AI Academy
- Machine Learning -

Junhee Seok, Ph.D.
Associate Professor
School of Electrical Engineering
Korea University, Seoul, Korea

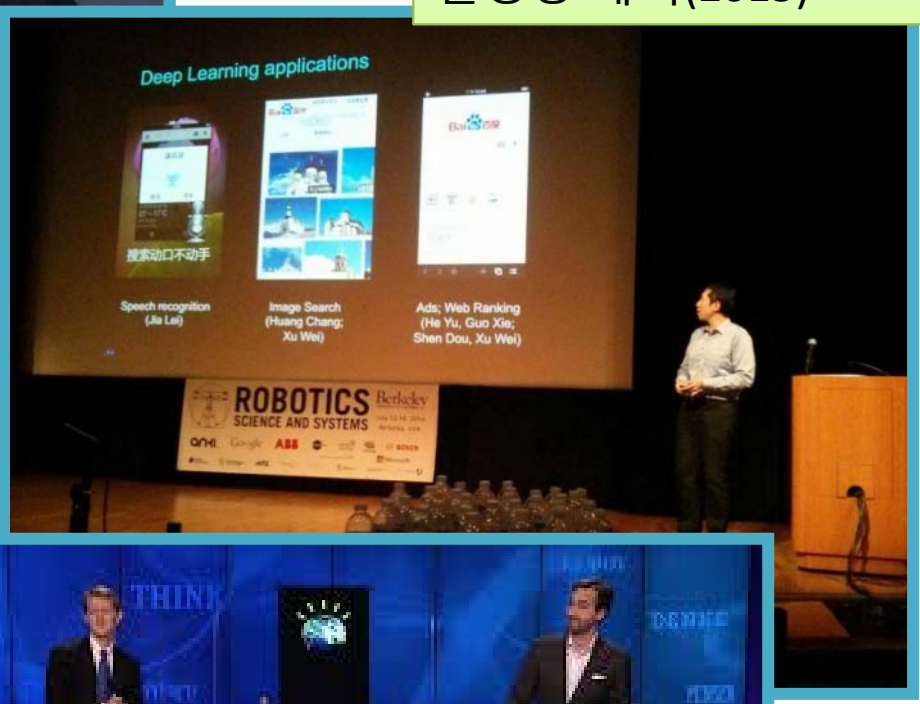
Contents

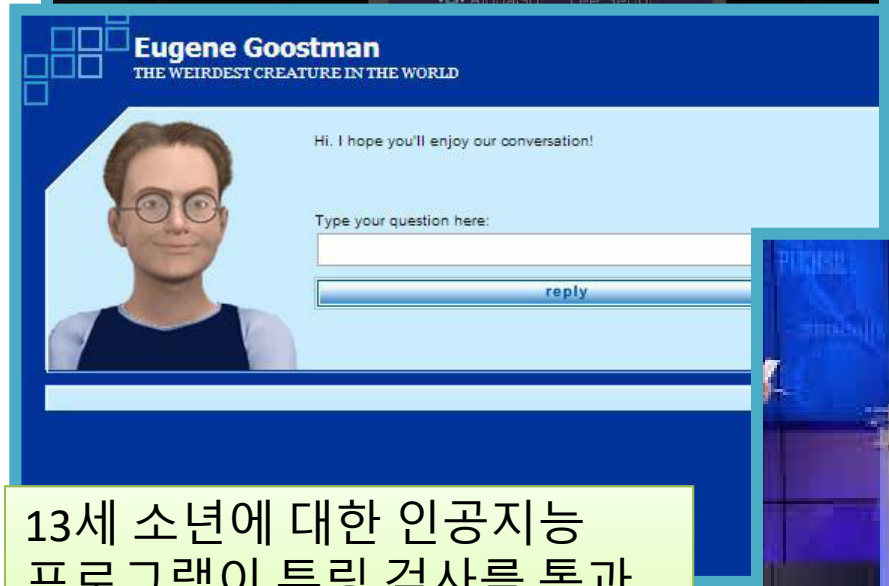
Overview of Machine Learning

인공지능: 인간처럼 생각하는 기계

알파고와 이세돌 9단의 대국 (2016)

바이두의 초거대 딥러닝 신경망 계획(2015)





13세 소년에 대한 인공지능 프로그램이 튜링 검사를 통과 (2014)

IBM 왓슨이 인간 챔피언과의 퀴즈 대결에서 승리 (2011)

인공지능의 정의

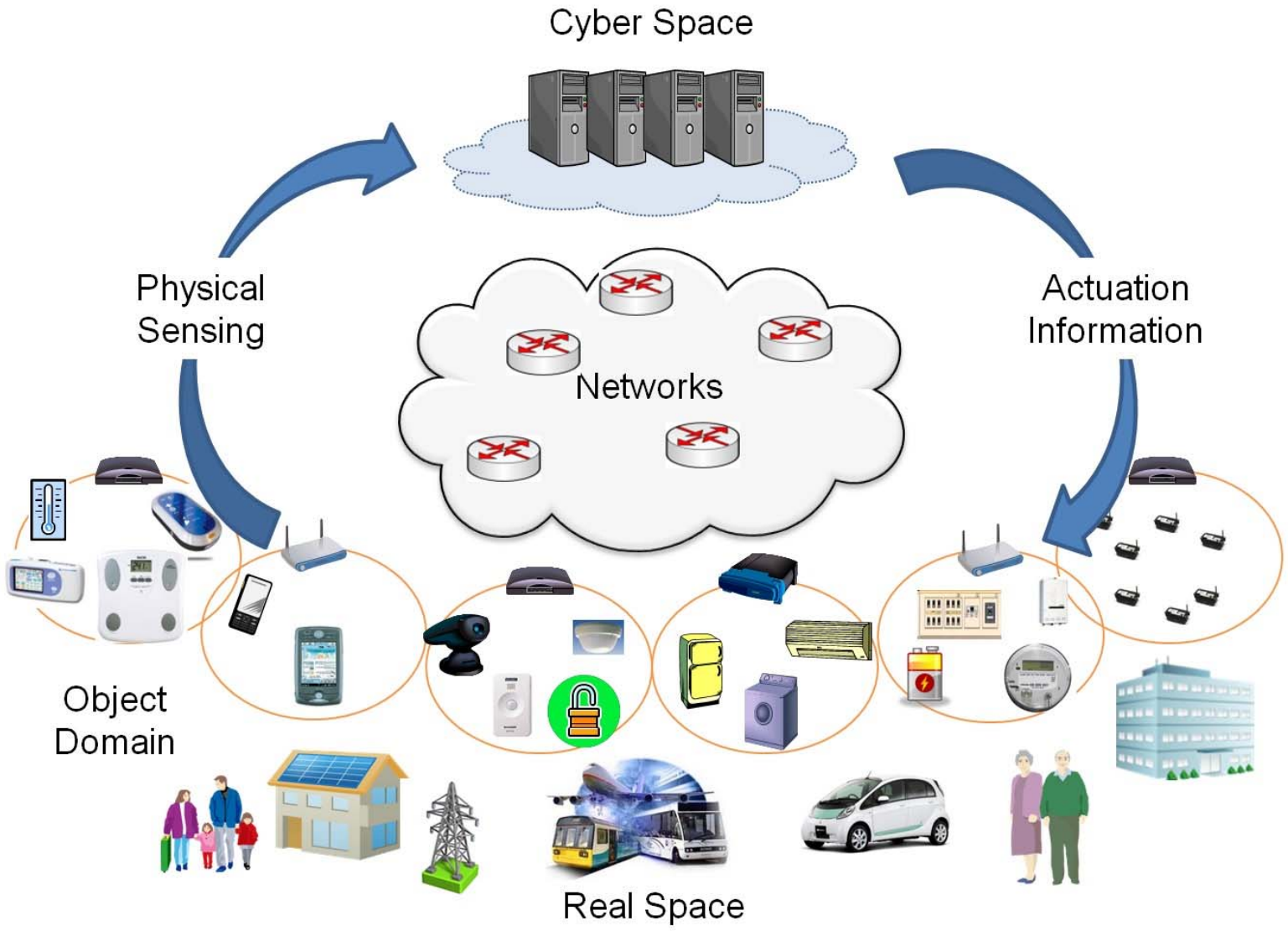
- 인공지능이란?
  - **지능이라는 추상적 개념을 과학적으로 구체화하고, 이를 인공적으로 재현하는 기술**

- 강한 인공지능 (Strong AI; 범용인공지능)
  - 인간처럼 혹은 초인간적인 방법으로 실제 사고를 통해 문제를 해결하는 기술
  - 예: 창조, 감성, 사고

- 약한 인공지능 (Weak AI)
  - 인간의 지능을 모방하여 지적 문제를 해결하는 기술
  - 예: 지식의 발굴, 자료 처리, 상황 판단

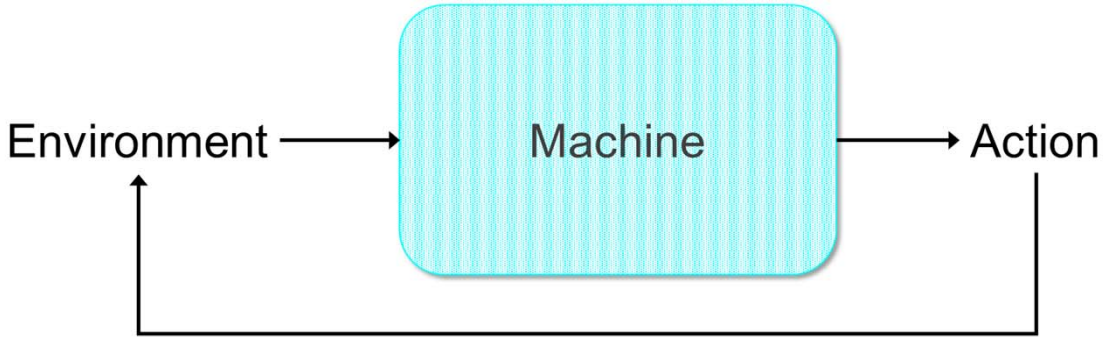
4차 산업혁명과 인공지능

- 사이버물리시스템: 4차 산업 혁명의 기술을 포괄하는 개념



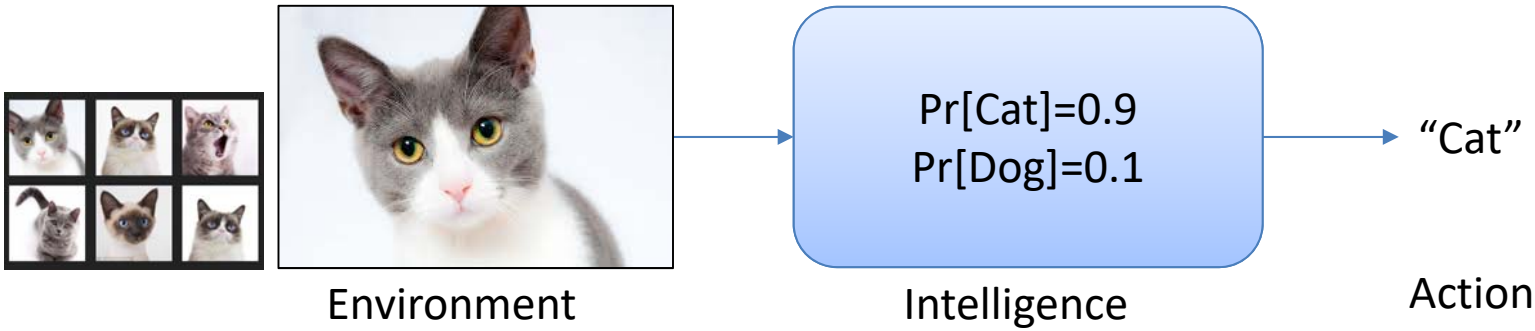
AI vs. Machine Learning

- Intelligence needs to "perceive and decide" for "actions" from "observed data"

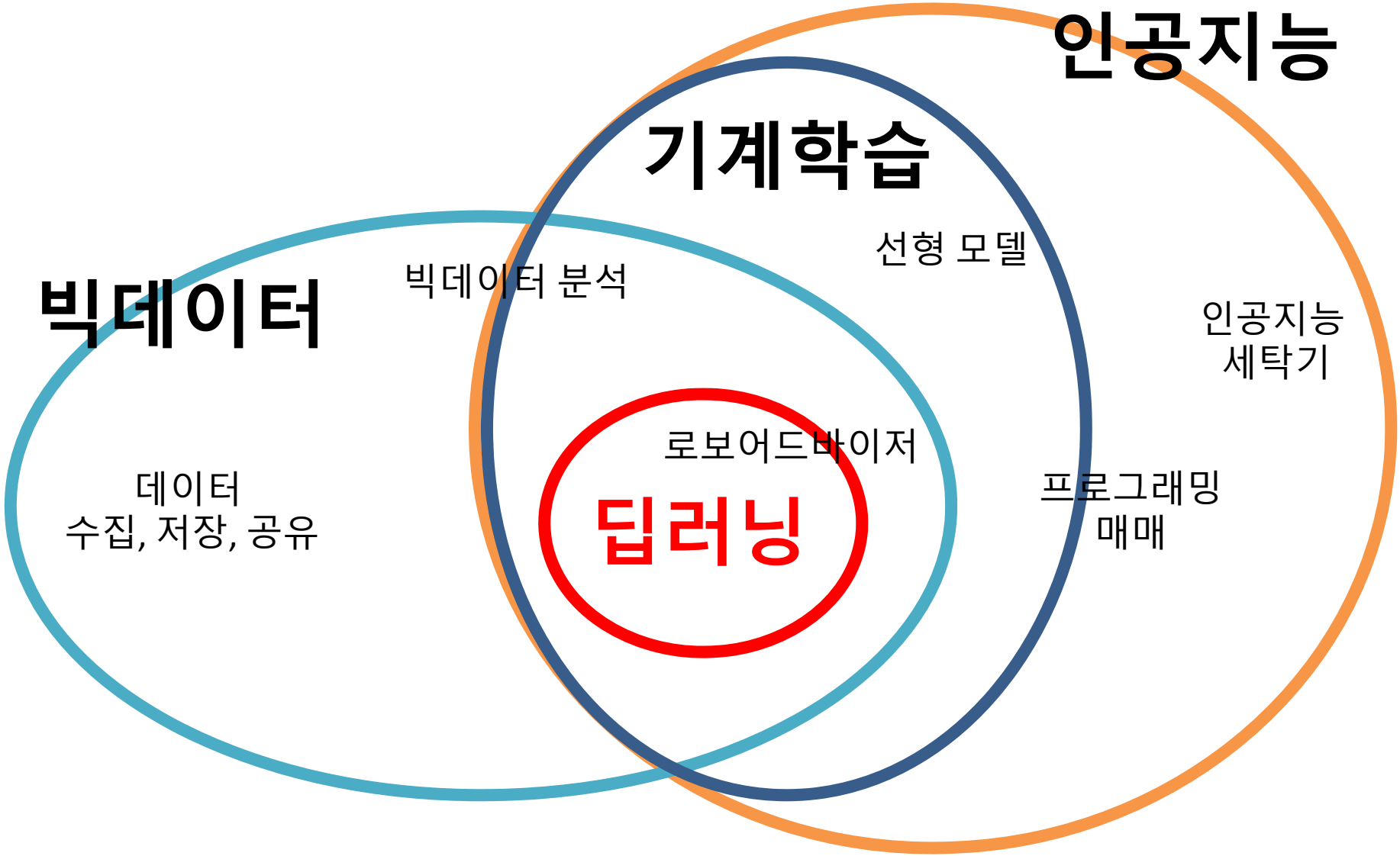


$$\Pr[Y|X]$$

- Environment: random data from a certain distribution (e.g. weather , speech, …)
- Action: often based on probabilistic predictions

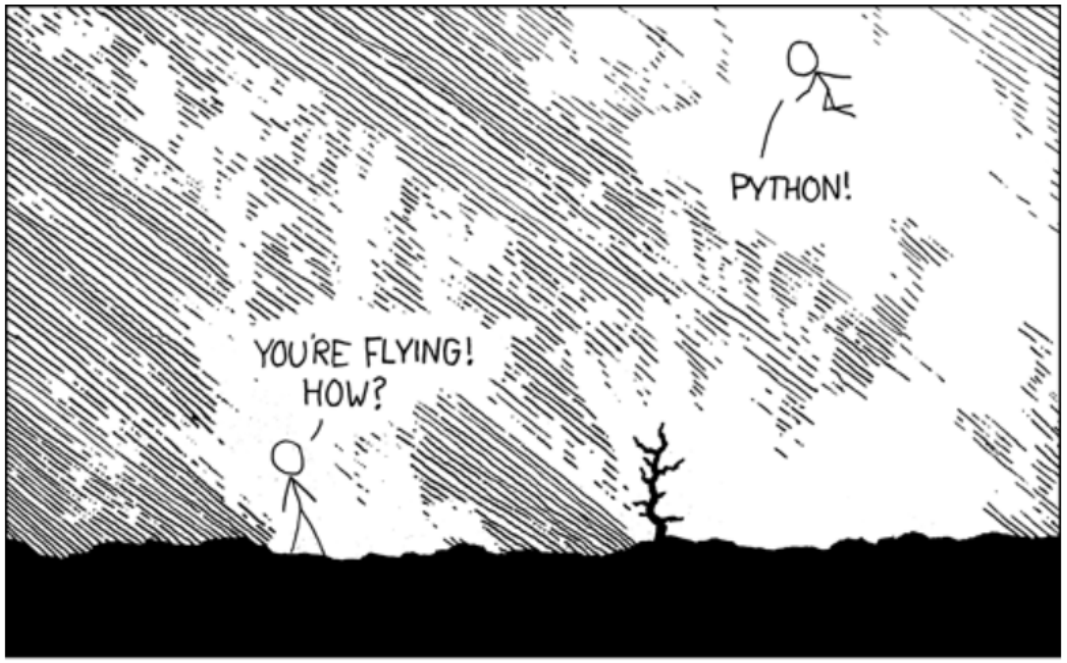


인공지능, 빅데이터, 기계학습, 딥러닝??



Python Review

What is Python?



https://xkcd.com/353/

- Multi-purpose (GUI, Web, Scripting, etc …)
- Object-oriented
- Interpreted
- Static and dynamic typing
- Focus on readability and productivity

Pandas Module

- An open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis
  - A set of labeled array data structures, the primary of which are Series and DataFrame
  - Index objects enabling both simple axis indexing and multi-level / hierarchical axis indexing
  - An integrated group by engine for aggregating and transforming data sets
  - Date range generation (date_range) and custom date offsets enabling the implementation of customized frequencies
  - Input/Output tools: loading tabular data from flat files (CSV, delimited, Excel 2003), and saving and loading pandas objects from the fast and efficient PyTables/HDF5 format.
  - Memory-efficient “sparse” versions of the standard data structures for storing data that is mostly missing or mostly constant (some fixed value)
  - Moving window statistics (rolling mean, rolling standard deviation, etc.)

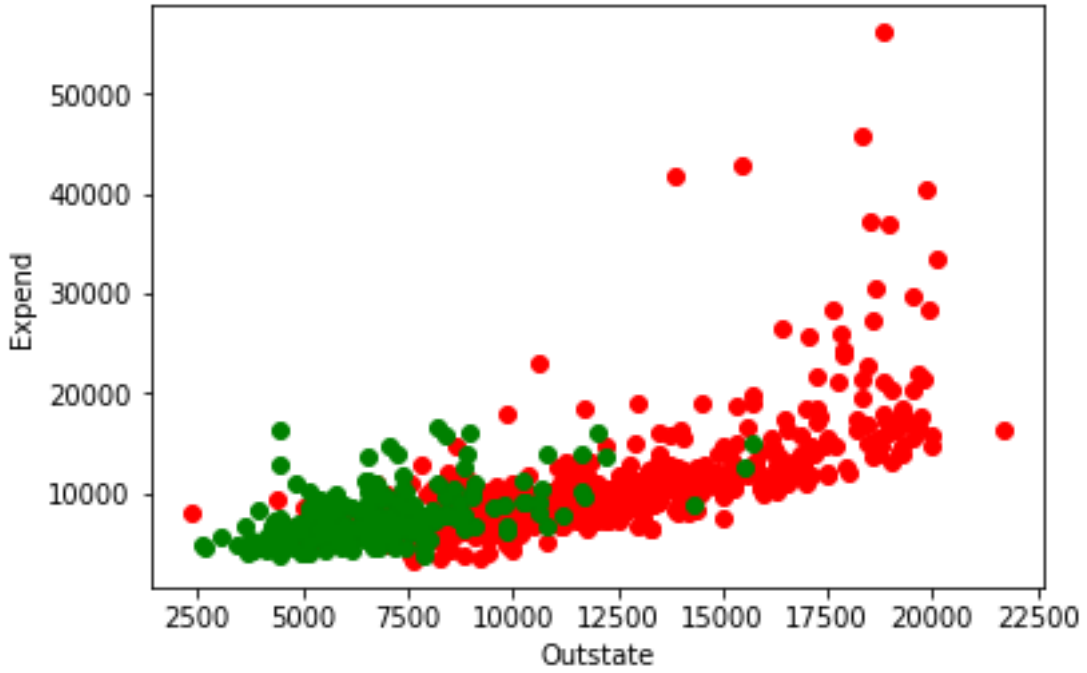
Pandas Module

- Usual Python data structures are for numeric data or string data.
  - Not suitable for data analytics with mixed data types
- Pandas provides a data frame that can include numeric and categorical data

- pandas.read_csv()
- pandas.DataFrame()
- pandas.Series()
- pandas.DataFrame.iloc(), pandas.DataFrame.loc()
- Pandas.DataFrame.plot()

Practice

- Read data02_college.csv and answer the following questions
  - How many colleges? How many private and public?
  - The average expend of private and public colleges?
  - What are the top 10 schools in terms of top 10% of high school class?
  - What are the top 10 schools in terms of acceptance ratio?
  - Plot outstate tuition and expend with different colors for private and public schools.



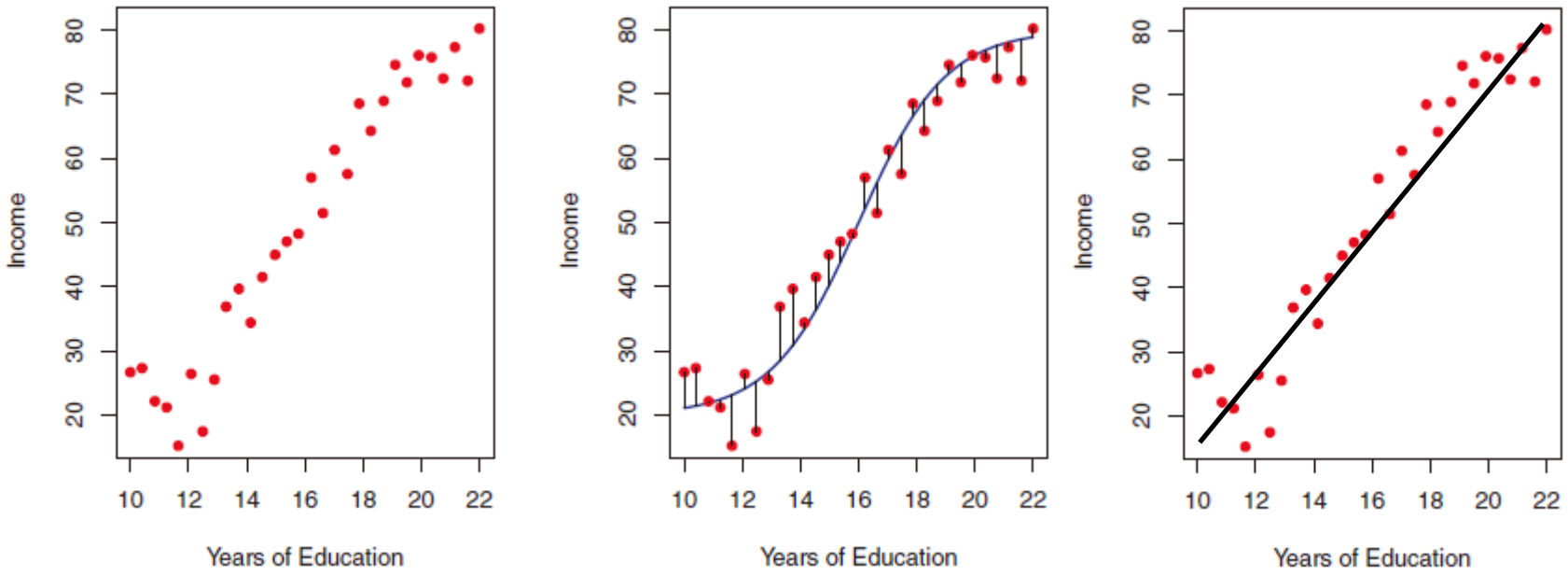
Machine Learning Fudementals

Machine Learning

- For an output Y and input $X = (X_1, X_2, \ldots, X_p)$, the relation can be generally presented by

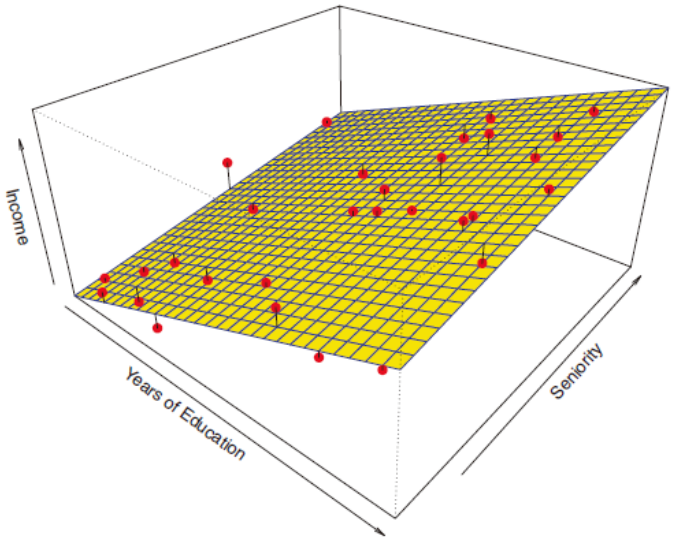
$$Y = f(X) + \epsilon$$

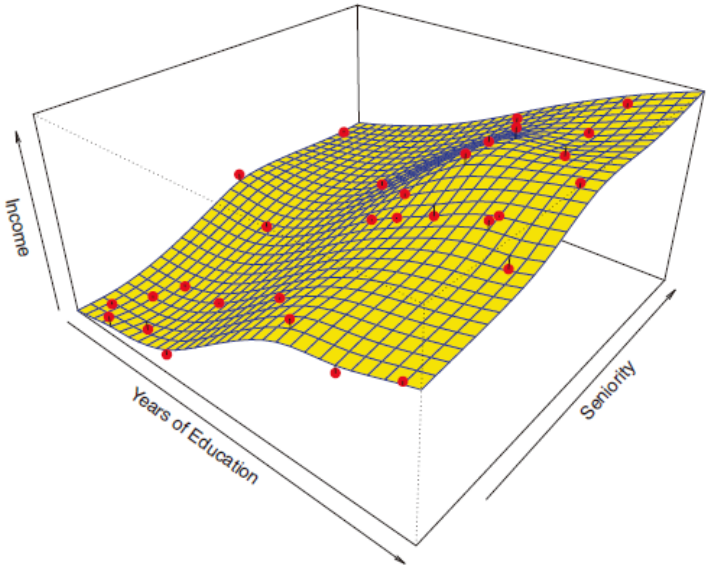
- $f()$ is unknown, can be a simple linear function or complicated non-linear form.
- We want to estimate $f()$ for *prediction* and *inference*.



Estimating *f*()

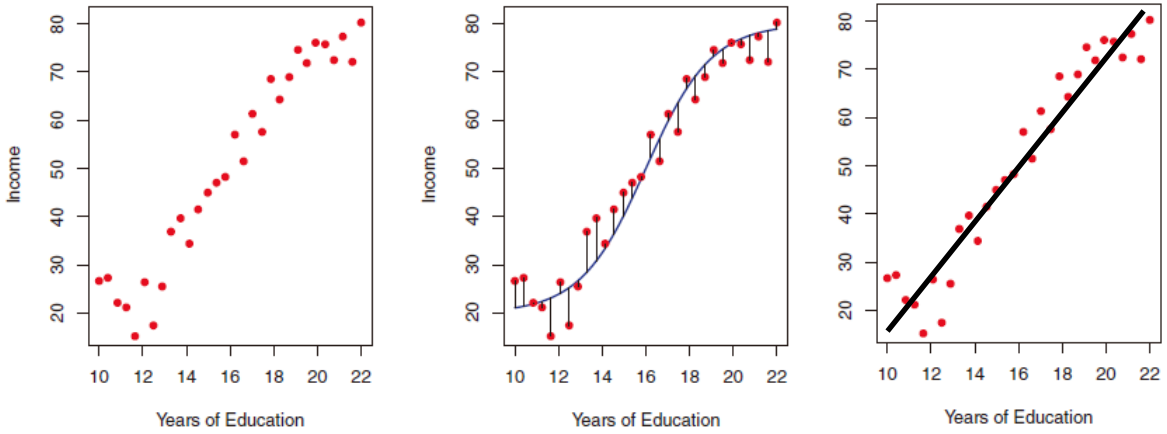
- **Parametric** approaches assumes a certain type of relation (usually linear).
  - e.g. income ~ β_0 + β_1 x education + β_2 x seniority
- **Nonparametric** approaches assumes no prior relation.
  - Fitting *Y* not much roughly and not much wiggly ☺
  - e.g. tree-based methods, splines



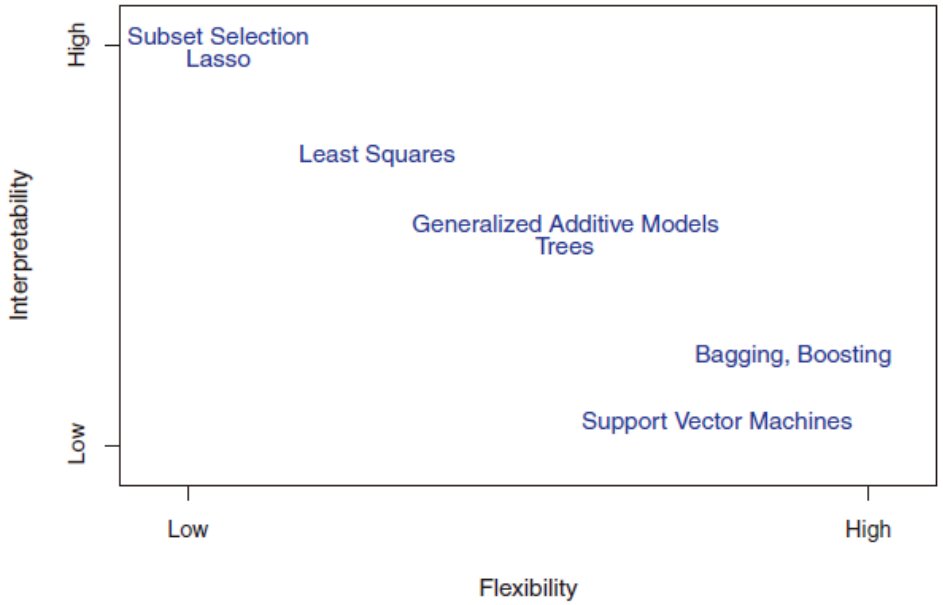


Trade-off between Flexibility and Interpretability

- The spline function is more flexible, but the linear line is more interpretable.



- Trade-off of various methods.

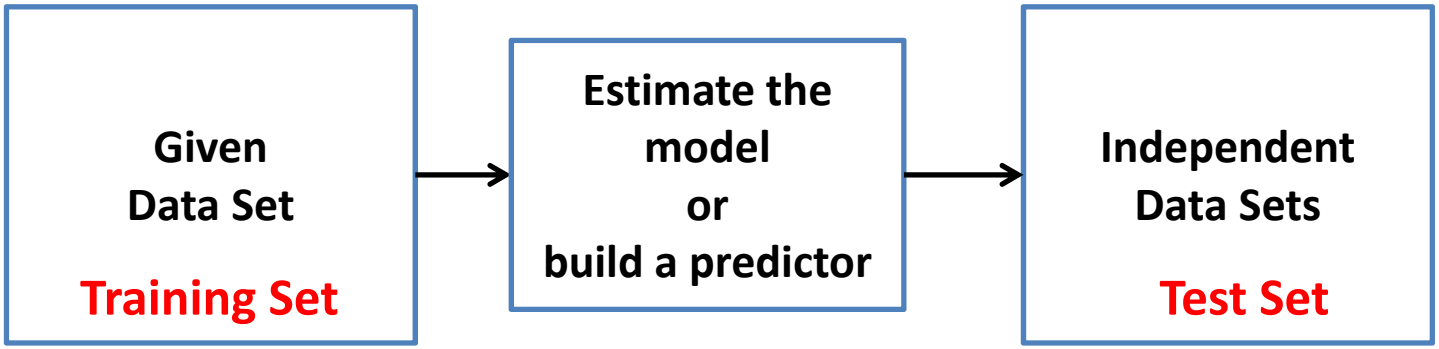


Supervised Learning Problem Setting

- In a real problem, the accuracy $E(Y - \hat{f}(X))^2$ is often measured by mean square error (MSE),

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{f}(x_i)\right)^2$$

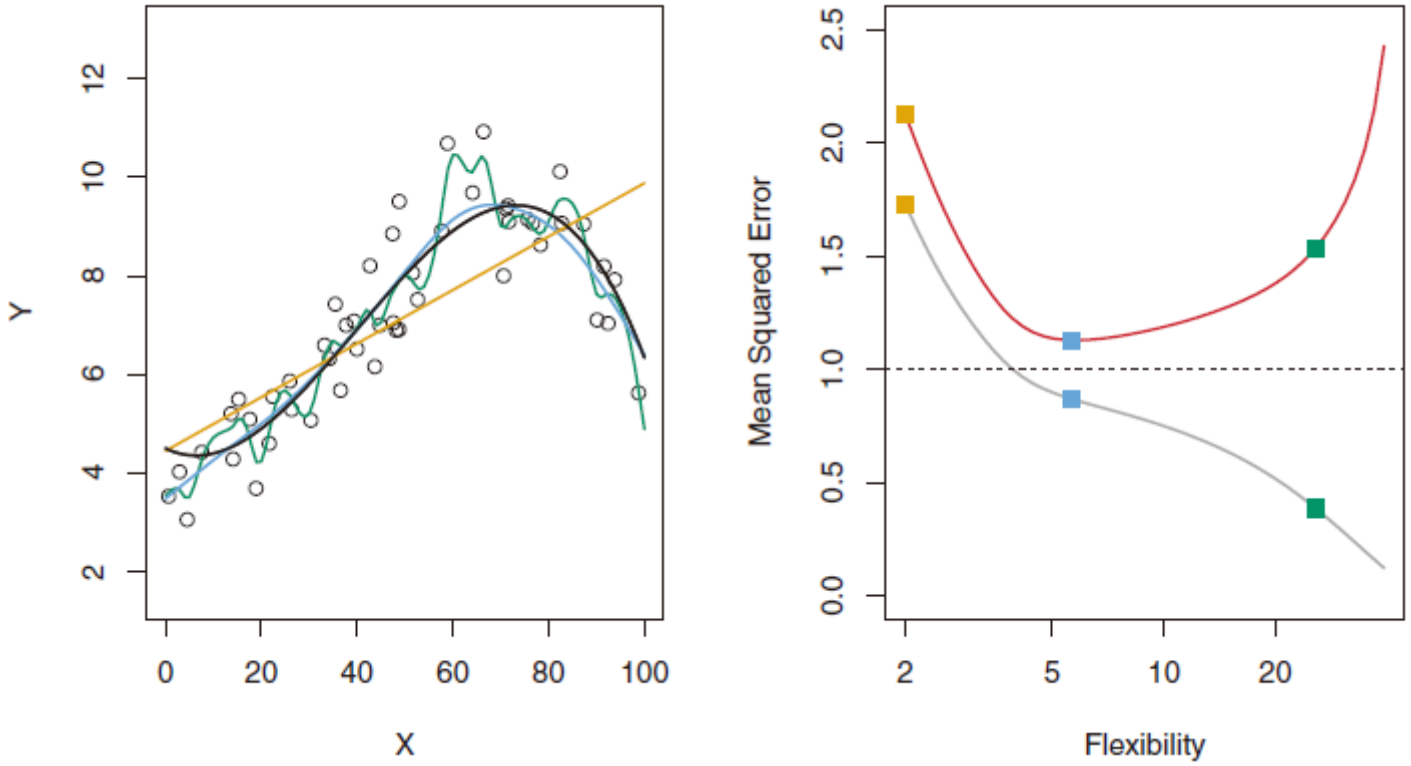
- *Given a data set*, we want to estimate *f*() that will make small MSEs in *other data sets*.
  - Estimating a relation between salary and education using data from Seoul, and confirming it using data from Pusan.
  - Building a predictor of heart attacks with blood pressure using data in 2001~2005, and testing its performance using data in 2006~2010.



Note that we know nothing about the test set when building a predictor.

Training MSE vs. Test MSE as Model Flexibility

- We can reduce the MSE of the training set as much as we want by increasing the model flexibility.
- **Overfitting**: too much flexibility increase the MSE of the test set.
- The model flexibility is often referred by the *degree of freedom*.



Machine Learning Contents

- Supervised learning

	Regression	Classification
Linear Model	Linear Regression	Logistic Regression
Discriminant Analysis		LDA/QDA
Nonparameteric	KNN	KNN, Naïve Bayesian
Tree	Regression Tree	Classification Tree
Ensemble	Random Forest, Boosted Tree	
Support Vector	Support Vector Regression	Support Vector Machine
Neural Networks	Multi-layer Perceptron and others Deep learning	

- Unsupervised learning
  - Dimension reduction: PCA, ICA, Autoencoder
  - Clustering: K-means, hierarchical
- Model selection
  - Cross-validation
  - Feature selection, penalization

General Procedure of Machine Learning

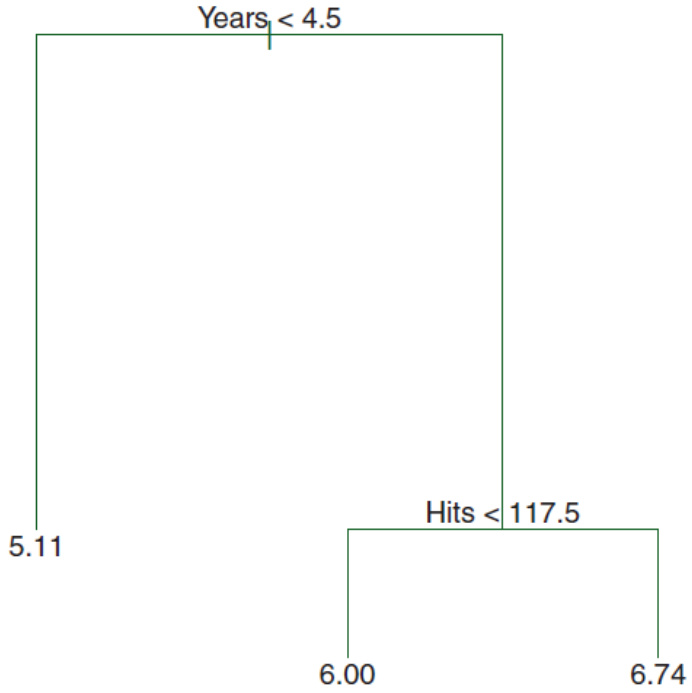
- Set up a problem → what is Y?
- Data collection → collect X and Y
  - (Traditionally) design a study, perform experiments, and collect data
  - (Now) dig up a database and collect any related data
- Preprocessing
  - Transform data into usable form
- Exploratory data analysis (EDA)
  - See how data looks like
- Data analysis (prediction)
  - Initial data size: n = 100M, p = 10k
  - Separate training and test set (often by data collection time)
  - Select features via univariate statistical test (t-test, cor-test)
  - (optionally) Dimension reduction (PCA)
  - Select learning methods (often based on intuition)
  - Model selection via cross-validation (methods & parameters)
  - Test performance over the test set
- Validation with a totally new data set (often not existing data when the model is built)

Decision Trees

Decision Tree

- Very intuitive and traditional
  - Often has high interpretability but less accuracy.

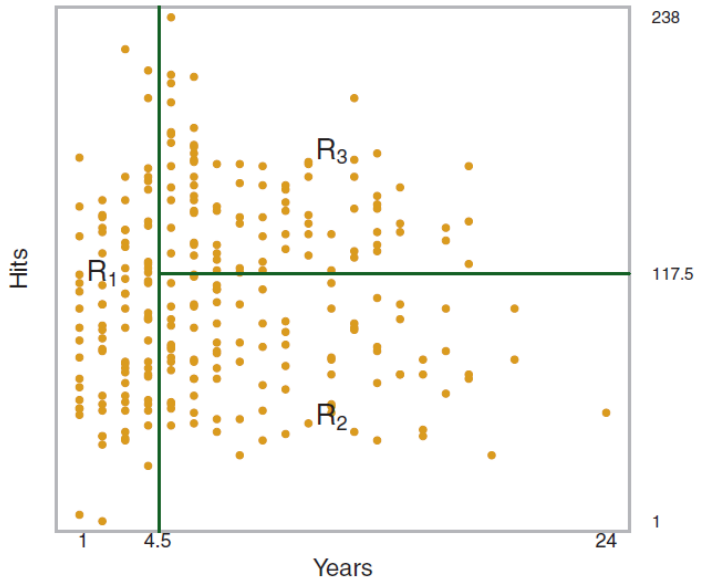
- Example: predicting log(salary) of baseball players

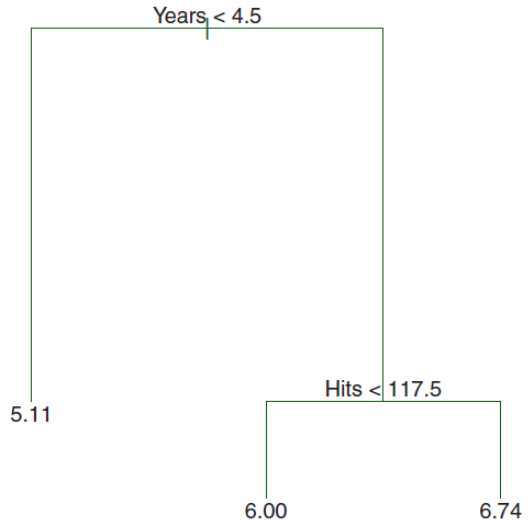


- CART, Random Forest, Bagging and Boosting.

Regression Tree

- Partitioning the sample space for regression





- The goal of the partitioning is to find the subregions R_1, R_2, …, R_J that minimizes the RSS

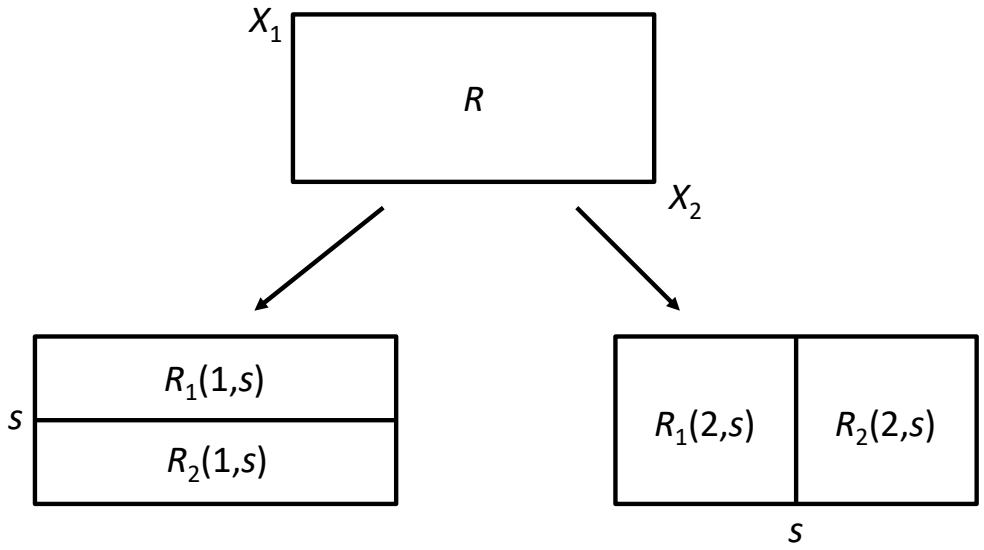
$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

The average y value in R_j.

Regression Tree

- Too many choices for partitioning: infeasible to explore all.

- **Regression tree** uses *recursive binary splitting* (top-down, greedy).

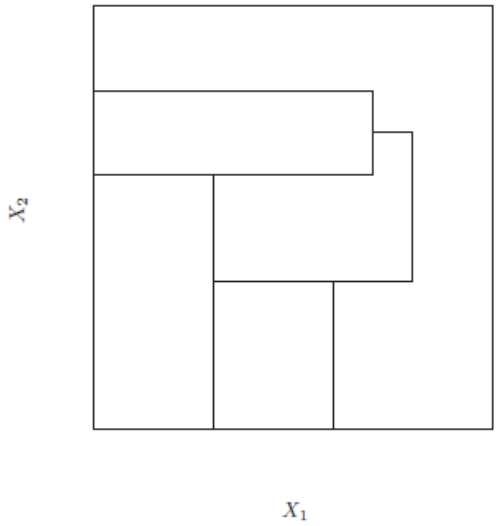


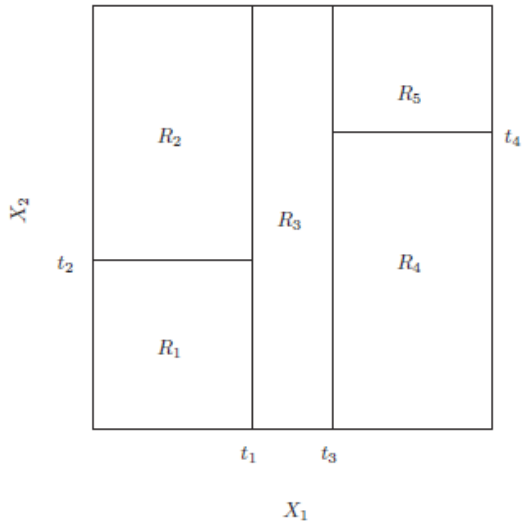
  - Binary splitting: $R_1(j,s) = \{X | X_j < s\}$ and $R_2(j,s) = \{X | X_j \geq s\}$
  - Find the j and s that minimize

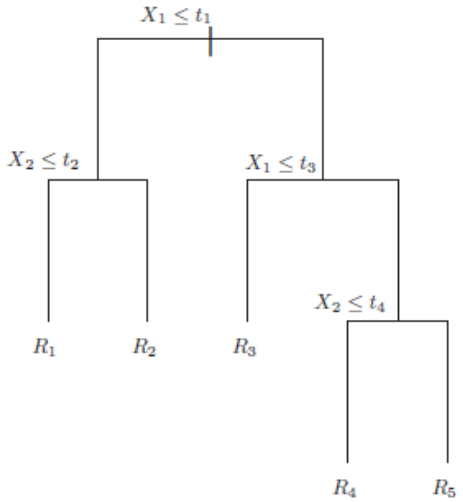
$$\sum_{i:\, x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\, x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2,$$

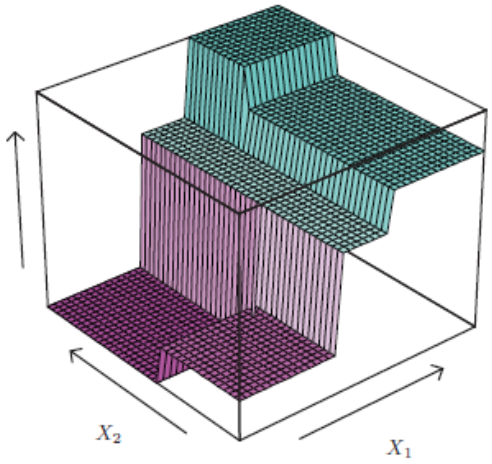
  - Repeat the binary splitting for subregions.

Example


X_2
X_1


R_5
R_2
t_4
X_2
R_3
t_2
R_4
R_1
t_1
t_3
X_1


$X_1 \leq t_1$
$X_2 \leq t_2$
$X_1 \leq t_3$
$X_2 \leq t_4$
R_1
R_2
R_3
R_4
R_5


X_2
X_1

Tree Pruning

- Recursive binary splitting eventually results in a huge tree (T_0) of which leaves have only one sample.
  - Overfitting: too small MSE in training data, but large MSE in test data.

- Pruning
  - Finding a sub-tree of T_0 that minimizes the test error through cross-validation.
    - Often difficult because there are to many subtrees.
  - Cost complexity pruning: introducing penalty term for the number of leaves.
    - α is a tuning parameter can be determined through cross-validation.
    - $|T|$ is the size of tree T, which is the number of leaves or end-nodes.

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

Regression Tree Algorithm

Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

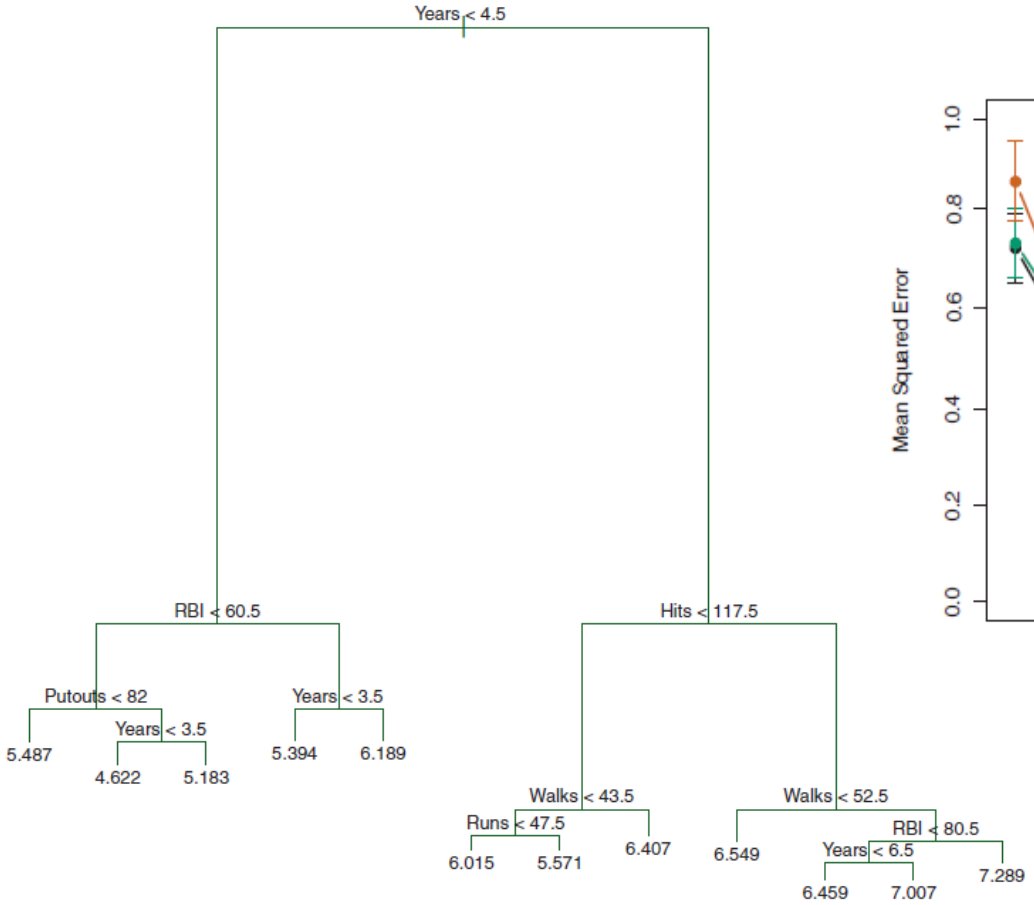
   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

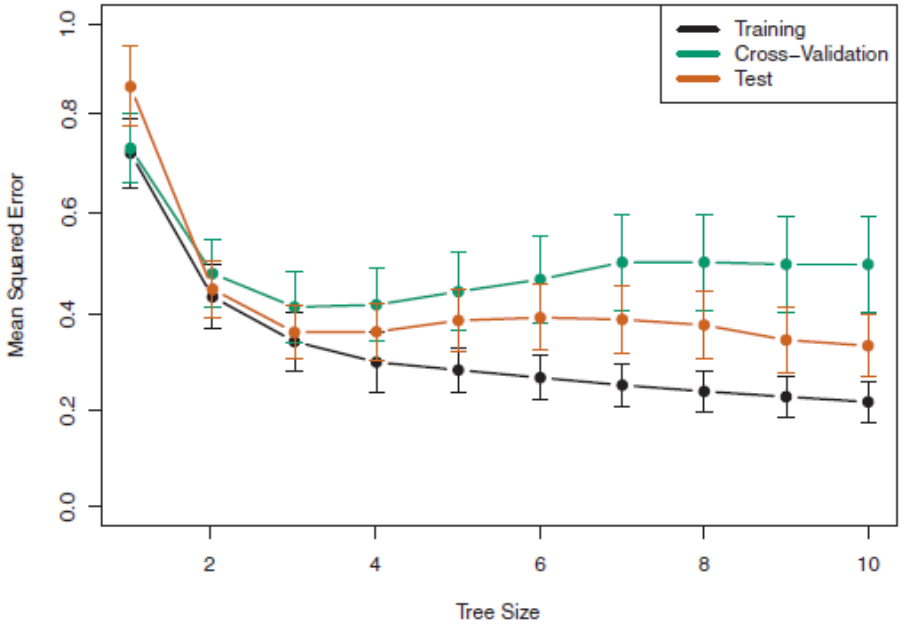
   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Example

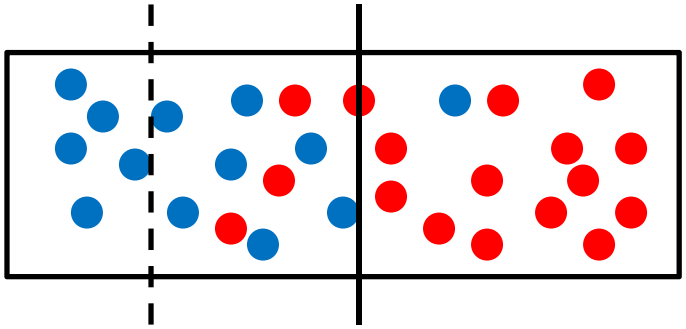
- Predicting log(salary) of baseball players with nine predictors.





Classification Tree

- Similar to regression tree, but for classification problems.
- It predicts the class of a region with the most commonly occurring class.



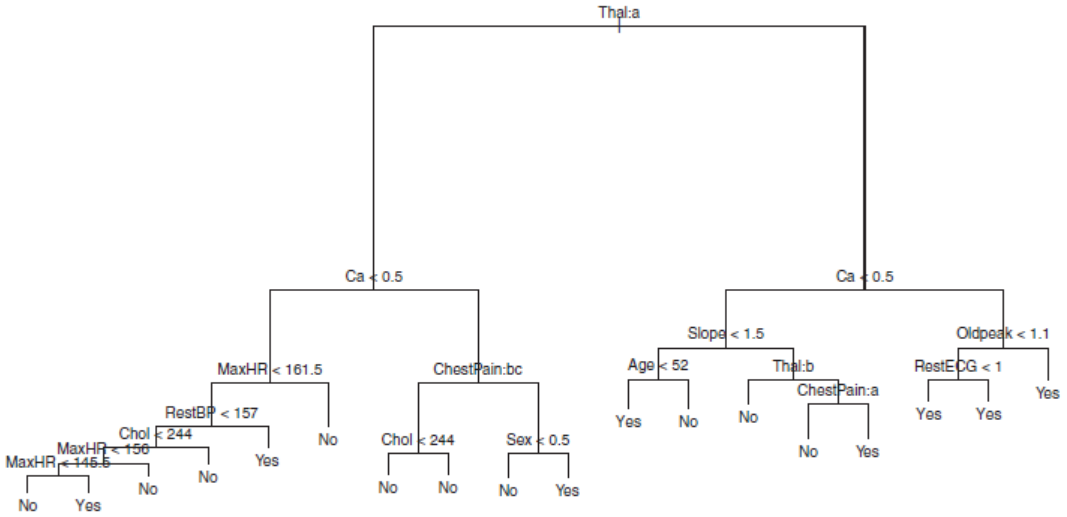
- Classification tree uses other metrics than RSS.
  - Classification error rate: $E = 1 - \max_k(\hat{p}_{mk})$.
  - Gini index:

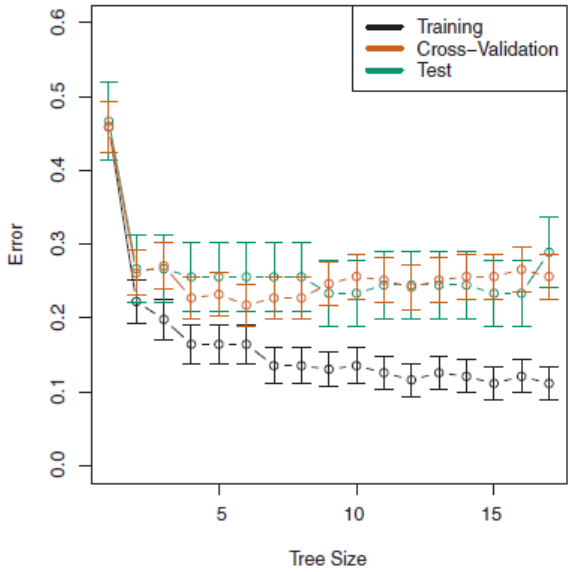
$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

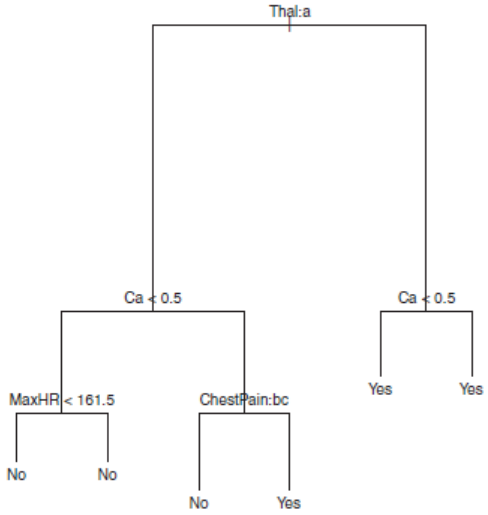
  - Cross-entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}.$$

Example

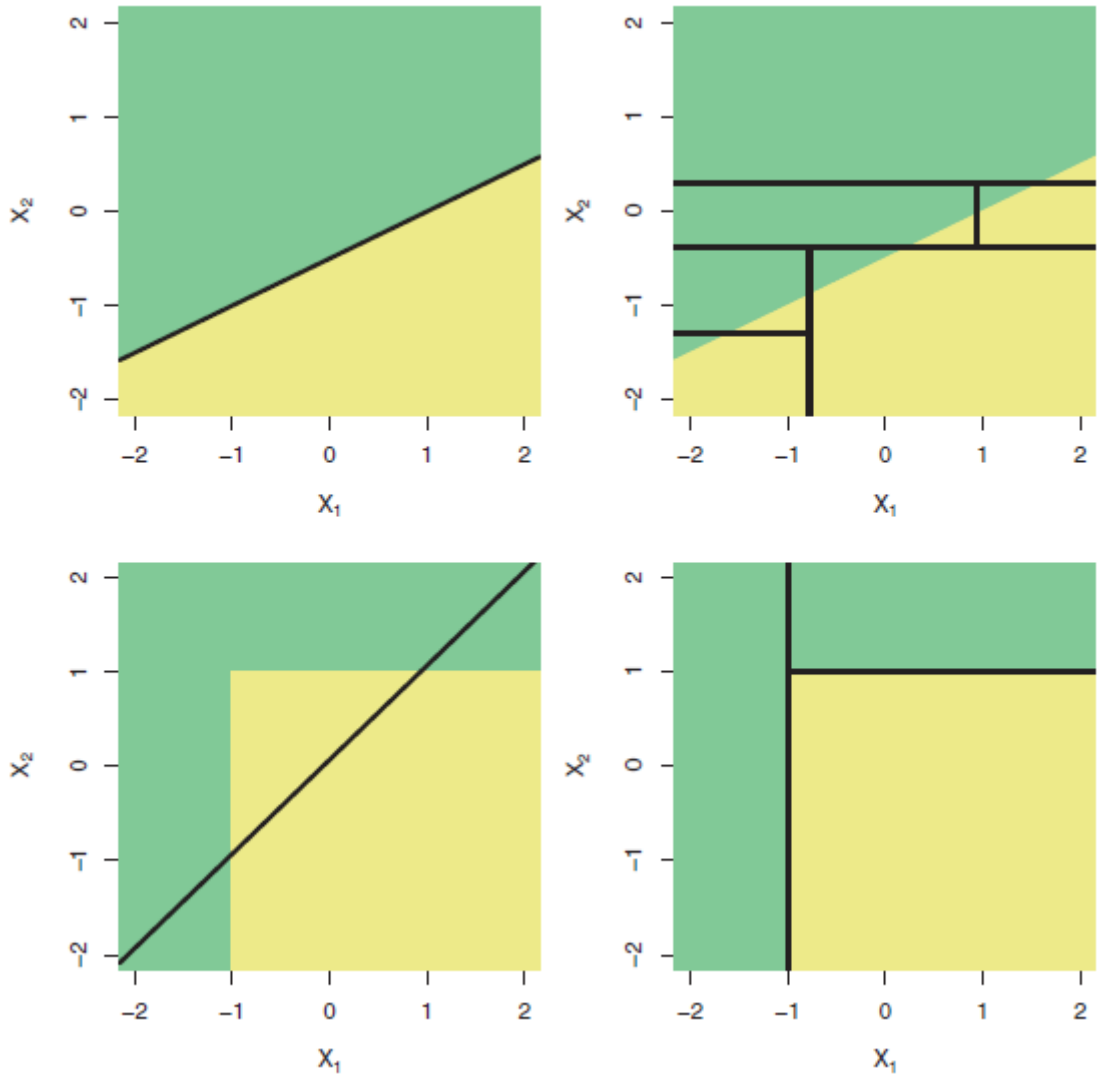

Thal:a
Ca < 0.5
Ca < 0.5
MaxHR < 161.5
ChestPain:bc
Slope < 1.5
Oldpeak < 1.1
Age < 52
Thal:b
RestECG < 1
RestBP < 157
Chol < 244
MaxHR < 156
MaxHR < 145.5
Chol < 244
Sex < 0.5
ChestPain:a
No
Yes


Training
Cross-Validation
Test
Error
Tree Size
0.0
0.1
0.2
0.3
0.4
0.5
0.6
5
10
15


Thal:a
Ca < 0.5
Ca < 0.5
MaxHR < 161.5
ChestPain:bc
No
Yes

Tree vs. Linear Model

- Efficiency depends on data.



Pros and Cons of CART

- **CART**: Classification and Regression Tree.

- Pros
  - Easy to interpret, even easier than linear regression. (e.g. diabetes if 2h glucose>11.1 or fasting gluecose>7 or HbA>6.5).
  - More closely mirror human decision than linear methods.
  - Graphical presentation is possible.
  - Non-parametric

- Cons
  - Low prediction accuracy
  - Bagging, boosting and random forest to improve the prediction accuracy.

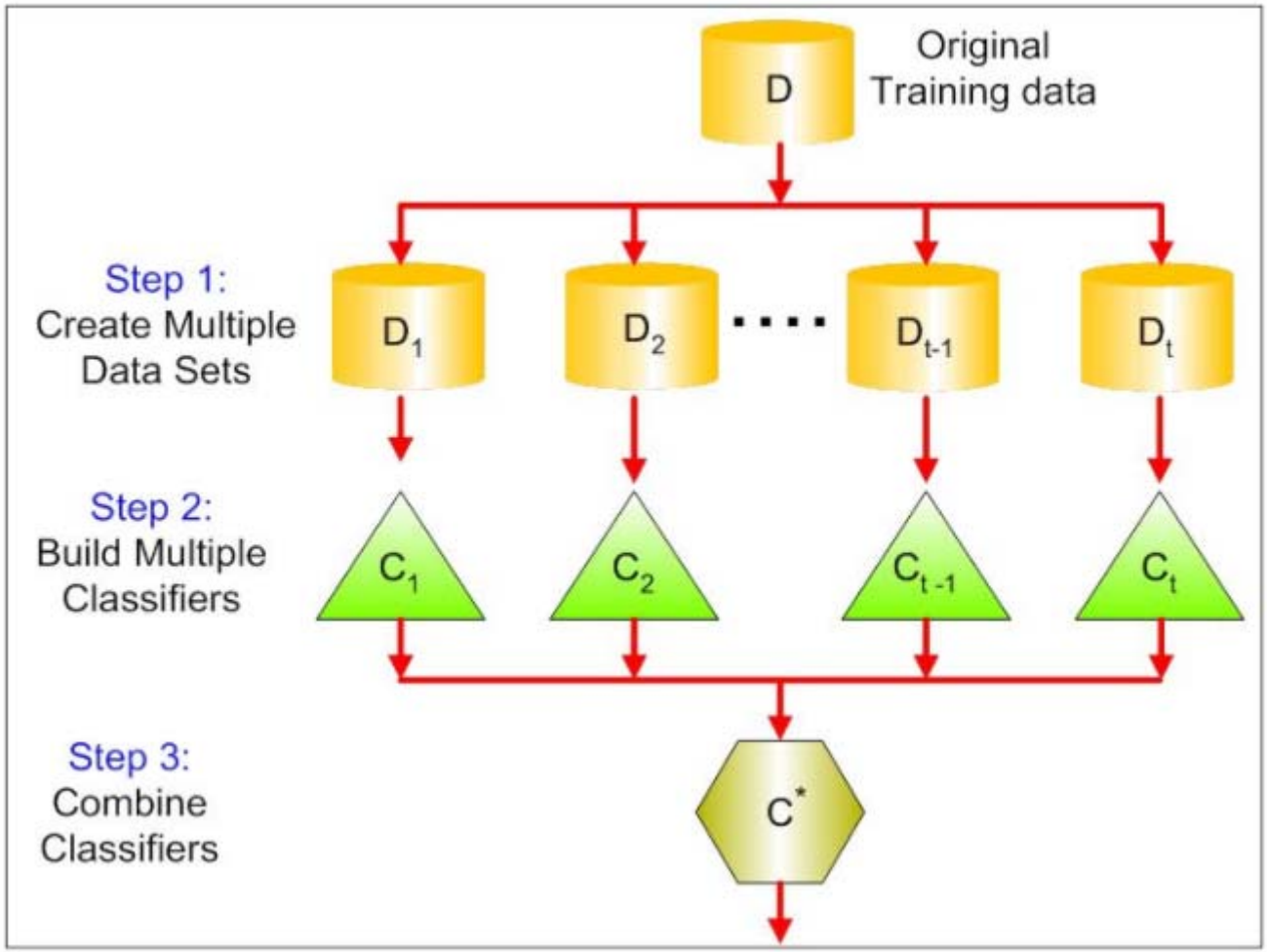
Practice

- For data05_boston.csv
  - Read the data file
  - Randomly build training and test set (training: 2/3, test: 1/3)
  - Using 5-fold cross-validation, tune the max_leaf_nodes of regression tree
  - If CV scores are too much wiggly, smooth it using spline
  - What is your best max_leaf_nodes for the best CV score?
  - What is your score for the test set?

Ensemble Methods

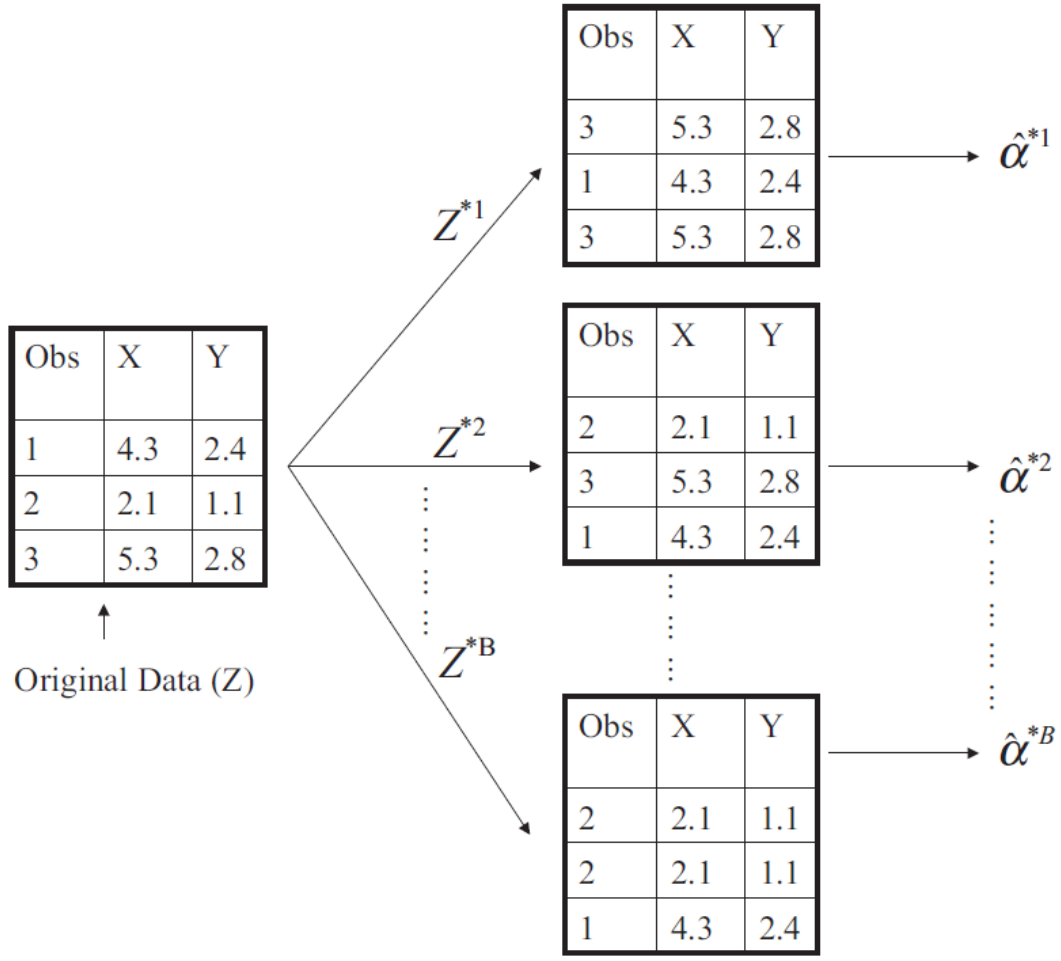
Ensemble Methods

- From a training data set, generate a set of predictors and make the final decision by aggregating the predictions of multiple predictors



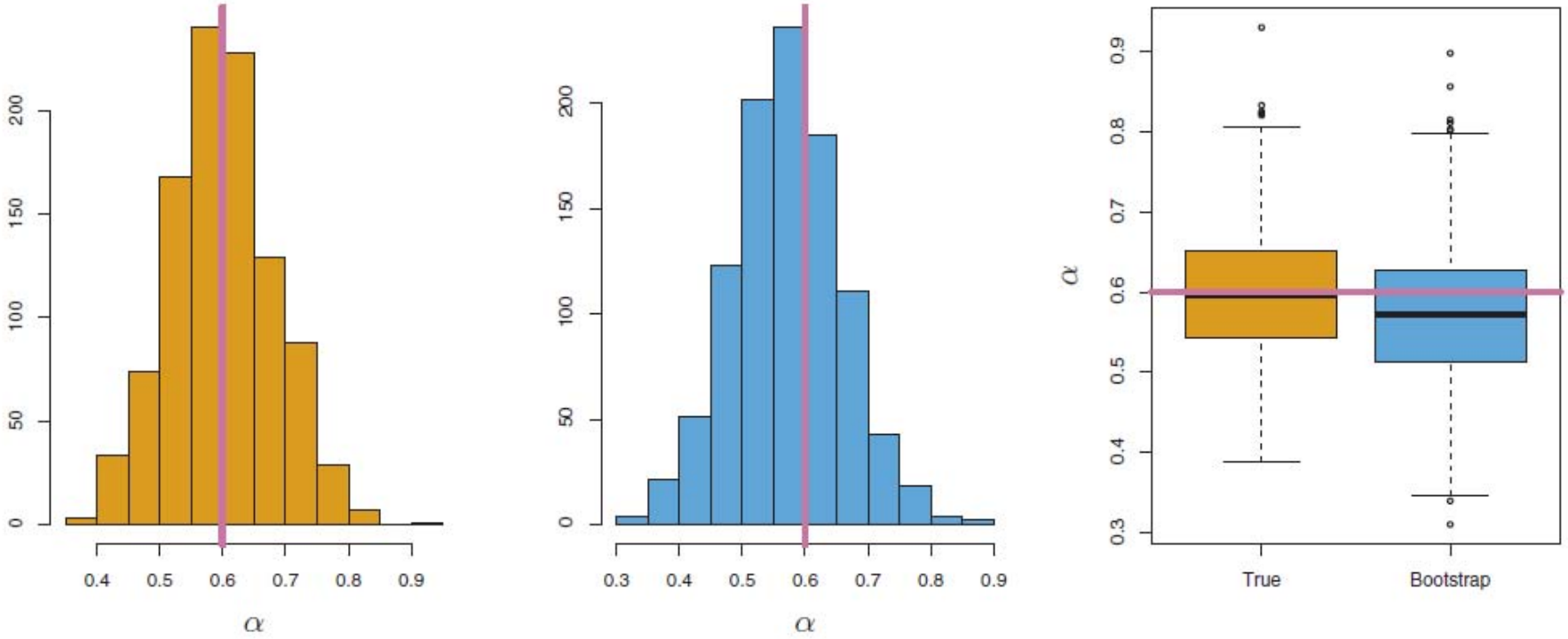
Bootstrap

- The **bootstrap** simulates the random sampling using the training set.
  - Randomly sampling *n* samples from a training set with replacement.
  - Calculating a statistic with the pseudo random samples.
  - Repeating it many times.



Bootstrap

- The result of the bootstrap gives a good approximation of the true distribution of a statistic.
  - Useful when the theoretical distribution of a statistic is difficult to be calculated.



Bagging (Bootstrap Aggregation)

- High variance is the main problem of CART.
- **Bagging** reduces the variance by averaging predictors.

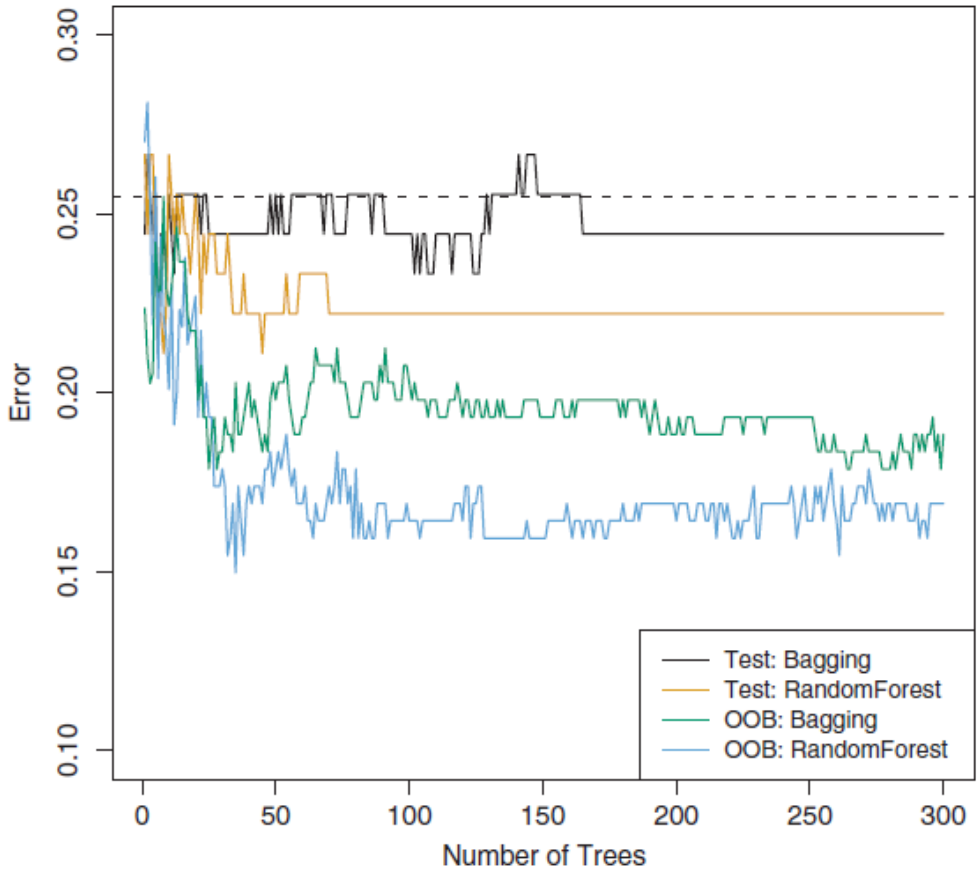
- Generating B different bootstrapped training data.
- For the b-th bootstrapped data, we apply CART and predict $\hat{f}^{*b}(x)$.
- By averaging 100 ~ 1000 prediction results, we make the final prediction,

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x).$$

  - Regression: averaging
  - Classification: majority voting

- Out-of-bag (OOB) error
  - One sample is not used for training (OOB) in about B/3 predictions.
  - We can use OOB samples to estimate test errors. (don't need cross-validation).

Example

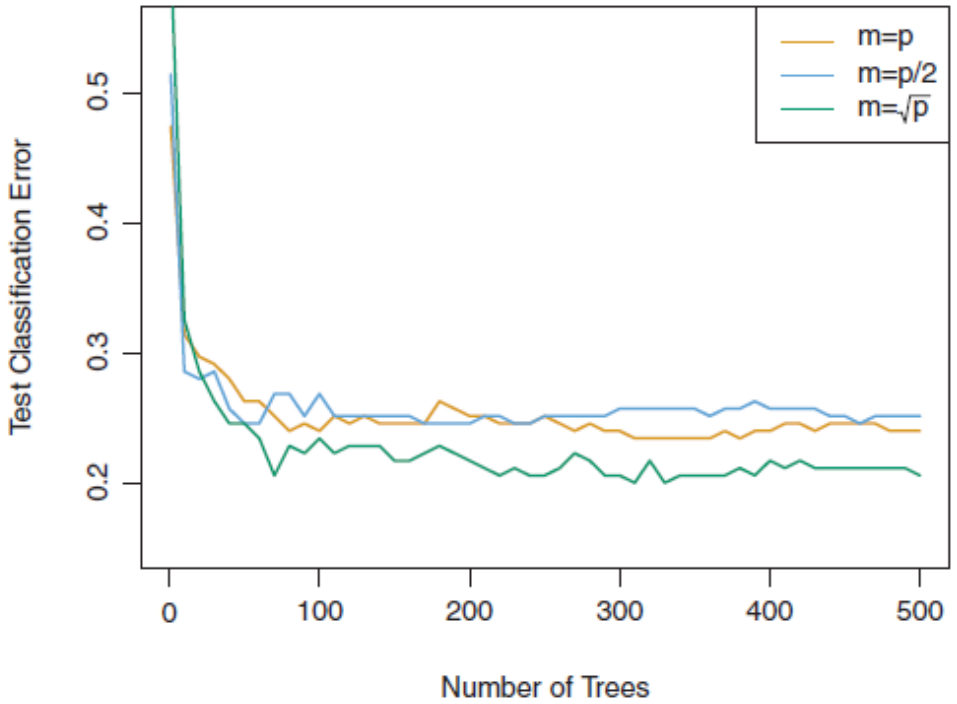


- Increasing *B* does not increase the variance.
- The error is stable after 100 trees.

Random Forests

- **Random forests** improve bagged trees by *decorrelating* the trees.
  - If there is a strong predictor, most of the bagged trees will have the predictor, which makes the begged trees similar to each other, or correlated.

- Random forests consider only m ($<p$) predictors randomly selected at each step of splitting.
  - Every others are the same with bagging.
  - m is often chosen to be $\sqrt{p}$.



Boosting

- **Boosting** is a general method to improve prediction accuracy, but here we are focusing on tree-based methods.
- Boosting learns sequentially and slowly using many simple trees, while bagging learns everything at once and combines many full trees.

Algorithm 8.2 *Boosting for Regression Trees*

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
   (a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).
   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$

   (c) Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$

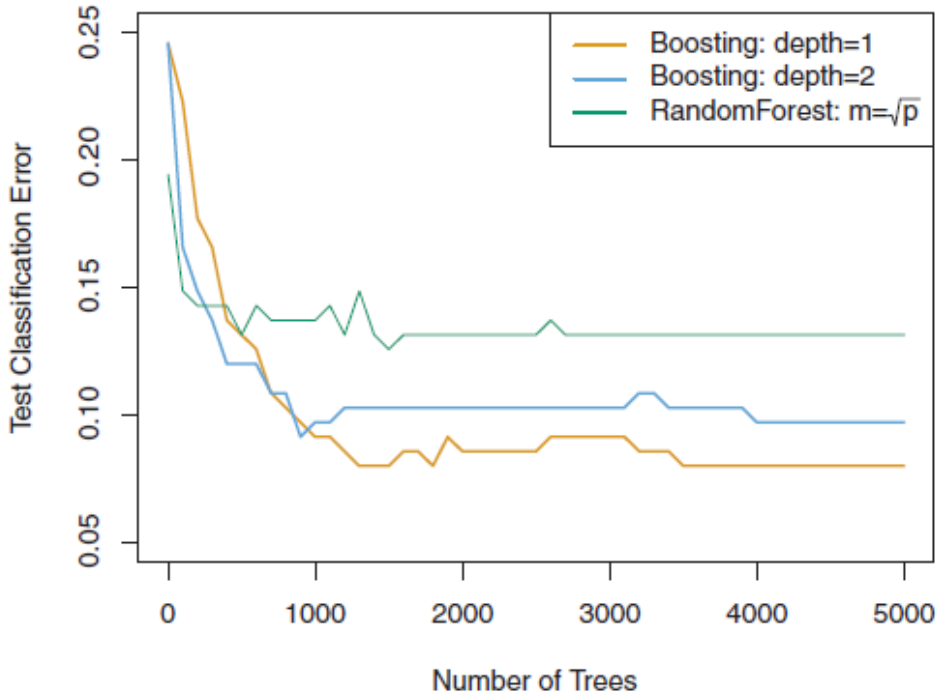
Recursively fit residuals.

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

Boosting Parameters

- **B**: number of trees
  - Overfitting if B is too large (it is not for bootstrap).
  - Determined through cross-validation
- λ: shrinkage parameter
  - It determines the speed of learning, and small λ requires large B.
  - Typical values are 0.01 or 0.001.
- **d**: number of splits in each tree
  - Often selected as 1, the simplest tree.



Practice

- Read data09_diabetes.csv and make the binary outcome by cutting Y at 140, as like the below code

```
df = pd.read_csv('data09_diabetes.csv')
df_data = df.iloc[:,:-1]
df_target = (df['Y']>140).factorize()[0]
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(
        df_data,df_target,test_size=0.33,random_state=0)
```

  - (1) Apply Bagging KNN Classifier, and find the optimal K (# of neighbors) via 5-fold cross-validation
  - (2) Apply Random Forest, and find the optimal tree size via OOB scores
  - (3) Apply Boosting Tree, and find the optimal number of tree estimators via OBB scores

Support Vector Machine

Support Vector Machine (SVM)

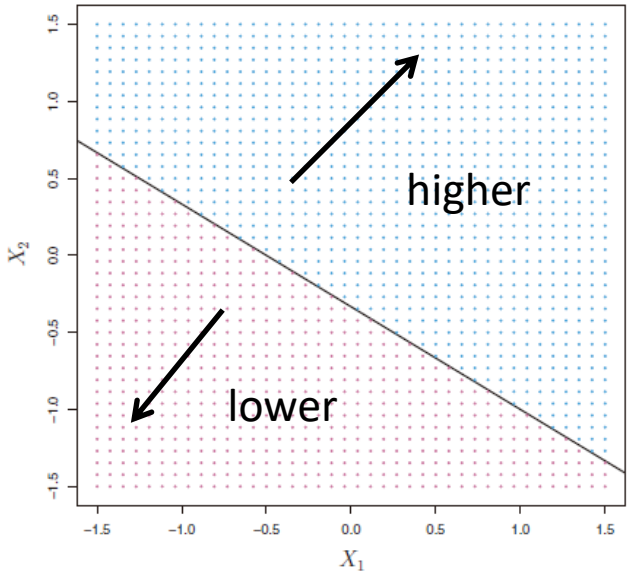
- The **support vector machine** is a classification approach that was developed in computer science in 1990's.

- It was a shock in machine learning community because of
  - High performance
  - Novel and mysterious mechanism
  - Marketing

- Scientists have developed the theoretical backgrounds of SVM.
  - SVM is categorized as *maximal margin classifier*.
  - It is closely related with other linear methods such as logistic regression.

Hyperplane

- A **hyperplane** in a *p*-dimensional space is a flat affine subspace of dimension *p*-1.



Black line: $1 + 2X_1 + 3X_2 = 0$
Red space: $1 + 2X_1 + 3X_2 < 0$
Blue space: $1 + 2X_1 + 3X_2 > 0$

- In general, a hyperplane is defined by

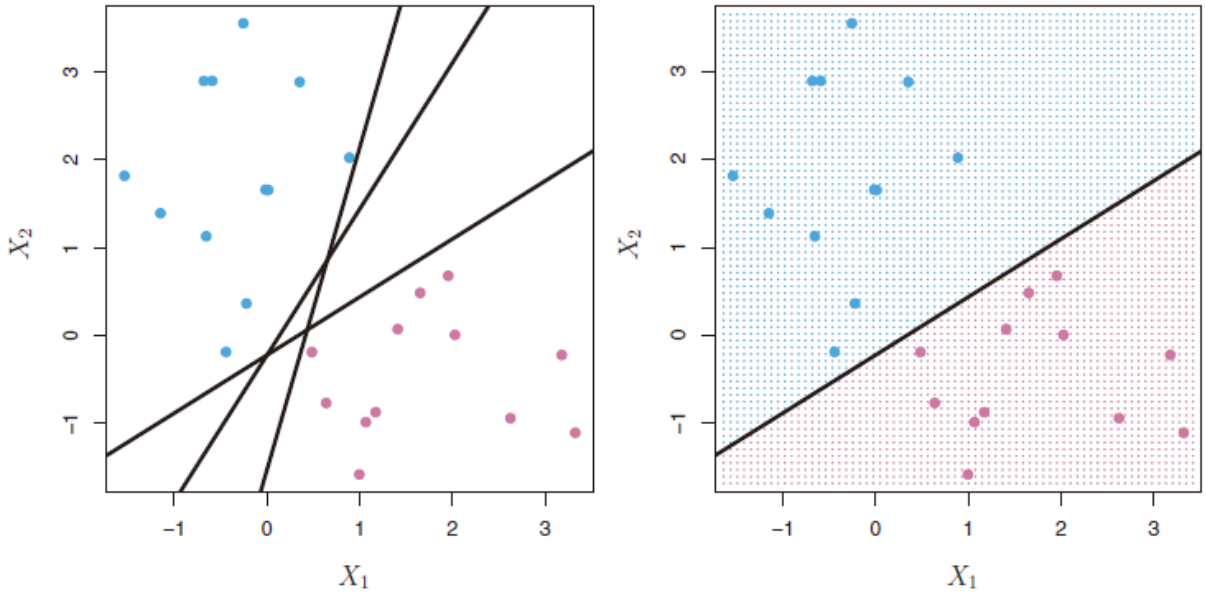
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- And divide the whole space into two subspaces by

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0.$$
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0,$$

Separating Hyperplane

- A hyperplane can separate two-class of observations in a *p*-dimensional space (or prediction with *p* predictors).



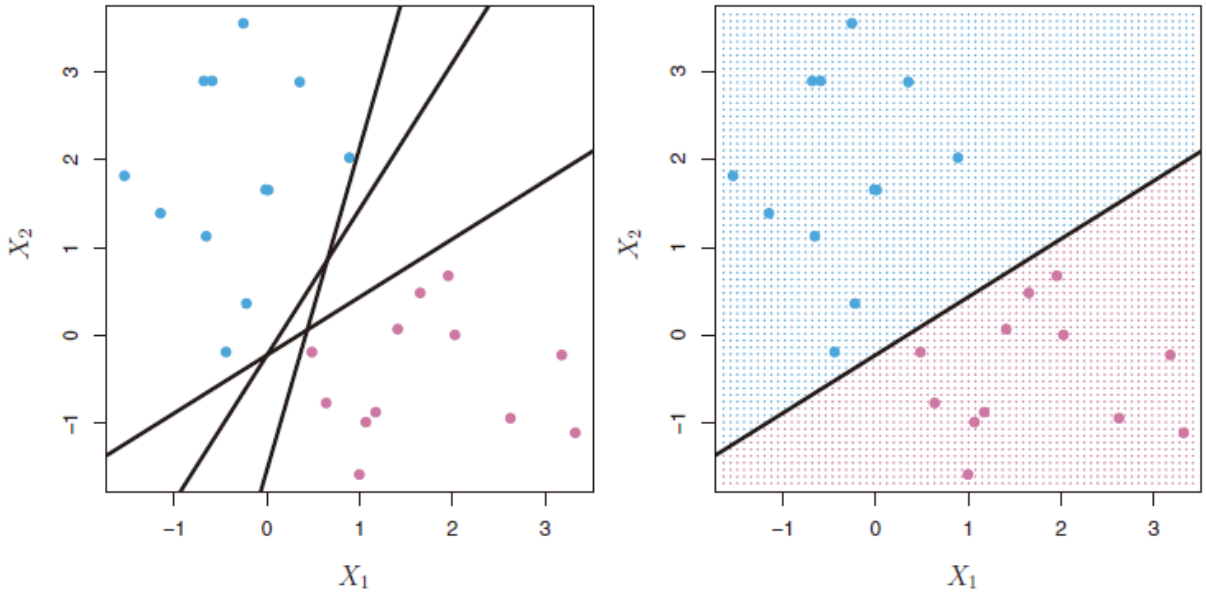
- When we indicate one class with $y = 1$, and the other class is $y = -1$, then a **separating hyperplane** satisfies

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

- **Margin** $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip})$
  - Points with high margins are far from the hyperplane.

Separating Hyperplane

- A hyperplane can separate two-class of observations in a p-dimensional space (or prediction with p predictors).



- When we indicate one class with $y = 1$, and the other class is $y = -1$, then a **separating hyperplane** satisfies

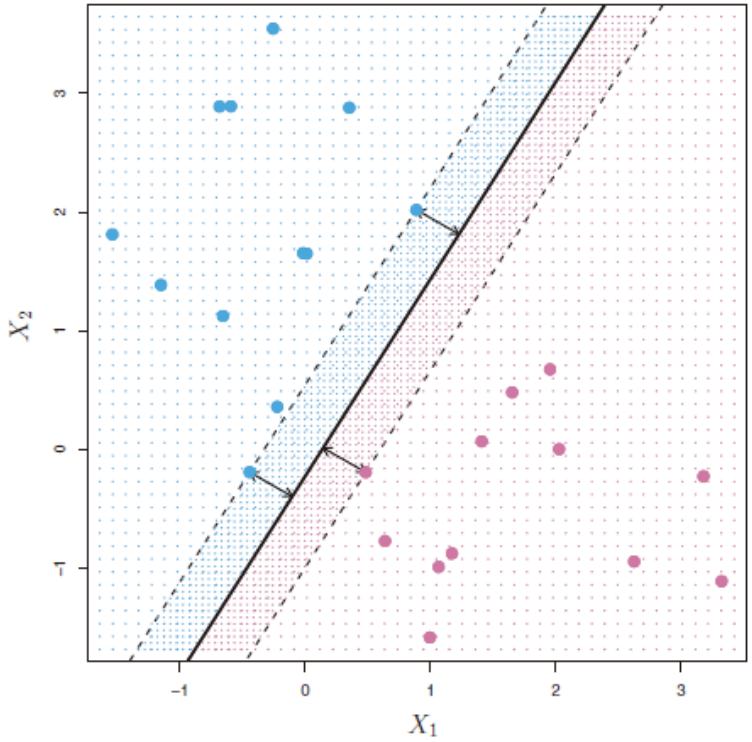
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

- There can be many separating hyperplane.

Maximal Margin Classifier

- **Margin**: the distance from the hyperplane to the closest point.
  - The perpendicular distance from the hyperplane to a point is calculated by $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip})$.

- **Maximal margin classifier**: classifying the class by the hyperplane with the maximal margin.



Maximal Margin Classifier

- The maximal margin classifier can be found through

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

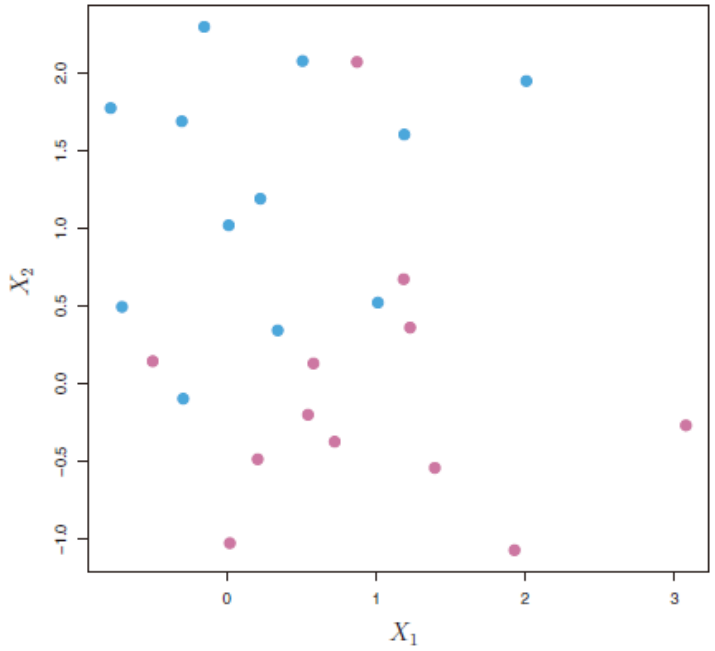
$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1, \longrightarrow$$

Not a real constraint, just scaling β's

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

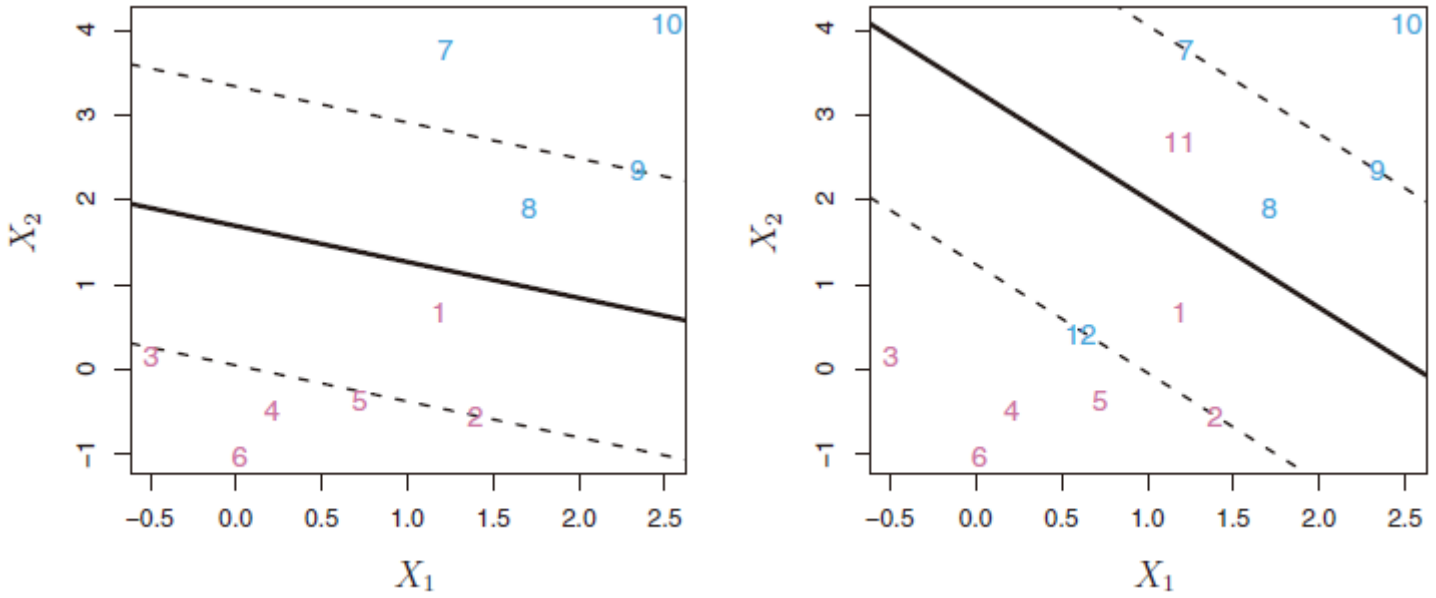
- Here we say the closest points to the separating hyperplane *support* the classifier.
- **Support vector**: the closest points to the separating hyperplane.

- Maximal margin classifier does not exist if two classes cannot be separable.



Support Vector Classifier

- **Support vector classifier**: the maximal margin classifier with soft margin.
  - It allows some samples are close to the hyperplane than the margin.
  - Better robustness and classification for most of the samples.



Support Vector Classifier

- The support vector classifier can be found by

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

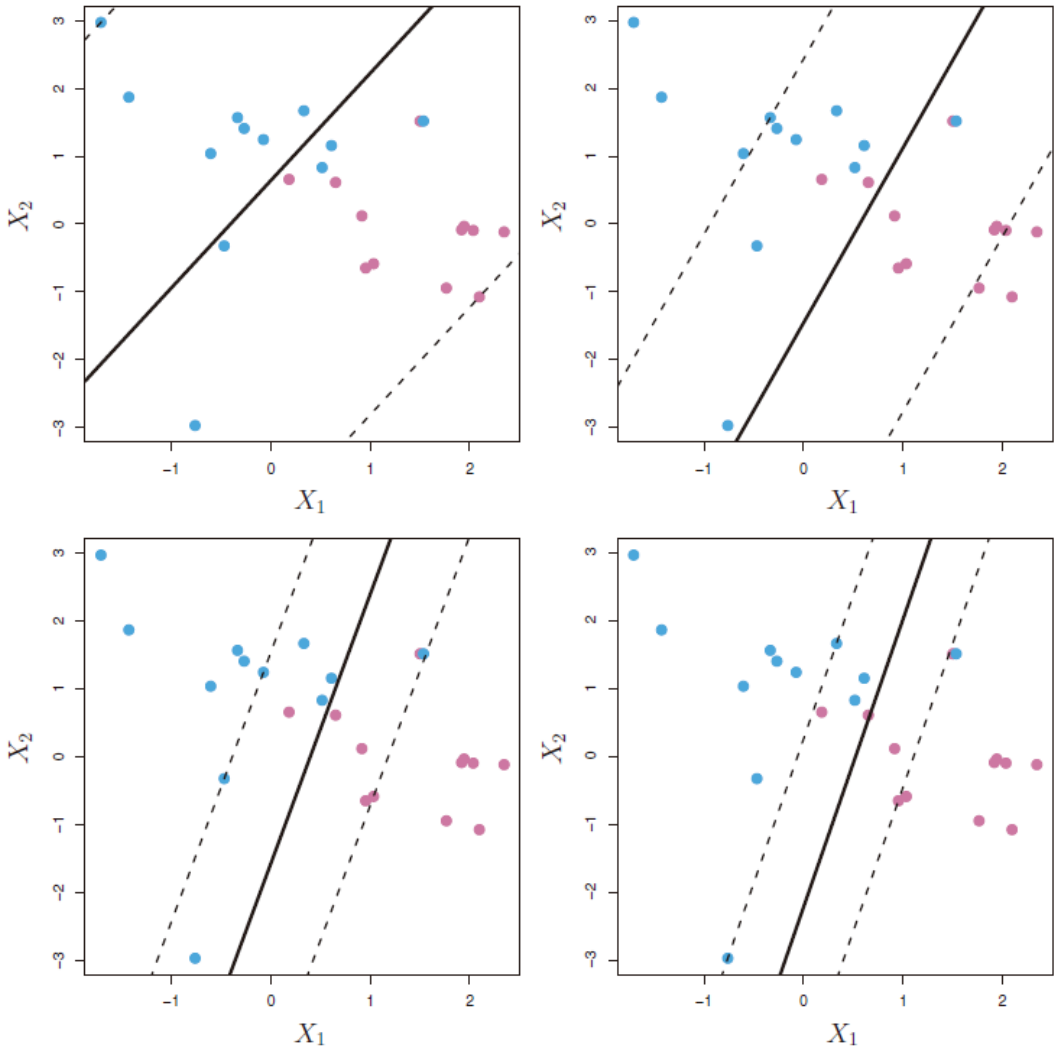
Soft margin

A tuning parameter that controls the amount of violation.

- Support vectors: samples on the margin boundary or on the wrong side.
  - Only support vectors determines the decision boundary. The other samples does not matter.

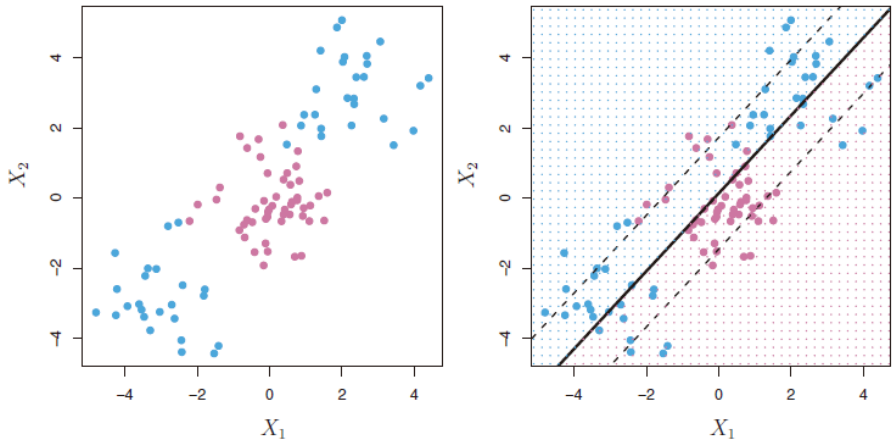
Support Vector Classifier

- C controls the bias-variance tradeoff.
  - Large C: wider margin, high bias and low variance.
  - Small C: narrower margin, low bias and high variance.



Non-linear Decision Boundary

- Support vector classifier provides only linear decision boundary, which sometimes fail.



- One solution is including high order terms such as $X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$.

$$\underset{\beta_0,\beta_{11},\beta_{12}\ldots,\beta_{p1},\beta_{p2},\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

$$\text{subject to } y_i \left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2 \right) \geq M(1-\epsilon_i),$$

$$\sum_{i=1}^{n} \epsilon_i \leq C, \quad \epsilon_i \geq 0, \quad \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1.$$

- Basically, we want to increase (or transform) the feature space through a **kernel**.

Support Vector Machine

- **Support vector machine** (SVM): support vector classifier with the use of kernels.

- Believe it or not,

$$\begin{aligned}
&\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M \\
&\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1, \\
&y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \\
&\epsilon_i \geq 0, \ \sum_{i=1}^{n} \epsilon_i \leq C,
\end{aligned}$$

  is equivalent to maximize

$$\tilde{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

- The inner product $x_i^T x_j =< x_i, x_j >= \sum_{k=1}^{p} x_{ik} x_{jk}$ is essential for linear boundary.
- Kernel expands the inner product with a kernel function $K(x_i,x_j)$.

Kernels

Linear Kernel

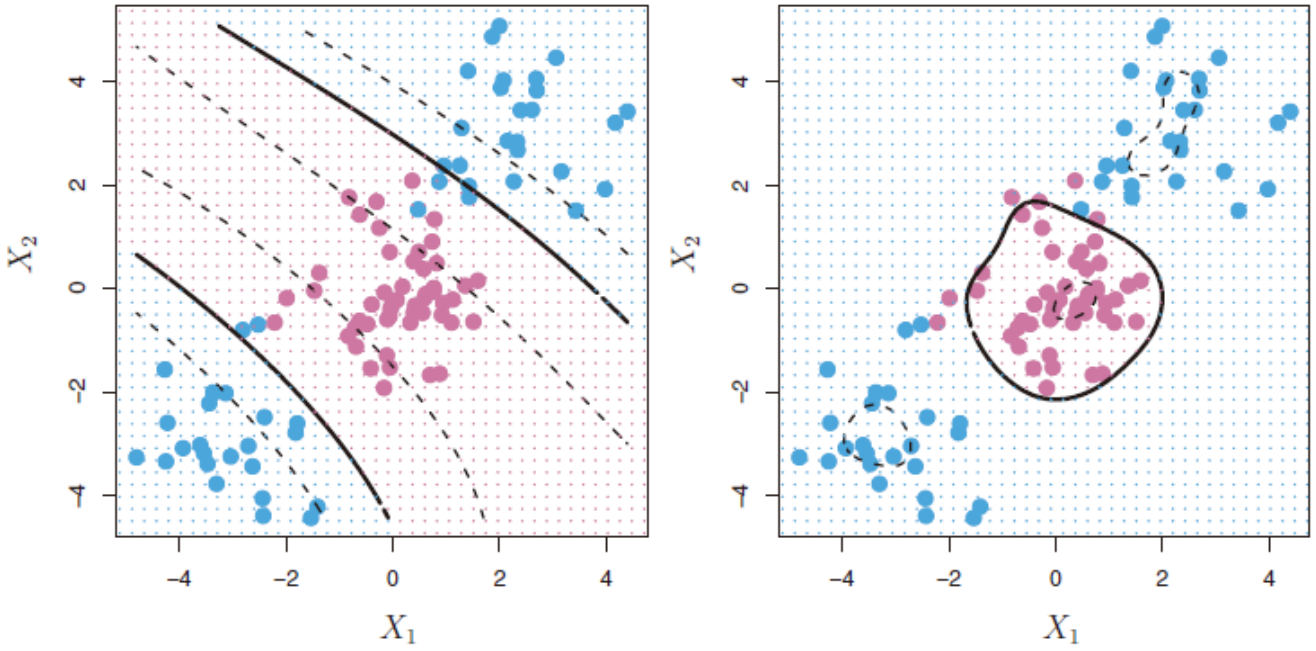
$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j},$$

Polynomial Kernel (d: tuning)

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d.$$

Radial (Gaussian) Kernel (λ: tuning)

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2).$$



Extension of SVM

- Prediction for K>2 classes
  - One-Versus-One: we build ${}_K\mathrm{C}_2$ pairwise predictors that compare all pairs of classes. A sample is predicted to the most frequently predicted class with the ${}_K\mathrm{C}_2$ predictors.
  - One-Versus-All: for each class, we build a predictor that compares the given class and all others. Then, we have K predictors. A sample is predicted to the class of which calculated distance from the hyperplane is the largest.

- Support vector regression
  - Linear regression inspired by support vector machine.
  - Minimize the residuals only larger than a certain value.

Practice

- SVM in Python
  - sklearn.svm.SVC
  - sklearn.svm.SVR

- Practice
  - Read data10_khan.csv file
  - Use the first 63 samples as the training set, and the rest as the test set
  - The first column is the output variable (breast cancer grade), and the rest columns are input variables (gene expression)
  - Using the support vector machine, classify cancer grades. What is your best performance? What is the corresponding kernel function and C? What is the misclassified samples?
  - Using the support vector regression, regress cancer grades. What is your best performance? What is the corresponding kernel function and C?

Neural Network

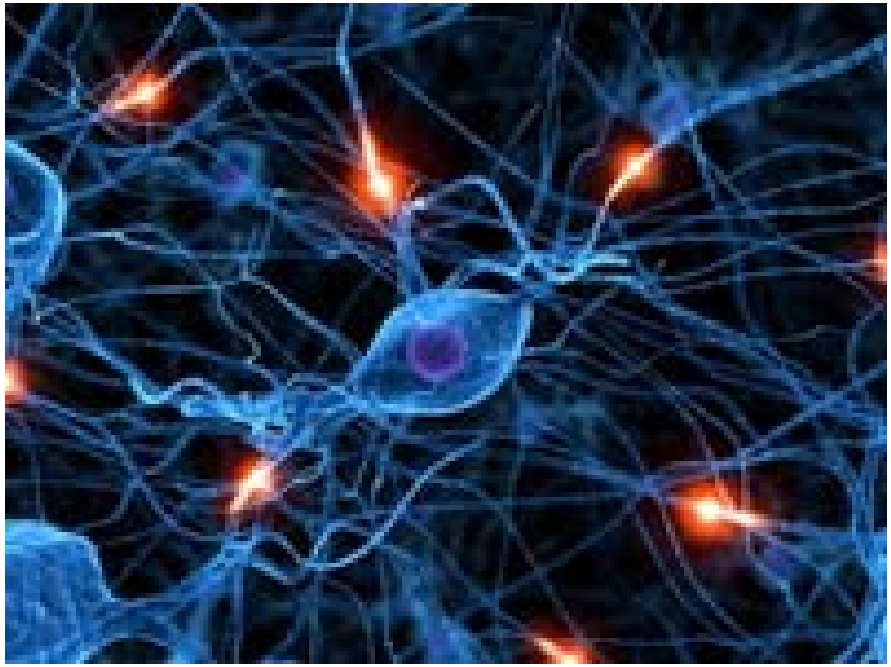
The Idea of Deep Learning

Easy for Human, Difficult for Machines

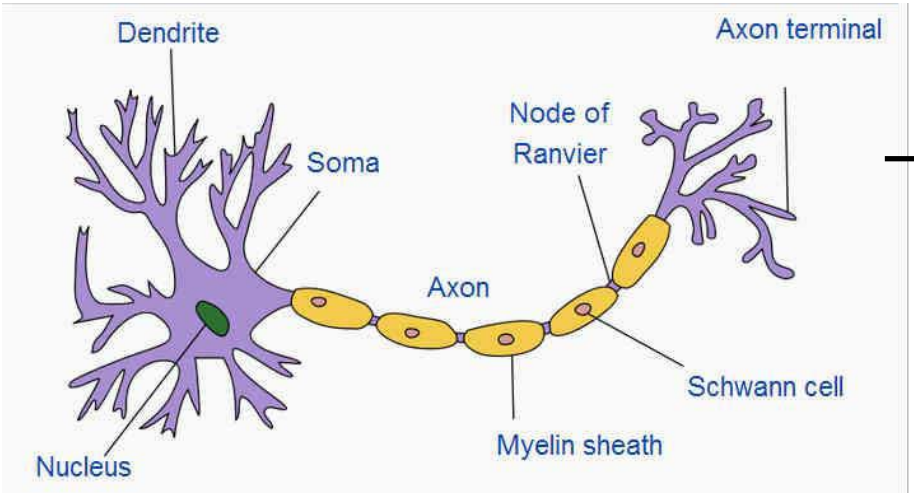
- Image classification
- Speech recognition
- etc..

Build learning algorithms that mimic human brain

Human Brain as a Network of Neurons



Stimulated ⟶

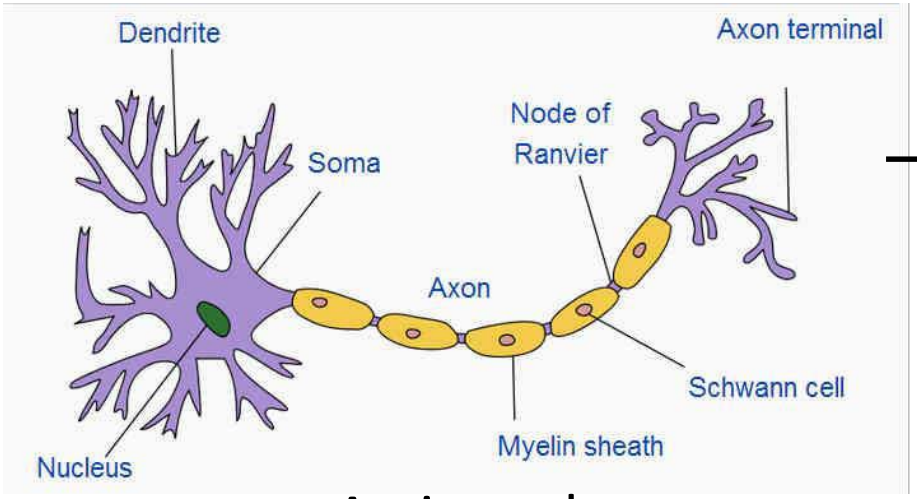


⟶ Chemicals

Activated

Perceptron: Mathematical Modeling of a Neuron

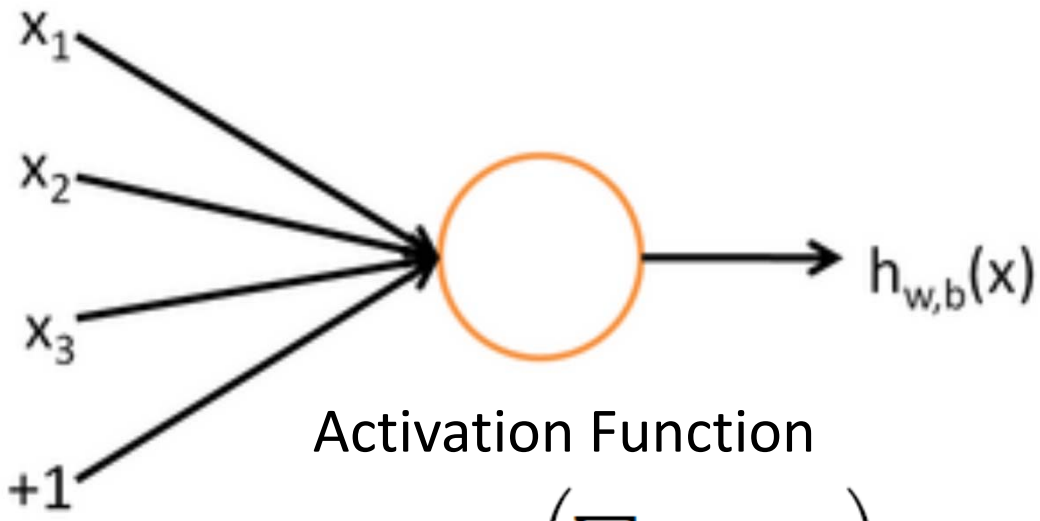
Stimulated →



→ Chemicals

Activated

Input



Output

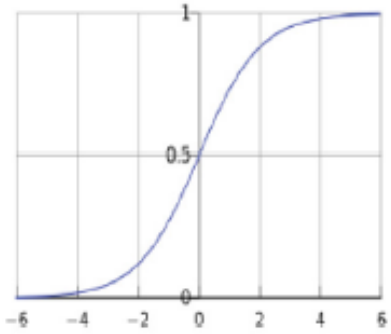
Activation Function

$$h_{w,b}(\boldsymbol{x}) = f\left(\sum x_i w_i + b\right)$$

Activation Function

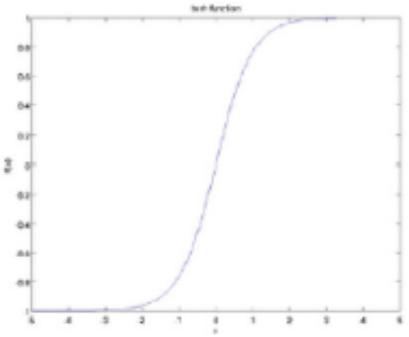
logistic ("sigmoid")

$$f(z) = \frac{1}{1 + \exp(-z)}.$$



tanh

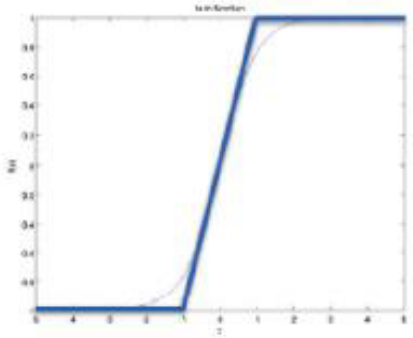
$$f(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}},$$



Function families that can switch on

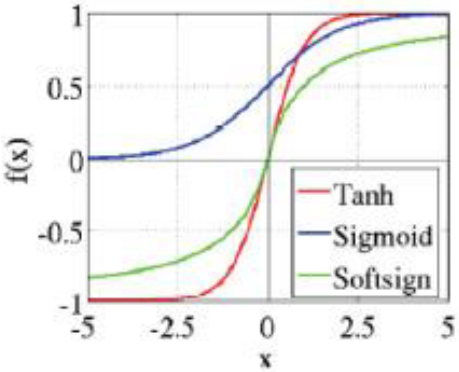
hard tanh

$$\text{HardTanh}(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } -1 <= x <= 1 \\ 1 & \text{if } x > 1 \end{cases}$$



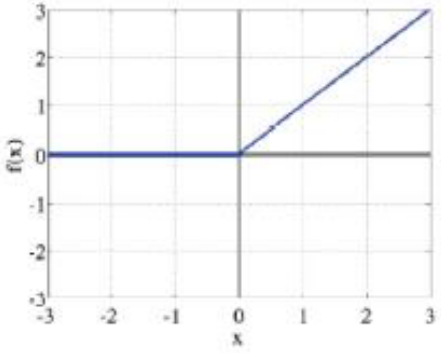
soft sign

$$\text{softsign}(z) = \frac{a}{1 + |a|}$$



rectifier

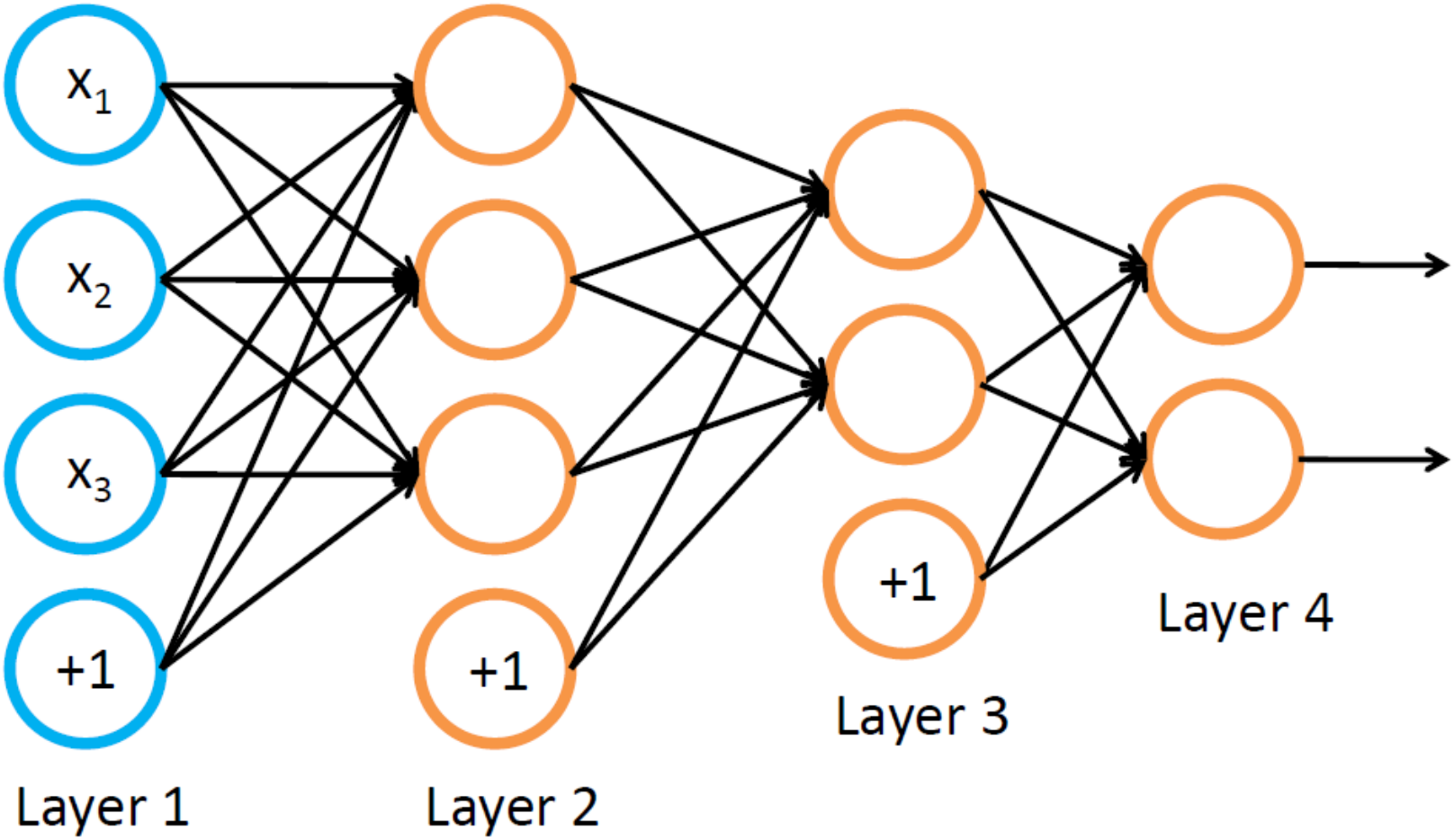
$$\text{rect}(z) = \max(z, 0)$$



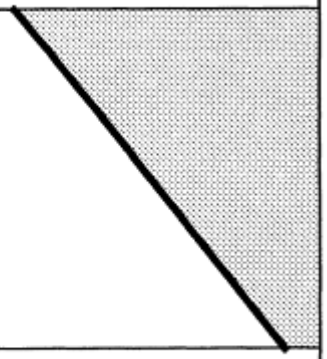
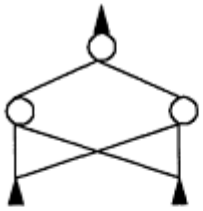
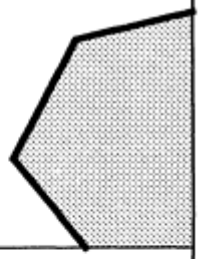
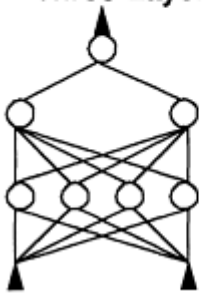
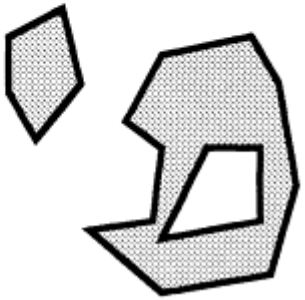
Neural Network for an Artificial Brain

ANN: Artificial Neural Network

Multi-layered Perceptron



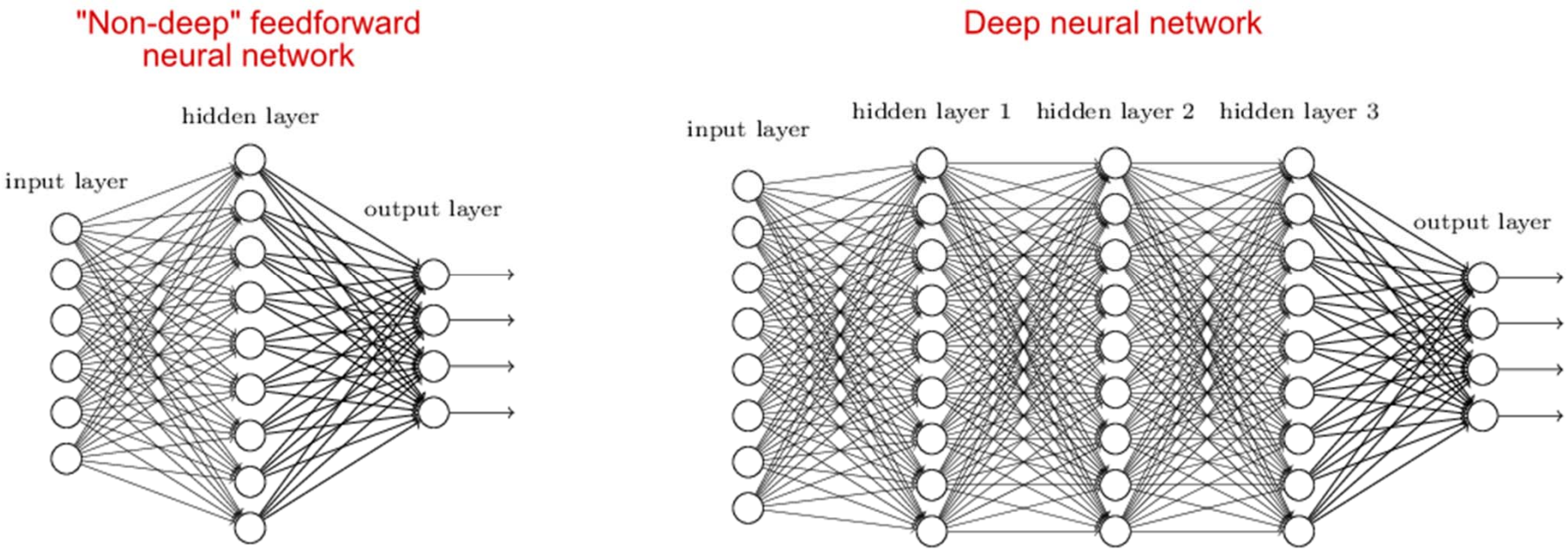
Complexity

Structure	Types of Decision Regions	Most General Region Shapes
Single-Layer 	Half-Plane Bounded by A Hyper-Plane	
Two-Layer 	Convex Open, or Closed Regions	
Three-Layer 	Arbitrary (Complexity Limited by the Number of Nodes)	

More Layers, More Complex

More Layers = Deeper

DNN: Deep Neural Network

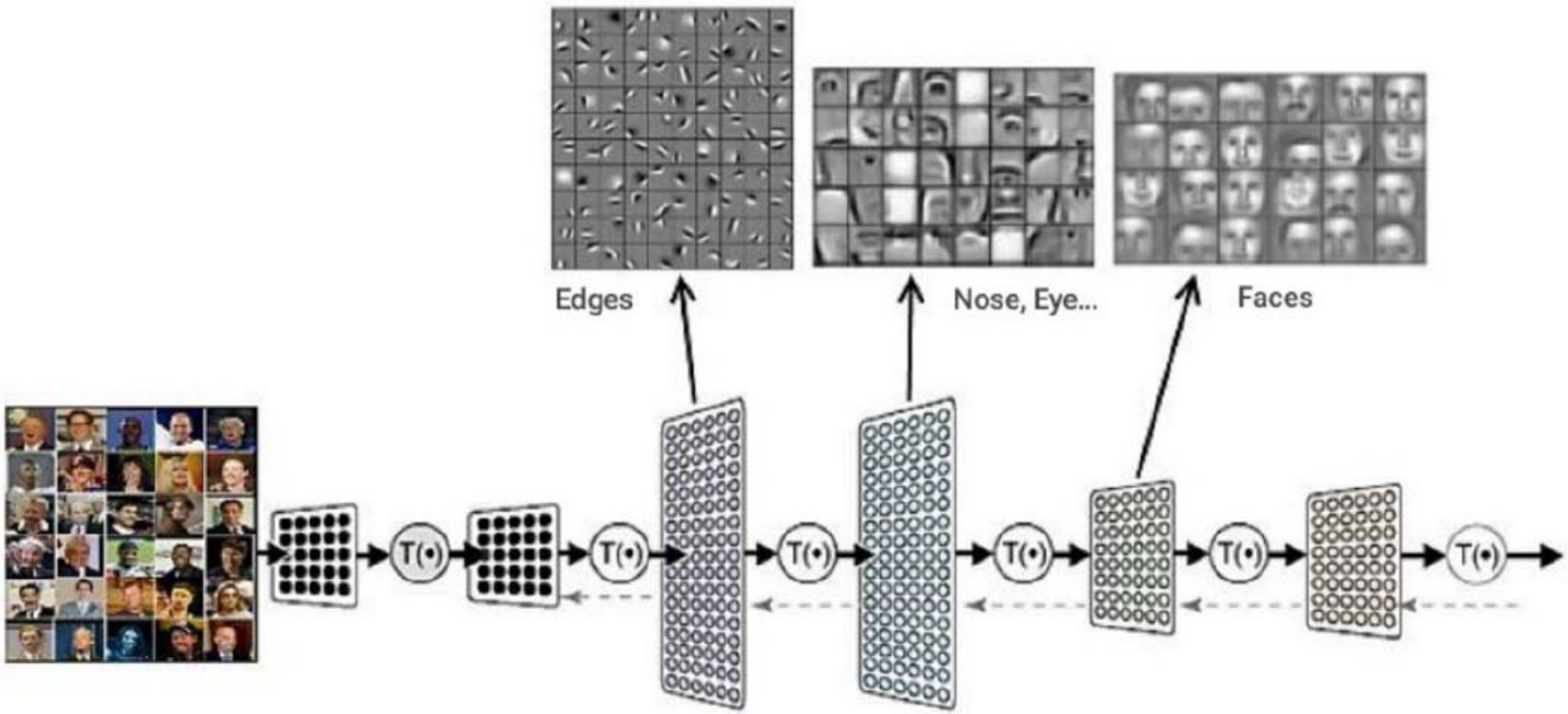


More layers for more complicated problems
Human brain also uses multiple layers of neurons

Deep Learning with multiple hidden layers

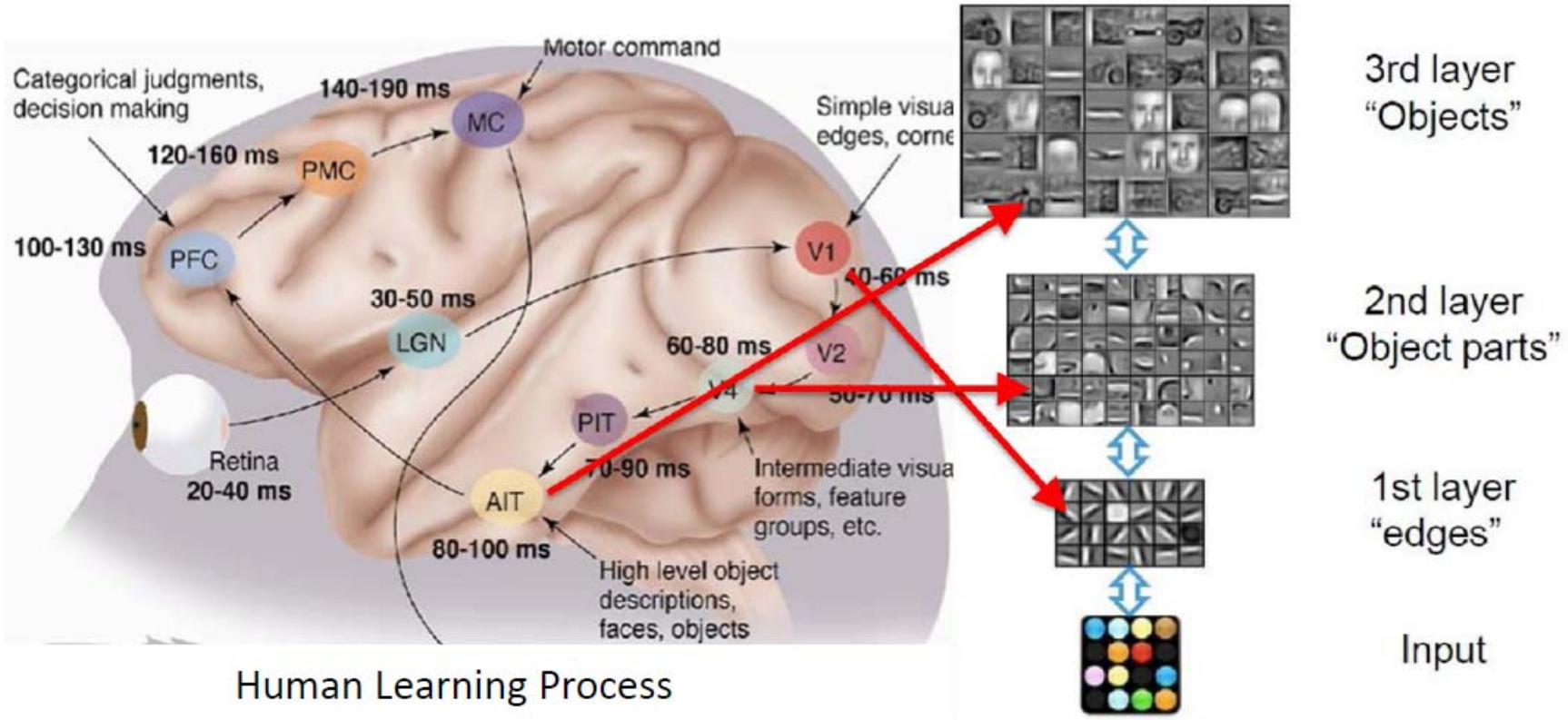
Shallow Learning with one or no hidden layers

DNN: Deep Neural Network



Deeper layer present more abstracted objects
Human brain also uses different abstractions

DNN vs. Human Brain

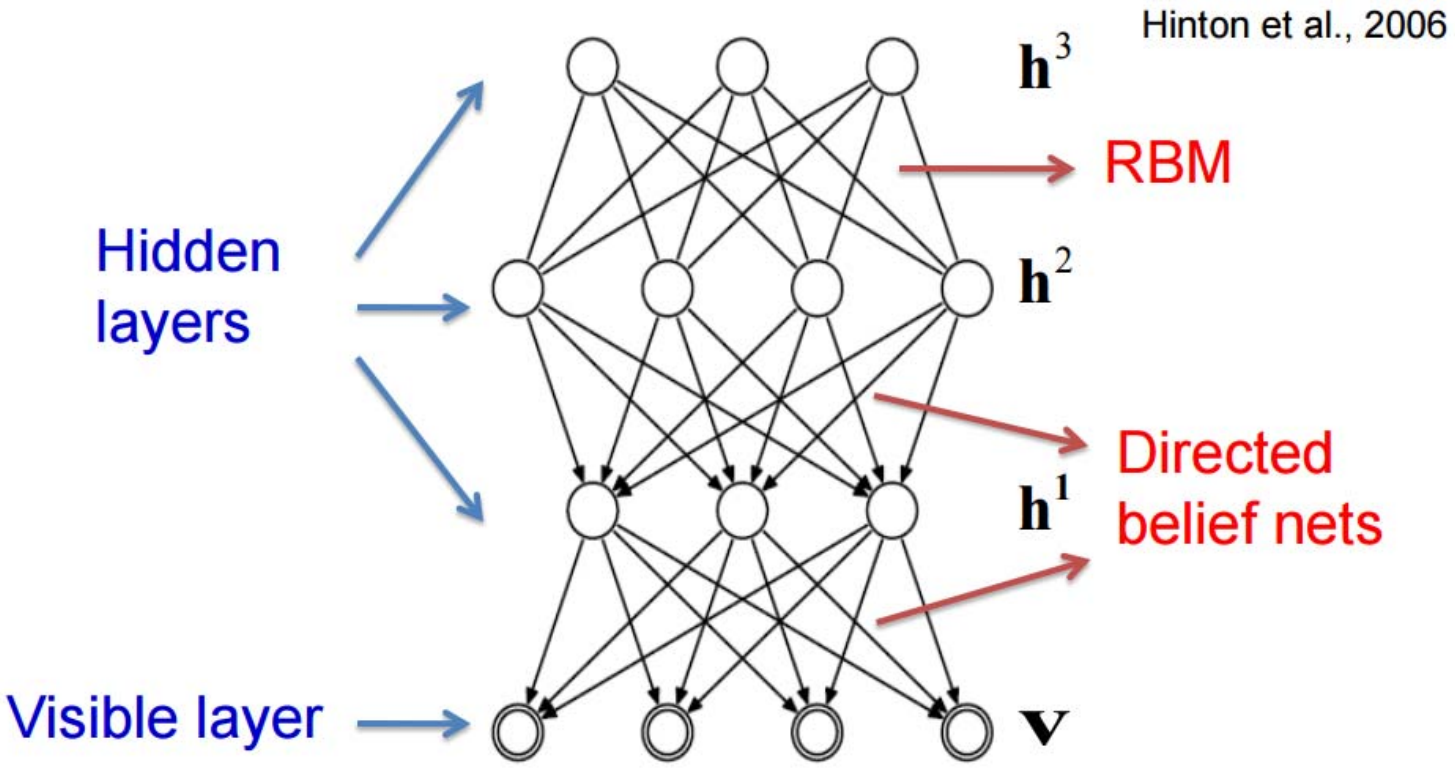


Human Learning Process

Deep Learning

DBN: Deep Belief Network

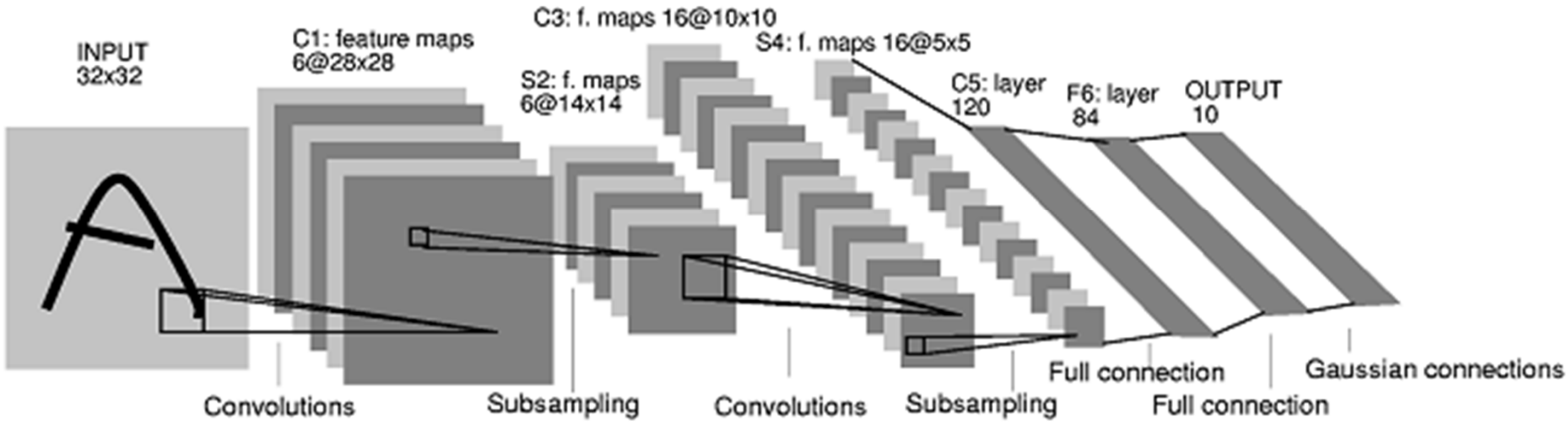
- **Pre-training**: initializing the network using untagged data
  - Multi-layered Restricted Boltzmann Machine (RBM)
- Extracting good features in an unsupervised way
- Supervised fine-tuning



$$P(\mathbf{v},\mathbf{h}^1,\mathbf{h}^2,\ldots,\mathbf{h}^l) = P(\mathbf{v}\,|\,\mathbf{h}^1)P(\mathbf{h}^1\,|\,\mathbf{h}^2)\ldots P(\mathbf{h}^{l-2}\,|\,\mathbf{h}^{l-1})P(\mathbf{h}^{l-1},\mathbf{h}^l)$$

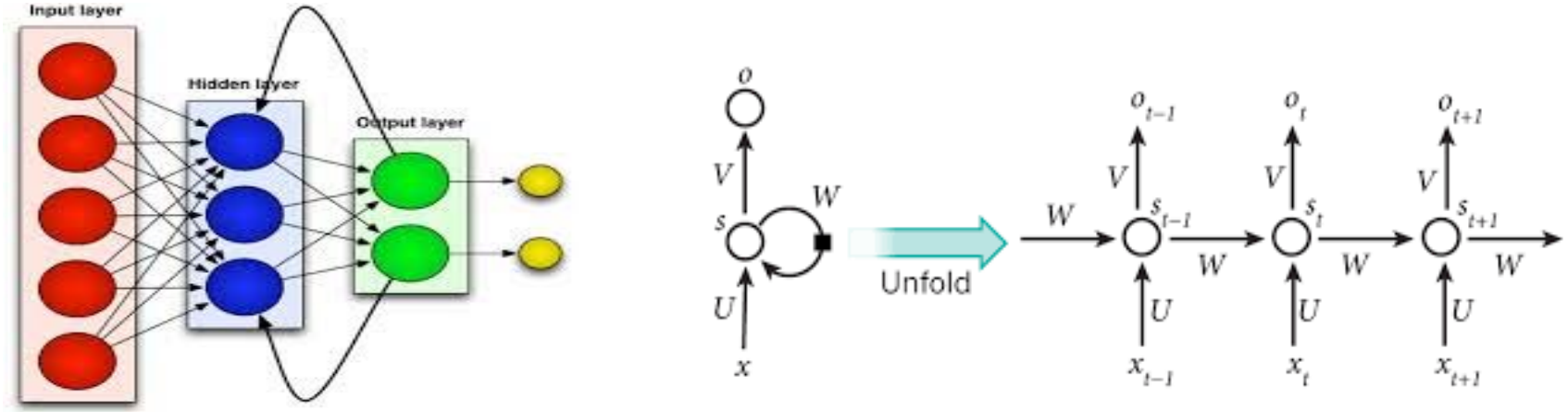
CNN: Convolutional Neural Network

- Convolution layer
  - Overlapped subsets of data
- From many small neural networks to one fully connected network
  - Often requiring subsampling to reduce the computation
- Recognizing spatial information (neighboring)



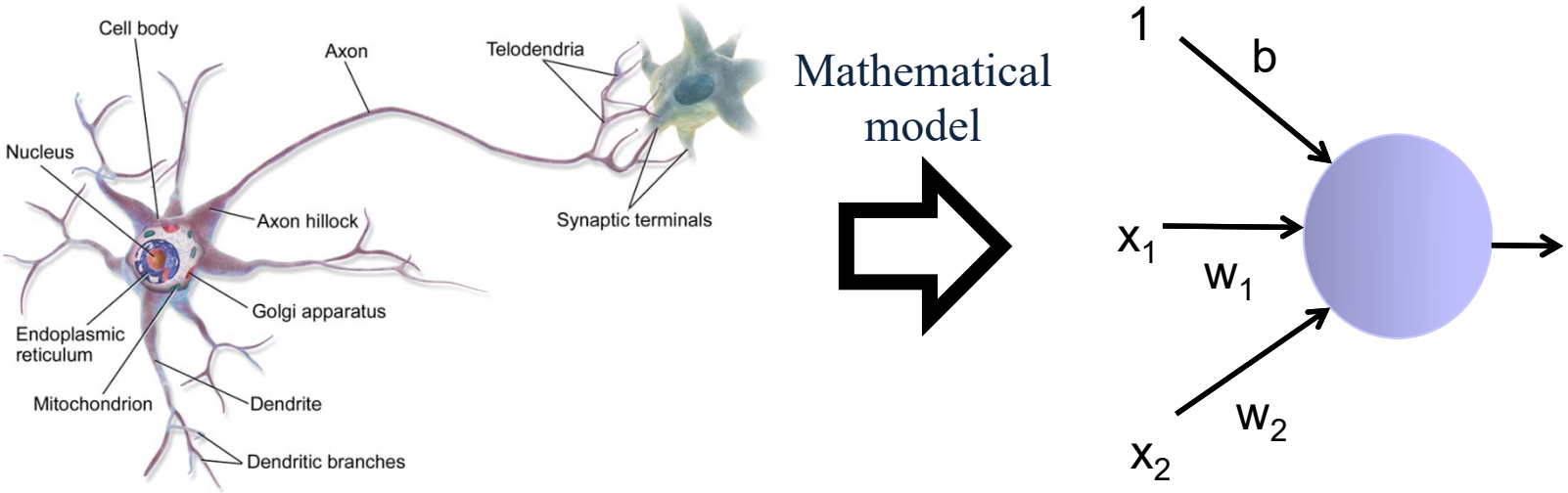
RNN: Recurrent Neural Network

- NNs with recurrent inputs, i.e. feedback outputs as inputs
- Can be used for time-series data
  - Handwriting recognition
- Traditionally difficult to scale up
  - Often preferred for a small network with single layer, a few node, and low input dimension.



Artificial Neuron

- The artificial neuron is an answer for the question
- An artificial neuron outputs signal if the sum of inputs is above a certain threshold

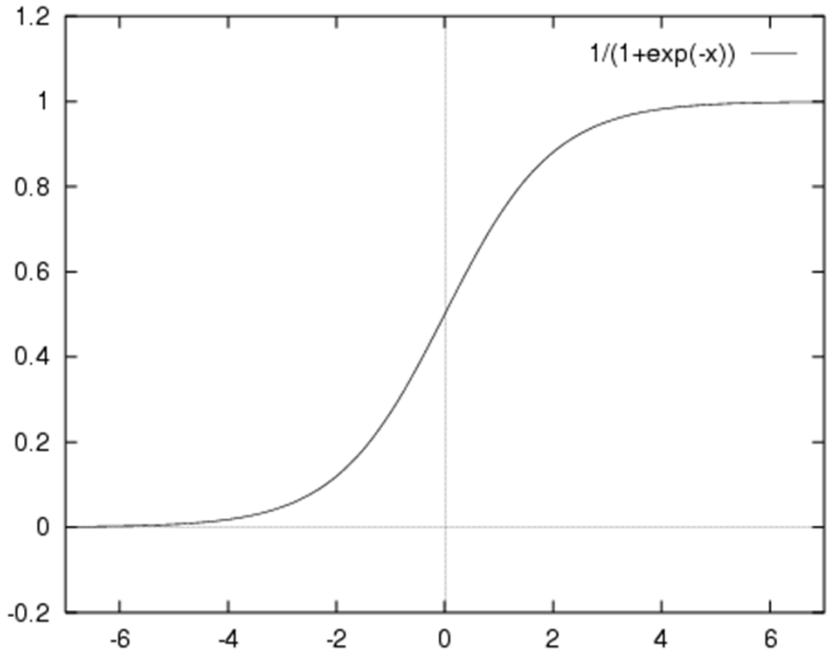


Activation function

- Differentiable activation function
  - For the backpropagation

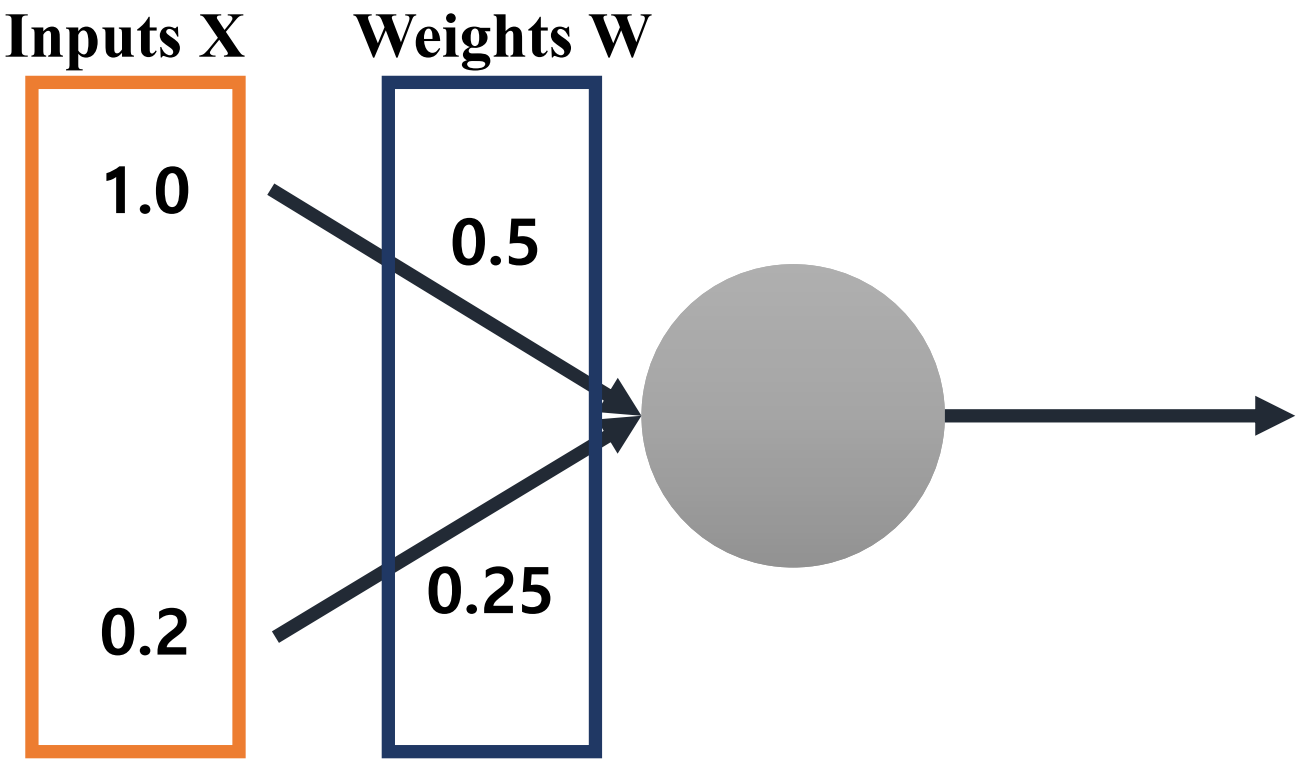
- Sigmoid function
  - It is similar to the threshold function
  - Converges to 0 and 1

$$f(x) = \frac{1}{1+e^{-x}}$$



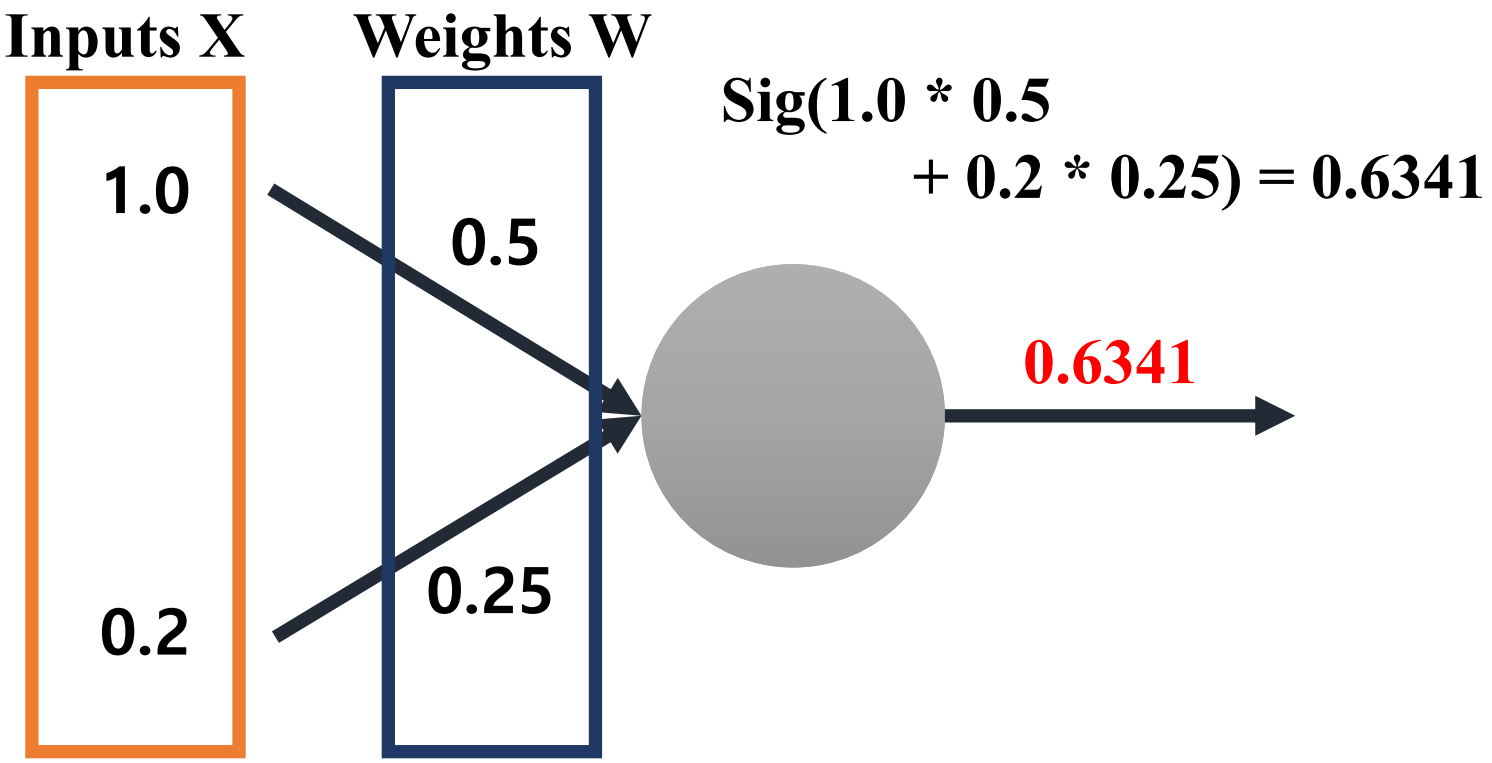
An example of feed forwarding

- F(x) = sigmoid(WX + b)



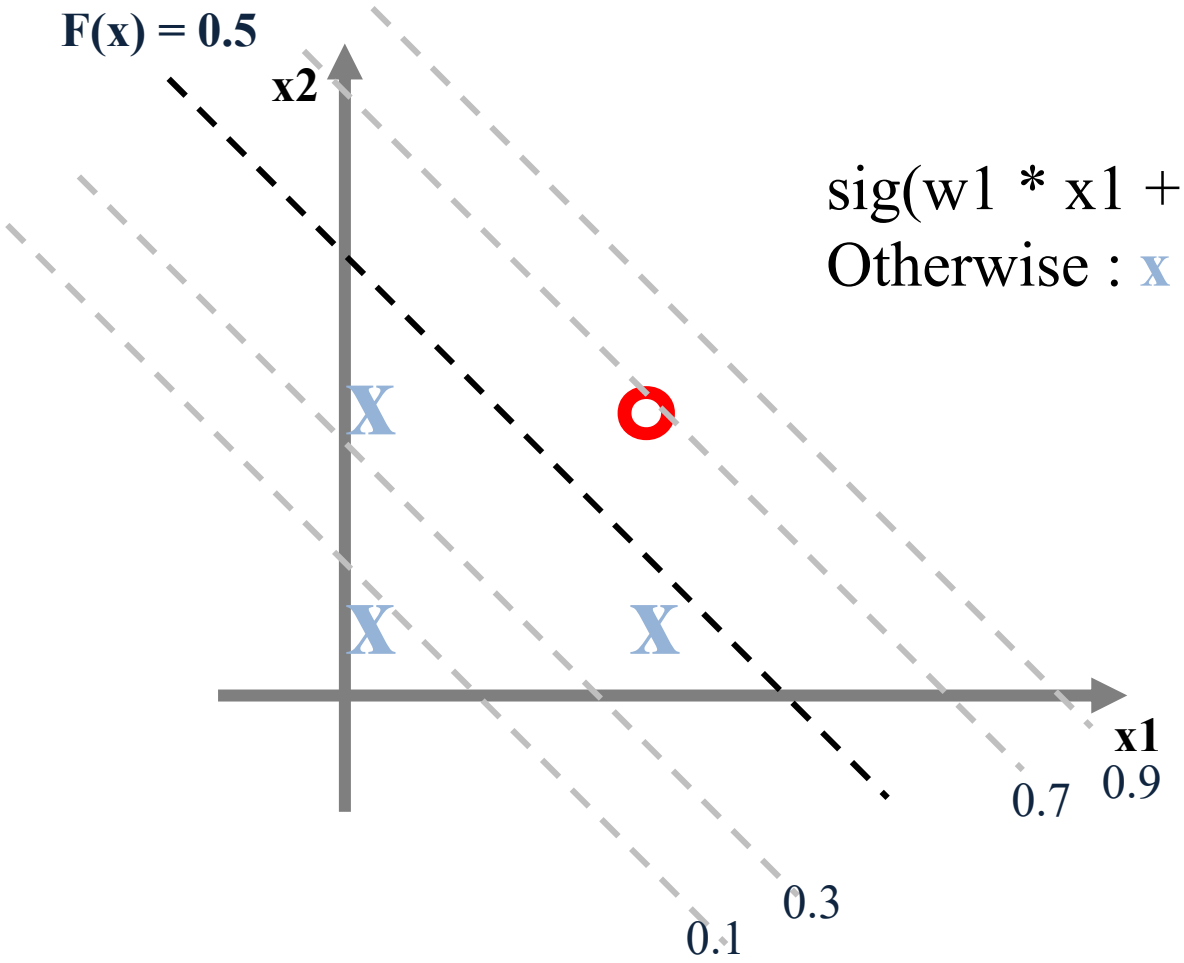
An example of feed forwarding

- F(x) = sigmoid(WX + b)



Linearity of the perceptron

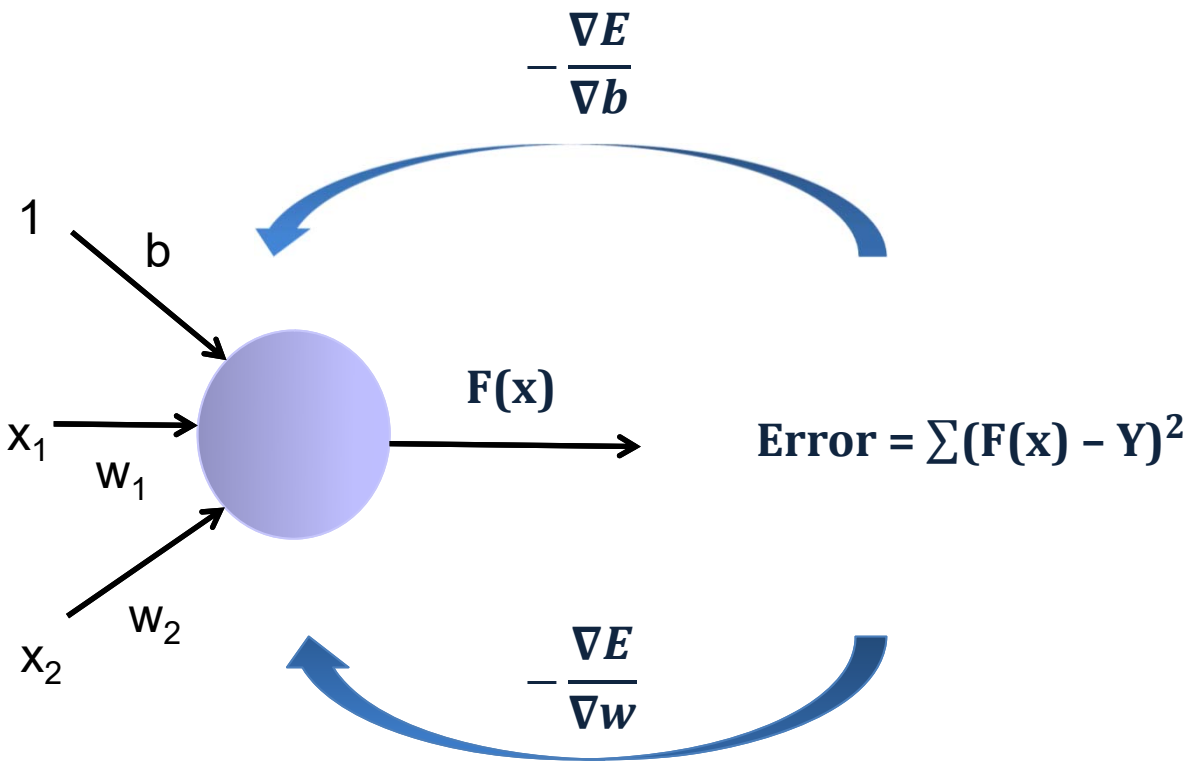
- In fact, the perceptron is a linear classifier
  - Therefore, the learning a perceptron corresponds to finding a good set of weights (e.g. w1, w2 and b in the figure)



sig(w1 * x1 + w2 * x2 + b) > 0.5: o
Otherwise : x

Backpropagation

- Neural networks can be trained by an algorithm called 'backpropagation'
- Backpropagation works by back-propagating the gradient of error



Backpropagation

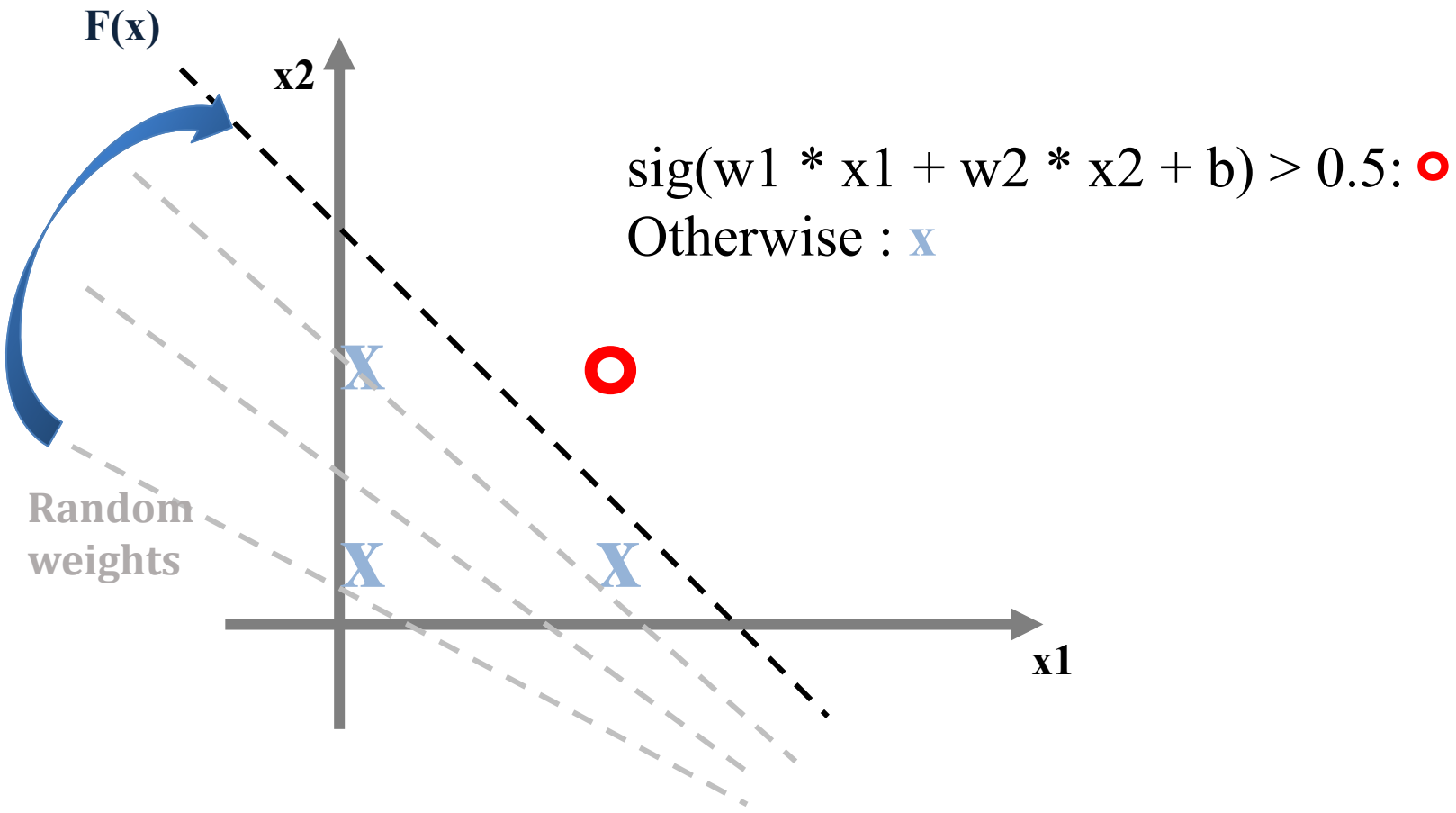
- New weights are given by the following formula:

$$W_{t+1} = W_t - \alpha \times \frac{\nabla(E)}{\nabla(W)}$$

- Where α is the learning rate which is usually set to a small value (e.g. 0.001)
- The weight matrix W_{t+M} converges with sufficient iteration

From random to optimal weights

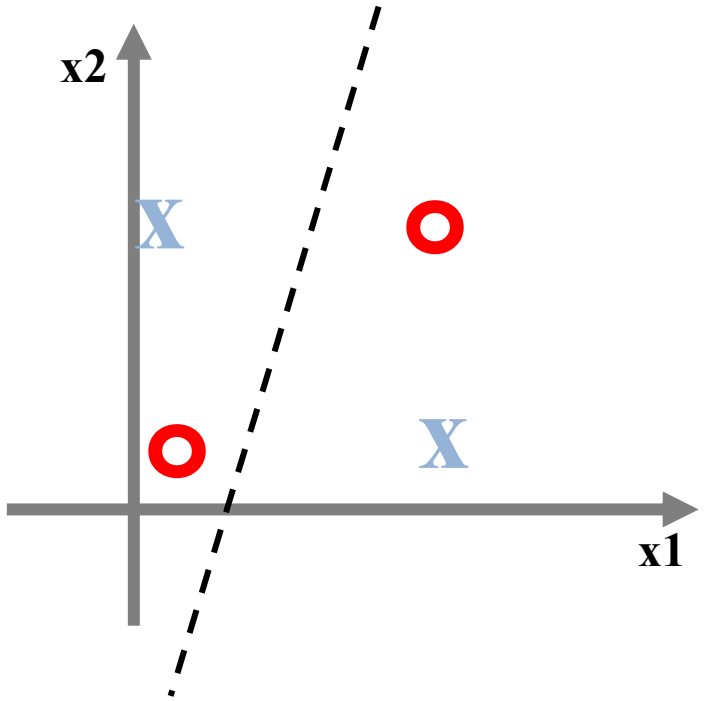
- In summary, the training of a perceptron corresponds to finding optimal weight for a given dataset by backpropagation algorithm
- The weights gradually move to the optimal point from random initialization



XOR Problem

- But, perceptron cannot solve the XOR problem because of its linear property
- Hence, there are many practical problems which cannot be solved with a line

XOR problem

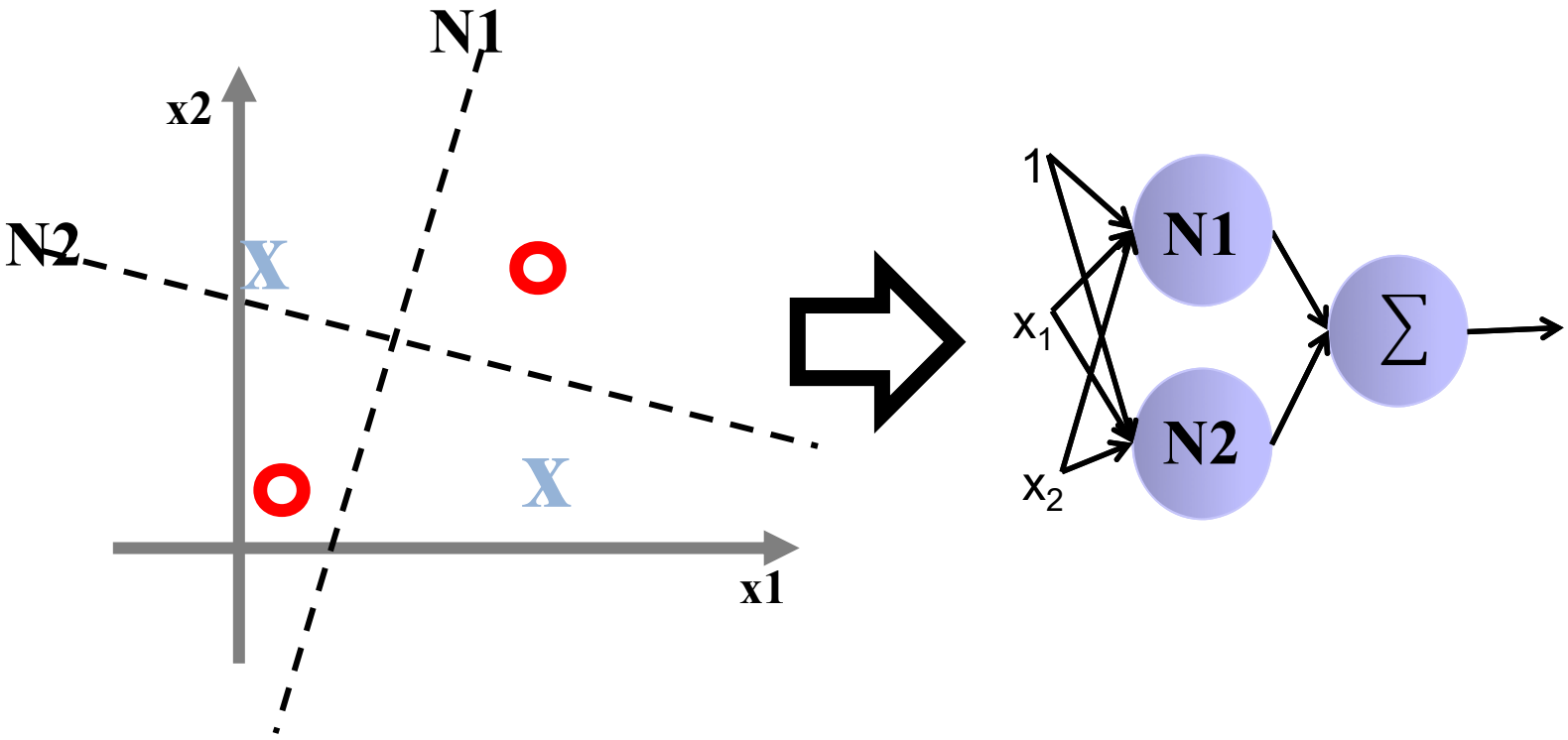


Input 1	Input 2	Output
1	1	0
1	0	1
0	1	1
0	0	0

Solution (MLP)

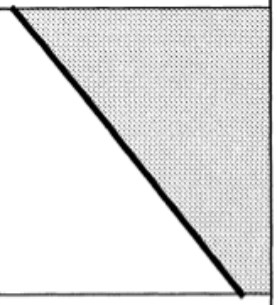
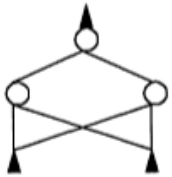
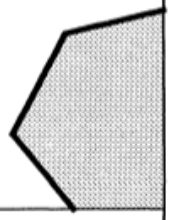
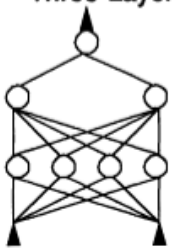
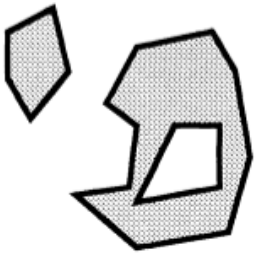
- However, the solution for the problem is very simple
- XOR problem can be solve with multiple lines

- And the multiple lines can be interpreted as multiple neurons
- Such models with multiple layers and neurons are called **Multi-Layer Perceptron (MLP)**



The meaning of deep and wide neural networks

- Complex region can be represented with deep and wide neural networks

Structure	Types of Decision Regions	Most General Region Shapes
Single-Layer 	Half-Plane Bounded by A Hyper-Plane	
Two-Layer 	Convex Open, or Closed Regions	
Three-Layer 	Arbitrary (Complexity Limited by the Number of Nodes)	

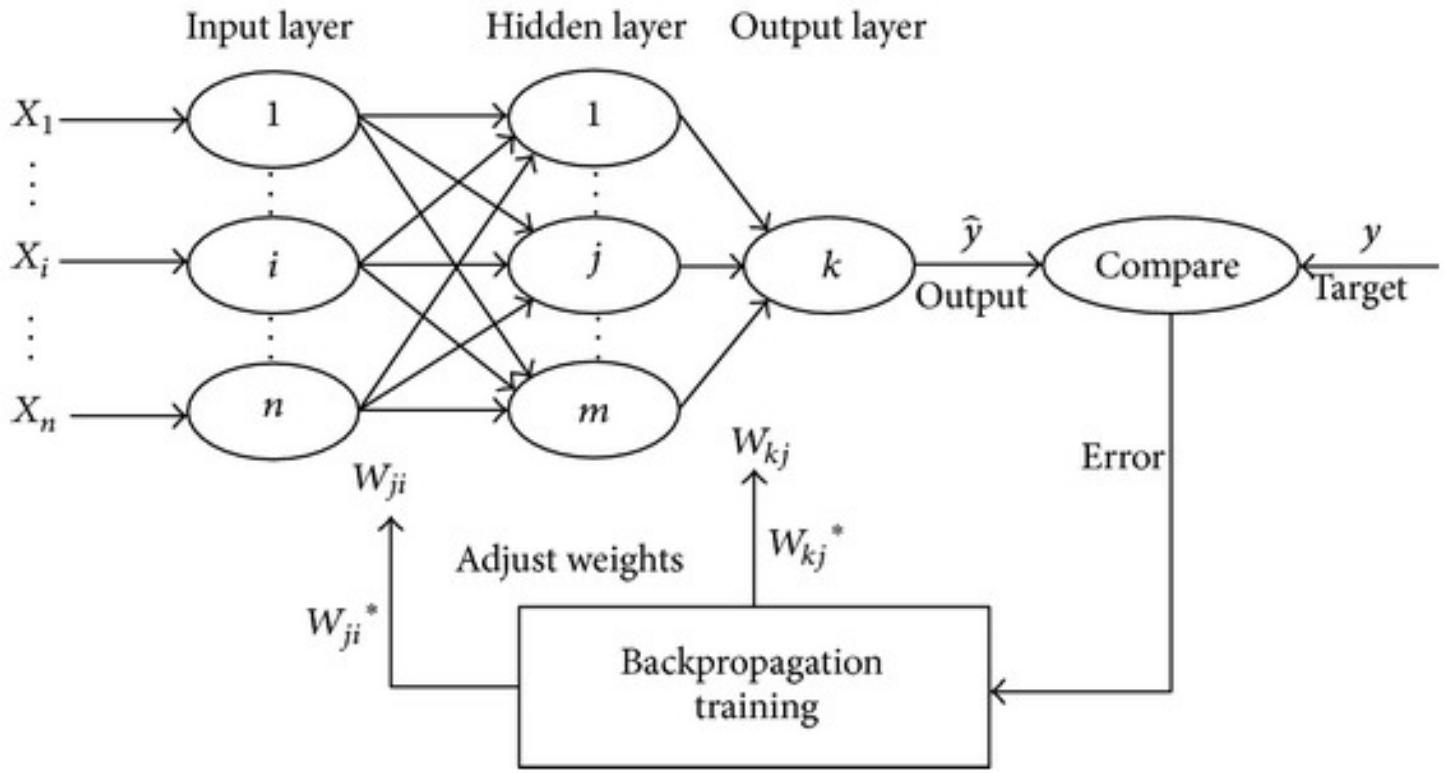
More layers

=

More Complex

Parameter Estimation

- Backpropagation

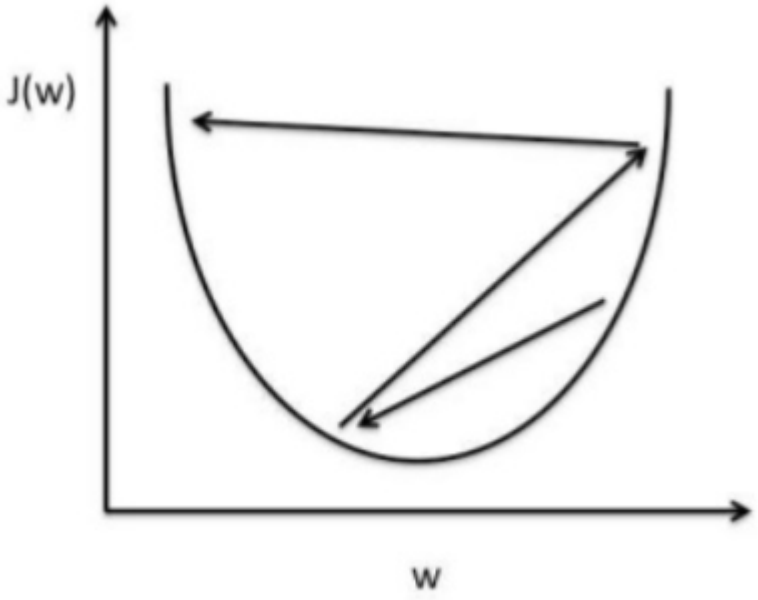


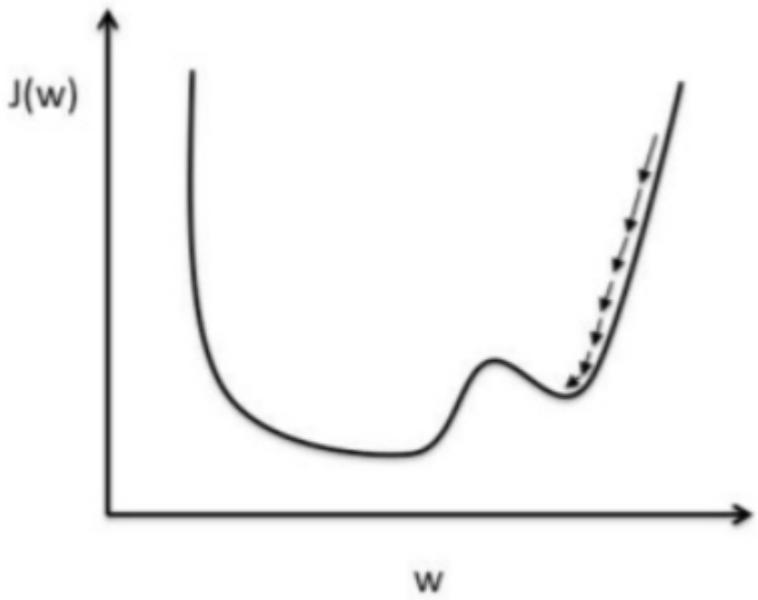
Parameter Estimation

- Stochastic Gradient Descent

$$\theta_t = \theta_{t-1} - \alpha \underbrace{\Delta_\theta L(\theta_{t-1})}_{\text{Evaluated on a mini-batch}}$$

- Learning rate





Practice

- Neural network in Python
  - Sklearn.neural_network.MLPClassifier
  - Sklearn.neural_network.MLPRegressor

- Exercise
  - Read data10_khan.csv file
  - Use the first 63 samples as the training set, and the rest as the test set
  - The first column is the output variable (breast cancer grade), and the rest columns are input variables (gene expression)
  - Using the MLP classifier, classify cancer grades. What is your best performance? What is the corresponding NN structure and options?
  - Using the MLP regressor, regress cancer grades. What is your best performance? What is the corresponding NN structure and options?

Non-linear Regression

Beyond Linearity

- Many traditional (and modern) methods are based on linearity.
  - Easy to calculate, develop the theory, and interpret the results.
- However, in the real world nothing is linear!
- Recently development of computational power enables non-linear methods.

- Polynomial regression
- Step functions
- Regression splines
- Smoothing splines
- Local regression
- Generalized additive model

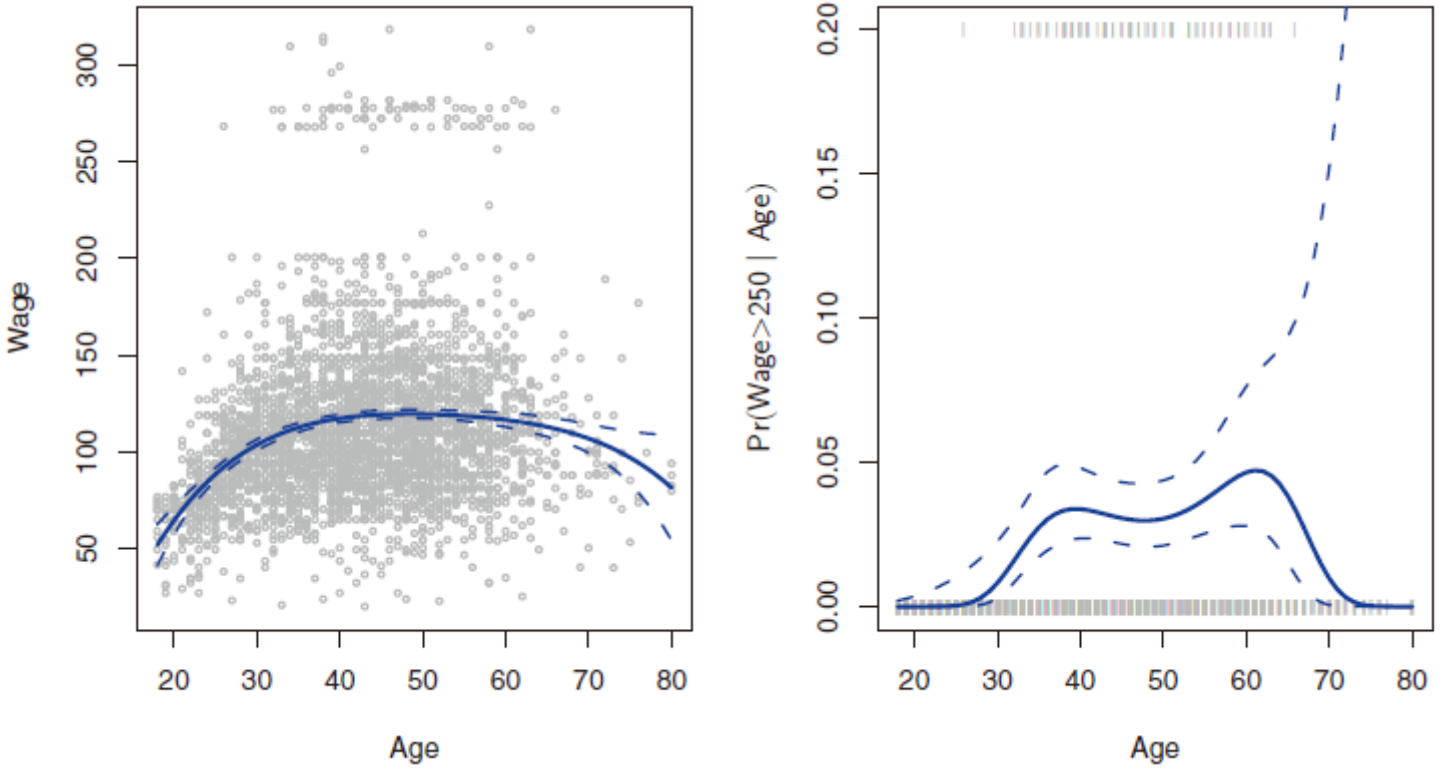
Polynomial Regression

- We use high-order terms of input variables: every others are the same.

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \ldots + \beta_d x_i^d + \epsilon_i,$$

$$\Pr(y_i > 250|x_i) = \frac{\exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \ldots + \beta_d x_i^d)}{1 + \exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \ldots + \beta_d x_i^d)}.$$

- Example (d=4)



Step Functions

- Breaking X into several bins, each of which has a constant Y value.
- In details, we create cutpoints (knots) c_1, c_2, …,, c_K in the range of X and construct step basis functions.

$$\begin{aligned} C_0(X) &= I(X < c_1), \\ C_1(X) &= I(c_1 \le X < c_2), \\ C_2(X) &= I(c_2 \le X < c_3), \\ &\vdots \\ C_{K-1}(X) &= I(c_{K-1} \le X < c_K), \\ C_K(X) &= I(c_K \le X), \end{aligned}$$

$$C_0(X) + C_1(X) + \ldots + C_K(X) = 1$$

- Then, fit Y by

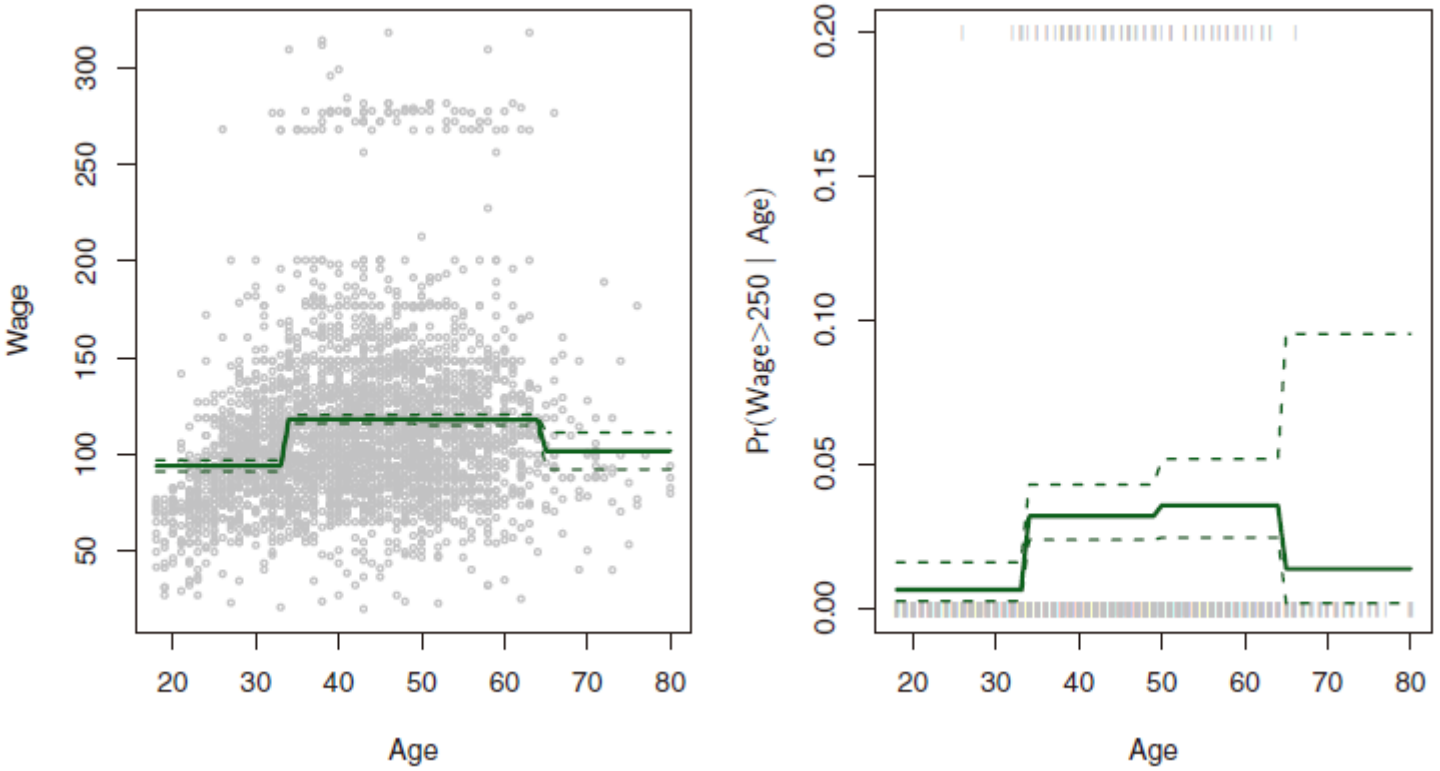
Basis functions

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \ldots + \beta_K C_K(x_i) + \epsilon_i.$$

$$\Pr(y_i > 250|x_i) = \frac{\exp(\beta_0 + \beta_1 C_1(x_i) + \ldots + \beta_K C_K(x_i))}{1 + \exp(\beta_0 + \beta_1 C_1(x_i) + \ldots + \beta_K C_K(x_i))}$$

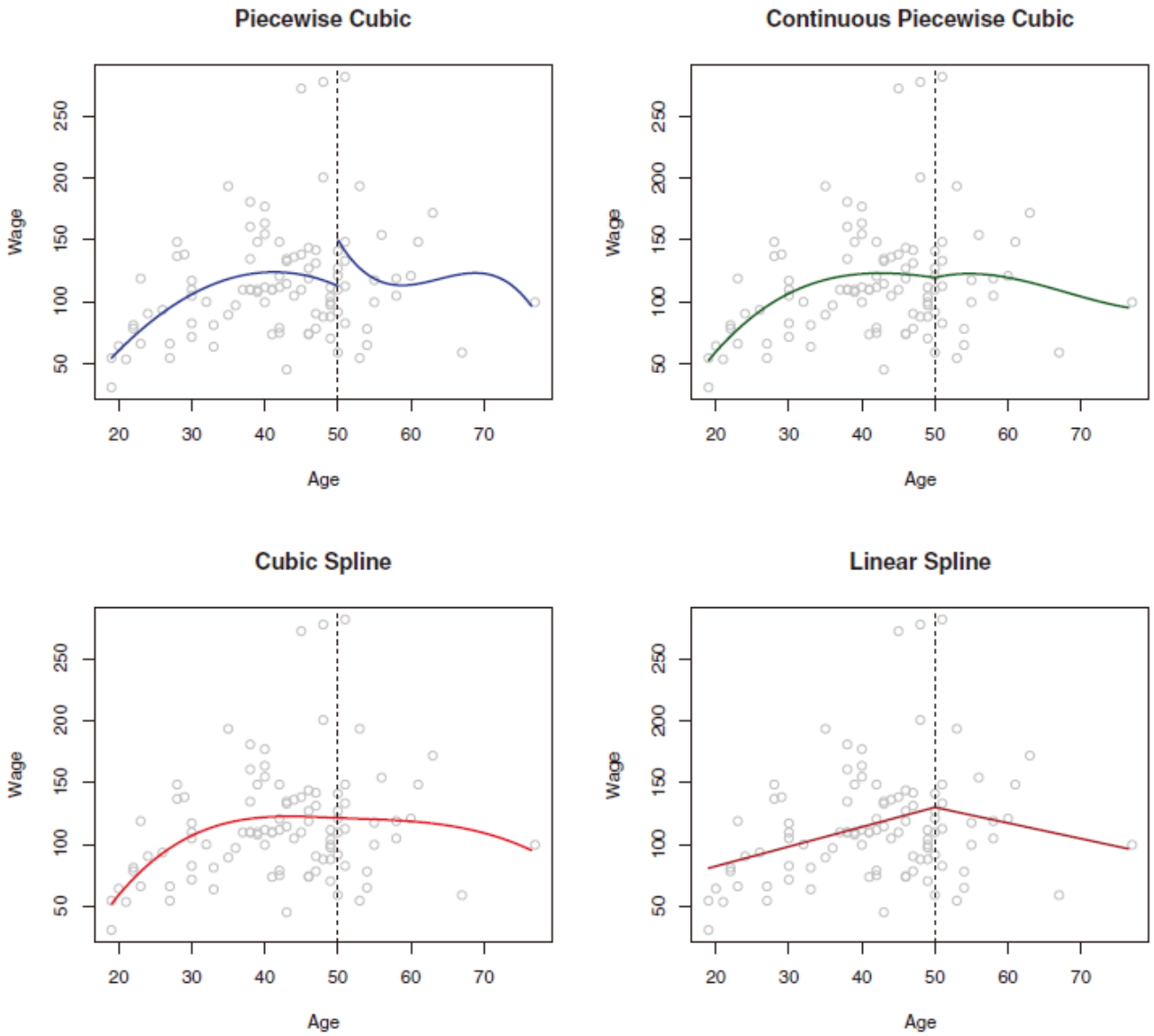
Step Functions

- Example



- Regression with step functions are easy to interpret.

Regression Splines



Regression Spline

- **Piecewise polynomial regression**
  - Divide X into several bins and fit each bin by different polynomial regressions.

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i & \text{if } x_i < c; \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i & \text{if } x_i \geq c. \end{cases}$$

  - Discontinuity.

- **Continuous piecewise polynomial regression**
  - Constraints for continuity, i.e. $f_1(c)=f_2(c)$.
  - Not smooth, or not differentiable.

- **Cubic spline**
  - Additional constraints for smoothness.
  - $f_1(c)=f_2(c)$, $f'_1(c)=f'_2(c)$, and $f''_1(c)=f''_2(c)$: use piecewise cubic polynomial.

- **Degree-d spline**
  - Piecewise degree-d polynomial, with continuity up to degree d-1 at each knot.
  - Cubic spline is degree-3 spline.

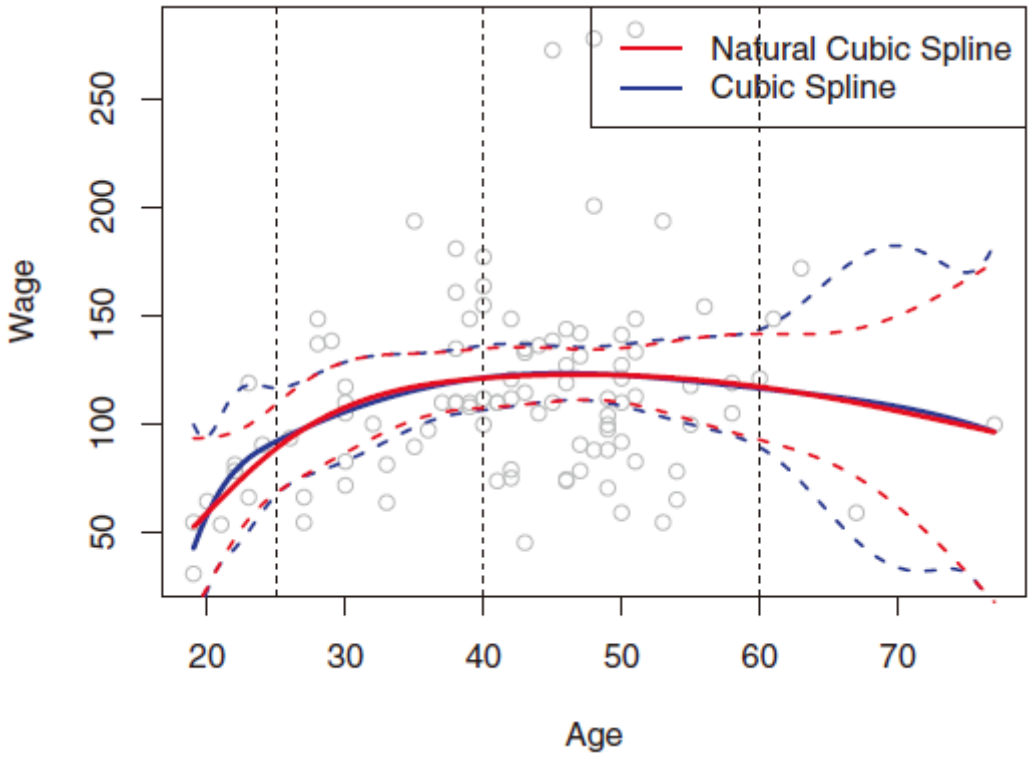
Regression Spline

- Cubic spline has K+4 degree of freedom.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - c_1)_+^3 + \cdots + \beta_{K+3} (x - c_K)_+^3$$

$$(x - \xi)_+^3 = \begin{cases} (x - \xi)^3 & \text{if } x > \xi \\ 0 & \text{otherwise,} \end{cases}$$

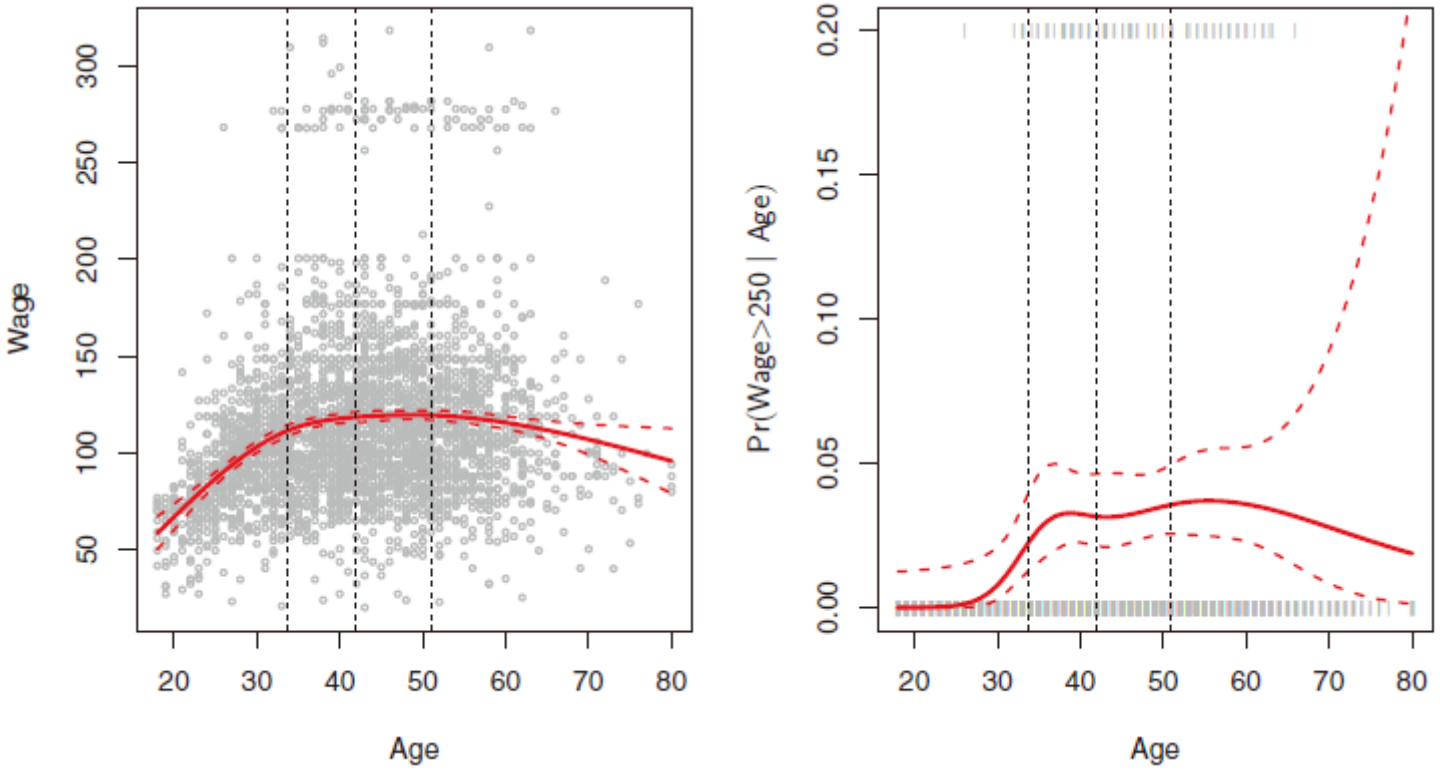
- **Natural spline**: <u>linear at the boundary</u> for the stability.



Regression Spline

- Location of knots
  - Ideally, more knots on more variable regions.
  - Practically, uniformly over the regions (e.g. 25th, 50th, 75th percentiles).

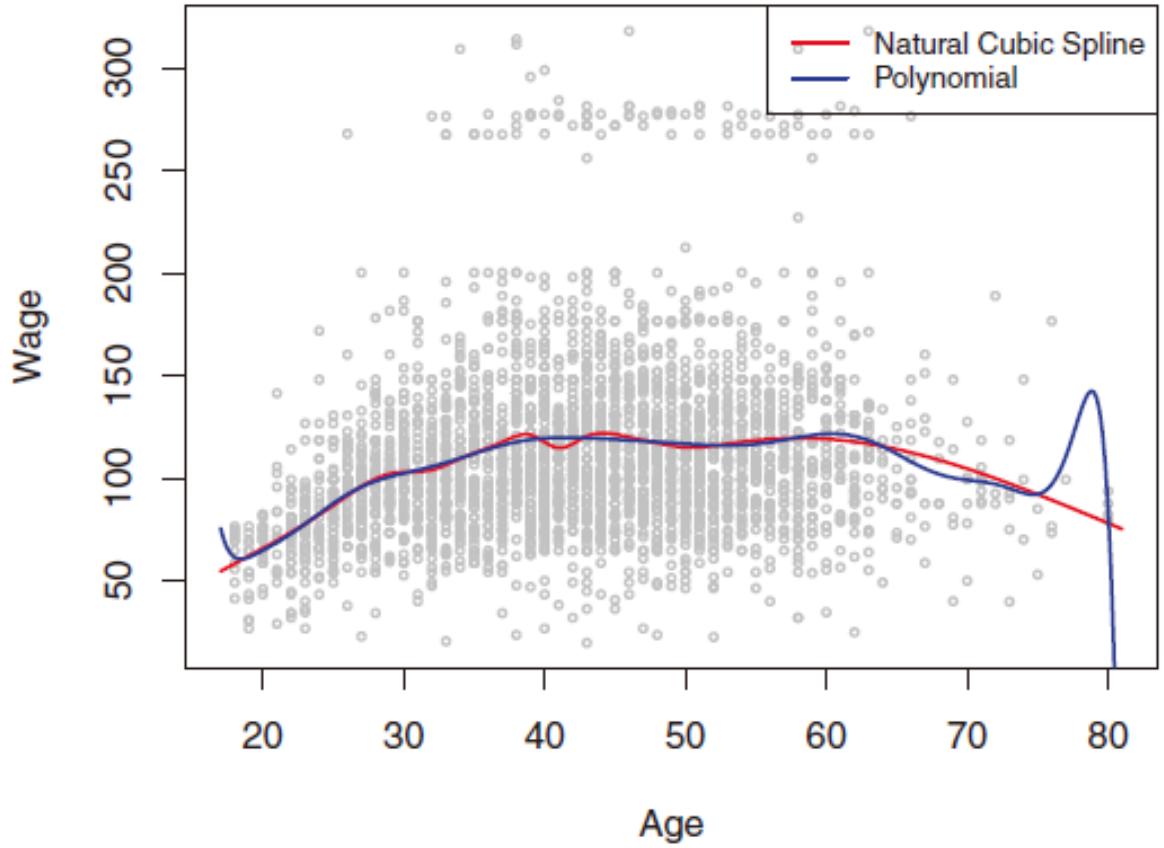
- Number of knots (or degree of freedom)
  - Through cross-validation.



Regression Splines vs. Polynomial Regression

- Polynomial regression: one high-order model
- Regression splines: many low-order models

- Regression splines often give superior and stable results.
  - E.g. Fitting results with the same degree of freedom (15).



Smoothing Splines

- A function $g(x)$ that minimizes $RSS = \sum(y_i - g(x_i))^2$ is too much wiggly.
- We want to find function $g(x)$ that makes RSS small but that is also smooth.

- Smoothing splines

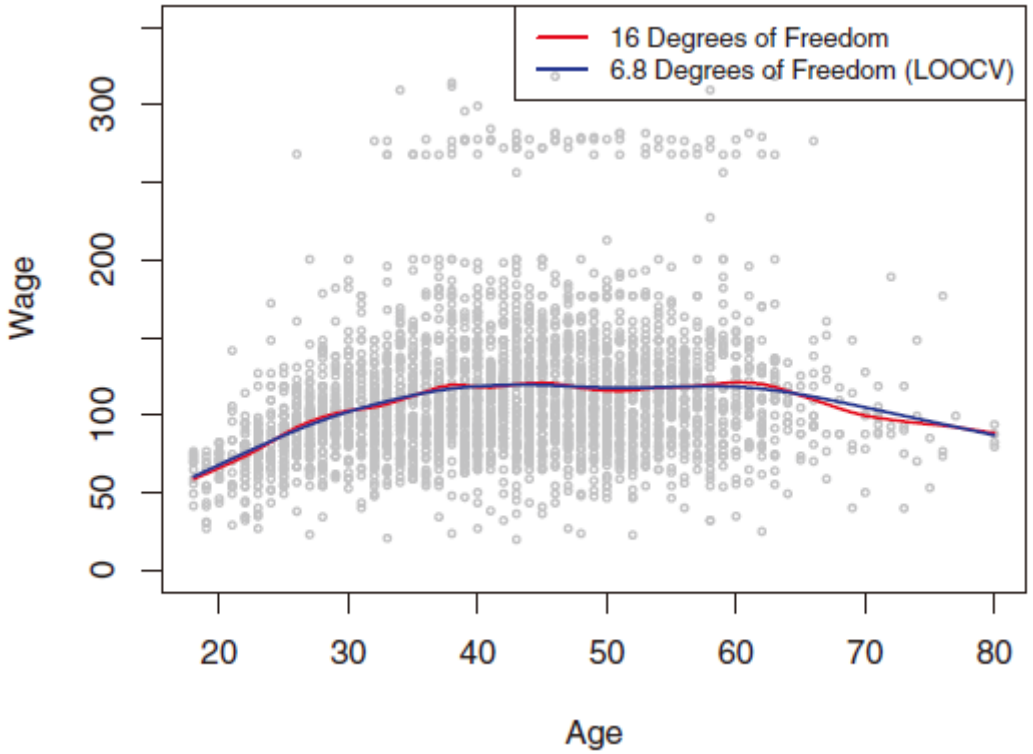
Total change of slopes

$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt$$

- Actually, smoothing splines turn out to be a shrunken version of natural cubic spline with knots at every data point $x_1, x_2, \ldots, x_n$.
  - The degree of freedom of smoothing splines is not n+4.
  - The effective d.f. is smaller and controlled by λ.
  - The tuning parameter can be determined through cross-validation.

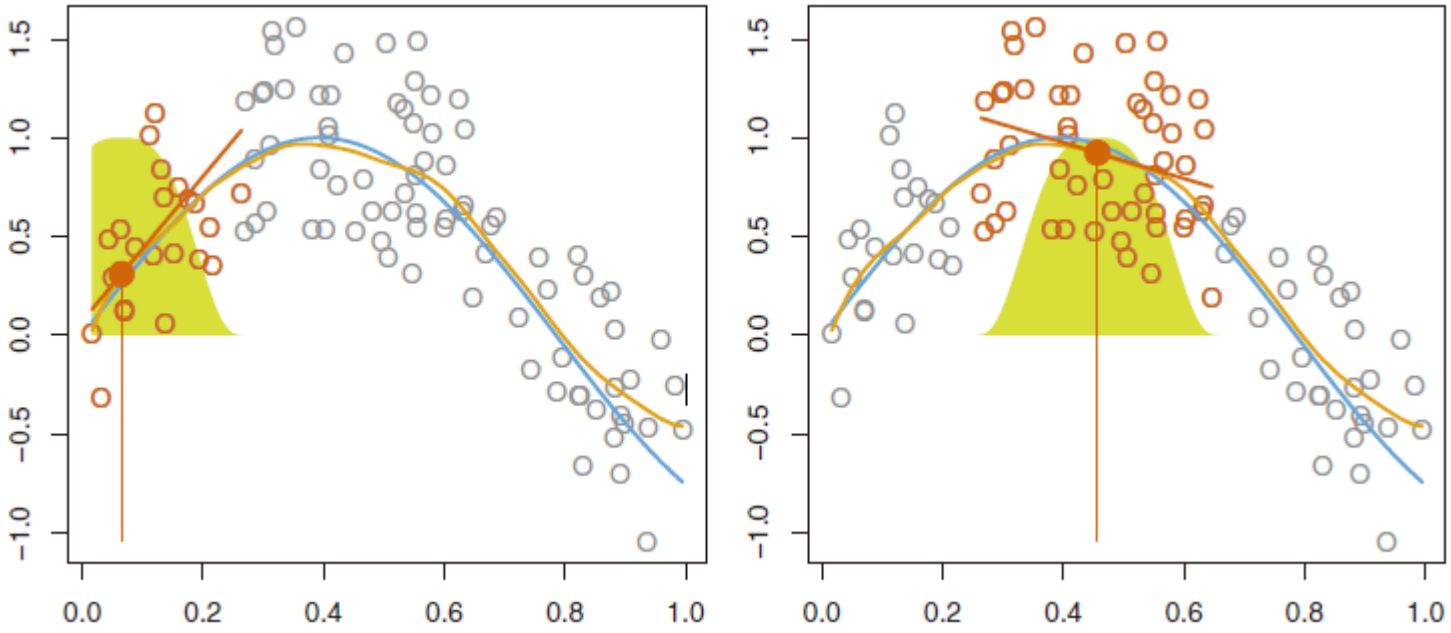
Smoothing Splines

- Example



Local Regression

- **LOESS/LOWESS**: locally weighted scatterplot smoothing or LOcal regrESSion.
  - Traditional non-parametric way (e.g. trend line).
- Fitting the value of x_0 from the nearby observations, often using weighted regression.
  - Weights are often given by a normal function or linear function.



Local Regression

- Algorithm

Algorithm 7.1 *Local Regression At* $X = x_0$

1. Gather the fraction $s = k/n$ of training points whose x_i are closest to x_0.

2. Assign a weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood, so that the point furthest from x_0 has weight zero, and the closest has the highest weight. All but these k nearest neighbors get weight zero.

3. Fit a *weighted least squares regression* of the y_i on the x_i using the aforementioned weights, by finding $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize

$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2. \tag{7.14}$$

4. The fitted value at x_0 is given by $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$.

Generalized Additive Model

- **Generalized additive model** (GAM) extends a linear model

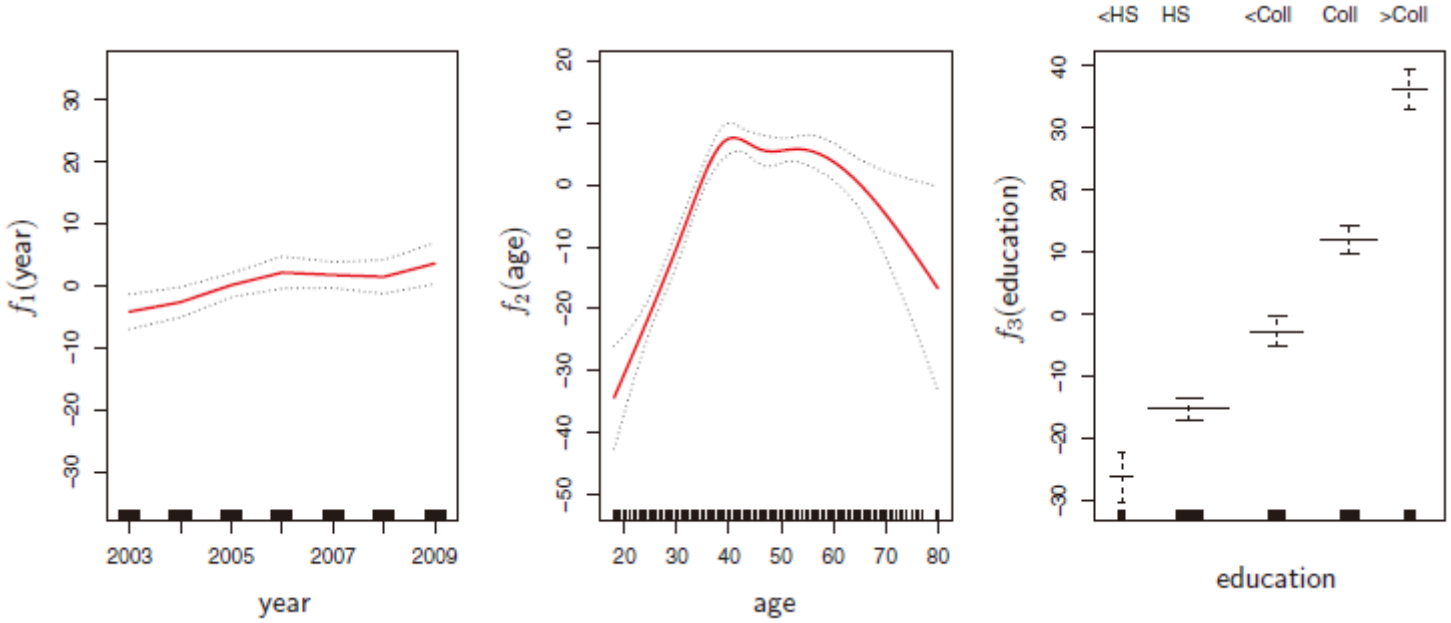
$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \epsilon_i$$

to an additive model of non-linear function *f*'s.

$$\begin{aligned} y_i &= \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i \\ &= \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \epsilon_i. \end{aligned}$$

  - *f*'s can be polynomial, step functions, splines and so on.

- Example: wage = $\beta_0 + f_1(\text{year}) + f_2(\text{age}) + f_3(\text{education}) + \varepsilon$.



Generalized Additive Model

- Pros
  - Allows non-linear relationship between X's and Y.
  - More accurate prediction is possible.
  - Possible to examine the effect of each X_i on Y.
  - Smoothness can be summarized via degrees of freedom.

- Cons
  - Restricted to be additive.
    - We might be able to use interaction term X_i x X_j for $f(X_i, X_j)$ using multi-dimensional local regression or splines.

Summary

- Efforts to overcome linearity.

- Traditional
  - Polynomial regression: using high-order terms.
  - Step function regression: superposition of non-overlapping step functions.
  - Local regression: regression with nearby points.

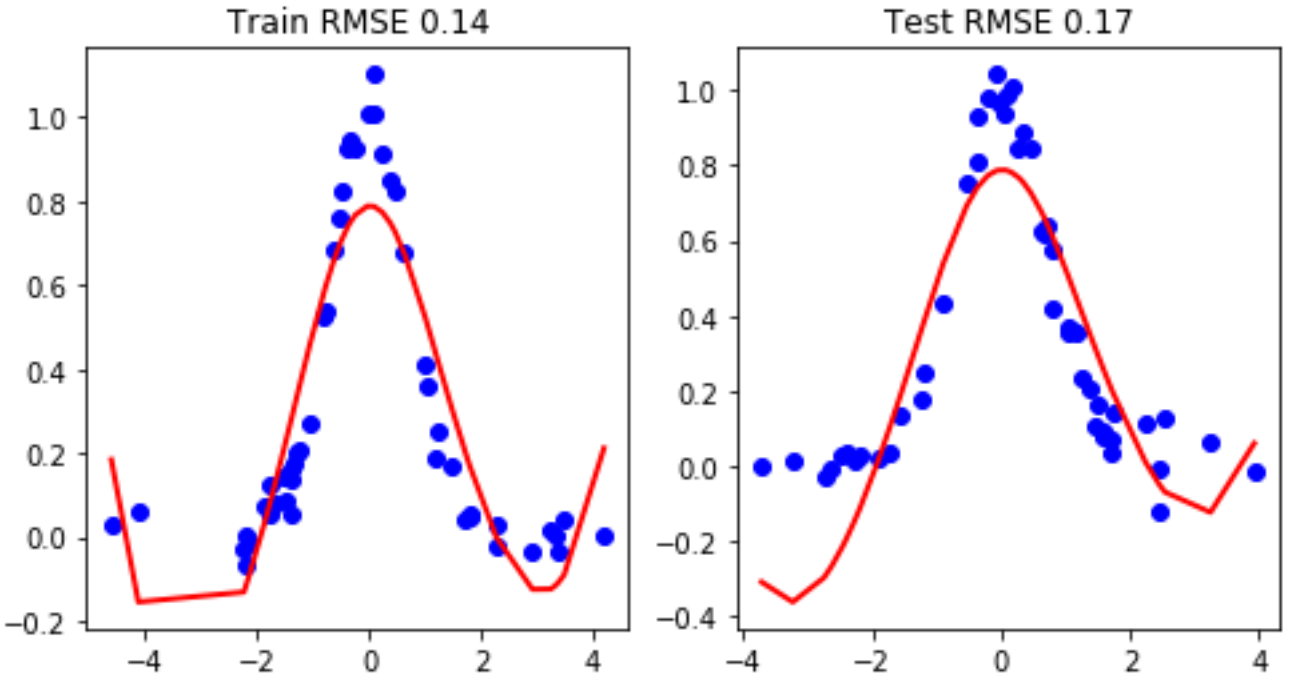
- Splines: a different polynomial regression in each bin with boundary constraints.

- Generalized additive model
  - Different predictor might have different non-linear relationship.
  - GAM considers the additive effects of different non-linear predictors.

Practice

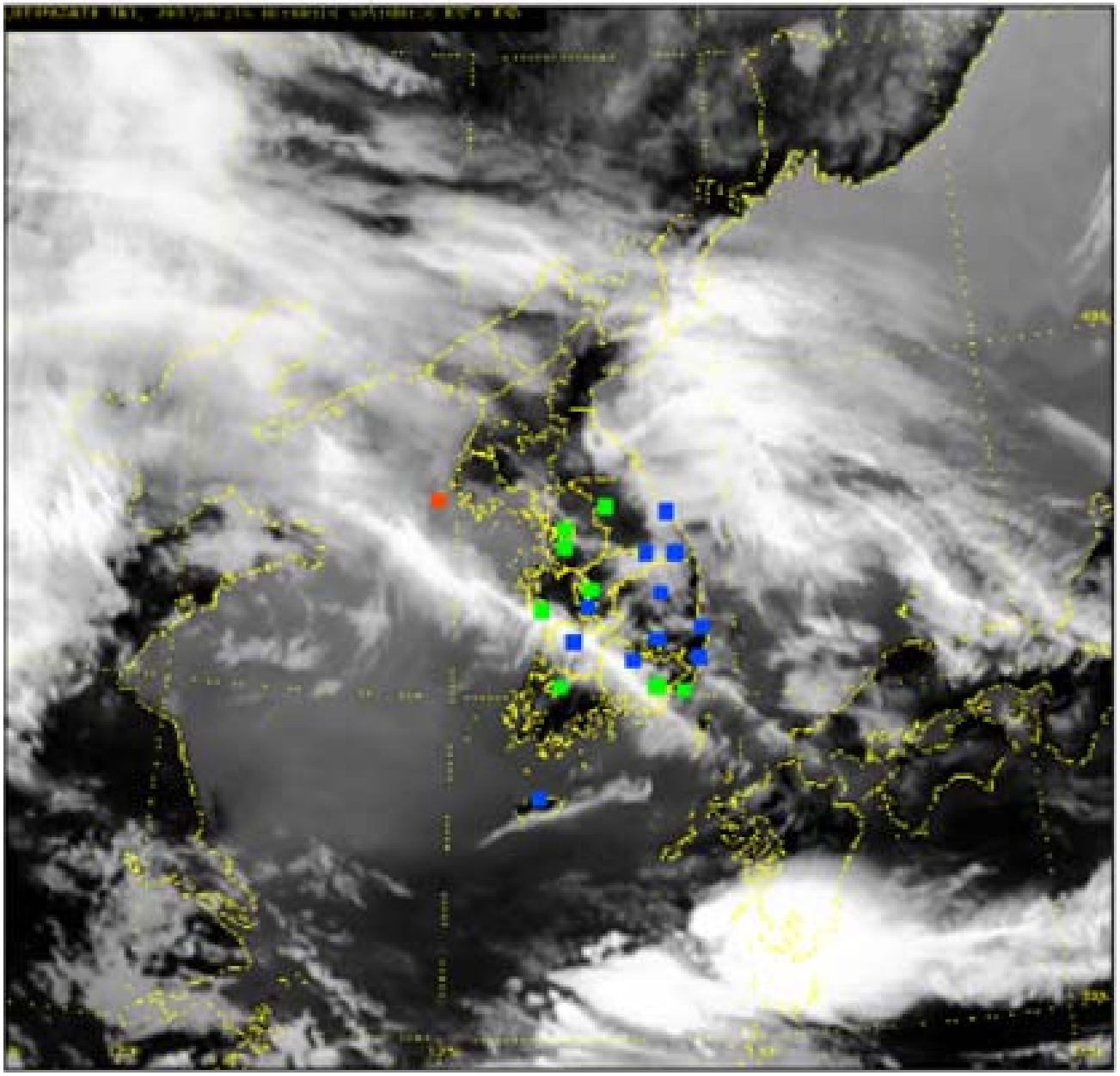
- For the synthesized data with the below code, please find your own best regression model for the test data

```
np.random.seed(1)
xtrain = 2*np.random.randn(50)
ytrain = np.exp(-xtrain**2) + 0.05*np.random.randn(50)
xtest = 2*np.random.randn(50)
ytest = np.exp(-xtest**2) + 0.05*np.random.randn(50)
```



Machine Learning Application Example

Prediction of Insolation from Satellite Images



2.4.2 연구 진행 사항

학습 데이터

- 학습 데이터 제공

데이터	데이터 종류	자료 제공	기간 (2개년)
입력	히마와리-8 기상 위성 데이터 16개 채널 이미지	기상위성센터	2016년 1월 1일 ~ 2017년 12월 31일
출력	일사량, 운저고도(구름), 시정거리(안개)	기상청 자료 개방 포털 ASOS	

표 2.4.3 학습 데이터

- Train / Test 구분

데이터	기간	시간 간격	관측소
Train	2016년 1월 1일 ~ 12월 31일	1시간 간격	20개
Test	2017년 1월 1일 ~ 12월 31일		

표 2.4.4 학습 데이터 셋 분류

Practice

- 다른 방식 (Lasso, Ridge, SVR 등)을 이용하여 일사량을 예측하시오.

Appendix

References

- Probability and Stochastic Processes: A Friendly Introduction to Electrical and Computer Engineers (3rd edition), Yates and Goodman, Wiley
- Probability, Statistics, and Random Processes for Electrical Engineering (3rd edition), Leon-Garcia, Pearson International Edition.
- An Introduction to Statistical Learning with Applications in R, James, Witten, Hastie, Tibshirani, Springer
- Pattern Recognition and Machine Learning, Bishop, Springer

About the Lecturer

- **Junhee Seok, PhD**
  - Assistant Professor, Electrical Engineering, Korea University
  - Director of Mirae Asset AI Fintech Research Center
  - Education
    - BS, Electrical Engineering, KAIST, 2001
    - PhD, Electrical Engineering, Stanford University, 2011
  - Professional Experiences
    - Postdoctoral Fellow, Statistics, Stanford University
    - Assistant Professor, HBMI, Northwestern University
  - Research Area
    - Big data analytics, Machine Learning, AI
    - Biomedicine, Finance, Climate, IoT, Materials, and etc.