Data Mining: Foundation, Techniques and Applications

Lesson 2: Machine Learning and Statistics

Li Cuiping(李翠平)
School of Information
Renmin University of China

Anthony Tung(鄧锦浩)
School of Computing
National University of Singapore

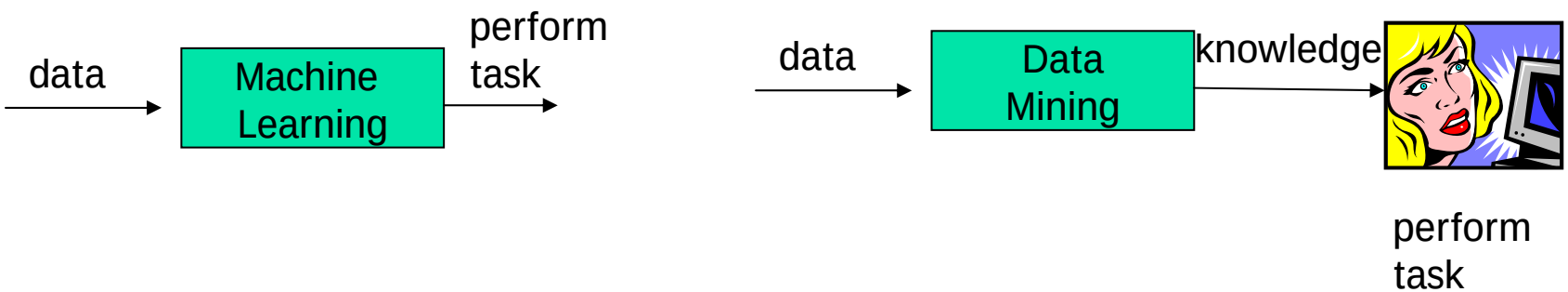
Machine Learning

- How to construct computer programs that automatically improve with experience
- Formal definition

  A computer program is said to learn from experience *E* w.r.t some classes of tasks *T* and performance *P*, if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.

Machine Learning vs Data Mining(I)

- Since data mining is essentially the use of historical data to improve decisions, we can see this as trying to learn from previous experience. Machine learning can provide many useful tools and techniques for this purpose
- Machine learning on the other hand does not need to worry about the interpretability of the knowledge being learned or discovered



Machine Learning vs Data Mining(II)

- Data mining also need to deal with the tasks of handling massive datasets which mean techniques from database research must be brought in
- Generally, we can say that machine learning deal with the effectiveness aspect of data mining while database research deal with the efficiency aspect

A Generalized View of ML(Or DM)

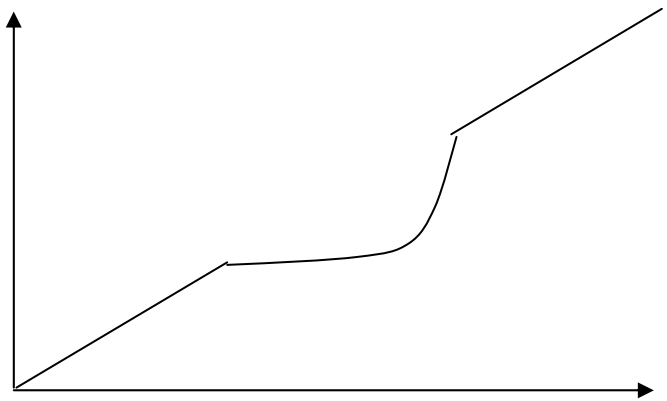
1. The *task* the algorithm is used to address (e.g. classification, clustering, etc.)
2. The *structure* of the model or pattern we are fitting to the data (e.g. a linear regression model)
3. The *score function* used to judge the quality of the fitted models or patterns (e.g. accuracy, BIC, etc.)
4. The *search or optimization method* used to search over parameters and structures (e.g. steepest descent, MCMC, etc.)
5. The *data management technique* used for storing, indexing, and retrieving data (critical when data too large to reside in memory)

Outline

- Introduction
- Models/Patterns
- Score Function
- Optimization and Search
- Conclusion

Models vs Patterns

- Models
  - Global summary of the dataset
  - Example: Fitting the line equation $Y=aX+c$ to all the data points
- Patterns
  - Local feature of the dataset. Limited to a subset of rows and attributes. Can be cause by concept drift.
  - Example: A small portion of the data above does not conform to $Y=aX^2+c$ but instead conform to $Y=aX+c$
- Boundary between models and patterns is not always clear



Types of Models

- Prediction Model
  - In a predictive model, one of the variable Y should be predicted from the other variables $X_1,\ldots,X_p$
  - Also called **supervised learning**
  - If Y is numerical, we call it **regression**
  - If Y is categorical, we call it **classification**
- Descriptive Model
  - Aim is to produce a summary or description of the data
  - Also called **unsupervised learning**
  - Example: Clustering, data cubes
- Models for Structure Data
  - To model situations in which data items affect one another
  - Example: time series, sequences, spatial data

Predictive Model: Regression

Models

Predictive

Descriptive

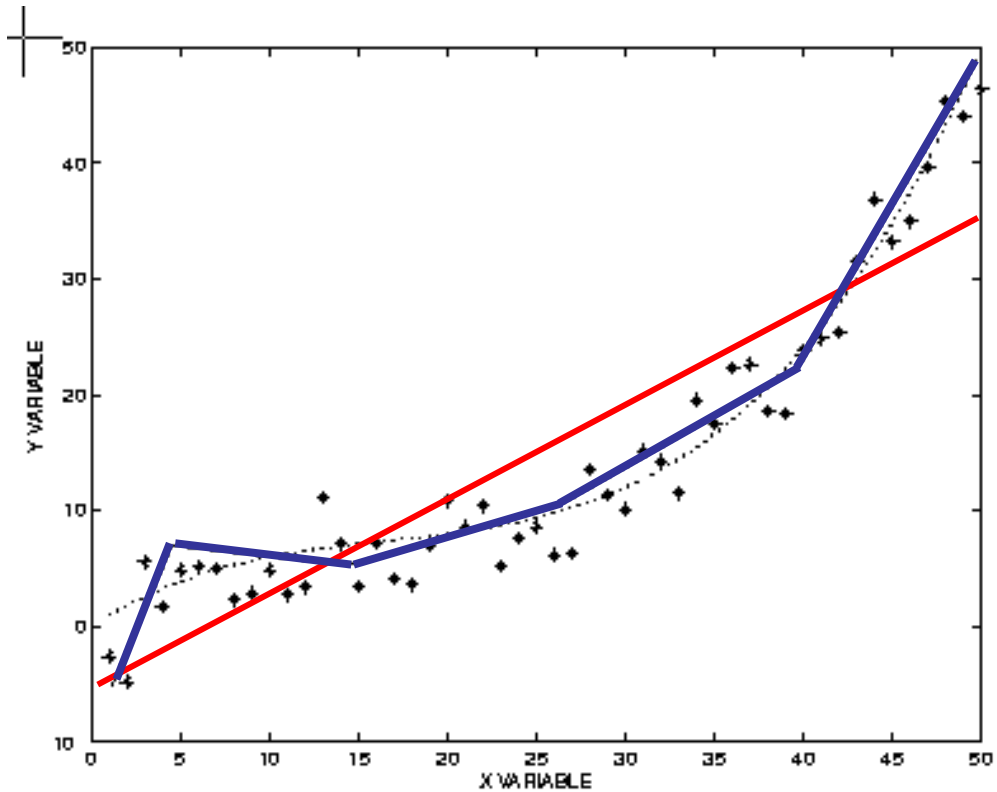
Structured Data

- Linear regression

$$Y = a_0 + \sum_{j=1}^{p} a_j X_j$$

$$Y = a_0 + \sum_{j=1}^{p} a_j f_j(X_j)$$

- Piecewise linear



Predictive Model:Non-parametric Regression

Models

Structured Data

Descriptive

Predictive

- Linear regression
- Piecewise linear
- Non-parametric regression

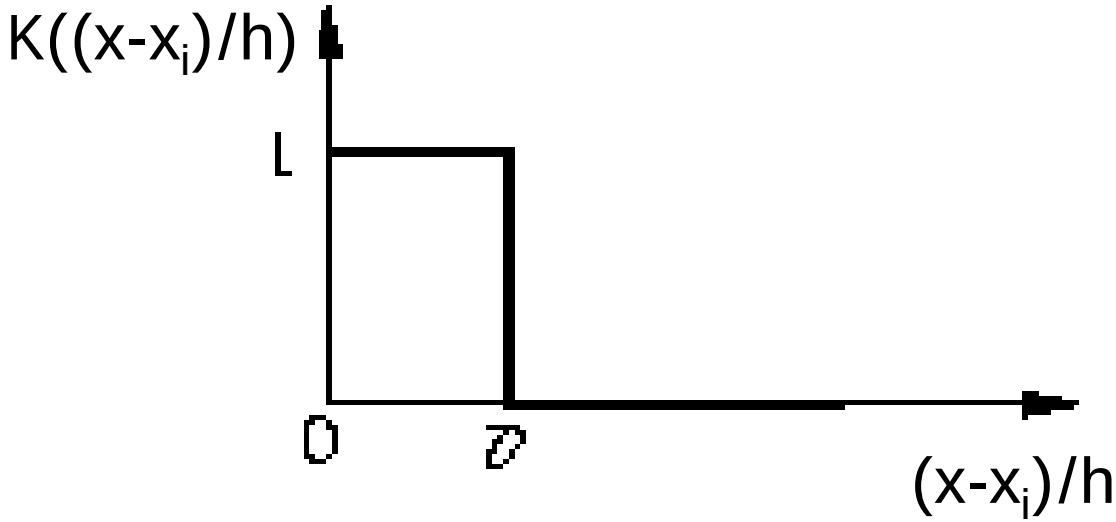
- Try to compute the value of a point based on the influences of all its neighbors
- Weight function: Use to model the fact that neighbors which are near have more influence while those which are far have less influence
- Related to nearest neighbor methods

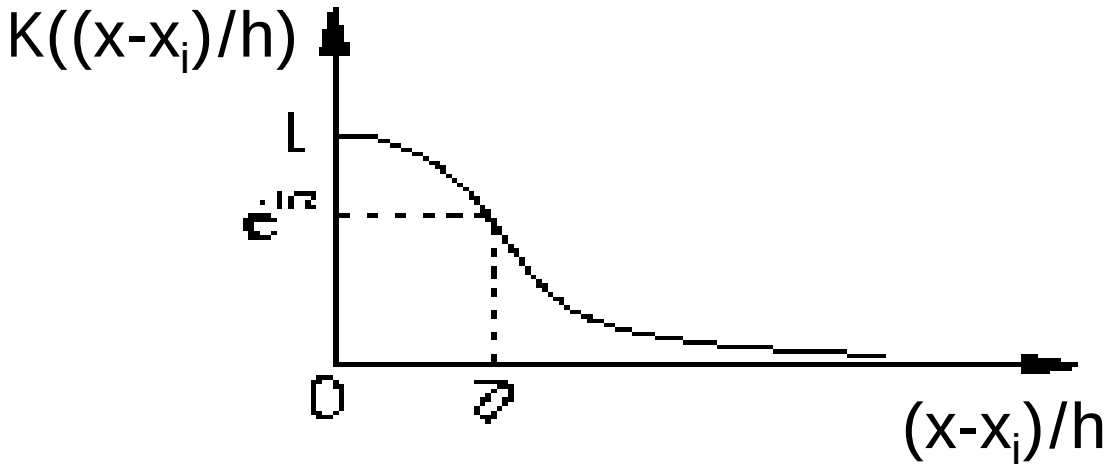
$$f(x) = \frac{1}{n}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

kernel function(eg. normal distribution)

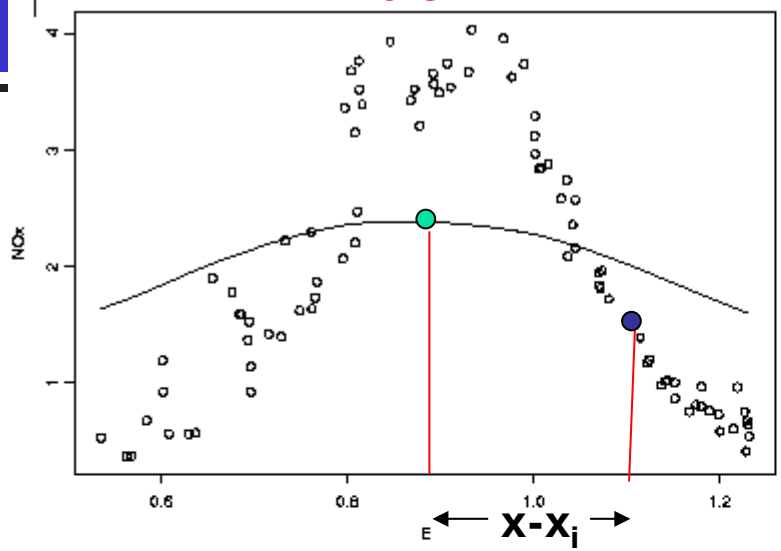
bandwidth

Examples of kernel function K()

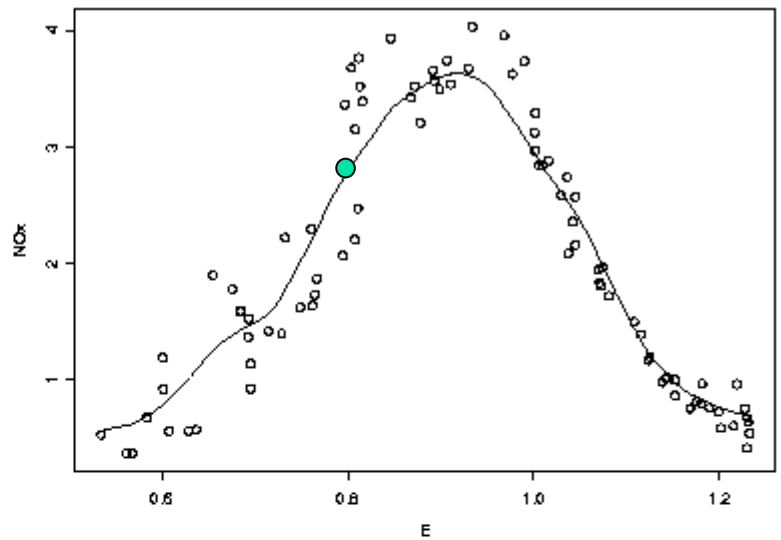




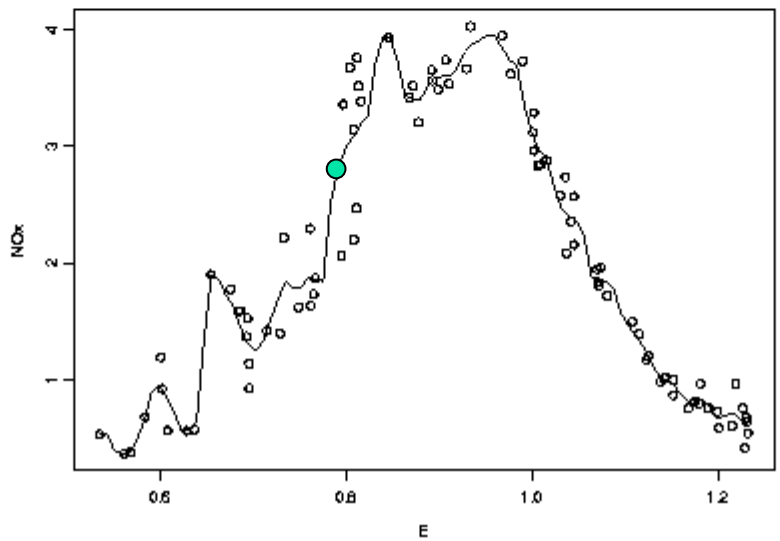
h=0.5



h=0.1



$$f(x)=\frac{1}{n}\sum_{i=1}^{n}K\left(\frac{x-x_i}{h}\right)$$



h=0.02

Predictive Model:Classification

Models

Predictive

Descriptive

Structured Data

- Linear regression
- Piecewise linear
- Nonparametric regression
- Stochastic Components

- A perfect functional relationship between the predictor variables $X_1,\ldots,X_p$ and the predicted variable Y is generally hard to find
- Introduce a random component into the model $y = g(x;\theta) + \mathbf{e}$

Predictive Model:Classification

Models

Predictive

Descriptive

Structured Data

- Linear regression
- Piecewise linear
- Nonparametric regression
- Stochastic Components
- Classification

Logistic Regression

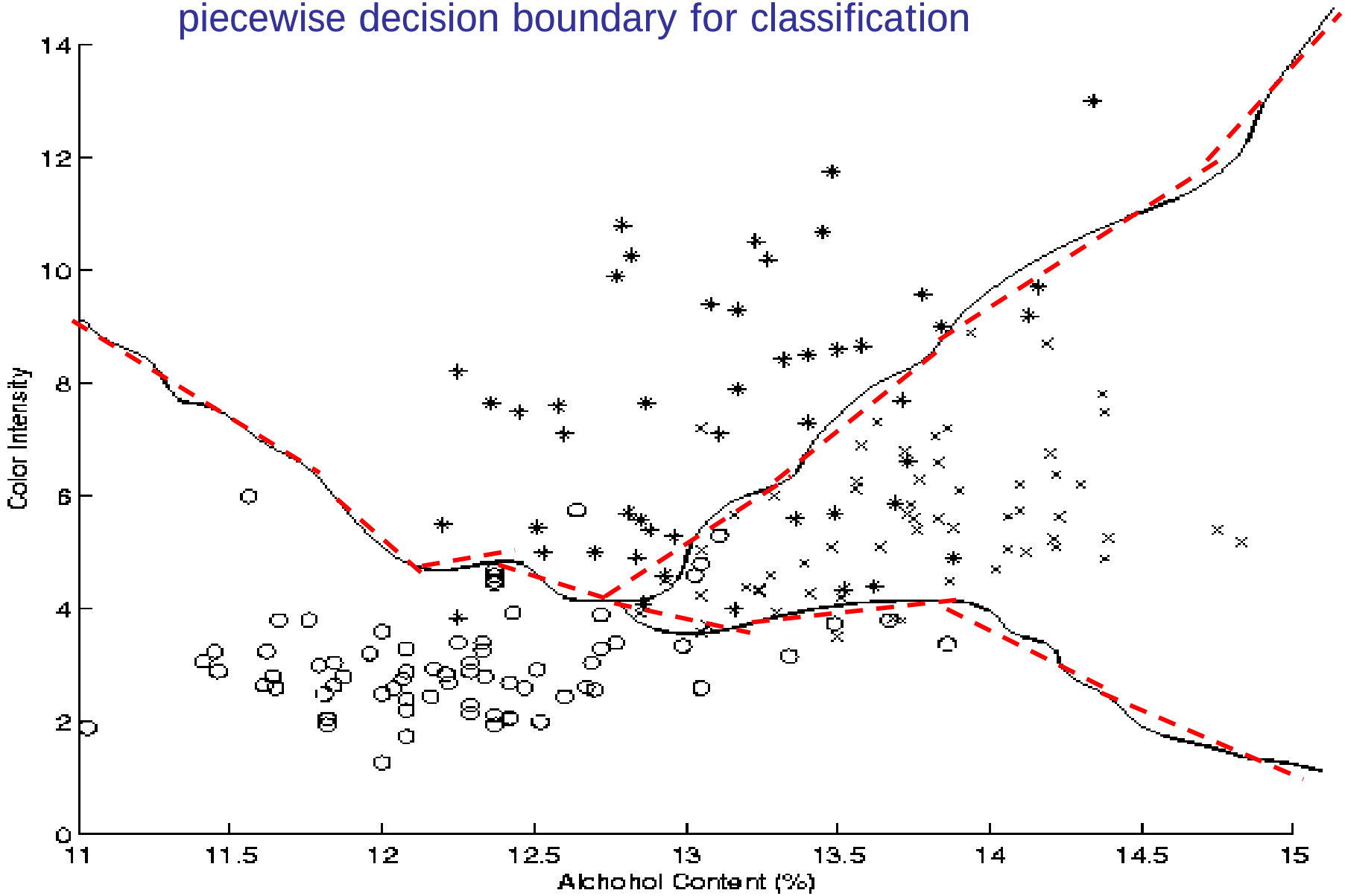
Naïve bayes/Bayesian networks

Nearest Neighbors

Support vector machines

Trees

piecewise decision boundary for classification



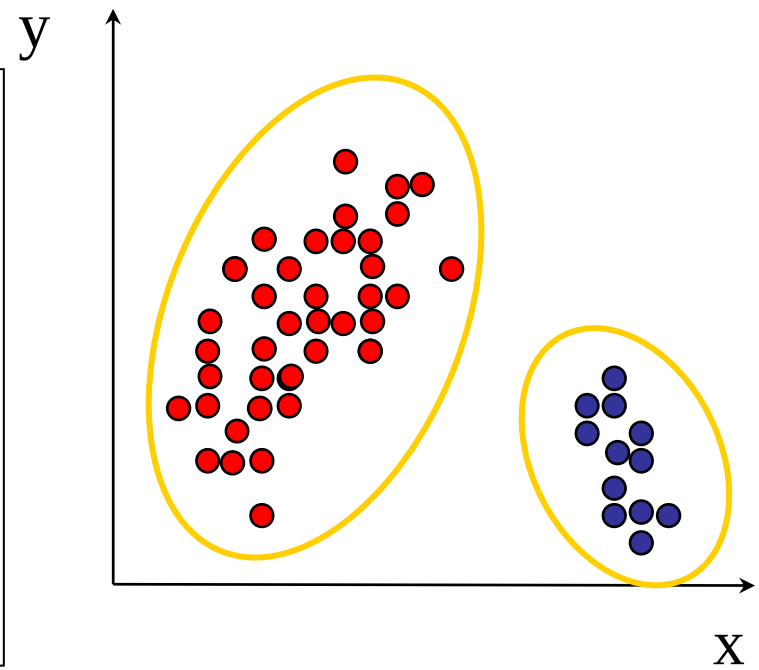
Descriptive Models

Models

Descriptive

Structured Data

- For estimation of probability distribution and density distribution
- Example: Summarize the data in the diagram into two oval shape (represented as a density distribution)



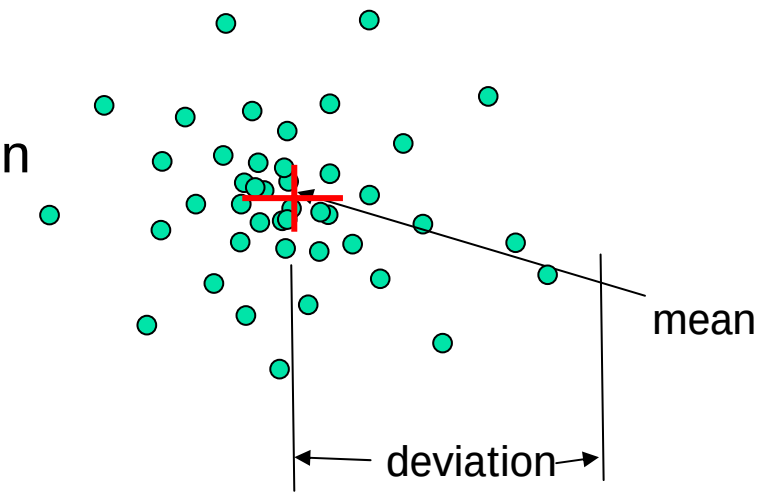
Descriptive Models: Parametric

Models

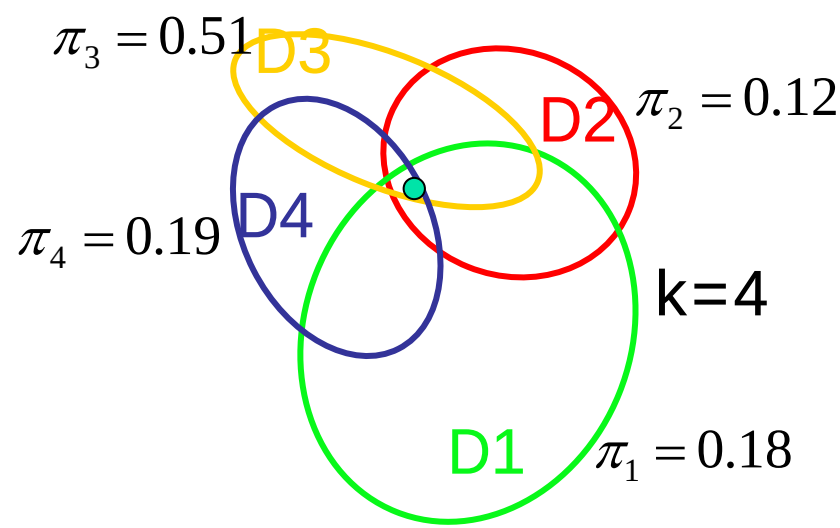
Descriptive

- Parametric models
- Mixture of Parametric Models

Gaussian Distribution



$$p(x) = \sum_{i=1}^{k} p_k(x|\theta_k)\pi_k, \ \sum_k \pi_k = 1$$



Descriptive Model: Categorical Data

Models

Descriptive

- Parametric models
- Mixture of Parametric Models
- Categorical data

contingency table

		Dementia		
		None	Mild	Severe
Smoker	No	426	66	132
	Yes	284	44	88

Categorical data can't be represented spatially. If small value of p (i.e. small no. of dimensions) and small number of attribute values, represent them as contingency table.

Descriptive Model: High Dimensional(I)

Models

Descriptive

- Parametric models
- Mixture of Parametric Models
- Categorical data
- Graphical Markov models (categorical, continuous, mixed)

- Dimensionality is a fundamental challenge in density and distribution estimation. **Model complexity tend to grow exponentially with dimensions.**
- Factorization and Independence
  - Assume independence of all variables

$$prob\ (x_1,\ldots,\ x_k) = \prod_{k=1}^{p} p_k(x_k)$$

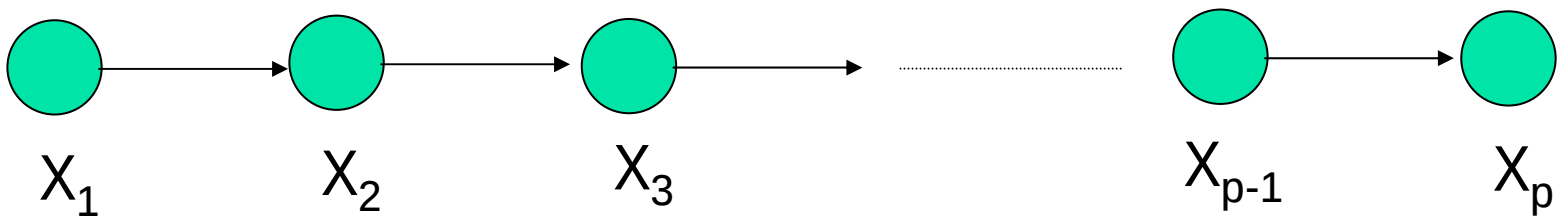
  - eg: if prob(smoker_yes)=50%, prob(demential_no)=20% and prob(smoker_yes, demential_no)=10%, then we **need NOT store prob(smoker_yes, demential_no).**

Descriptive Model: High Dimensional(II)

- But in real life, assuming independence of all variables means a very inaccurate model. A tradeoff is needed

fully independence (simple but inaccurate model) ⟷ **specified joint probability (accurate but complex model)**

- Markov chain assumption
  - Assume that the variable x_k is only dependent on $x_1,\ldots,x_{k-1}$
  - $$prob\,(x_1,\ldots,x_k) = p_1(x_1)\prod_{k=2}^{p} p(x_k \mid x_1,\ldots,x_{k-1})$$



Descriptive Model: High Dimensional(III)

Models

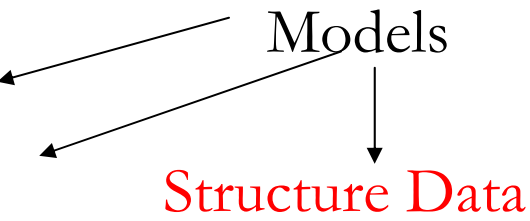
Descriptive

- Parametric models
- Mixture of Parametric Models
- Categorical data
- Graphical Markov models (categorical, continuous, mixed)
- Curse of Dimensionality

X_i	X_j	$Y = X_i$ XOR X_j
0	0	0
0	1	1
1	0	1
1	1	0

- Other ways to deal with dimensionality
- Variables selection
  - also call features selection
  - given predictor variable Y, find variables that are independent of Y
  - although Y might be independent of a variable X_i, it does not mean that Y is independent of X_i when Xi is combined with other variable (eg, X_j)
  - rely on heuristic, optimal is difficult
- Transformations
  - Projection Pursuit Regression
  - Principal Components Analysis

Models for Structure Data



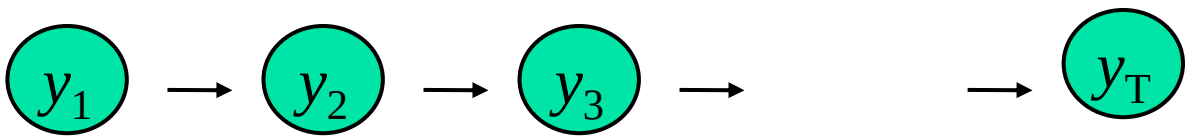
- Instead of modeling only dependency between the variables, we need to model the relationship between the attribute values of data items
- Example: time series, sequences, spatial data
- Challenge: We have even more combinations. How can we try to fit a reasonable model to such data without blowing up the complexity of the model ?

Structure Data: Markov Models

First-order: $$p(y_1,\ldots,y_T) = p_1(y_1)\prod_{t=2}^{T} p_t(y_t \mid y_{t-1})$$

First-order (stationary): $$p(y_1,\ldots,y_T) = p_1(y_1)\prod_{t=2}^{T} p(y_t \mid y_{t-1})$$

Given that each variable y_i have m states, the first equation require $O(m^2T)$ to store all the conditional probability. The second equation need $O(m^2)$.



Structure Data: Markov Models(II)

- For real-valued Y, we can have

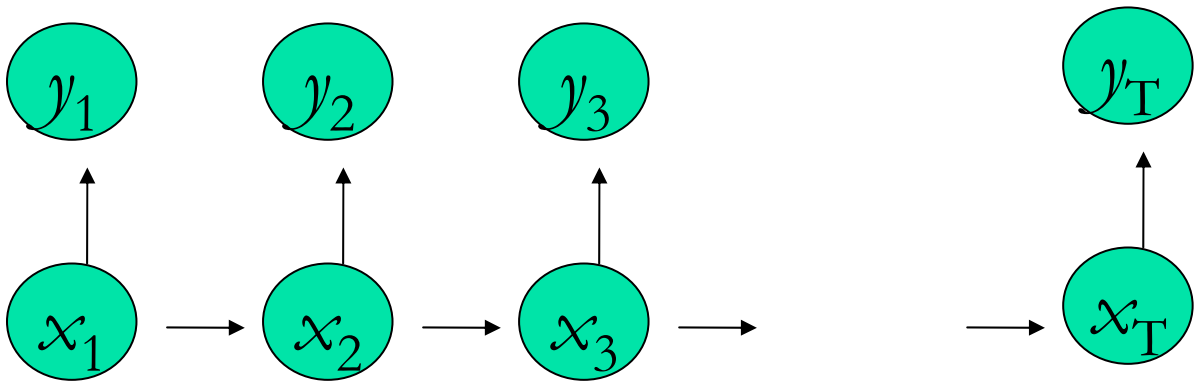
$$p(y_t \mid y_{t-1}) = \frac{1}{\sqrt{2\pi}\sigma} \exp - \frac{1}{2}\left(\frac{y_t - g(y_{t-1})}{\sigma}\right)^2$$

  - if we have $g(y_{t-1}) = \alpha_0 + \alpha_1 y_{t-1}$, then we call it first-order autoregressive model

- Can be extended to kth order of Markov model

$$p(y_t \mid y_{t-1}) = \frac{1}{\sqrt{2\pi}\sigma} \exp - \frac{1}{2}\left(\frac{y_t - g(y_{t-1}, y_{t-2}, y_{t-3}, \ldots, y_{t-k})}{\sigma}\right)^2$$

Structure Data: Hidden Markov Models



Hidden state X is categorical with m states and is first order Markov. Having generated state x_t at time t (based on the Markov chain), generate y_t based on $p(y_t | x_t)$. Similarly to mixture model.

$$p(y_1,\ldots,y_T,x_1,\ldots,x_T) = p_1(x_1)p_1(y_1 \mid x_1)\prod_{t=2}^{T} p(y_t \mid x_t)p(x_t \mid x_{t-1})$$

Note: to compute $p(y_1,\ldots,y_T)$ need to sum/integrate over all possible state sequences...

Score Function

Based on Chapter 7 of Hand, Mannila, & Smyth

David Madigan

Score Function: Introduction

- E.g. how to pick the "best" a and b in $Y = aX + b$
- Usual score in this case is the sum of squared errors
- Scores for patterns versus scores for models
- Predictive versus descriptive scores
- Score function for model of fixed complexity and for models of different complexity

Scoring Pattern

- Search for local patterns in data is relatively recent, hence far smaller toolbox of techniques for scoring patterns (compared to for models)
- Reason: Usefulness of a pattern lies in the eye of the beholder. One person's noisy outlier may be another person's Nobel Prize
- General approach: Measure degree of interestingness compared to some expected values. Example: $p(b)=0.25$ and $p(b|a)=0.75$, then this is interesting knowledge
- Coverage of the pattern: The proportion of the data to which a pattern applies

Predictive Model Scores

$$S_{SSE}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(\hat{f}(x(i);\theta) - y(i))^2$$

$$S_{0/1}(\theta) = \frac{1}{N}\sum_{i=1}^{N} I(\hat{f}(x(i);\theta), y(i))$$

- Assume all observations equally important
- Depend on differences rather than values
- Symmetric

Descriptive Model Scores

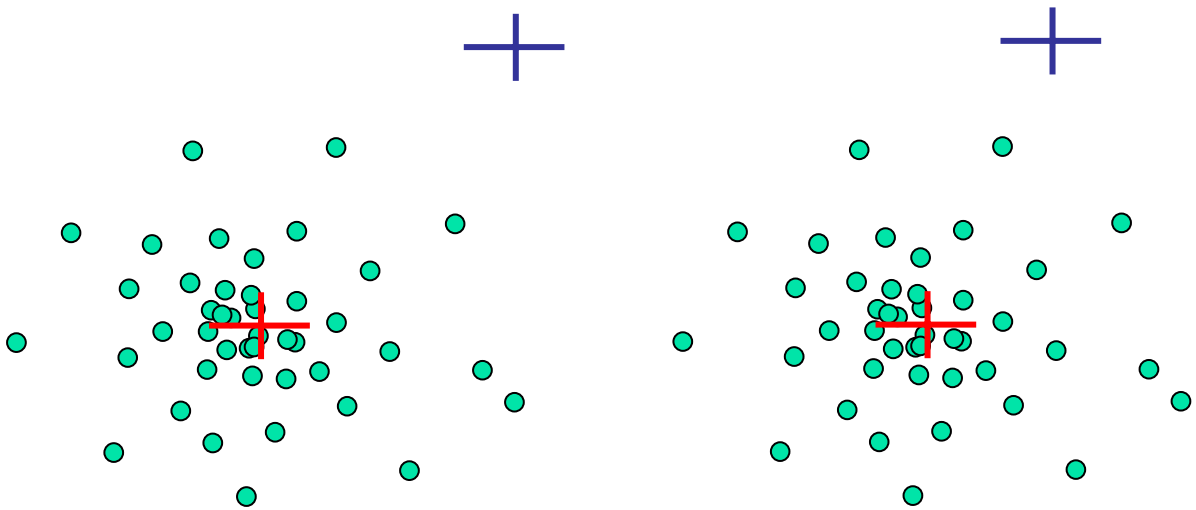
$$L(\theta) = \prod_{i=1}^{N} \hat{p}(x(i);\theta)$$

- "Pick the model that assigns highest probability to what actually happened"
- Many different scoring rules for non-probabilistic models
- Taking negative log on both side give us the log likelihood function

$$S_L(\theta) = -\log L(\theta) = -\sum_{i=1}^{N} \log \hat{p}(x(i);\theta)$$

Descriptive Model Scores :Example

- We wish to pick two Gaussian distributions to model the following data points, where would you pick the mean of the Gaussian distributions ?



General Concepts in Comparing Models

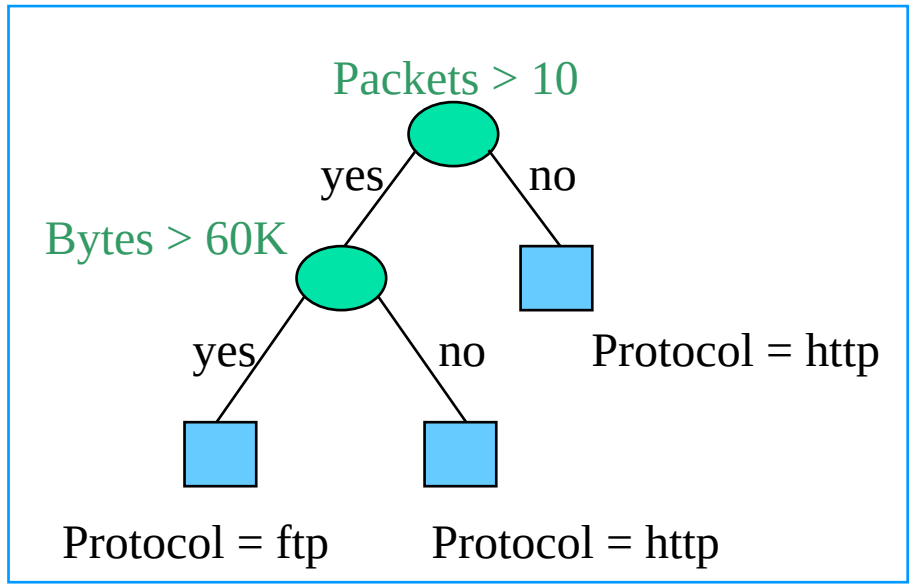
- General technique is based on compression and information-theoretic arguments, our score function is generally decomposed as:

$$S_I(\theta, M) = \text{number of bits to describe the data given the model} + \text{number of bits to describe the model(and parameters)}$$

- Occam Razor:Finding the shortest consistent hypothesis.
- Also call the Minimum Description Length (MDL) Principle where we perform compression by storing the model and the error

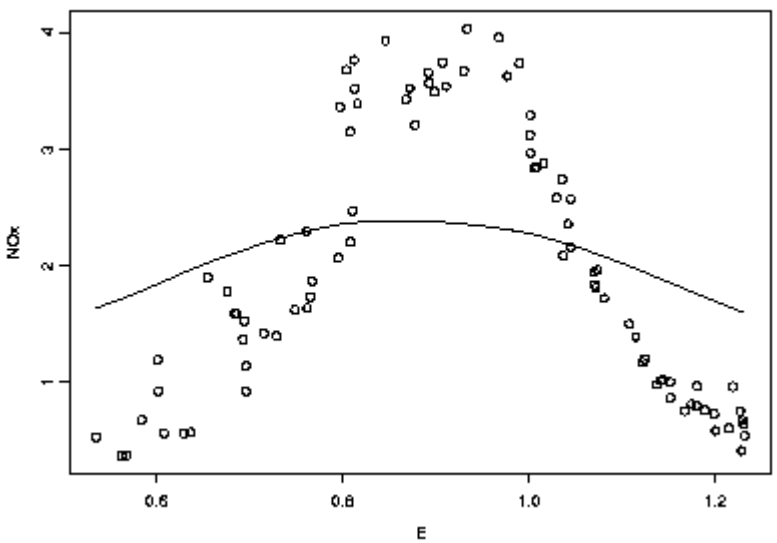
MDL Example: Compression with Classification Trees

bytes	packets	protocol
20K	3	http
24K	5	http
20K	8	http
40K	11	ftp
58K	18	http
100K	24	ftp
300K	35	ftp
80K	15	http

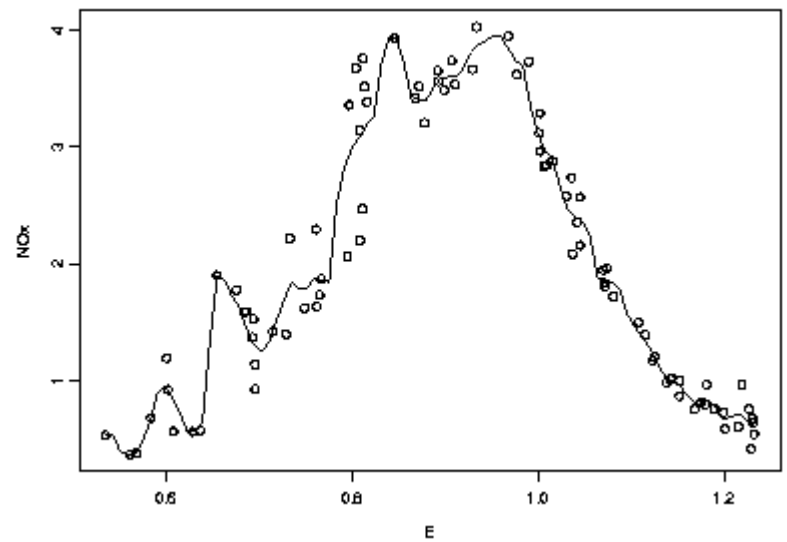


Outlier: Row 4, protocol=ftp,
Row 8, protocol =http

Bias-Variance Tradeoff



High Bias - Low Variance



Low Bias - High Variance

"overfitting" - modeling the random component

parameters ↑, estimation accuracy ↓

Score function should embody the compromise

Scoring Models with Different Complexities

Bias-variance tradeoff

$$MSE(X) = E[\hat{y} - \mu_y]^2 = E[\hat{y} - E(\hat{y})]^2 + E[E(\hat{y}) - \mu_y]^2$$

(MSE=Variance + Bias^2)

score(model) = error(model) + penalty-function(model)

AIC: $S_{AIC}(M_k) = 2S_L(\hat{\theta}_k; M_k) + 2d_k, \quad k = 1, \ldots, K$

where S_L is the negative log-likelihood

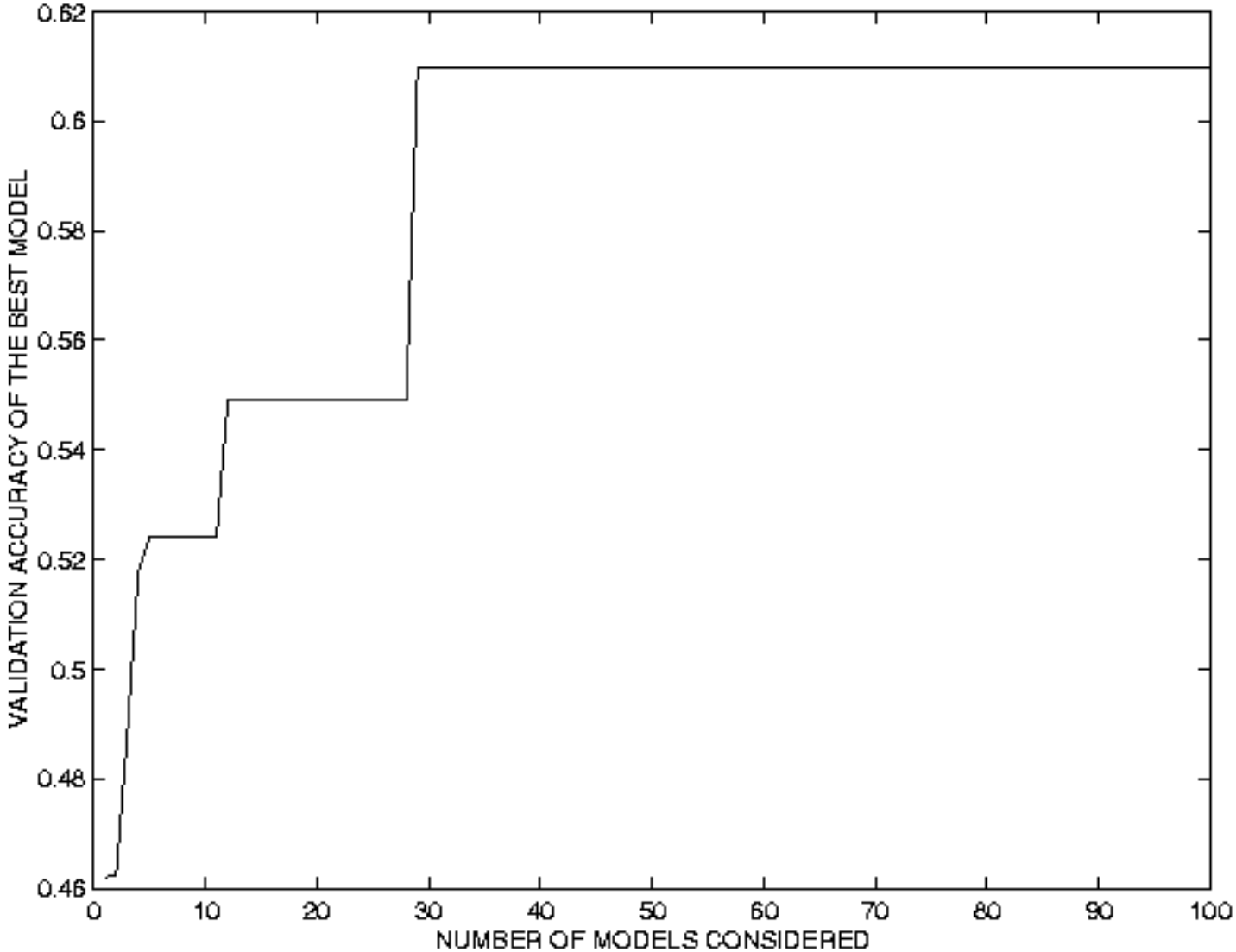
Bayesian Information Criterion

$$S_{BIC}(M_k) = 2S_L(\hat{\theta}_k; M_k) + d_k \log n, \quad k = 1, \ldots, K$$

- BIC is an O(1) approximation to $p(D|M)$
- Error term dominate penalty function as n grow larger. Hence make sense to try to reduce error as n grow larger.

Score Functions on External Validation

- Instead of penalizing complexity, look at performance by separating dataset into a design part and validation part
- Note: even using hold-out data, performance results can be optimistically biased
- Need a third data set call test set



classification performance when data are in fact noise...

Search and Optimization Methods

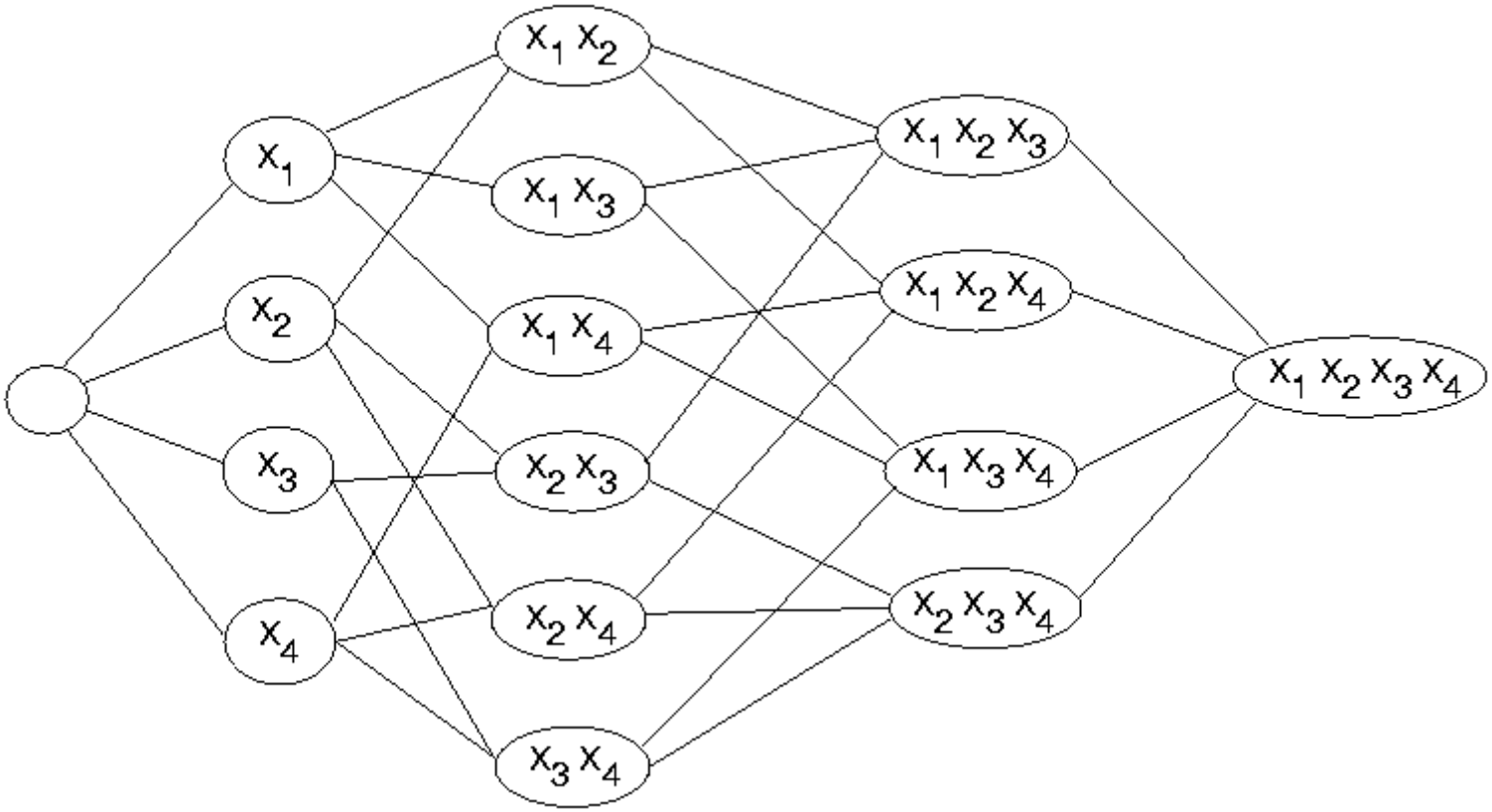
Based in part on Chapter 8 of Hand, Mannila, & Smyth

David Madigan

Search and Optimization: Introduction

- This chapter is about finding the models and parameters that minimize a general score function S
- Often have to conduct a parameter search for each visited model
- The number of possible structures can be immense. For example, there are 3.6×10^{13} undirected graphical models with 10 vertices

State-Space Formulation



Types of search methodologies

- Greedy Search
- Systematic Search and Search Heuristics
- Branch-and Bound

Greedy Search

1. Initialize. Chose an initial state M_k

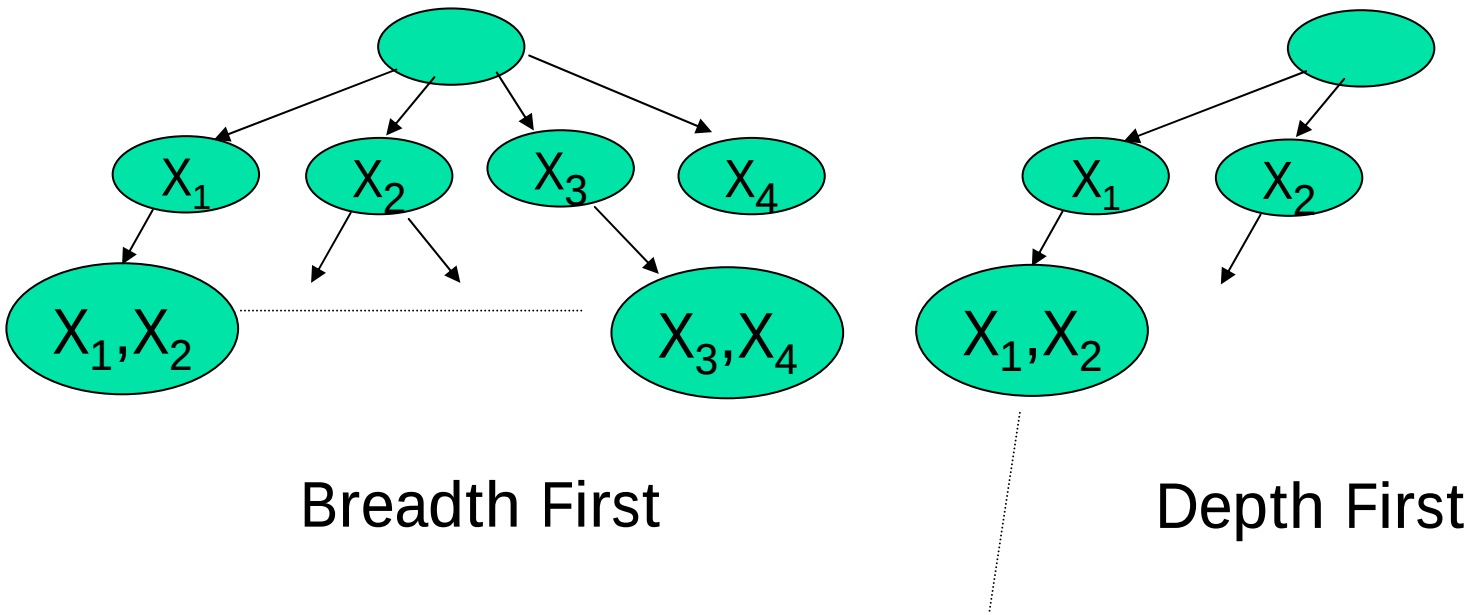
2. Iterate. Evaluate the score function at all adjacent states and move to the best one

3. Stopping Criterion. Repeat step 2 until no further improvement can be made.

4. Multiple Restarts. Repeat 1-3 from different starting points and choose the best solution found.

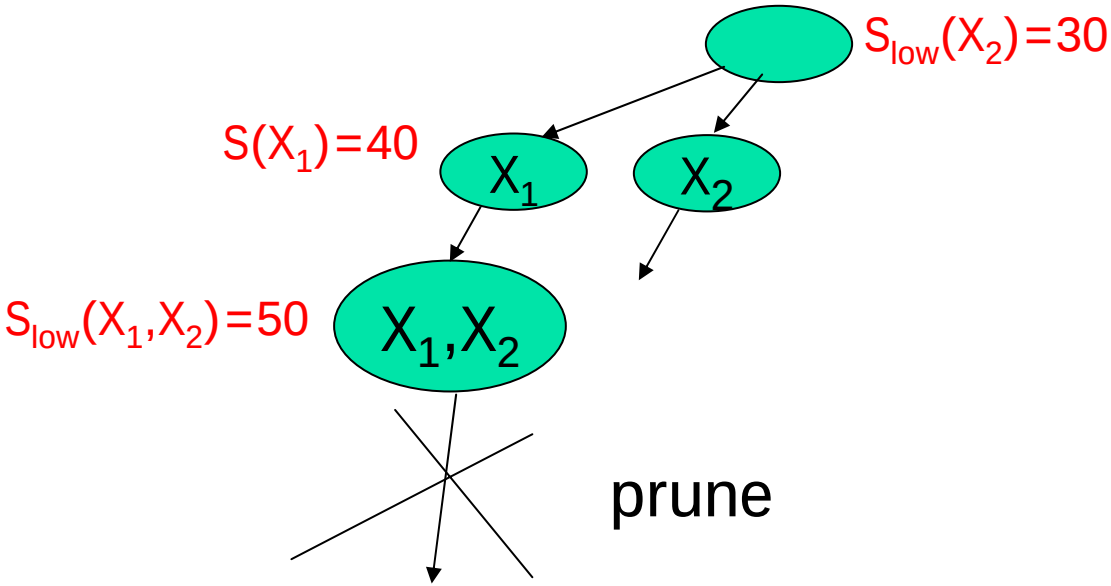
Systematic Search and Search Heuristics

- Monitor more than one models at the same time in the search
- Approach of a search tree
  - Dynamically construct as the search proceed



Branch and Bound

- When exploring a search tree, keep track of the best score S so far. If it is possible to have an lower bound S_{low} of the score function for a subtree, prune off subtree if $S_{low} > S$



Parameter Optimization

Finding the parameters θ that minimize a score function $S(\theta)$ is usually equivalent to the problem of minimizing a complicated function in a high-dimensional space

Define the gradient function is S as:

$$g(\theta) = \nabla_\theta S(\theta) = \left(\frac{\partial S(\theta)}{\partial \theta_1}, \ldots, \frac{\partial S(\theta)}{\partial \theta_d} \right)$$

When closed form solutions to $\nabla S(\theta)=0$ exist, no need for numerical methods.

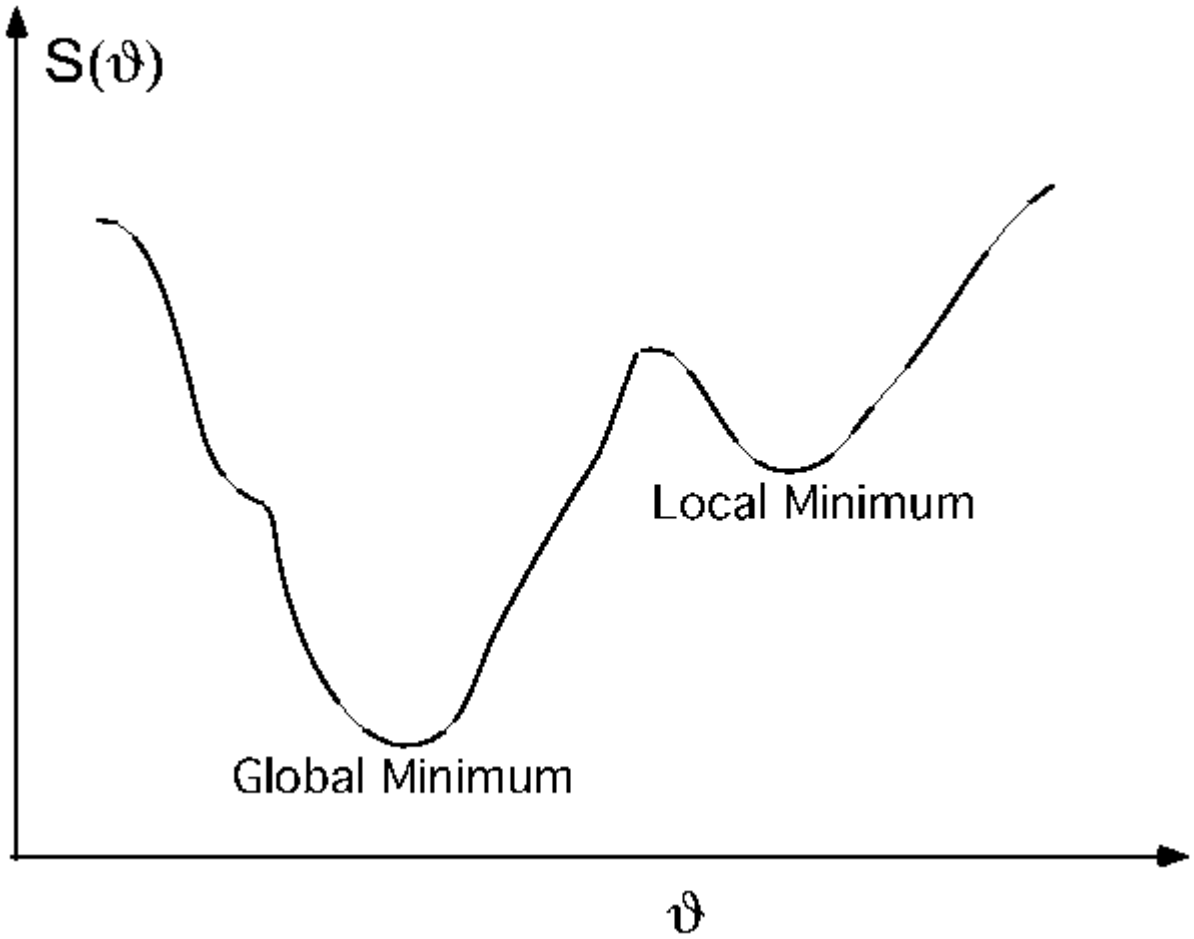
Gradient-Based Methods

1. Initialize. Choose an initial value for $\theta = \theta^0$

2. Iterate. Starting with i=0, let $\theta^{i+1} = \theta^i + \lambda^i \boldsymbol{v}^i$ where $\boldsymbol{v}$ is the direction of the next step and lambda is the distance. Generally choose v to be a direction that improves the score

3. Convergence. Repeat step 2 until S appears to have reached a local minimum.

4. Multiple Restarts. Repeat steps 1-3 from different initial starting points and choose the best minimum found.


S(ϑ)
Local Minimum
Global Minimum
ϑ

Univariate Optimization

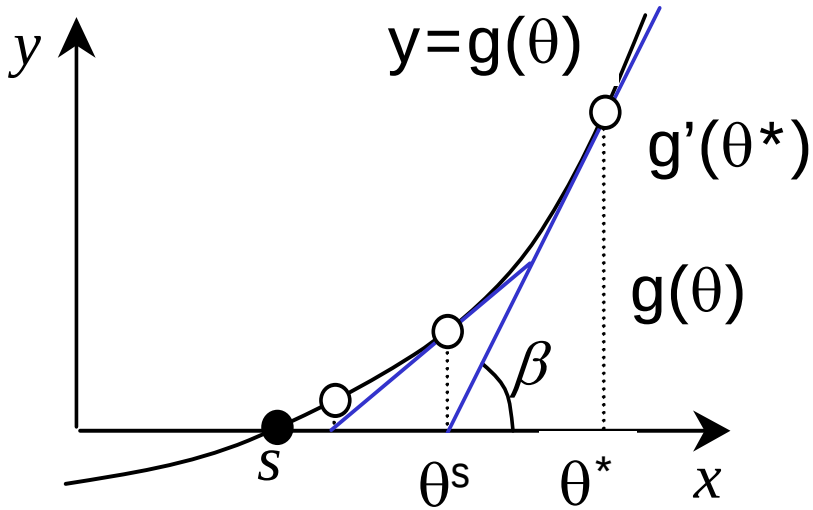
Let g(θ)=S'(θ). Newton-Raphson proceeds as follows. Suppose $g(\theta^s)=0$. Then:

$$g(\theta^s) \approx g'(\theta^*)(\theta^s - \theta^*) + g(\theta^*)$$

$$\Rightarrow \theta^s \approx \theta^* - \frac{g(\theta^*)}{g'(\theta^*)}$$

i - th step is given by :

$$\theta^{i+1} = \theta^i - \frac{g(\theta^i)}{g'(\theta^i)}$$



1-D Gradient-Descent

$$\theta^{i+1} = \theta^{i} - \lambda g(\theta^{i})$$

- λ usually chosen to be quite small
- Special case of NR where $1/g'(\theta^{i})$ is replaced by a constant

Multivariate Case

Curse-of-Dimensionality again. For example, suppose S is defined on a d-dimensional unit hypercube. Suppose we know that non of the components of θ are less than 1/2 at the optimum.

if d=1, have eliminated half the parameter space

if d=2, have eliminated 1/4 of the parameter space

if d=20, have eliminated 1/1,000,000 of the parameter space!

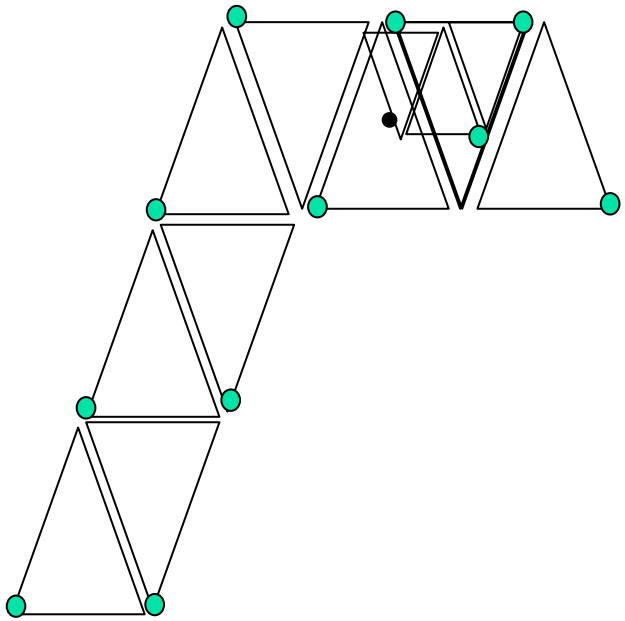
Multivariate Gradient Descent

$$\theta^{i+1} = \theta^{i} - \lambda g(\theta^{i})$$

- $-g(\theta^{i})$ points in the direction of *steepest descent*
- Guaranteed to converge if λ small enough
- Can replace λ with second-derivative information ("quasi-Newton" uses approx).

Simplex Search Method

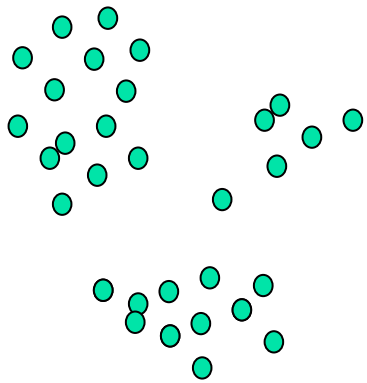
1. Evaluates d+1 points arranged in a hyper-tetrahedron
2. For example, with d=2, evaluates S at the vertices of an equilateral triangle
3. Reflect the triangle in the side opposite the vertex with the highest value
4. Repeat until oscillation occurs, then half the sides of the triangle
5. No calculation of derivatives...



EM with Missing Data(I)

- The (Expectation Maximization) EM algorithm is an important algorithm for maximizing a likelihood score function when some of the data are missing
- Special case: Clustering where the class attribute are all missing

x	y	H
1	3	
5	7	
1	8	
. . .		
6	20	



EM with Missing Data(II)

$$l(\theta) = \log\ p(D \mid \theta) = \log \sum_{H} p(D, H \mid \theta)$$

- Challenge: We must now find of set of values to maximize *l(θ)*
  - *The set of parameters θ*
  - *The set of values for H*
- General Approach:Do the following iteratively
  - Fixed *θ, vary H to maximize l(θ)*
  - *Fixed H, vary θ to maximize l(θ)*

EM with Missing Data(III)

$$l(\theta) = \log \; p(D \mid \theta) = \log \sum_H p(D, H \mid \theta)$$

Let $Q(H)$ denote a probability distribution for the missing data

$$\log \sum_H p(D, H \mid \theta) = \log \sum_H Q(H) \frac{p(D, H \mid \theta)}{Q(H)}$$

$$\geq \sum_H Q(H) \log \frac{p(D, H \mid \theta)}{Q(H)}$$

$$= \sum_H Q(H) \log p(D, H \mid \theta) - \sum_H Q(H) \log Q(H)$$

$$= F(Q, \theta)$$

This is a lower bound on $l(\theta)$

EM (continued)

$$\text{E-Step:} \quad Q^{k+1} = \arg\max_{Q} F(Q^k, \theta^k)$$

$$\text{M-Step:} \quad \theta^{k+1} = \arg\max_{\theta} F(Q^{k+1}, \theta^k)$$

In the E-Step, Max is achieved when $Q^{k+1} = p(H \mid D, \theta^k)$

In the M-Step, need to maximize:

$$\theta^{k+1} = \arg\max_{\theta} \sum_{H} p(H \mid D, \theta^k) \log p(D, H \mid \theta^k)$$

Example: *K-Means* Clustering

- Aim: Separate a set of points into k-clusters such that sum of square error is minimized.
- The model in this case are the k centers of the clusters, $(c_1,\ldots,c_k)$
- The *k-means* algorithm is implemented in 4 steps:
  - Partition objects into *k* nonempty subsets
  - M-step: Compute seed points is the center (mean point) of the cluster.
  - E-step: Assign each object to the cluster with the nearest center
  - Go back to Step 2, stop when no more new assignment.

EM Normal Mixture Example(I)

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x; \mu_k, \sigma_k)$$

$$\text{Let } \theta = (p_1, \ldots, p_k, \mu_1 \ldots, \mu_k, \sigma_1 \ldots, \sigma_k)$$

$$\text{E-Step}: p(k \mid x) = \frac{\pi_k f_k(x; \mu_k, \sigma_k)}{f(x)}$$

EM Normal Mixture Example(II)

$$\text{M-Step:}$$

$$\hat{\pi}_k = \frac{1}{n}\sum_{i=1}^{n} p(k \mid x(i))$$

$$\hat{\mu}_k = \frac{1}{n\hat{\pi}_k}\sum_{i=1}^{n} p(k \mid x(i))x(i)$$

$$\hat{\sigma}_k = \frac{1}{n\hat{\pi}_k}\sum_{i=1}^{n} p(k \mid x(i))(x(i) - \hat{\mu}_k)^2$$

Summary

- In this lecture, we look in details the following components in the machine learning framework:
  - Model/Pattern which capture the try to capture the underlying structure of the data
  - Scoring functions for evaluation the models and their parameters
  - Search and optimization techniques for search the best models and parameters based on the scoring function

Reference

- "Principles of Data Mining", David Hand, Heikki Mannila and Padhraic Smyth. Chapter 6,7 and 8.