COM 525000 – Statistical Learning

Lectures 1 & 2 – Introduction to Statistical Learning

Y.-W. Peter Hong

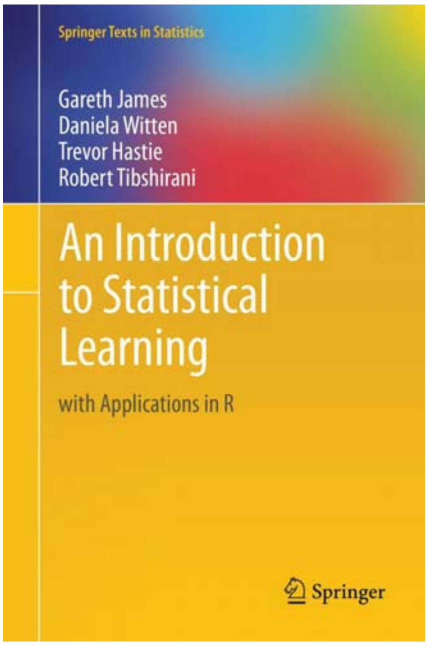
Course Information I

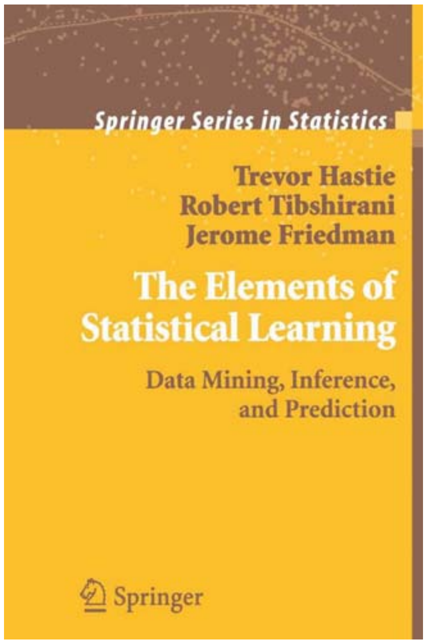
- **Goal and Overview:**
  - Introductory course on statistical learning, where we introduce basic tools for data analysis and modelling.
    - Linear and Nonlinear Regression
    - Classification (e.g., Logistic Regression, Linear Discriminant Analysis, k-Nearest Neighbors etc)
    - Model selection and regularization
    - Tree-based methods
    - Support Vector Machines
    - Unsupervised Learning (Principal Component Analysis, Clustering)
- **Required background:**
  - Probability; Linear Algebra. [Random Process and/or Statistics also helps]

- **Textbooks:**

  G. James, D. Witten, T. Hastie and R. Tibshirani, *An Introduction to Statistical Learning with Applications in R*, Springer, 2013.

  (Ref.) T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning Data Mining, Inference, and Prediction*, 2nd ed., Springer, 2009.





Course Information II

- **Instructor:** Y.-W. Peter Hong (洪樂文)
- **Email:** ywhong@ee.nthu.edu.tw
- **Office Hour:** Wednesday 13:30~15:30 (Delta 815)
- **TA：** Gin-Hao Liu
- **TA Email:** glrt2793@gmail.com
- **TA Office:** EECS 613 **Office Hour:** Monday 10:00~11:00
- **Course Website:** http://lms.nthu.edu.tw/
- **Grades:** Homework 30%; Midterm 35%; Final 35%

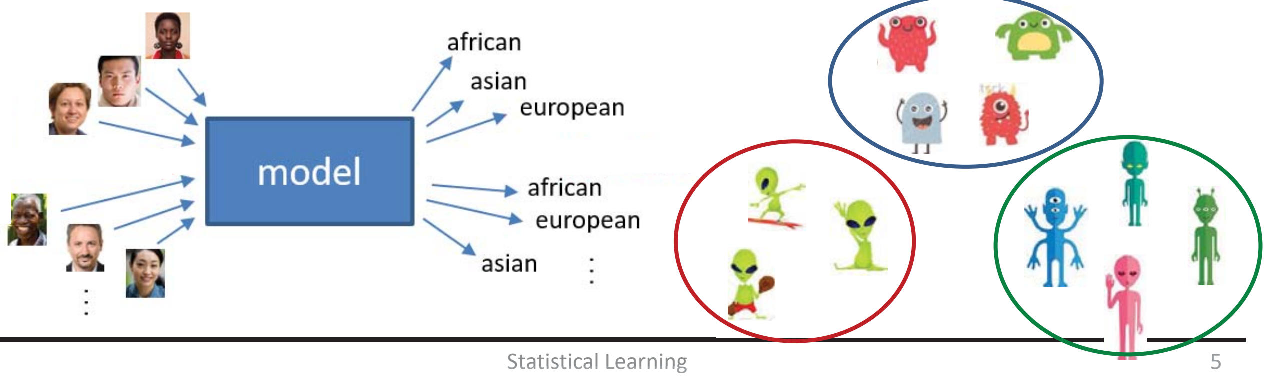
(Academic integrity is strictly enforced!! Any form of cheating in HW or exams will result in failure of the course. No Warnings!!)

Important Dates:

- **Midterm Exam 11/7 Thursday 9:00-12:00 in class**
- **Final Exam 1/9 Thursday 9:00-12:00 in class**

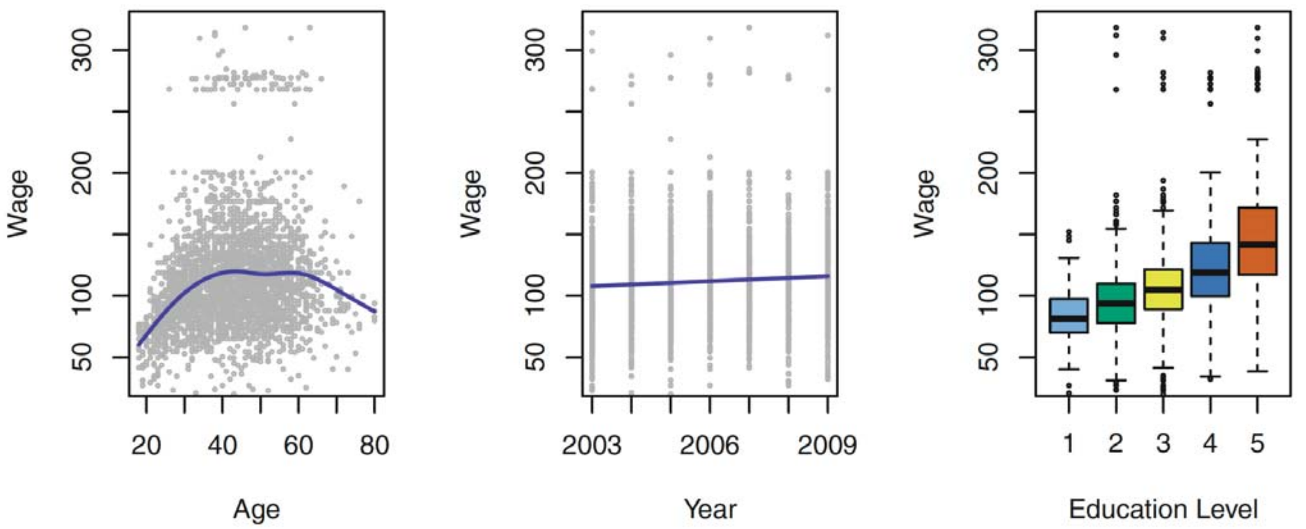
Introduction

- **Statistical learning** refers to the set of tools used for *understanding data*. (➔supervised and unsupervised)
- Supervised learning involves building a statistical model for predicting outputs based on inputs using a known set of inputs and corresponding outputs.
- Unsupervised learning learns relationships and structure from the data (without known outputs).



Example: Wage Data

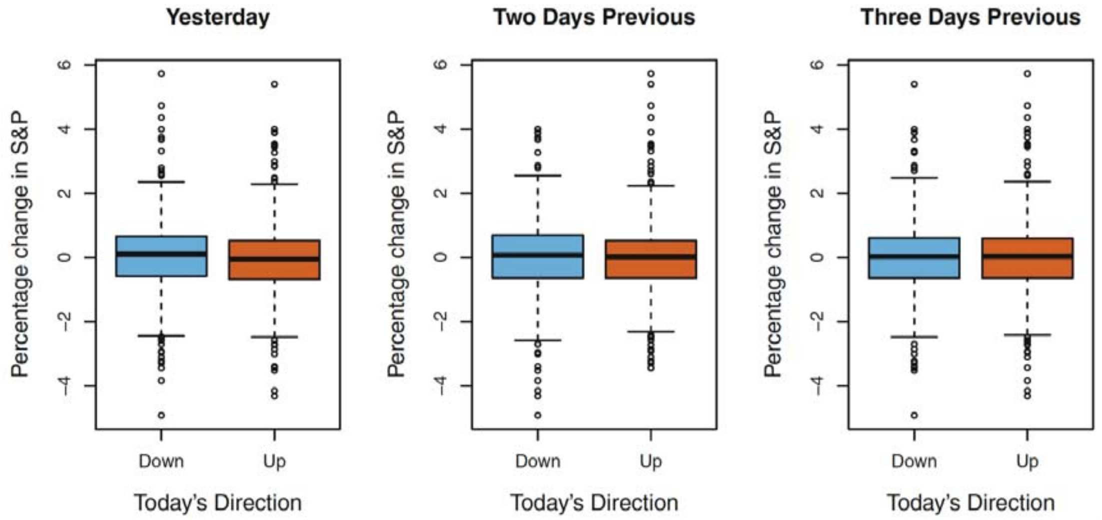
- Wages for a group of males from the Atlantic region.
- **Goal:** To understand the impact of age, education, and calendar year on a person’s wage.



- Prediction on a quantitative output. (➔ regression)

Example: Stock Market Data

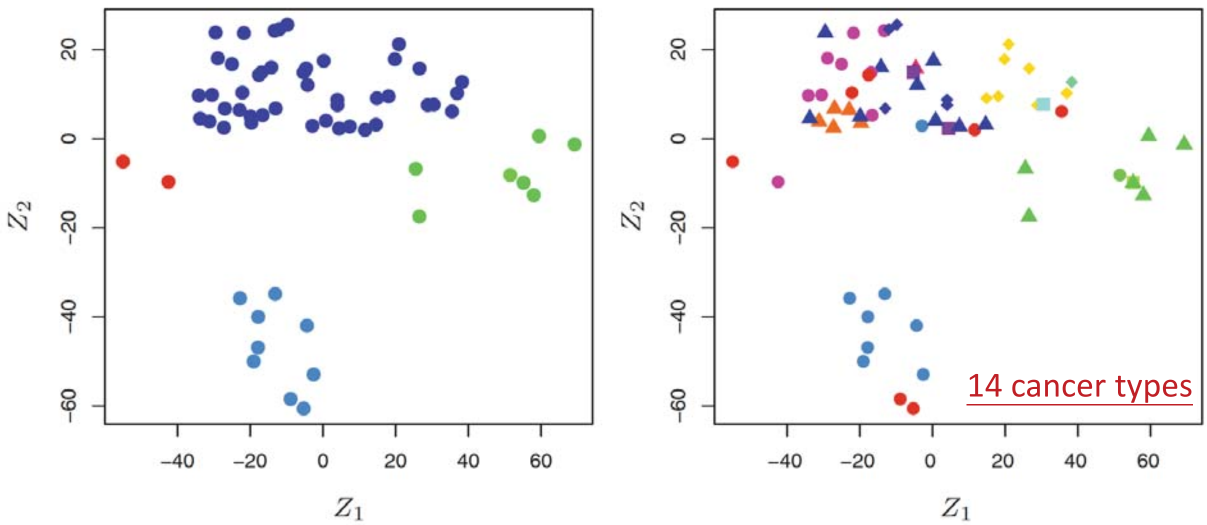
- Daily movements in the S&P stock index (2001-2005).
- **Goal:** Predict if the index will go *up* or *down* on a given day using percentage changes of past few days.



- Prediction on qualitative output. (➔ classification)

Example: Gene Expression Data

- 6830 gene expression measurements from 64 cancer cell lines.
- **Goal:** Explore the relation between cancer cell lines.
- Z_1 and Z_2 are two principal components.



Statistical Learning vs Machine Learning

- Both statistical and machine learning focus on learning from data. The difference is subtle.

Statistical Learning:
formalization of relationships between variables in the form of mathematical equations.

- Mathematics (statistics).
- Makes assumptions on the observed data.
- Focus on understanding relation between variables (inference)
- More theoretical foundations.

Machine Learning:
an algorithm that can learn from data without relying on rule-based programming.

- Computer science.
- No (or less) assumptions on the observed data.
- Focus on prediction accuracy (regardless of understanding)
- Better algorithmic designs.

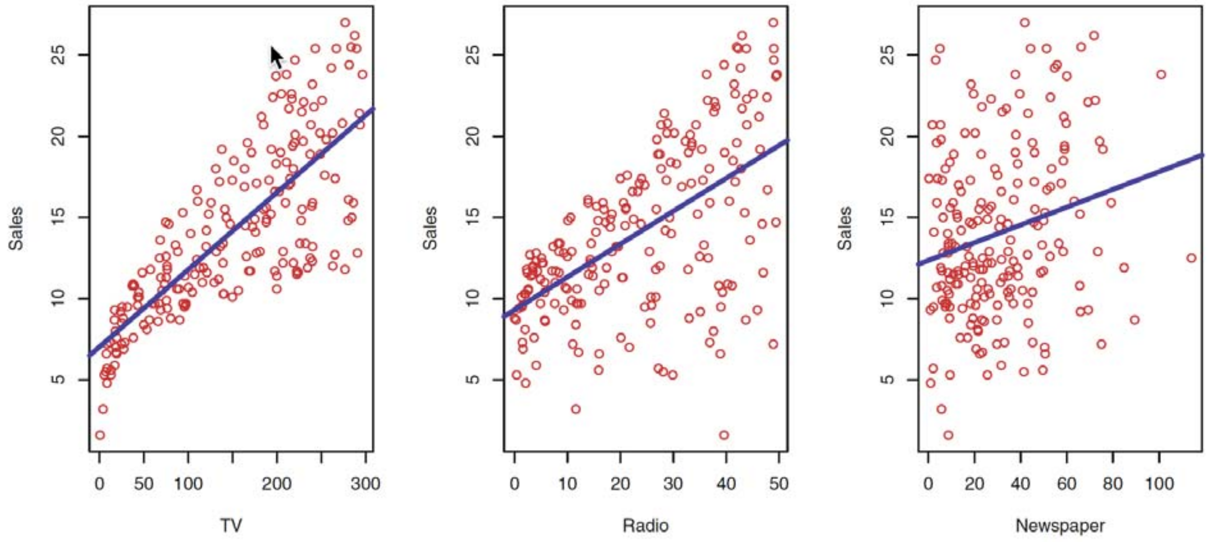
Notations

- Let n be the number of data points, and p be the number of variables.
- x_{ij}: the value of the jth variable for the ith observation, where $j = 1, \ldots, p$ and $i = 1, \ldots, n$.
- $x_i = [x_{i1}, \ldots, x_{ip}]^T$: the ith observation vector.
- $\mathbf{x}_j = [x_{1j}, \ldots, x_{nj}]^T$: length-$n$ vector of the jth variable values over all data points.
- $\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p]$
- Similarly, $\mathbf{y} = [y_1, \ldots, y_n]^T$.

Motivating Example (1/3)

Example: (Advertising Data Set)

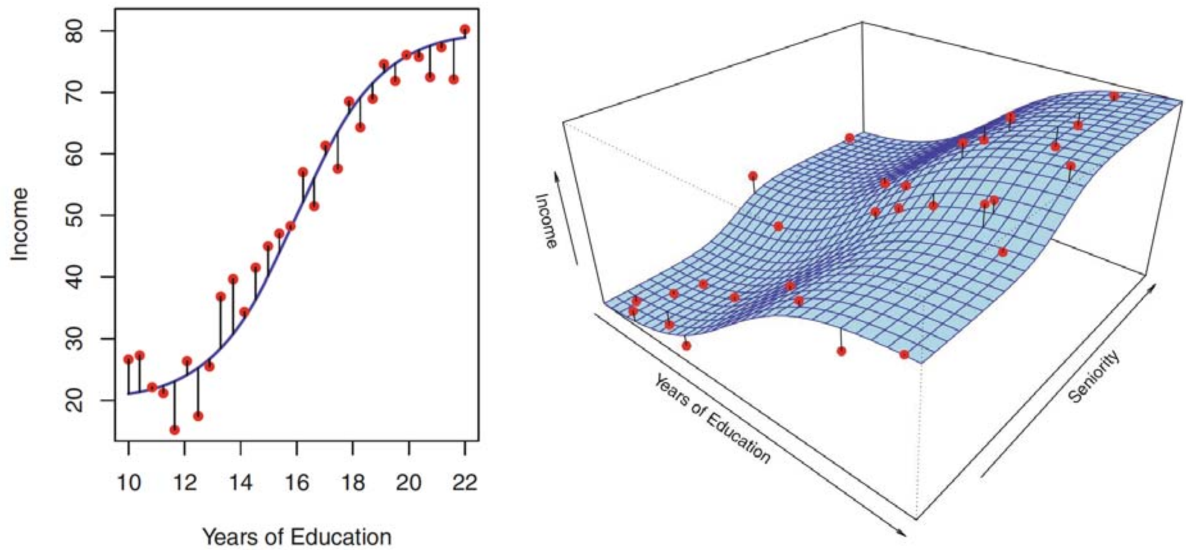
- Sales in 200 different markets versus the advertising budgets for TV, radio, and newspaper.



Motivating Example (2/3)

Example: (Income Data Set)

- Income of 30 individuals versus the number of years of education and/or seniority.



Motivating Example (3/3)

- In the advertising example:
  - What will the sales be for a certain budget assignment in TV, radio, and newspaper? ➔ Prediction problem.
  - Which media causes the most increase in sales and by how much? ➔ Inference problem.
- In the income example:
  - Given the person's age and education level, what is the person's income? ➔ Prediction problem.
  - How does the number of years of education affect the income? ➔ Inference problem.

More formally…

- $X_1, X_2, \ldots, X_p$: *input variables*, *predictors*, *independent variables*, *features* (or simply *variables*)
- Y : *output variable*, *response*, or *dependent variable*
- We assume that there is some function f such that

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

where $X = (X_1, \ldots, X_p)$ and ϵ is a random error term.

Statistical learning in essence refers to the set of approaches that can be used for estimating f.

Question: How do we estimate f?

➔ E.g., Linear least squares and k-nearest neighbors.

Linear Least Squares for Regression (1/2)

- **Linear Model:** Given $X=(X_1,\ldots,X_p)^T$, we want to estimate output Y via the *linear model*

$$\hat{Y} = \hat{f}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p.$$

- Alternatively, we can define $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p)^T$ and $X=(1, X_1,\ldots,X_p)^T$ such that $\hat{Y} = \hat{f}(X) = X^T\hat{\beta}$.
- **Least Squares Fit:** Fit the model to the set of training data using the least squares method where $\hat{\beta}$ is chosen to minimize the *residual sum of squares*

$$\mathrm{RSS}(\hat{\beta}) = \sum_{i=1}^{n}(y_i - x_i^T\hat{\beta})^2$$

where $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ is the training data.

Linear Least Squares for Regression (2/2)

- The residual sum of squares can be written as

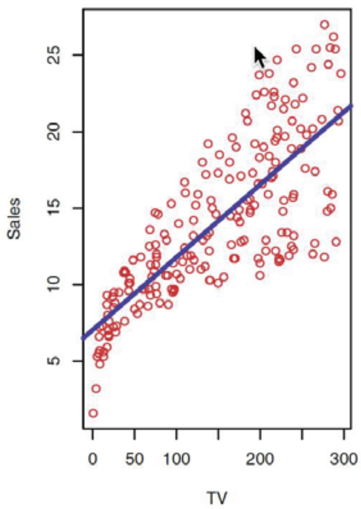
$$\mathrm{RSS}(\hat{\beta}) = (\mathbf{y} - \mathbf{X}\hat{\beta})^T(\mathbf{y} - \mathbf{X}\hat{\beta})$$

where $\mathbf{X}$ is the $n \times p$ matrix with each row an input vector and $\mathbf{y}$ is the n-vector of training outputs.
- Differentiating w.r.t. $\hat{\beta}$ and setting it to zero yields

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\hat{\beta}) = \mathbf{0}.$$

- The solution is $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$.
- For new input x_0, the predicted output is

$$\hat{y}_0 = \hat{f}(x_0) = x_0^T\hat{\beta} = x_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$



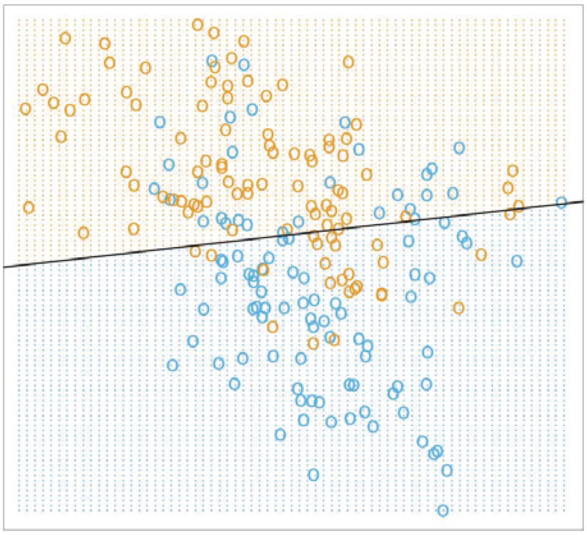
Linear Least Squares for Classification

- Consider a *classification problem* where the output class variable is $G \in \{\text{BLUE}, \text{ORANGE}\}$ (categorical).
- Let us encode response $Y=0$ if G is BLUE, and $Y=1$ if G is ORANGE.
- Step 1: For new input x_0, estimate response y_0 using

$$\hat{y}_0 = \hat{f}(x_0) = x_0^T \hat{\beta}.$$

- Step 2: Determine the class by taking

$$\hat{G}(x_0) = \begin{cases} \text{ORANGE}, & \hat{y}_0 > 0.5, \\ \text{BLUE}, & \hat{y}_0 \leq 0.5. \end{cases}$$



KNN for Regression and Classification

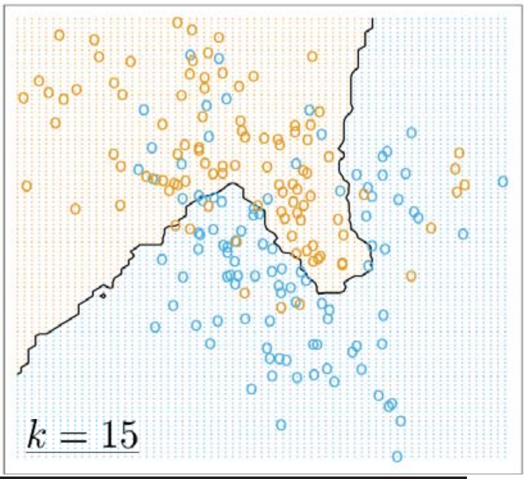
- For regression, the k-nearest neighbor (kNN) prediction for new input x_0 is defined as

$$\hat{y}_0 = \hat{f}(x_0) = \frac{1}{k} \sum_{i:x_i \in \mathcal{N}_k(x_0)} y_i$$

where $\mathcal{N}_k(x_0)$ represents the set of k nearest $x_i's$ in in the vicinity of x_0.

- For classification, the predicted class for input x_0 is given by

$$\hat{G}(x_0) = \begin{cases} \text{ORANGE}, & \frac{1}{k} \sum_{i:x_i \in \mathcal{N}_k(x_0)} y_i > 0.5 \\ \text{BLUE}, & \frac{1}{k} \sum_{i:x_i \in \mathcal{N}_k(x_0)} y_i \leq 0.5 \end{cases}$$



Statistical Estimation Theory

- Let $X \in \mathbb{R}^p$ be the random input vector, and let $Y \in \mathbb{R}$ be the random output, with joint PDF $p(X, Y)$.
- **Goal:** Find function f that yields the *best* prediction on Y given X.
- Define the *loss function* as $L(Y, f(X))$ (e.g., the squared error loss $L(Y, f(X)) = (Y - f(X))^2$).
- In this case, the expected prediction error (EPE) is

$$\mathrm{EPE}(f) = E\left[(Y - f(X))^2\right] =$$

- It suffices to minimize EPE pointwise, for every x:

$$f^*(x) = \arg\min_c E_{Y|X}\left[(Y - c)^2 | X = x\right]$$

kNN Insights from Estimation Theory

- The solution that minimizes the EPE is

$$f^*(x) = E[Y|X = x]$$

(i.e., the regression function).

- In KNN, we approximate $f^*(x)$ by

$$\hat{f}_{\mathrm{kNN}}(x) = \frac{1}{k} \sum_{i: x_i \in \mathcal{N}_k(x)} y_i \left(\triangleq \mathrm{Ave}\big(y_i | x_i \in \mathcal{N}_k(x)\big) \right).$$

  - Expectation approximated by sample average.
  - Conditioning at x is relaxed to conditioning on a region close to x.

➔ The average is more stable for large k, but the training sample size n must be even larger to make the k points close.

➔ $\hat{f}_{\mathrm{kNN}}(x) \to E[Y|X = x]$ when $n, k \to \infty$ such that $k/n \to 0$.

LS Insights from Estimation Theory

- In linear least squares regression, we take the model-based approach where we assume that $f(x) \approx x^T \beta$.
- In this case, the EPE is $\text{EPE}(f) = E\big[(Y - X^T\beta)^2\big]$ and is minimized by taking

$$\beta = E[XX^T]^{-1}E[XY].$$

- Hence, the *linear regression function* is

$$f^*(x) = x^T E[XX^T]^{-1}E[XY].$$

- The least squares solution yields instead

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n} x_i y_i\right).$$

  – Expectation replaced by averages over the training data.

Statistical Decision Theory

- In classification, the output $G \in \mathcal{G}$ is categorical (e.g., $\mathcal{G} \triangleq \{\text{happy}, \text{sad}, \text{angry}\}$) or, encode as $Y \in \mathcal{Y} \triangleq \{0, \dots, J-1\}$.
- **Goal:** Find the classifier that minimizes $E[L(Y, f(X)]$.
- This yields the *Bayes classifier*

$$\begin{aligned} f^*(x) &= \arg\min_{\hat{y}\in\mathcal{Y}} E[L(Y, \hat{y})|X = x] \\ &= \arg\min_{\hat{y}\in\mathcal{Y}} \sum_{y=0}^{J-1} L(y, \hat{y}) \Pr(Y = y|X = x). \end{aligned}$$

- For *zero-one loss function* $L(Y, f(X)) = \begin{cases} 1, & Y \neq f(X) \\ 0, & Y = f(X) \end{cases}$ we have,

$$f^*(x) = \arg\min_{\hat{y}\in\mathcal{Y}} [1 - \Pr(Y = \hat{y}|X = x)] = \arg\max_{\hat{y}\in\mathcal{Y}} \Pr(Y = \hat{y}|X = x).$$

Insights from Decision Theory

- In kNN, we approximate $f^*(x)$ by

$$\hat{f}_{\mathrm{kNN}}(x) = \arg\max_{\hat{y}\in\mathcal{Y}} \frac{1}{k} \sum_{i:x_i\in\mathcal{N}_k(x)} I(y_i = \hat{y}).$$

- In least squares (for the binary case $\mathcal{Y} \triangleq \{0,1\}$), we first compute the estimate

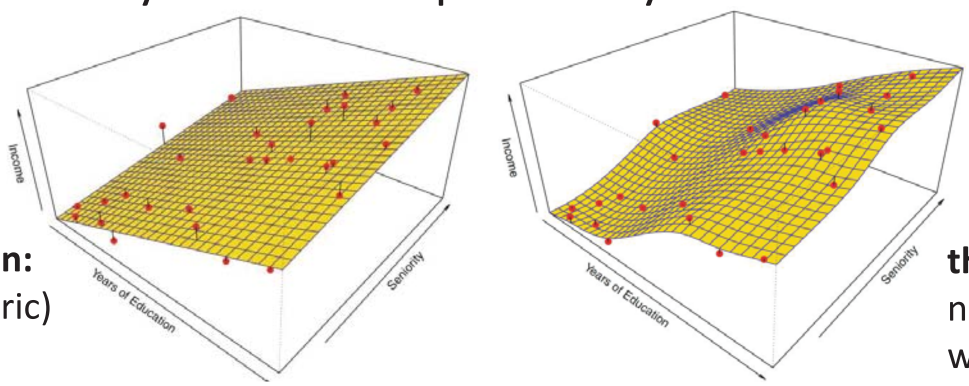
$$\Pr(Y=1|X=x)=1-\Pr(Y=0|X=x) \approx x^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Then, approximate the classifier as

$$\hat{f}_{\mathrm{LS}}(x) = \begin{cases} 1, & x^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} > 0.5 \\ 0, & x^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \le 0.5 \end{cases}.$$

Selecting Learning Methods

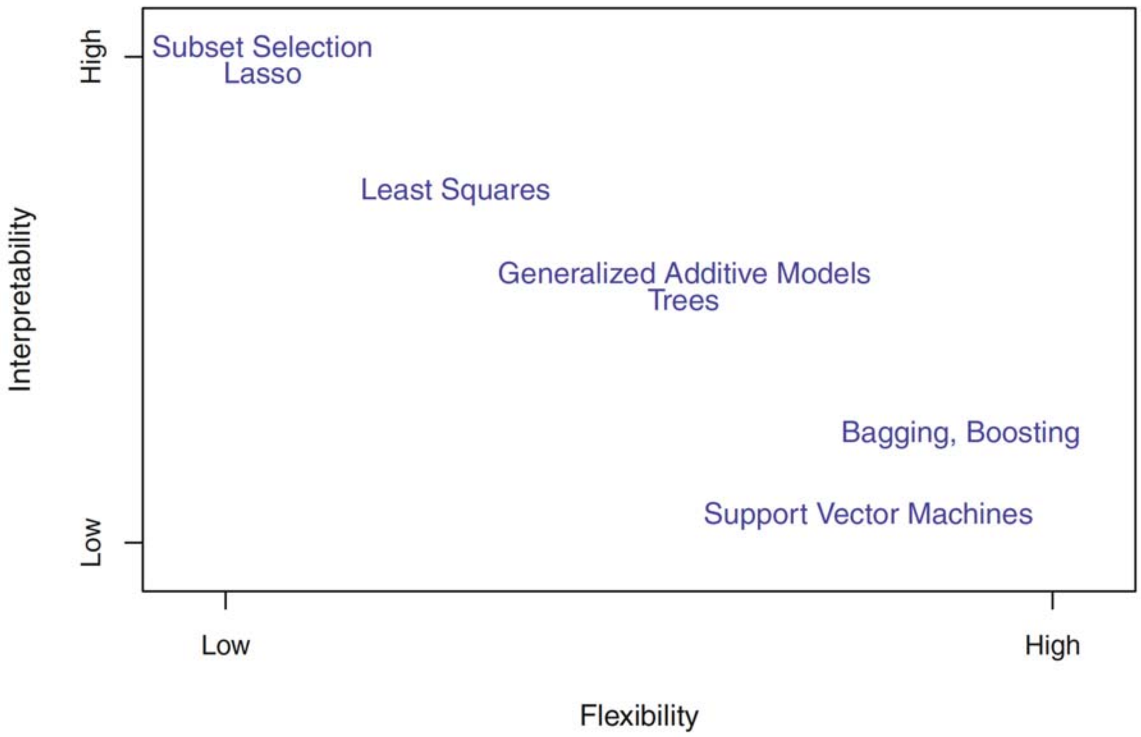
- Different learning methods have different underlying assumptions.
  - LLS assumes that $f(x)$ can be approx. by a linear function.
  - kNN assumes that $f(x)$ can be approx. by a locally constant function.
- Flexibility versus interpretability



linear regression: (parametric)
$\texttt{income} \approx \beta_0 + \beta_1 \times \texttt{education} + \beta_2 \times \texttt{seniority}$

thin plate spline: non-parametric with smoothness constraints

Flexibility vs Interpretability



Assessing Model Accuracy

- No one statistical learning method dominates all others over all possible data sets. (It also depends on the parameters chosen for each method.)

  ➔ Model assessment and selection is important.
- Performance measure for model assessment:
  - For regression, the most common performance measure is the mean squared error (MSE)

$$\text{MSE} = E\big[E[(Y - \hat{f}(X; \mathcal{D}))^2 | \mathcal{D}]\big]$$

where $\hat{f}(\cdot; \mathcal{D})$ is the regression function fitted to the training data.

Training and Test Data Sets

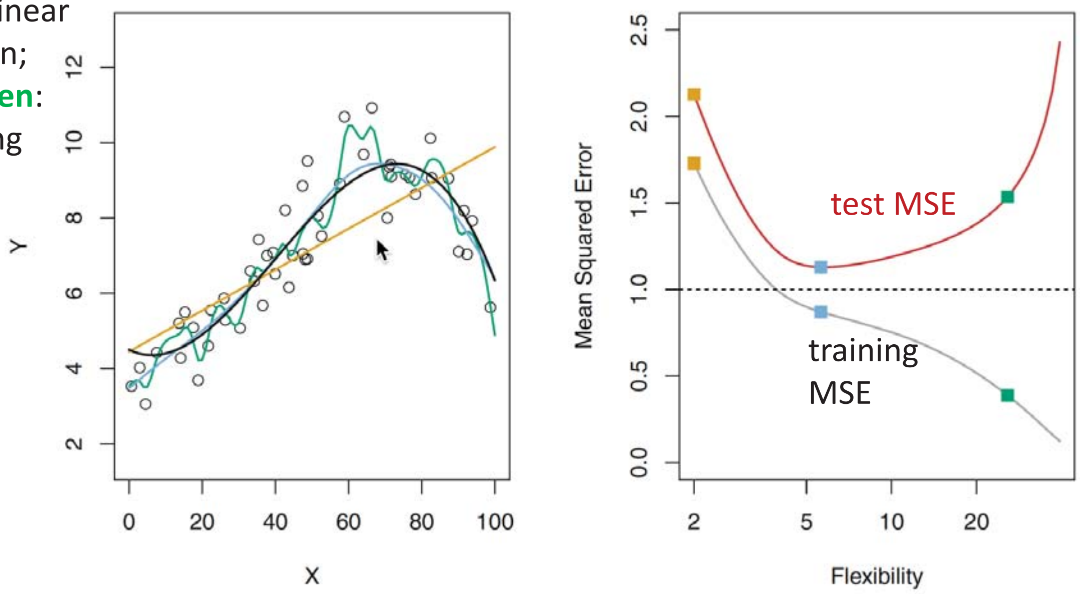
- In practice, we don't know the underlying distribution and, thus, $E[(Y - \hat{f}(X;\mathcal{D}))^2|\mathcal{D}]$ can only be computed as
  - Training MSE: $\frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} (y - \hat{f}(x;\mathcal{D}_{\text{train}}))^2$
  - Test MSE: $\frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(x,y)\in\mathcal{D}_{\text{test}}} (y - \hat{f}(x;\mathcal{D}_{\text{train}}))^2$
- The model fitting is often performed by minimizing the training MSE to yield the estimate $\hat{f}(\cdot;\mathcal{D}_{\text{train}})$, but the model assessment is done using the test MSE.

Training data set $\mathcal{D}_{\text{train}}$	Test data set $\mathcal{D}_{\text{test}}$

Total available data is split into training and test data sets.

Training MSE vs Test MSE (1/3)

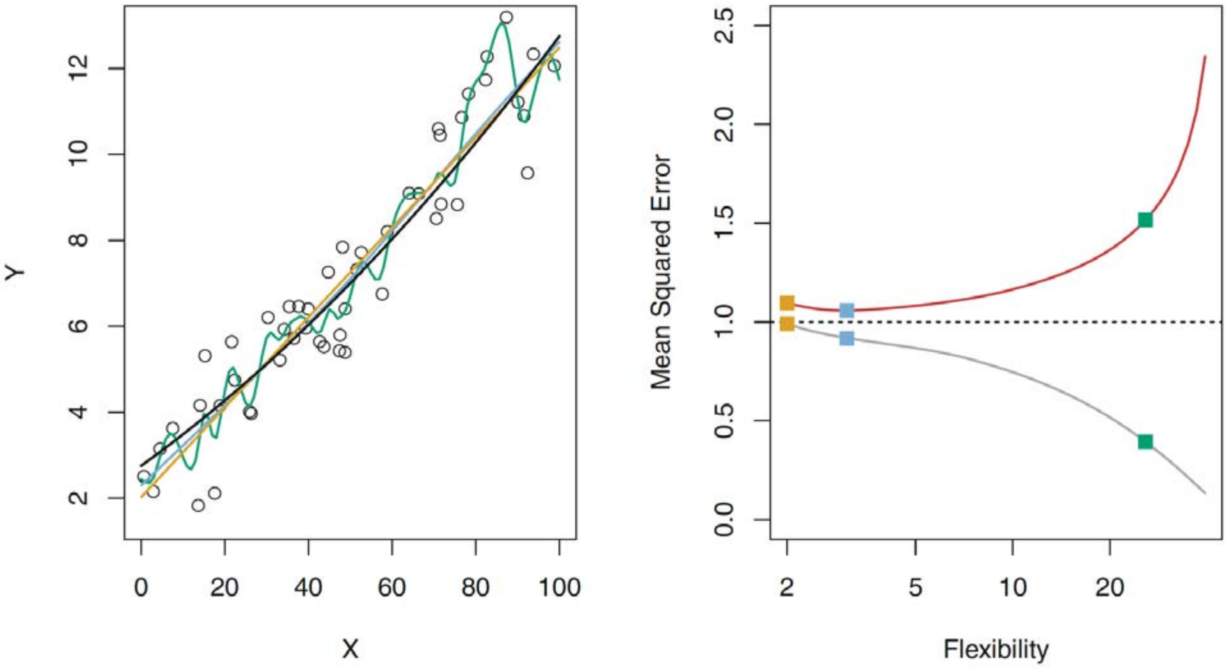
black: true f; **orange**: linear regression; **blue**, **green**: smoothing splines;



- Training MSE is always less than the test MSE, and decreases with flexibility.
- Too much flexibility may result in **overfitting**, causing the test MSE to increase.

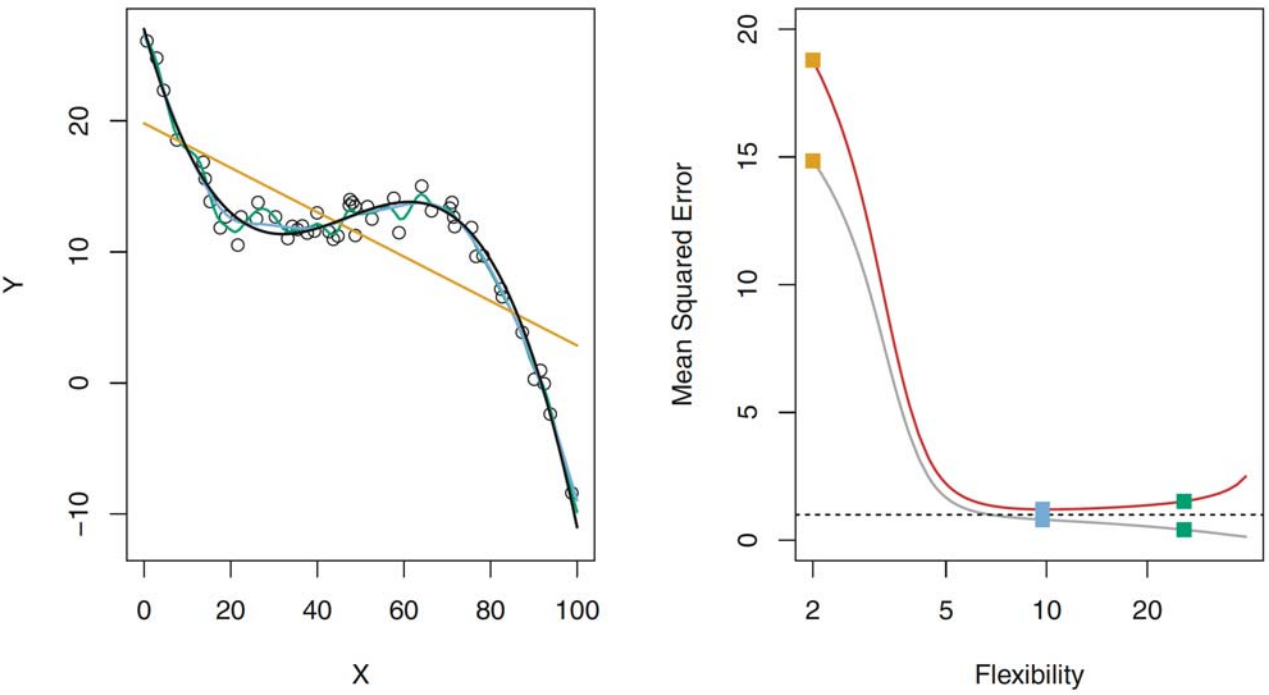
Training MSE vs Test MSE (2/3)

- For f that is approximately linear:



Training MSE vs Test MSE (2/3)

- For f that is highly nonlinear:



Bias-Variance Tradeoff (1/2)

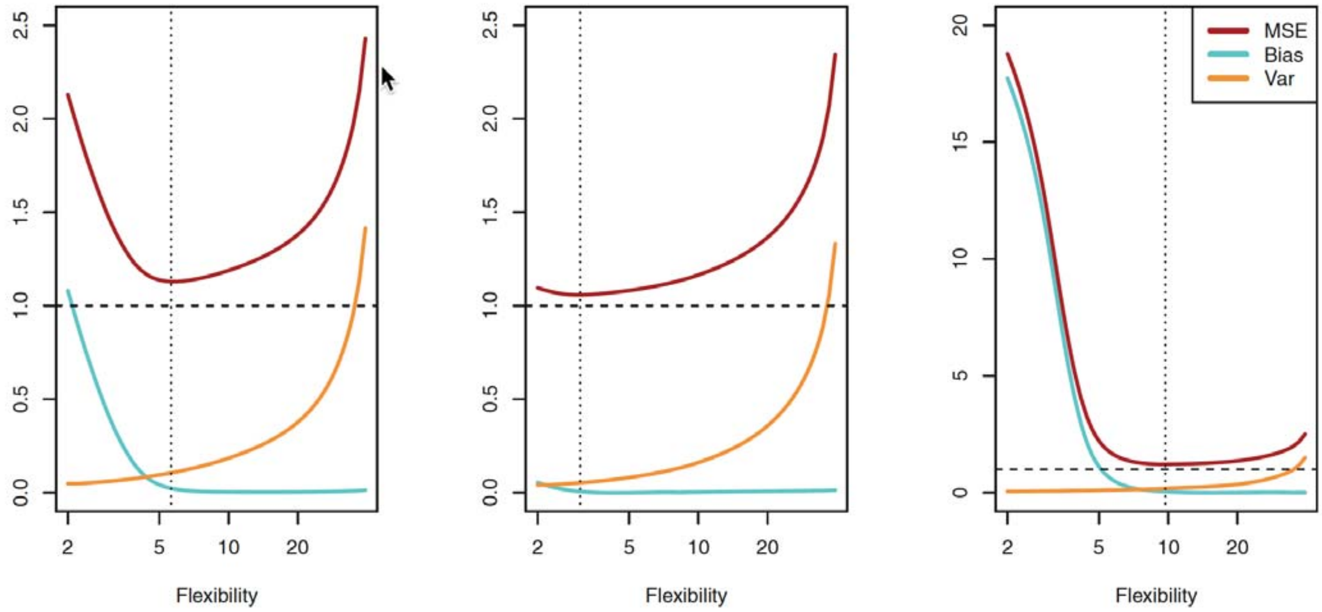
- **Bias-Variance Decomposition:** The expected test MSE can always be decomposed as
  $E\big[(Y - \hat{f}(X;\mathcal{D}))^2\big] =$

Bias-Variance Tradeoff (2/2)

- Variance $E\big[(\hat{f}(X;\mathcal{D} - E_{\mathcal{D}}[\hat{f}(X;\mathcal{D})|X])^2\big]$ measures the amount by which $\hat{f}$ would change if fitted to a different training data set.
- Bias $E\big[(f(X) - E_{\mathcal{D}}[\hat{f}(X;\mathcal{D})|X])^2\big]$ measures the error that is introduced by approximating a complex real-life problem by a simpler model.
- Noise $\mathrm{Var}(\epsilon)$ measures error caused by factors not predictable by X.

➔ Flexible models have small bias, but high variance.

Example of Bias-Variance Tradeoff



Non-linear f Linear Highly non-linear

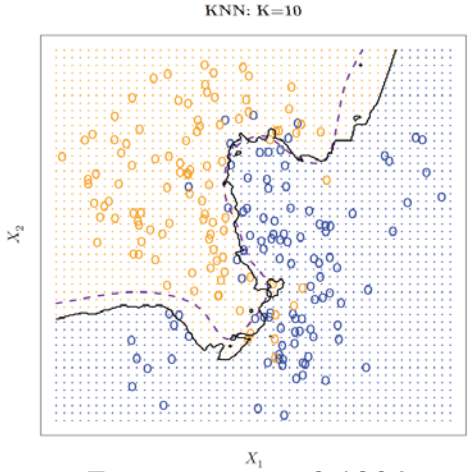
Classification Setting

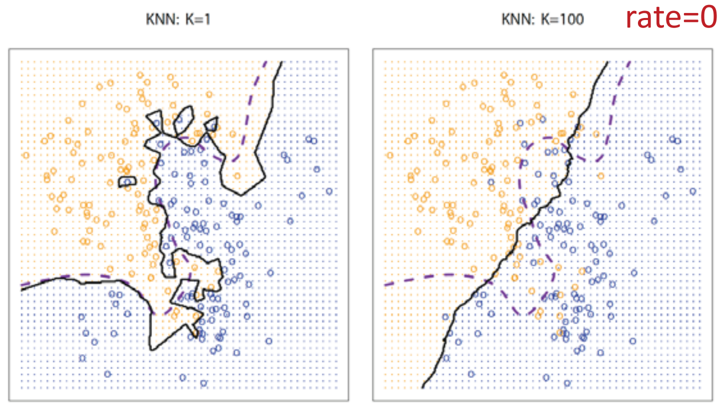
- Recall that, in kNN, classification is determined by

$$\hat{f}_{\text{kNN}}(x) = \arg\max_{\hat{y}\in\mathcal{Y}} \frac{1}{k} \sum_{i: x_i \in \mathcal{N}_k(x)} I(y_i = \hat{y}).$$

and the test error rate is

$$\text{test error rate} = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(x,y)\in\mathcal{D}_{\text{test}}} I(y - \hat{f}_{\text{kNN}}(x; \mathcal{D}_{\text{train}}))$$





Bayes error rate=0.1304.

Test error rate=0.1304 0.1695 0.1925

Training vs Test Error Rates