Machine Learning and Pattern Recognition

Basic Concepts

Fabio G. Cozman

September 9, 2019

Varieties of statistical learning

- Supervised learning (mostly classification and regression).
- Unsupervised learning (mostly clustering and representation learning).
- Semi-supervised learning.
- Reinforcement learning.
- Another dimension: active learning.
- Another dimension: online and offline (batch) learning.

- There are too many approaches and variants; in this course we will select a few of them, but we will also give some space to traditional model-based methods.

Data

- A collection of data is a *dataset*.
- Usually a database contains tables: each row contains observed values for variables $X_1, \ldots, X_n$.
- If a variable is never observed, it is *hidden*; often referred to as a *latent* variable.
- We have *missing data* whenever a variable is not observed.
  - Missing data may be missing at random (MAR).
  - Missing data may be missing due to some systematic reason.
  - (Could be: sensor limits, sensor failure, typing mistakes, absent subject, etc.)

Data preparation

- Usual tasks:
  - merging tables (data integration),
  - understanding variables (tables, images, graphs, time series, text, audio),
  - discretizing and normalizing data,
  - turning categories into numbers (indexing, one-hot encoding),
  - removing outliers,
  - handling missing data,
  - feature selection and feature engineering.
- Also, some exploratory analysis:
  - box-plots, scatter plot matrix,
  - correlation analysis.
- And baseline models to get a sense of what can be expected for the task of interest.
- Often takes 80% of project time.

Handling missing data

- Removal (line/column).
- Imputation by mean, mode.
- Regression from other variables.
- Prediction if part of time series.
- Estimation from other variables and background model.

- Basic strategy: try several techniques, select the best...
- Often useful to visit data collection system to understand the missing data process and to think about transformations on the data.

Feature engineering

- First, feature selection: several techniques to be discussed later...
- Features can be transformed to guarantee better properties (for instance, normality).
- Also, features can be built out of other features.
- Some techniques are affected by scale (for instance, kNN and neural networks), so normalization may be needed.

Normalization

- Linear: $x' = (x - \min(x))/(\max(x) - \min(x))$.
- z-score:
  $x' = (x - \text{mean}(x))/\text{standard-deviation}(x)$.
- These schemes are sensitive to outliers.
- Other ideas: use median and quartiles (more computational effort).

Unbalanced datasets

- Problem: some labels are rare compared to others.
- Important to use performance measures that are sensitive to this (for instance, precision/recall/F1... to be seen later).
- Often useful: over-sample the labels that are rare during training.

Supervised/unsupervised classification

- Consider a database containing records $(\mathbf{X}, Y)$.
- Variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ are observed; they are the *features/attributes*.
- Labels are values of a *class variable* Y.

Supervised/unsupervised classification

- Consider a database containing records $(\mathbf{X}, Y)$.
- Variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ are observed; they are the *features/attributes*.
- Labels are values of a *class variable* Y.
- When every record contains a label (no missing label), we have *supervised learning*.
- Then:
  - When every label Y is missing, we have *unsupervised learning*.
  - General case: some labels missing: *semi-supervised learning*.
- Note: in all cases, records may also contain missing data.

Classification

- Classifier is a function $g(X_1, \ldots, X_n)$ from attributes to *labels*.
- Thus:

$$\hat{Y} = g(X_1, \ldots, X_n)$$

  and the goal is to have $\hat{Y}$ equal to Y.
- So, learning here is: produce a function g using data — this is referred to as *training* the classifier.

Error rates

- To evaluate classifiers, usual metric is the probability of error:

$$e_g = \mathbb{P}(Y \neq g(\mathbf{X})).$$

  So, we use the joint distribution $p(\mathbf{X}, Y)$.
- The *training (empirical) error rate* is

$$\hat{e}_g = \frac{1}{N} \sum_{i=1}^{N} I_{Y \neq g(\mathbf{x})}(\mathbf{X}, Y).$$

Training and testing data

- Usually a classifier is learned using a portion of the available data: the *training data*.
- The remainder of the data are used to test the classifier (for instance, by estimating the probability of error): the *testing data*.

Overfitting

- Testing data is important to detect *overfitting*.
- That is, classifier is excellent for the training data but fails for other data.
- Example: 1-nearest-neighbor.

Overfitting: 1-NN

- Simple idea: $g(\mathbf{X})$ is equal to the label of closest neighbor of $\mathbf{X}$, for some distance.
- Zero probability of error in the training set.
- Possibly high probabilty of error in a testing set.

Cross-validation

- If not enough data to create a *holdout* to test, at least *cross-validation* must be used.
- Idea: separate a fraction of the data for testing, then repeat over the whole database.
- Five-fold or ten-fold cross-validation are very common.
- Leave-one-out validation is also common but it demands more computation.

Optimal classification

- The optimal classifier (called the *Bayes* classifier) is:

$$g^* = \arg\min_g e_g = \arg\min_g \mathbb{P}(Y \neq g(\mathbf{X})) .$$

- If Y were binary, and we had $p(\mathbf{X}, Y)$, what is g^*?

The Bayes classifier

- Suppose we observe $\mathbf{X} = \mathbf{x}$.
- Then:

$$\mathbb{P}(Y \neq g(\mathbf{x})|\mathbf{X} = \mathbf{x}) = \begin{cases} \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) \\ \qquad \text{if } g(\mathbf{x}) = 0; \\ \mathbb{P}(Y = 0|\mathbf{X} = \mathbf{x}) \\ \qquad \text{if } g(\mathbf{x}) = 1. \end{cases}$$

The Bayes classifier

- Suppose we observe $\mathbf{X} = \mathbf{x}$.
- Then:

$$\mathbb{P}(Y \neq g(\mathbf{x})|\mathbf{X} = \mathbf{x}) = \begin{cases} \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) \\ \qquad \text{if } g(\mathbf{x}) = 0; \\ \mathbb{P}(Y = 0|\mathbf{X} = \mathbf{x}) \\ \qquad \text{if } g(\mathbf{x}) = 1. \end{cases}$$

- Thus choose:

$$g^*(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) < 1/2, \\ 1 & \text{if } \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) \geq 1/2. \end{cases}$$

The Bayes classifier and the Bayes error

- The Bayes classifier:

$$g^*(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) < 1/2, \\ 1 & \text{if } \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) \geq 1/2. \end{cases}$$

- The minimum e_g is the *Bayes error rate*, obtained with the Bayes classifier:

$$e_{g^*} = \mathbb{P}(Y \neq g^*(\mathbf{X}))$$

The Bayes classifier and the Bayes error

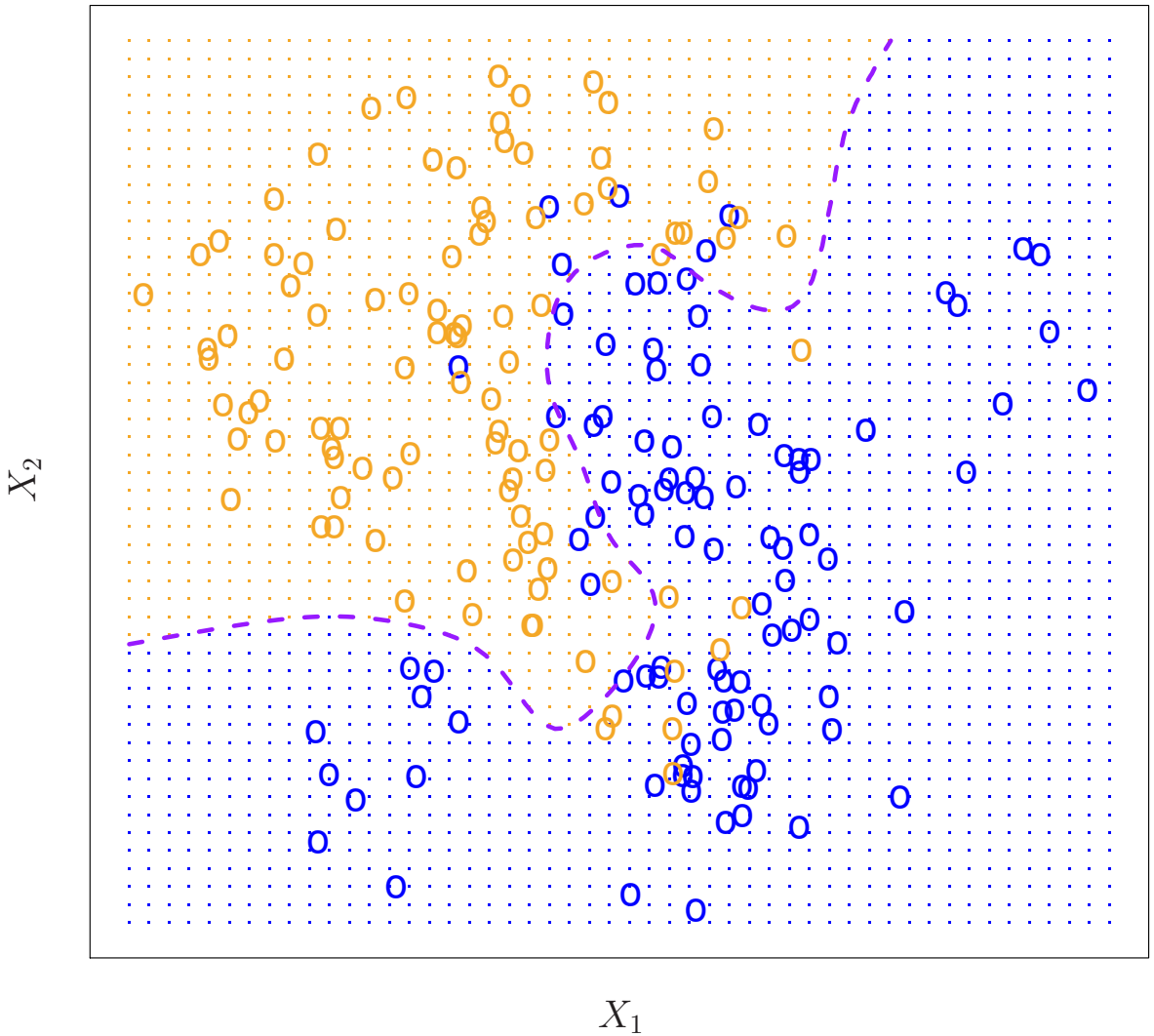
- The Bayes classifier:

$$g^*(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) < 1/2, \\ 1 & \text{if } \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) \geq 1/2. \end{cases}$$

- The minimum e_g is the *Bayes error rate*, obtained with the Bayes classifier:

$$\begin{aligned} e_{g^*} &= \mathbb{P}(Y \neq g^*(\mathbf{X})) \\ &= \mathbb{E}[\min(\mathbb{P}(Y = 1|\mathbf{X}), 1 - \mathbb{P}(Y = 1|\mathbf{X}))]. \end{aligned}$$

Bayes error



Plug-in classifiers

- In practice we usually do not have $p(\mathbf{X}, Y)$.
  - Common strategy is to estimate $p(\mathbf{X}, Y)$ and use it in the optimal classifier scheme.
  - Estimation may be *parametric* or *non-parametric*.

- Classifiers that represent only $\mathbb{P}(Y = 1|\mathbf{X})$ are called *diagnostic classifiers*; classifiers that represent $p(\mathbf{X}, Y)$ are called *generative classifiers*.

Warning

- If $p(\mathbf{X}, Y)$ is not given, there is no universally optimal method to generate classifiers.
- For any method, there is always a distribution $p(\mathbf{X}, Y)$ such that the method is worse than some other method.

Warning

- If $p(\mathbf{X}, Y)$ is not given, there is no universally optimal method to generate classifiers.
- For any method, there is always a distribution $p(\mathbf{X}, Y)$ such that the method is worse than some other method.
- Thus, it makes sense to look for "simple" and "intuitive" methods that work "most often".
- There is a trade-off betwen accuracy and interpretability; the most accurate technique may be too hard to interpret.

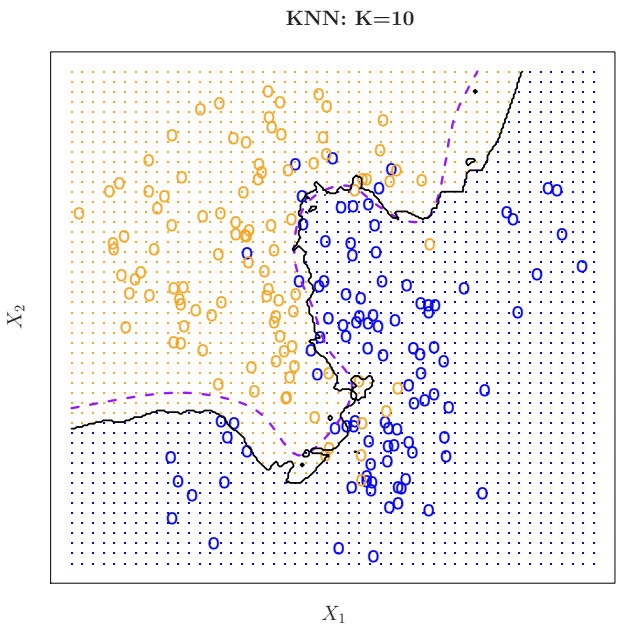
Example: Nearest neighbor

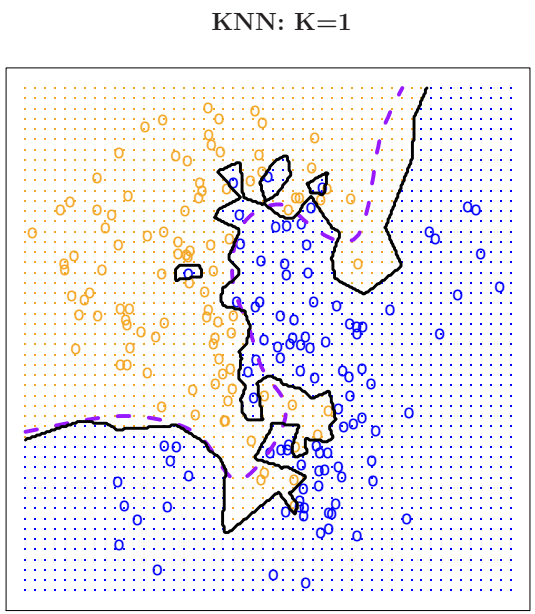
- Simple idea: $g(\mathbf{X})$ is equal to the majority of labels in a k neighborhood of $\mathbf{X}$, given some distance.
- If $k = k(n) \to \infty$ such that $k(n)/n \to 0$, then $\lim_{n\to\infty} \mathbb{E}[e_{g_n}] = e_{g^*}$ (very nice).
- Moreover, for 1-nearest neighbor, $\lim_{n\to\infty} \mathbb{E}[e_{g_n}] \leq 2e_{g^*}$ (amazing!).

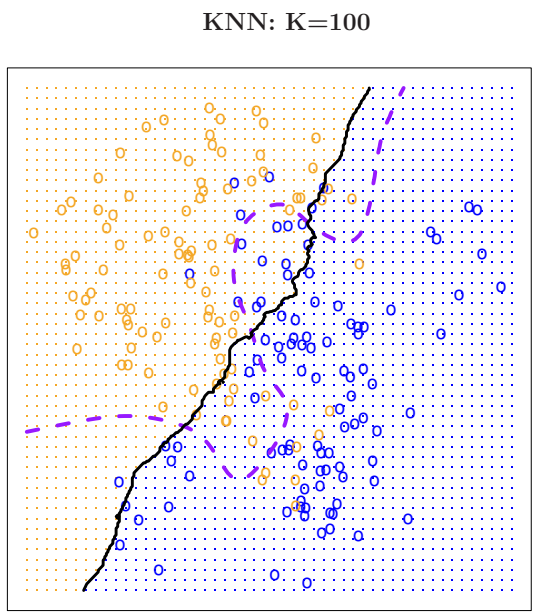
Digression

- If a method generates classifiers g_n for training data of size n, and $\mathbb{E}[e_{g_n}] \to e_{g^*}$, the method is *consistent*.
- Thus nearest neighbor classifiers are (in a special way) consistent.

10-NN; 1-NN and 100-NN







How to select k?

- There is quite a bit of theory about it, but no real theoretical guideline.
- Practical matter: use cross-validation.

In short, the basic schemes:

- There are two basic schemes when this distribution is not available but data is collected.
- **Plug-in:** The distribution is estimated and the classifier is built using the estimates.
- **Ad hoc:** The classifier is directly built from data, possibly using estimates to evaluate the process, maybe minimizing some criterion, maybe approximating some function.

A result on plug-in classification

- For estimated $\hat{\mathbb{P}}(Y = 1|\mathbf{X})$,

$$\mathbb{P}(Y \neq g(\mathbf{X})) - e_{g^*} \leq 2\mathbb{E}\left[\mathbb{P}(Y = 1|\mathbf{X}) - \hat{\mathbb{P}}(Y = 1|\mathbf{X})\right].$$

- To show this, we need some probability theory...

Regression

- Assume:
$$Y = f(\mathbf{X}) + \epsilon.$$
- One may want to *predict* Y for some observed $\mathbf{x}$.
- Or one may want to understand the relationship between $\mathbf{X}$ and Y (which features affect Y? is this relationship linear?). Often this is referred to as *statistical inference*.

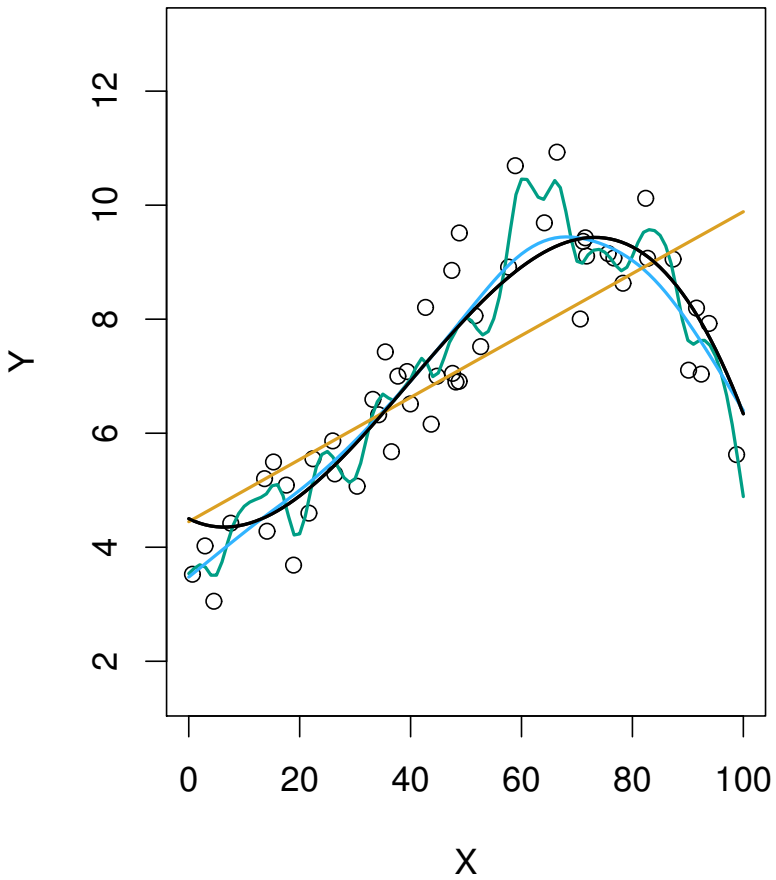
Expected quadratic error

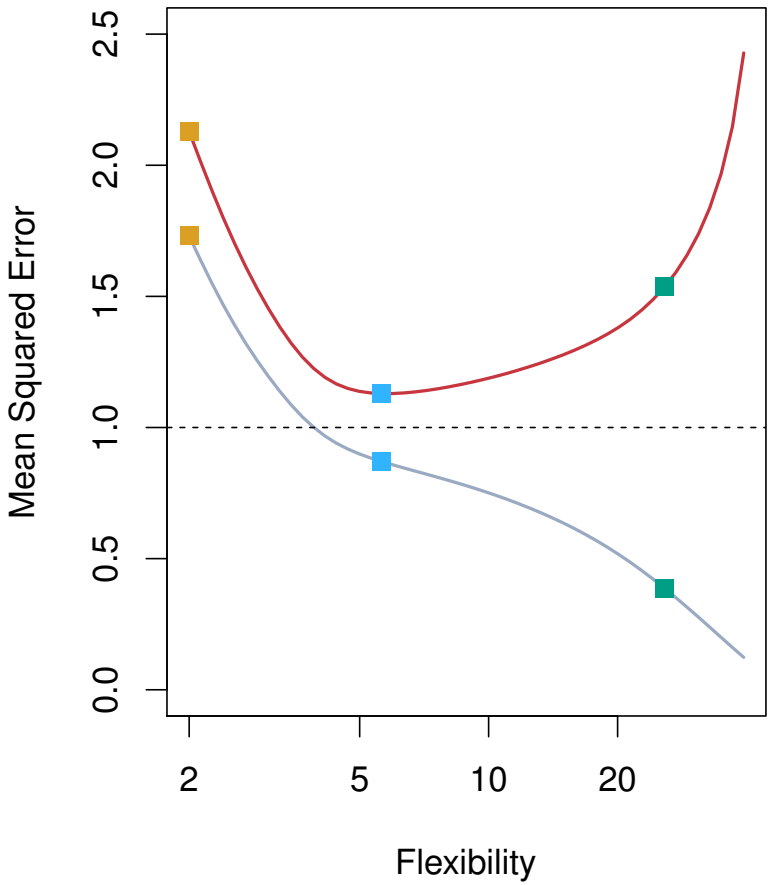
- We assume
$$Y = f(\mathbf{X}) + \epsilon,$$
where usually ϵ is independent zero mean "noise" with variance σ^2.
- We want
$$\hat{Y} = \hat{f}(\mathbf{X})$$
for some estimated $\hat{f}$.
- We evaluate by
$$\mathbb{E}\left[(Y - \hat{Y})^2\right]$$
or some similar measure.

Selecting $\hat{f}$

- It is nice to have a simple, easily interpretable $\hat{f}$.
- Too simple a $\hat{f}$ may lead to *underfitting*.
- But too complex a $\hat{f}$ may lead to *overfitting*.
  - Example: polynomial of degree 1000 interpolates 100 given points exactly, but does it poorly for other points.
- Usual choices: linear function, polynomial function, spline, piecewise function (tree-based).

Overfitting in regression





Assessing fit

- Monitor the *mean squared error*:

$$\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{f}(\mathbf{x}_i)\right)^2$$

  with respect to testing data.
- Statistical tests indicate when this quantity is "large" or "small".

Bias and variance at an observation

- Consider $\hat{Y} = \hat{f}(\mathbf{X}, D)$.
- Look at

$$\mathbb{E}\Big[(Y - \hat{Y})^2 | \mathbf{X} = \mathbf{x}\Big] = \mathbb{E}\Big[(Y - \hat{f}(X, D))^2 | \mathbf{X} = \mathbf{x}\Big]$$

where $g = f(\mathbf{x})$ and $h = \hat{f}(\mathbf{x}, D)$.

Bias and variance at an observation

- Consider

$$\mathbb{E}\left[(Y-\hat{Y})^2|\mathbf{X}=\mathbf{x}\right]=\mathbb{E}\left[(Y-\hat{f}(\mathbf{x},D))^2|\mathbf{X}=\mathbf{x}\right].$$

Bias and variance at an observation

- Consider

$$\mathbb{E}\Big[(Y-\hat{Y})^2|\mathbf{X}=\mathbf{x}\Big]=\mathbb{E}\Big[(Y-\hat{f}(\mathbf{x},D))^2|\mathbf{X}=\mathbf{x}\Big].$$

- Then $\mathbb{E}\Big[(Y-\hat{Y})^2|\mathbf{x}\Big]=$

$$\begin{aligned}
&= \mathbb{E}\big[(g-h)^2-2\epsilon(g-h)+\epsilon^2|\mathbf{x}\big]\\
&= \sigma^2+\mathbb{E}\big[(g-h)^2|\mathbf{x}\big]\\
&= \sigma^2+g^2-2g\mathbb{E}[h|\mathbf{x}]+\mathbb{E}\big[h^2|\mathbf{x}\big]\pm\mathbb{E}[h|\mathbf{x}]^2\\
&= \sigma^2+(g-\mathbb{E}[h|\mathbf{x}])^2+V[h|\mathbf{x}].
\end{aligned}$$

The bias/variance tradeoff

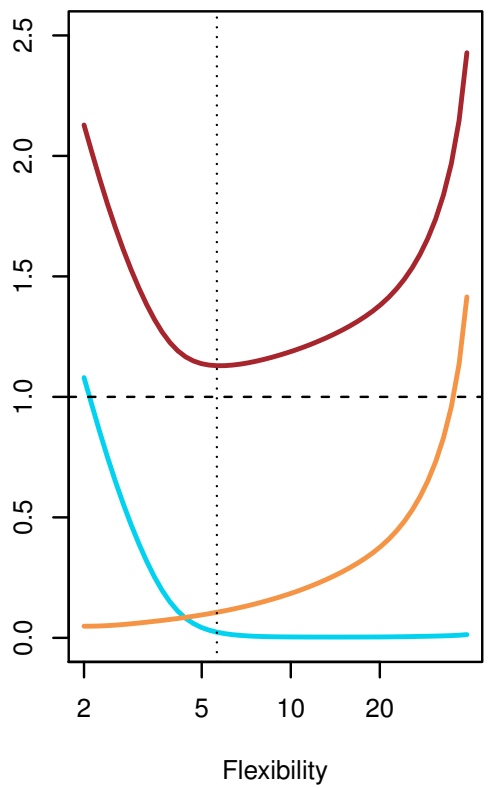
- We have:

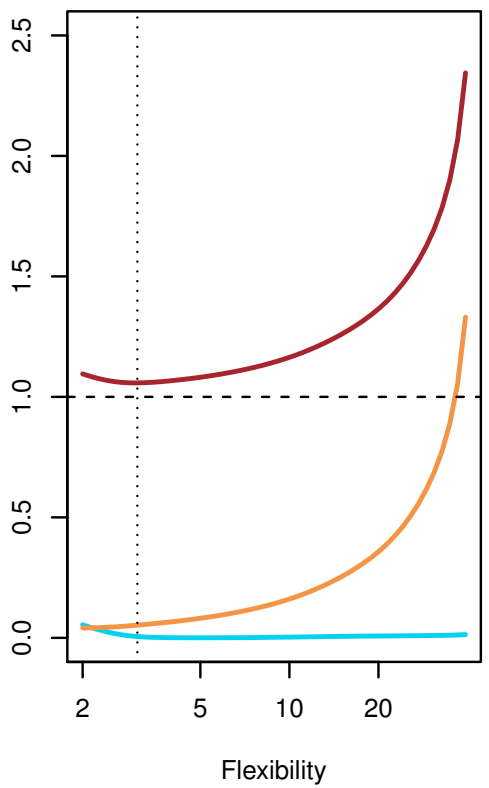
$$\mathbb{E}\left[(Y - \hat{Y})^2 | \mathbf{x}\right] = \sigma^2 + \mathbb{E}\left[\text{bias}^2\right] + \mathbb{E}[\text{variance}] \,.$$

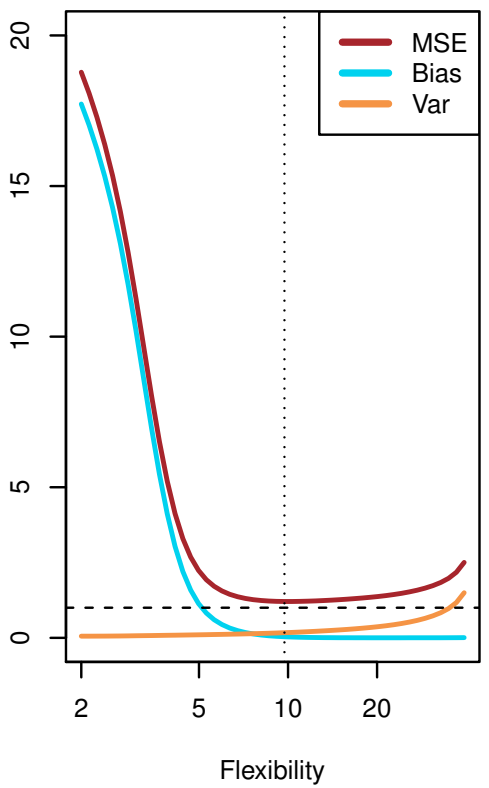
- Sometimes a simple estimator has large bias but small variance, and works well.

- A complex estimator, with many parameters, may have less bias, with large variance.

The bias/variance tradeoff







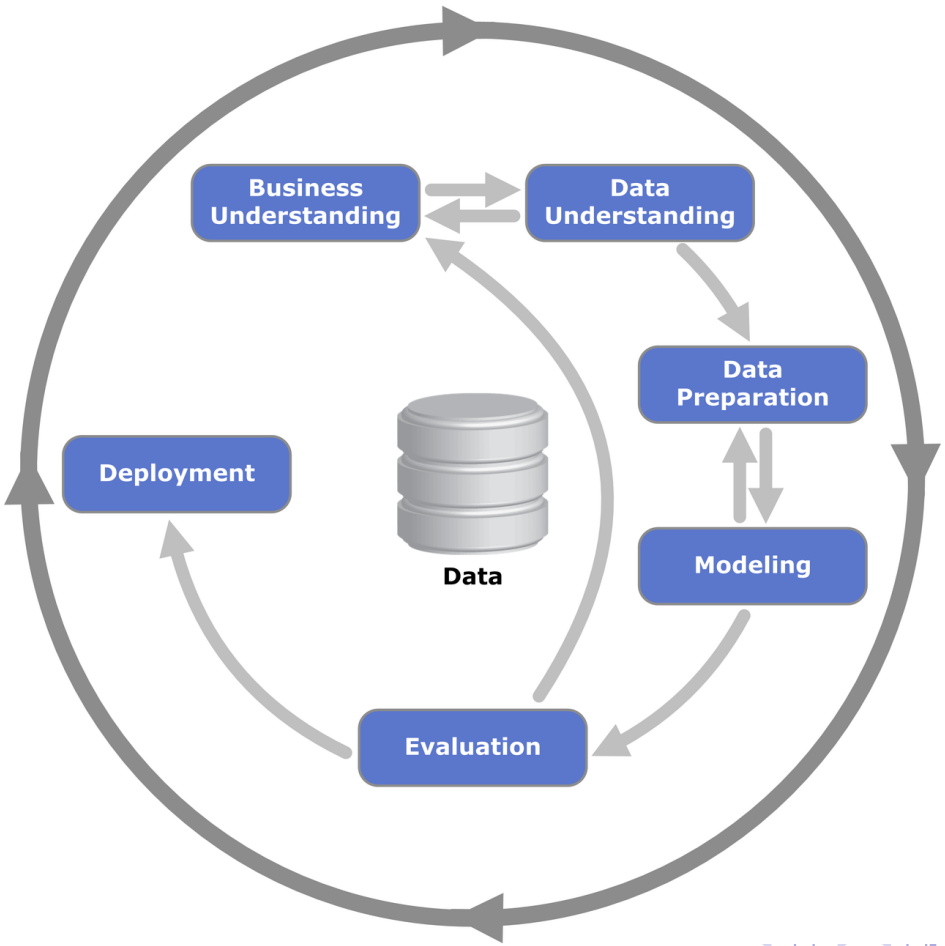
In practice

- We never know exactly which classifier/regressor to use.
- Simple ones work well: nearest neighbor, Naive Bayes, decision trees, linear regression...
- Many classifiers/regressors can be understood as instances of statistical models, others have ad hoc justifications.
- Other: neural networks (deep learning), support vector machines...

- Important to test, so as to select a reasonable one!

The CRISP-DM framework

- CRoss-Industry Standard Process for Data Mining: started in 1996 in EU; later adopted by many companies (in particular IBM).



A final note

Some of the figures in this presentation are taken from *An Introduction to Statistical Learning, with applications in R* (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani.