Lecture 2: Supervised vs. unsupervised learning, bias-variance tradeoff

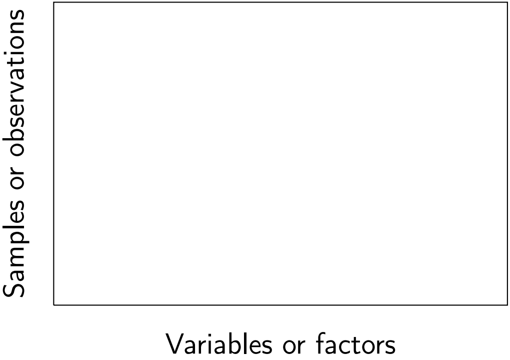
Reading: Chapter 2

STATS 202: Data mining and analysis

September 25, 2019

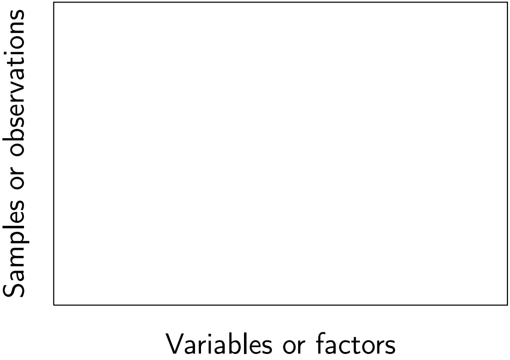
Supervised vs. unsupervised learning

In **unsupervised learning** we seek to understand the relationships between variables in a data matrix:



Supervised vs. unsupervised learning

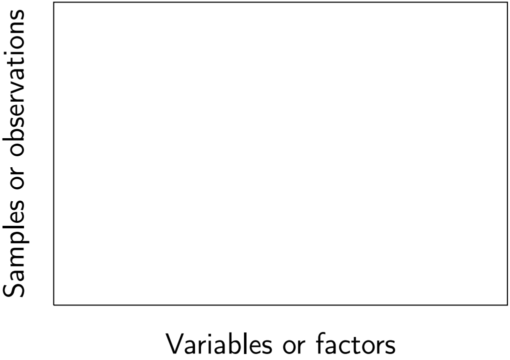
In **unsupervised learning** we seek to understand the relationships between variables in a data matrix:



Quantitative variables, eg. weight, height, number of children, ...

Supervised vs. unsupervised learning

In **unsupervised learning** we seek to understand the relationships between variables in a data matrix:



Qualitative variables, eg. college major, profession, gender, ...

Supervised vs. unsupervised learning

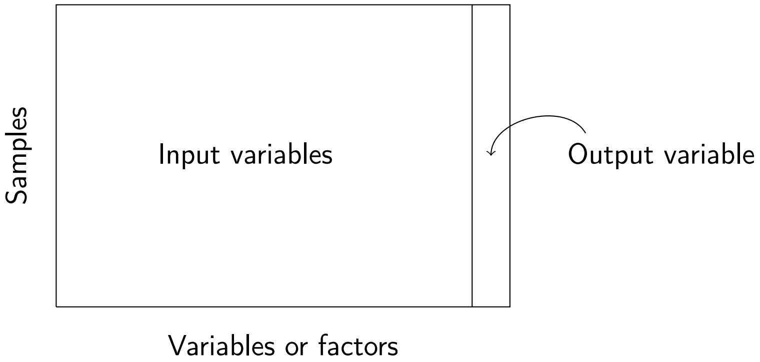
In **unsupervised learning** we seek to understand the relationships between variables in a data matrix:

Our goal is to:

- Find meaningful relationships between the variables. Correlation analysis.
- Find low-dimensional representations of the data which make it easy to visualize the variables. PCA, ICA, isomap, locally linear embeddings, etc.
- Find meaningful groupings of the data. Clustering.

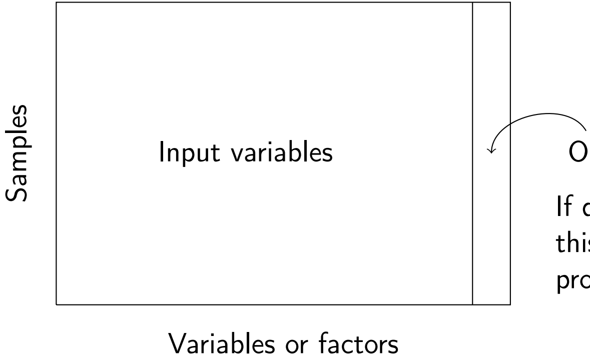
Supervised vs. unsupervised learning

In **supervised learning**, there are *input* variables, and *output* variables:



Supervised vs. unsupervised learning

In **supervised learning**, there are *input* variables, and *output* variables:

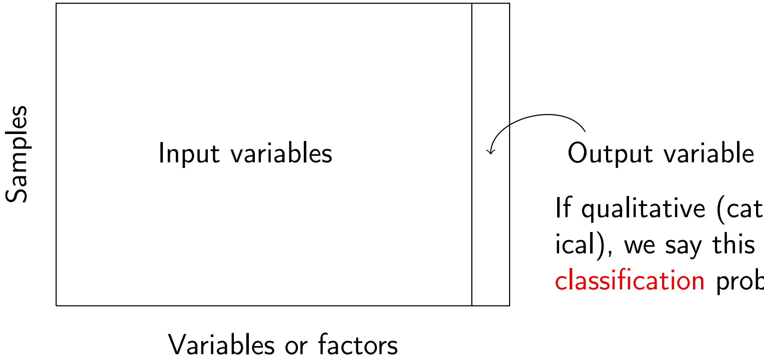


Output variable

If quantitative, we say this is a regression problem

Supervised vs. unsupervised learning

In **supervised learning**, there are *input* variables, and *output* variables:



If qualitative (categorical), we say this is a classification problem

Supervised vs. unsupervised learning

In **supervised learning**, there are *input* variables, and *output* variables:

If X is the vector of inputs for a particular sample. The output variable is modeled by:

$$Y = f(X) + \underbrace{\varepsilon}_{\text{Random error}}$$

The function f captures the systematic relationship between X and Y. f is fixed and unknown.

ϵ represents the unpredictable "noise" in the problem.

Our goal is to learn the function f, using a set of training samples.

Supervised vs. unsupervised learning

$$Y = f(X) + \underbrace{\varepsilon}_{\text{Random error}}$$

Motivations:

- **Prediction:** Useful when the input variable is readily available, but the output variable is not.

  Example: Predict stock prices next month using data from last year.

- **Inference:** A model for f can help us understand the structure of the data — which variables influence the output, and which don't? What is the relationship between each variable and the output, e.g. linear, non-linear?

  Example: What is the influence of genetic variations on the incidence of heart disease.

Parametric and nonparametric methods:

Most supervised learning methods fall into one of two classes:

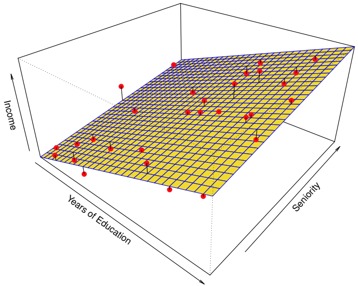
- **Parametric methods:** We assume that f takes a specific form. For example, a linear form:

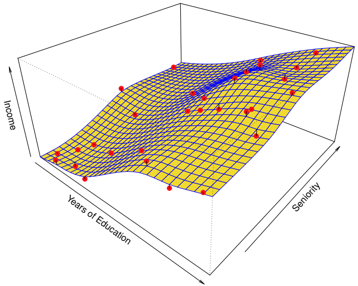
  $$f(X) = X_1\beta_1 + \cdots + X_p\beta_p$$

  with parameters $\beta_1, \ldots, \beta_p$. Using the training data, we try to *fit* the parameters.

- **Non-parametric methods:** We don't make any assumptions on the form of f, but we restrict how "wiggly" or "rough" the function can be.

Parametric vs. nonparametric prediction





Figures 2.4 and 2.5

Parametric methods have a limit of fit quality. Non-parametric methods keep improving as we add more data to fit.

Parametric methods are often simpler to interpret.

Prediction error

Training data: $(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)$
Predicted function: $\hat{f}$, which is based on these data.

What is a good choice for $\hat{f}$?
Our goal in supervised learning is to minimize the prediction error:
For a new datapoint (x_0, y_0) (not used in training) we want the squared error $(y_0 - \hat{f}(x_0))^2$ to be small.

Given many test data $\{(x'_i, y'_i); i = 1, \dots, m\}$ which were not used to fit the model, a common measure of quality of $\hat{f}$ is the test mean squared error (MSE):

$$MSE_{\mathsf{test}}(\hat{f}) = \frac{1}{m} \sum_{i=1}^{m} (y'_i - \hat{f}(x'_i))^2.$$

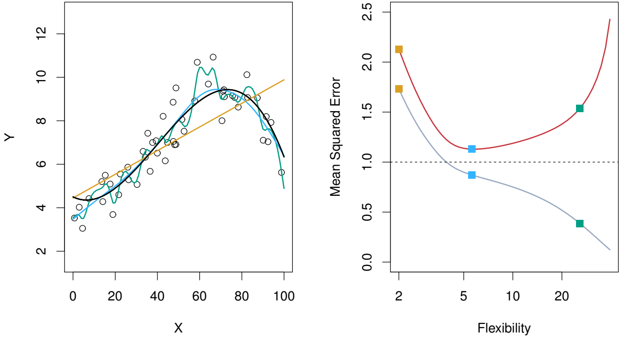
Prediction error

What if we don't have any test data? It is tempting to use instead the training MSE:

$$MSE_{\text{training}}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{f}(x_i))^2.$$

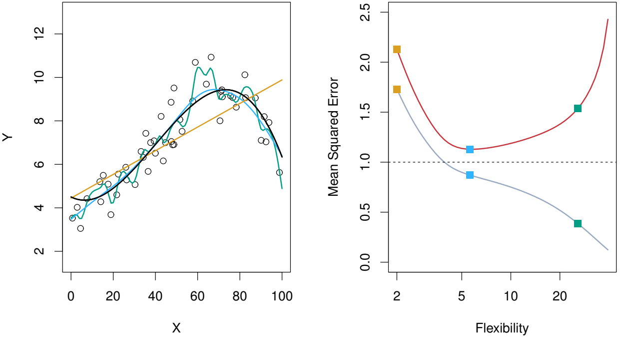
The main challenge of statistical learning is that *a low training MSE does not imply a low test MSE.*

Figure 2.9.



The circles are simulated data from the black curve. In this artificial example, we *know* what f is.

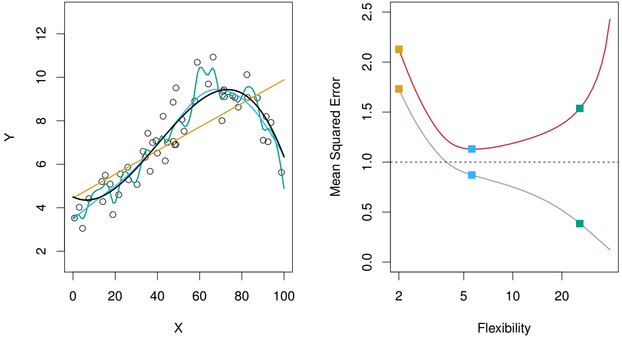
Figure 2.9.



Three estimates $\hat{f}$ are shown:

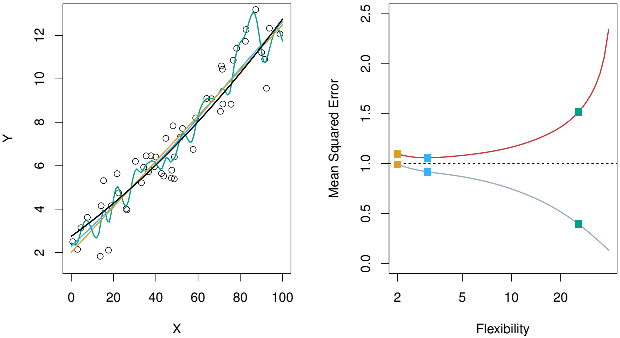
1. Linear regression.
2. Splines (very smooth).
3. Splines (quite rough).

Figure 2.9.



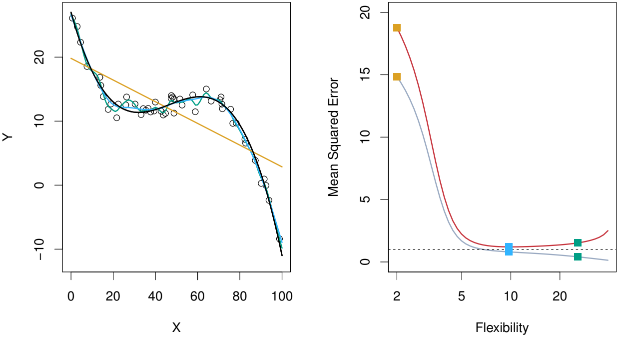
Red line: Test MSE.
Gray line: Training MSE.

Figure 2.10



The function f is now almost linear.

Figure 2.11



When the noise ε has small variance, the third method does well.

The bias variance decomposition

Let x_0 be a fixed test point, $y_0 = f(x_0) + \varepsilon_0$, and $\hat{f}$ be estimated from n training samples $(x_1, y_1) \ldots (x_n, y_n)$.

Let E denote the expectation over y_0 and the training data. Then, the expected test MSE at x_0 can be decomposed:

$$E(y_0 - \hat{f}(x_0))^2 = \mathsf{Var}(\hat{f}(x_0)) + [\mathsf{Bias}(\hat{f}(x_0))]^2 + \mathsf{Var}(\varepsilon_0).$$

The bias variance decomposition

Let x_0 be a fixed test point, $y_0 = f(x_0) + \varepsilon_0$, and $\hat{f}$ be estimated from n training samples $(x_1, y_1) \ldots (x_n, y_n)$.

Let E denote the expectation over y_0 and the training data. Then, the expected test MSE at x_0 can be decomposed:

$$E(y_0 - \hat{f}(x_0))^2 = \mathsf{Var}(\hat{f}(x_0)) + [\mathsf{Bias}(\hat{f}(x_0))]^2 + \mathsf{Var}(\varepsilon_0).$$

Irreducible error

The bias variance decomposition

Let x_0 be a fixed test point, $y_0 = f(x_0) + \varepsilon_0$, and $\hat{f}$ be estimated from n training samples $(x_1, y_1) \ldots (x_n, y_n)$.

Let E denote the expectation over y_0 and the training data. Then, the expected test MSE at x_0 can be decomposed:

$$E(y_0 - \hat{f}(x_0))^2 = \text{Var}(\hat{f}(x_0)) + [\text{Bias}(\hat{f}(x_0))]^2 + \text{Var}(\varepsilon_0).$$

The variance of the estimate of Y: $E[\hat{f}(x_0) - E(\hat{f}(x_0))]^2$

This measures how much the estimate of $\hat{f}$ at x_0 changes when we sample new training data.

The bias variance decomposition

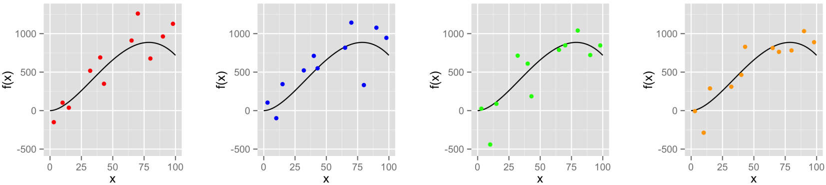
Let x_0 be a fixed test point, $y_0 = f(x_0) + \varepsilon_0$, and $\hat{f}$ be estimated from n training samples $(x_1, y_1) \ldots (x_n, y_n)$.

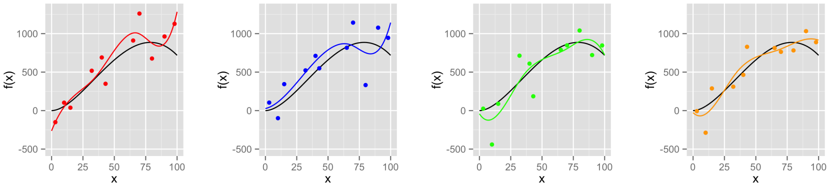
Let E denote the expectation over y_0 and the training data. Then, the expected test MSE at x_0 can be decomposed:

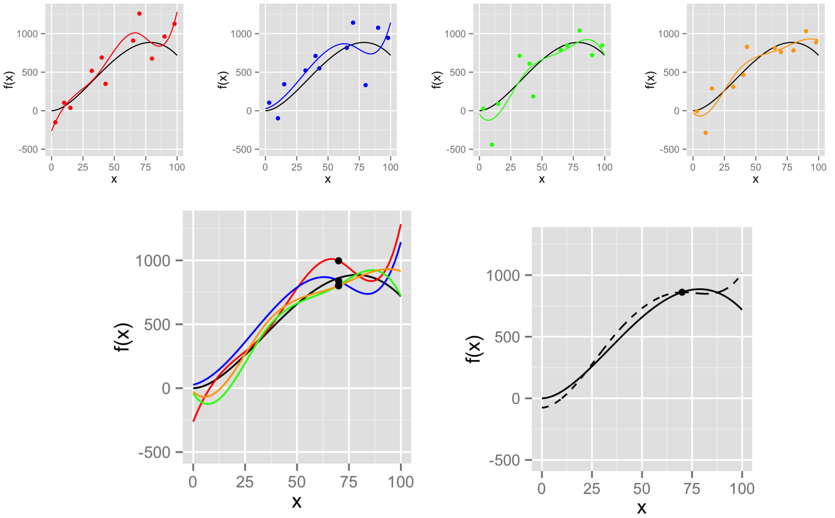
$$E(y_0 - \hat{f}(x_0))^2 = \mathsf{Var}(\hat{f}(x_0)) + [\mathsf{Bias}(\hat{f}(x_0))]^2 + \mathsf{Var}(\varepsilon_0).$$

The squared bias of the estimate of Y: $[E(\hat{f}(x_0)) - f(x_0)]^2$

This measures the deviation of the average prediction $\hat{f}(x_0)$ from the truth $f(x_0)$.


f(x)
1000
500
0
-500
0
25
50
75
100
x


1000
500
0
-500
f(x)
0
25
50
75
100
x


f(x)
1000
500
0
-500
0
25
50
75
100
x

Implications of bias variance decomposition

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

- The expected MSE is never smaller than the irreducible error.
- Both small bias and small variance are needed for small expected MSE.
- Easy to find zero variance procedure with high bias (predict a constant value) and very low bias procedure with high variance (choose $\hat{f}$ passing through every training point).
- In practice, best expected MSE achieved by incurring some bias to decrease variance and vice-versa: this is the **bias-variance trade-off**.
  Rule of thumb:
  More flexible methods $\Rightarrow$ Higher variance and Lower Bias.
- **Example:** Linear fit may not change substantially with training data but has high bias if f is very non-linear.

Squiggly f, high noise   Linear f, high noise   Squiggly f, low noise

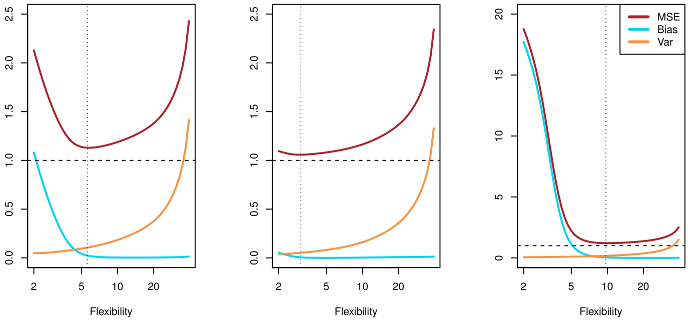


Figure 2.12

Classification problems

In a classification setting, the output takes values in a discrete set.

For example, if we are predicting the brand of a car based on a number of variables, the function f takes values in the set $\{$Ford, Toyota, Mercedes-Benz, $\ldots\}$.

We adopt some additional notation:

$$\begin{aligned} P(X,Y) &: \text{joint distribution of } (X,Y), \\ P(Y \mid X) &: \text{conditional distribution of } Y \text{ given } X, \\ \hat{y}_i &: \text{prediction for } x_i. \end{aligned}$$

Loss function for classification

There are many ways to measure the error of a classification prediction. One of the most common is the 0-1 loss:

$$\mathbf{1}(y_0 \neq \hat{y}_0)$$

As with squared error, we can compute average test predition error (called **test error rate** under 0-1 loss) using previously unseen test data $\{(x'_i, y'_i); i = 1, \ldots, m\}$:

$$\frac{1}{m}\sum_{i=1}^{m}\mathbf{1}(y'_i \neq \hat{y}'_i)$$

Similarly, we can compute the (usually optimistic) **training error rate**

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

Bayes classifier

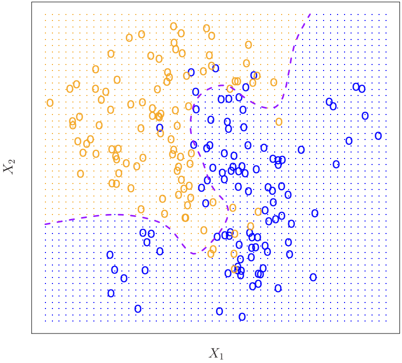


Figure 2.13

In practice, we never know the joint probability P. However, we can assume that it exists.

The Bayes classifier assigns:

$$\hat{y}_i = \text{argmax}_j \;\; P(Y = j \mid X = x_i)$$

It can be shown that this is the best classifier under the 0-1 loss.