STATS 415: K Nearest Neighbors

Prof. Liza Levina

Department of Statistics, University of Michigan

Regression

- Regression: predicting a continuous outcome y from predictors $x = (x_1, x_2, \ldots, x_p)$
- In all cases, we model the relationship as

$$y_i = f(x_i) + \varepsilon_i$$

- Goal: "learn" f from training data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ in order to do prediction and inference
- Better prediction does not always imply better inference, and vice versa

Parametric vs Non-parametric methods

- Parametric methods: assume specific functional form of $f(\cdot)$, then only need to estimate parameters of that function
- A linear model is a parametric method:

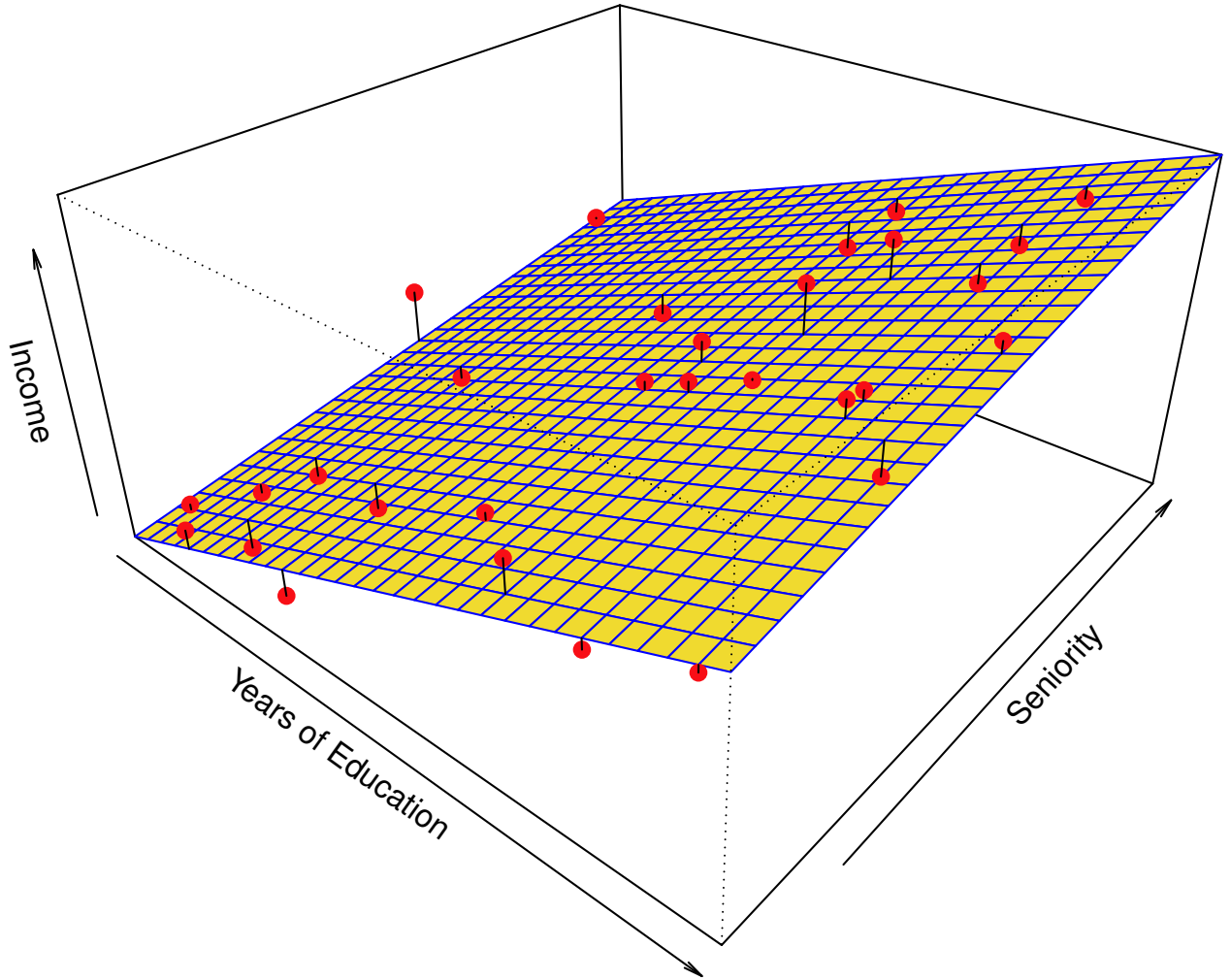
$$f(x_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}$$

- Only need to estimate β's (which give $\hat{f}$ which gives $\hat{y}$)
- Non-parametric methods: do not assume a particular functional form of f

Example: Income vs Education and Seniority

A parametric linear model:

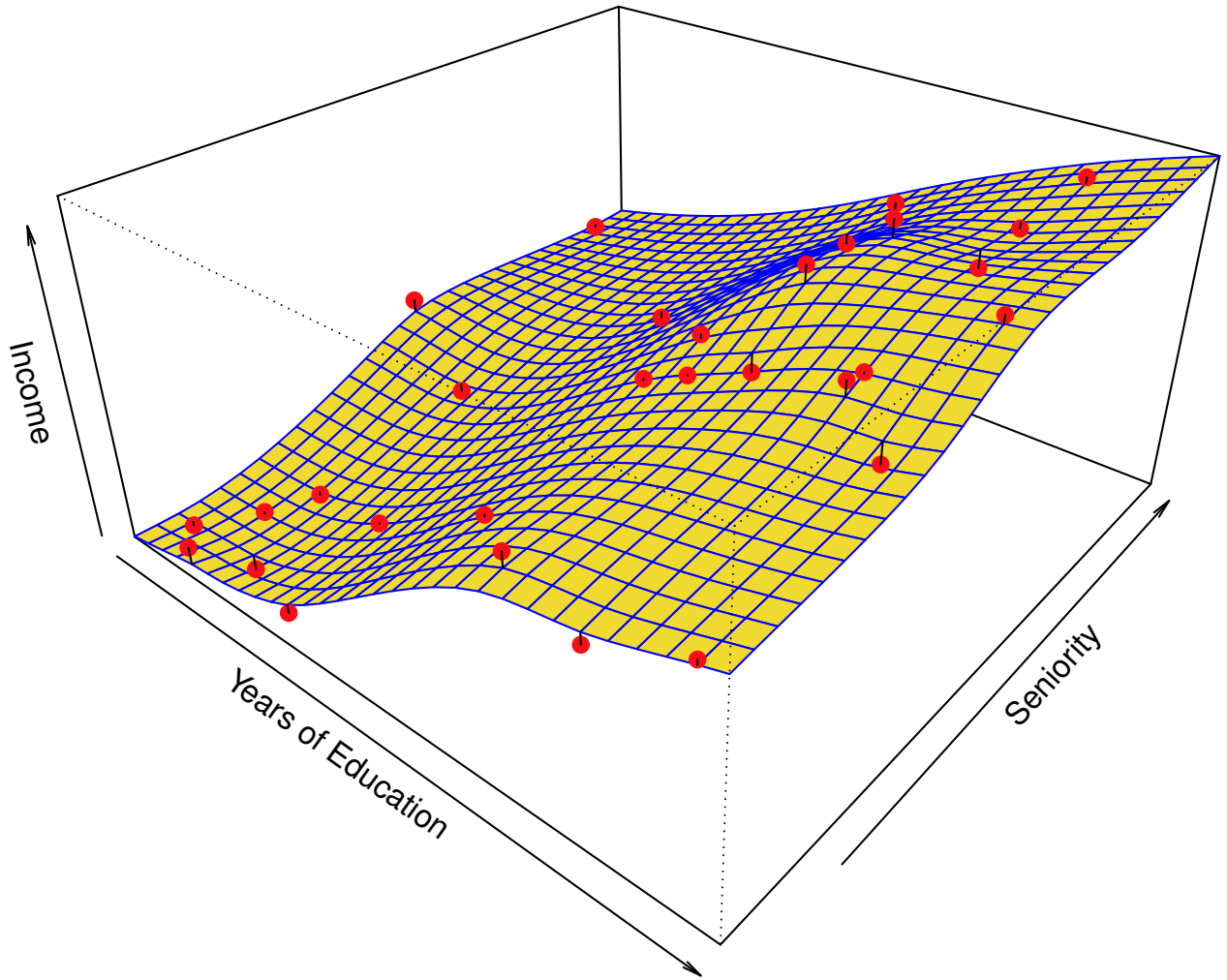
$$\text{income} = \beta_0 + \beta_1 \times \text{education} + \beta_2 \times \text{seniority} + \varepsilon$$



Example: Income vs Education and Seniority

A non-parametric model (no explicit equation)

$$y = f(x) + \varepsilon$$



Example of non-parametric regression: KNN

- KNN = K Nearest Neighbors
- Requires computing distances between all data points $d(x_i, x_{i'})$
- For each point x_i, find K closest neighbors
- To predict y for a given value of x, consider K closest training points to x (by sorting all the $d(x, x_i)$ and finding K smallest ones) and take the average of their responses, i.e.,

$$\hat{f}(x) = \frac{1}{K} \sum_{i: x_i \in N_K(x)} y_i$$

- $N_K(x)$: the set of K nearest neighbors of x

Choice of distance function

- "Neighbors" can mean different things for different datasets
- Euclidean distances are a popular choice (default in R) if all predictors are quantitative:

$$d(x_i, x_{i'}) = \sqrt{(x_{i1} - x_{i'1})^2 + (x_{i2} - x_{i'2})^2 + \cdots + (x_{ip} - x_{i'p})^2}$$

- Consider standardizing if units are very different
- Other similarity measures can be used
- In particular, weighted distances have been used to emphasize some variables over others

Is KNN better than linear regression?

Depends...

- Choice of distance
- Choice of K
- True relationship between x and y
- Number of predictors p and sample size n

Measuring Quality of Fit

- Suppose we have a regression problem.
- One common measure of accuracy is the mean squared error (MSE) i.e.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where $\hat{y}_i$ is the prediction our method gives for the ith observation in our training data.

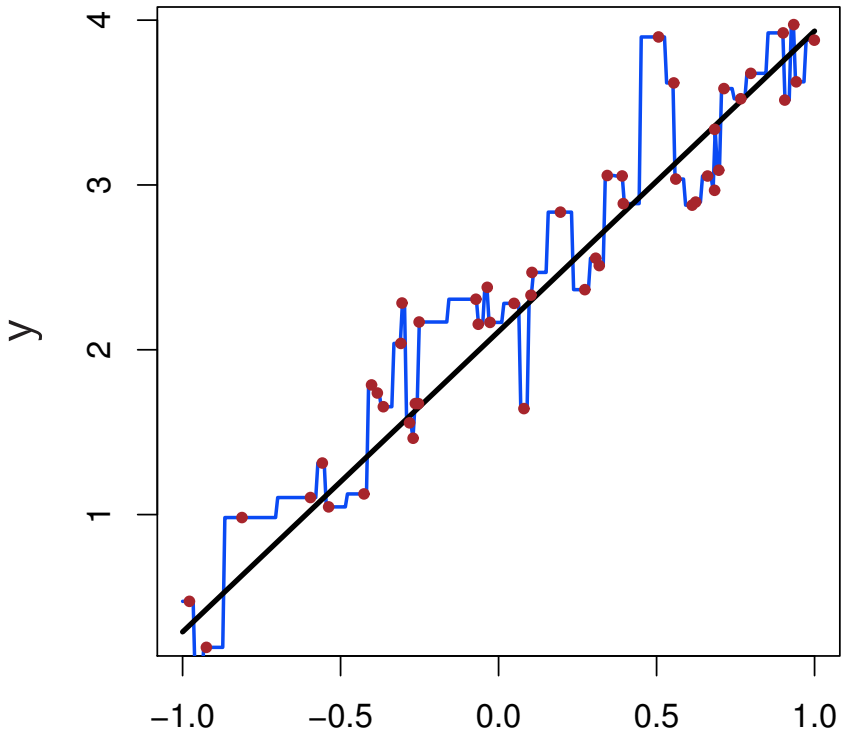
A Problem

- In general our methods have been designed to make MSE small on the training data we are looking at, e.g. with linear regression we choose the line such that MSE is minimized.
- What we really care about is how well the method works on new data. We call this new data "Test Data".
- There is no guarantee that the method with the smallest training MSE will have the smallest test (i.e. new data) MSE.

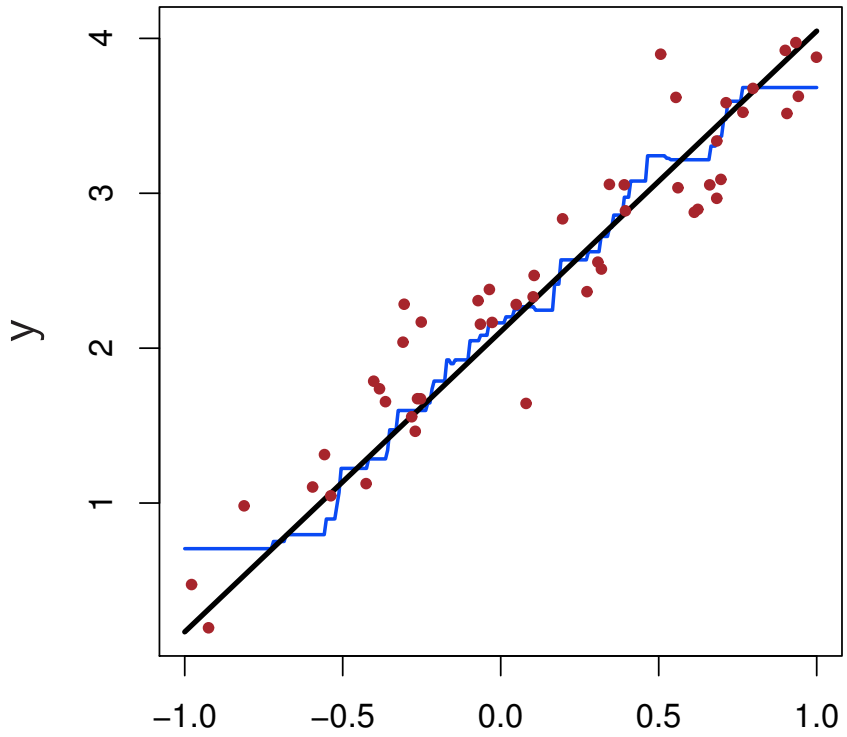
Training vs Test MSEs

- In general the more flexible a method is the lower its training MSE will be, i.e. it will "fit" or explain the training data very well.
- However, the test MSE may be higher or lower for a more flexible method compared to a simple approach
  - Side note: less flexible methods are usually easier to interpret

KNN with one predictor (K=1 and K=9)

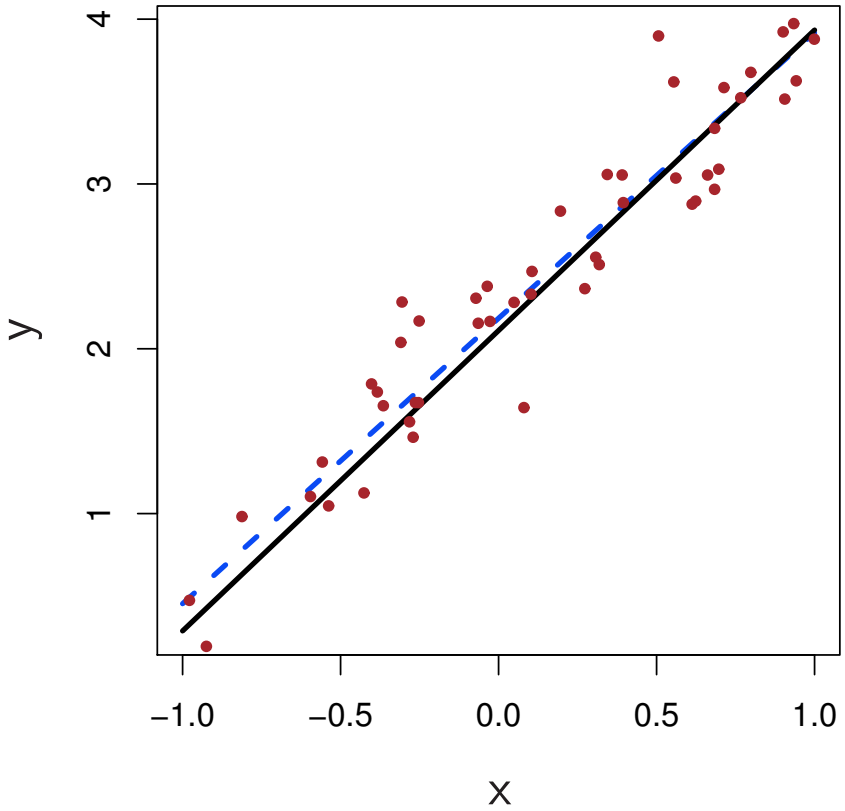


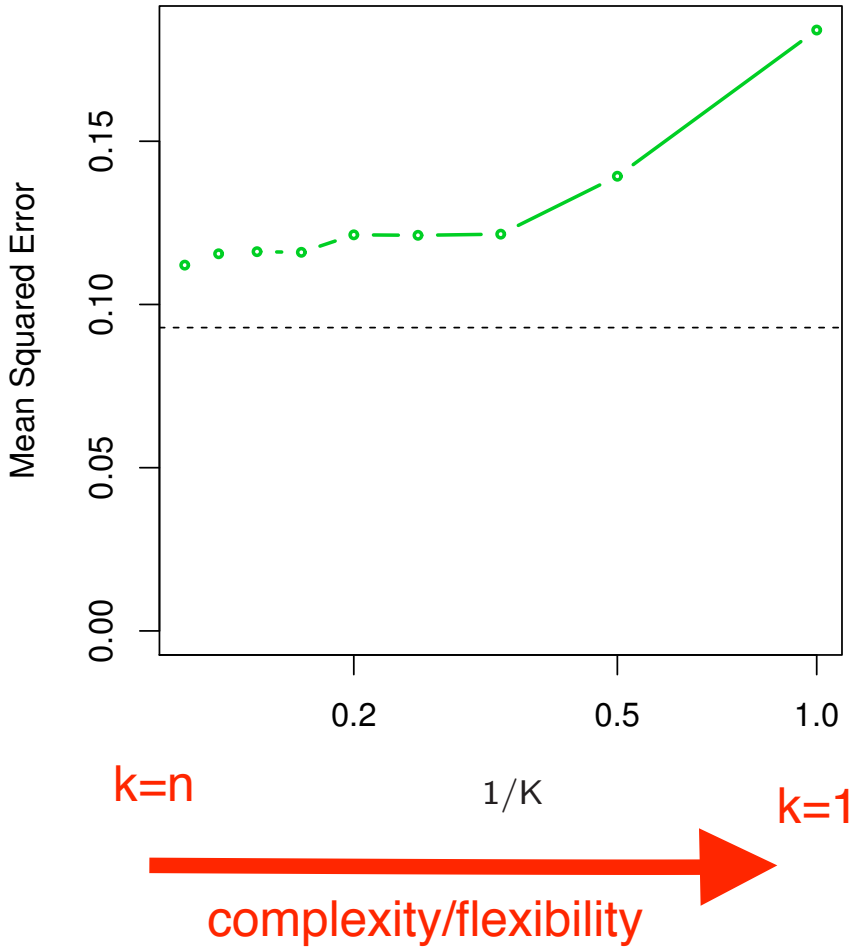
Training:blue line & red points



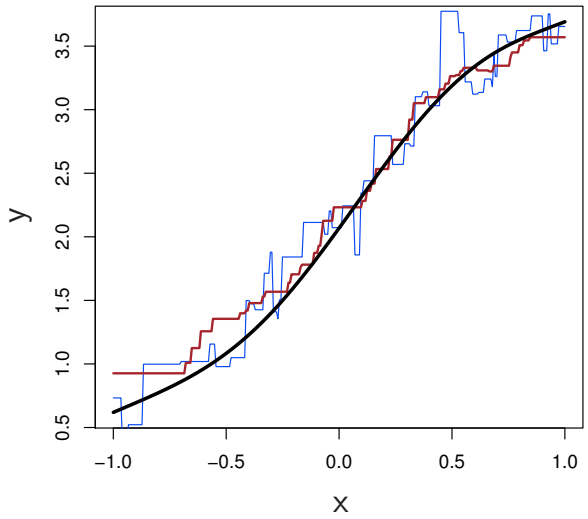
blue & black

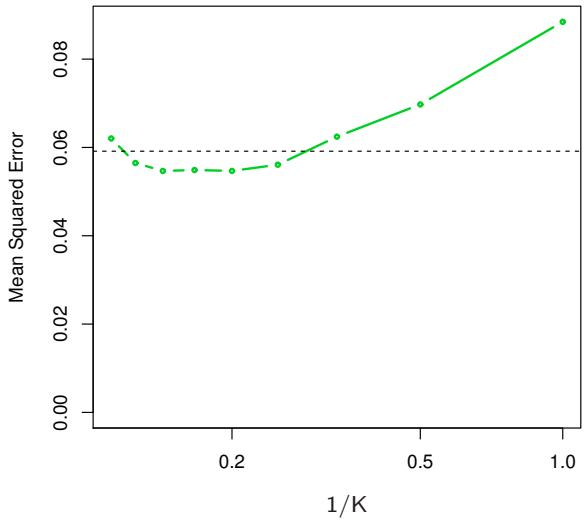
Linear Regression Fit



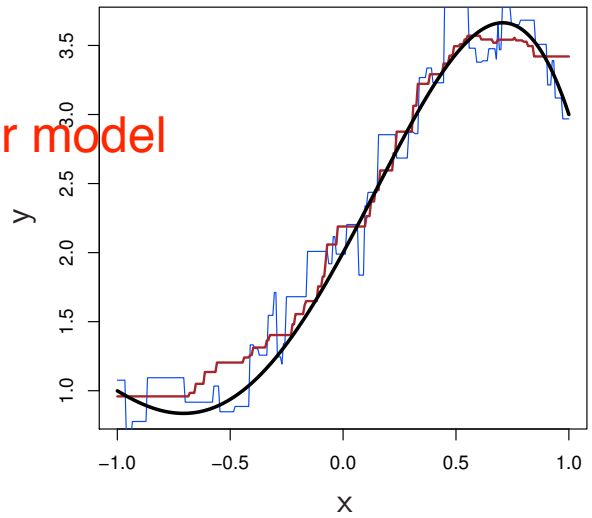


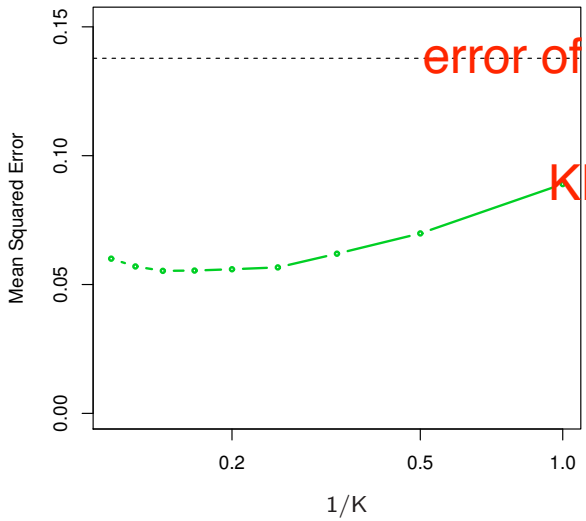
KNN vs. Linear Regression





bad linear model



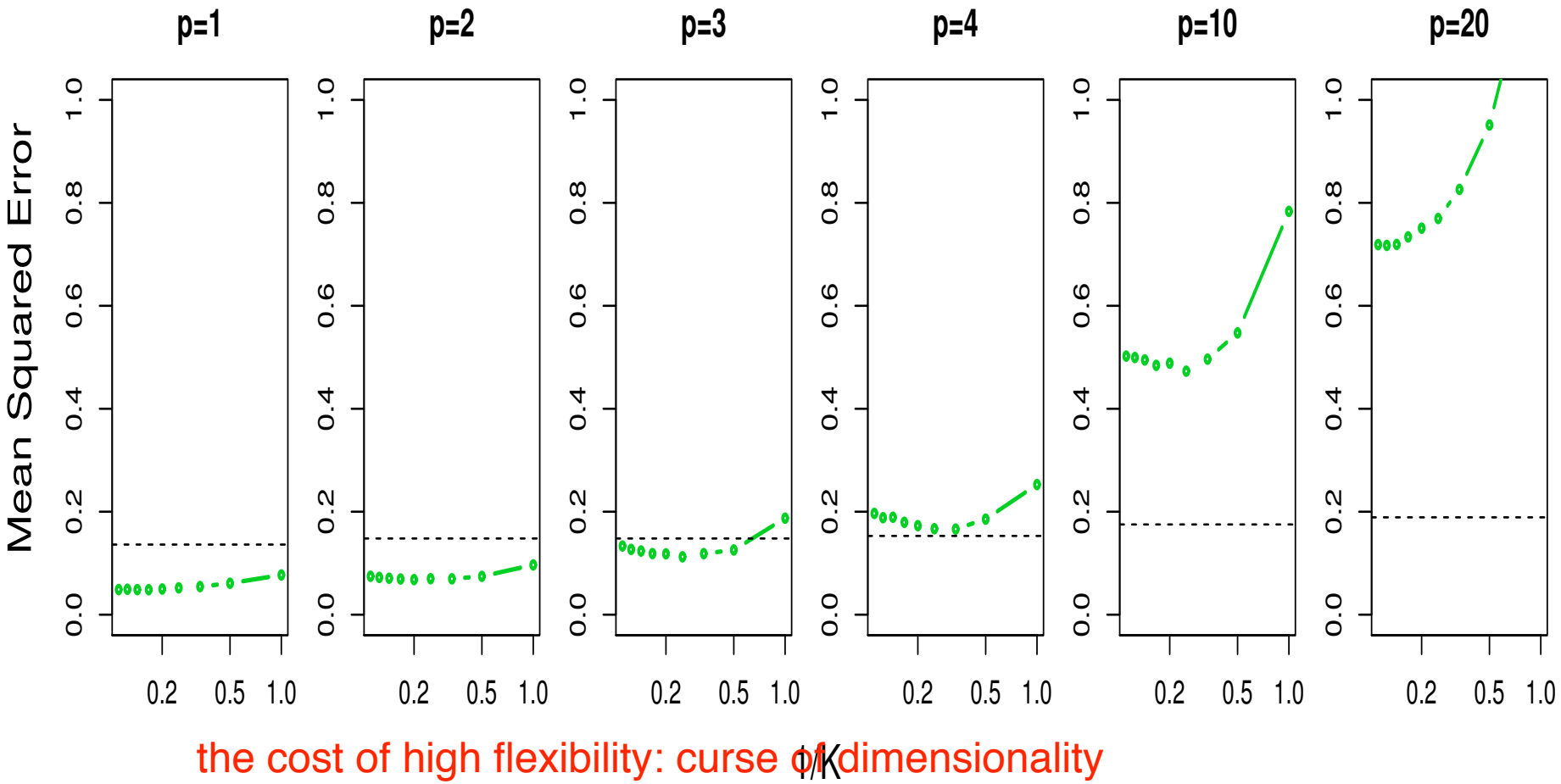


error of linear regression

KNN

Number of predictors matters!

The error of linear regression is also growing, but very slowly. better than KNN



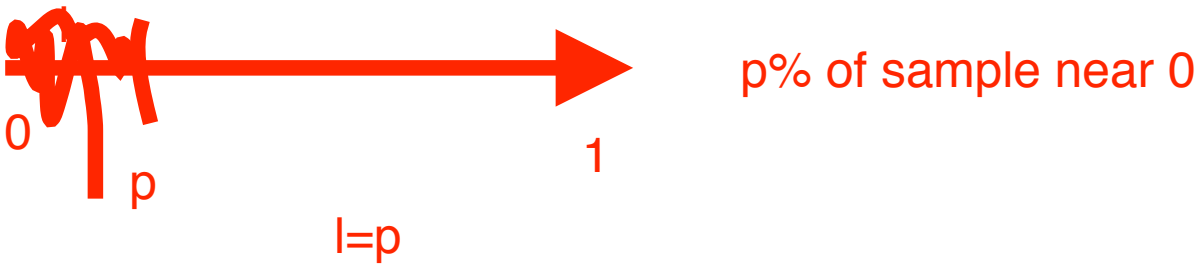
the cost of high flexibility: curse of dimensionality

Curse of Dimensionality

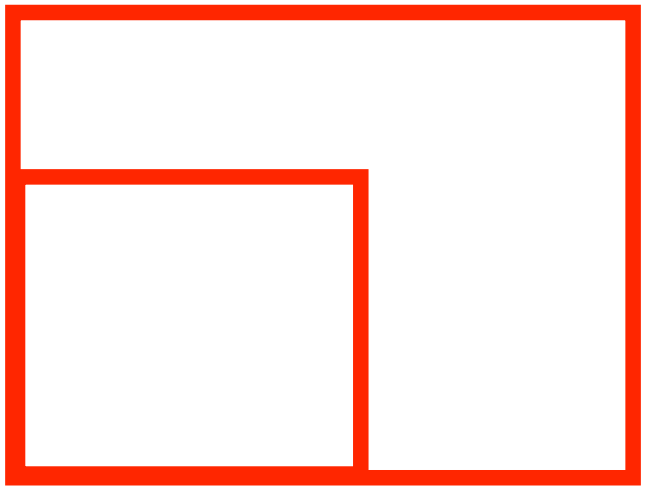
K-nearest neighbors can fail in high dimensions, because it becomes difficult to find K observations close to a target point x_0:

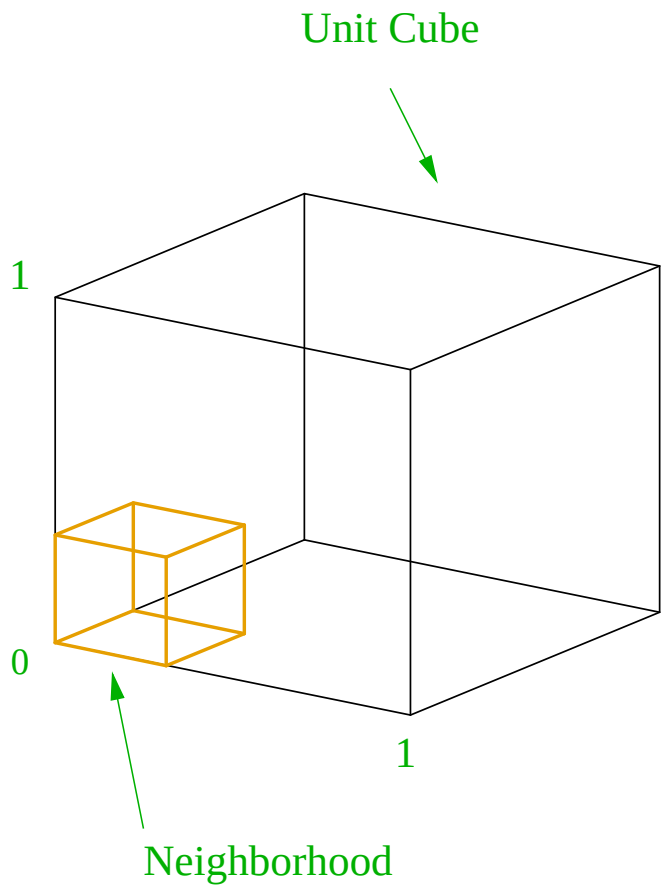
- Points in high dimensions are far apart; finding K neighbors means taking a large spatial neighborhod. (Increased bias).
- Reducing the spatial size of the neighborhood means reducing K; the predictions become noisier (Increased variance).

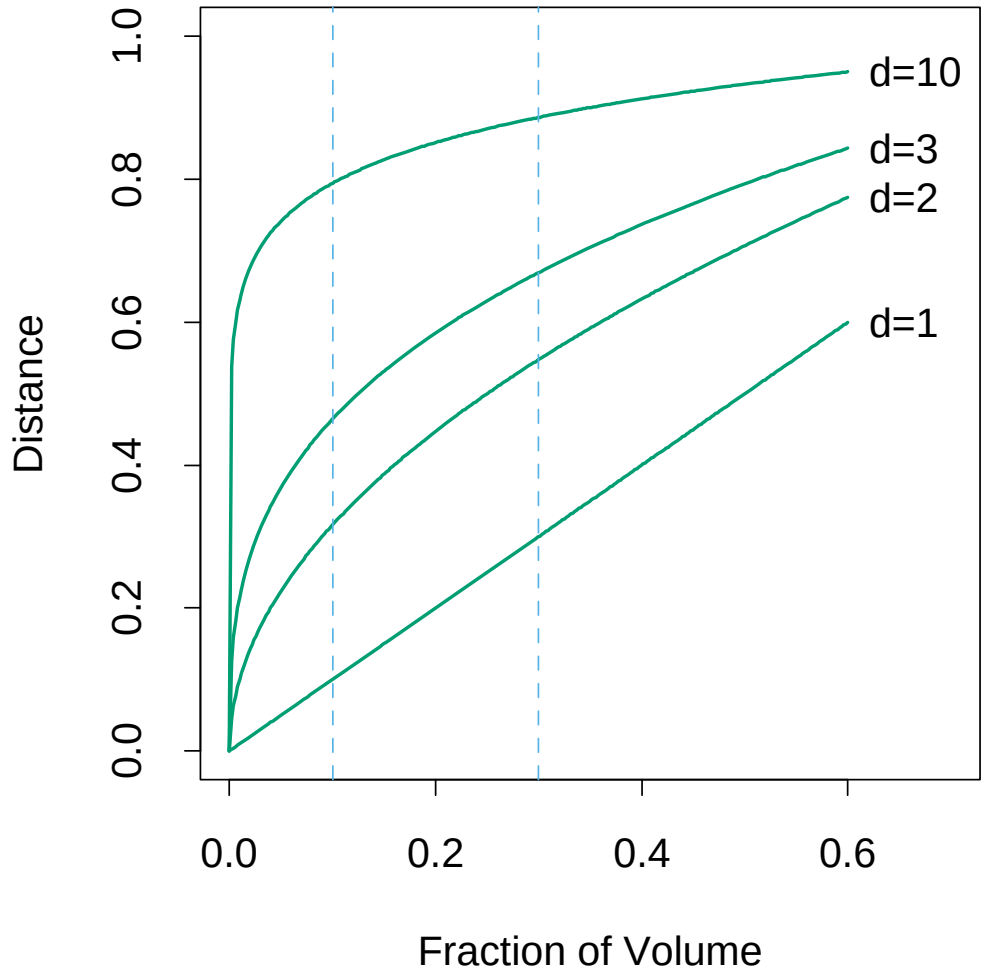
Geometry of the curse of dimensionality



- Suppose the points are uniformly distributed in a p-dimensional unit (hyper)cube, with side 1.
- Consider a smaller (hyper)cube with edge $\ell < 1$ inside the unit cube (think of it as neighborhood). What fraction ρ of the observations will it capture?




Unit Cube
1
0
1
Neighborhood


Distance
0.0
0.2
0.4
0.6
0.8
1.0
d=10
d=3
d=2
d=1
0.0
0.2
0.4
0.6
Fraction of Volume

Some calculations

- Volume of the large unit cube: 1^d=1
- Volume of the small cube with edge ℓ: l^d
- Fraction of observations that fall in the cube: l^d/1
- For a given fraction ρ, need a cube with edge length

  l=ρ^(1/d)

  dimension p, fraction ρ
- When $p = 1$: If $\rho = 0.01$, $\ell = 0.01$ and if $\rho = 0.1$, $\ell = 0.1$.
- When $p = 10$: If $\rho = 0.01$, $\ell = 0.63$ and if $\rho = 0.1$, $\ell = 0.80$.

When $p = 10$, in order to capture 10% of the data, we must cover 80% of the range of each input.

Implications of the curse

- "Local" methods are no longer local when the dimension p increases.
- Exponential growth in n: if 100 points are sufficient to estimate a function in $\mathbb{R}^1$, 100^{10} are needed to achieve a similar accuracy in $\mathbb{R}^{10}$.
- So why does anything still work in high dimensions at all? Structure!

K-Nearest Neighbors (KNN) for Classification

- Classification setting: y is a categorical variable (class)
- For any given x we find the K closest neighbors to x in the training data, and examine their classes.
- Assign x to the class corresponding to the majority votes of the K nearest neighbors
- If a tie occurs, choose at random (K is usually taken to be odd to avoid ties for two classes)

The binary case: 2-class problem

- Code $Y = 1$ for class 1, and $Y = -1$ for class 2.
- Take a vote on a new point x:

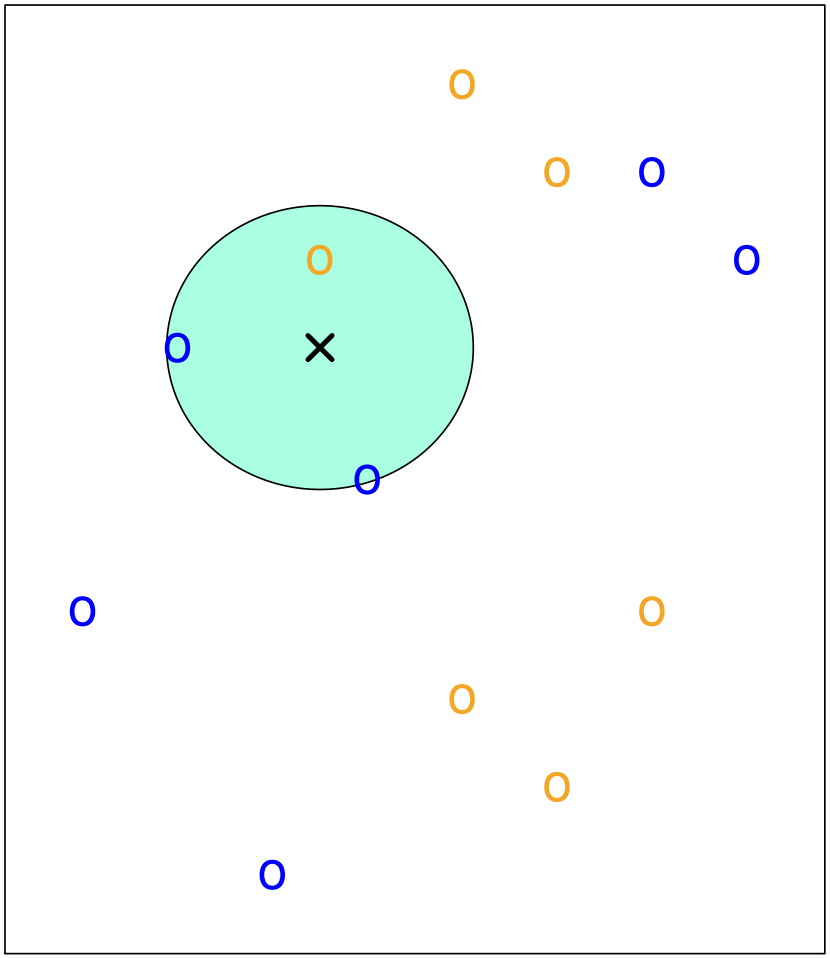
$$\hat{f}(x) = \frac{1}{K} \sum_{x_i \in N_K(x)} y_i$$

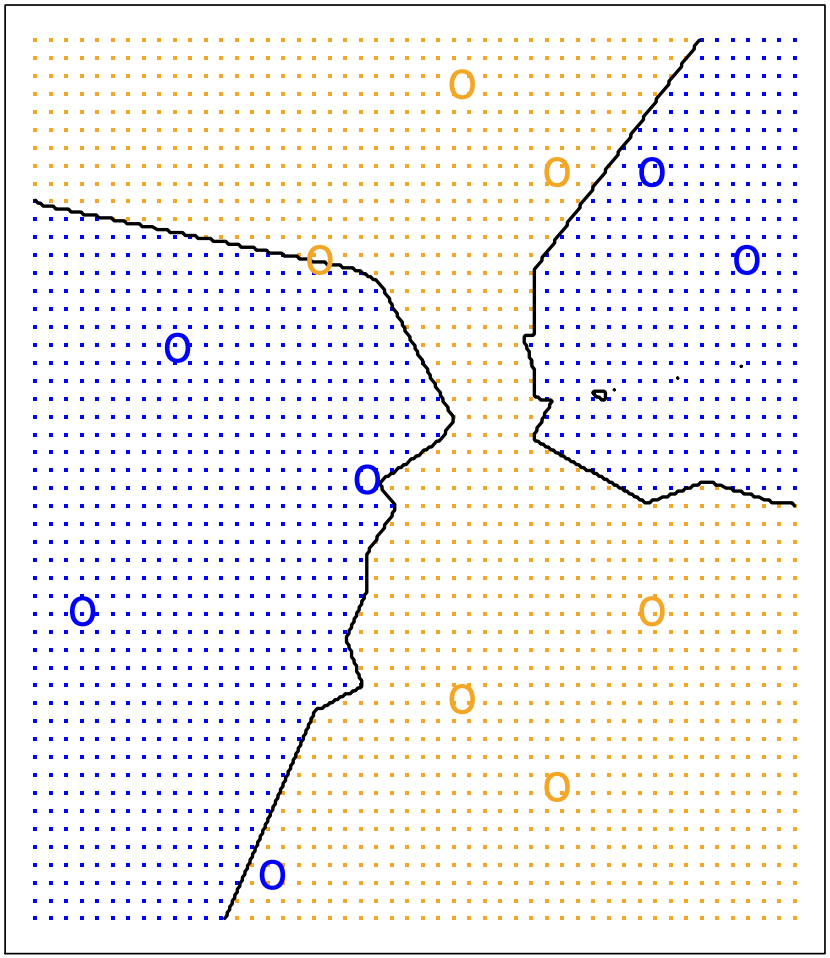
where $N_K(x)$ consists of the k closest points to x in the training data (k-nearest neighborhood).

- The classification rule is

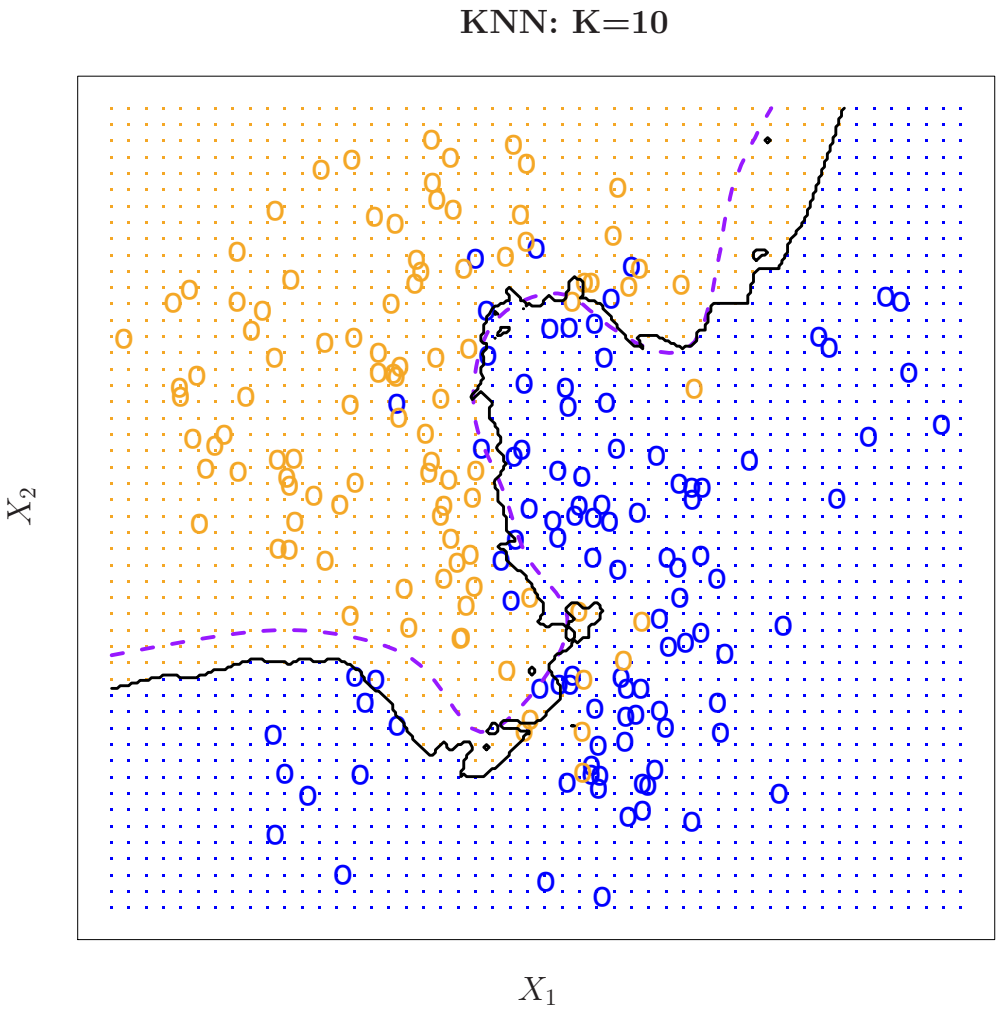
$$\hat{c}(x) = \begin{cases} 1 & \text{if } \hat{f}(x) > 0 \\ -1 & \text{if } \hat{f}(x) < 0 \end{cases}$$

KNN Example with $K = 3$

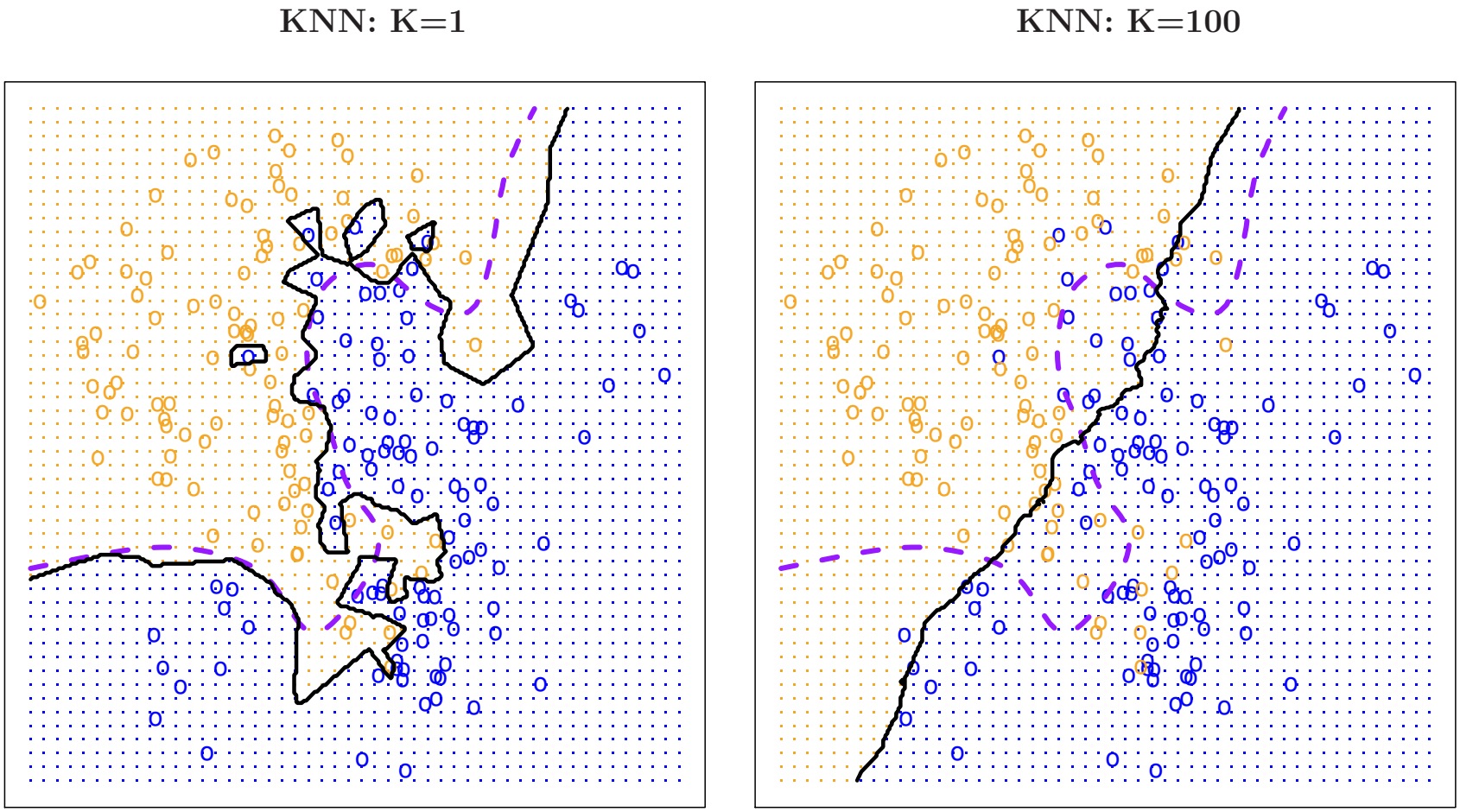




Simulated Data with $K = 10$



$K = 1$ and $K = 100$

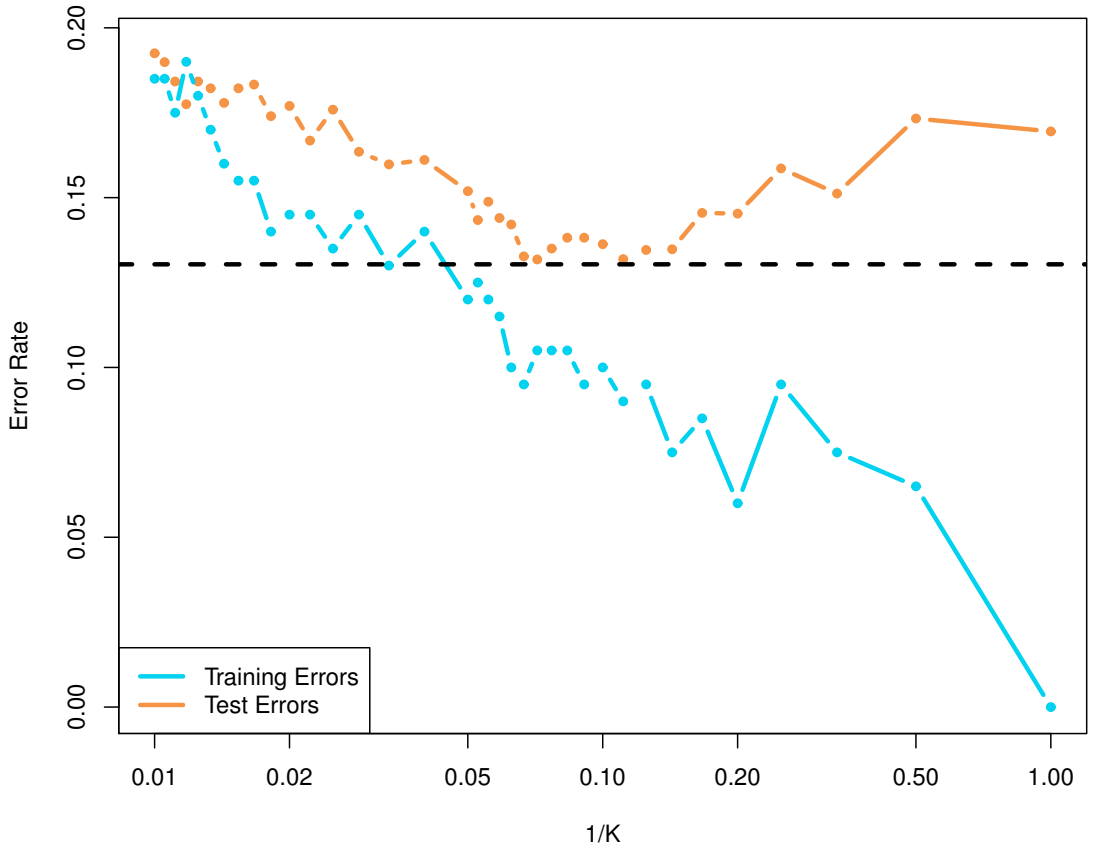


Model complexity for K-NN

- How many parameters does K-nearest neighbors have?
- It's a non-parametric method! K is not a parameter in the sense of β in linear regression.
- But each neighborhood makes its own local estimate; there are n training points, K neighbors for each, and thus roughly n/K different local estimates
- K controls the model complexity: the smaller K, the more different local estimates, the more complex the model.

Training vs Test Error Rates on the Simulated Data

- Notice that training error rates keep going down as K decreases or equivalently as the flexibility increases.
- However, the test error rate at first decreases but then starts to increase again.

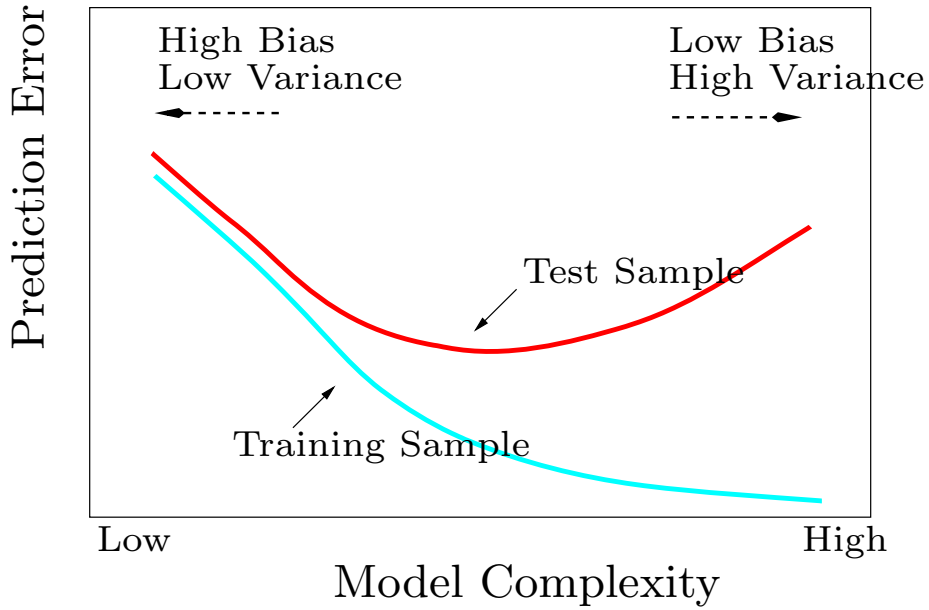


Choice of K

- Can we pick K minimize the training error?
  - No. When $K = 1$, the training error is zero. (overfitting)
- Choose K to minimize the misclassification error on a separate data set.
- Ideally, should use validation data to choose k and an independent test set to estimate test error.
- Will later learn cross-validation - validating without losing too much data

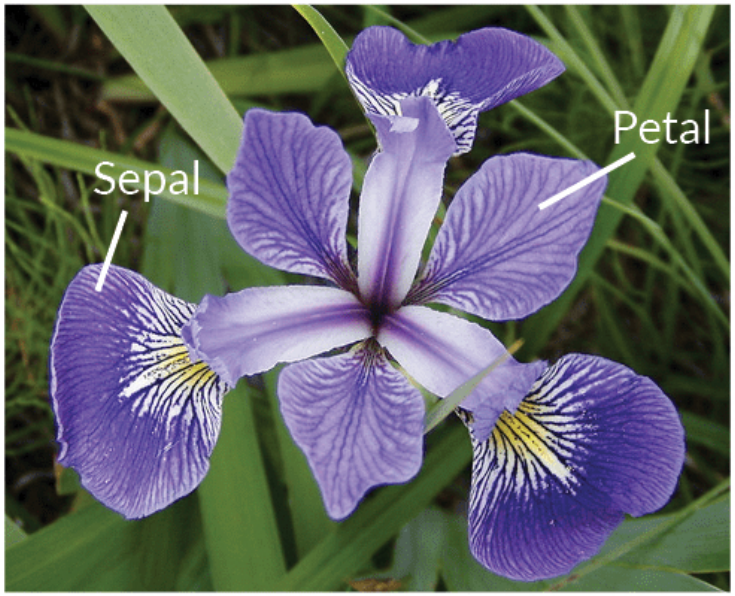
A Fundamental Picture

- In general, as the model complexity increases, training errors will always decline.
- However, test errors will decline at first (as reductions in bias dominate) but will then start to increase again (as increases in variance dominate).
- We must always keep this in mind when choosing a learning method. More flexible/complicated is not always better!



Recall Iris data

$n_1 = n_2 = n_3 = 50,\ n = 150,\ p = 4$

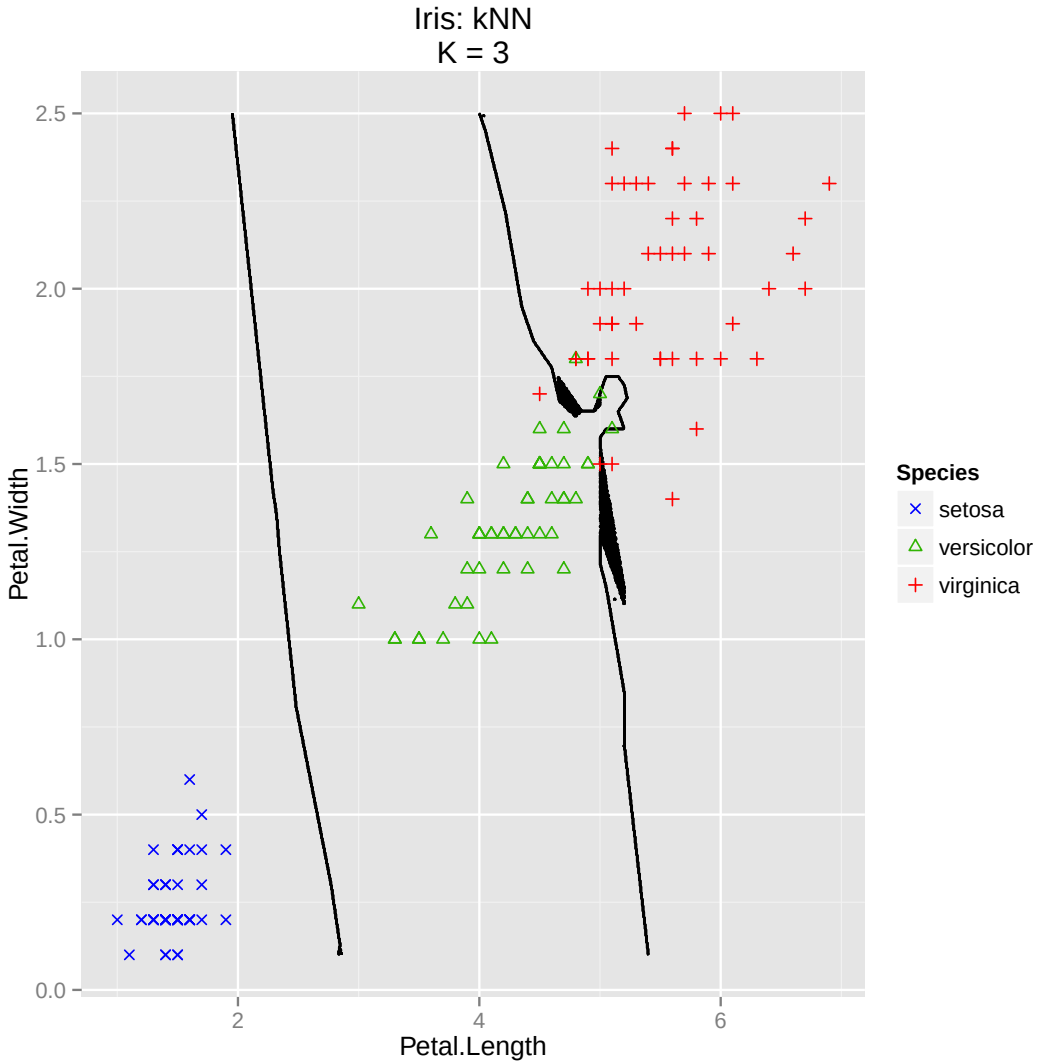


Iris Versicolor

Iris Setosa

Iris Virginica

3-NN results for the iris data



Makes two errors

Training and validation error for the iris data