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

Reading: Chapter 2

STATS 202: Data mining and analysis

Sergio Bacallado

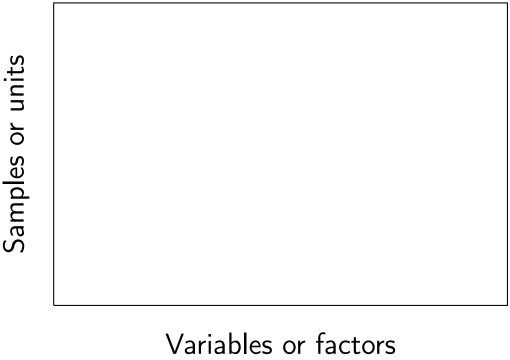
October 1, 2013

Announcements

- Piazza forum up. For the moment, you can use it to form teams.
- Homework 1, Kaggle invitations will go out this afternoon.
- If you still don't have access to the website, please email me your SUNet ID.

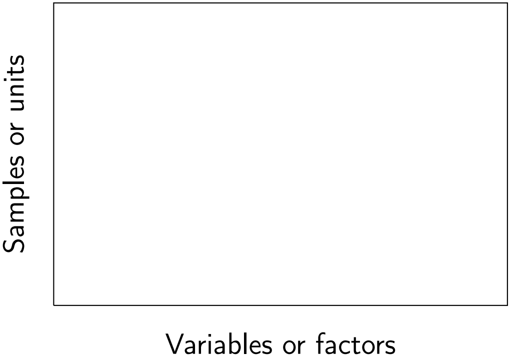
Supervised vs. unsupervised learning

In **unsupervised learning** we start with a data matrix:



Supervised vs. unsupervised learning

In **unsupervised learning** we start with a data matrix:

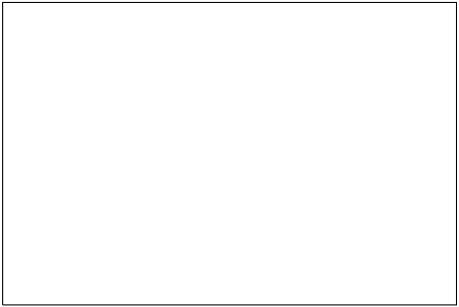


Quantitative, eg. weight, height, number of children, ...

Supervised vs. unsupervised learning

In **unsupervised learning** we start with a data matrix:

Samples or units



Variables or factors

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

Supervised vs. unsupervised learning

In **unsupervised learning** we start with a data matrix:

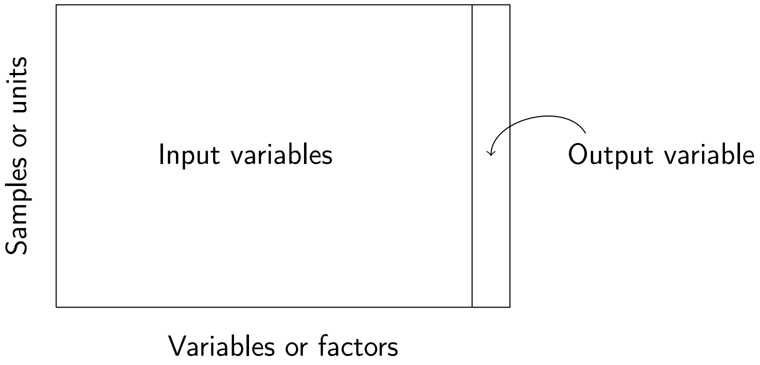
Our goal is to:

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

Unsupervised learning is also known in Statistics as **exploratory data analysis**.

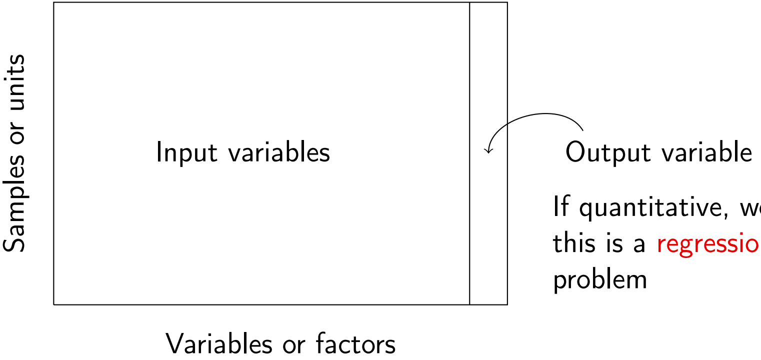
Supervised vs. unsupervised learning

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



Supervised vs. unsupervised learning

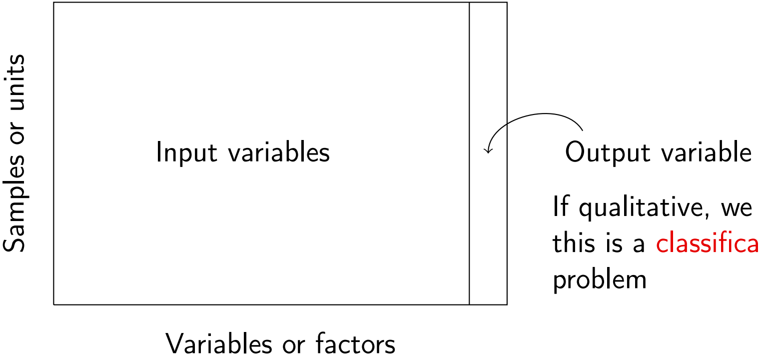
In **supervised learning**, there are *input* variables, and *output* variables:



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, 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:

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

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?

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.
- **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:

There are two kinds of supervised learning method:

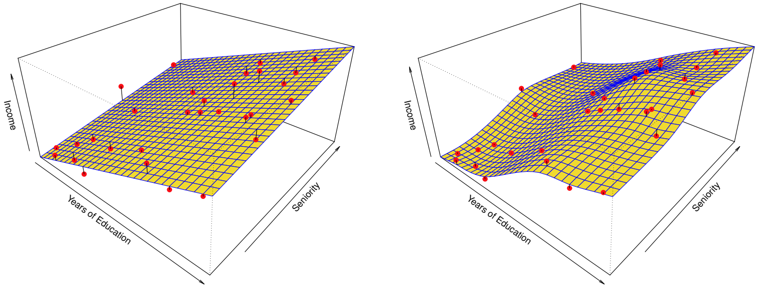
- **Parametric methods:** We assume that f takes a specific form. For example, a linear form:

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

  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

Our goal in supervised learning is to minimize the prediction error. For regression models, this is typically the *Mean Squared Error*:

$$MSE(\hat{f}) = E(y_0 - \hat{f}(x_0))^2.$$

Unfortunately, this quantity cannot be computed, because we don't know the joint distribution of (X, Y).

We can compute a sample average using the training data; this is known as the training MSE:

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

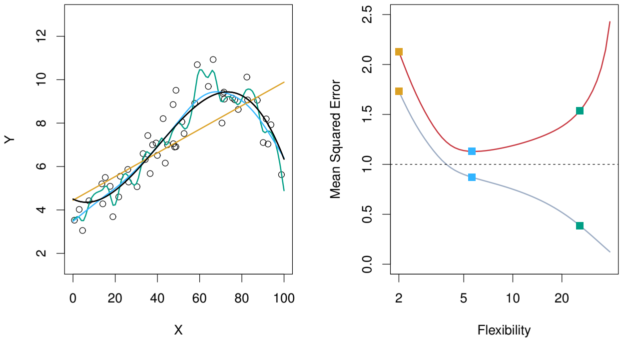
Prediction error

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

If there are test data $\{x_i', y_i'; i = 1, \ldots, m\}$ available which were not used to fit the model, a better measure of quality for $\hat{f}$ is the test MSE:

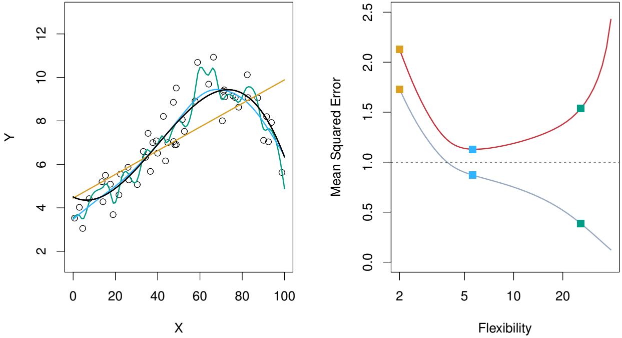
$$MSE_{\text{test}}(\hat{f}) = \frac{1}{m} \sum_{i=1}^{m} (y_i' - \hat{f}(x_i'))^2.$$

Figure 2.9.



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

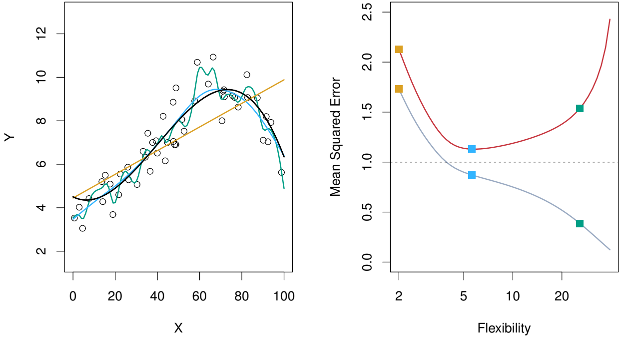
Figure 2.9.



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

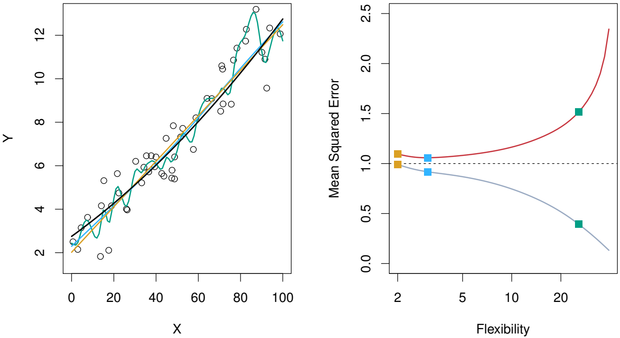
1. Linear regression.
2. Splines (very smooth).
3. Splines (quite rough).

Figure 2.9.



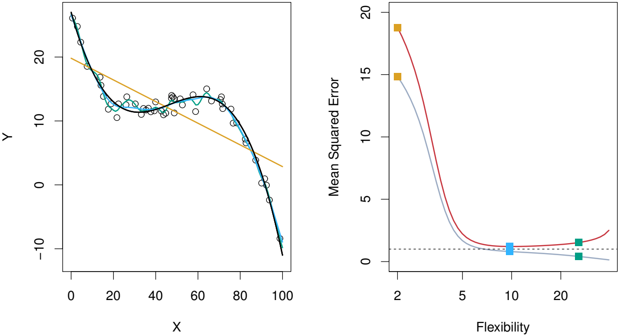
Red line: Test MSE.
Gray line: Training MSE.

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 fixed, $y_0 = f(x_0) + \varepsilon$, and $\hat{f}$ be estimated from n separate training samples. Let E denote the expectation over y_0 and the training samples. Then, the Mean Squared Error at x_0 can be decomposed:

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

The bias variance decomposition

Let x_0 be fixed, $y_0 = f(x_0) + \varepsilon$, and $\hat{f}$ be estimated from n separate training samples. Let E denote the expectation over y_0 and the training samples. Then, the Mean Squared Error at x_0 can be decomposed:

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

Irreducible error

The bias variance decomposition

Let x_0 be fixed, $y_0 = f(x_0) + \varepsilon$, and $\hat{f}$ be estimated from n separate training samples. Let E denote the expectation over y_0 and the training samples. Then, the Mean Squared Error at x_0 can be decomposed:

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

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 fixed, $y_0 = f(x_0) + \varepsilon$, and $\hat{f}$ be estimated from n separate training samples. Let E denote the expectation over y_0 and the training samples. Then, the Mean Squared Error at x_0 can be decomposed:

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

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

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

The bias variance decomposition

Let x_0 be fixed, $y_0 = f(x_0) + \varepsilon$, and $\hat{f}$ be estimated from n separate training samples. Let E denote the expectation over y_0 and the training samples. Then, the Mean Squared Error at x_0 can be decomposed:

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

Both variance and squared bias are always positive.

Higher variance $\iff$ More flexibility $\iff$ Lower bias.

We will aim to minimize both sources of error simultaneously.

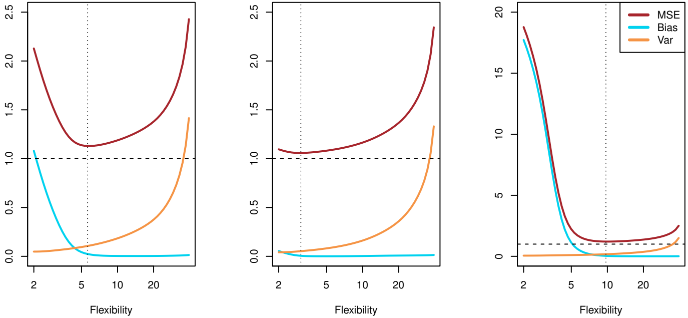

0.0
0.5
1.0
1.5
2.0
2.5
2
5
10
20
Flexibility
0.0
0.5
1.0
1.5
2.0
2.5
2
5
10
20
Flexibility
0
5
10
15
20
2
5
10
20
Flexibility
MSE
Bias
Var

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 $\{\text{Ford, Toyota, Mercedes-Benz}, \ldots\}$.

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$, \ldots\}$.

The model:

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

becomes insufficient, as X is not necessarily real-valued.

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$, \ldots\}$.

The model:

$$\cancel{Y = f(X) + \varepsilon}$$

becomes insufficient, as X is not necessarily real-valued.

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 will use slightly different notation:

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

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:

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

Like the MSE, this quantity can be estimated from training and test data by taking a sample average:

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

Bayes classifier

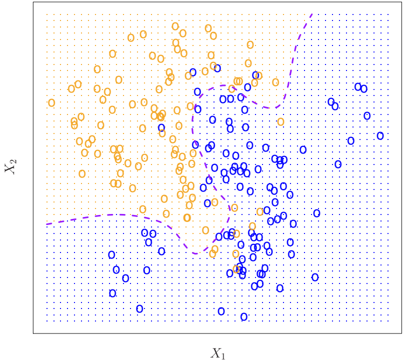


Figure 2.13

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.