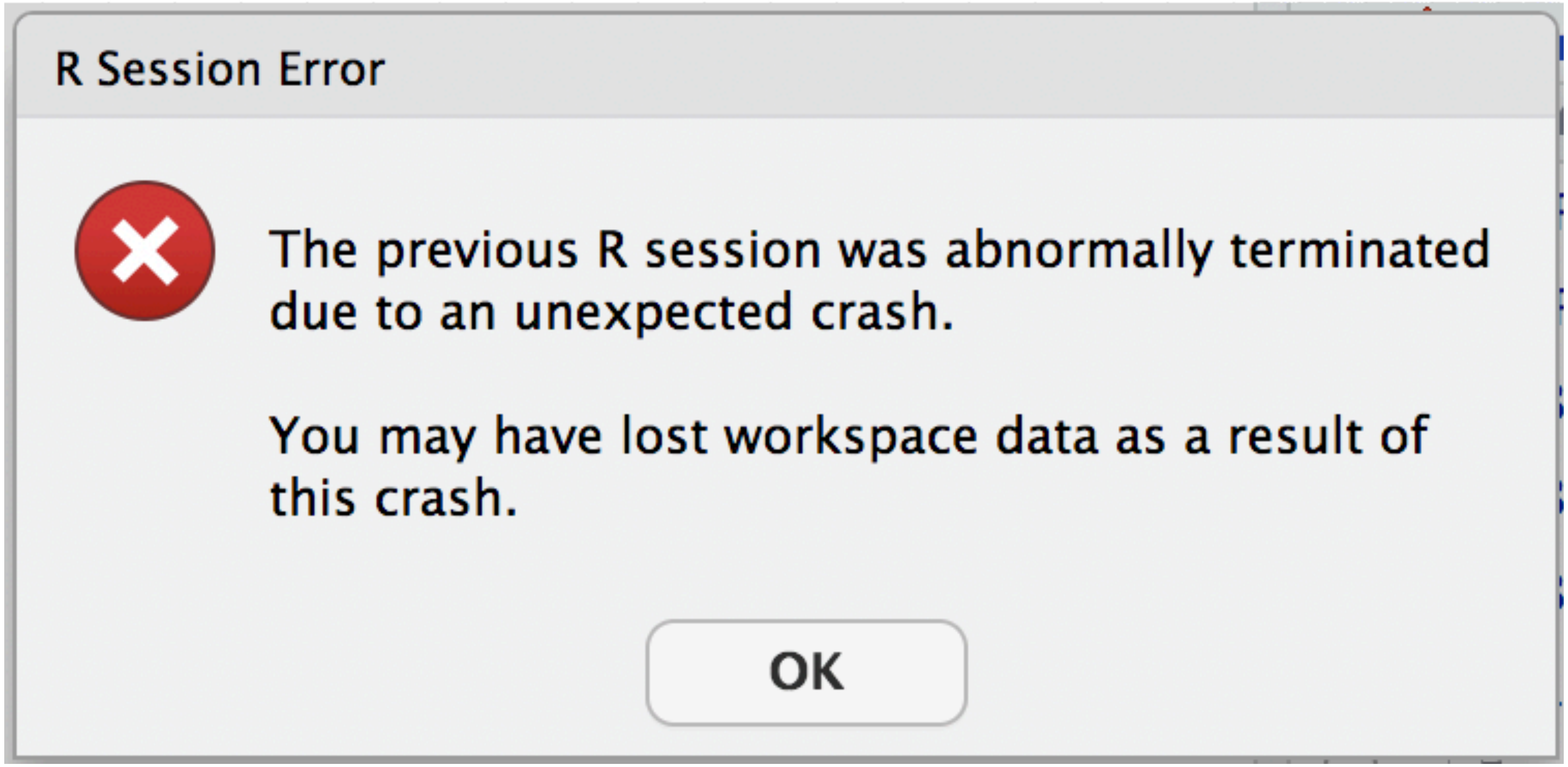
All R packages needed in this class have been installed

Please post on Piazza if you notice any issues, or want any package that is not installed

Occasionally it takes 1~2 mins to log in

Before we start…

Cloud based Rstudio Service https://pstat131.lsit.ucsb.edu/



This could occur when logging in. Don't worry about it.

Last time…

Machine Learning vs Statistical Machine Learning

Supervised Learning vs Unsupervised Learning

Regression vs Classification

Model flexibility vs interpretability

Training MSE vs Test MSE

Bias-variance decomposition —> Bias-variance tradeoff

Bias-variance decomposition

$$Y = \underbrace{f(X)}_{\text{non-random}} + \underbrace{\varepsilon}_{\text{zero-mean noise}}$$

$$\mathrm{E}\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = \mathrm{Var}(\hat{f}(\mathbf{x}_0)) + \left[\mathrm{Bias}(\hat{f}(\mathbf{x}_0))\right]^2 + \mathrm{Var}(\varepsilon),$$

Expected test MSE = Variance + Bias^2 + Irreducible error

$$= \mathrm{E}\left[\left(\hat{f}(\mathbf{x}_0) - \mathrm{E}\hat{f}(\mathbf{x}_0)\right)^2\right] + \left[\mathrm{E}\left[\hat{f}(\mathbf{x}_0)\right] - f(\mathbf{x}_0)\right]^2 + \mathrm{Var}(\varepsilon)$$

If we take

$$\hat{f}(\mathbf{x}_0) = \mathrm{E}\left[Y \mid X = \mathbf{x}_0\right]$$

then

$$\mathrm{E}\left[\left(\hat{f}(\mathbf{x}_0) - \mathrm{E}\hat{f}(\mathbf{x}_0)\right)^2\right] = 0 \quad \textbf{and} \quad \left[\mathrm{E}\left[\hat{f}(\mathbf{x}_0)\right] - f(\mathbf{x}_0)\right]^2 = 0$$

Bias-variance decomposition

$$Y = f(X) + \varepsilon$$

non-random zero-mean noise

If we take

$$\hat{f}(\mathbf{x}_0) = \mathrm{E}\left[Y \,|\, X = \mathbf{x}_0\right]$$

$$\mathrm{E}\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = \mathrm{Var}(\hat{f}(\mathbf{x}_0)) + \left[\mathrm{Bias}(\hat{f}(\mathbf{x}_0))\right]^2 + \mathrm{Var}(\varepsilon),$$

Expected test MSE $=$ Variance $+$ Bias^2 $+$ Irreducible error

$$= \underbrace{\mathrm{E}\left[\left(\hat{f}(\mathbf{x}_0) - \mathrm{E}\hat{f}(\mathbf{x}_0)\right)^2\right] + \left[\mathrm{E}\left[\hat{f}(\mathbf{x}_0)\right] - f(\mathbf{x}_0)\right]^2}_{\text{minimized!}} + \mathrm{Var}(\varepsilon)$$

$$= \mathrm{Var}(\varepsilon)$$

Irreducible error

Bias-variance decomposition

$$Y = f(X) + \varepsilon$$

non-random zero-mean noise

The "best" we can do

$$\hat{f}(\mathbf{x}_0) = \mathrm{E}\left[Y \,|\, X = \mathbf{x}_0\right]$$

Unknown in practice!!

Because the joint distribution of (X, Y) is unknown in practice

Today…

Machine Learning vs **Statistical Machine Learning**

Supervised Learning vs **Unsupervised Learning**

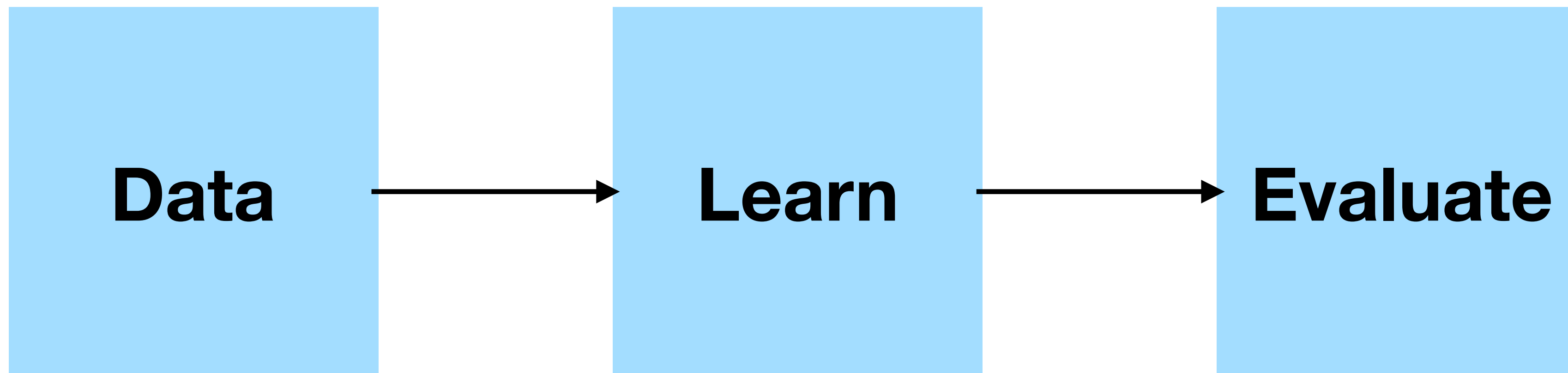
Regression vs **Classification**

Training error rate vs Test error rate

Bias-variance tradeoff

kNN methods

Classification setting



Training set

$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$

$\hat{f}$

$y_1, \ldots, y_n$: **class label (qualitative), i.e.,**

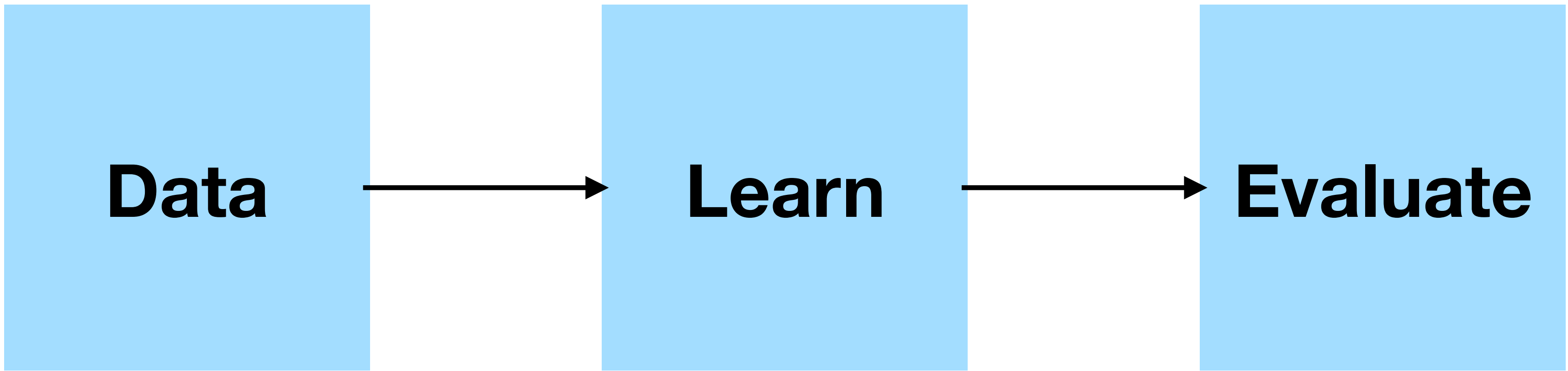
$y_1 = \text{dog}, y_2 = \text{cat}, y_3 = \text{cat}, \ldots, y_n = \text{dog}$

$\hat{y}_1, \ldots, \hat{y}_n$: **predicted class label using** $\hat{f}$**, i.e.,**

$\hat{y}_1 = \text{dog}, \hat{y}_2 = \text{dog}, \hat{y}_3 = \text{cat}, \ldots, \hat{y}_n = \text{cat}$

$\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$

Training error rate



Training set

$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$

$\hat{f}$

how well does $\hat{f}$ learn?

Indicator function

$$I\left(y_i \neq \hat{y}_i\right) = \begin{cases} 1 & \textbf{if} \quad y_i \neq \hat{y}_i \\ 0 & \textbf{if} \quad y_i = \hat{y}_i \end{cases}$$

$$\text{Training error rate} = \frac{1}{n} \sum_{i=1}^{n} I\left(y_i \neq \hat{y}_i\right)$$

predicted class label that $\hat{f}$ gives for the ith training observation

Training Error Rate

fraction of **incorrect** classifications in **training** set

Test error rate

Data → **Learn** → **Evaluate**

Training set
$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$

$\hat{f}$

how well does $\hat{f}$ learn?

Again, we are less interested in how $\hat{f}$ performs on training set

Consider test set $\{(\tilde{\mathbf{x}}_1, \tilde{y}_1), (\tilde{\mathbf{x}}_2, \tilde{y}_2), \ldots, (\tilde{\mathbf{x}}_m, \tilde{y}_m)\}$, not seen or used to train $\hat{f}$

$$\text{Test error rate} = \frac{1}{n}\sum_{i=1}^{n} I\left(\tilde{y}_i \neq \hat{\tilde{y}}_i\right)$$

predicted class label that $\hat{f}$ gives for the ith test observation

Test Error Rate

fraction of **incorrect** classifications in **test** set

Training error rate vs Test error rate

Training set

$$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$$

Test set

$$\{(\tilde{\mathbf{x}}_1, \tilde{y}_1), (\tilde{\mathbf{x}}_2, \tilde{y}_2), \ldots, (\tilde{\mathbf{x}}_m, y_m)\}$$

Training error rate

$$\frac{1}{n}\sum_{i=1}^{n} I\left(y_i \neq \hat{y}_i\right)$$

predicted class label that $\hat{f}$ gives for the ith training observation

Test error rate

$$\frac{1}{n}\sum_{i=1}^{n} I\left(\tilde{y}_i \neq \hat{\tilde{y}}_i\right)$$

predicted class label that $\hat{f}$ gives for the ith test observation

$\hat{f}$ **is obtained on the training set!**

Same discussion as in the regression setting

A method that has lowest training error rate does NOT necessarily imply that it also has the lowest test error rate

Higher model flexibility $\rightarrow$ lower training error rate

Higher model flexibility $\nrightarrow$ lower test error rate

Overfitting = Large test error rate + small training error rate

Estimating test error rate

Computing training error rate is easy… computing test error rate is NOT easy!

What should we do when test set is not available?

Cross-validation:

a method for estimating test error rate using only training data!

later in the course…

Bias-variance tradeoff

Still want to **simultaneously** achieve **low variance** and **low bias**

A simple model → high bias + low variance

A flexible model → low bias + high variance

Challenge: find a method for which both the variance and bias are low

The Bayes classifier

In classification setting, the "best" we can do

$$\hat{y}_0 = \operatorname{argmax}_j \left\{ \text{Prob}\left(Y = j \,|\, X = \mathbf{x}_0\right) \right\}$$

conditional probability of $Y = j$,
given the observed predictor $\mathbf{x}_0$

Bayes classifier: assigns each observation to the most likely class, given its predictor values

For example, we want to classify species of an animal (dog, cat, bird) given its picture

$$\text{Prob}\left(Y = \textbf{dog} \,|\, X = \mathbf{x}_0\right) = 0.5$$
$$\text{Prob}\left(Y = \textbf{cat} \,|\, X = \mathbf{x}_0\right) = 0.3$$
$$\text{Prob}\left(Y = \textbf{bird} \,|\, X = \mathbf{x}_0\right) = 0.2$$

$$\longrightarrow \quad \hat{y}_0 = \textbf{dog}$$

In a two-class ($Y = 1$ or $Y = 2$) classification problem,

$$\hat{y}_0 = \begin{cases} 1 & \textbf{if } \text{Prob}\left(Y = 1 \,|\, X = \mathbf{x}_0\right) > 0.5 \\ 2 & \textbf{if } \text{Prob}\left(Y = 1 \,|\, X = \mathbf{x}_0\right) \leq 0.5 \end{cases}$$

The Bayes classifier

$$\hat{y}_0 = \text{argmax}_j \left\{ \text{Prob}\left(Y = j \,|\, X = \mathbf{x}_0\right) \right\}$$

Each circle is an observation

location: predictor X

color: class label Y, $Y = 1$ vs. $Y = 2$

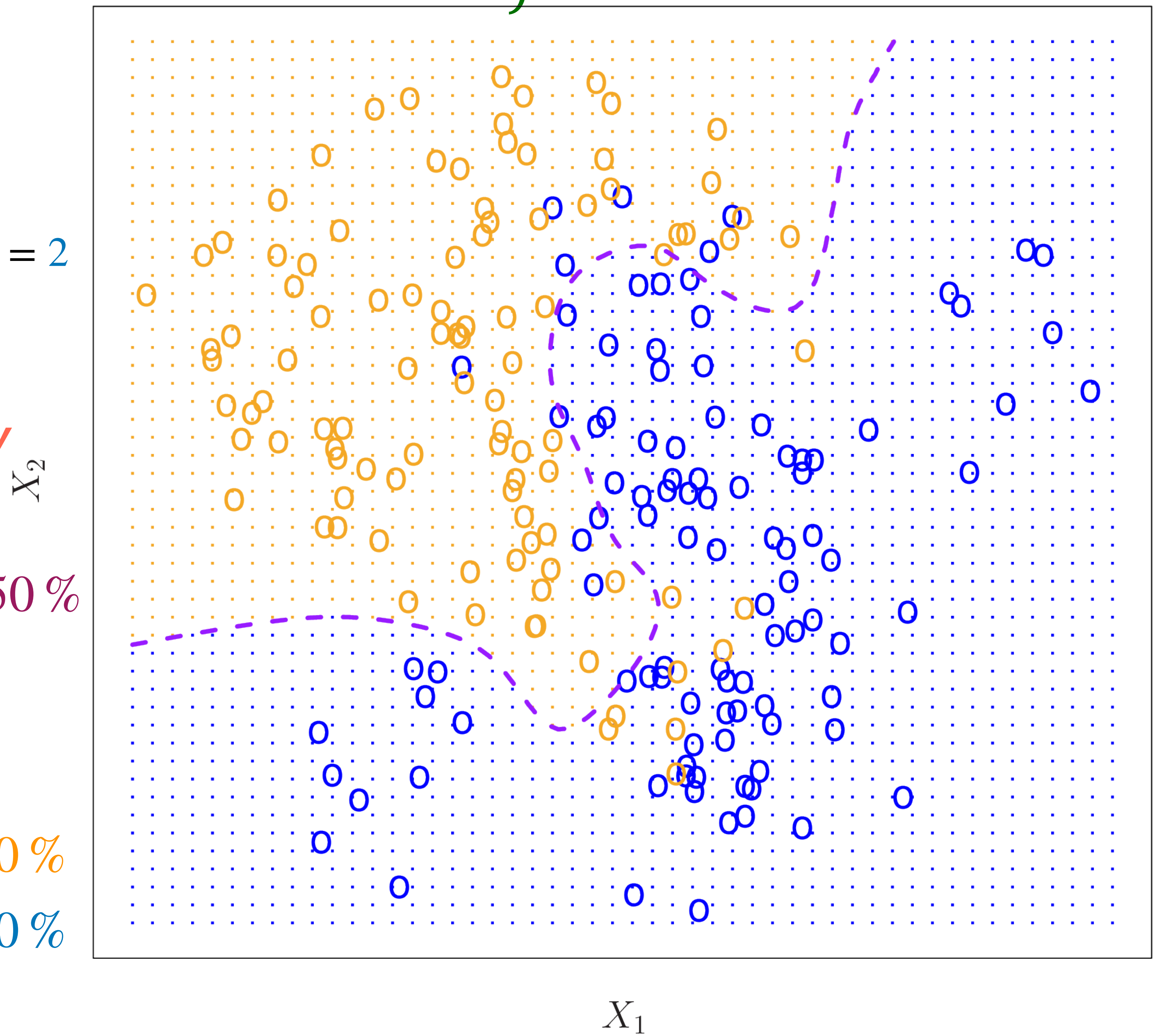
Purple dashed line:

Bayesian decision boundary

$$\text{Prob}\left(Y = 1 \,|\, X = \mathbf{x}_0\right)$$
$$= \text{Prob}\left(Y = 2 \,|\, X = \mathbf{x}_0\right) = 50\,\%$$

Areas:

$$\text{Prob}\left(Y = 1 \,|\, X = \mathbf{x}_0\right) > 50\,\%$$
$$\text{Prob}\left(Y = 2 \,|\, X = \mathbf{x}_0\right) > 50\,\%$$



The Bayes classifier

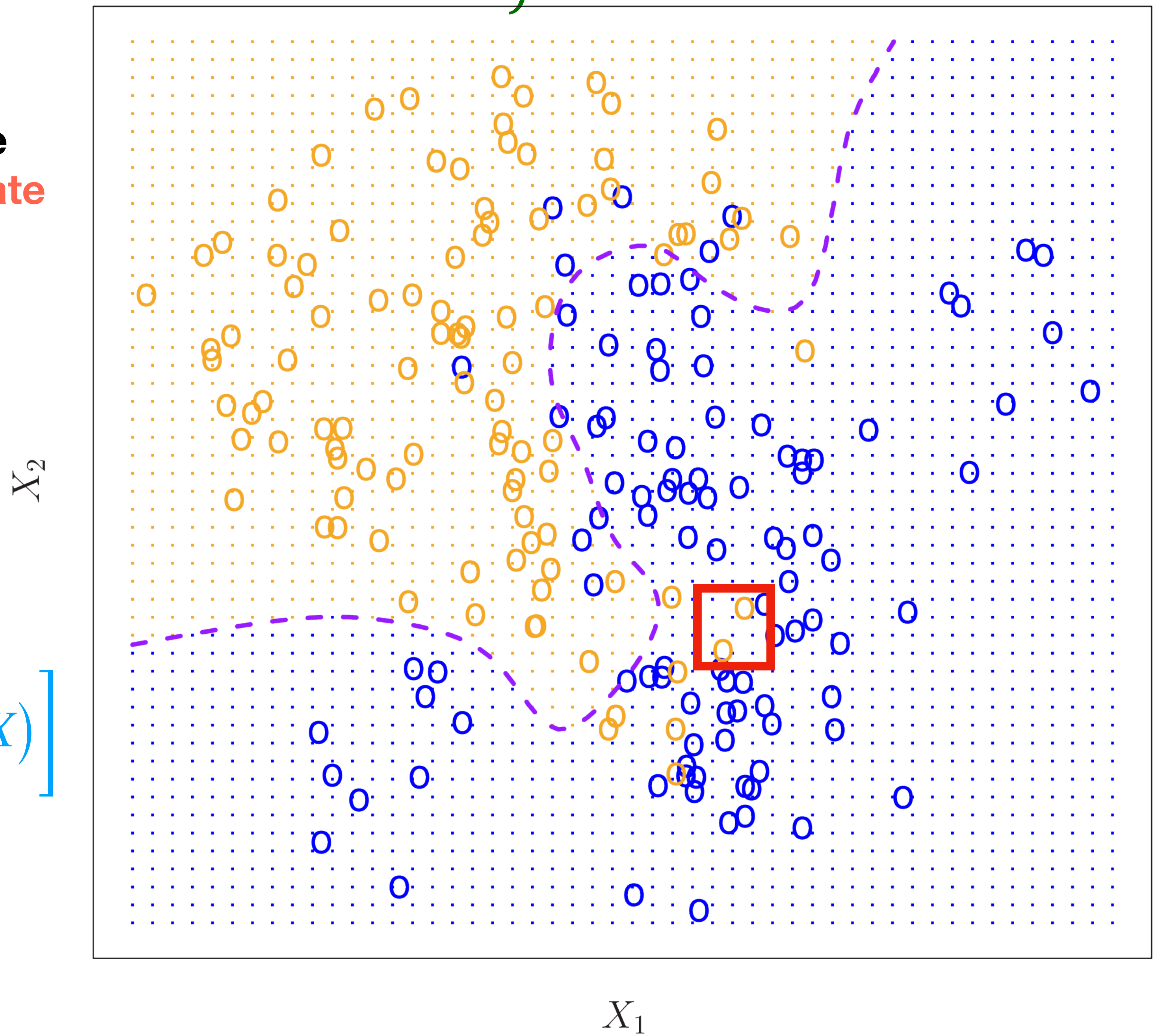
$$\hat{y}_0 = \text{argmax}_j \left\{ \text{Prob}\left(Y = j \,|\, X = \mathbf{x}_0\right) \right\}$$

The Bayes is best in the sense that it produces the lowest possible test error rate

The Bayes classifier still makes mistakes!!

Bayes error rate

$$1 - \mathrm{E}\left[\max_j \text{Prob}\left(Y = j \,|\, X\right)\right]$$



Regression

X

quantitative Y

Training/Test
Mean Squared Error (MSE)

$$\hat{f}(\mathbf{x}_0) = \mathrm{E}\left[Y \mid X = \mathbf{x}_0\right]$$

$$\mathrm{Var}(\varepsilon)$$

Classification

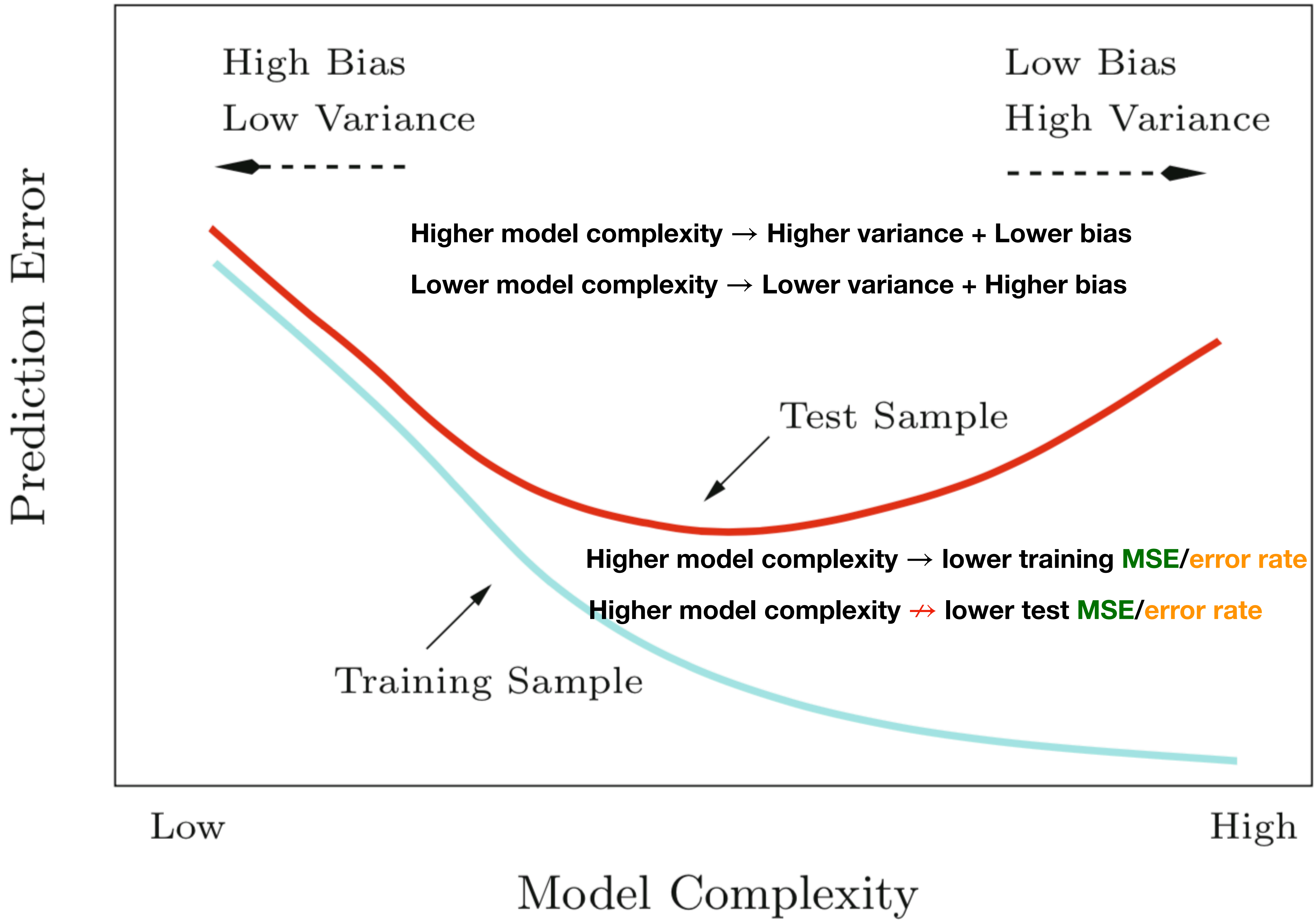
X

qualitative Y

Training/Test
Error Rate

$$\hat{y}_0 = \mathrm{argmax}_j \left\{ \mathrm{Prob}\left(Y = j \mid X = \mathbf{x}_0\right) \right\}$$

$$1 - \mathrm{E}\left[\max_j \mathrm{Prob}\left(Y = j \mid X\right)\right]$$


High Bias
Low Variance
Low Bias
High Variance
Higher model complexity → Higher variance + Lower bias
Lower model complexity → Lower variance + Higher bias
Test Sample
Higher model complexity → lower training MSE/error rate
Higher model complexity ↛ lower test MSE/error rate
Training Sample
Prediction Error
Low
High
Model Complexity

K-Nearest Neighbors

Regression

$$\hat{f}(\mathbf{x}_0) = \mathrm{E}\left[Y \,|\, X = \mathbf{x}_0\right]$$

Classification

$$\hat{y}_0 = \mathrm{argmax}_j \left\{ \mathrm{Prob}\left(Y = j \,|\, X = \mathbf{x}_0\right) \right\}$$

In practice, we never know the conditional distribution of Y given X

Estimate!!

Idea: look at the K-nearest neighbors of $\mathbf{x}_0$

K-Nearest Regression

$$\hat{f}(\mathbf{x}_0) = \mathrm{E}\left[Y \,|\, X = \mathbf{x}_0\right]$$

Estimation idea: look at the K-nearest neighbors of $\mathbf{x}_0$

$$\mathcal{N}_0 = \left\{ i : \mathbf{x}_i \textbf{ is among the K nearest neighbors of } \mathbf{x}_0 \right\}$$

$$\hat{f}_K(\mathbf{x}_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i$$

K-Nearest Regression: bias-variance tradeoff

$$\hat{f}_K(\mathbf{x}_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i$$

$$\underbrace{Y = f(X)}_{\text{non-random}} + \underbrace{\varepsilon}_{\text{zero-mean noise}}$$

For fixed $(\mathbf{x}_0, y_0)$

$$\mathrm{E}\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = \mathrm{Var}(\hat{f}(\mathbf{x}_0)) + \left[\mathrm{Bias}(\hat{f}(\mathbf{x}_0))\right]^2 + \mathrm{Var}(\varepsilon),$$

$$= \frac{\mathrm{Var}(\varepsilon)}{k} + \left[f(\mathbf{x}_0) - \frac{1}{k}\sum_{i \in \mathcal{N}_0} f(\mathbf{x}_i)\right]^2 + \mathrm{Var}(\varepsilon),$$

k increase	**decrease**	**increase**	**beyond our control**
k decrease	**increase**	**decrease**	

K-Nearest Classification

$$\hat{y}_0 = \text{argmax}_j \left\{ \text{Prob}\left(Y = j \,|\, X = \mathbf{x}_0 \right) \right\}$$

Estimation idea: look at the K-nearest neighbors of $\mathbf{x}_0$

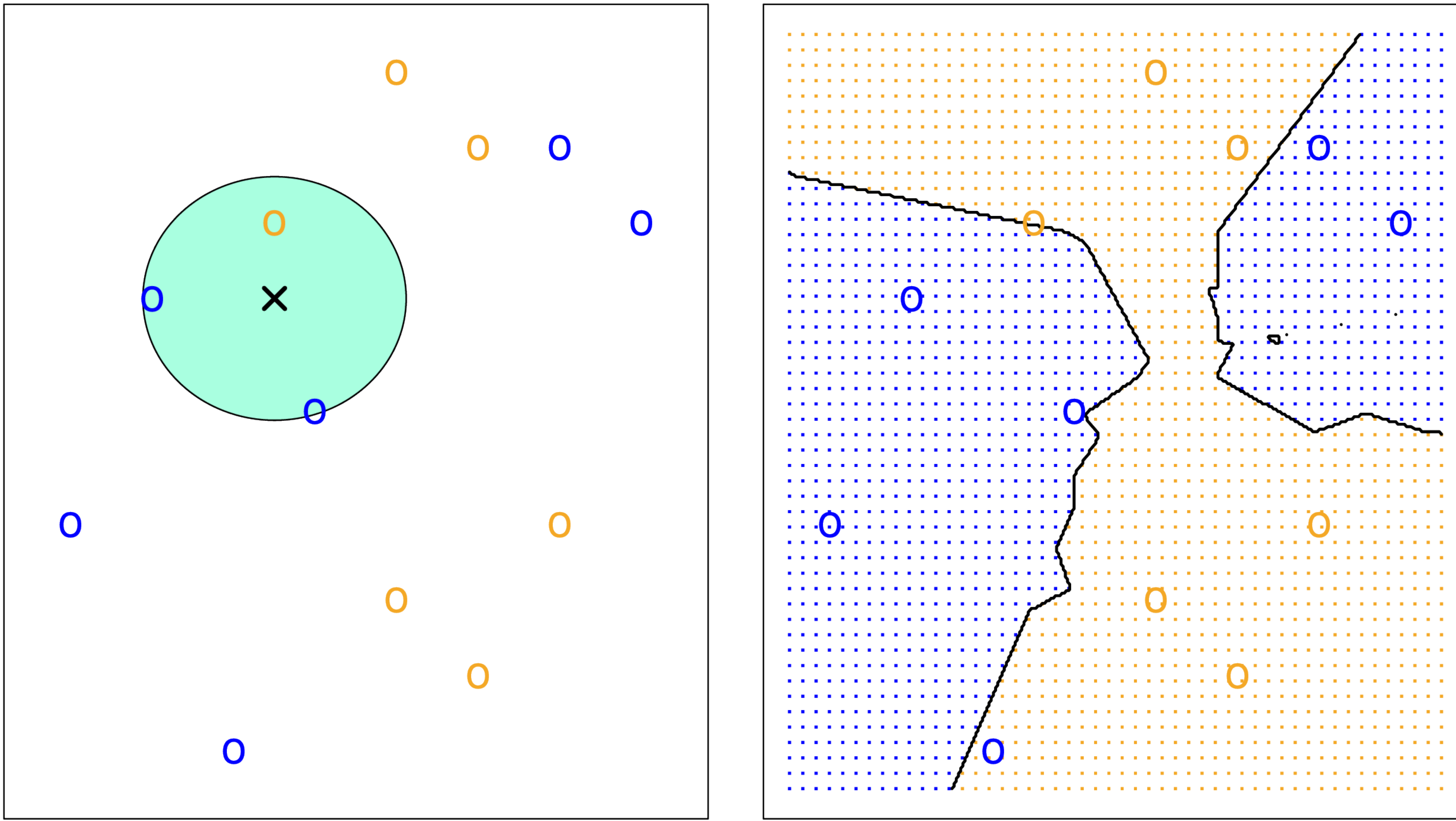
$\mathcal{N}_0 = \{ i : \mathbf{x}_i$ **is among the K nearest neighbors of** $\mathbf{x}_0 \}$

$$\hat{y}_K = \text{argmax}_j \left\{ \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j) \right\}$$

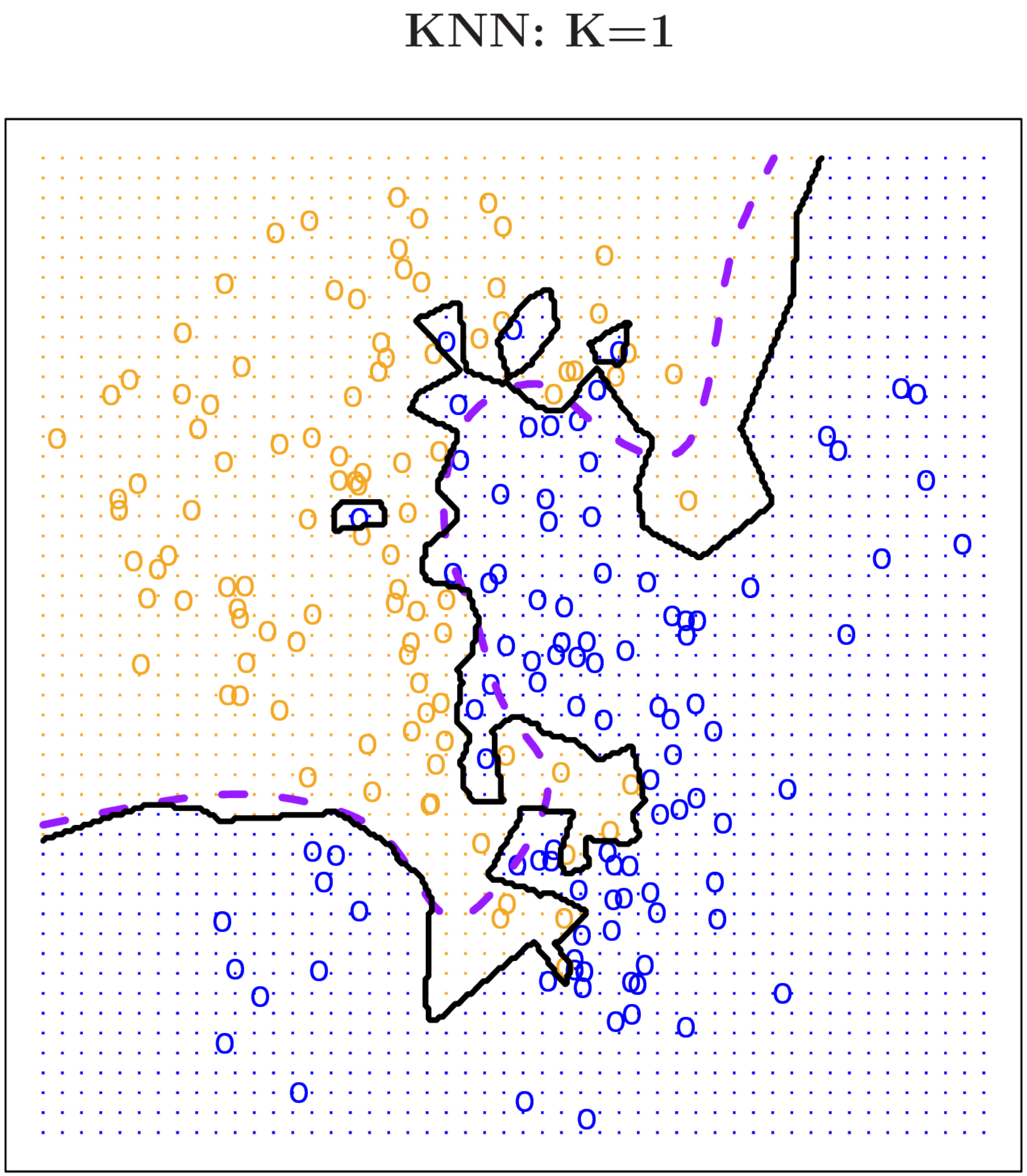
K-Nearest Classification

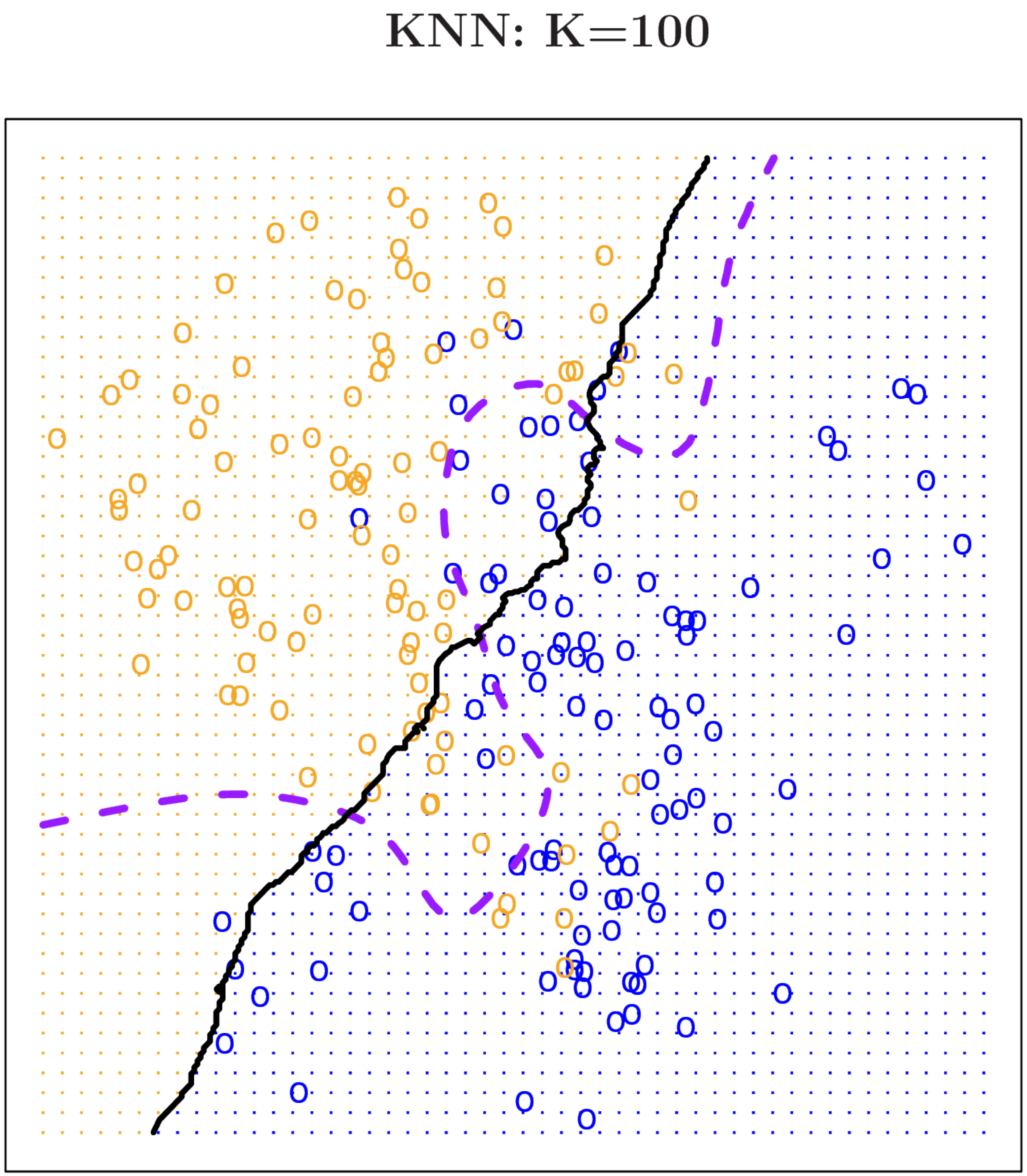
$$\hat{y}_K = \text{argmax}_j \left\{ \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j) \right\}$$

Take K = 3 as an example…



K-Nearest Classification: bias-variance tradeoff

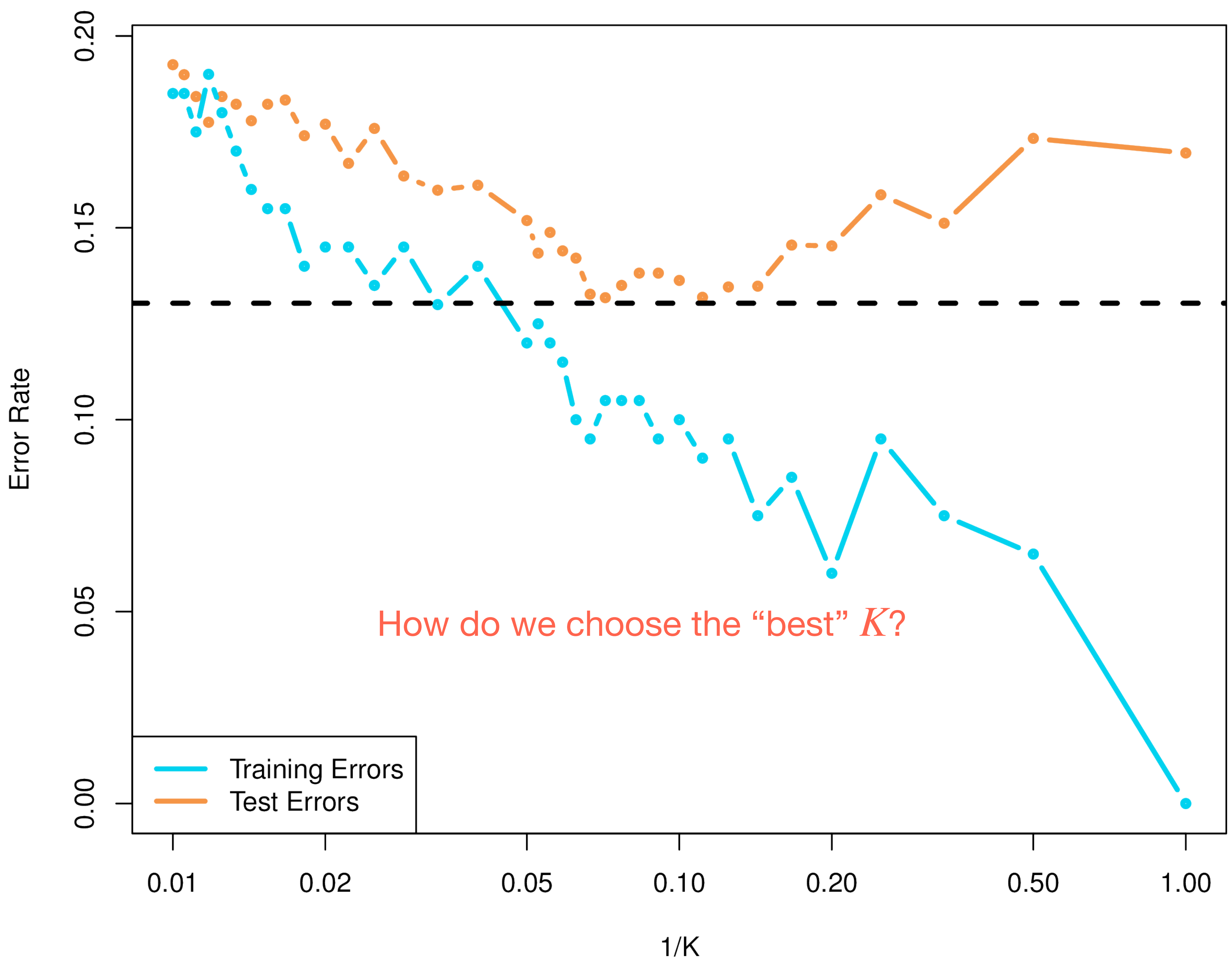




More wiggly decision boundary
High model complexity/flexibility
Low bias + High variance

Less wiggly decision boundary
Low model complexity/flexibility
High bias + Low variance

K-Nearest Classification: bias-variance tradeoff



K-Nearest Methods: Limitation

K-nearest neighbors methods can be pretty good for small p (i.e., $p \leq 4$) and large n

Many more sophisticated versions of non-parametric methods, e.g., kernel estimators…

Fail when p is very large: there are very few (almost no) data points in the neighbors

Curse of Dimensionality!

In summary

Classification is really not too different from Regression

Bias-variance tradeoff

K-Nearest Neighbor Methods

Next…

End of the general discussion

Linear methods: for regression and for classification