START: 1/18/2022 Tuesday Week 1 (Lecture 1)

Introduction to Data Mining

Yang Ni
Assistant Professor
Department of Statistics
Texas A&M University

About me

- Born and raised in Shanghai, China
- BS in Mathematics from Fudan University
- PhD in Statistics from Rice University
- Postdoc in UT Austin
- Joined Texas A&M Fall, 2018
- I'd like to do various sports & outdoor activities: basketball, weight lifting, swimming, hiking...
- I am a Bayesian statistician. My research interest includes
  - Graphical models: reconstructing gene regulatory networks
  - Clustering: finding disease subgroups
  - Variable selection: discovering disease biomarkers
  - Big data computation: phenotyping large electronic health records data
  - More on `https://www.stat.tamu.edu/~yni/`

About this course

- Cross listed as ECEN 758 and CSCE 676.
- Aim to teach you how to use the method/tool, not how to invent or build one
- To understand the model, intuitions, strengths and weaknesses of various approaches, not the theories.
- Hands-on experience of R for data mining
- You might be disappointed if you have taken ISEN 413 or STAT 636

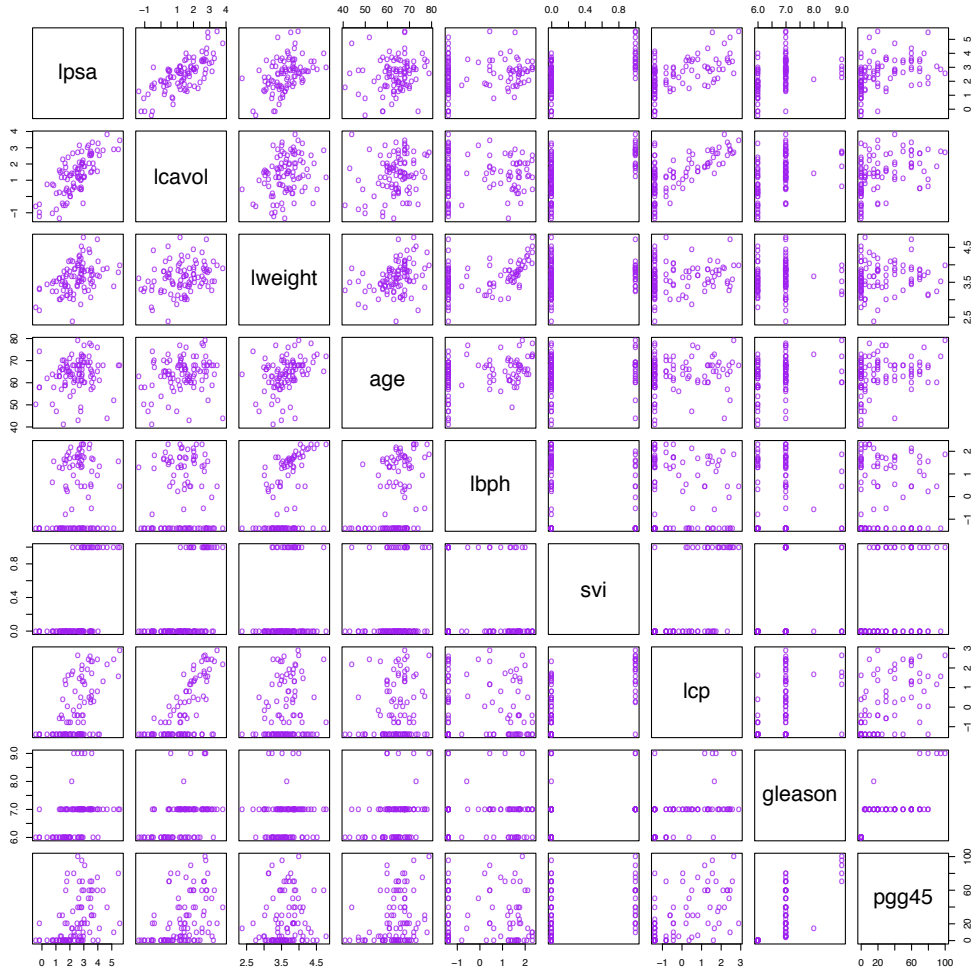
What is Data Mining?

- Data mining is the process designed to explore data in search of consistent patterns and/or systematic relationships between variables, and then to validate the findings by applying the detected patterns to new subsets of data.
- It involves methods at the intersection of Statistics and Artificial Intelligence.
- Close relatives: statistical learning, machine learning, pattern recognition.

Data Mining Problems

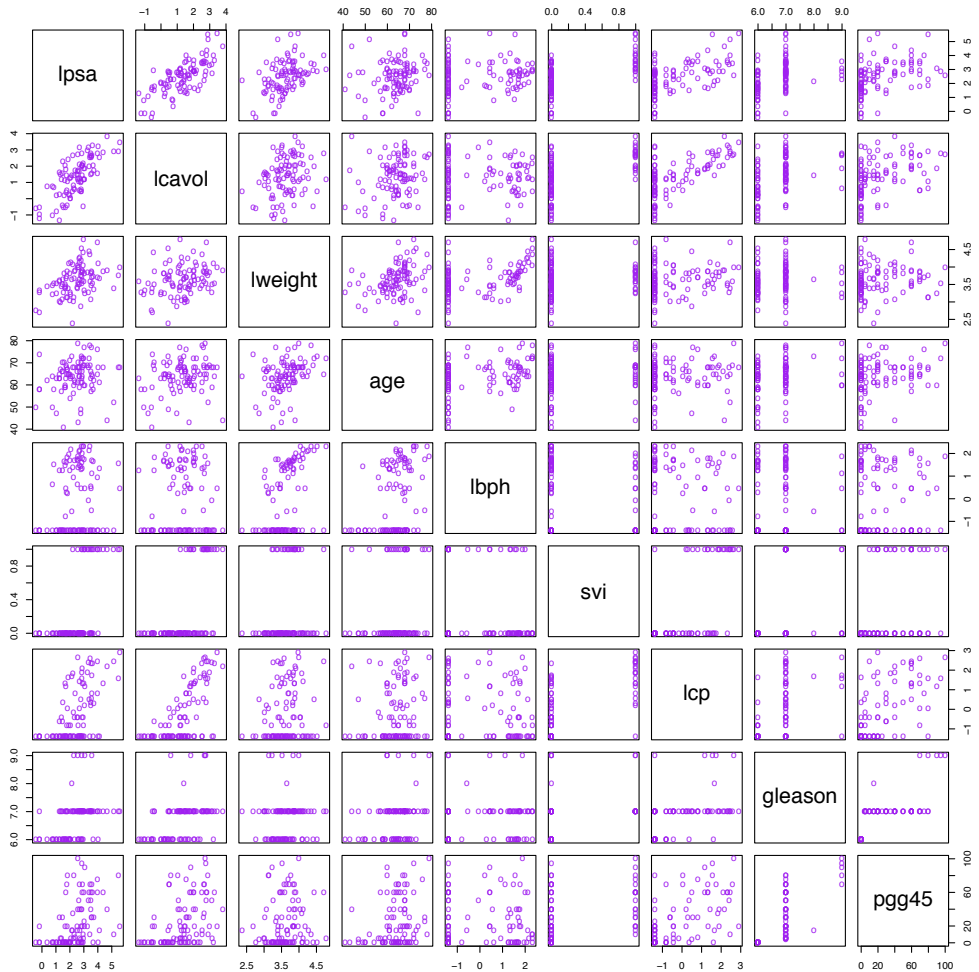
- Identify the risk factors for prostate cancer.
- Predict whether someone will have a heart attack on the basis of demographic, diet and clinical measurements.
- Customize an email spam detection system.
- Identify the numbers in a handwritten zip code.
- Establish the relationship between salary and demographic variables in population survey data.
- Classify the pixels in a LANDSAT image, by usage.
- Discover breast cancer subtypes.

Risk factors for prostate cancer



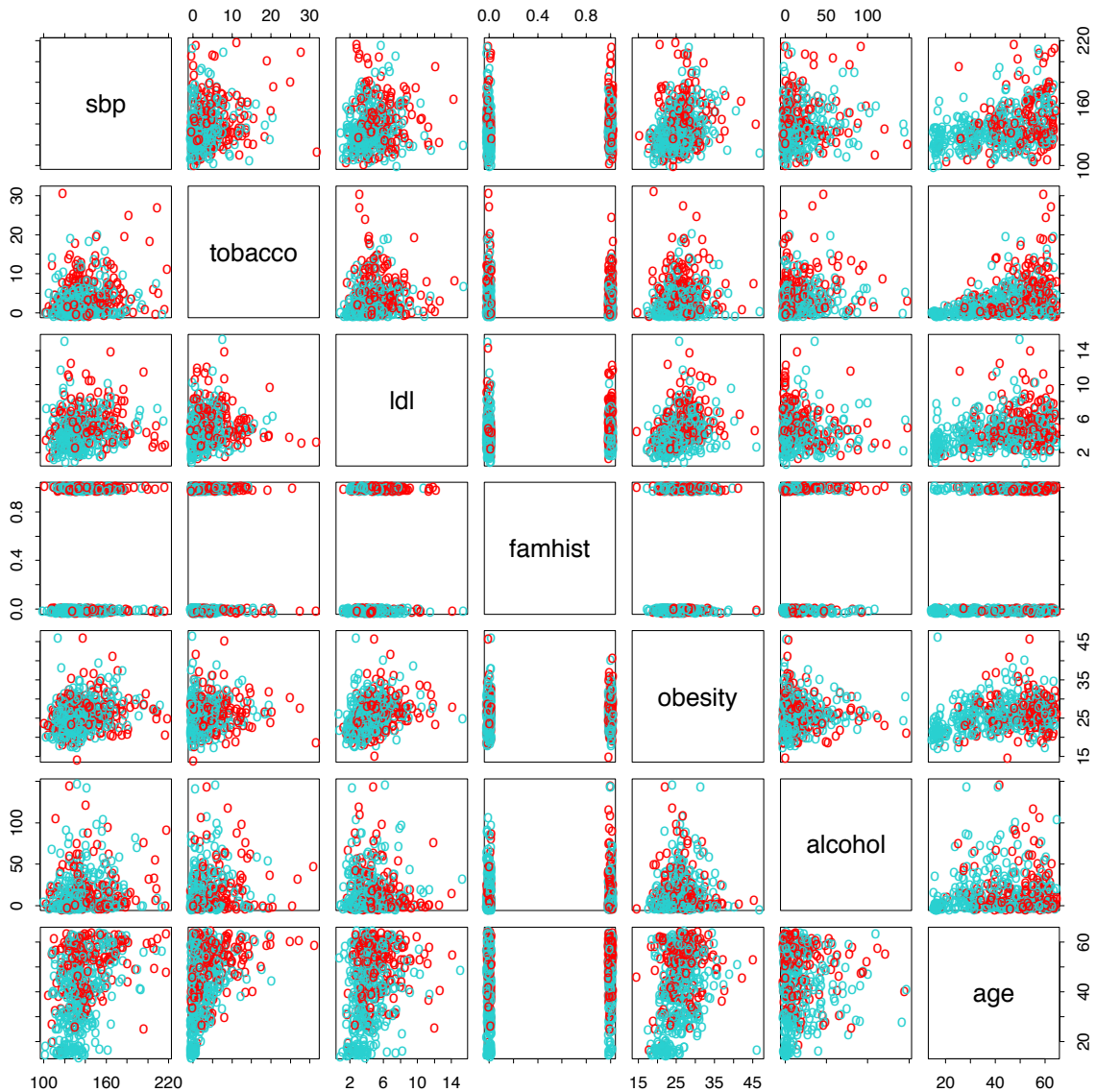
lpsa: (log) prostate specific antigen, lcavol: (log) cancer volume, lweight: (log) prostate weight, lbph: (log) benign prostatic hyperplasia amount, svi (discrete variable): seminal vesicle invasion, lcp: (log) capsular penetration, gleason (discrete variable): Gleason score, pgg45: percent of Gleason scores 4 or 5

Risk factors for prostate cancer



While it is easy to see some correlation/dependence structure between variables, it's hard to, for example, predict lpsa based on other variables by eye.

Predict heart attack



Red: heart attack, Blue: healthy

Spam Detection

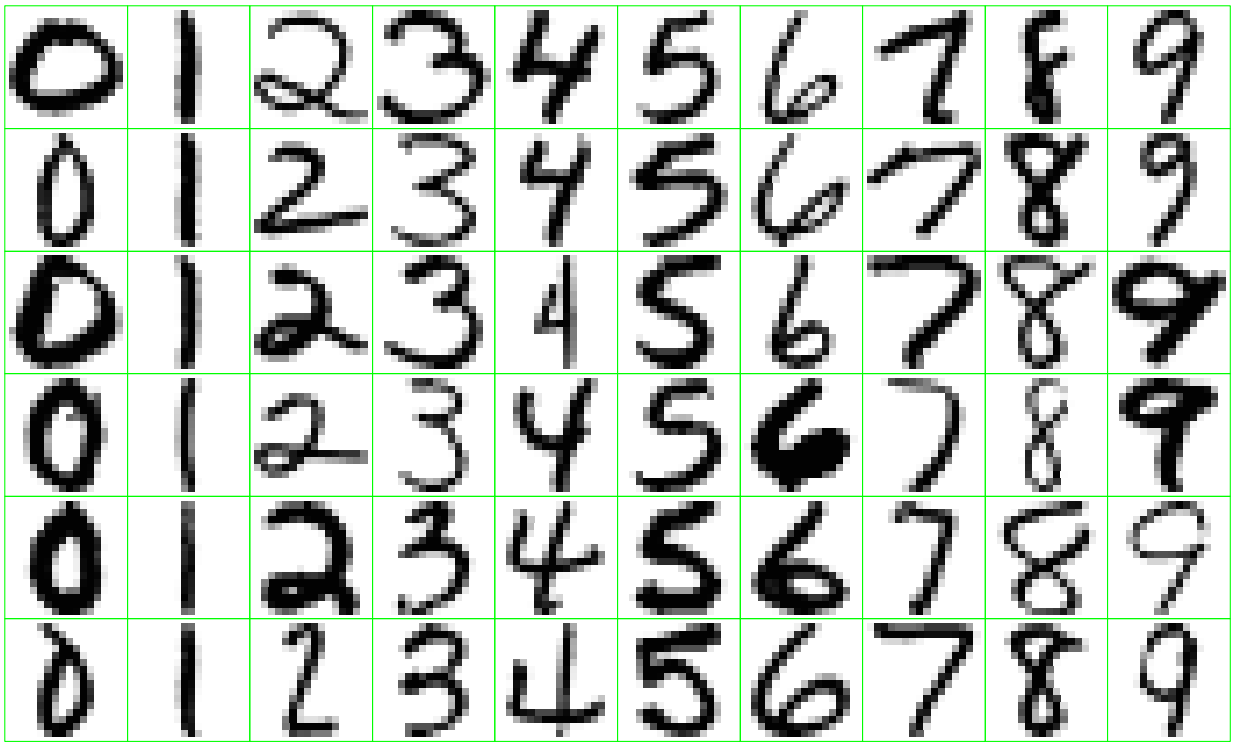
- Data from 4601 emails sent to an individual (named George, at HP labs, before 2000). Each is labeled as spam or email.
- Goal: build a customized spam filter.
- Input features: relative frequencies of 57 of the most commonly occurring words and punctuation marks in these email messages.

	george	you	hp	free	!	edu	remove
spam	0.00	2.26	0.02	0.52	0.51	0.01	0.28
email	1.27	1.27	0.90	0.07	0.11	0.29	0.01

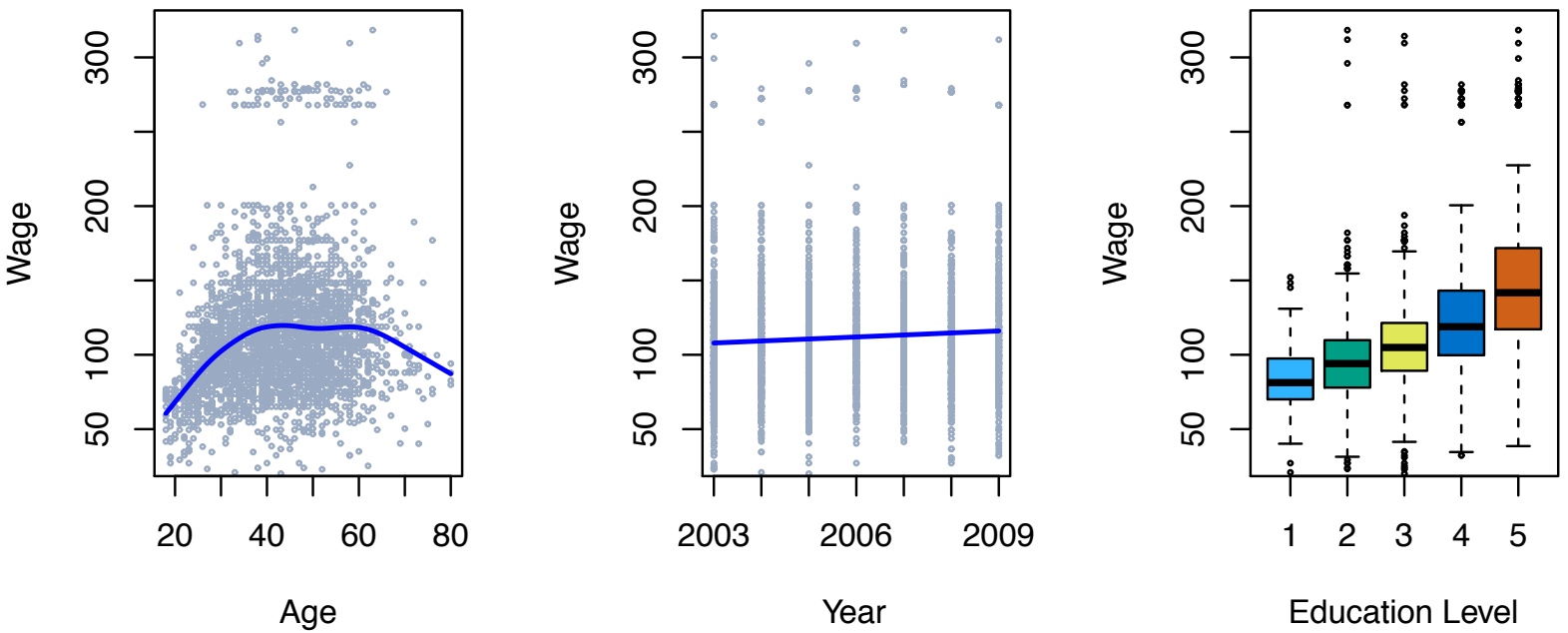
Average percentage of words or characters in an email message equal to the indicated word or character. We have chosen the words and characters showing the largest difference between spam and email.

Handwritten digits

10 classification levels.



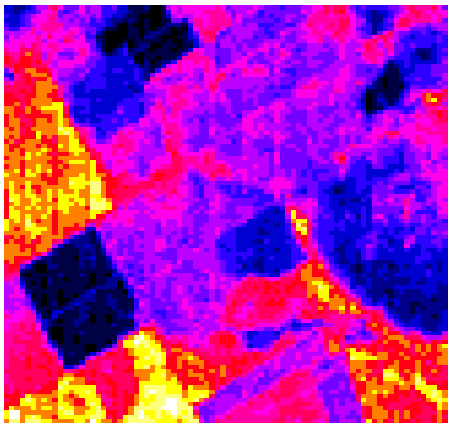
Salary and demographics



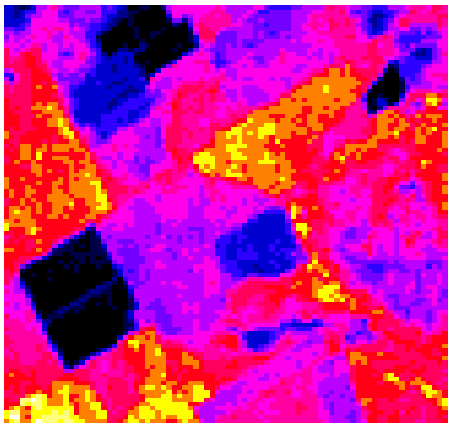
Income survey data for males from the central Atlantic region of the USA in 2009.

Pixel classification

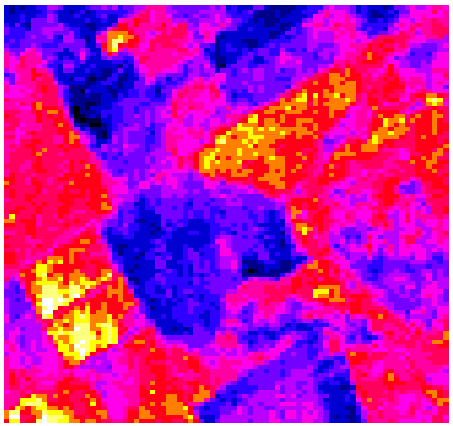
Spectral Band 1



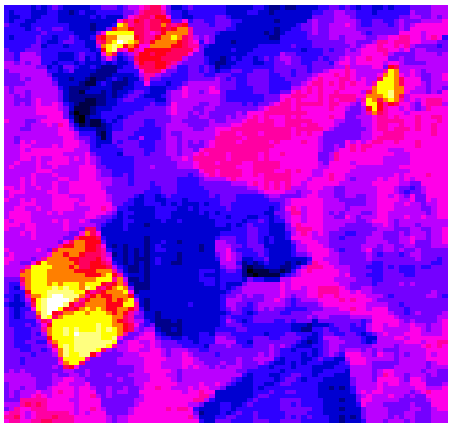
Spectral Band 2



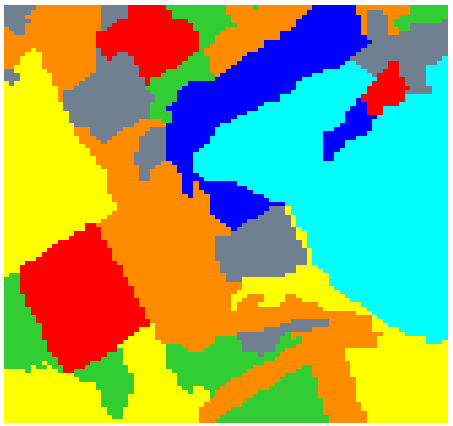
Spectral Band 3



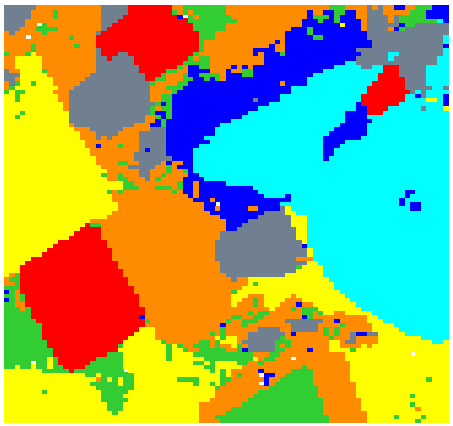
Spectral Band 4



Land Usage

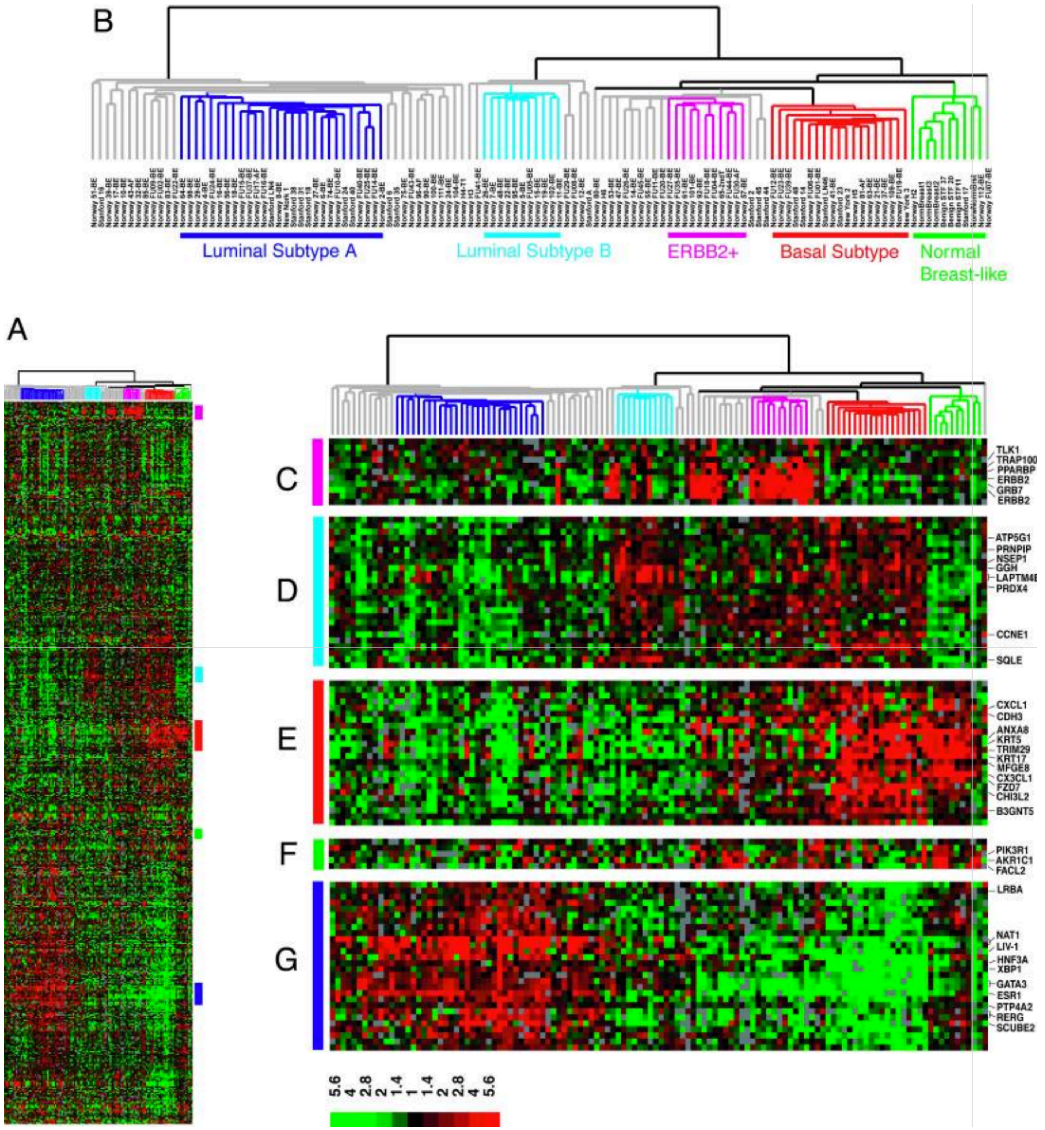


Predicted Land Usage



Usage $\in$ {red soil, cotton, vegetation stubble, mixture, gray soil, damp gray soil}

Subtype breast cancer



Br cluster Analysis

No well-defined outcome.

Supervised Learning

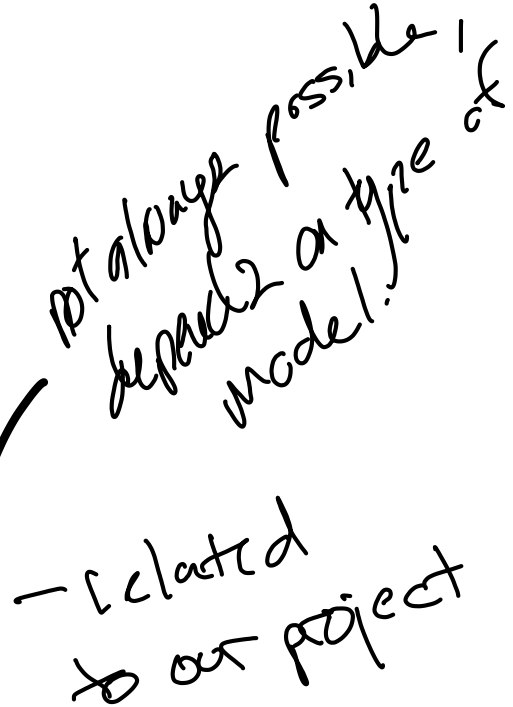
Starting point:

- Outcome measurement Y (also called dependent variable, response, target).
- Vector of p predictor measurements X (also called inputs, regressors, covariates, features, independent variables).
- In the **regression** problem, Y is quantitative (e.g price, blood pressure).
- In the **classification** problem, Y takes values in a finite, unordered set (survived/died, digit 0-9, cancer class of tissue sample).
- We have training data $(x_1, y_1), \ldots, (x_n, y_n)$. These are observations (examples, instances, realizations) of these measurements.

Objectives

On the basis of the training data we would like to:

- Accurately predict unseen test cases.
- Understand which inputs affect the outcome, and how.
- Assess the quality of our predictions and inferences.



STOP Tuesday 1/18/22 (Week 1, Lecture 1)

Philosophy

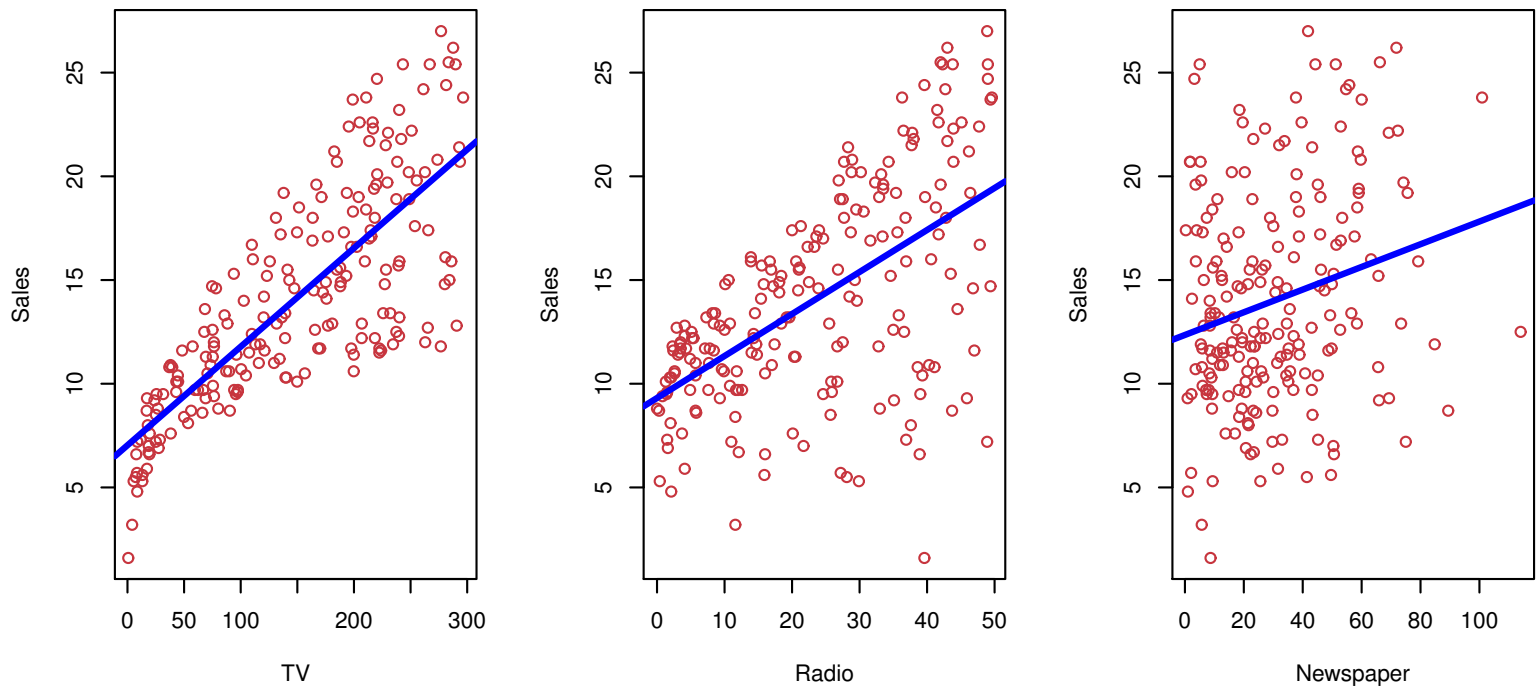
START: Thursday 1/20/22 (week 1, lecture 2)

- It is important to understand the ideas behind the various techniques, in order to know how and when to use them.
- One has to understand the simpler methods first, in order to grasp the more sophisticated ones.
- It is important to accurately assess the performance of a method, to know how well or how badly it is working [simpler methods often perform as well as fancier ones!]
- This is an exciting research area, having important applications in science, industry and finance.
- Data mining is a fundamental ingredient in the training of a modern **data scientist**.

Unsupervised learning

- No outcome variable, just a set of predictors (features) measured on a set of samples.
- Objective is more fuzzy — find groups of samples that behave similarly, find features that behave similarly, find linear combinations of features with the most variation.
- Difficult to know how well your are doing.
- Different from supervised learning, but can be useful as a pre-processing step for supervised learning or as an exploratory analysis tool.

Regression Problems



- Shown are Sales vs TV, Radio and Newspaper, with a blue linear-regression line fit separately to each.
- Can we predict Sales using these three?
- Perhaps we can do better using a model

$$\text{Sales} \approx f(\text{TV}, \text{Radio}, \text{Newspaper})$$

Notation

- Here Sales is a response that we wish to predict. We generically refer to the response as Y.
- TV is a feature, or input, or predictor; we name it X_1.
- Likewise name Radio as X_2, and so on.
- We can refer to the input vector collectively as

$$X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix}$$

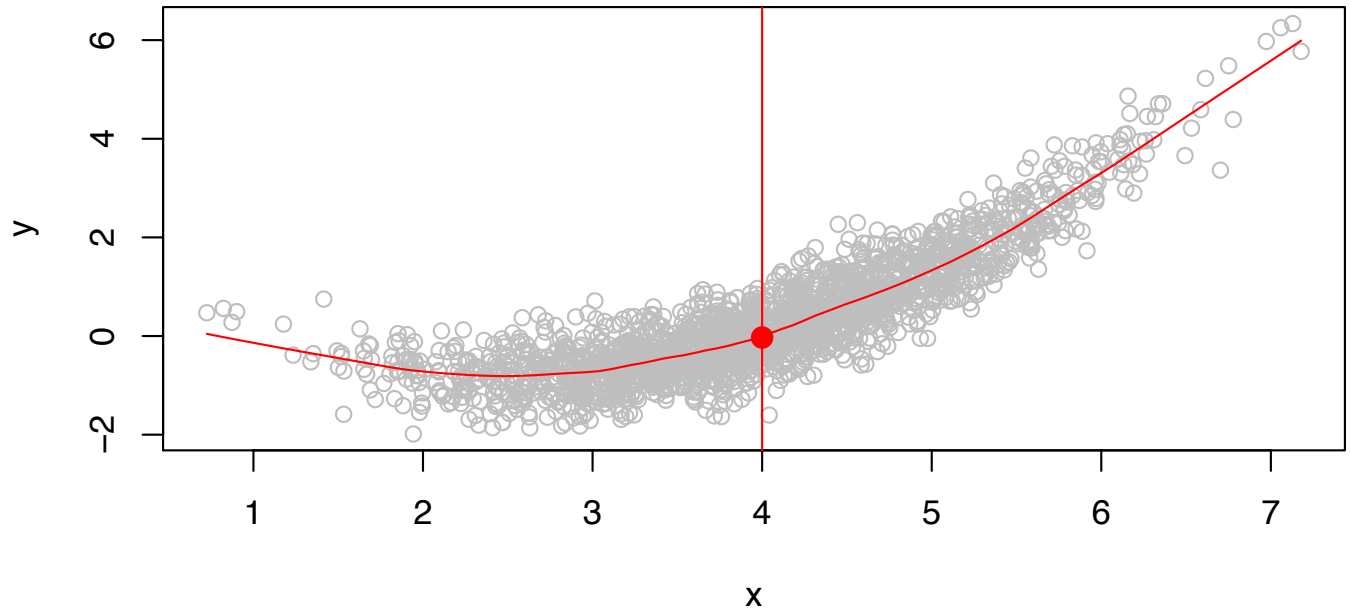
- Now we write our model as

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

where ϵ captures measurement errors and other discrepancies.

What is $f(X)$ good for?

- With a good f we can make predictions of Y at new points $X = x$.
- We can understand which components of $X = (X_1, X_2, ..., X_p)$ are important in explaining Y, and which are irrelevant. e.g. Seniority and Years of Education have a big impact on Income, but Marital Status typically does not.
- Depending on the complexity of f, we may be able to understand how each component X_j of X affects Y.



- Is there an ideal $f(X)$? In particular, what is a good value for $f(X)$ at any selected value of X, say $X = 4$? There can be many Y values at $X = 4$. A good value is

$$f(4) = E(Y|X = 4)$$

- $E(Y|X = 4)$ means **expected value** (average) of Y given $X = 4$.
- This ideal $f(x) = E(Y|X = x)$ is called the **regression function**.

The regression function $f(x)$

- $f(x)$ is also defined for vector X; e.g. $f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$.
- $f(x)$ is the **optimal** predictor of Y with regard to mean-squared prediction error: $f(x) = E(Y|X = x)$ is the function that minimizes $E[(Y - g(X))^2|X = x]$ over all functions g at all points $X = x$.
- $\epsilon = Y - f(x)$ is the **irreducible error** — i.e. even if we knew $f(x)$, we would still make errors in prediction, since at each $X = x$ there is typically a distribution of possible Y values.
- For any estimate $\hat{f}(x)$ of $f(x)$, we have

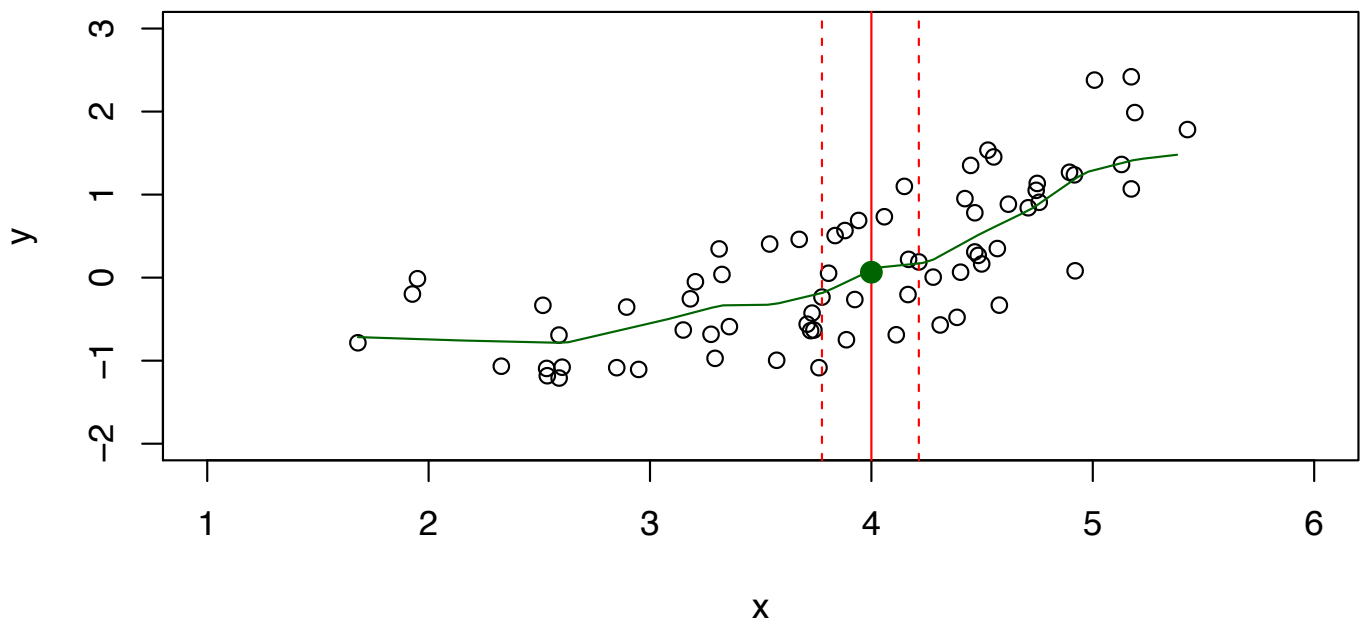
$$E[(Y - \hat{f}(X))^2|X = x] = \underbrace{[f(x) - \hat{f}(x)]^2}_{\textit{Reducible}} + \underbrace{Var(\epsilon)}_{\textit{Irreducible}}$$

How to estimate f

- Typically we have few if any data points with $X = 4$ exactly.
- So we cannot compute $E(Y|X = x)$!
- Relax the definition and let

$$\hat{f}(x) = Ave(Y|X \in \mathcal{N}(x))$$

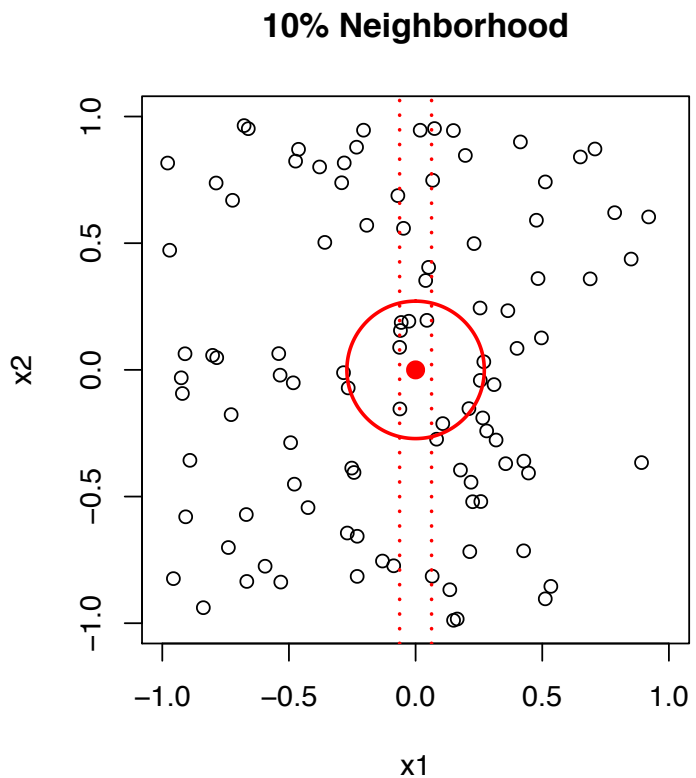
where $\mathcal{N}(x)$ is some neighborhood of x.

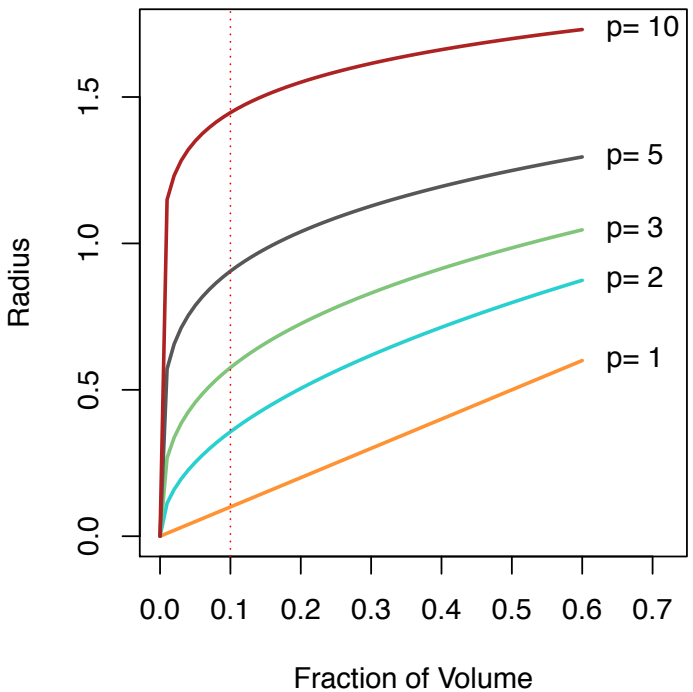


need d small dimensional.

- Nearest neighbor averaging can be pretty good for small p — i.e. $p \leqslant 4$ and large-ish n.
- Nearest neighbor methods can be lousy when p is large.
- Reason: the **curse of dimensionality**. Nearest neighbors tend to be far away in high dimensions.
  - We need to get a reasonable fraction of the n values of y_i to average to bring the variance down—e.g. 10%.
  - A 10% neighborhood in high dimensions need no longer be local, so we lose the spirit of estimating $E(Y|X = x)$ by local averaging.

The curse of dimensionality



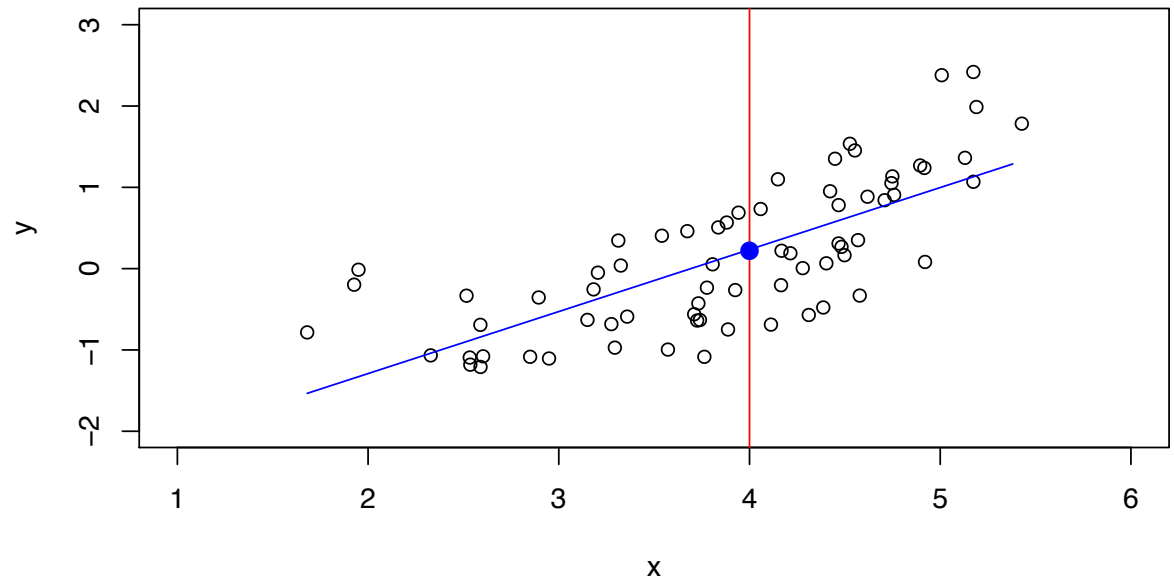


The **linear** model is an important example of a parametric model:

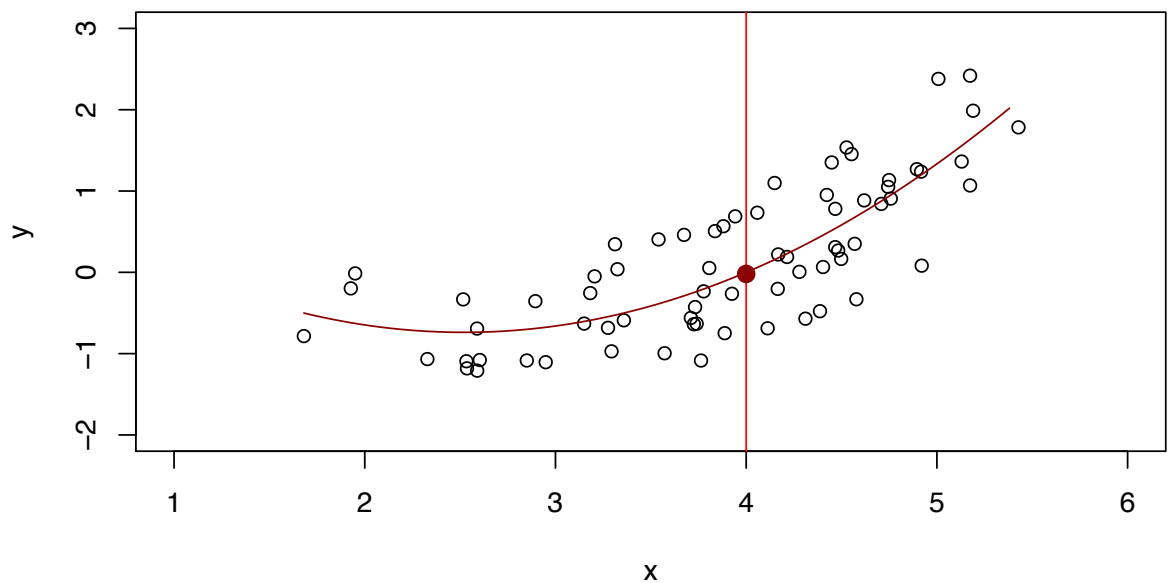
$$f_L(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

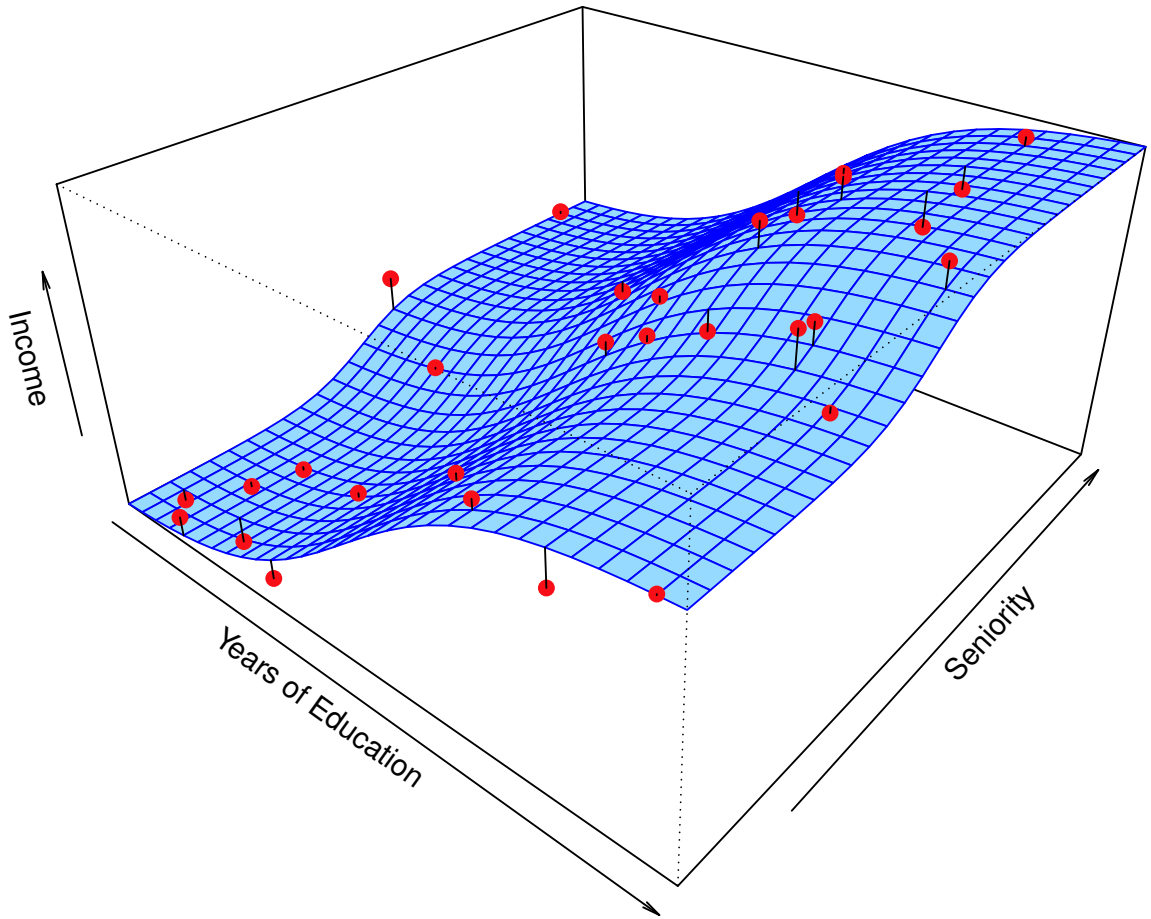
- A linear model is specified in terms of $p + 1$ parameters $\beta_0, \beta_1, \ldots, \beta_p$
- We estimate the parameters by fitting the model to training data.
- Although it is **almost never correct**, a linear model often serves as a good and interpretable approximation to the unknown true function $f(X)$.

A linear model $\hat{f}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1$ gives a reasonable fit here



A quadratic model $\hat{f}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_1^2$ fits slightly better.

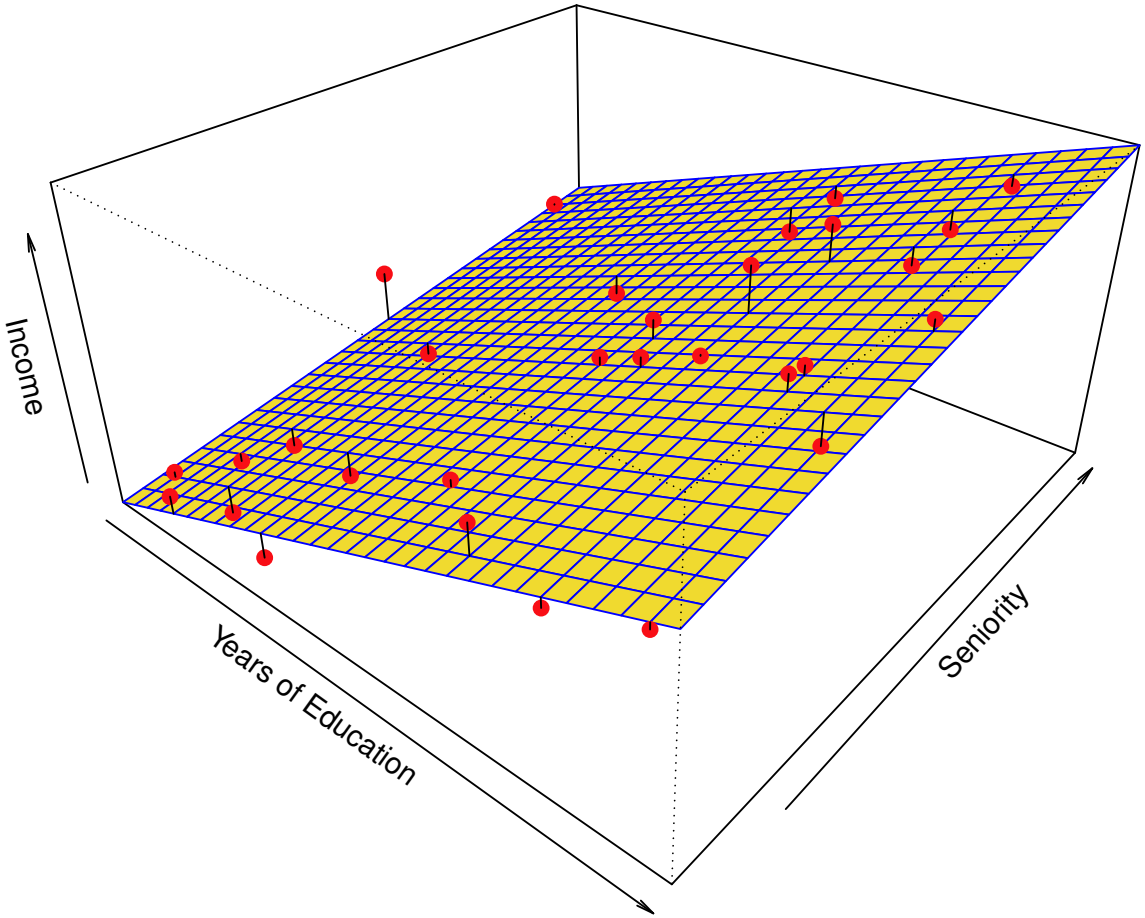




Simulated example. Red points are simulated values for income from the model

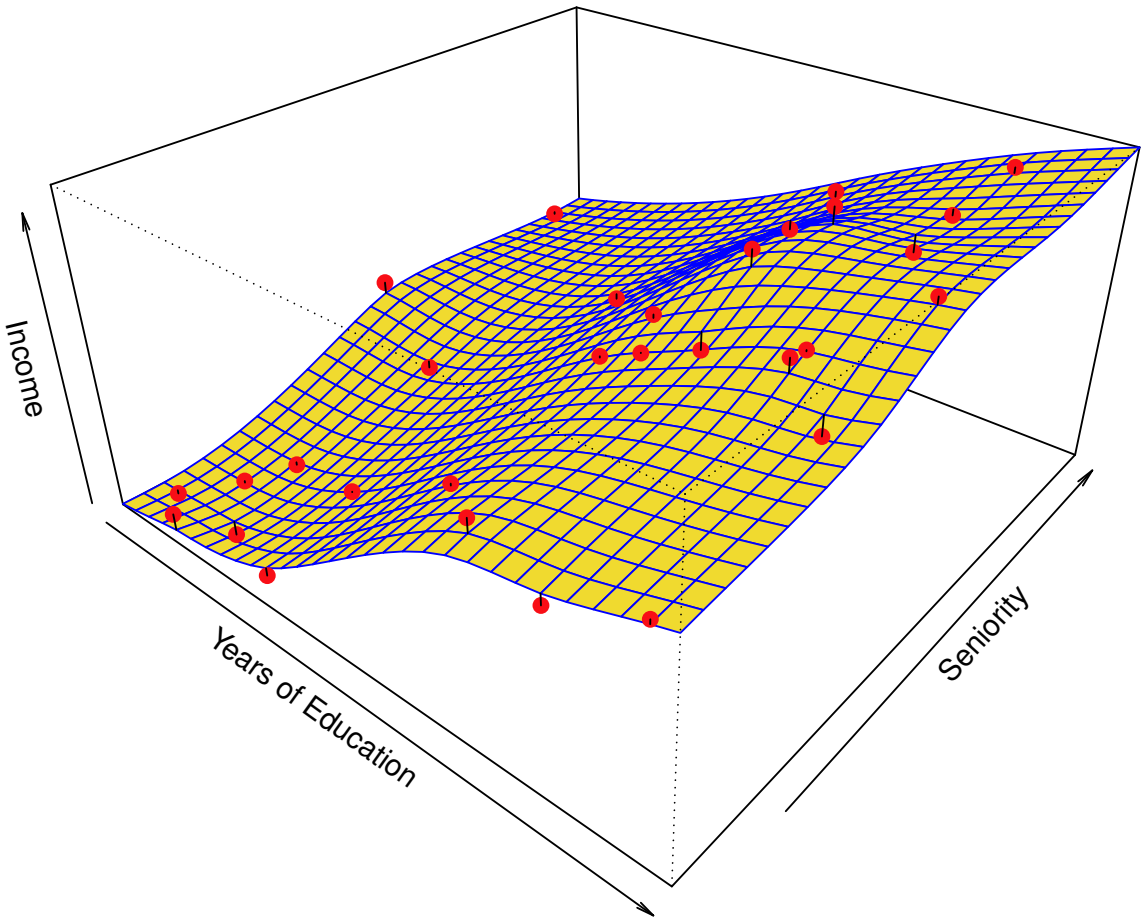
$$\text{income} = f(\text{education}, \text{seniority}) + \epsilon$$

f is the blue surface.

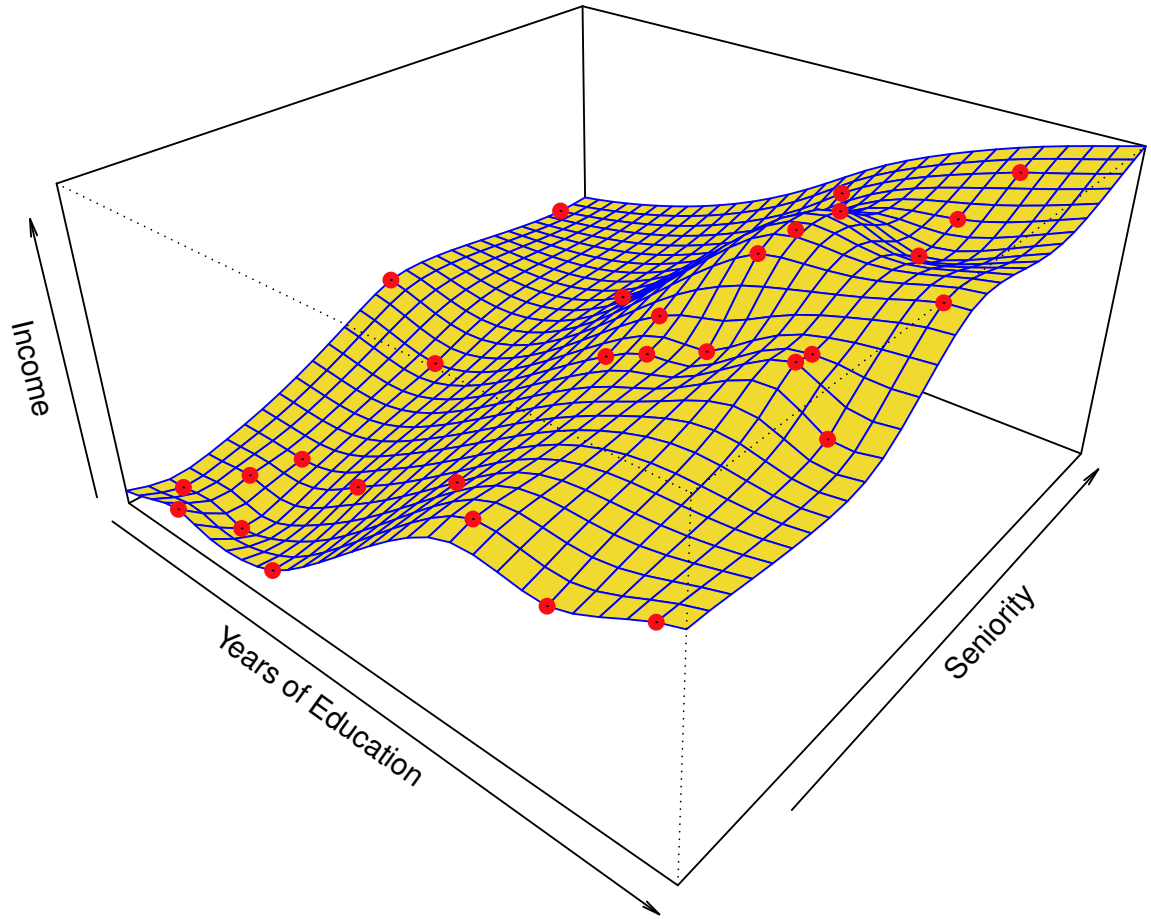


Linear regression model fit to the simulated data.

$$\hat{f}(\text{education}, \text{seniority}) = \hat{\beta}_0 + \hat{\beta}_1 \times \text{education} + \hat{\beta}_2 \times \text{seniority}$$



More flexible regression model $\hat{f}(\text{education}, \text{seniority})$ fit to the simulated data.



Even more flexible nonlinear regression model $\hat{f}(\text{education}, \text{seniority})$ fit to the simulated data. Here the fitted model makes no errors on the training data! Also known as **overfitting**.

Some trade-offs

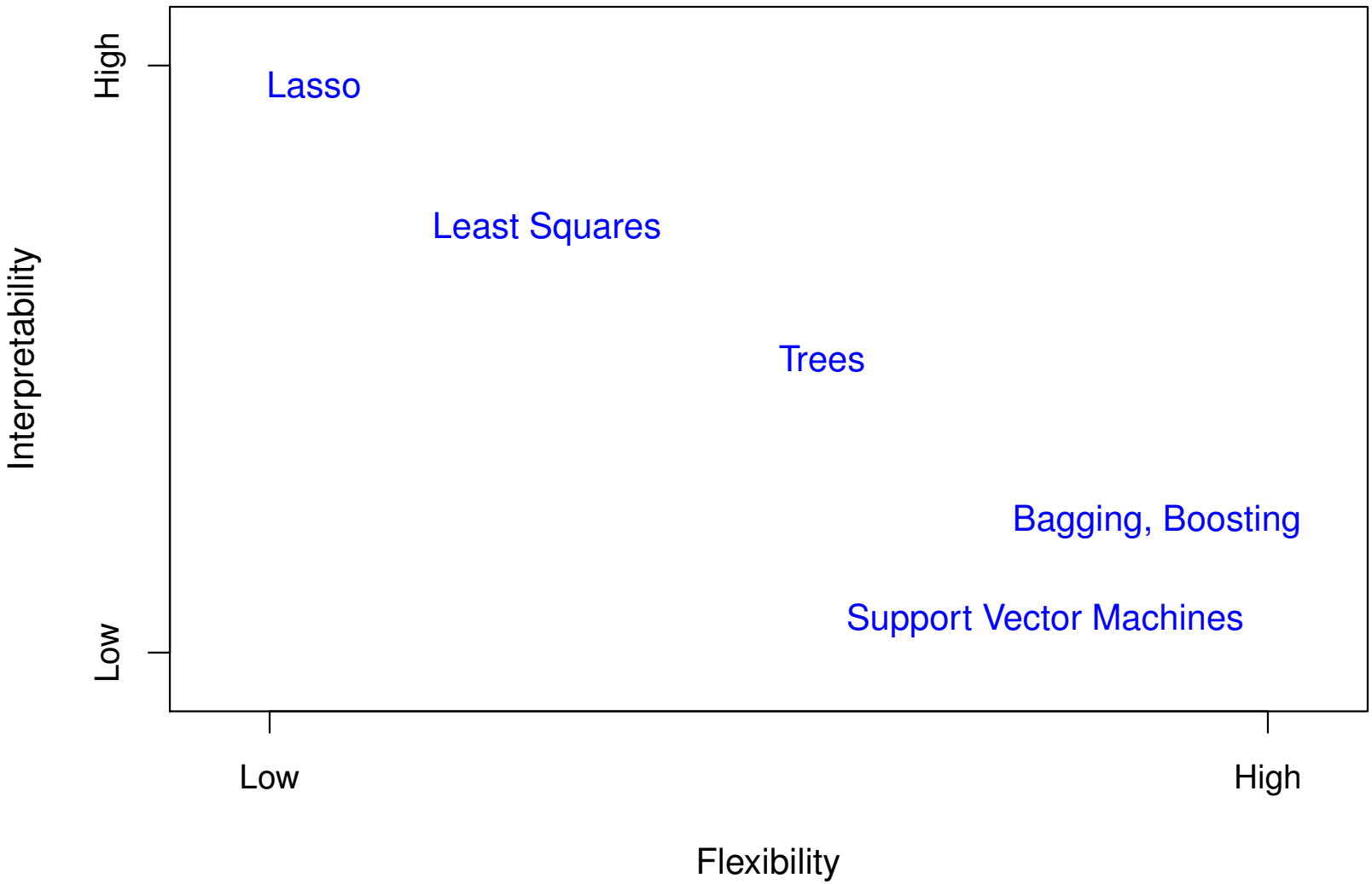
- Prediction accuracy versus interpretability. Linear models are easy to interpret; non-linear are not.
- Good fit versus over-fit or under-fit.
- Why under-fitting is bad?

Some trade-offs

- Prediction accuracy versus interpretability. Linear models are easy to interpret; non-linear are not.
- Good fit versus over-fit or under-fit.
- Why under-fitting is bad? — cannot predict well in training
- Why over-fitting is bad?

Some trade-offs

- Prediction accuracy versus interpretability. Linear models are easy to interpret; non-linear are not.
- Good fit versus over-fit or under-fit.
- Why under-fitting is bad?
- Why over-fitting is bad?
- How do we know when the fit is just right?
- Parsimony versus black-box. We often prefer a simpler model involving fewer variables over a black-box predictor involving them all.


Interpretability
High
Low
Lasso
Least Squares
Trees
Bagging, Boosting
Support Vector Machines
Low
High
Flexibility

Assessing Model Accuracy

- Suppose we fit a model $\hat{f}(x)$ to some training data $Tr = \{x_i, y_i\}_1^n$, and we wish to see how well it performs.
  - We could compute the average squared prediction error over Tr:

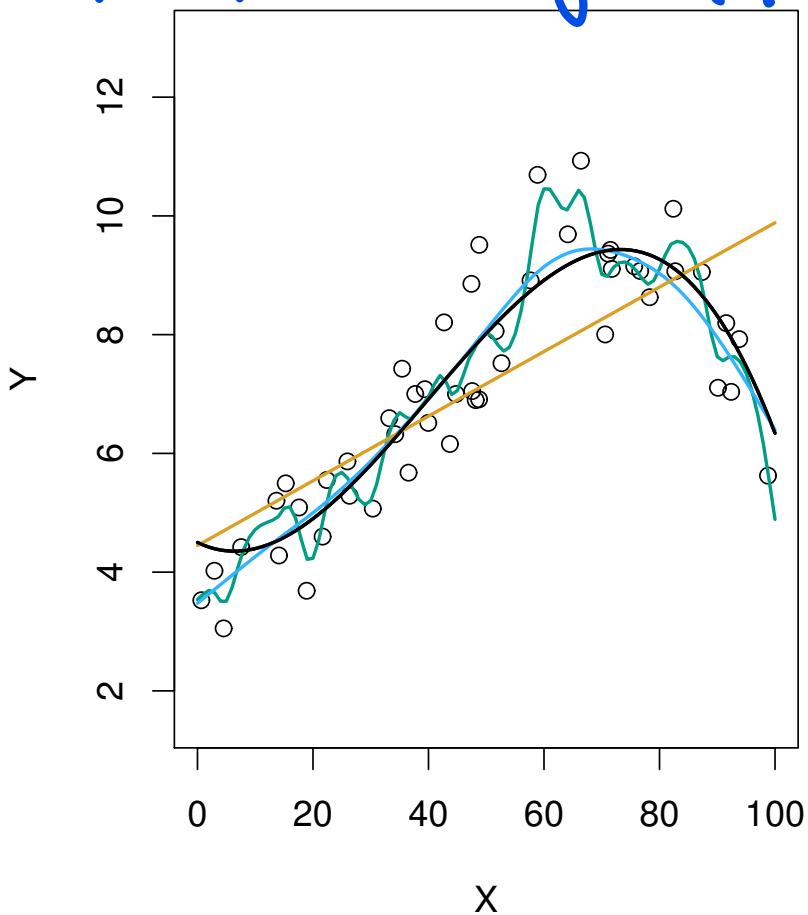
$$MSE_{Tr} = Ave_{i \in Tr}[y_i - \hat{f}(x_i)]^2$$

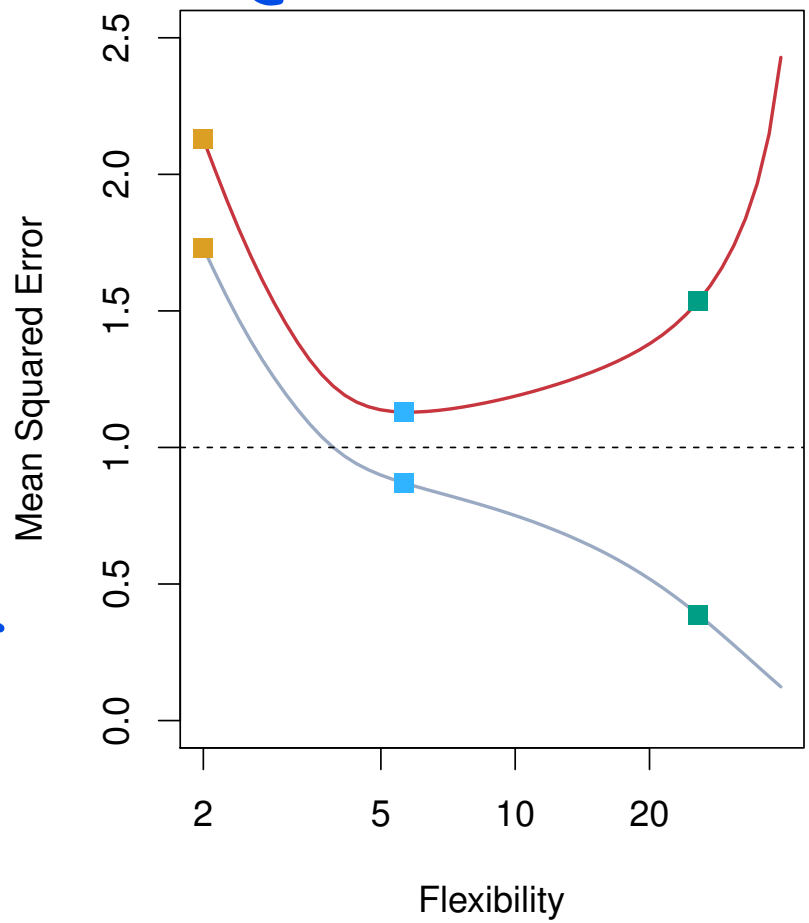
- This may be biased toward more overfit models.
  - Instead we should, if possible, compute it using fresh **test** data $Te = \{x_i, y_i\}_1^m$:

$$MSE_{Te} = Ave_{i \in Te}[y_i - \hat{f}(x_i)]^2$$

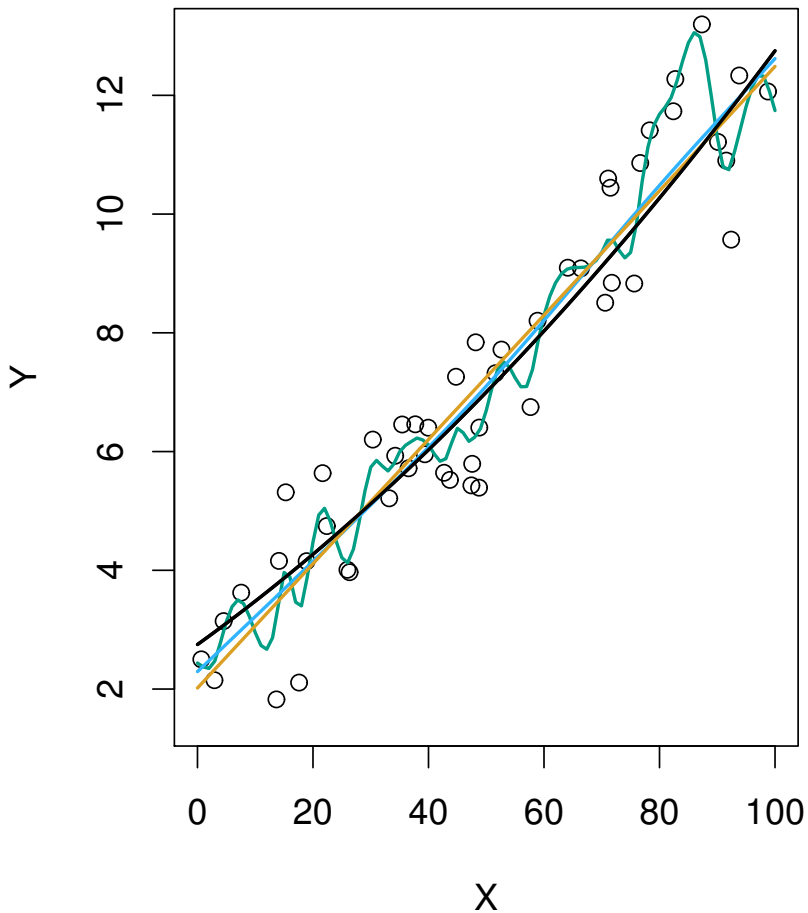
STOP Thursday 1/20/22 (Week 1, Lecture 2)

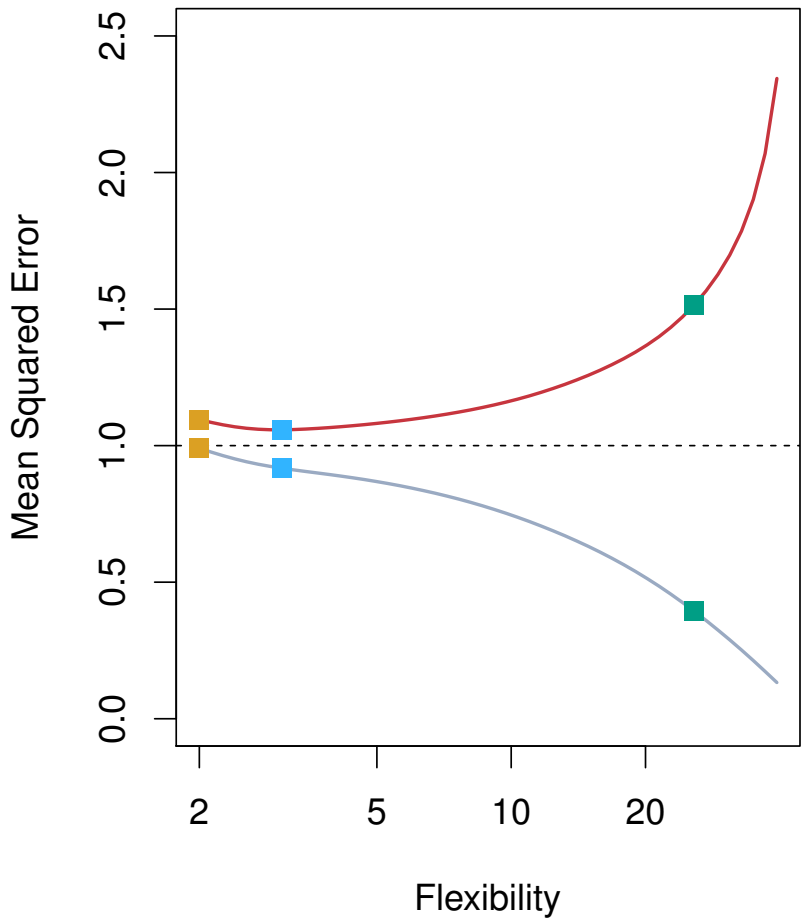
START Tuesday 1/25/22 (Week 2, Lecture 3)



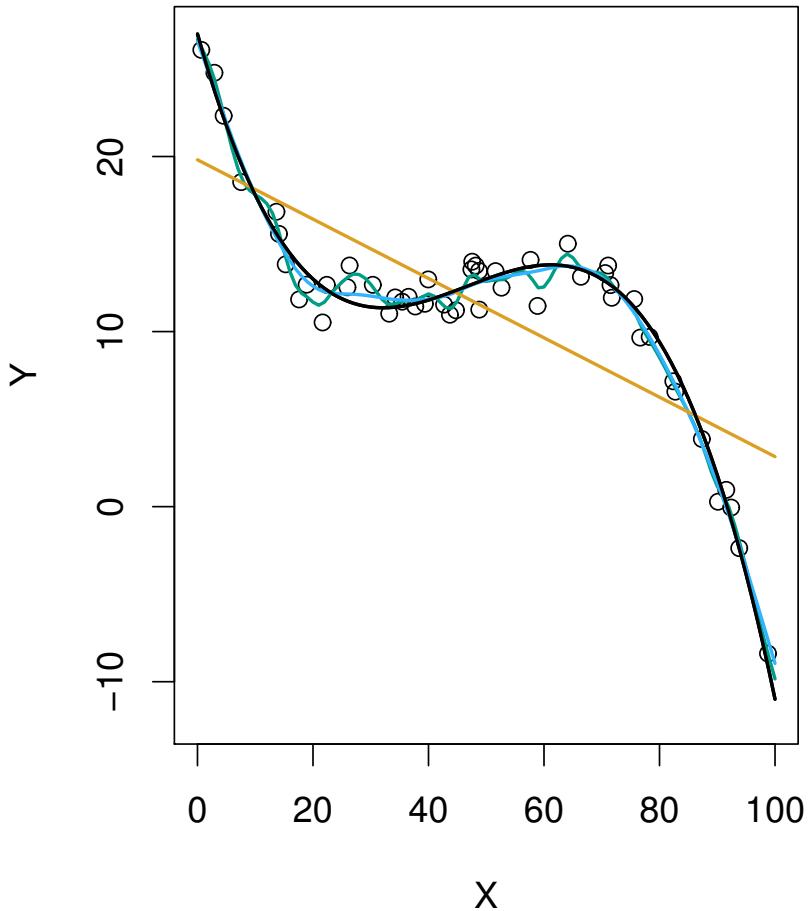


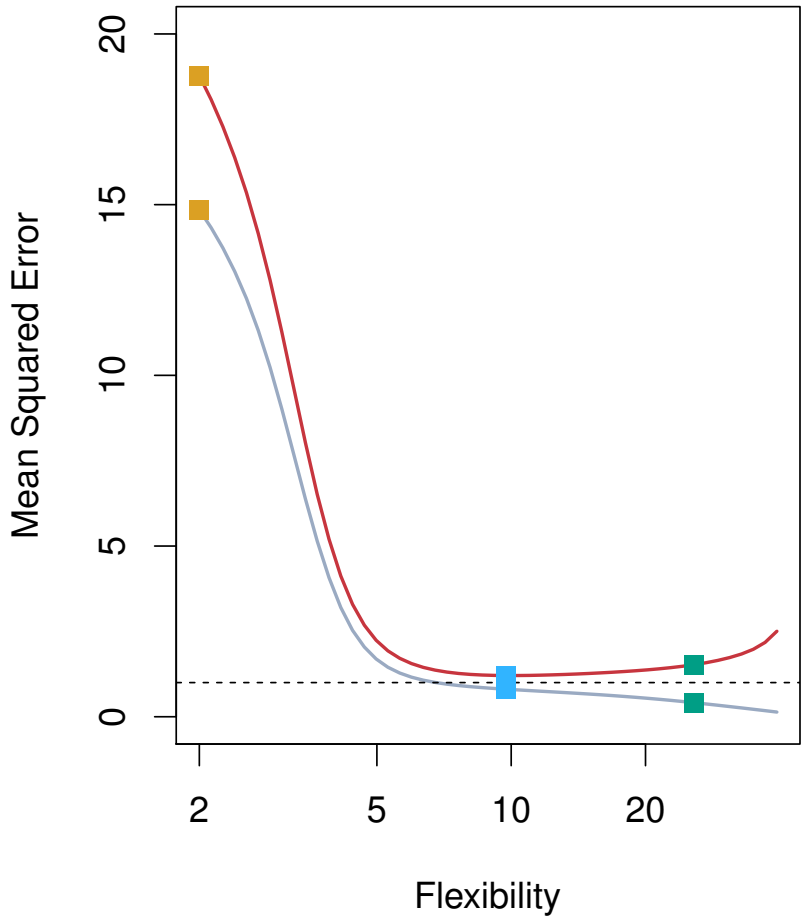
Black curve is truth. Red curve on right is MSE_{Te}, grey curve is MSE_{Tr}. Orange, blue and green curves/squares correspond to fits of different flexibility.





Here the truth is smoother, so the smoother fit and linear model do really well.





Here the truth is wiggly and the noise is low, so the more flexible fits do the best.

Bias-Variance Trade-off

- Suppose we have fit a model $\hat{f}(x)$ to some training data Tr, and let (x_0, y_0) be a test observation drawn from the population. If the true model is $Y = f(X) + \epsilon$ (with $f(x) = E(Y|X = x)$), then

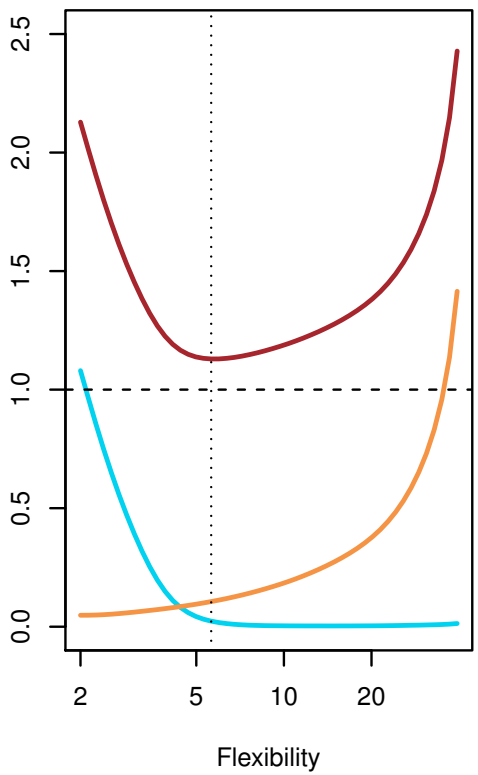
$$E\left(y_0 - \hat{f}(x_0)\right)^2 = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon).$$

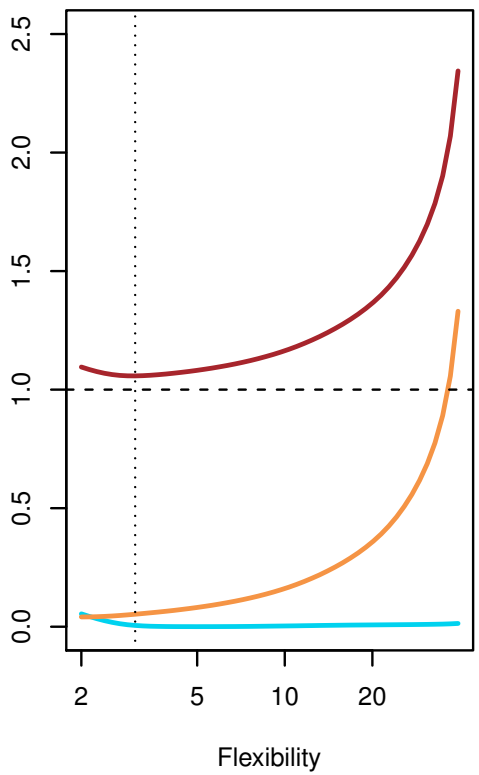
  The expectation averages over the variability of y_0 as well as the variability in Tr. Note that $Bias(\hat{f}(x_0)) = E[\hat{f}(x_0)] - f(x_0)$.

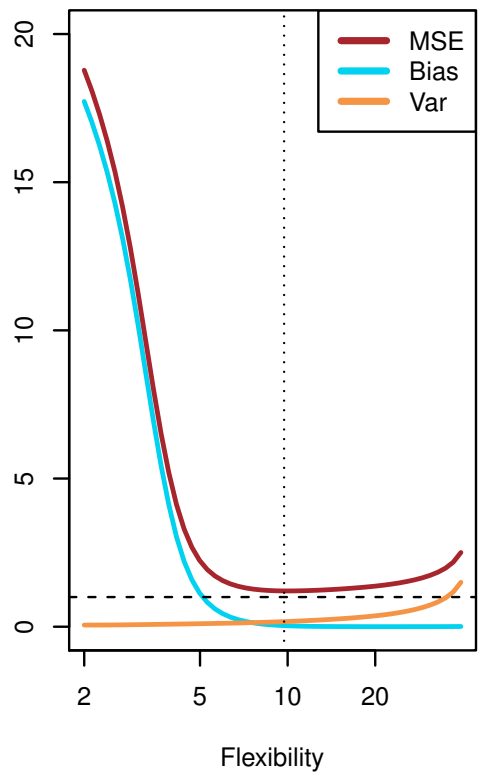
- Typically as the **flexibility** of $\hat{f}$ increases, its variance increases, and its bias decreases. So choosing the flexibility based on average test error amounts to a **bias-variance trade-off**.

Bias-variance trade-off for the three examples

$$MSE = Var(\hat{f}(x_0)) + Bias(\hat{f}(x_0))^2 + Var(\epsilon)$$







Classification Problems

Here the response variable Y is **qualitative** — e.g. email is one of $\mathcal{C} = (\text{spam}, \text{ham})$ (ham =good email), digit class is one of $\mathcal{C} = \{0, 1, \ldots, 9\}$. Our goals are to:

- Build a classifier $C(X)$ that assigns a class label from $\mathcal{C}$ to a future unlabeled observation X.
- Assess the uncertainty in each classification
- Understand the roles of the different predictors among $X = (X_1, X_2, \ldots, X_p)$.

Bayes optimal classifier

In regression, recall the ideal function is $E[Y|X_i]$

- Is there an ideal $C(X)$? Suppose the K elements in $\mathcal{C}$ are numbered $1, 2, \ldots, K$. Let

$$p_k(x) = Pr(Y = k|X = x), k = 1, 2, \ldots, K.$$

- These are the **conditional/posterior class probabilities** at x. Suppose those class probabilities are **known**, the **Bayes optimal** classifier at x is

$$C(x) = j \text{ if } p_j(x) = \max\{p_1(x), p_2(x), \ldots, p_K(x)\}$$

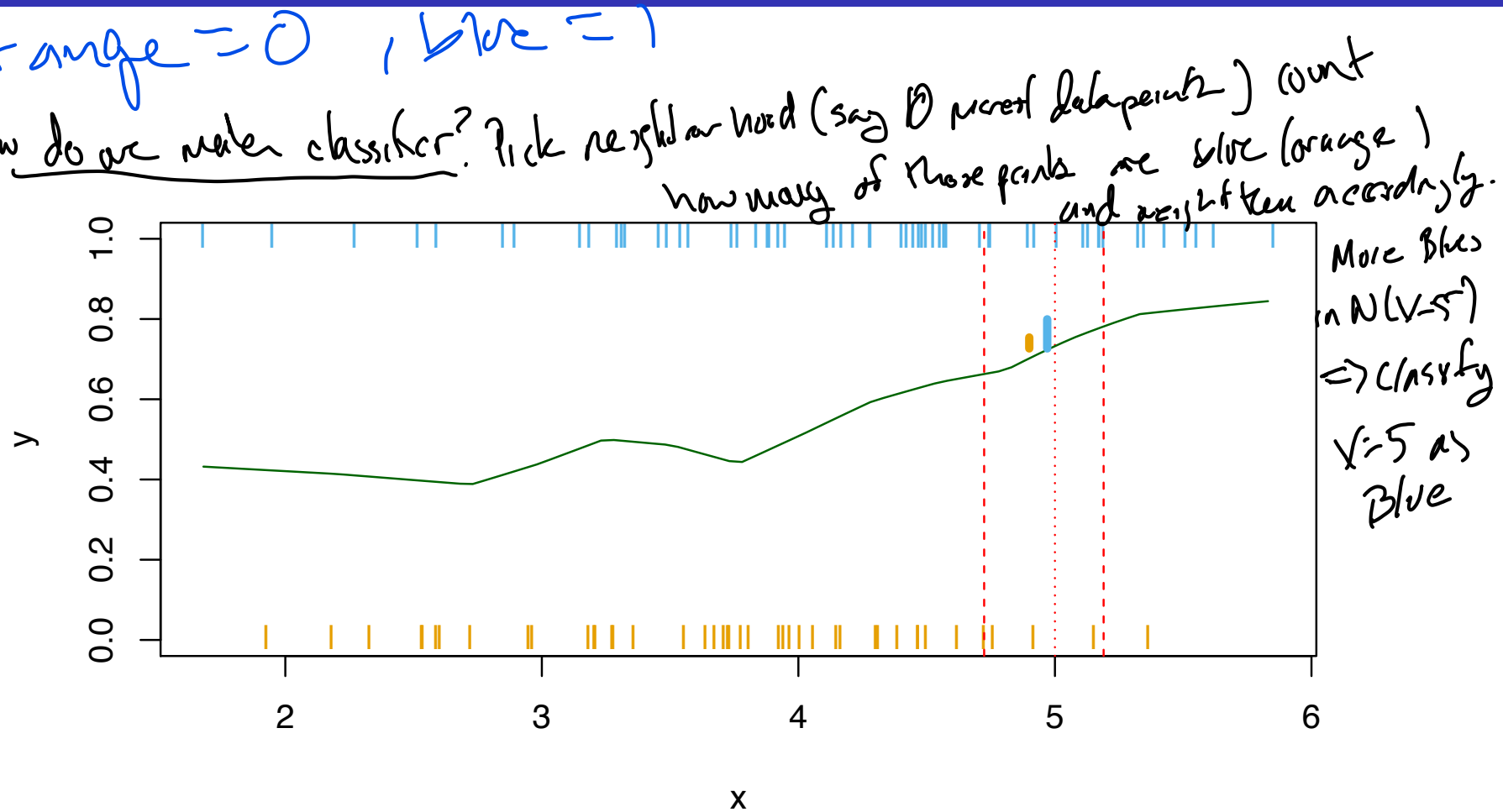
the most likely class for

- See

`BayesOptimal_handwritten_notes.pdf`

for more explanations.

Nearest neighbor classifier



Nearest-neighbor averaging can be used as before. Also breaks down as dimension grows.

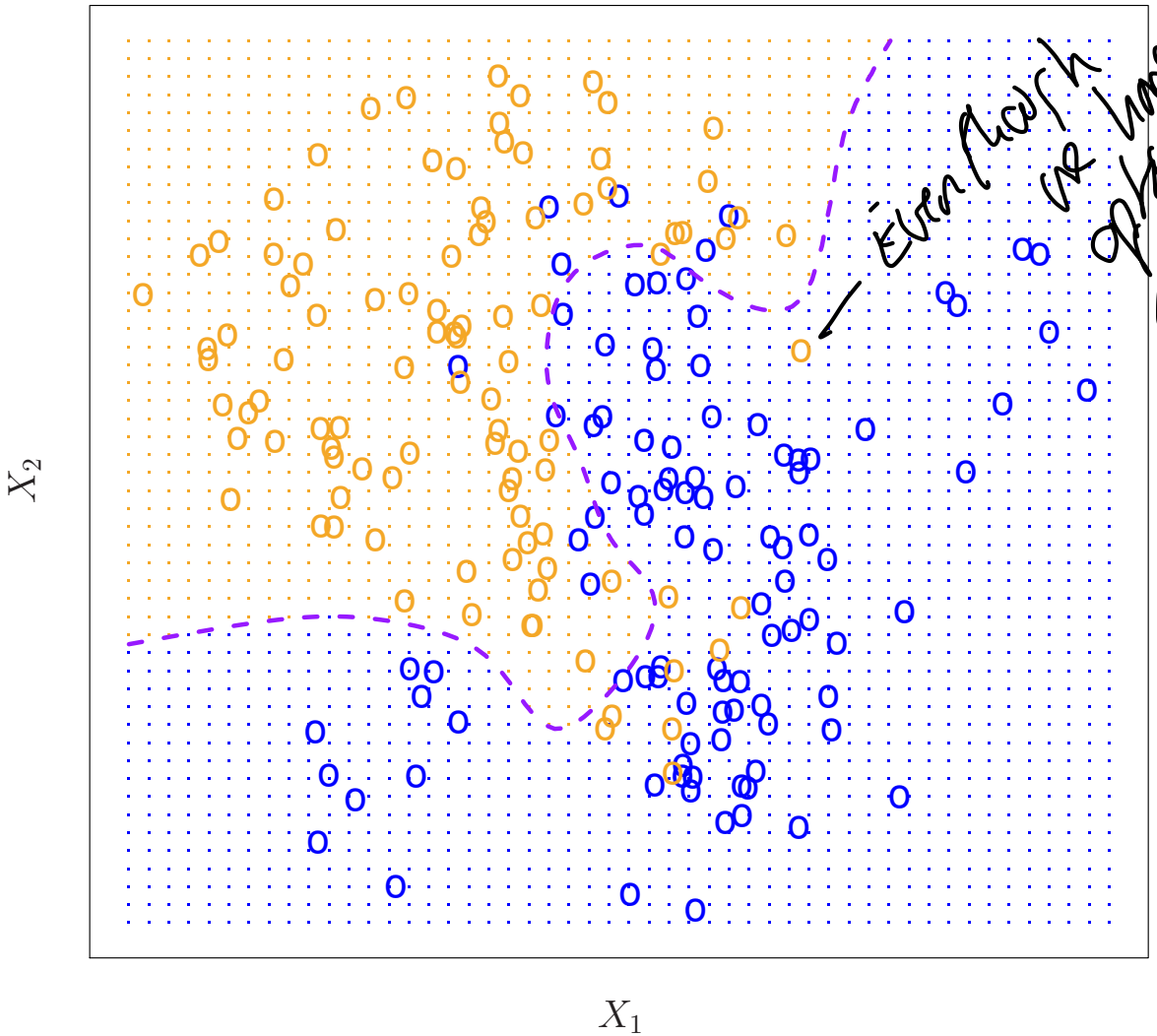
Classification: some details

- Typically we measure the performance of $\hat{C}(x)$ using the misclassification error rate:

$$Err_{Te} = Ave_{i \in Te} I[y_i \neq \hat{C}(x_i)]$$

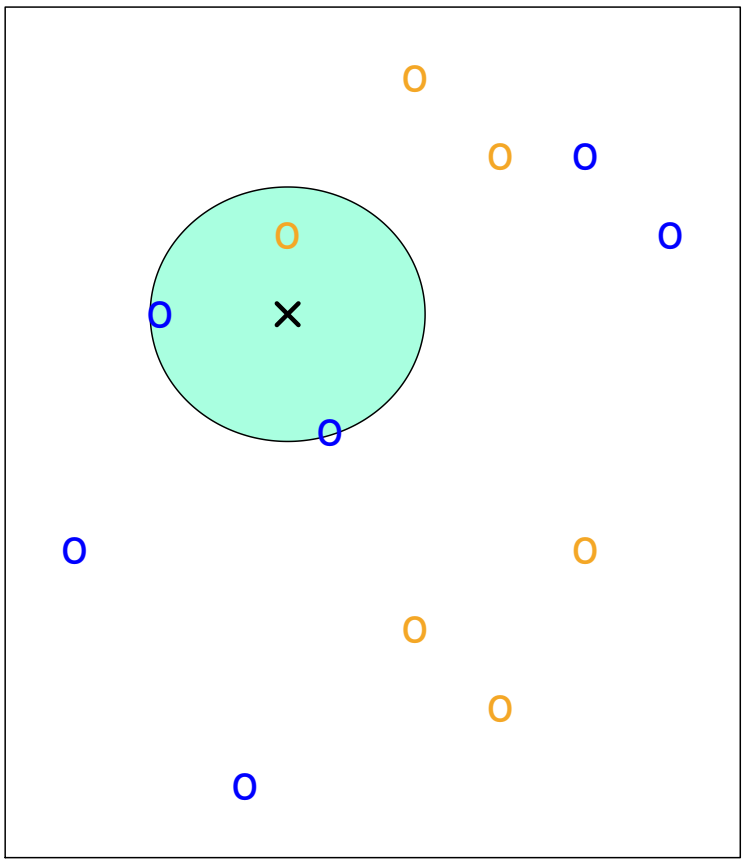
- The Bayes classifier (using the true $p_k(x)$) has smallest error (in the population).
- Support-vector machines build structured models for $C(x)$.
- We will also build structured models for representing the $p_k(x)$. e.g. Logistic regression.

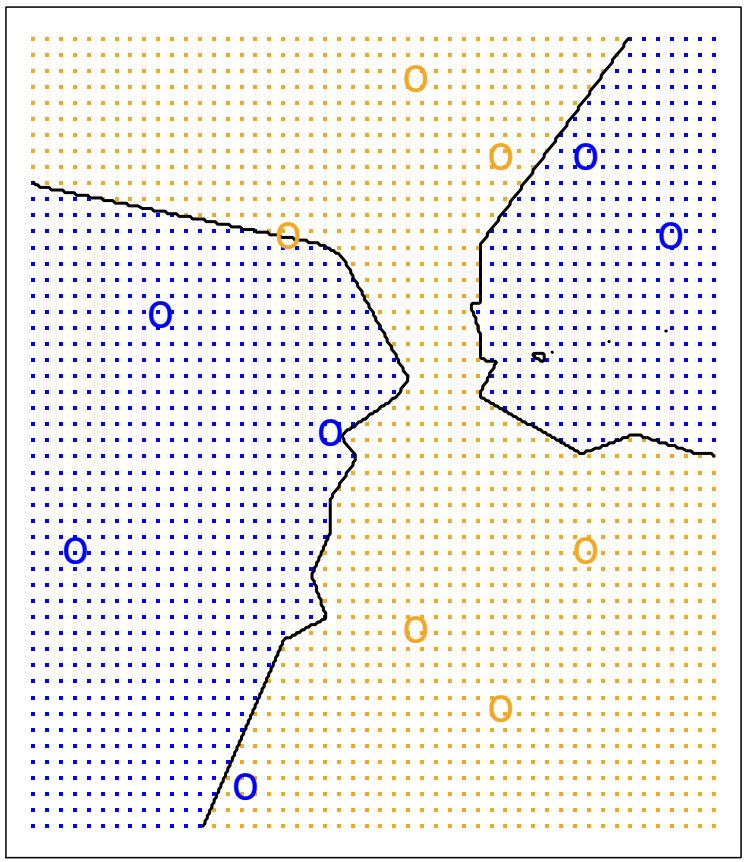
Simulated example: K-nearest neighbors in two dimensions



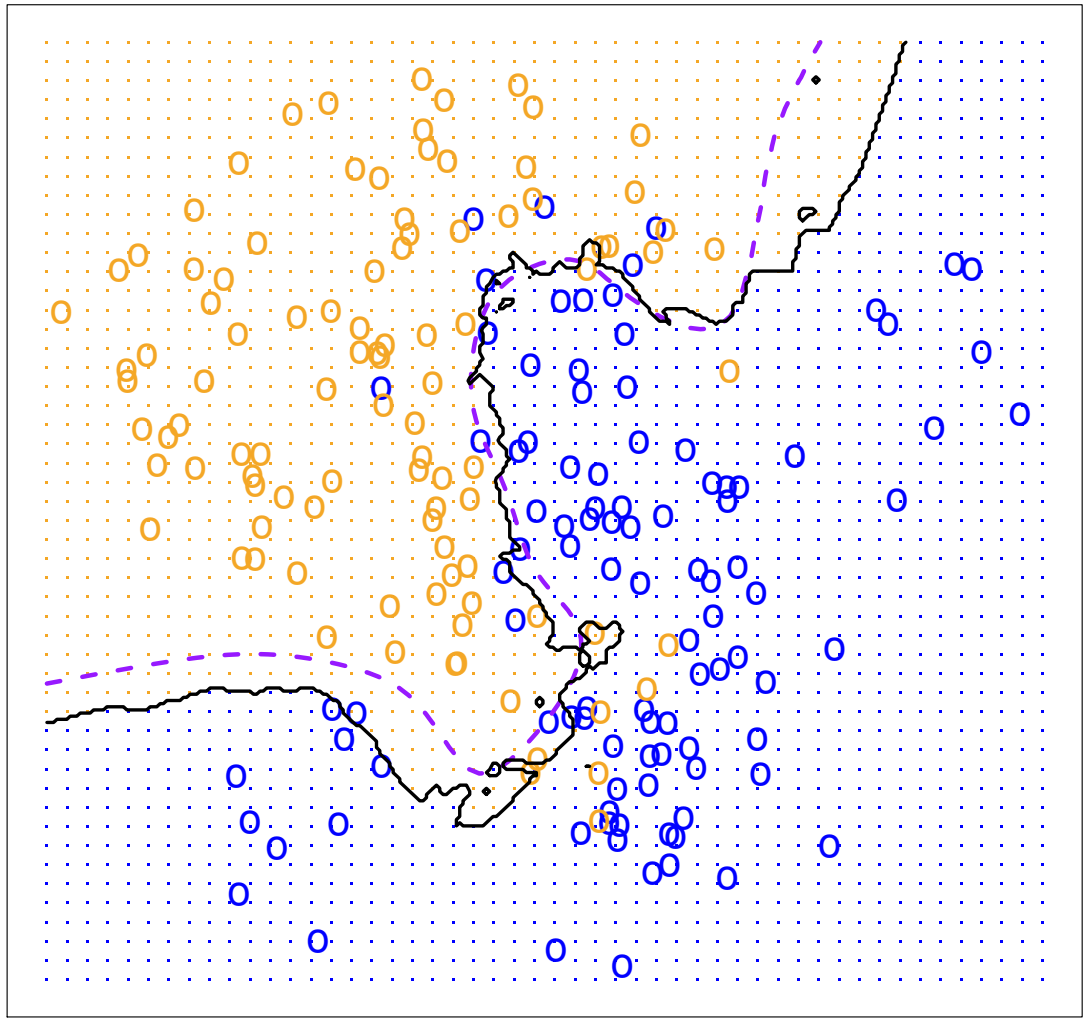
Purple dashed line: Bayes decision boundary.

Illustration of KNN with $K = 3$



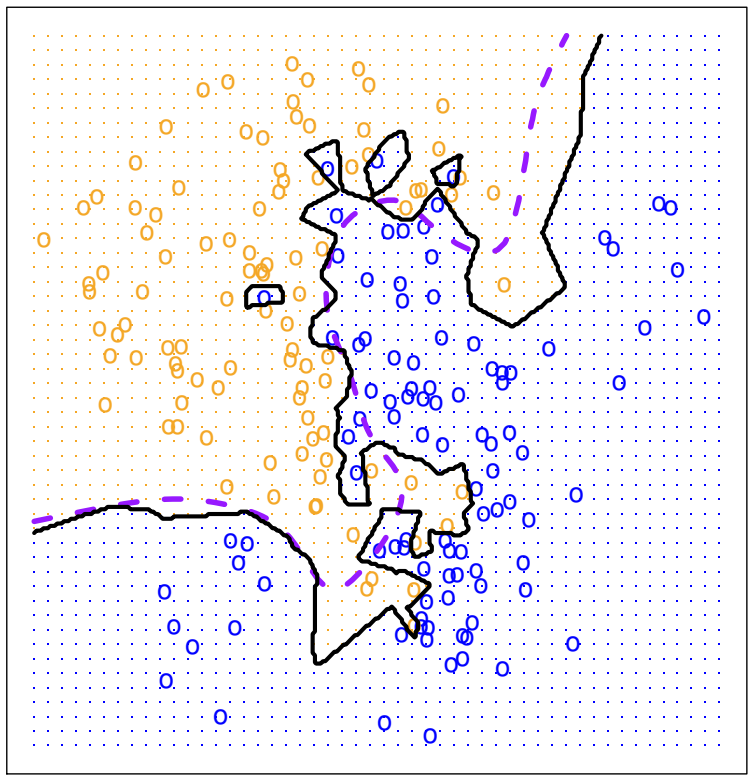


KNN: K=10

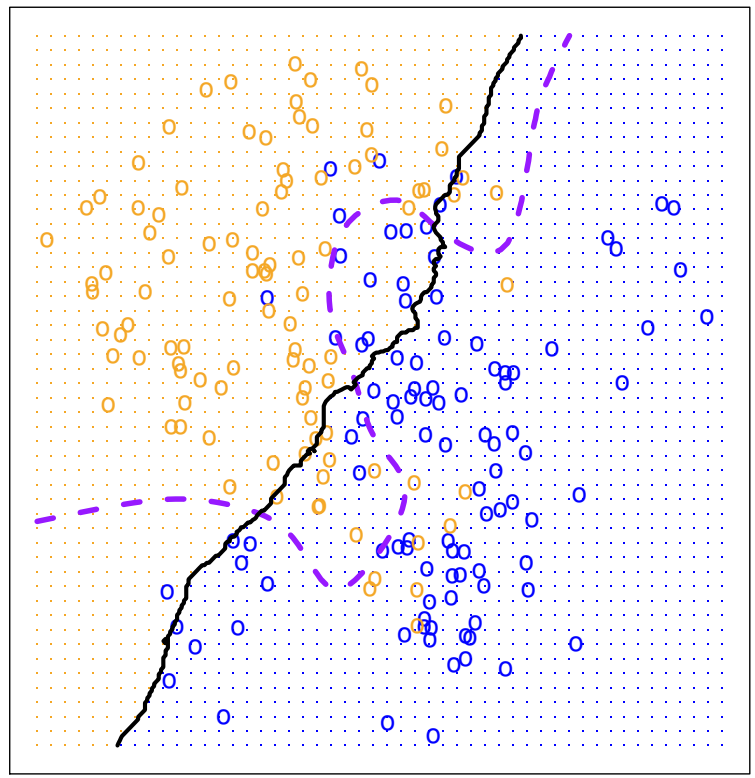


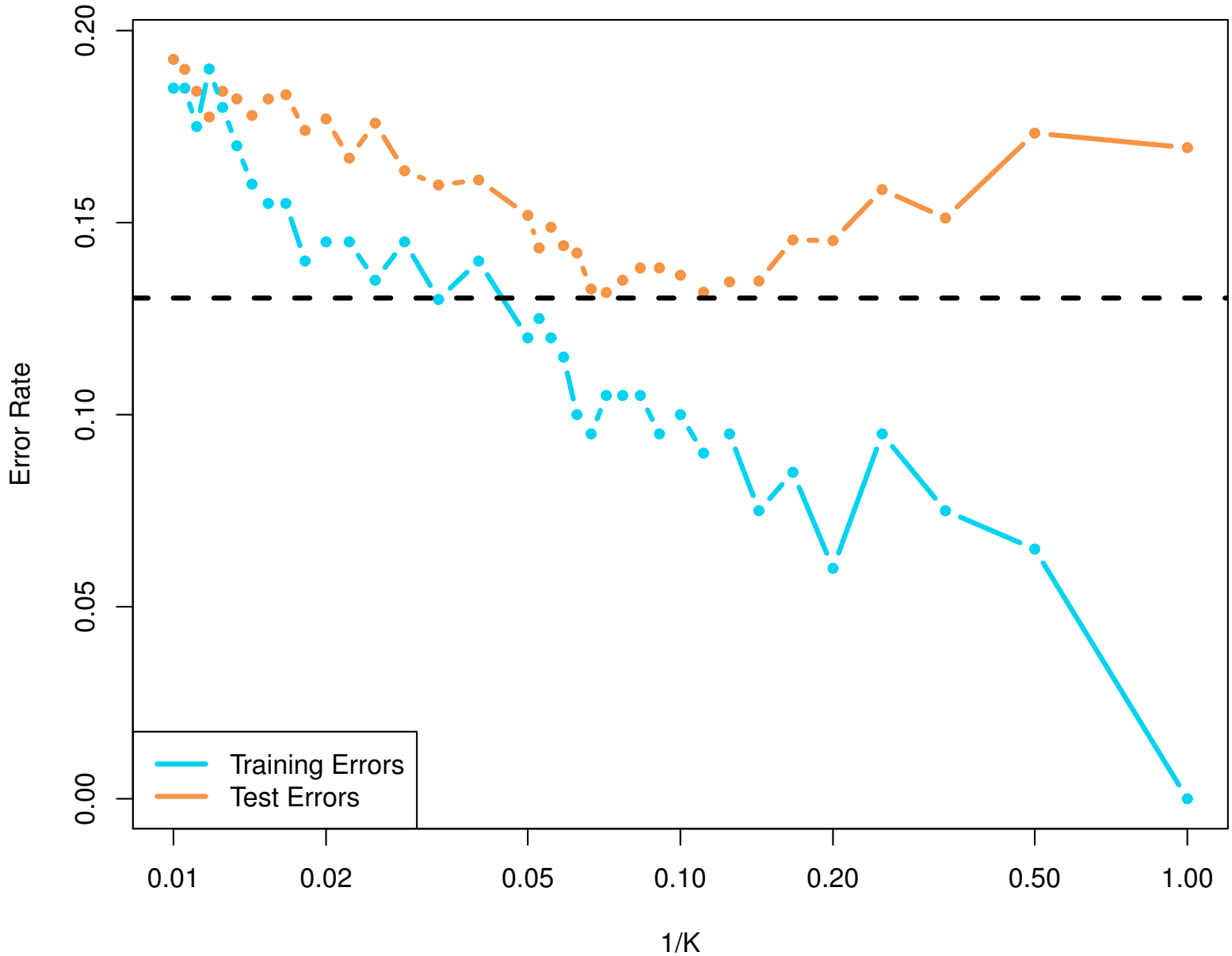
Smaller $K \Rightarrow$ more flexible
larger $K \Rightarrow$ less flexible.

KNN: K=1



KNN: K=100




Error Rate
0.00
0.05
0.10
0.15
0.20
0.01
0.02
0.05
0.10
0.20
0.50
1.00
1/K
Training Errors
Test Errors