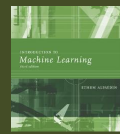
i2ml3e-chap01.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr

http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 1:

INTRODUCTION

Big Data

3

- Widespread use of personal computers and wireless communication leads to "big data"
- We are both producers and consumers of data
- Data is not random, it has structure, e.g., customer behavior
- We need "big theory" to extract that structure from data for
  (a) Understanding the process
  (b) Making predictions for the future

Why "Learn" ?

4

- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
- There is no need to "learn" to calculate payroll
- Learning is used when:
  - Human expertise does not exist (navigating on Mars),
  - Humans are unable to explain their expertise (speech recognition)
  - Solution changes in time (routing on a computer network)
  - Solution needs to be adapted to particular cases (user biometrics)

What We Talk About When We Talk About "Learning"

5

- Learning general models from a data of particular examples
- Data is cheap and abundant (data warehouses, data marts); knowledge is expensive and scarce.
- Example in retail: Customer transactions to consumer behavior:
  People who bought "Blink" also bought "Outliers" (www.amazon.com)
- Build a model that is *a good and useful approximation* to the data.

Data Mining

6

- Retail: Market basket analysis, Customer relationship management (CRM)
- Finance: Credit scoring, fraud detection
- Manufacturing: Control, robotics, troubleshooting
- Medicine: Medical diagnosis
- Telecommunications: Spam filters, intrusion detection
- Bioinformatics: Motifs, alignment
- Web mining: Search engines
- ...

What is Machine Learning?

7

- Optimize a performance criterion using example data or past experience.
- Role of Statistics: Inference from a sample
- Role of Computer science: Efficient algorithms to
  - Solve the optimization problem
  - Representing and evaluating the model for inference

Applications

8

- Association
- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
- Reinforcement Learning

Learning Associations

9

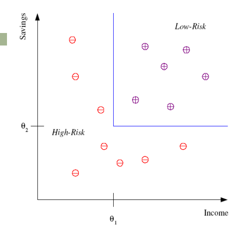
- Basket analysis:
  $P(Y \mid X)$ probability that somebody who buys X also buys Y where X and Y are products/services.

  Example: P (chips | beer) = 0.7

Classification

10

- Example: Credit scoring
- Differentiating between low-risk and high-risk customers from their *income* and *savings*



Discriminant: IF *income* > θ_1 AND *savings* > θ_2
THEN low-risk ELSE high-risk

Classification: Applications

11

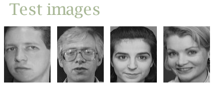
- Aka Pattern recognition
- Face recognition: Pose, lighting, occlusion (glasses, beard), make-up, hair style
- Character recognition: Different handwriting styles.
- Speech recognition: Temporal dependency.
- Medical diagnosis: From symptoms to illnesses
- Biometrics: Recognition/authentication using physical and/or behavioral characteristics: Face, iris, signature, etc
- Outlier/novelty detection:

Face Recognition

12

Training examples of a person

Test images

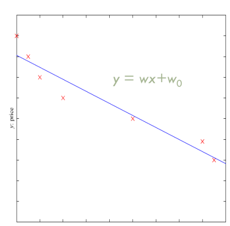


ORL dataset,
AT&T Laboratories, Cambridge UK

Regression

13

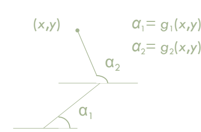
- Example: Price of a used car
- x : car attributes
  y : price
  $y = g(x \mid \theta)$
  $g(\,)$ model,
  θ parameters

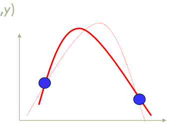


Regression Applications

14

- Navigating a car: Angle of the steering
- Kinematics of a robot arm





- Response surface design

Supervised Learning: Uses

15

- Prediction of future cases: Use the rule to predict the output for future inputs
- Knowledge extraction: The rule is easy to understand
- Compression: The rule is simpler than the data it explains
- Outlier detection: Exceptions that are not covered by the rule, e.g., fraud

Unsupervised Learning

- Learning "what normally happens"
- No output
- Clustering: Grouping similar instances
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs

Reinforcement Learning

- Learning a policy: A sequence of outputs
- No supervised output but delayed reward
- Credit assignment problem
- Game playing
- Robot in a maze
- Multiple agents, partial observability, ...

Resources: Datasets

- UCI Repository: http://www.ics.uci.edu/~mlearn/MLRepository.html
- Statlib: http://lib.stat.cmu.edu/

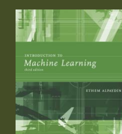
Resources: Journals

- Journal of Machine Learning Research www.jmlr.org
- Machine Learning
- Neural Computation
- Neural Networks
- IEEE Trans on Neural Networks and Learning Systems
- IEEE Trans on Pattern Analysis and Machine Intelligence
- Journals on Statistics/Data Mining/Signal Processing/Natural Language Processing/Bioinformatics/...

Resources: Conferences

- International Conference on Machine Learning (ICML)
- European Conference on Machine Learning (ECML)
- Neural Information Processing Systems (NIPS)
- Uncertainty in Artificial Intelligence (UAI)
- Computational Learning Theory (COLT)
- International Conference on Artificial Neural Networks (ICANN)
- International Conference on AI & Statistics (AISTATS)
- International Conference on Pattern Recognition (ICPR)
- ...

i2ml3e-chap02.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 2:

SUPERVISED LEARNING

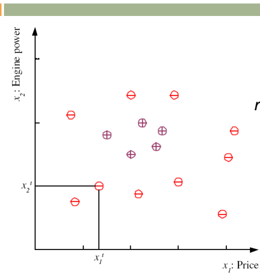
Learning a Class from Examples

- Class C of a "family car"
  - Prediction: Is car x a family car?
  - Knowledge extraction: What do people expect from a family car?
- Output:

  Positive (+) and negative (–) examples
- Input representation:

  x_1: price, x_2 : engine power

Training set $\mathcal{X}$



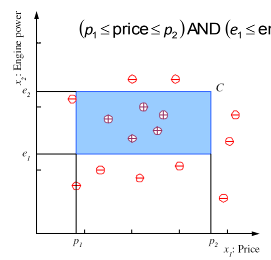
$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

$$r = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is positive} \\ 0 \text{ if } \mathbf{x} \text{ is negative} \end{cases}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

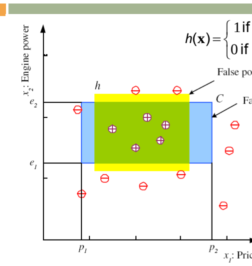
Class C

$$(p_1 \leq \text{price} \leq p_2) \text{ AND } (e_1 \leq \text{engine power} \leq e_2)$$



Hypothesis class $\mathcal{H}$

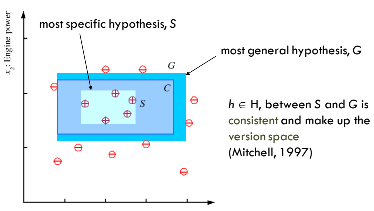
$$h(\mathbf{x}) = \begin{cases} 1 \text{ if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 \text{ if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$



Error of h on $\mathcal{H}$

$$E(h \mid \mathcal{X}) = \sum_{t=1}^{N} 1\left(h(\mathbf{x}^t) \neq r^t\right)$$

S, G, and the Version Space

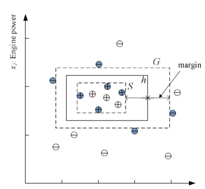


$h \in$ H, between S and G is consistent and make up the version space

(Mitchell, 1997)

Margin

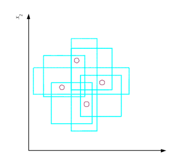
- Choose h with largest margin



VC Dimension

- N points can be labeled in 2^N ways as +/–
- $\mathcal{H}$ shatters N if there exists $h \in \mathcal{H}$ consistent for any of these:

  VC($\mathcal{H}$) = N



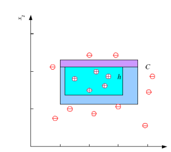
An axis-aligned rectangle shatters 4 points only !

Probably Approximately Correct (PAC) Learning

- How many training examples N should we have, such that with probability at least $1 - \delta$, h has error at most ε ?

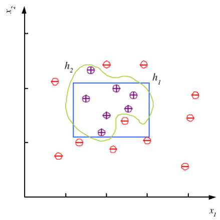
  (Blumer et al., 1989)

- Each strip is at most $\varepsilon/4$
- Pr that we miss a strip $1 - \varepsilon/4$
- Pr that N instances miss a strip $(1 - \varepsilon/4)^N$
- Pr that N instances miss 4 strips $4(1 - \varepsilon/4)^N$
- $4(1 - \varepsilon/4)^N \leq \delta$ and $(1 - x) \leq \exp(-x)$
- $4\exp(-\varepsilon N/4) \leq \delta$ and $N \geq (4/\varepsilon)\log(4/\delta)$

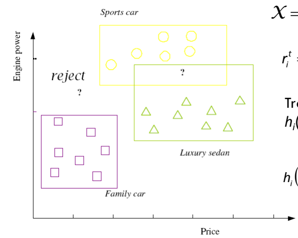


Noise and Model Complexity

Use the simpler one because
- Simpler to use (lower computational complexity)
- Easier to train (lower space complexity)
- Easier to explain (more interpretable)
- Generalizes better (lower variance - Occam's razor)



Multiple Classes, C_i i=1,...,K



$$\mathcal{X} = \{\mathbf{x}^t, \mathbf{r}^t\}_{t=1}^N$$

$$r_i^t = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Train hypotheses $h_i(\mathbf{x})$, $i = 1,...,K$:

$$h_i(\mathbf{x}^t) = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Regression

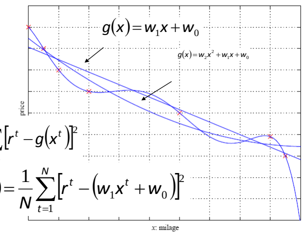
$$\mathcal{X} = \{x^t, r^t\}_{t=1}^N$$

$$r^t \in \Re$$

$$r^t = f(x^t) + \varepsilon$$

$$E(g \mid \mathcal{X}) = \frac{1}{N}\sum_{t=1}^N \left[r^t - g(x^t)\right]^2$$

$$E(w_1, w_0 \mid \mathcal{X}) = \frac{1}{N}\sum_{t=1}^N \left[r^t - (w_1 x^t + w_0)\right]^2$$



Model Selection & Generalization

- Learning is an ill-posed problem; data is not sufficient to find a unique solution
- The need for inductive bias, assumptions about $\mathcal{H}$
- Generalization: How well a model performs on new data
- Overfitting: $\mathcal{H}$ more complex than C or f
- Underfitting: $\mathcal{H}$ less complex than C or f

Triple Trade-Off

- There is a trade-off between three factors (Dietterich, 2003):
  1. Complexity of $\mathcal{H}$, c ($\mathcal{H}$),
  2. Training set size, N,
  3. Generalization error, E, on new data
- As $N\uparrow$, $E\downarrow$
- As c ($\mathcal{H}$)$\uparrow$, first $E\downarrow$ and then $E\uparrow$

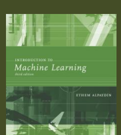
Cross-Validation

- To estimate generalization error, we need data unseen during training. We split the data as
  - Training set (50%)
  - Validation set (25%)
  - Test (publication) set (25%)
- Resampling when there is few data

Dimensions of a Supervised Learner

1. Model: $g(\mathbf{x} \mid \theta)$
2. Loss function: $E(\theta \mid \mathcal{X}) = \sum_t L(r^t, g(\mathbf{x}^t \mid \theta))$
3. Optimization procedure:

$$\theta^* = \arg\min_\theta E(\theta \mid \mathcal{X})$$



Lecture Slides for
INTRODUCTION TO MACHINE LEARNING
3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 3:
BAYESIAN DECISION THEORY

Probability and Inference

- Result of tossing a coin is $\in$ {Heads,Tails}
- Random var $X \in \{1,0\}$

  Bernoulli: $P\{X=1\} = p_o^X (1 - p_o)^{(1-X)}$
- Sample: $\mathbf{X} = \{x^t\}_{t=1}^N$

  Estimation: $p_o = \#\{\text{Heads}\}/\#\{\text{Tosses}\} = \sum_t x^t / N$
- Prediction of next toss:

  Heads if $p_o > ½$, Tails otherwise

Classification

- Credit scoring: Inputs are income and savings.
  Output is low-risk vs high-risk
- Input: $\mathbf{x} = [x_1, x_2]^T$, Output: C Î {0,1}
- Prediction:

$$\text{choose} \begin{cases} C = 1 \text{ if } P(C=1 \mid x_1, x_2) > 0.5 \\ C = 0 \text{ otherwise} \end{cases}$$

or

$$\text{choose} \begin{cases} C = 1 \text{ if } P(C=1 \mid x_1, x_2) > P(C=0 \mid x_1, x_2) \\ C = 0 \text{ otherwise} \end{cases}$$

Bayes' Rule

$$\overbrace{P(C \mid \mathbf{x})}^{\text{posterior}} = \frac{\overbrace{P(C)}^{\text{prior}}\,\overbrace{p(\mathbf{x} \mid C)}^{\text{likelihood}}}{\underbrace{p(\mathbf{x})}_{\text{evidence}}}$$

$$P(C=0) + P(C=1) = 1$$

$$p(\mathbf{x}) = p(\mathbf{x} \mid C=1)P(C=1) + p(\mathbf{x} \mid C=0)P(C=0)$$

$$p(C=0 \mid \mathbf{x}) + P(C=1 \mid \mathbf{x}) = 1$$

Bayes' Rule: $K>2$ Classes

$$P(C_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_i)P(C_i)}{p(\mathbf{x})} = \frac{p(\mathbf{x} \mid C_i)P(C_i)}{\sum_{k=1}^K p(\mathbf{x} \mid C_k)P(C_k)}$$

$$P(C_i) \geq 0 \text{ and } \sum_{i=1}^K P(C_i) = 1$$

$$\text{choose } C_i \text{ if } P(C_i \mid \mathbf{x}) = \max_k P(C_k \mid \mathbf{x})$$

Losses and Risks

- Actions: α_i
- Loss of α_i when the state is C_k : λ_{ik}
- Expected risk (Duda and Hart, 1973)

$$R(\alpha_i \mid \mathbf{x}) = \sum_{k=1}^K \lambda_{ik} P(C_k \mid \mathbf{x})$$

$$\text{choose } \alpha_i \text{ if } R(\alpha_i \mid \mathbf{x}) = \min_k R(\alpha_k \mid \mathbf{x})$$

Losses and Risks: 0/1 Loss

$$\lambda_{ik} = \begin{cases} 0 \text{ if } i = k \\ 1 \text{ if } i \neq k \end{cases}$$

$$R(\alpha_i \mid \mathbf{x}) = \sum_{k=1}^K \lambda_{ik} P(C_k \mid \mathbf{x}) = \sum_{k \neq i} P(C_k \mid \mathbf{x}) = 1 - P(C_i \mid \mathbf{x})$$

For minimum risk, choose the most probable class

Losses and Risks: Reject

$$\lambda_{ik} = \begin{cases} 0 & \text{if } i = k \\ \lambda & \text{if } i = K+1, \quad 0 < \lambda < 1 \\ 1 & \text{otherwise} \end{cases}$$

$$R(\alpha_{K+1} | \mathbf{x}) = \sum_{k=1}^{K} \lambda P(C_k | \mathbf{x}) = \lambda$$

$$R(\alpha_i | \mathbf{x}) = \sum_{k \neq i} P(C_k | \mathbf{x}) = 1 - P(C_i | \mathbf{x})$$

choose C_i if $P(C_i | \mathbf{x}) > P(C_k | \mathbf{x}) \; \forall k \neq i$ and $P(C_i | \mathbf{x}) > 1 - \lambda$

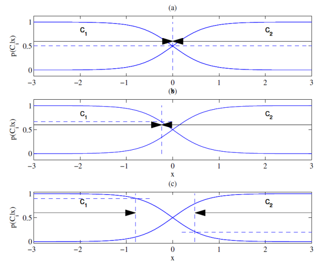
reject otherwise

Different Losses and Reject

Equal losses

Unequal losses

With reject

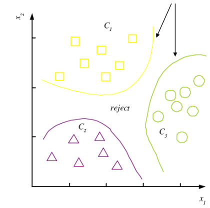


Discriminant Functions

choose C_i if $g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})$

$$g_i(\mathbf{x}) = \begin{cases} -R(\alpha_i | \mathbf{x}) \\ P(C_i | \mathbf{x}) \\ p(\mathbf{x} | C_i) P(C_i) \end{cases}$$

$g_i(\mathbf{x}), i = 1, \ldots, K$



K decision regions $\mathcal{R}_1, \ldots, \mathcal{R}_K$

$$\mathcal{R}_i = \{\mathbf{x} | g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})\}$$

K=2 Classes

- Dichotomizer (K=2) vs Polychotomizer (K>2)
- $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$

$$\text{choose} \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}$$

- *Log odds:* $\log \dfrac{P(C_1 | \mathbf{x})}{P(C_2 | \mathbf{x})}$

Utility Theory

- Prob of state k given exidence $\mathbf{x}$: $P(S_k | \mathbf{x})$
- Utility of α_i when state is k: U_{ik}
- Expected utility:

$$EU(\alpha_i | \mathbf{x}) = \sum_k U_{ik} P(S_k | \mathbf{x})$$

Choose α_i if $EU(\alpha_i | \mathbf{x}) = \max_j EU(\alpha_j | \mathbf{x})$

Association Rules

- Association rule: $X \rightarrow Y$
- *People who buy/click/visit/enjoy X are also likely to buy/click/visit/enjoy Y.*
- A rule implies association, not necessarily causation.

Association measures

- Support ($X \rightarrow Y$):

$$P(X,Y) = \frac{\#\{\text{customers who bought } X \text{ and } Y\}}{\#\{\text{customers}\}}$$

- Confidence ($X \rightarrow Y$):

$$P(Y | X) = \frac{P(X,Y)}{P(X)}$$

$$= \frac{\#\{\text{customers who bought } X \text{ and } Y\}}{\#\{\text{customers who bought } X\}}$$

- Lift ($X \rightarrow Y$):

$$= \frac{P(X,Y)}{P(X)P(Y)} = \frac{P(Y | X)}{P(Y)}$$

Example

Transaction	Items in basket
1	milk, bananas, chocolate
2	milk, chocolate
3	milk, bananas
4	chocolate
5	chocolate
6	milk, chocolate

SOLUTION:

milk → bananas : Support = 2/6, Confidence = 2/4

bananas → milk : Support = 2/6, Confidence = 2/2

milk → chocolate : Support = 3/6, Confidence = 3/4

chocolate → milk : Support = 3/6, Confidence = 3/5

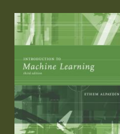
Apriori algorithm (Agrawal et al., 1996)

- For (X,Y,Z), a 3-item set, to be frequent (have enough support), (X,Y), (X,Z), and (Y,Z) should be frequent.
- If (X,Y) is not frequent, none of its supersets can be frequent.
- Once we find the frequent k-item sets, we convert them to rules: X, Y → Z, ...

  and X → Y, Z, ...

L Mon Sep 24 12:39:54 2018 4

i2ml3e-chap04.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr

http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 4:

PARAMETRIC METHODS

Parametric Estimation

- $\mathcal{X} = \{ x^t \}_t$ where $x^t \sim p(x)$
- Parametric estimation:

  Assume a form for $p(x | \theta)$ and estimate θ, its sufficient statistics, using $\mathcal{X}$

  e.g., $N(\mu, \sigma^2)$ where $\theta = \{\mu, \sigma^2\}$

Maximum Likelihood Estimation

- Likelihood of θ given the sample $\mathcal{X}$

  $l(\theta | \mathcal{X}) = p(\mathcal{X} | \theta) = \prod_t p(x^t | \theta)$

- Log likelihood

  $\mathcal{L}(\theta | \mathcal{X}) = \log l(\theta | \mathcal{X}) = \sum_t \log p(x^t | \theta)$

- Maximum likelihood estimator (MLE)

  $\theta^* = \text{argmax}_\theta \mathcal{L}(\theta | \mathcal{X})$

Examples: Bernoulli/Multinomial

- Bernoulli: Two states, failure/success, x in {0,1}

  $P(x) = p_o^{x} (1 - p_o)^{(1-x)}$

  $\mathcal{L}(p_o | \mathcal{X}) = \log \prod_t p_o^{x^t} (1 - p_o)^{(1 - x^t)}$

  MLE: $p_o = \sum_t x^t / N$

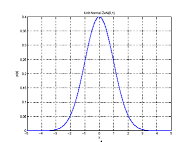
- Multinomial: K>2 states, x_i in {0,1}

  $P(x_1, x_2, \ldots, x_K) = \prod_i p_i^{x_i}$

  $\mathcal{L}(p_1, p_2, \ldots, p_K | \mathcal{X}) = \log \prod_t \prod_i p_i^{x_i^t}$

  MLE: $p_i = \sum_t x_i^t / N$

Gaussian (Normal) Distribution



- $p(x) = \mathcal{N}(\mu, \sigma^2)$

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x - \mu)^2}{2\sigma^2} \right]$$

- MLE for μ and σ^2:

$$m = \frac{\sum_t x^t}{N} \qquad s^2 = \frac{\sum_t (x^t - m)^2}{N}$$

Bias and Variance

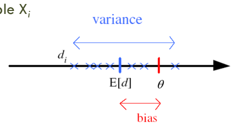
7

Unknown parameter θ

Estimator $d_i = d(X_i)$ on sample X_i

Bias: $b_\theta(d) = E[d] - \theta$

Variance: $E[(d - E[d])^2]$



Mean square error:

$r(d,\theta) = E[(d-\theta)^2]$

$= (E[d] - \theta)^2 + E[(d - E[d])^2]$

$=$ Bias^2 + Variance

Bayes' Estimator

8

- Treat θ as a random var with prior $p(\theta)$
- Bayes' rule: $p(\theta|\mathcal{X}) = p(\mathcal{X}|\theta)\, p(\theta) / p(\mathcal{X})$
- Full: $p(x|\mathcal{X}) = \int p(x|\theta)\, p(\theta|\mathcal{X})\, d\theta$
- Maximum a Posteriori (MAP): $\theta_{MAP} = \text{argmax}_\theta\, p(\theta|\mathcal{X})$
- Maximum Likelihood (ML): $\theta_{ML} = \text{argmax}_\theta\, p(\mathcal{X}|\theta)$
- Bayes': $\theta_{Bayes'} = E[\theta|\mathcal{X}] = \int \theta\, p(\theta|\mathcal{X})\, d\theta$

Bayes' Estimator: Example

- $x^t \sim \mathcal{N}(\theta, \sigma_o^2)$ and $\theta \sim \mathcal{N}(\mu, \sigma^2)$
- $\theta_{ML} = m$
- $\theta_{MAP} = \theta_{Bayes'} =$

$$E[\theta|\mathcal{X}] = \frac{N/\sigma_0^2}{N/\sigma_0^2 + 1/\sigma^2} m + \frac{1/\sigma^2}{N/\sigma_0^2 + 1/\sigma^2} \mu$$

Parametric Classification

10

$$g_i(x) = p(x|C_i)P(C_i)$$

or

$$g_i(x) = \log p(x|C_i) + \log P(C_i)$$

$$p(x|C_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left[-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right]$$

$$g_i(x) = -\frac{1}{2}\log 2\pi - \log\sigma_i - \frac{(x-\mu_i)^2}{2\sigma_i^2} + \log P(C_i)$$

- Given the sample $\mathcal{X} = \{x^t, r^t\}_{t=1}^N$

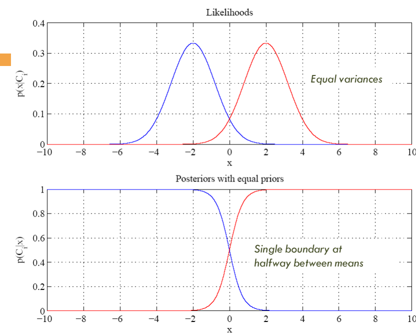
$$x \in \Re \qquad r_i^t = \begin{cases} 1 \text{ if } x^t \in C_i \\ 0 \text{ if } x^t \in C_j, j \neq i \end{cases}$$

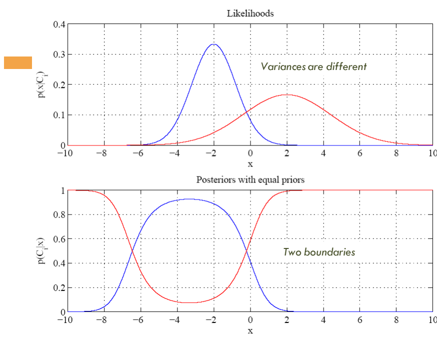
- ML estimates are

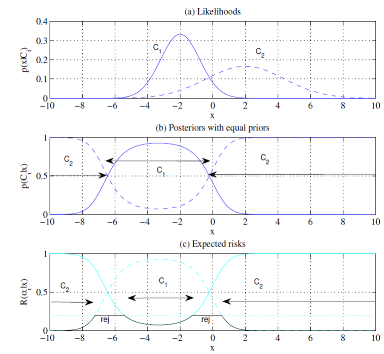
$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N} \quad m_i = \frac{\sum_t x^t r_i^t}{\sum_t r_i^t} \quad s_i^2 = \frac{\sum_t (x^t - m_i)^2 r_i^t}{\sum_t r_i^t}$$

- Discriminant $g_i(x) = -\frac{1}{2}\log 2\pi - \log s_i - \frac{(x-m_i)^2}{2s_i^2} + \log\hat{P}(C_i)$

11







14

Regression

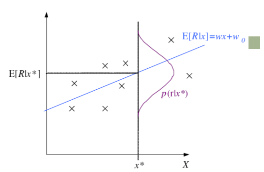
15

$r = f(x) + \varepsilon$

estimator $g(x|\theta)$

$\varepsilon \sim \mathcal{N}(0, \sigma^2)$

$p(r|x) \sim \mathcal{N}(g(x|\theta), \sigma^2)$



$$\mathcal{L}(\theta|\mathcal{X}) = \log\prod_{t=1}^N p(x^t, r^t)$$

$$= \log\prod_{t=1}^N p(r^t|x^t) + \log\prod_{t=1}^N p(x^t)$$

Regression: From LogL to Error

16

$$\mathcal{L}(\theta|\mathcal{X}) = \log\prod_{t=1}^N \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{[r^t - g(x^t|\theta)]^2}{2\sigma^2}\right]$$

$$= -N\log\sqrt{2\pi}\sigma - \frac{1}{2\sigma^2}\sum_{t=1}^N [r^t - g(x^t|\theta)]^2$$

$$E(\theta|\mathcal{X}) = \frac{1}{2}\sum_{t=1}^N [r^t - g(x^t|\theta)]^2$$

Linear Regression

$$g(x^t|w_1, w_0) = w_1 x^t + w_0$$

$$\sum_t r^t = N w_0 + w_1 \sum_t x^t$$

$$\sum_t r^t x^t = w_0 \sum_t x^t + w_1 \sum_t (x^t)^2$$

$$\mathbf{A} = \begin{bmatrix} N & \sum_t x^t \\ \sum_t x^t & \sum_t (x^t)^2 \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} \sum_t r^t \\ \sum_t r^t x^t \end{bmatrix}$$

$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{y}$$

Polynomial Regression

$$g(x^t|w_k, \ldots, w_2, w_1, w_0) = w_k (x^t)^k + \cdots + w_2 (x^t)^2 + w_1 x^t + w_0$$

$$\mathbf{D} = \begin{bmatrix} 1 & x^1 & (x^1)^2 & \cdots & (x^1)^k \\ 1 & x^2 & (x^2)^2 & \cdots & (x^2)^k \\ \vdots & & & & \\ 1 & x^N & (x^N)^2 & \cdots & (x^N)^k \end{bmatrix} \quad \mathbf{r} = \begin{bmatrix} r^1 \\ r^2 \\ \vdots \\ r^N \end{bmatrix}$$

$$\mathbf{w} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{r}$$

Other Error Measures

19

- Square Error: $E(\theta|\mathcal{X}) = \frac{1}{2}\sum_{t=1}^N [r^t - g(x^t|\theta)]^2$
- Relative Square Error: $E(\theta|\mathcal{X}) = \frac{\sum_{t=1}^N [r^t - g(x^t|\theta)]^2}{\sum_{t=1}^N [r^t - \bar{r}]^2}$
- Absolute Error: $E(\theta|X) = \sum_t |r^t - g(x^t|\theta)|$
- ε-sensitive Error: $E(\theta|X) = \sum_t 1(|r^t - g(x^t|\theta)| > \varepsilon)\,(|r^t - g(x^t|\theta)| - \varepsilon)$

Bias and Variance

20

$$E\left[(r - g(x))^2 | x\right] = \underbrace{E\left[(r - E[r|x])^2 | x\right]}_{noise} + \underbrace{(E[r|x] - g(x))^2}_{squared\ error}$$

$$E_{\mathcal{X}}\left[(E[r|x] - g(x))^2 | x\right] = \underbrace{(E[r|x] - E_{\mathcal{X}}[g(x)])^2}_{bias} + \underbrace{E_{\mathcal{X}}\left[(g(x) - E_{\mathcal{X}}[g(x)])^2\right]}_{variance}$$

Estimating Bias and Variance

21

- M samples $X_i = \{x^t_i, r^t_i\}$, $i = 1, \ldots, M$ are used to fit $g_i(x)$, $i = 1, \ldots, M$

$$\text{Bias}^2(g) = \frac{1}{N}\sum_t [\bar{g}(x^t) - f(x^t)]^2$$

$$\text{Variance}(g) = \frac{1}{NM}\sum_t\sum_i [g_i(x^t) - \bar{g}(x^t)]^2$$

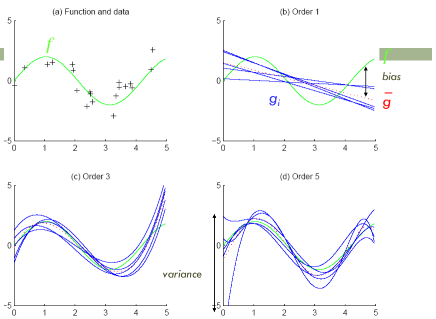
$$\bar{g}(x) = \frac{1}{M}\sum_t g_i(x)$$

Bias/Variance Dilemma

22

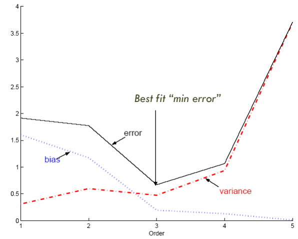
- Example: $g_i(x) = 2$ has no variance and high bias
  $g_i(x) = \sum_t r^t_i / N$ has lower bias with variance
- As we increase complexity,
  bias decreases (a better fit to data) and
  variance increases (fit varies more with data)
- Bias/Variance dilemma: (Geman et al., 1992)

23

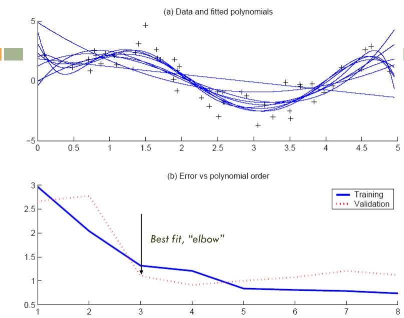


Polynomial Regression

24



25



Model Selection

26

- Cross-validation: Measure generalization accuracy by testing on data unused during training
- Regularization: Penalize complex models

  E'=error on data + λ model complexity

  Akaike's information criterion (AIC), Bayesian information criterion (BIC)
- Minimum description length (MDL): Kolmogorov complexity, shortest description of data
- Structural risk minimization (SRM)

Bayesian Model Selection

27

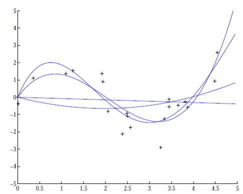
- Prior on models, $p(\text{model})$

$$p(\text{model}\mid\text{data})=\frac{p(\text{data}\mid\text{model})\,p(\text{model})}{p(\text{data})}$$

- Regularization, when prior favors simpler models
- Bayes, MAP of the posterior, $p(\text{model}\mid\text{data})$
- Average over a number of models with high posterior (voting, ensembles: Chapter 17)

Regression example

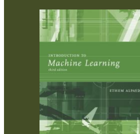
28



Coefficients increase in magnitude as order increases:
1: [-0.0769, 0.0016]
2: [0.1682, -0.6657, 0.0080]
3: [0.4238, -2.5778, 3.4675, -0.0002
4: [-0.1093, 1.4356, -5.5007, 6.0454, -0.0019]

Regularization (L2): $E(\mathbf{w}\mid\mathcal{X})=\frac{1}{2}\sum_{t=1}^{N}\left[r^t-g(x^t\mid\mathbf{w})\right]^2+\lambda\sum_i w_i^2$

i2ml3e-chap05.pdf



Lecture Slides for
INTRODUCTION TO MACHINE LEARNING
3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 5:
MULTIVARIATE METHODS

Multivariate Data

3

- Multiple measurements (sensors)
- d inputs/features/attributes: d-variate
- N instances/observations/examples

$$\mathbf{X}=\begin{bmatrix} X_1^1 & X_2^1 & \cdots & X_d^1 \\ X_1^2 & X_2^2 & \cdots & X_d^2 \\ \vdots & & & \\ X_1^N & X_2^N & \cdots & X_d^N \end{bmatrix}$$

Multivariate Parameters

4

$$\text{Mean}: E[\mathbf{x}]=\boldsymbol{\mu}=[\mu_1,\ldots,\mu_d]^T$$
$$\text{Covariance}: \sigma_{ij}\equiv\text{Cov}(X_i,X_j)$$
$$\text{Correlation}: \text{Corr}(X_i,X_j)\equiv\rho_{ij}=\frac{\sigma_{ij}}{\sigma_i\sigma_j}$$

$$\Sigma\equiv\text{Cov}(\mathbf{X})=E\left[(\mathbf{X}-\boldsymbol{\mu})(\mathbf{X}-\boldsymbol{\mu})^T\right]=\begin{bmatrix}\sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1d}\\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2d}\\ \vdots & & & \\ \sigma_{d1} & \sigma_{d2} & \cdots & \sigma_d^2\end{bmatrix}$$

Parameter Estimation

$$\text{Sample mean } \mathbf{m}: m_i=\frac{\sum_{t=1}^N x_i^t}{N}, i=1,\ldots,d$$

$$\text{Covariance matrix } \mathbf{S}: s_{ij}=\frac{\sum_{t=1}^N (x_i^t-m_i)(x_j^t-m_j)}{N}$$

$$\text{Correlation matrix } \mathbf{R}: r_{ij}=\frac{s_{ij}}{s_i s_j}$$

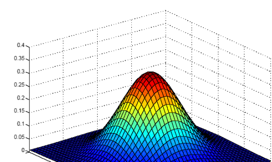
Estimation of Missing Values

6

- What to do if certain instances have missing attributes?
- Ignore those instances: not a good idea if the sample is small
- Use 'missing' as an attribute: may give information
- Imputation: Fill in the missing value
  - Mean imputation: Use the most likely value (e.g., mean)
  - Imputation by regression: Predict based on other attributes

Multivariate Normal Distribution

7



$$\mathbf{x}\sim\mathcal{N}_d(\boldsymbol{\mu},\Sigma)$$

$$p(\mathbf{x})=\frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

Multivariate Normal Distribution

8

- Mahalanobis distance: $(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})$

  measures the distance from $\mathbf{x}$ to $\boldsymbol{\mu}$ in terms of Σ (normalizes for difference in variances and correlations)
- Bivariate: $d = 2$

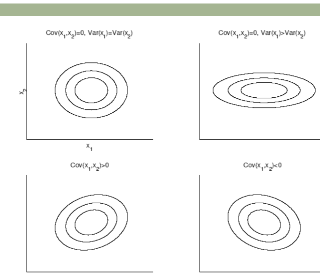
$$\Sigma=\begin{bmatrix}\sigma_1^2 & \rho\sigma_1\sigma_2\\ \rho\sigma_1\sigma_2 & \sigma_2^2\end{bmatrix}$$

$$p(x_1,x_2)=\frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left[-\frac{1}{2(1-\rho^2)}\left(z_1^2-2\rho z_1z_2+z_2^2\right)\right]$$

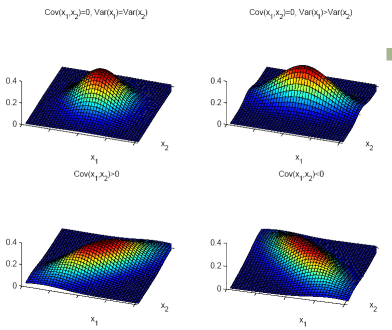
$$z_i=(x_i-\mu_i)/\sigma_i$$

Bivariate Normal

9



10



Independent Inputs: Naive Bayes

11

- If x_i are independent, offdiagonals of Σ are 0, Mahalanobis distance reduces to weighted (by $1/\sigma_i$) Euclidean distance:

$$p(\mathbf{x})=\prod_{i=1}^{d} p_i(x_i)=\frac{1}{(2\pi)^{d/2}\prod_{i=1}^{d}\sigma_i}\exp\left[-\frac{1}{2}\sum_{i=1}^{d}\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2\right]$$

- If variances are also equal, reduces to Euclidean distance

Parametric Classification

- If $p(\mathbf{x} \mid C_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$

$$p(\mathbf{x}\mid C_i)=\frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right]$$

- Discriminant functions

$$g_i(\mathbf{x})=\log p(\mathbf{x}\mid C_i)+\log P(C_i)$$
$$=-\frac{d}{2}\log 2\pi-\frac{1}{2}\log|\Sigma_i|-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)+\log P(C_i)$$

Estimation of Parameters

13

$$\hat{P}(C_i)=\frac{\sum_t r_i^t}{N}$$
$$\mathbf{m}_i=\frac{\sum_t r_i^t\mathbf{x}^t}{\sum_t r_i^t}$$
$$\mathbf{S}_i=\frac{\sum_t r_i^t(\mathbf{x}^t-\mathbf{m}_i)(\mathbf{x}^t-\mathbf{m}_i)^T}{\sum_t r_i^t}$$
$$g_i(\mathbf{x})=-\frac{1}{2}\log|\mathbf{S}_i|-\frac{1}{2}(\mathbf{x}-\mathbf{m}_i)^T\mathbf{S}_i^{-1}(\mathbf{x}-\mathbf{m}_i)+\log\hat{P}(C_i)$$

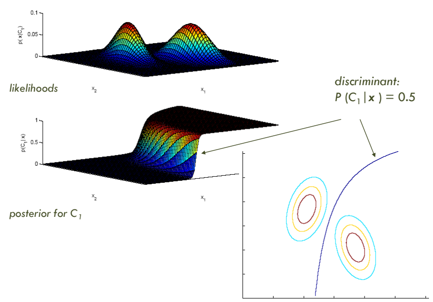
Different $\mathbf{S}_i$

- Quadratic discriminant

$$g_i(\mathbf{x})=-\frac{1}{2}\log|\mathbf{S}_i|-\frac{1}{2}\left(\mathbf{x}^T\mathbf{S}_i^{-1}\mathbf{x}-2\mathbf{x}^T\mathbf{S}_i^{-1}\mathbf{m}_i+\mathbf{m}_i^T\mathbf{S}_i^{-1}\mathbf{m}_i\right)+\log\hat{P}(C_i)$$
$$=\mathbf{x}^T\mathbf{W}_i\mathbf{x}+\mathbf{w}_i^T\mathbf{x}+w_{i0}$$

where

$$\mathbf{W}_i=-\frac{1}{2}\mathbf{S}_i^{-1}$$
$$\mathbf{w}_i=\mathbf{S}_i^{-1}\mathbf{m}_i$$
$$w_{i0}=-\frac{1}{2}\mathbf{m}_i^T\mathbf{S}_i^{-1}\mathbf{m}_i-\frac{1}{2}\log|\mathbf{S}_i|+\log\hat{P}(C_i)$$



15

Common Covariance Matrix $\mathbf{S}$

16

- Shared common sample covariance $\mathbf{S}$

$$\mathbf{S}=\sum_i\hat{P}(C_i)\mathbf{S}_i$$

- Discriminant reduces to

$$g_i(\mathbf{x})=-\frac{1}{2}(\mathbf{x}-\boldsymbol{m}_i)^T\mathbf{S}^{-1}(\mathbf{x}-\boldsymbol{m}_i)+\log\hat{P}(C_i)$$

which is a linear discriminant

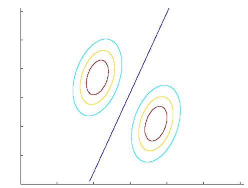
$$g_i(\mathbf{x})=\mathbf{w}_i^T\mathbf{x}+w_{i0}$$

where

$$\mathbf{w}_i=\mathbf{S}^{-1}\mathbf{m}_i \quad w_{i0}=-\frac{1}{2}\mathbf{m}_i^T\mathbf{S}^{-1}\mathbf{m}_i+\log\hat{P}(C_i)$$

Common Covariance Matrix $\mathbf{S}$

17



Diagonal $\mathbf{S}$

18

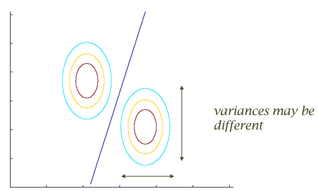
- When $x_j\ j=1,..d$, are independent, Σ is diagonal $p(\mathbf{x}\mid C_i)=\prod_j p(x_j\mid C_i)$ (Naive Bayes' assumption)

$$g_i(\mathbf{x})=-\frac{1}{2}\sum_{j=1}^{d}\left(\frac{x_j^t-m_{ij}}{s_j}\right)^2+\log\hat{P}(C_i)$$

Classify based on weighted Euclidean distance (in s_j units) to the nearest mean

Diagonal $\mathbf{S}$

19



Diagonal $\mathbf{S}$, equal variances

20

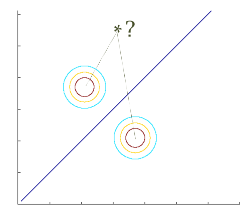
- Nearest mean classifier: Classify based on Euclidean distance to the nearest mean

$$g_i(\mathbf{x})=-\frac{\|\mathbf{x}-\mathbf{m}_i\|^2}{2s^2}+\log\hat{P}(C_i)$$
$$=-\frac{1}{2s^2}\sum_{j=1}^{d}\left(x_j^t-m_{ij}\right)^2+\log\hat{P}(C_i)$$

- Each mean can be considered a prototype or template and this is template matching

Diagonal $\mathbf{S}$, equal variances

21

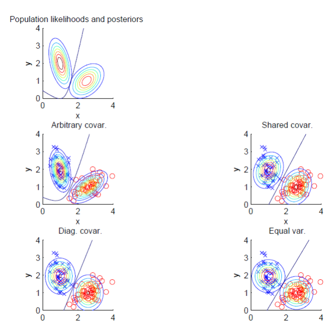


Model Selection

22

Assumption	Covariance matrix	No of parameters
Shared, Hyperspheric	$\mathbf{S}_i=\mathbf{S}=s^2\mathbf{I}$	1
Shared, Axis-aligned	$\mathbf{S}_i=\mathbf{S}$, with $s_{ij}=0$	d
Shared, Hyperellipsoidal	$\mathbf{S}_i=\mathbf{S}$	$d(d+1)/2$
Different, Hyperellipsoidal	$\mathbf{S}_i$	$K\ d(d+1)/2$

- As we increase complexity (less restricted $\mathbf{S}$), bias decreases and variance increases
- Assume simple models (allow some bias) to control variance (regularization)



23

Discrete Features

24

- Binary features: $p_{ij}\equiv p(x_j=1\mid C_i)$

if x_j are independent (Naive Bayes')

$$p(x\mid C_i)=\prod_{j=1}^{d}p_{ij}^{x_j}(1-p_{ij})^{(1-x_j)}$$

the discriminant is linear

$$g_i(\mathbf{x})=\log p(\mathbf{x}\mid C_i)+\log P(C_i)$$
$$=\sum_j\left[x_j\log p_{ij}+(1-x_j)\log(1-p_{ij})\right]+\log P(C_i)$$

Estimated parameters $\quad \hat{p}_{ij}=\frac{\sum_t x_j^t r_i^t}{\sum_t r_i^t}$

Discrete Features

25

- Multinomial (1-of-n_j) features: $x_j \in \{v_1, v_2,..., v_{n_j}\}$

$$p_{ijk}\equiv p(z_{jk}=1\mid C_i)=p(x_j=v_k\mid C_i)$$

if x_j are independent

$$p(x\mid C_i)=\prod_{j=1}^{d}\prod_{k=1}^{n_j}p_{ijk}^{z_{jk}}$$
$$g_i(\mathbf{x})=\sum_j\sum_k z_{jk}\log p_{ijk}+\log P(C_i)$$
$$\hat{p}_{ijk}=\frac{\sum_t z_{jk}^t r_i^t}{\sum_t r_i^t}$$

Multivariate Regression

$$r^t = g(x^t \mid w_0, w_1, \ldots, w_d) + \varepsilon$$

Multivariate linear model

$$w_0 + w_1 x_1^t + w_2 x_2^t + \cdots + w_d x_d^t$$

$$E(w_0, w_1, \ldots, w_d \mid X) = \frac{1}{2}\sum_t \left[r^t - w_0 - w_1 x_1^t - \cdots - w_d x_d^t\right]^2$$

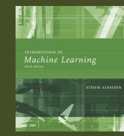
Multivariate polynomial model:

Define new higher-order variables

$z_1 = x_1$, $z_2 = x_2$, $z_3 = x_1^2$, $z_4 = x_2^2$, $z_5 = x_1 x_2$

and use the linear model in this new z space

(basis functions, kernel trick: Chapter 13)



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr

http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 6:

DIMENSIONALITY REDUCTION

Why Reduce Dimensionality?

- Reduces time complexity: Less computation
- Reduces space complexity: Fewer parameters
- Saves the cost of observing the feature
- Simpler models are more robust on small datasets
- More interpretable; simpler explanation
- Data visualization (structure, groups, outliers, etc) if plotted in 2 or 3 dimensions

Feature Selection vs Extraction

- Feature selection: Choosing $k<d$ important features, ignoring the remaining $d-k$

  Subset selection algorithms
- Feature extraction: Project the

  original x_i , $i = 1,\ldots,d$ dimensions to

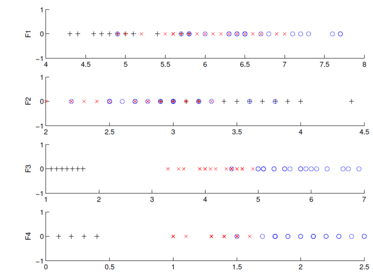
  new $k<d$ dimensions, z_j , $j = 1,\ldots,k$

Subset Selection

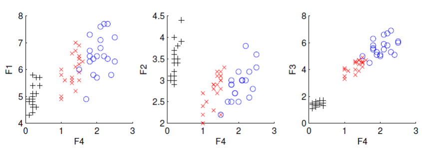
- There are 2^d subsets of d features
- Forward search: Add the best feature at each step
  - Set of features F initially Ø.
  - At each iteration, find the best new feature

    $j = \text{argmin}_i\ E(F \cup x_i)$
  - Add x_j to F if $E(F \cup x_j) < E(F)$
- Hill-climbing $O(d^2)$ algorithm
- Backward search: Start with all features and remove one at a time, if possible.
- Floating search (Add k, remove l)

Iris data: Single feature



Iris data: Add one more feature to F4



Principal Components Analysis

- Find a low-dimensional space such that when $\mathbf{x}$ is projected there, information loss is minimized.
- The projection of $\mathbf{x}$ on the direction of $\mathbf{w}$ is: $z = \mathbf{w}^T\mathbf{x}$
- Find $\mathbf{w}$ such that Var(z) is maximized

  $\text{Var}(z) = \text{Var}(\mathbf{w}^T\mathbf{x}) = E[(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\boldsymbol{\mu})^2]$

  $= E[(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\boldsymbol{\mu})(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\boldsymbol{\mu})]$

  $= E[\mathbf{w}^T(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T\mathbf{w}]$

  $= \mathbf{w}^T E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T]\mathbf{w} = \mathbf{w}^T \sum \mathbf{w}$

  where $\text{Var}(\mathbf{x}) = E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] = \sum$

- Maximize Var(z) subject to $\|\mathbf{w}\|=1$

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \Sigma \mathbf{w}_1 - \alpha(\mathbf{w}_1^T\mathbf{w}_1 - 1)$$

$\sum \mathbf{w}_1 = \alpha \mathbf{w}_1$ that is, $\mathbf{w}_1$ is an eigenvector of $\sum$

Choose the one with the largest eigenvalue for Var(z) to be max

- Second principal component: Max Var(z_2), s.t., $\|\mathbf{w}_2\|=1$ and orthogonal to $\mathbf{w}_1$

$$\max_{\mathbf{w}_2} \mathbf{w}_2^T \Sigma \mathbf{w}_2 - \alpha(\mathbf{w}_2^T\mathbf{w}_2 - 1) - \beta(\mathbf{w}_2^T\mathbf{w}_1 - 0)$$

$\sum \mathbf{w}_2 = \alpha\ \mathbf{w}_2$ that is, $\mathbf{w}_2$ is another eigenvector of $\sum$

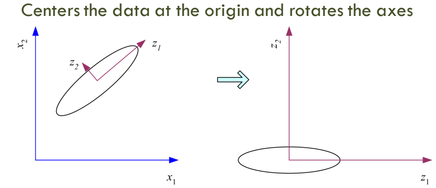
and so on.

What PCA does

$$\mathbf{z} = \mathbf{W}^T(\mathbf{x} - \mathbf{m})$$

where the columns of $\mathbf{W}$ are the eigenvectors of $\sum$ and $\mathbf{m}$ is sample mean

Centers the data at the origin and rotates the axes

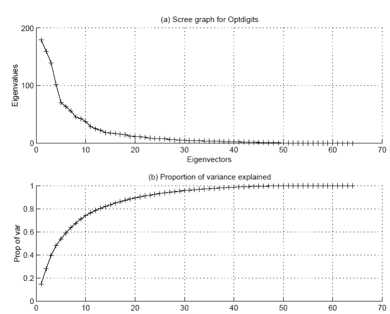


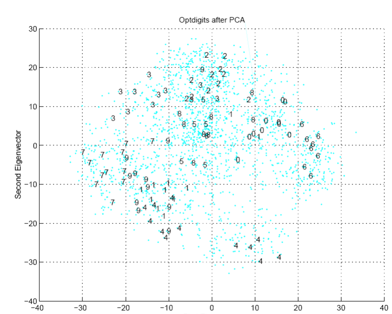
How to choose k ?

- Proportion of Variance (PoV) explained

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

when λ_i are sorted in descending order
- Typically, stop at PoV>0.9
- Scree graph plots of PoV vs k, stop at "elbow"





Feature Embedding

- When $\boldsymbol{X}$ is the Nxd data matrix,

$\boldsymbol{X}^T\boldsymbol{X}$ is the dxd matrix (covariance of features, if mean-centered)

$\boldsymbol{X}\boldsymbol{X}^T$ is the NxN matrix (pairwise similarities of instances)

- PCA uses the eigenvectors of $\boldsymbol{X}^T\boldsymbol{X}$ which are d-dim and can be used for projection
- Feature embedding uses the eigenvectors of $\boldsymbol{X}\boldsymbol{X}^T$ which are N-dim and which give directly the coordinates after projection
- Sometimes, we can define pairwise similarities (or distances) between instances, then we can use feature embedding without needing to represent instances as vectors.

Factor Analysis

15

- Find a small number of factors $\mathbf{z}$, which when combined generate $\boldsymbol{x}$:

$$x_i - \mu_i = v_{i1}z_1 + v_{i2}z_2 + \dots + v_{ik}z_k + \varepsilon_i$$

where z_j, $j = 1,\dots,k$ are the latent factors with

$E[\,z_j\,]=0$, $\mathrm{Var}(z_j)=1$, $\mathrm{Cov}(z_i, z_j)=0$, $i \neq j$,

ε_i are the noise sources

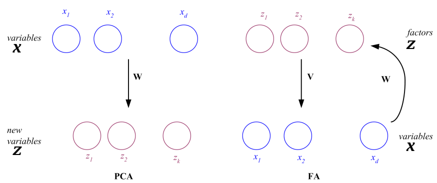
$E[\,\varepsilon_i\,]= \psi_i$, $\mathrm{Cov}(\varepsilon_i, \varepsilon_j) =0$, $i \neq j$, $\mathrm{Cov}(\varepsilon_i, z_j) =0$,

and v_{ij} are the factor loadings

PCA vs FA

16

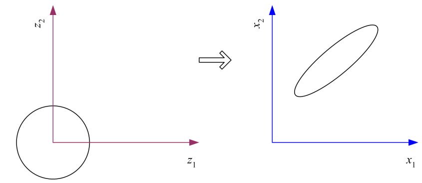
- PCA From $\boldsymbol{x}$ to $\boldsymbol{z}$ $\quad \boldsymbol{z} = \mathbf{W}^T(\boldsymbol{x} - \boldsymbol{\mu})$
- FA From $\boldsymbol{z}$ to $\boldsymbol{x}$ $\quad \boldsymbol{x} - \boldsymbol{\mu} = \mathbf{V}\boldsymbol{z} + \boldsymbol{\varepsilon}$



Factor Analysis

17

- In FA, factors z_j are stretched, rotated and translated to generate $\boldsymbol{x}$



Singular Value Decomposition and Matrix Factorization

18

- Singular value decomposition: $\boldsymbol{X}=\boldsymbol{VAW}^T$

$\boldsymbol{V}$ is NxN and contains the eigenvectors of $\boldsymbol{XX}^T$

$\boldsymbol{W}$ is dxd and contains the eigenvectors of $\boldsymbol{X}^T\boldsymbol{X}$

and $\mathbf{A}$ is Nxd and contains singular values on its first k diagonal

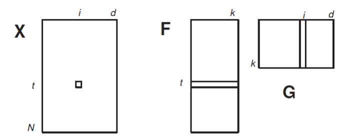
- $\boldsymbol{X}=\boldsymbol{u}_1 a_1 \boldsymbol{v}_1^T+\dots+\boldsymbol{u}_k a_k \boldsymbol{v}_k^T$ where k is the rank of $\boldsymbol{X}$

Matrix Factorization

19

- Matrix factorization: $\boldsymbol{X}=\boldsymbol{FG}$

$\boldsymbol{F}$ is Nxk and $\boldsymbol{G}$ is kxd



$$X_{ti} = \mathbf{F}_t^T \mathbf{G}_i = \sum_{j=1}^{k} \mathbf{F}_{tj}\mathbf{G}_{ji}$$

Latent semantic indexing

Multidimensional Scaling

20

- Given pairwise distances between N points,

d_{ij}, $i,j = 1,\dots,N$

place on a low-dim map s.t. distances are preserved (by feature embedding)

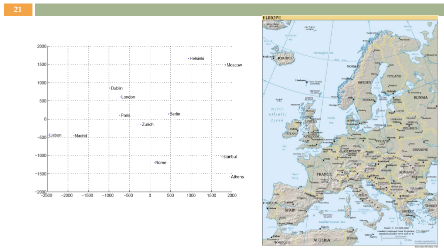
- $\mathbf{z} = \mathbf{g}(\mathbf{x} \mid \theta)$ Find θ that min Sammon stress

$$E(\theta \mid \mathcal{X}) = \sum_{r,s} \frac{\left(\|\mathbf{z}^r - \mathbf{z}^s\| - \|\mathbf{x}^r - \mathbf{x}^s\|\right)^2}{\|\mathbf{x}^r - \mathbf{x}^s\|^2}$$

$$= \sum_{r,s} \frac{\left(\|\mathbf{g}(\mathbf{x}^r \mid \theta) - \mathbf{g}(\mathbf{x}^s \mid \theta)\| - \|\mathbf{x}^r - \mathbf{x}^s\|\right)^2}{\|\mathbf{x}^r - \mathbf{x}^s\|^2}$$

Map of Europe by MDS

21



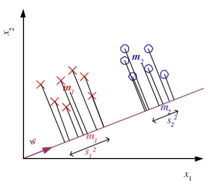
Map from CIA – The World Factbook: http://www.cia.gov/

Linear Discriminant Analysis

- Find a low-dimensional space such that when $\boldsymbol{x}$ is projected, classes are well-separated.
- Find $\boldsymbol{w}$ that maximizes

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

$$m_1 = \frac{\sum_t \mathbf{w}^T \mathbf{x}^t r^t}{\sum_t r^t} \qquad s_1^2 = \sum_t \left(\mathbf{w}^T \mathbf{x}^t - m_1\right)^2 r^t$$



- Between-class scatter:

$$(m_1 - m_2)^2 = \left(\mathbf{w}^T\mathbf{m}_1 - \mathbf{w}^T\mathbf{m}_2\right)^2$$
$$= \mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T \mathbf{w}$$
$$= \mathbf{w}^T \mathbf{S}_B \mathbf{w} \text{ where } \mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$$

- Within-class scatter:

$$s_1^2 = \sum_t \left(\mathbf{w}^T\mathbf{x}^t - m_1\right)^2 r^t$$
$$= \sum_t \mathbf{w}^T(\mathbf{x}^t - \mathbf{m}_1)(\mathbf{x}^t - \mathbf{m}_1)^T \mathbf{w} r^t = \mathbf{w}^T \mathbf{S}_1 \mathbf{w}$$

$$\text{where } \mathbf{S}_1 = \sum_t (\mathbf{x}^t - \mathbf{m}_1)(\mathbf{x}^t - \mathbf{m}_1)^T r^t$$

$$s_1^2 + s_2^2 = \mathbf{w}^T \mathbf{S}_W \mathbf{w} \text{ where } \mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$$

23

Fisher's Linear Discriminant

24

- Find $\boldsymbol{w}$ that max

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} = \frac{\left|\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)\right|^2}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

- LDA soln:

$$\mathbf{w} = c \cdot \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$$

- Parametric soln:

$$\mathbf{w} = \Sigma^{-1}(\mu_1 - \mu_2)$$
$$\text{when } p(\mathbf{x} \mid C_i) \sim \mathcal{N}(\mu_i, \Sigma)$$

K>2 Classes

25

- Within-class scatter:

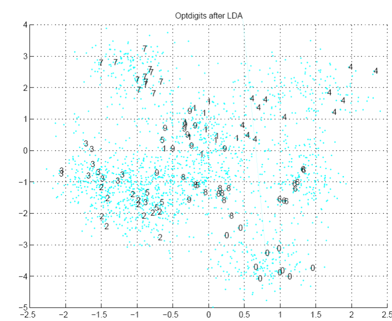
$$\mathbf{S}_W = \sum_{i=1}^{K} \mathbf{S}_i \qquad \mathbf{S}_i = \sum_t r_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T$$

- Between-class scatter:

$$\mathbf{S}_B = \sum_{i=1}^{K} N_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \qquad \mathbf{m} = \frac{1}{K}\sum_{i=1}^{K} \mathbf{m}_i$$

- Find $\mathbf{W}$ that max $\quad J(\mathbf{W}) = \frac{\left|\mathbf{W}^T \mathbf{S}_B \mathbf{W}\right|}{\left|\mathbf{W}^T \mathbf{S}_W \mathbf{W}\right|}$

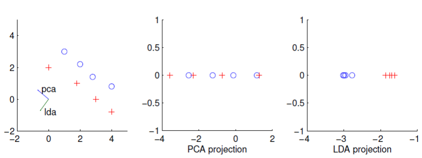
The largest eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$; maximum rank of K-1



26

PCA vs LDA

27



Canonical Correlation Analysis

28

- $\boldsymbol{X}=\{\boldsymbol{x}^t,\boldsymbol{y}^t\}_t$; two sets of variables $\boldsymbol{x}$ and $\boldsymbol{y}$ $\boldsymbol{x}$
- We want to find two projections $\boldsymbol{w}$ and $\boldsymbol{v}$ st when $\boldsymbol{x}$ is projected along $\boldsymbol{w}$ and $\boldsymbol{y}$ is projected along $\boldsymbol{v}$, the correlation is maximized:

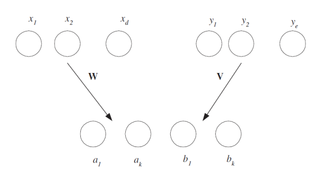
$$\rho = \mathrm{Corr}(\boldsymbol{w}^T\boldsymbol{x}, \boldsymbol{v}^T\boldsymbol{y}) = \frac{\mathrm{Cov}(\boldsymbol{w}^T\boldsymbol{x}, \boldsymbol{v}^T\boldsymbol{y})}{\sqrt{\mathrm{Var}(\boldsymbol{w}^T\boldsymbol{x})}\sqrt{\mathrm{Var}(\boldsymbol{v}^T\boldsymbol{y})}}$$

$$= \frac{\boldsymbol{w}^T\mathrm{Cov}(\boldsymbol{x},\boldsymbol{y})\boldsymbol{v}}{\sqrt{\boldsymbol{w}^T\mathrm{Var}(\boldsymbol{x})\boldsymbol{w}}\sqrt{\boldsymbol{v}^T\mathrm{Var}(\boldsymbol{y})\boldsymbol{v}}} = \frac{\boldsymbol{w}^T\mathrm{S}_{xy}\boldsymbol{v}}{\sqrt{\boldsymbol{w}^T\mathrm{S}_{xx}\boldsymbol{w}}\sqrt{\boldsymbol{v}^T\mathrm{S}_{yy}\boldsymbol{v}}}$$

CCA

29

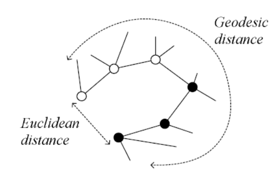
- $\boldsymbol{x}$ and $\boldsymbol{y}$ may be two different views or modalities; e.g., image and word tags, and CCA does a joint mapping



Isomap

30

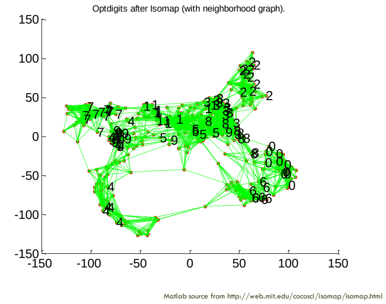
- Geodesic distance is the distance along the manifold that the data lies in, as opposed to the Euclidean distance in the input space



Isomap

31

- Instances r and s are connected in the graph if $||x^r - x^s|| < \varepsilon$ or if x^s is one of the k neighbors of x^r. The edge length is $||x^r - x^s||$
- For two nodes r and s not connected, the distance is equal to the shortest path between them
- Once the $N \times N$ distance matrix is thus formed, use MDS to find a lower-dimensional mapping



Matlab source from http://web.mit.edu/cocosci/isomap/isomap.html

32

Locally Linear Embedding

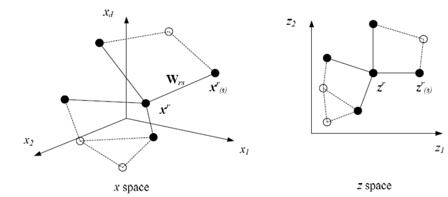
33

1. Given $\mathbf{x}^r$ find its neighbors $\mathbf{x}^s_{(r)}$
2. Find $\mathbf{W}_{rs}$ that minimize

$$E(\mathbf{W}|X) = \sum_r \left\| \mathbf{x}^r - \sum_s \mathbf{W}_{rs}\mathbf{x}^s_{(r)} \right\|^2$$

3. Find the new coordinates $\mathbf{z}^r$ that minimize

$$E(\mathbf{z}|\mathbf{W}) = \sum_r \left\| z^r - \sum_s \mathbf{W}_{rs} z^s_{(r)} \right\|^2$$



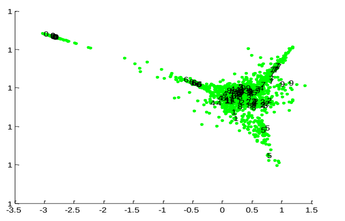
34

L Mon Sep 24 12:39:54 2018 7

i2ml3e-chap07.pdf

LLE on Optdigits

35



Matlab source from http://www.cs.toronto.edu/~roweis/lle/code.html

Laplacian Eigenmaps

36

- Let r and s be two instances and B_{rs} is their similarity, we want to find $\mathbf{z}^r$ and $\mathbf{z}^s$ that

$$\min \sum_{r,s} \|\mathbf{z}^r - \mathbf{z}^s\|^2 B_{rs}$$

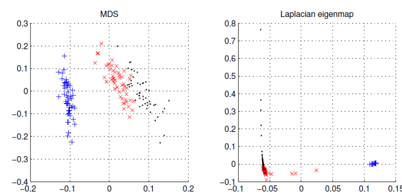
- B_{rs} can be defined in terms of similarity in an original space: 0 if $\mathbf{x}^r$ and $\mathbf{x}^s$ are too far, otherwise

$$B_{rs} = \exp\left[-\frac{\|\mathbf{x}^r - \mathbf{x}^s\|^2}{2\sigma^2}\right]$$

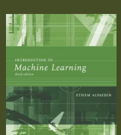
- Defines a graph Laplacian, and feature embedding returns $\mathbf{z}^r$

Laplacian Eigenmaps on Iris

37



Spectral clustering (chapter 7)



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 7:

CLUSTERING

Semiparametric Density Estimation

3

- Parametric: Assume a single model for $p(\mathbf{x} | C_i)$ (Chapters 4 and 5)
- Semiparametric: $p(\mathbf{x}|C_i)$ is a mixture of densities
  Multiple possible explanations/prototypes:
  Different handwriting styles, accents in speech
- Nonparametric: No model; data speaks for itself (Chapter 8)

Mixture Densities

4

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x}|G_i)P(G_i)$$

where G_i the components/groups/clusters,

$P(G_i)$ mixture proportions (priors),

$p(\mathbf{x}|G_i)$ component densities

Gaussian mixture where $p(\mathbf{x}|G_i) \sim N(\boldsymbol{\mu}_i, \sum_i)$

parameters $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \sum_i\}_{i=1}^k$

unlabeled sample $X=\{\mathbf{x}^t\}_t$ (unsupervised learning)

Classes vs. Clusters

5

- Supervised: $X = \{\mathbf{x}^t, \mathbf{r}^t\}_t$
- Classes C_i $i=1,...,K$

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x}|C_i)P(C_i)$$

where $p(\mathbf{x}|C_i) \sim N(\boldsymbol{\mu}_i, \sum_i)$

- $\Phi = \{P(C_i), \boldsymbol{\mu}_i, \sum_i\}_{i=1}^K$

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N} \quad \mathbf{m}_i = \frac{\sum_t r_i^t \mathbf{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T}{\sum_t r_i^t}$$

- Unsupervised : $X = \{\mathbf{x}^t\}_t$
- Clusters G_i $i=1,...,k$

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x}|G_i)P(G_i)$$

where $p(\mathbf{x}|G_i) \sim N(\boldsymbol{\mu}_i, \sum_i)$

- $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \sum_i\}_{i=1}^k$

Labels r^t_i ?

k-Means Clustering

6

- Find k reference vectors (prototypes/codebook vectors/codewords) which best represent data
- Reference vectors, $\mathbf{m}_j$, $j = 1,...,k$
- Use nearest (most similar) reference:

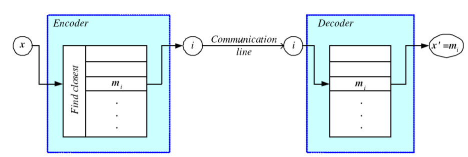
$$\|\mathbf{x}^t - \mathbf{m}_i\| = \min_j \|\mathbf{x}^t - \mathbf{m}_j\|$$

- Reconstruction error $E\left(\{\mathbf{m}_i\}_{i=1}^k | \mathcal{X}\right) = \sum_t \sum_i b_i^t \|\mathbf{x}^t - \mathbf{m}_i\|$

$$b_i^t = \begin{cases} 1 & \text{if } \|\mathbf{x}^t - \mathbf{m}_i\| = \min_j \|\mathbf{x}^t - \mathbf{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

Encoding/Decoding

7



k-means Clustering

8

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

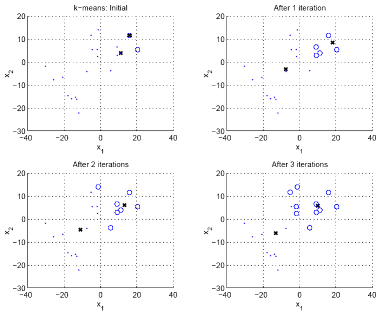
For all $\boldsymbol{x}^t \in \mathcal{X}$

$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

For all $\boldsymbol{m}_i, i = 1, \ldots, k$

$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

Until $\boldsymbol{m}_i$ converge



9

Expectation-Maximization (EM)

10

- Log likelihood with a mixture model

$$\mathcal{L}(\Phi \mid \mathcal{X}) = \log \prod_t p(\mathbf{x}^t \mid \Phi)$$

$$= \sum_t \log \sum_{i=1}^{k} p(\mathbf{x}^t \mid G_i) P(G_i)$$

- Assume hidden variables z, which when known, make optimization much simpler
- Complete likelihood, $L_c(\Phi \mid X,Z)$, in terms of ***x*** and ***z***
- Incomplete likelihood, $L(\Phi \mid X)$, in terms of ***x***

E- and M-steps

11

Iterate the two steps

1. E-step: Estimate *z* given X and current Φ
2. M-step: Find new Φ' given *z*, X, and old Φ.

$$\text{E-step: } \mathcal{Q}(\Phi \mid \Phi^l) = E\left[\mathcal{L}_C(\Phi \mid \mathcal{X}, \mathcal{Z}) \mid \mathcal{X}, \Phi^l\right]$$

$$\text{M-step: } \Phi^{l+1} = \arg\max_{\Phi} \mathcal{Q}(\Phi \mid \Phi^l)$$

An increase in Q increases incomplete likelihood

$$\mathcal{L}(\Phi^{l+1} \mid \mathcal{X}) \geq \mathcal{L}(\Phi^l \mid \mathcal{X})$$

EM in Gaussian Mixtures

12

- $z^t_i = 1$ if $\boldsymbol{x}^t$ belongs to G_i, 0 otherwise (labels $\boldsymbol{r}^t_i$ of supervised learning); assume $p(\boldsymbol{x} \mid G_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$
- E-step:

$$E\left[z_i^t \mid \mathcal{X}, \Phi^l\right] = \frac{p(\mathbf{x}^t \mid G_i, \Phi^l) P(G_i)}{\sum_j p(\mathbf{x}^t \mid G_j, \Phi^l) P(G_j)}$$

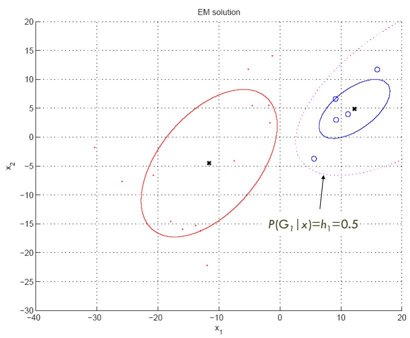
$$= P(G_i \mid \mathbf{x}^t, \Phi^l) \equiv h_i^t$$

- M-step:

$$P(G_i) = \frac{\sum_t h_i^t}{N} \qquad \mathbf{m}_i^{l+1} = \frac{\sum_t h_i^t \mathbf{x}^t}{\sum_t h_i^t}$$

$$\mathbf{S}_i^{l+1} = \frac{\sum_t h_i^t (\mathbf{x}^t - \mathbf{m}_i^{l+1})(\mathbf{x}^t - \mathbf{m}_i^{l+1})^T}{\sum_t h_i^t}$$

Use estimated labels in place of unknown labels



13

Mixtures of Latent Variable Models

14

Regularize clusters

1. Assume shared/diagonal covariance matrices
2. Use PCA/FA to decrease dimensionality: Mixtures of PCA/FA

$$p(\mathbf{x}_t \mid G_i) = \mathcal{N}(\mathbf{m}_i, \mathbf{V}_i \mathbf{V}_i^T + \boldsymbol{\psi}_i)$$

Can use EM to learn $\mathbf{V}_i$ (Ghahramani and Hinton, 1997; Tipping and Bishop, 1999)

After Clustering

15

- Dimensionality reduction methods find correlations between features and group features
- Clustering methods find similarities between instances and group instances
- Allows knowledge extraction through
  - number of clusters,
  - prior probabilities,
  - cluster parameters, i.e., center, range of features.

Example: CRM, customer segmentation

Clustering as Preprocessing

16

- Estimated group labels h_j (soft) or b_j (hard) may be seen as the dimensions of a new *k* dimensional space, where we can then learn our discriminant or regressor.
- Local representation (only one b_j is 1, all others are 0; only few h_j are nonzero) vs

  Distributed representation (After PCA; all z_j are nonzero)

Mixture of Mixtures

17

- In classification, the input comes from a mixture of classes (supervised).
- If each class is also a mixture, e.g., of Gaussians, (unsupervised), we have a mixture of mixtures:

$$p(\mathbf{x} \mid C_i) = \sum_{j=1}^{k_i} p(\mathbf{x} \mid G_{ij}) P(G_{ij})$$

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x} \mid C_i) P(C_i)$$

Spectral Clustering

18

- Cluster using predefined pairwise similarities B_{rs} instead of using Euclidean or Mahalanobis distance
- Can be used even if instances not vectorially represented
- Steps:
  - I. Use Laplacian Eigenmaps (chapter 6) to map to a new **z** space using B_{rs}
  - II. Use *k*-means in this new **z** space for clustering

Hierarchical Clustering

19

- Cluster based on similarities/distances
- Distance measure between instances $\boldsymbol{x}^r$ and $\boldsymbol{x}^s$

Minkowski (L_p) (Euclidean for $p = 2$)

$$d_m(\mathbf{x}^r, \mathbf{x}^s) = \left[\sum_{j=1}^{d} (x_j^r - x_j^s)^p\right]^{1/p}$$

City-block distance

$$d_{cb}(\mathbf{x}^r, \mathbf{x}^s) = \sum_{j=1}^{d} \left| x_j^r - x_j^s \right|$$

Agglomerative Clustering

20

- Start with *N* groups each with one instance and merge two closest groups at each iteration
- Distance between two groups G_i and G_j:
  - Single-link:

$$d(G_i, G_j) = \min_{\mathbf{x}^r \in G_i, \mathbf{x}^s \in G_j} d(\mathbf{x}^r, \mathbf{x}^s)$$

  - Complete-link:

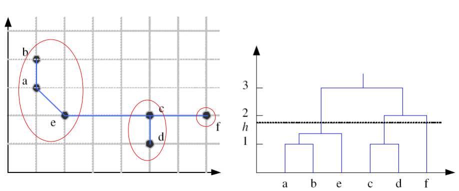
$$d(G_i, G_j) = \max_{\mathbf{x}^r \in G_i, \mathbf{x}^s \in G_j} d(\mathbf{x}^r, \mathbf{x}^s)$$

  - Average-link, centroid

$$d(G_i, G_j) = \operatorname*{ave}_{\mathbf{x}^r \in G_i, \mathbf{x}^s \in G_j} d(\mathbf{x}^r, \mathbf{x}^s)$$

Example: Single-Link Clustering

21



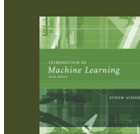
Dendrogram

Choosing *k*

22

- Defined by the application, e.g., image quantization
- Plot data (after PCA) and check for clusters
- Incremental (leader-cluster) algorithm: Add one at a time until "elbow" (reconstruction error/log likelihood/intergroup distances)
- Manually check for meaning

i2ml3e-chap08.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 8:

NONPARAMETRIC METHODS

Nonparametric Estimation

- Parametric (single global model), semiparametric (small number of local models)
- Nonparametric: Similar inputs have similar outputs
- Functions (pdf, discriminant, regression) change smoothly
- Keep the training data;"let the data speak for itself"
- Given x, find a small number of closest training instances and interpolate from these
- Aka lazy/memory-based/case-based/instance-based learning

Density Estimation

- Given the training set $X=\{x^t\}_t$ drawn iid from $p(x)$
- Divide data into bins of size h
- Histogram:

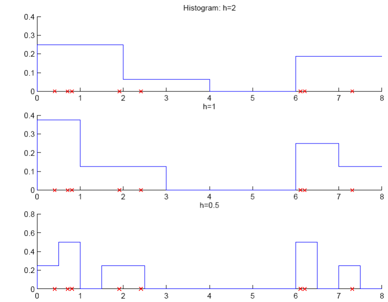
$$\hat{p}(x) = \frac{\#\{x^t \text{ in the same bin as } x\}}{Nh}$$

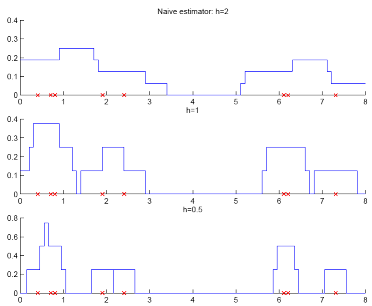
- Naive estimator:

$$\hat{p}(x) = \frac{\#\{x-h < x^t \le x+h\}}{2Nh}$$

or

$$\hat{p}(x) = \frac{1}{Nh}\sum_{t=1}^{N} w\left(\frac{x-x^t}{h}\right) \quad w(u) = \begin{cases} 1/2 & \text{if } |u|<1 \\ 0 & \text{otherwise} \end{cases}$$





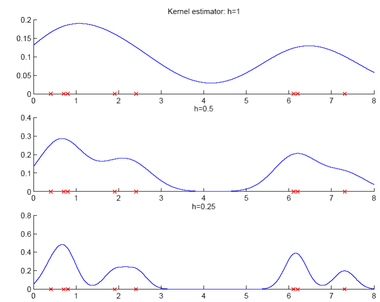
Kernel Estimator

- Kernel function, e.g., Gaussian kernel:

$$K(u) = \frac{1}{\sqrt{2\pi}}\exp\left[-\frac{u^2}{2}\right]$$

- Kernel estimator (Parzen windows)

$$\hat{p}(x) = \frac{1}{Nh}\sum_{t=1}^{N} K\left(\frac{x-x^t}{h}\right)$$

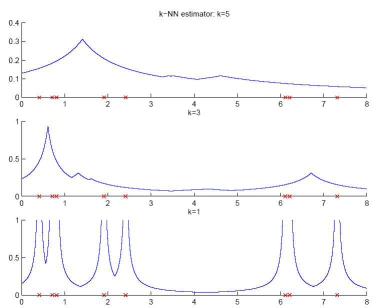


k-Nearest Neighbor Estimator

- Instead of fixing bin width h and counting the number of instances, fix the instances (neighbors) k and check bin width

$$\hat{p}(x) = \frac{k}{2Nd_k(x)}$$

$d_k(x)$, distance to kth closest instance to x



Multivariate Data

- Kernel density estimator

$$\hat{p}(\mathbf{x}) = \frac{1}{Nh^d}\sum_{t=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}^t}{h}\right)$$

Multivariate Gaussian kernel

spheric $$K(\mathbf{u}) = \left(\frac{1}{\sqrt{2\pi}}\right)^d \exp\left[-\frac{\|\mathbf{u}\|^2}{2}\right]$$

ellipsoid $$K(\mathbf{u}) = \frac{1}{(2\pi)^{d/2}|\mathbf{S}|^{1/2}}\exp\left[-\frac{1}{2}\mathbf{u}^T\mathbf{S}^{-1}\mathbf{u}\right]$$

Nonparametric Classification

- Estimate $p(\mathbf{x}|C_i)$ and use Bayes' rule
- Kernel estimator

$$\hat{p}(\mathbf{x}|C_i) = \frac{1}{N_i h^d}\sum_{t=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}^t}{h}\right) r_i^t \quad \hat{P}(C_i) = \frac{N_i}{N}$$

$$g_i(\mathbf{x}) = \hat{p}(\mathbf{x}|C_i)\hat{P}(C_i) = \frac{1}{Nh^d}\sum_{t=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}^t}{h}\right) r_i^t$$

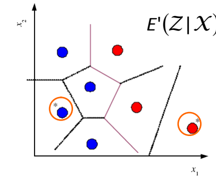
- k-NN estimator

$$\hat{p}(\mathbf{x}|C_i) = \frac{k_i}{N_i V^k(\mathbf{x})} \quad \hat{P}(C_i|\mathbf{x}) = \frac{\hat{p}(\mathbf{x}|C_i)\hat{P}(C_i)}{\hat{p}(\mathbf{x})} = \frac{k_i}{k}$$

Condensed Nearest Neighbor

- Time/space complexity of k-NN is O (N)
- Find a subset Z of X that is small and is accurate in classifying X (Hart, 1968)

$$E'(\mathcal{Z}|\mathcal{X}) = E(\mathcal{X}|\mathcal{Z}) + \lambda|\mathcal{Z}|$$



Condensed Nearest Neighbor

- Incremental algorithm: Add instance if needed

```
Z ← ∅
Repeat
    For all x ∈ X (in random order)
        Find x' ∈ Z s.t. ||x − x'|| = min_{x^j ∈ Z} ||x − x^j||
        If class(x)≠class(x') add x to Z
Until Z does not change
```

Distance-based Classification

- Find a distance function $D(\mathbf{x}^r,\mathbf{x}^s)$ such that

  if $\mathbf{x}^r$ and $\mathbf{x}^s$ belong to the same class, distance is small and if they belong to different classes, distance is large
- Assume a parametric model and learn its parameters using data, e.g.,

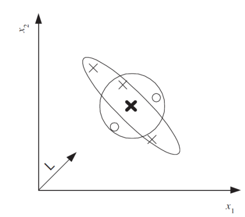
$$\mathcal{D}(\mathbf{x},\mathbf{x}^t|\mathbf{M}) = (\mathbf{x}-\mathbf{x}^t)^T\mathbf{M}(\mathbf{x}-\mathbf{x}^t)$$

Learning a Distance Function

- The three-way relationship between distances, dimensionality reduction, and feature extraction.
- $\mathbf{M}=\mathbf{L}^T\mathbf{L}$ is dxd and $\mathbf{L}$ is kxd

$$\begin{aligned}\mathcal{D}(\mathbf{x},\mathbf{x}^t|\mathbf{M}) &= (\mathbf{x}-\mathbf{x}^t)^T\mathbf{M}(\mathbf{x}-\mathbf{x}^t) = (\mathbf{x}-\mathbf{x}^t)^T\mathbf{L}^T\mathbf{L}(\mathbf{x}-\mathbf{x}^t) \\ &= (\mathbf{L}(\mathbf{x}-\mathbf{x}^t))^T(\mathbf{L}(\mathbf{x}-\mathbf{x}^t)) = (\mathbf{L}\mathbf{x}-\mathbf{L}\mathbf{x}^t)^T(\mathbf{L}\mathbf{x}-\mathbf{L}\mathbf{x}^t) \\ &= (\mathbf{z}-\mathbf{z}^t)^T(\mathbf{z}-\mathbf{z}^t) = \|\mathbf{z}-\mathbf{z}^t\|^2\end{aligned}$$

- Similarity-based representation using similarity scores
- Large-margin nearest neighbor (chapter 13)



Euclidean distance (circle) is not suitable,
Mahalanobis distance using an **M** (ellipse) is suitable.
After the data is projected along **L**, Euclidean distance can be used.

Outlier Detection

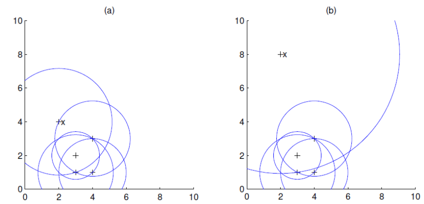
18

- Find outlier/novelty points
- Not a two-class problem because outliers are very few, of many types, and seldom labeled
- Instead, one-class classification problem: Find instances that have low probability
- In nonparametric case: Find instances far away from other instances

Local Outlier Factor

19

$$\mathrm{LOF}(\mathbf{x}) = \frac{d_k(\mathbf{x})}{\sum_{\mathbf{s}\in\mathcal{N}(\mathbf{x})} d_k(\mathbf{s})/|\mathcal{N}(\mathbf{x})|}$$



Nonparametric Regression

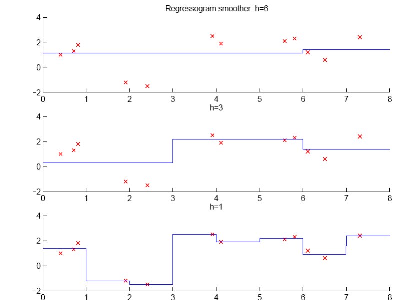
20

- Aka smoothing models
- Regressogram

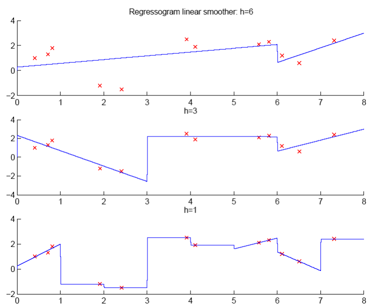
$$\hat{g}(x) = \frac{\sum_{t=1}^{N} b(x,x^t)r^t}{\sum_{t=1}^{N} b(x,x^t)}$$

where

$$b(x,x^t) = \begin{cases} 1 & \text{if } x^t \text{ is in the same bin with } x \\ 0 & \text{otherwise} \end{cases}$$



21



22

Running Mean/Kernel Smoother

23

- Running mean smoother

$$\hat{g}(x) = \frac{\sum_{t=1}^{N} w\left(\frac{x-x^t}{h}\right)r^t}{\sum_{t=1}^{N} w\left(\frac{x-x^t}{h}\right)}$$

where

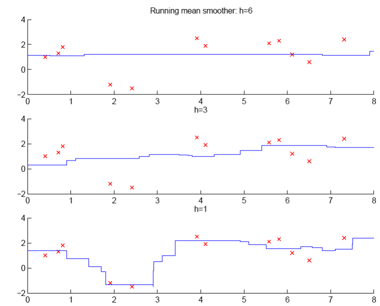
$$w(u) = \begin{cases} 1 & \text{if } |u| < 1 \\ 0 & \text{otherwise} \end{cases}$$

- Running line smoother
- Kernel smoother

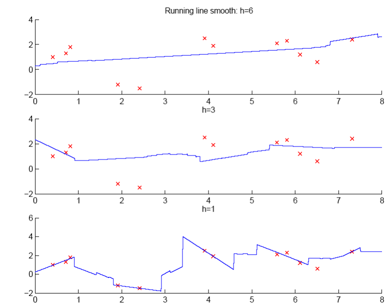
$$\hat{g}(x) = \frac{\sum_{t=1}^{N} K\left(\frac{x-x^t}{h}\right)r^t}{\sum_{t=1}^{N} K\left(\frac{x-x^t}{h}\right)}$$

where $K(\,)$ is Gaussian

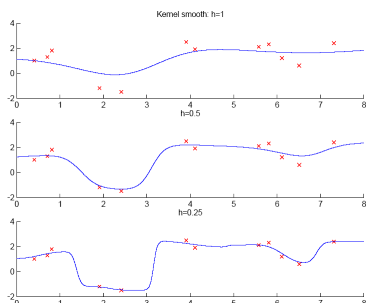
- Additive models (Hastie and Tibshirani, 1990)



24



25

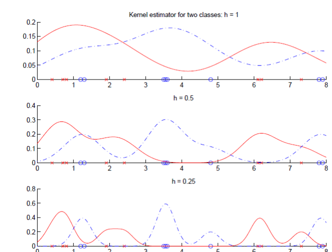


26

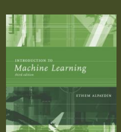
How to Choose *k* or *h* ?

27

- When *k* or *h* is small, single instances matter; bias is small, variance is large (undersmoothing): High complexity
- As *k* or *h* increases, we average over more instances and variance decreases but bias increases (oversmoothing): Low complexity
- Cross-validation is used to finetune *k* or *h*.



28



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

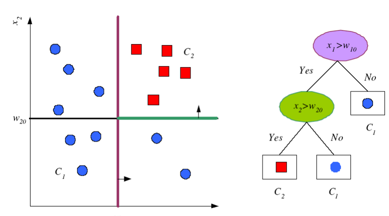
alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 9:

DECISION TREES

Tree Uses Nodes and Leaves

3



Divide and Conquer

4

- Internal decision nodes
  - Univariate: Uses a single attribute, x_i
    - Numeric x_i : Binary split : $x_i > w_m$
    - Discrete x_i : n-way split for n possible values
  - Multivariate: Uses all attributes, $\mathbf{x}$
- Leaves
  - Classification: Class labels, or proportions
  - Regression: Numeric; r average, or local fit
- Learning is greedy; find the best split recursively (Breiman et al, 1984; Quinlan, 1986, 1993)

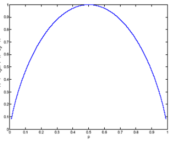
Classification Trees (ID3,CART,C4.5)

5

- For node m, N_m instances reach m, N^i_m belong to C_i

$$\hat{P}(C_i \mid \mathbf{x}, m) \equiv p^i_m = \frac{N^i_m}{N_m}$$

- Node m is pure if p^i_m is 0 or 1
- Measure of impurity is entropy



$$\mathcal{I}_m = -\sum_{i=1}^{K} p^i_m \log_2 p^i_m$$

Best Split

6

- If node m is pure, generate a leaf and stop, otherwise split and continue recursively
- Impurity after split: N_{mj} of N_m take branch j. N^i_{mj} belong to C_i

$$\hat{P}(C_i \mid \mathbf{x}, m, j) \equiv p^i_{mj} = \frac{N^i_{mj}}{N_{mj}} \qquad \mathcal{I}'_m = -\sum_{j=1}^{n} \frac{N_{mj}}{N_m} \sum_{i=1}^{K} p^i_{mj} \log_2 p^i_{mj}$$

- Find the variable and split that min impurity (among all variables -- and split positions for numeric variables)

```
GenerateTree(X)
   If NodeEntropy(X)< θ_I /* eq. 9.3
      Create leaf labelled by majority class in X
      Return
   i ← SplitAttribute(X)
   For each branch of x_i
      Find X_i falling in branch
      GenerateTree(X_i)
SplitAttribute(X)
   MinEnt← MAX
   For all attributes i = 1,...,d
      If x_i is discrete with n values
         Split X into X_1,...,X_n by x_i
         e ← SplitEntropy(X_1,...,X_n) /* eq. 9.8 */
         If e<MinEnt MinEnt ← e; bestf ← i
      Else /* x_i is numeric */
         For all possible splits
            Split X into X_1,X_2 on x_i
            e←SplitEntropy(X_1,X_2)
            If e<MinEnt MinEnt ← e; bestf ← i
   Return bestf
```

7

Regression Trees

8

- Error at node m:

$$b_m(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in \mathcal{X}_m : \mathbf{x} \text{ reaches node } m \\ 0 & \text{otherwise} \end{cases}$$

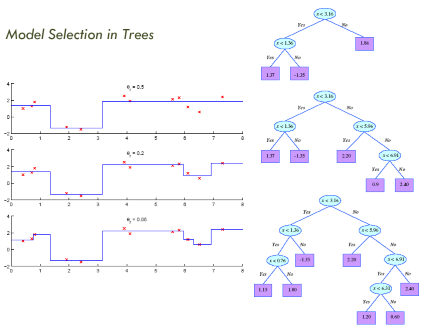
$$E_m = \frac{1}{N_m} \sum_t \left(r^t - g_m\right)^2 b_m(\mathbf{x}^t) \qquad g_m = \frac{\sum_t b_m(\mathbf{x}^t) r^t}{\sum_t b_m(\mathbf{x}^t)}$$

- After splitting:

$$b_{mj}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in \mathcal{X}_{mj} : \mathbf{x} \text{ reaches node } m \text{ and branch } j \\ 0 & \text{otherwise} \end{cases}$$

$$E'_m = \frac{1}{N_m} \sum_j \sum_t \left(r^t - g_{mj}\right)^2 b_{mj}(\mathbf{x}^t) \qquad g_{mj} = \frac{\sum_t b_{mj}(\mathbf{x}^t) r^t}{\sum_t b_{mj}(\mathbf{x}^t)}$$

Model Selection in Trees



9

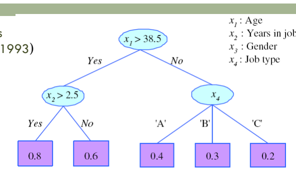
Pruning Trees

10

- Remove subtrees for better generalization (decrease variance)
  - Prepruning: Early stopping
  - Postpruning: Grow the whole tree then prune subtrees that overfit on the pruning set
- Prepruning is faster, postpruning is more accurate (requires a separate pruning set)

Rule Extraction from Trees

C4.5Rules (Quinlan, 1993)



R1: IF (age>38.5) AND (years-in-job>2.5) THEN y =0.8
R2: IF (age>38.5) AND (years-in-job≤2.5) THEN y =0.6
R3: IF (age≤38.5) AND (job-type='A') THEN y =0.4
R4: IF (age≤38.5) AND (job-type='B') THEN y =0.3
R5: IF (age≤38.5) AND (job-type='C') THEN y =0.2

Learning Rules

12

- Rule induction is similar to tree induction but
  - tree induction is breadth-first,
  - rule induction is depth-first; one rule at a time
- Rule set contains rules; rules are conjunctions of terms
- Rule covers an example if all terms of the rule evaluate to true for the example
- Sequential covering: Generate rules one at a time until all positive examples are covered
- IREP (Fürnkrantz and Widmer, 1994), Ripper (Cohen, 1995)

```
Ripper(Pos,Neg,k)
   RuleSet ← LearnRuleSet(Pos,Neg)
   For k times
      RuleSet ← OptimizeRuleSet(RuleSet,Pos,Neg)
LearnRuleSet(Pos,Neg)
   RuleSet ← ∅
   DL ← DescLen(RuleSet,Pos,Neg)
   Repeat
      Rule ← LearnRule(Pos,Neg)
      Add Rule to RuleSet
      DL' ← DescLen(RuleSet,Pos,Neg)
      If DL'>DL+64
         PruneRuleSet(RuleSet,Pos,Neg)
         Return RuleSet
      If DL'<DL DL ← DL'
         Delete instances covered from Pos and Neg
   Until Pos = ∅
   Return RuleSet
```

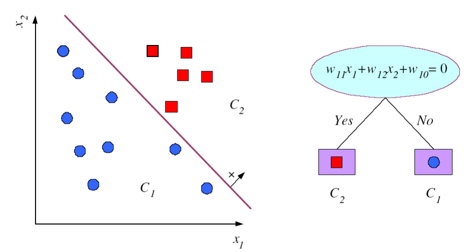
13

```
PruneRuleSet(RuleSet,Pos,Neg)
   For each Rule ∈ RuleSet in reverse order
      DL ← DescLen(RuleSet,Pos,Neg)
      DL' ← DescLen(RuleSet-Rule,Pos,Neg)
      IF DL'<DL Delete Rule from RuleSet
   Return RuleSet
OptimizeRuleSet(RuleSet,Pos,Neg)
   For each Rule ∈ RuleSet
      DL0 ← DescLen(RuleSet,Pos,Neg)
      DL1 ← DescLen(RuleSet-Rule+
         ReplaceRule(RuleSet,Pos,Neg),Pos,Neg)
      DL2 ← DescLen(RuleSet-Rule+
         ReviseRule(RuleSet,Rule,Pos,Neg),Pos,Neg)
      If DL1=min(DL0,DL1,DL2)
         Delete Rule from RuleSet and
            add ReplaceRule(RuleSet,Pos,Neg)
      Else If DL2=min(DL0,DL1,DL2)
         Delete Rule from RuleSet and
            add ReviseRule(RuleSet,Rule,Pos,Neg)
   Return RuleSet
```

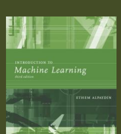
14

Multivariate Trees

15



i2ml3e-chap10.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 10:

LINEAR DISCRIMINATION

Likelihood- vs. Discriminant-based Classification

3

- Likelihood-based: Assume a model for $p(\mathbf{x} \mid C_i)$, use Bayes' rule to calculate $P(C_i \mid \mathbf{x})$

  $g_i(\mathbf{x}) = \log P(C_i \mid \mathbf{x})$
- Discriminant-based: Assume a model for $g_i(\mathbf{x} \mid \Phi_i)$; no density estimation
- Estimating the boundaries is enough; no need to accurately estimate the densities inside the boundaries

Linear Discriminant

4

- Linear discriminant:

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

- Advantages:
  - Simple: O(d) space/computation
  - Knowledge extraction: Weighted sum of attributes; positive/negative weights, magnitudes (credit scoring)
  - Optimal when $p(\mathbf{x} \mid C_i)$ are Gaussian with shared cov matrix; useful when classes are (almost) linearly separable

Generalized Linear Model

- Quadratic discriminant:

$$g_i(\mathbf{x} \mid \mathbf{W}_i, \mathbf{w}_i, w_{i0}) = \mathbf{x}^T \mathbf{W}_i \mathbf{x} + \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

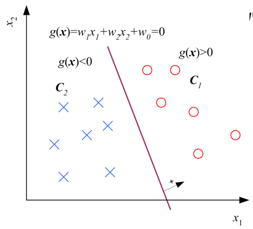
- Higher-order (product) terms:

$$z_1 = x_1,\ z_2 = x_2,\ z_3 = x_1^2,\ z_4 = x_2^2,\ z_5 = x_1 x_2$$

Map from $\mathbf{x}$ to $\mathbf{z}$ using nonlinear basis functions and use a linear discriminant in $\mathbf{z}$-space

$$g_i(\mathbf{x}) = \sum_{j=1}^{k} w_{ij}\phi_j(\mathbf{x})$$

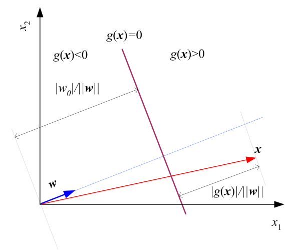
Two Classes



$$\begin{aligned} g(\mathbf{x}) &= g_1(\mathbf{x}) - g_2(\mathbf{x}) \\ &= (\mathbf{w}_1^T \mathbf{x} + w_{10}) - (\mathbf{w}_2^T \mathbf{x} + w_{20}) \\ &= (\mathbf{w}_1 - \mathbf{w}_2)^T \mathbf{x} + (w_{10} - w_{20}) \\ &= \mathbf{w}^T \mathbf{x} + w_0 \end{aligned}$$

$$\text{choose} \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}$$

Geometry

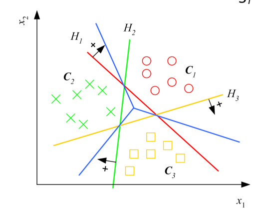


Multiple Classes

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

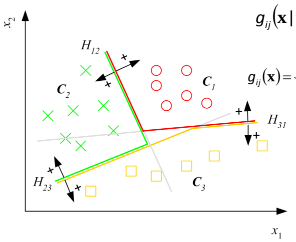
Choose C_i if

$$g_i(\mathbf{x}) = \max_{j=1}^{K} g_j(\mathbf{x})$$



Classes are linearly separable

Pairwise Separation



$$g_{ij}(\mathbf{x} \mid \mathbf{w}_{ij}, w_{ij0}) = \mathbf{w}_{ij}^T \mathbf{x} + w_{ij0}$$

$$g_{ij}(\mathbf{x}) = \begin{cases} > 0 & \text{if } \mathbf{x} \in C_i \\ \le 0 & \text{if } \mathbf{x} \in C_j \\ \text{don't care} & \text{otherwise} \end{cases}$$

choose C_i if

$$\forall j \ne i, g_{ij}(\mathbf{x}) > 0$$

From Discriminants to Posteriors

When $p(\mathbf{x} \mid C_i) \sim N(\boldsymbol{\mu}_i, \Sigma)$

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

$$\mathbf{w}_i = \Sigma^{-1}\boldsymbol{\mu}_i \quad w_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i^T \Sigma^{-1}\boldsymbol{\mu}_i + \log P(C_i)$$

$$y \equiv P(C_1 \mid \mathbf{x}) \text{ and } P(C_2 \mid \mathbf{x}) = 1 - y$$

$$\text{choose } C_1 \text{ if} \begin{cases} y > 0.5 \\ y/(1-y) > 1 \\ \log[y/(1-y)] > 0 \end{cases} \text{ and } C_2 \text{ otherwise}$$

$$\begin{aligned} \operatorname{logit}(P(C_1 \mid \mathbf{x})) &= \log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})} \\ &= \log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} + \log \frac{P(C_1)}{P(C_2)} \\ &= \log \frac{(2\pi)^{-d/2}|\Sigma|^{-1/2}\exp[-(1/2)(\mathbf{x}-\boldsymbol{\mu}_1)^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)]}{(2\pi)^{-d/2}|\Sigma|^{-1/2}\exp[-(1/2)(\mathbf{x}-\boldsymbol{\mu}_2)^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}_2)]} + \log \frac{P(C_1)}{P(C_2)} \\ &= \mathbf{w}^T \mathbf{x} + w_0 \end{aligned}$$

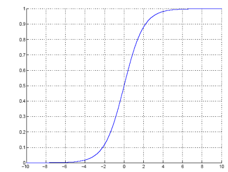
$$\text{where } \mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \quad w_0 = -\frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)^T \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

The inverse of logit

$$\log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \mathbf{w}^T \mathbf{x} + w_0$$

$$P(C_1 \mid \mathbf{x}) = \operatorname{sigmoid}(\mathbf{w}^T \mathbf{x} + w_0) = \frac{1}{1 + \exp[-(\mathbf{w}^T \mathbf{x} + w_0)]}$$

Sigmoid (Logistic) Function



Calculate $g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$ and choose C_1 if $g(\mathbf{x}) > 0$, or

Calculate $y = \operatorname{sigmoid}(\mathbf{w}^T \mathbf{x} + w_0)$ and choose C_1 if $y > 0.5$

Gradient-Descent

- $E(\mathbf{w} \mid X)$ is error with parameters $\mathbf{w}$ on sample X

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} E(\mathbf{w} \mid X)$$

- Gradient

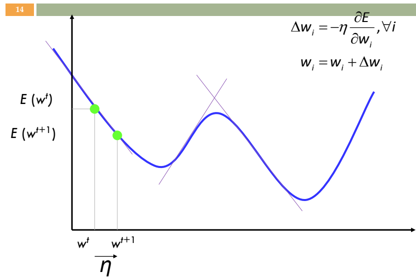
$$\nabla_{\mathbf{w}} E = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_d}\right]^T$$

- Gradient-descent:
  - Starts from random $\mathbf{w}$ and updates $\mathbf{w}$ iteratively in the negative direction of gradient

Gradient-Descent

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \forall i$$

$$w_i = w_i + \Delta w_i$$



Logistic Discrimination

Two classes: Assume log likelihood ratio is linear

$$\log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} = \mathbf{w}^T \mathbf{x} + w_0^o$$

$$\begin{aligned} \operatorname{logit}(P(C_1 \mid \mathbf{x})) &= \log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} + \log \frac{P(C_1)}{P(C_2)} \\ &= \mathbf{w}^T \mathbf{x} + w_0 \end{aligned}$$

$$\text{where } w_0 = w_0^o + \log \frac{P(C_1)}{P(C_2)}$$

$$y = \hat{P}(C_1 \mid \mathbf{x}) = \frac{1}{1 + \exp[-(\mathbf{w}^T \mathbf{x} + w_0)]}$$

Training: Two Classes

$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_t \quad r^t \mid \mathbf{x}^t \sim \text{Bernoulli}(y^t)$$

$$y = P(C_1 \mid \mathbf{x}) = \frac{1}{1 + \exp[-(\mathbf{w}^T \mathbf{x} + w_0)]}$$

$$l(\mathbf{w}, w_0 \mid \mathcal{X}) = \prod_t (y^t)^{(r^t)} (1 - y^t)^{(1 - r^t)}$$

$$E = -\log l$$

$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + (1 - r^t)\log(1 - y^t)$$

Training: Gradient-Descent

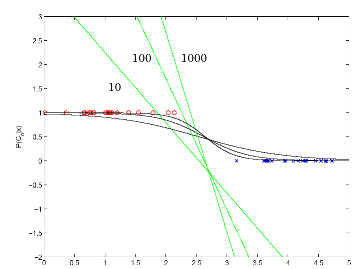
$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + (1 - r^t)\log(1 - y^t)$$

$$\text{If } y = \operatorname{sigmoid}(a) \quad \frac{dy}{da} = y(1 - y)$$

$$\begin{aligned} \Delta w_j &= -\eta \frac{\partial E}{\partial w_j} = \eta \sum_t \left(\frac{r^t}{y^t} - \frac{1 - r^t}{1 - y^t}\right) y^t (1 - y^t) x_j^t \\ &= \eta \sum_t (r^t - y^t) x_j^t, j = 1, \ldots, d \end{aligned}$$

$$\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = \eta \sum_t (r^t - y^t)$$

```
For j = 0,...,d
    w_j ← rand(-0.01,0.01)
Repeat
    For j = 0,...,d
        Δw_j ← 0
    For t = 1,...,N
        o ← 0
        For j = 0,...,d
            o ← o + w_j x_j^t
        y ← sigmoid(o)
        Δw_j ← Δw_j + (r^t − y)x_j^t
    For j = 0,...,d
        w_j ← w_j + ηΔw_j
Until convergence
```

K>2 Classes

$$\mathcal{X} = \{\mathbf{x}^t, \mathbf{r}^t\}_t \quad r^t \mid \mathbf{x}^t \sim \text{Mult}_K(1, \mathbf{y}^t)$$

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{w}_i^T \mathbf{x} + w_{i0}^o$$

$$y = \hat{P}(C_i \mid \mathbf{x}) = \frac{\exp[\mathbf{w}_i^T \mathbf{x} + w_{i0}]}{\sum_{j=1}^{K} \exp[\mathbf{w}_j^T \mathbf{x} + w_{j0}]}, i = 1, \ldots, K \qquad \textit{softmax}$$

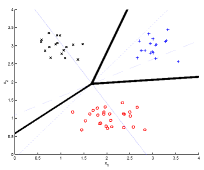
$$l(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = \prod_t \prod_i (y_i^t)^{(r_i^t)}$$

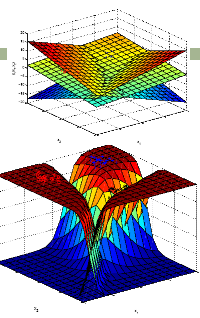
$$E(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = -\sum_t r_i^t \log y_i^t$$

$$\Delta \mathbf{w}_j = \eta \sum_t (r_j^t - y_j^t)\mathbf{x}^t \quad \Delta w_{j0} = \eta \sum_t (r_j^t - y_j^t)$$

```
For i = 1,...,K, For j = 0,...,d, w_ij ← rand(−0.01,0.01)
Repeat
    For i = 1,...,K, For j = 0,...,d, Δw_ij ← 0
    For t = 1,...,N
        For i = 1,...,K
            o_i ← 0
            For j = 0,...,d
                o_i ← o_i + w_ij x_j^t
        For i = 1,...,K
            y_i ← exp(o_i)/ Σ_k exp(o_k)
        For i = 1,...,K
            For j = 0,...,d
                Δw_ij ← Δw_ij + (r_i^t − y_i)x_j^t
    For i = 1,...,K
        For j = 0,...,d
            w_ij ← w_ij + ηΔw_ij
Until convergence
```


Example





Generalizing the Linear Model

- Quadratic:

$$\log\frac{p(\mathbf{x}|C_i)}{p(\mathbf{x}|C_K)}=\mathbf{x}^T\mathbf{W}_i\mathbf{x}+\mathbf{w}_i^T\mathbf{x}+w_{i0}$$

- Sum of basis functions:

$$\log\frac{p(\mathbf{x}|C_i)}{p(\mathbf{x}|C_K)}=\mathbf{w}_i^T\phi(\mathbf{x})+w_{i0}$$

where $\phi(x)$ are basis functions. Examples:
  - Hidden units in neural networks (Chapters 11 and 12)
  - Kernels in SVM (Chapter 13)

Discrimination by Regression

- Classes are NOT mutually exclusive and exhaustive

$$r^t=y^t+\varepsilon \text{ where } \varepsilon\sim\mathcal{N}(0,\sigma^2)$$

$$y^t=\text{sigmoid}(\mathbf{w}^T\mathbf{x}^t+w_0)=\frac{1}{1+\exp[-(\mathbf{w}^T\mathbf{x}^t+w_0)]}$$

$$l(\mathbf{w},w_0|\mathcal{X})=\prod_t\frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(r^t-y^t)^2}{2\sigma^2}\right]$$

$$E(\mathbf{w},w_0|\mathcal{X})=\frac{1}{2}\sum_t(r^t-y^t)^2$$

$$\Delta\mathbf{w}=\eta\sum_t(r^t-y^t)y^t(1-y^t)\mathbf{x}^t$$

Learning to Rank

- Ranking: A different problem than classification or regression
- Let us say $\mathbf{x}^u$ and $\mathbf{x}^v$ are two instances, e.g., two movies

  We prefer u to v implies that $g(\mathbf{x}^u)>g(\mathbf{x}^v)$

  where $g(\mathbf{x})$ is a score function, here linear:

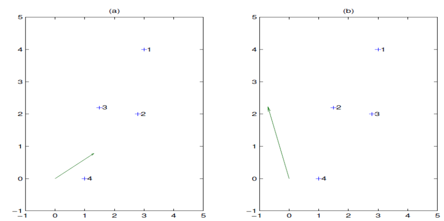
  $g(\mathbf{x})=\mathbf{w}^T\mathbf{x}$
- Find a direction $\mathbf{w}$ such that we get the desired ranks when instances are projected along $\mathbf{w}$

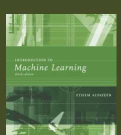
Ranking Error

- We prefer u to v implies that $g(\mathbf{x}^u)>g(\mathbf{x}^v)$, so error is $g(\mathbf{x}^v)$-$g(\mathbf{x}^u)$, if $g(\mathbf{x}^u)<g(\mathbf{x}^v)$

$$E(\mathbf{w}|\{r^u,r^v\})=\sum_{r^u<r^v}[g(\mathbf{x}^v|\theta)-g(\mathbf{x}^u|\theta)]_+$$

where a_+ is equal to a if $a\geq 0$ and 0 otherwise.





Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

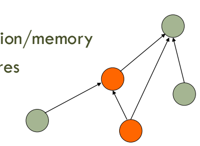
alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 11:

MULTILAYER PERCEPTRONS

Neural Networks

- Networks of processing units (neurons) with connections (synapses) between them
- Large number of neurons: 10^{10}
- Large connectitivity: 10^5
- Parallel processing
- Distributed computation/memory
- Robust to noise, failures



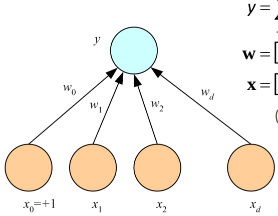
Understanding the Brain

- Levels of analysis (Marr, 1982)
  1. Computational theory
  2. Representation and algorithm
  3. Hardware implementation
- Reverse engineering: From hardware to theory
- Parallel processing: SIMD vs MIMD

  Neural net: SIMD with modifiable local memory

  Learning: Update by training/experience

Perceptron



$$y=\sum_{j=1}^d w_jx_j+w_0=\mathbf{w}^T\mathbf{x}$$

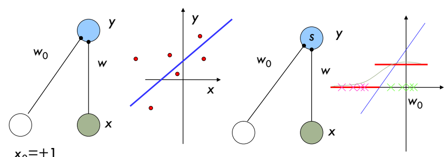
$$\mathbf{w}=[w_0,w_1,\ldots,w_d]^T$$

$$\mathbf{x}=[1,x_1,\ldots,x_d]^T$$

(Rosenblatt, 1962)

What a Perceptron Does

- Regression: $y=wx+w_0$
- Classification: $y=1(wx+w_0>0)$



$$y=\text{sigmoid}(o)=\frac{1}{1+\exp[-\mathbf{w}^T\mathbf{x}]}$$

K Outputs

Regression:

$$y_i=\sum_{j=1}^d w_{ij}x_j+w_{i0}=\mathbf{w}_i^T\mathbf{x}$$

$$\mathbf{y}=\mathbf{W}\mathbf{x}$$

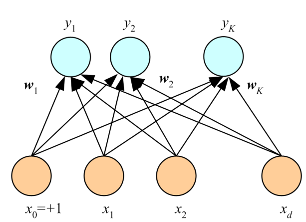
Classification:

$$o_i=\mathbf{w}_i^T\mathbf{x}$$

$$y_i=\frac{\exp o_i}{\sum_k \exp o_k}$$

choose C_i

if $y_i=\max_k y_k$



Training

- Online (instances seen one by one) vs batch (whole sample) learning:
  - No need to store the whole sample
  - Problem may change in time
  - Wear and degradation in system components
- Stochastic gradient-descent: Update after a single pattern
- Generic update rule (LMS rule):

$$\Delta w_{ij}^t=\eta(r_i^t-y_i^t)x_j^t$$

Update=LearningFactor·(DesiredOutput−ActualOutput)·Input

Training a Perceptron: Regression

- Regression (Linear output):

$$E^t(\mathbf{w}\mid\mathbf{x}^t,r^t)=\frac{1}{2}\left(r^t-y^t\right)^2=\frac{1}{2}\left[r^t-\left(\mathbf{w}^T\mathbf{x}^t\right)\right]^2$$

$$\Delta w_j^t=\eta\left(r^t-y^t\right)x_j^t$$

Classification

- Single sigmoid output

$$y^t=\text{sigmoid}\left(\mathbf{w}^T\mathbf{x}^t\right)$$

$$E^t(\mathbf{w}\mid\mathbf{x}^t,\mathbf{r}^t)=-r^t\log y^t-\left(1-r^t\right)\log\left(1-y^t\right)$$

$$\Delta w_j^t=\eta\left(r^t-y^t\right)x_j^t$$

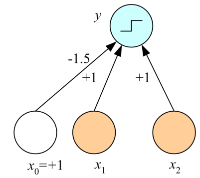
- K>2 softmax outputs

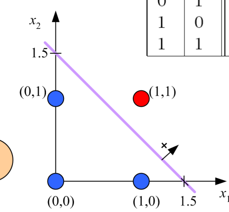
$$y^t=\frac{\exp \mathbf{w}_i^T\mathbf{x}^t}{\sum_k \exp \mathbf{w}_k^T\mathbf{x}^t}\qquad E^t\left(\{\mathbf{w}_i\}_i\mid\mathbf{x}^t,\mathbf{r}^t\right)=-\sum_i r_i^t\log y_i^t$$

$$\Delta w_{ij}^t=\eta\left(r_i^t-y_i^t\right)x_j^t$$

Learning Boolean AND

x_1	x_2	r
0	0	0
0	1	0
1	0	0
1	1	1



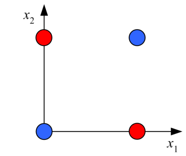


XOR

x_1	x_2	r
0	0	0
0	1	1
1	0	1
1	1	0

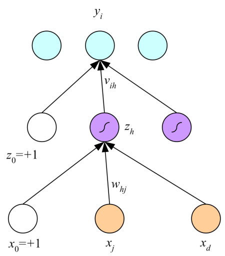
- No w_0, w_1, w_2 satisfy:

$$\begin{aligned} & & & w_0 &\le 0\\ & & w_2+ & w_0 &> 0\\ w_1+ & & & w_0 &> 0\\ w_1+ & & w_2+ & w_0 &\le 0\end{aligned}$$



(Minsky and Papert, 1969)

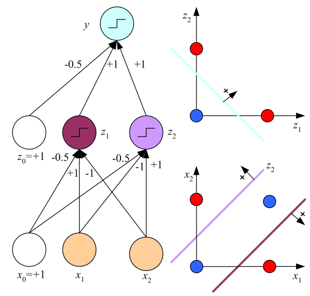
Multilayer Perceptrons



$$y_i=\mathbf{v}_i^T\mathbf{z}=\sum_{h=1}^{H}v_{ih}z_h+v_{i0}$$

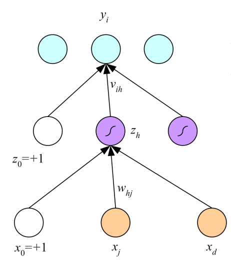
$$z_h=\text{sigmoid}\left(\mathbf{w}_h^T\mathbf{x}\right)=\frac{1}{1+\exp\left[-\left(\sum_{j=1}^{d}w_{hj}x_j+w_{h0}\right)\right]}$$

(Rumelhart et al., 1986)



x_1 XOR x_2 = (x_1 AND ~x_2) OR (~x_1 AND x_2)

Backpropagation



$$y_i=\mathbf{v}_i^T\mathbf{z}=\sum_{h=1}^{H}v_{ih}z_h+v_{i0}$$

$$z_h=\text{sigmoid}\left(\mathbf{w}_h^T\mathbf{x}\right)=\frac{1}{1+\exp\left[-\left(\sum_{j=1}^{d}w_{hj}x_j+w_{h0}\right)\right]}$$

$$\boxed{\frac{\partial E}{\partial w_{hj}}=\frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial z_h}\frac{\partial z_h}{\partial w_{hj}}}$$

Regression

$$E(\mathbf{W},\mathbf{v}\mid\mathcal{X})=\frac{1}{2}\sum_t\left(r^t-y^t\right)^2$$

$$y^t=\sum_{h=1}^{H}v_hz_h^t+v_0$$

Forward

$$z_h=\text{sigmoid}\left(\mathbf{w}_h^T\mathbf{x}\right)$$

$\mathbf{x}$

$$\boxed{\Delta v_h=\sum_t\left(r^t-y^t\right)z_h^t}$$

Backward

$$\begin{aligned}\Delta w_{hj}&=-\eta\frac{\partial E}{\partial w_{hj}}\\&=-\eta\sum_t\frac{\partial E}{\partial y^t}\frac{\partial y^t}{\partial z_h^t}\frac{\partial z_h^t}{\partial w_{hj}}\\&=-\eta\sum_t-\left(r^t-y^t\right)v_hz_h^t\left(1-z_h^t\right)x_j^t\\&=\eta\sum_t\left(r^t-y^t\right)v_hz_h^t\left(1-z_h^t\right)x_j^t\end{aligned}$$

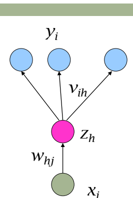
Regression with Multiple Outputs

$$E(\mathbf{W},\mathbf{V}\mid\mathcal{X})=\frac{1}{2}\sum_t\sum_i\left(r_i^t-y_i^t\right)^2$$

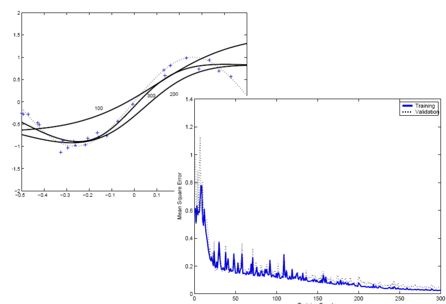
$$y_i^t=\sum_{h=1}^{H}v_{ih}z_h^t+v_{i0}$$

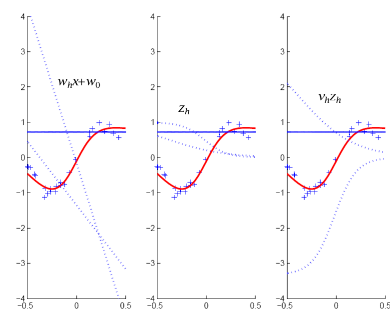
$$\Delta v_{ih}=\eta\sum_t\left(r_i^t-y_i^t\right)z_h^t$$

$$\Delta w_{hj}=\eta\sum_t\left[\sum_i\left(r_i^t-y_i^t\right)v_{ih}\right]z_h^t\left(1-z_h^t\right)x_j^t$$




```
Initialize all v_ih and w_hj to rand(-0.01, 0.01)
Repeat
    For all (x^t, r^t) ∈ X in random order
        For h = 1,...,H
            z_h ← sigmoid(w_h^T x^t)
        For i = 1,...,K
            y_i = v_i^T z
        For i = 1,...,K
            Δv_i = η(r_i^t − y_i^t) z
        For h = 1,...,H
            Δw_h = η(Σ_i (r_i^t − y_i^t) v_ih) z_h (1 − z_h) x^t
        For i = 1,...,K
            v_i ← v_i + Δv_i
        For h = 1,...,H
            w_h ← w_h + Δw_h
Until convergence
```





Two-Class Discrimination

- One sigmoid output y^t for $P(C_1\mid\mathbf{x}^t)$ and $P(C_2\mid\mathbf{x}^t)\equiv 1-y^t$

$$y^t=\text{sigmoid}\left(\sum_{h=1}^{H}v_hz_h^t+v_0\right)$$

$$E(\mathbf{W},\mathbf{v}\mid\mathcal{X})=-\sum_t r^t\log y^t+\left(1-r^t\right)\log\left(1-y^t\right)$$

$$\Delta v_h=\eta\sum_t\left(r^t-y^t\right)z_h^t$$

$$\Delta w_{hj}=\eta\sum_t\left(r^t-y^t\right)v_hz_h^t\left(1-z_h^t\right)x_j^t$$

K>2 Classes

$$o_i^t=\sum_{h=1}^{H}v_{ih}z_h^t+v_{i0}\qquad y_i^t=\frac{\exp o_i^t}{\sum_k\exp o_k^t}\equiv P\left(C_i\mid\mathbf{x}^t\right)$$

$$E(\mathbf{W},\mathbf{v}\mid\mathcal{X})=-\sum_t\sum_i r_i^t\log y_i^t$$

$$\Delta v_{ih}=\eta\sum_t\left(r_i^t-y_i^t\right)z_h^t$$

$$\Delta w_{hj}=\eta\sum_t\left[\sum_i\left(r_i^t-y_i^t\right)v_{ih}\right]z_h^t\left(1-z_h^t\right)x_j^t$$

Multiple Hidden Layers

- MLP with one hidden layer is a universal approximator (Hornik et al., 1989), but using multiple layers may lead to simpler networks

$$z_{1h}=\text{sigmoid}\left(\mathbf{w}_{1h}^T\mathbf{x}\right)=\text{sigmoid}\left(\sum_{j=1}^{d}w_{1hj}x_j+w_{1h0}\right),h=1,\ldots,H_1$$

$$z_{2l}=\text{sigmoid}\left(\mathbf{w}_{2l}^T\mathbf{z}_1\right)=\text{sigmoid}\left(\sum_{h=1}^{H_1}w_{2lh}z_{1h}+w_{2l0}\right),l=1,\ldots,H_2$$

$$y=\mathbf{v}^T\mathbf{z}_2=\sum_{l=1}^{H_2}v_lz_{2l}+v_0$$

Improving Convergence

- Momentum

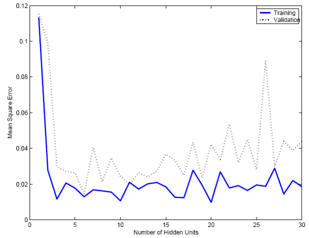
$$\Delta w_i^t=-\eta\frac{\partial E^t}{\partial w_i}+\alpha\Delta w_i^{t-1}$$

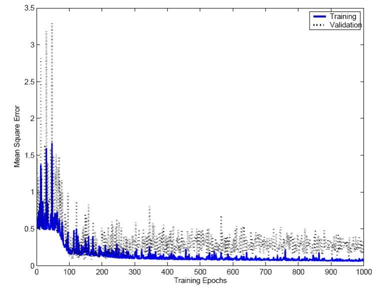
- Adaptive learning rate

$$\Delta\eta=\begin{cases}+a & \text{if } E^{t+\tau}<E^t\\ -b\eta & \text{otherwise}\end{cases}$$

Overfitting/Overtraining

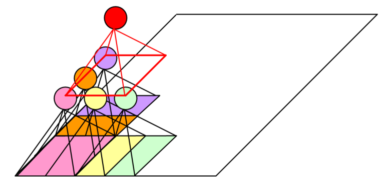
Number of weights: $H(d+1)+(H+1)K$





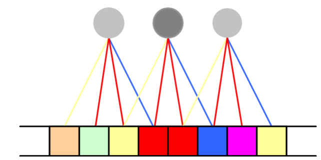
Structured MLP

- Convolutional networks (Deep learning)



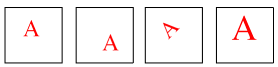
(Le Cun et al, 1989)

Weight Sharing



Hints

- Invariance to translation, rotation, size



- Virtual examples (Abu-Mostafa, 1995)
- Augmented error: $E'=E+\lambda_h E_h$

If $\mathbf{x}'$ and $\mathbf{x}$ are the "same": $E_h=[g(x|\theta)-g(x'|\theta)]^2$

Approximation hint:

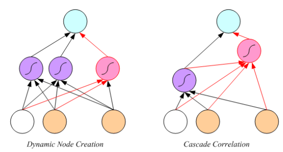
$$E_h=\begin{cases}0 & \text{if } g(x|\theta)\in[a_x,b_x]\\ (g(x|\theta)-a_x)^2 & \text{if } g(x|\theta)<a_x\\ (g(x|\theta)-b_x)^2 & \text{if } g(x|\theta)>b_x\end{cases}$$

Tuning the Network Size

- Destructive
  Weight decay:
- Constructive
  Growing networks

$$\Delta w_i=-\eta\frac{\partial E}{\partial w_i}-\lambda w_i$$

$$E'=E+\frac{\lambda}{2}\sum_i w_i^2$$



(Ash, 1989) (Fahlman and Lebiere, 1989)

Bayesian Learning

- Consider weights w_i as random vars, prior $p(w_i)$

$$p(\mathbf{w}|\mathcal{X})=\frac{p(\mathcal{X}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{X})}\qquad \hat{\mathbf{w}}_{MAP}=\arg\max_{\mathbf{w}}\log p(\mathbf{w}|\mathcal{X})$$

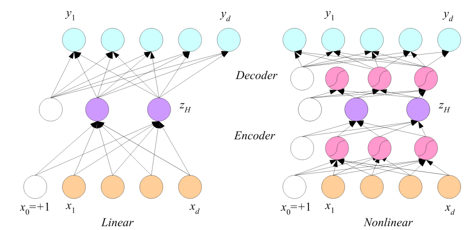
$$\log p(\mathbf{w}|\mathcal{X})=\log p(\mathcal{X}|\mathbf{w})+\log p(\mathbf{w})+C$$

$$p(\mathbf{w})=\prod_i p(w_i)\text{ where } p(w_i)=c\cdot\exp\left[-\frac{w_i^2}{2(1/2\lambda)}\right]$$

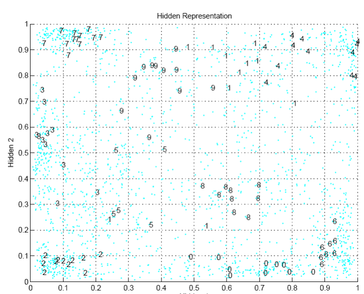
$$E'=E+\lambda\|\mathbf{w}\|^2$$

- Weight decay, ridge regression, regularization
  cost=data-misfit + λ complexity
  More about Bayesian methods in chapter 14

Dimensionality Reduction



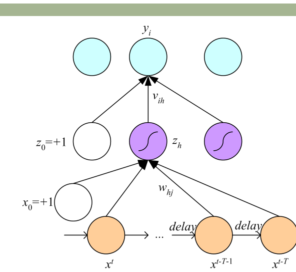
Autoencoder networks



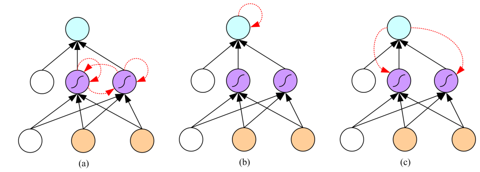
Learning Time

- Applications:
  - Sequence recognition: Speech recognition
  - Sequence reproduction: Time-series prediction
  - Sequence association
- Network architectures
  - Time-delay networks (Waibel et al., 1989)
  - Recurrent networks (Rumelhart et al., 1986)

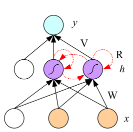
Time-Delay Neural Networks

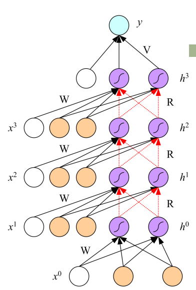


Recurrent Networks



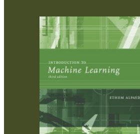
Unfolding in Time





Deep Networks

- Layers of feature extraction units
- Can have local receptive fields as in convolution networks, or can be fully connected
- Can be trained layer by layer using an autoencoder in an unsupervised manner
- No need to craft the right features or the right basis functions or the right dimensionality reduction method; learns multiple layers of abstraction all by itself given a lot of data and a lot of computation
- Applications in vision, language processing, ...



Lecture Slides for
INTRODUCTION TO MACHINE LEARNING
3RD EDITION

ETHEM ALPAYDIN

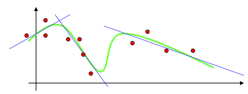
alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 12:

LOCAL MODELS

Introduction

- Divide the input space into local regions and learn simple (constant/linear) models in each patch



- Unsupervised: Competitive, online clustering
- Supervised: Radial-basis functions, mixture of experts

Competitive Learning

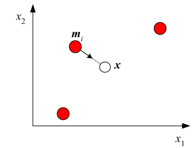
$$E\left(\{\mathbf{m}_i\}_{i=1}^k \mid \mathcal{X}\right) = \sum_t \sum_i b_i^t \left\|\mathbf{x}^t - \mathbf{m}_i\right\|$$

$$b_i^t = \begin{cases} 1 & \text{if } \left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_l \left\|\mathbf{x}^t - \mathbf{m}_l\right\| \\ 0 & \text{otherwise} \end{cases}$$

Batch k-means: $\mathbf{m}_i = \dfrac{\sum_t b_i^t \mathbf{x}^t}{\sum_t b_i^t}$

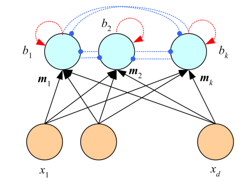
Online k-means:

$$\Delta m_{ij} = -\eta \frac{\partial E^t}{\partial m_{ij}} = \eta b_i^t \left(x_j^t - m_{ij}\right)$$



Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$
Repeat
    For all $\boldsymbol{x}^t \in \mathcal{X}$ in random order
        $i \leftarrow \arg\min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\|$
        $\boldsymbol{m}_i \leftarrow \boldsymbol{m}_i + \eta(\boldsymbol{x}^t - \boldsymbol{m}_i)$
Until $\boldsymbol{m}_i$ converge

Winner-take-all network



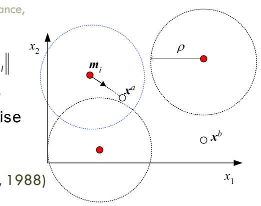
Adaptive Resonance Theory

- Incremental; add a new cluster if not covered; defined by vigilance, ρ

$$b_i^t = \left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_{l=1}^{k} \left\|\mathbf{x}^t - \mathbf{m}_l\right\|$$

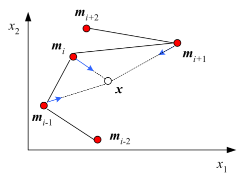
$$\begin{cases} \mathbf{m}_{k+1} \leftarrow \mathbf{x}^t & \text{if } b_i > \rho \\ \Delta \mathbf{m}_i = \eta\left(\mathbf{x}^t - \mathbf{m}_i\right) & \text{otherwise} \end{cases}$$

(Carpenter and Grossberg, 1988)



Self-Organizing Maps

- Units have a neighborhood defined; $\boldsymbol{m}_i$ is "between" $\boldsymbol{m}_{i-1}$ and $\boldsymbol{m}_{i+1}$, and are all updated together
- One-dim map:



(Kohonen, 1990)

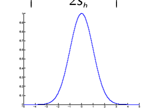
$$\Delta \mathbf{m}_l = \eta e(l,i)\left(\mathbf{x}^t - \mathbf{m}_l\right)$$

$$e(l,i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(l-i)^2}{2\sigma^2}\right]$$

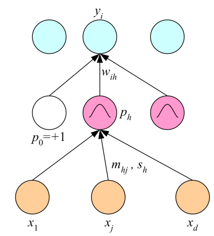
Radial-Basis Functions

- Locally-tuned units:

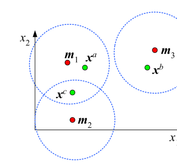
$$p_h^t = \exp\left[-\frac{\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2}{2s_h^2}\right]$$



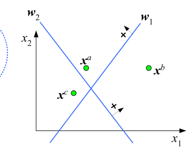
$$y^t = \sum_{h=1}^{H} w_h p_h^t + w_0$$



Local vs Distributed Representation



Local representation in the space of (p_1, p_2, p_3)
x^a: (1.0, 0.0, 0.0)
x^b: (0.0, 0.0, 1.0)
x^c: (1.0, 1.0, 0.0)



Distributed representation in the space of (h_1, h_2)
x^a: (1.0, 1.0)
x^b: (0.0, 1.0)
x^c: (1.0, 0.0)

Training RBF

- Hybrid learning:
  - First layer centers and spreads:
    Unsupervised k-means
  - Second layer weights:
    Supervised gradient-descent
- Fully supervised

(Broomhead and Lowe, 1988; Moody and Darken, 1989)

Regression

$$E\left(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}\right) = \frac{1}{2}\sum_t \sum_i \left(r_i^t - y_i^t\right)^2$$

$$y_i^t = \sum_{h=1}^{H} w_{ih} p_h^t + w_{i0}$$

$$\Delta w_{ih} = \eta \sum_t \left(r_i^t - y_i^t\right) p_h^t$$

$$\Delta m_{hj} = \eta \sum_t \left[\sum_i \left(r_i^t - y_i^t\right) w_{ih}\right] p_h^t \frac{\left(x_j^t - m_{hj}\right)}{s_h^2}$$

$$\Delta s_h = \eta \sum_t \left[\sum_i \left(r_i^t - y_i^t\right) w_{ih}\right] p_h^t \frac{\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2}{s_h^3}$$

Classification

$$E\left(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}\right) = -\sum_t \sum_i r_i^t \log y_i^t$$

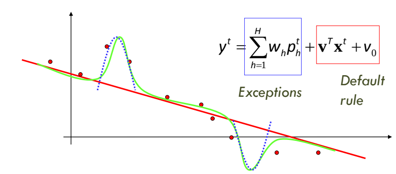
$$y_i^t = \frac{\exp\left[\sum_h w_{ih} p_h^t + w_{i0}\right]}{\sum_k \exp\left[\sum_h w_{kh} p_h^t + w_{k0}\right]}$$

Rules and Exceptions

$$y^t = \boxed{\sum_{h=1}^{H} w_h p_h^t} + \boxed{\mathbf{v}^T \mathbf{x}^t + v_0}$$

Exceptions

Default rule



Rule-Based Knowledge

IF $((x_1 \approx a)$ AND $(x_2 \approx b))$ OR $(x_3 \approx c)$ THEN $y = 0.1$

$$p_1 = \exp\left[-\frac{(x_1 - a)^2}{2s_1^2}\right] \cdot \exp\left[-\frac{(x_2 - b)^2}{2s_2^2}\right] \text{ with } w_1 = 0.1$$

$$p_2 = \exp\left[-\frac{(x_3 - c)^2}{2s_3^2}\right] \text{ with } w_2 = 0.1$$

- Incorporation of prior knowledge (before training)
- Rule extraction (after training) (Tresp et al., 1997)
- Fuzzy membership functions and fuzzy rules

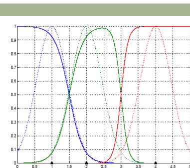
Normalized Basis Functions

$$g_h^t = \frac{p_h^t}{\sum_{l=1}^{H} p_l^t} = \frac{\exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2 / 2s_h^2\right]}{\sum_l \exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_l\right\|^2 / 2s_l^2\right]}$$

$$y_i^t = \sum_{h=1}^{H} w_{ih} g_h^t$$

$$\Delta w_{ih} = \eta \sum_t \left(r_i^t - y_i^t\right) g_h^t$$

$$\Delta m_{hj} = \eta \sum_t \sum_i \left(r_i^t - y_i^t\right)\left(w_{ih} - y_i^t\right) g_h^t \frac{\left(x_j^t - m_{hj}\right)}{s_h^2}$$



Competitive Basis Functions

- Mixture model: $p(\mathbf{r}^t \mid \mathbf{x}^t) = \sum_{h=1}^{H} p(h \mid \mathbf{x}^t) p(\mathbf{r}^t \mid h, \mathbf{x}^t)$

$$p(h \mid \mathbf{x}^t) = \frac{p(\mathbf{x}^t \mid h) p(h)}{\sum_l p(\mathbf{x}^t \mid l) p(l)}$$

$$g_h^t = \frac{a_h \exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2 / 2s_h^2\right]}{\sum_l a_l \exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_l\right\|^2 / 2s_l^2\right]}$$

Regression

$$p(\mathbf{r}^t \mid \mathbf{x}^t) = \prod_i \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{\left(r_i^t - y_i^t\right)^2}{2\sigma^2}\right]$$

$$\mathcal{L}\left(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}\right) = \sum_t \log \sum_h g_h^t \exp\left[-\frac{1}{2}\sum_i \left(r_i^t - y_{ih}^t\right)^2\right]$$

$y_{ih}^t = w_{ih}$ is the constant fit

$$\Delta w_{ih} = \eta \sum_t \left(r_i^t - y_{ih}^t\right) f_h^t \qquad \Delta m_{hj} = \eta \sum_t \left(f_h^t - g_h^t\right) \frac{\left(x_j^t - m_{hj}\right)}{s_h^2}$$

$$f_h^t = \frac{g_h^t \exp\left[-(1/2)\sum_i \left(r_i^t - y_{ih}^t\right)^2\right]}{\sum_l g_l^t \exp\left[-(1/2)\sum_i \left(r_i^t - y_{il}^t\right)^2\right]}$$

$$p(h \mid \mathbf{r}, \mathbf{x}) = \frac{p(h \mid \mathbf{x}) p(\mathbf{r} \mid h, \mathbf{x})}{\sum_l p(l \mid \mathbf{x}) p(\mathbf{r} \mid l, \mathbf{x})}$$

Classification

$$\mathcal{L}(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}) = \sum_t \log \sum_h g_h^t \prod_i (y_{ih}^t)^{r_i^t}$$

$$= \sum_t \log \sum_h g_h^t \exp\left[\sum_i r_i^t \log y_{ih}^t\right]$$

$$y_{ih}^t = \frac{\exp w_{ih}}{\sum_k \exp w_{kh}}$$

$$f_h^t = \frac{g_h^t \exp\left[\sum_i r_i^t \log y_{ih}^t\right]}{\sum_l g_l^t \exp\left[\sum_i r_i^t \log y_{il}^t\right]}$$

EM for RBF (Supervised EM)

- E-step: $f_h^t \equiv p(\mathbf{r} \mid h, \mathbf{x}^t)$
- M-step:

$$\mathbf{m}_h = \frac{\sum_t f_h^t \mathbf{x}^t}{\sum_t f_h^t}$$

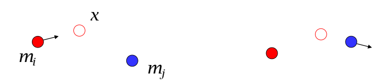
$$s_h = \frac{\sum_t f_h^t (\mathbf{x}^t - \mathbf{m}_h)(\mathbf{x}^t - \mathbf{m}_h)^T}{\sum_t f_h^t}$$

$$w_{ih} = \frac{\sum_t f_h^t r_i^t}{\sum_t f_h^t}$$

Learning Vector Quantization

- H units per class prelabeled (Kohonen, 1990)
- Given $\mathbf{x}$, $\mathbf{m}_i$ is the closest:

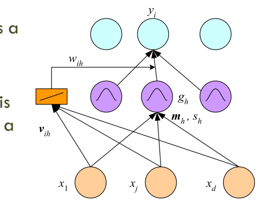
$$\begin{cases} \Delta \mathbf{m}_i = \eta(\mathbf{x}^t - \mathbf{m}_i) & \text{if label}(\mathbf{x}^t) = \text{label}(\mathbf{m}_i) \\ \Delta \mathbf{m}_i = -\eta(\mathbf{x}^t - \mathbf{m}_i) & \text{otherwise} \end{cases}$$



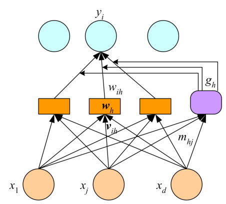
Mixture of Experts

- In RBF, each local fit is a constant, w_{ih}, second layer weight
- In MoE, each local fit is a linear function of $\mathbf{x}$, a local expert: $w_{ih}^t = \mathbf{v}_{ih}^T \mathbf{x}^t$

(Jacobs et al., 1991)



MoE as Models Combined



- Radial gating:

$$g_h^t = \frac{\exp\left[-\|\mathbf{x}^t - \mathbf{m}_h\|^2 / 2s_h^2\right]}{\sum_l \exp\left[-\|\mathbf{x}^t - \mathbf{m}_l\|^2 / 2s_l^2\right]}$$

- Softmax gating:

$$g_h^t = \frac{\exp\left[\mathbf{m}_h^T \mathbf{x}^t\right]}{\sum_l \exp\left[\mathbf{m}_l^T \mathbf{x}^t\right]}$$

Cooperative MoE

- Regression

$$E(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}) = \frac{1}{2}\sum_t \sum_i (r_i^t - y_i^t)^2$$

$$\Delta \mathbf{v}_{ih} = \eta \sum_t (r_i^t - y_{ih}^t) g_h^t \mathbf{x}^t$$

$$\Delta m_{hj} = \eta \sum_t (r_i^t - y_{ih}^t)(w_{ih}^t - y_i^t) g_h^t x_j^t$$

Competitive MoE: Regression

$$\mathcal{L}(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}) = \sum_t \log \sum_h g_h^t \exp\left[-\frac{1}{2}\sum_i (r_i^t - y_{ih}^t)^2\right]$$

$$y_{ih}^t = w_{ih} = \mathbf{v}_{ih}\mathbf{x}^t$$

$$\Delta \mathbf{v}_{ih} = \eta \sum_t (r_i^t - y_{ih}^t) f_h^t \mathbf{x}^t$$

$$\Delta \mathbf{m}_h = \eta \sum_t (f_h^t - g_h^t) \mathbf{x}^t$$

Competitive MoE: Classification

$$\mathcal{L}(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}) = \sum_t \log \sum_h g_h^t \prod_i (y_{ih}^t)^{r_i^t}$$

$$= \sum_t \log \sum_h g_h^t \exp\left[\sum_i r_i^t \log y_{ih}^t\right]$$

$$y_{ih}^t = \frac{\exp w_{ih}}{\sum_k \exp w_{kh}} = \frac{\exp \mathbf{v}_{ih}\mathbf{x}^t}{\sum_k \exp \mathbf{v}_{kh}\mathbf{x}^t}$$

Hierarchical Mixture of Experts

- Tree of MoE where each MoE is an expert in a higher-level MoE
- Soft decision tree: Takes a weighted (gating) average of all leaves (experts), as opposed to using a single path and a single leaf
- Can be trained using EM (Jordan and Jacobs, 1994)

i2ml3e-chap13.pdf

Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 13:

KERNEL MACHINES

Kernel Machines

- Discriminant-based: No need to estimate densities first
- Define the discriminant in terms of support vectors
- The use of kernel functions, application-specific measures of similarity
- No need to represent instances as vectors
- Convex optimization problems with a unique solution

Optimal Separating Hyperplane

$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_t \text{ where } r^t = \begin{cases} +1 & \text{if } \mathbf{x}^t \in C_1 \\ -1 & \text{if } \mathbf{x}^t \in C_2 \end{cases}$$

find $\mathbf{w}$ and w_0 such that

$\mathbf{w}^T\mathbf{x}^t + w_0 \geq +1$ for $r^t = +1$

$\mathbf{w}^T\mathbf{x}^t + w_0 \leq +1$ for $r^t = -1$

which can be rewritten as

$r^t(\mathbf{w}^T\mathbf{x}^t + w_0) \geq +1$

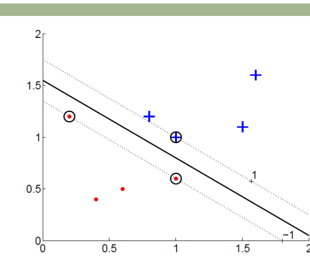
(Cortes and Vapnik, 1995; Vapnik, 1995)

Margin

- Distance from the discriminant to the closest instances on either side
- Distance of x to the hyperplane is $\frac{|\mathbf{w}^T\mathbf{x}^t + w_0|}{\|\mathbf{w}\|}$
- We require $\frac{r^t(\mathbf{w}^T\mathbf{x}^t + w_0)}{\|\mathbf{w}\|} \geq \rho, \forall t$
- For a unique sol'n, fix $\rho\|\mathbf{w}\|=1$, and to max margin

$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t(\mathbf{w}^T\mathbf{x}^t + w_0) \geq +1, \forall t$$

Margin



$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \ge +1, \forall t$$

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N}\alpha^t\left[r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) - 1\right]$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N}\alpha^t r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) + \sum_{t=1}^{N}\alpha^t$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{t=1}^{N}\alpha^t r^t \mathbf{x}^t$$

$$\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \sum_{t=1}^{N}\alpha^t r^t = 0$$

$$L_d = \frac{1}{2}\left(\mathbf{w}^T\mathbf{w}\right) - \mathbf{w}^T\sum_t \alpha^t r^t \mathbf{x}^t - w_0\sum_t \alpha^t r^t + \sum_t \alpha^t$$

$$= -\frac{1}{2}\left(\mathbf{w}^T\mathbf{w}\right) + \sum_t \alpha^t$$

$$= -\frac{1}{2}\sum_t\sum_s \alpha^t\alpha^s r^t r^s \left(\mathbf{x}^t\right)^T\mathbf{x}^s + \sum_t \alpha^t$$

$$\text{subject to } \sum_t \alpha^t r^t = 0 \text{ and } \alpha^t \ge 0, \forall t$$

Most α^t are 0 and only a small number have $\alpha^t > 0$; they are the support vectors

Soft Margin Hyperplane

- Not linearly separable

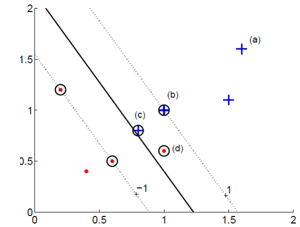
$$r^t\left(\mathbf{w}^T x^t + w_0\right) \ge 1 - \xi^t$$

- Soft error

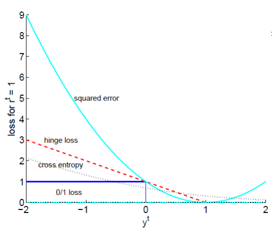
$$\sum_t \xi^t$$

- New primal is

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_t \xi^t - \sum_t \alpha^t\left[r^t\left(\mathbf{w}^T x^t + w_0\right) - 1 + \xi^t\right] - \sum_t \mu^t \xi^t$$



Hinge Loss



$$\begin{cases} 0 & \text{if } y^t r^t \ge 1 \\ 1 - y^t r^t & \text{otherwise} \end{cases}$$

ν-SVM

$$\min \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{N}\sum_t \xi^t$$

subject to

$$r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \ge \rho - \xi^t, \xi^t \ge 0, \rho \ge 0$$

$$L_d = -\frac{1}{2}\sum_{t=1}^{N}\sum_s \alpha^t\alpha^s r^t r^s \left(x^t\right)^T x^s$$

subject to

$$\sum_t \alpha^t r^t = 0, 0 \le \alpha^t \le \frac{1}{N}, \sum_t \alpha^t \le \nu$$

ν controls the fraction of support vectors

Kernel Trick

- Preprocess input $\mathbf{x}$ by basis functions

  $\mathbf{z} = \boldsymbol{\varphi}(\mathbf{x})$ $\qquad g(\mathbf{z}) = \mathbf{w}^T\mathbf{z}$

  $g(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x})$

- The SVM solution

$$\mathbf{w} = \sum_t \alpha^t r^t \mathbf{z}^t = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\mathbf{x}^t\right)$$

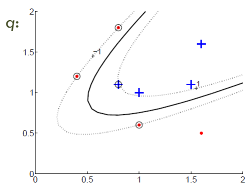
$$g(\mathbf{x}) = \mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}) = \sum_t \alpha^t r^t \boxed{\boldsymbol{\varphi}\left(\mathbf{x}^t\right)^T \boldsymbol{\varphi}(\mathbf{x})}$$

$$g(\mathbf{x}) = \sum_t \alpha^t r^t \boxed{K\left(\mathbf{x}^t, \mathbf{x}\right)}$$

Vectorial Kernels

- Polynomials of degree q:

$$K\left(\mathbf{x}^t, \mathbf{x}\right) = \left(\mathbf{x}^T\mathbf{x}^t + 1\right)^q$$



$$K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T\mathbf{y} + 1\right)^2$$

$$= \left(x_1y_1 + x_2y_2 + 1\right)^2$$

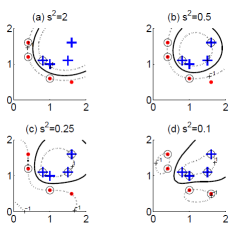
$$= 1 + 2x_1y_1 + 2x_2y_2 + 2x_1x_2y_1y_2 + x_1^2y_1^2 + x_2^2y_2^2$$

$$\phi(\mathbf{x}) = \left[1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2\right]^T$$

Vectorial Kernels

- Radial-basis functions:

$$K\left(\mathbf{x}^t, \mathbf{x}\right) = \exp\left[-\frac{\left\|\mathbf{x}^t - \mathbf{x}\right\|^2}{2s^2}\right]$$



Defining kernels

- Kernel "engineering"
- Defining good measures of similarity
- String kernels, graph kernels, image kernels, ...
- Empirical kernel map: Define a set of templates $\boldsymbol{m}_i$ and score function $s(\boldsymbol{x}, \boldsymbol{m}_i)$

  $\phi(\boldsymbol{x}^t) = [s(\boldsymbol{x}^t, \boldsymbol{m}_1), s(\boldsymbol{x}^t, \boldsymbol{m}_2), \ldots, s(\boldsymbol{x}^t, \boldsymbol{m}_M)]$

  and

  $K(\boldsymbol{x}, \boldsymbol{x}^t) = \phi(\boldsymbol{x})^T \phi(\boldsymbol{x}^t)$

Multiple Kernel Learning

- Fixed kernel combination

$$K(\mathbf{x}, \mathbf{y}) = \begin{cases} cK(\mathbf{x}, \mathbf{y}) \\ K_1(\mathbf{x}, \mathbf{y}) + K_2(\mathbf{x}, \mathbf{y}) \\ K_1(\mathbf{x}, \mathbf{y})K_2(\mathbf{x}, \mathbf{y}) \end{cases}$$

- Adaptive kernel combination

$$K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m}\eta_i K_i(\mathbf{x}, \mathbf{y})$$

$$L_d = \sum_t \alpha^t - \frac{1}{2}\sum_t\sum_s \alpha^t\alpha^s r^t r^s \sum_i \eta_i K_i\left(\mathbf{x}^t, \mathbf{x}^s\right)$$

$$g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i K_i\left(\mathbf{x}^t, \mathbf{x}\right)$$

- Localized kernel combination $\quad g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i(\mathbf{x} \mid \theta) K_i\left(\mathbf{x}^t, \mathbf{x}\right)$

Multiclass Kernel Machines

- 1-vs-all
- Pairwise separation
- Error-Correcting Output Codes (section 17.5)
- Single multiclass optimization

$$\min \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{w}_i\|^2 + C\sum_i\sum_t \xi_i^t$$

subject to

$$\mathbf{w}_{z^t}{}^T\mathbf{x}^t + w_{z^t0} \ge \mathbf{w}_i{}^T\mathbf{x}^t + w_{i0} + 2 - \xi_i^t, \forall i \ne z^t, \xi_i^t \ge 0$$

SVM for Regression

- Use a linear model (possibly kernelized)

  $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$

- Use the ε-sensitive error function

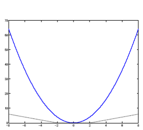
$$e_\varepsilon\left(r^t, f\left(\mathbf{x}^t\right)\right) = \begin{cases} 0 & \text{if } \left|r^t - f\left(\mathbf{x}^t\right)\right| < \varepsilon \\ \left|r^t - f\left(\mathbf{x}^t\right)\right| - \varepsilon & \text{otherwise} \end{cases}$$

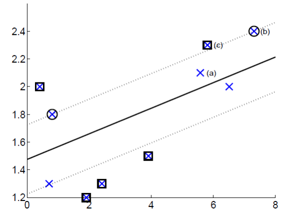
$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_t\left(\xi_+^t + \xi_-^t\right)$$

$$r^t - \left(\mathbf{w}^T\mathbf{x} + w_0\right) \le \varepsilon + \xi_+^t$$

$$\left(\mathbf{w}^T\mathbf{x} + w_0\right) - r^t \le \varepsilon + \xi_-^t$$

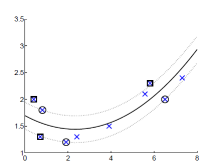
$$\xi_+^t, \xi_-^t \ge 0$$



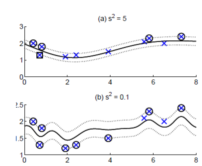


Kernel Regression

- Polynomial kernel



- Gaussian kernel



Kernel Machines for Ranking

- We require not only that scores be correct order but at least +1 unit margin.
- Linear case:

$$\min \frac{1}{2}\|\mathbf{w}_i\|^2 + C\sum_t \xi_i^t$$

subject to

$$\mathbf{w}^T\mathbf{x}^u \ge \mathbf{w}^T\mathbf{x}^v + 1 - \xi^t, \forall t : r^u \prec r^v, \xi_i^t \ge 0$$

One-Class Kernel Machines

- Consider a sphere with center $\mathbf{a}$ and radius R

$$\min R^2 + C\sum_t \xi^t$$

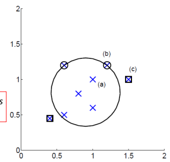
subject to

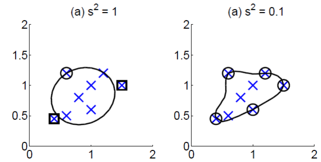
$$\|\mathbf{x}^t - \mathbf{a}\| \le R^2 + \xi^t, \xi^t \ge 0$$

$$L_d = \boxed{\sum_t \alpha^t (x^t)^T x^s} - \sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \boxed{(x^t)^T x^s}$$

subject to

$$0 \le \alpha^t \le C, \sum_t \alpha^t = 1$$





Large Margin Nearest Neighbor

- Learns the matrix $\mathbf{M}$ of Mahalanobis metric
  $D(\mathbf{x}^i, \mathbf{x}^j) = (\mathbf{x}^i - \mathbf{x}^j)^T \mathbf{M} (\mathbf{x}^i - \mathbf{x}^j)$
- For three instances i, j, and l, where i and j are of the same class and l different, we require
  $D(\mathbf{x}^i, \mathbf{x}^l) > D(\mathbf{x}^i, \mathbf{x}^j) + 1$
  and if this is not satisfied, we have a slack for the difference and we learn M to minimize the sum of such slacks over all i,j,l triples (j and l being one of k neighbors of i, over all i)

Learning a Distance Measure

- LMNN algorithm (Weinberger and Saul 2009)

$$(1-\mu)\sum_{i,j} \mathcal{D}(\mathbf{x}^i, \mathbf{x}^j) + \mu \sum_{i,j,l} (1 - y_{il})\xi_{ijl}$$

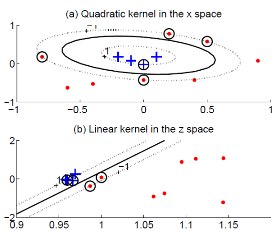
subject to

$$\mathcal{D}(\mathbf{x}^i, \mathbf{x}^l) \ge \mathcal{D}(\mathbf{x}^i, \mathbf{x}^j) + 1 - \xi^{ijl}, \text{ if } r^i = r^j \text{ and } r^i \ne r^l$$
$$\xi^{ijl} \ge 0$$

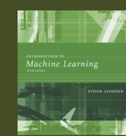
- LMCA algorithm (Torresani and Lee 2007) uses a similar approach where $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ and learns $\mathbf{L}$

Kernel Dimensionality Reduction

- Kernel PCA does PCA on the kernel matrix (equal to canonical PCA with a linear kernel)
- Kernel LDA, CCA



i2ml3e-chap14.pdf



Lecture Slides for
INTRODUCTION TO MACHINE LEARNING
3RD EDITION

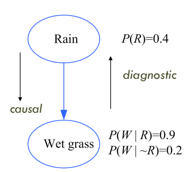
ETHEM ALPAYDIN
alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 16:
GRAPHICAL MODELS

Graphical Models

- Aka Bayesian networks, probabilistic networks
- Nodes are hypotheses (random vars) and the probabilities corresponds to our belief in the truth of the hypothesis
- Arcs are direct influences between hypotheses
- The structure is represented as a directed acyclic graph (DAG)
- The parameters are the conditional probabilities in the arcs (Pearl, 1988, 2000; Jensen, 1996; Lauritzen, 1996)

Causes and Bayes' Rule



Diagnostic inference: Knowing that the grass is wet, what is the probability that rain is the cause?

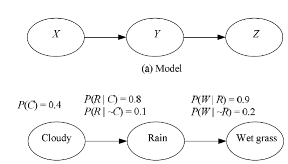
$$P(R|W) = \frac{P(W|R)P(R)}{P(W)}$$
$$= \frac{P(W|R)P(R)}{P(W|R)P(R) + P(W|\sim R)P(\sim R)}$$
$$= \frac{0.9 \times 0.4}{0.9 \times 0.4 + 0.2 \times 0.6} = 0.75$$

Conditional Independence

- X and Y are independent if
  $P(X,Y) = P(X)P(Y)$
- X and Y are conditionally independent given Z if
  $P(X,Y|Z) = P(X|Z)P(Y|Z)$
  or
  $P(X|Y,Z) = P(X|Z)$
- Three canonical cases: Head-to-tail, Tail-to-tail, head-to-head

Case 1: Head-to-Head

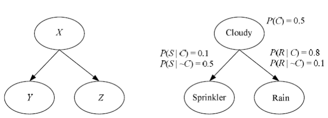
- $P(X,Y,Z) = P(X)P(Y|X)P(Z|Y)$



- $P(W|C) = P(W|R)P(R|C) + P(W|\sim R)P(\sim R|C)$

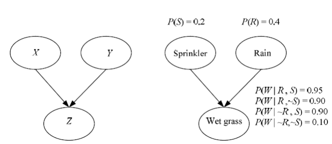
Case 2: Tail-to-Tail

- $P(X,Y,Z) = P(X)P(Y|X)P(Z|X)$

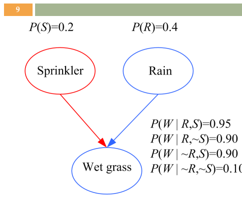


Case 3: Head-to-Head

- $P(X,Y,Z) = P(X)P(Y)P(Z|X,Y)$



Causal vs Diagnostic Inference

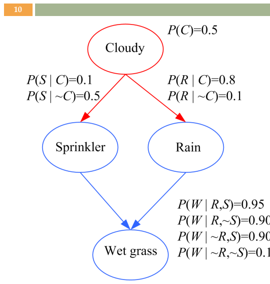


Causal inference: If the sprinkler is on, what is the probability that the grass is wet?

P(W|S) = P(W|R,S) P(R|S) + P(W|~R,S) P(~R|S)
= P(W|R,S) P(R) + P(W|~R,S) P(~R)
= 0.95 0.4 + 0.9 0.6 = 0.92

Diagnostic inference: If the grass is wet, what is the probability that the sprinkler is on? P(S|W) = 0.35 > 0.2 P(S)
P(S|R,W) = 0.21 *Explaining away: Knowing that it has rained decreases the probability that the sprinkler is on.*

Causes



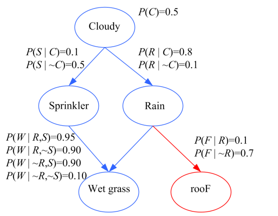
Causal inference:
P(W|C) = P(W|R,S) P(R,S|C) + P(W|~R,S) P(~R,S|C) + P(W|R,~S) P(R,~S|C) + P(W|~R,~S) P(~R,~S|C)

and use the fact that
P(R,S|C) = P(R|C) P(S|C)

Diagnostic: P(C|W) = ?

Exploiting the Local Structure

11



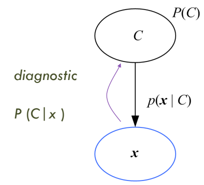
P (F | C) = ?

$$P(C,S,R,W,F)=P(C)P(S|C)P(R|C)P(W|S,R)P(F|R)$$

$$P(X_1,\dots X_d)=\prod_{i=1}^{d}P(X_i\,|\,\text{parents}(X_i))$$

Classification

12

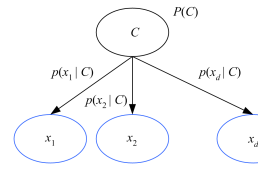


Bayes' rule inverts the arc:

$$P(C|\mathbf{x})=\frac{p(\mathbf{x}|C)P(C)}{p(\mathbf{x})}$$

Naive Bayes' Classifier

13

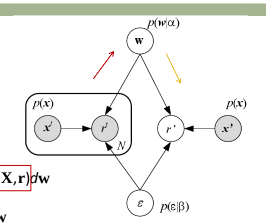


Given C, x_j are independent:

$p(\mathbf{x}|C) = p(x_1|C)\, p(x_2|C) \dots p(x_d|C)$

Linear Regression

14



$$p(r'|\mathbf{x}',\mathbf{r},\mathbf{X})=\int \boxed{p(r'|\mathbf{x}',\mathbf{w})}\boxed{p(\mathbf{w}\,|\,\mathbf{X},\mathbf{r})}d\mathbf{w}$$

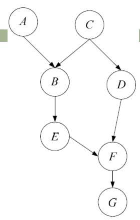
$$=\int p(r'|\mathbf{x}',\mathbf{w})\frac{p(\mathbf{r}\,|\,\mathbf{X},\mathbf{w})p(\mathbf{w})}{p(\mathbf{r})}d\mathbf{w}$$

$$\propto \int p(r'|\mathbf{x}',\mathbf{w})\prod_t p(r^t\,|\,\mathbf{x}^t,\mathbf{w})p(\mathbf{w})d\mathbf{w}$$

d-Separation

15

- A path from node A to node B is blocked if
  - a) The directions of edges on the path meet head-to-tail (case 1) or tail-to-tail (case 2) and the node is in C, or
  - b) The directions of edges meet head-to-head (case 3) and neither that node nor any of its descendants is in C.
- If all paths are blocked, A and B are d-separated (conditionally independent) given C.

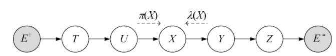


BCDF is blocked given C.
BEFG is blocked by *F*.
BEFD is blocked unless *F* (or G) is given.

Belief Propagation (Pearl, 1988)

16

- Chain:



$$P(X|E)=\frac{P(E|X)P(X)}{P(E)}=\frac{P(E^+,E^-|X)P(X)}{P(E)}$$

$$=\frac{P(E^+|X)P(E^-|X)P(X)}{P(E)}=\alpha\pi(X)\lambda(X)$$

$$\pi(X)=\sum_U P(X|U)\pi(U)$$

$$\lambda(X)=\sum_Y P(Y|X)\lambda(Y)$$

Trees

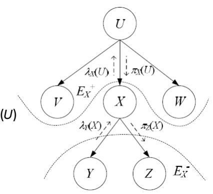
17

$$\lambda(X)=P(E_X^-\,|\,X)=\lambda_Y(X)\lambda_Z(X)$$

$$\lambda_X(U)=\sum_X \lambda(X)P(X\,|\,U)$$

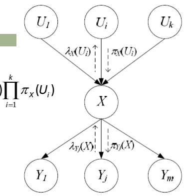
$$\pi(X)=P(X\,|\,E_X^+)=\sum_U P(X\,|\,U)\pi_X(U)$$

$$\pi_y(X)=\alpha\lambda_Z(X)\pi(X)$$



Polytrees

18



$$\pi(X)=P(X\,|\,E_X^+)=\sum_{U_1}\sum_{U_2}\cdots\sum_{U_k}P(X\,|\,U_1,U_2,\cdots,U_k)\prod_{i=1}^{k}\pi_X(U_i)$$

$$\pi_{y_j}(X)=\alpha\prod_{s\neq j}\lambda_{Y_s}(X)\pi(X)$$

$$\lambda_X(U_i)=\beta\sum_X\lambda(X)\sum_{U_{r\neq i}}P(X\,|\,U_1,U_2,\cdots,U_k)\prod_{r\neq i}\pi_X(U_r)$$

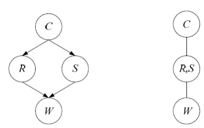
$$\lambda(X)=\prod_{j=1}^{m}\lambda_{Y_j}(X)$$

How can we model $P(X|U_1,U_2,\dots,U_k)$ cheaply?

Junction Trees

19

- If X does not separate E^+ and E^-, we convert it into a junction tree and then apply the polytree algorithm



Tree of moralized, clique nodes

Undirected Graphs: Markov Random Fields

20

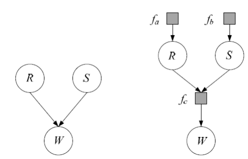
- In a Markov random field, dependencies are symmetric, for example, pixels in an image
- In an undirected graph, A and B are independent if removing C makes them unconnected.
- Potential function $\psi_c(X_c)$ shows how favorable is the particular configuration X over the clique C
- The joint is defined in terms of the clique potentials

$$p(X)=\frac{1}{Z}\prod_C\psi_C(X_C)\text{ where normalizer }Z=\sum_X\prod_C\psi_C(X_C)$$

Factor Graphs

21

- Define new factor nodes and write the joint in terms of them



$$p(X)=\frac{1}{Z}\prod_S f_S(X_S)$$

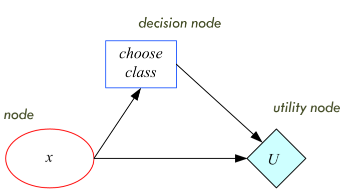
Learning a Graphical Model

22

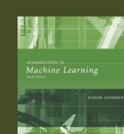
- Learning the conditional probabilities, either as tables (for discrete case with small number of parents), or as parametric functions
- Learning the structure of the graph: Doing a state-space search over a score function that uses both goodness of fit to data and some measure of complexity

Influence Diagrams

23



i2ml3e-chap15.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 15:

HIDDEN MARKOV MODELS

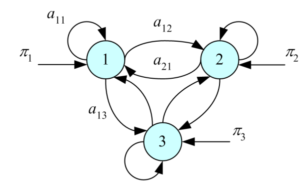
Introduction

- Modeling dependencies in input; no longer iid
- Sequences:
  - Temporal: In speech; phonemes in a word (dictionary), words in a sentence (syntax, semantics of the language).
    In handwriting, pen movements
  - Spatial: In a DNA sequence; base pairs

Discrete Markov Process

- N states: $S_1, S_2, ..., S_N$ State at "time" t, $q_t = S_i$
- First-order Markov
  $P(q_{t+1}=S_j \mid q_t=S_i, q_{t-1}=S_k,...) = P(q_{t+1}=S_j \mid q_t=S_i)$
- Transition probabilities
  $a_{ij} \equiv P(q_{t+1}=S_j \mid q_t=S_i)$ $\quad a_{ij} \geq 0$ and $\Sigma_{j=1}^N a_{ij}=1$
- Initial probabilities
  $\pi_i \equiv P(q_1=S_i)$ $\quad \Sigma_{j=1}^N \pi_i=1$

Stochastic Automaton



Example: Balls and Urns

- Three urns each full of balls of one color
  S_1: red, S_2: blue, S_3: green

$$\Pi = [0.5, 0.2, 0.3]^T \qquad \mathbf{A} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

$$O = \{S_1, S_1, S_3, S_3\}$$
$$P(O \mid \mathbf{A}, \Pi) = P(S_1) \cdot P(S_1 \mid S_1) \cdot P(S_3 \mid S_1) \cdot P(S_3 \mid S_3)$$
$$= \pi_1 \cdot a_{11} \cdot a_{13} \cdot a_{33}$$
$$= 0.5 \cdot 0.4 \cdot 0.3 \cdot 0.8 = 0.048$$

Balls and Urns: Learning

- Given K example sequences of length T

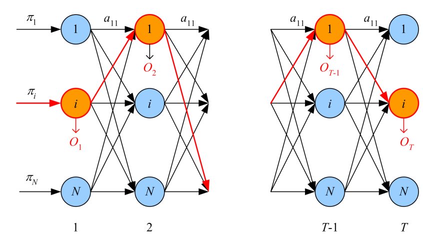
$$\hat{\pi}_i = \frac{\#\{\text{sequences starting with } S_i\}}{\#\{\text{sequences}\}} = \frac{\sum_k 1(q_1^k = S_i)}{K}$$

$$\hat{a}_{ij} = \frac{\#\{\text{transitions from } S_i \text{ to } S_j\}}{\#\{\text{transitions from } S_i\}} = \frac{\sum_k \sum_{t=1}^{T-1} 1(q_t^k = S_i \text{ and } q_{t+1}^k = S_j)}{\sum_k \sum_{t=1}^{T-1} 1(q_t^k = S_i)}$$

Hidden Markov Models

- States are not observable
- Discrete observations $\{v_1, v_2, ..., v_M\}$ are recorded; a probabilistic function of the state
- Emission probabilities
  $b_j(m) \equiv P(O_t=v_m \mid q_t=S_j)$
- Example: In each urn, there are balls of different colors, but with different probabilities.
- For each observation sequence, there are multiple state sequences

HMM Unfolded in Time



Elements of an HMM

- N: Number of states
- M: Number of observation symbols
- $\mathbf{A} = [a_{ij}]$: N by N state transition probability matrix
- $\mathbf{B} = b_j(m)$: N by M observation probability matrix
- $\mathbf{\Pi} = [\pi_i]$: N by 1 initial state probability vector

$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$, parameter set of HMM

Three Basic Problems of HMMs

1. Evaluation: Given λ, and O, calculate $P(O \mid \lambda)$
2. State sequence: Given λ, and O, find Q^* such that
   $P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$
3. Learning: Given $X=\{O^k\}_k$, find λ^* such that
   $P(X \mid \lambda^*) = \max_\lambda P(X \mid \lambda)$

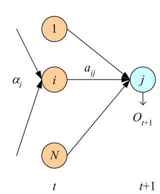
(Rabiner, 1989)

Evaluation

- Forward variable:

$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$

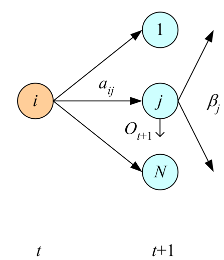
Initialization:
$$\alpha_1(i) = \pi_i b_i(O_1)$$
Recursion:
$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij}\right] b_j(O_{t+1})$$
$$P(O \mid \lambda) = \sum_{i=1}^N \alpha_T(i)$$



Backward variable:

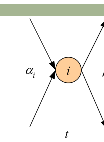
$$\beta_t(i) \equiv P(O_{t+1} \cdots O_T \mid q_t = S_i, \lambda)$$

Initialization:
$$\beta_T(i) = 1$$
Recursion:
$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$



Finding the State Sequence

$$\gamma_t(i) \equiv P(q_t = S_i \mid O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^N \alpha_t(j)\beta_t(j)}$$



Choose the state that has the highest probability, for each time step:
$q_t^* = \arg\max_i \gamma_t(i)$

No!

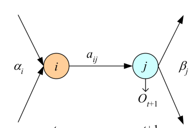
Viterbi's Algorithm

$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t=S_i, O_1 \cdots O_t \mid \lambda)$

- Initialization:
  $\delta_1(i) = \pi_i b_i(O_1)$, $\psi_1(i) = 0$
- Recursion:
  $\delta_t(j) = \max_i \delta_{t-1}(i) a_{ij} b_j(O_t)$, $\psi_t(j) = \text{argmax}_i\, \delta_{t-1}(i) a_{ij}$
- Termination:
  $p^* = \max_i \delta_T(i)$, $q_T^* = \text{argmax}_i\, \delta_T(i)$
- Path backtracking:
  $q_t^* = \psi_{t+1}(q_{t+1}^*)$, $t=T-1, T-2, ..., 1$

Learning

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)$$
$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_k \sum_l \alpha_t(k) a_{kl} b_l(O_{t+1}) \beta_{t+1}(l)}$$



Baum-Welch algorithm (EM):

$$z_i^t = \begin{cases} 1 & \text{if } q_t = S_i \\ 0 & \text{otherwise} \end{cases} \qquad z_{ij}^t = \begin{cases} 1 & \text{if } q_t = S_i \text{ and } q_{t+1} = S_j \\ 0 & \text{otherwise} \end{cases}$$

Baum-Welch (EM)

E-step: $E[z_i^t] = \gamma_t(i) \quad E[z_{ij}^t] = \xi_t(i,j)$

M-step:

$$\hat{\pi}_i = \frac{\sum_{k=1}^K \gamma_1^k(i)}{K} \qquad \hat{a}_{ij} = \frac{\sum_{k=1}^K \sum_{t=1}^{T_k-1} \xi_t^k(i,j)}{\sum_{k=1}^K \sum_{t=1}^{T_k-1} \gamma_t^k(i)}$$

$$\hat{b}_j(m) = \frac{\sum_{k=1}^K \sum_{t=1}^{T_k-1} \gamma_t^k(j) 1(O_t^k = v_m)}{\sum_{k=1}^K \sum_{t=1}^{T_k-1} \gamma_t^k(i)}$$

Continuous Observations

- Discrete:
  $$P(O_t \mid q_t = S_j, \lambda) = \prod_{m=1}^M b_j(m)^{r_m^t} \qquad r_m^t = \begin{cases} 1 & \text{if } O_t = v_m \\ 0 & \text{otherwise} \end{cases}$$
- Gaussian mixture (Discretize using k-means):
  $$P(O_t \mid q_t = S_j, \lambda) = \sum_{l=1}^L P(G_{jl}) p(O_t \mid q_t = S_j, G_l, \lambda)$$
  $$\sim \mathcal{N}(\mu_l, \Sigma_l)$$
- Continuous:
  $$P(O_t \mid q_t = S_j, \lambda) \sim \mathcal{N}(\mu_j, \sigma_j^2)$$

Use EM to learn parameters, e.g.,
$$\hat{\mu}_j = \frac{\sum_t \gamma_t(j) O_t}{\sum_t \gamma_t(j)}$$

HMM with Input

19

- Input-dependent observations:

$$P(O_t \mid q_t = S_j, x^t, \lambda) \sim \mathcal{N}(g_j(x^t \mid \theta_j), \sigma_j^2)$$

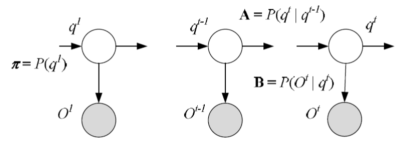
- Input-dependent transitions (Meila and Jordan, 1996; Bengio and Frasconi, 1996):

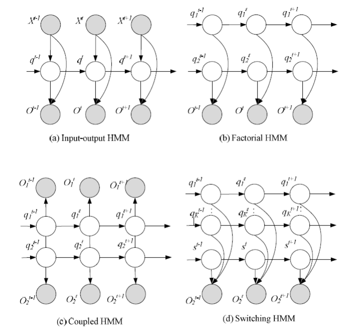
$$P(q_{t+1} = S_j \mid q_t = S_i, x^t)$$

- Time-delay input: $\mathbf{x}^t = \mathbf{f}(O_{t-\tau}, \ldots, O_{t-1})$

HMM as a Graphical Model

20





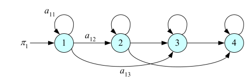
21

Model Selection in HMM

22

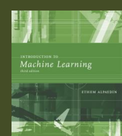
- Left-to-right HMMs:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$



- In classification, for each C_i, estimate $P(O \mid \lambda_i)$ by a separate HMM and use Bayes' rule

$$P(\lambda_i \mid O) = \frac{P(O \mid \lambda_i) P(\lambda_i)}{\sum_j P(O \mid \lambda_j) P(\lambda_j)}$$



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr

http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 16:

BAYESIAN ESTIMATION

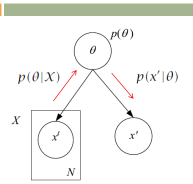
Rationale

3

- Parameters θ not constant, but random variables with a prior, $p(\theta)$
- Bayes' Rule: $p(\theta \mid X) = \dfrac{p(\theta) p(X \mid \theta)}{p(X)}$

Generative Model

4



$$p(x', X, \theta) = p(\theta) p(X \mid \theta) p(x' \mid \theta)$$

$$\begin{aligned} p(x' \mid X) &= \frac{p(x', X)}{p(X)} = \frac{\int p(x', X, \theta) d\theta}{p(X)} = \frac{\int p(\theta) p(X \mid \theta) p(x' \mid \theta) d\theta}{p(X)} \\ &= \int p(x' \mid \theta) p(\theta \mid X) d\theta \end{aligned}$$

Bayesian Approach

5

1. Prior $p(\theta)$ allows us to concentrate on region where θ is likely to lie, ignoring regions where it's unlikely
2. Instead of a single estimate with a single θ, we generate several estimates using several θ and average, weighted by how their probabilities

Even if prior $p(\theta)$ is uninformative, (2) still helps.

MAP estimator does not make use of (2):

$$\theta_{MAP} = \arg\max_\theta p(\theta \mid X)$$

Bayesian Approach

6

$$p(x' \mid X) = \int p(x' \mid \theta) p(\theta \mid X) d\theta$$

- In certain cases, it is easy to integrate
- Conjugate prior: Posterior has the same density as prior
- Sampling (Markov Chain Monte Carlo): Sample from the posterior and average
- Approximation: Approximate the posterior with a model easier to integrate
  - Laplace approximation: Use a Gaussian
  - Variational approximation: Split the multivariate density into a set of simpler densities using independencies

Estimating the Parameters of a Distribution: Discrete case

- x^t_i=1 if in instance t is in state i, probability of state i is q_i
- Dirichlet prior, α_i are hyperparameters

$$Dirichlet(\mathbf{q} \mid \boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1) \cdots \Gamma(\alpha_K)} \prod_{i=1}^K q_i^{\alpha_i - 1}$$

- Sample likelihood

$$p(X \mid \mathbf{q}) = \prod_{t=1}^N \prod_{i=1}^K q_i^{x_i^t}$$

- Posterior

$$\begin{aligned} p(\mathbf{q} \mid \boldsymbol{\alpha}) &= \frac{\Gamma(\alpha_0 + N)}{\Gamma(\alpha_1 + N_1) \cdots \Gamma(\alpha_K + N_K)} \prod_{i=1}^K q_i^{\alpha_i + N_i - 1} \\ &= Dirichlet(\mathbf{q} \mid \boldsymbol{\alpha} + \mathbf{n}) \end{aligned}$$

- Dirichlet is a conjugate prior
- With K=2, Dirichlet reduced to Beta

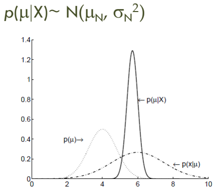
Estimating the Parameters of a Distribution: Continuous case

8

- $p(x^t) \sim N(\mu, \sigma^2)$
- Gaussian prior for μ, $p(\mu) \sim N(\mu_0, \sigma_0^2)$
- Posterior is also Gaussian $p(\mu \mid X) \sim N(\mu_N, \sigma_N^2)$ where

$$\mu_N = \frac{\sigma^2}{N\sigma_0^2 + \sigma^2} \mu_0 + \frac{N\sigma_0^2}{N\sigma_0^2 + \sigma^2} m$$

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_0^2} + \frac{N}{\sigma^2}$$



Gaussian: Prior on Variance

9

- Let's define a prior (gamma) on precision $\lambda = 1/\sigma^2$

$$p(\lambda) \sim \text{gamma}(a_0, b_0) = \frac{1}{\Gamma(a_0)} b_0^{a_0} \lambda^{a_0 - 1} \exp(-b_0 \lambda)$$

$$\begin{aligned} p(\mathcal{X} \mid \lambda) &= \prod_t \frac{\lambda^{1/2}}{\sqrt{2\pi}} \exp\left[-\frac{\lambda}{2}(x^t - \mu)^2\right] \\ &= \lambda^{N/2} (2\pi)^{-N/2} \exp\left[-\frac{\lambda}{2} \sum_t (x^t - \mu)^2\right] \end{aligned}$$

$$\begin{aligned} p(\lambda \mid \mathcal{X}) &\propto p(\mathcal{X} \mid \lambda) p(\lambda) \\ &\sim \text{gamma}(a_N, b_N) \end{aligned}$$

$$\begin{aligned} a_N &= a_0 + N/2 = \frac{\nu_0 + N}{2} \\ b_N &= b_0 + \frac{N}{2} s^2 = \frac{\nu_0}{2} s_0^2 + \frac{N}{2} s^2 \end{aligned}$$

Joint Prior and Making a Prediction

10

$$p(\mu, \lambda) = p(\mu \mid \lambda) p(\lambda)$$

$$p(\mu, \lambda \mid \mathcal{X}) \sim \text{normal-gamma}(\mu_N, \kappa_N, a_N, b_N)$$

where

$$\begin{aligned} \kappa_N &= \kappa_0 + N \\ \mu_N &= \frac{\kappa_0 \mu_0 + Nm}{\kappa_N} \\ a_N &= a_0 + N/2 \\ b_N &= b_0 + \frac{N}{2} s^2 + \frac{\kappa_0 N}{2\kappa_N} (m - \mu_0)^2 \end{aligned}$$

$$\begin{aligned} p(x \mid \mathcal{X}) &= \int\int p(x \mid \mu, \lambda) p(\mu, \lambda \mid \mathcal{X}) d\mu d\lambda \\ &\sim t_{2a_N}\left(\mu_N, \frac{b_N(\kappa_N + 1)}{a_N \kappa_N}\right) \end{aligned}$$

Multivariate Gaussian

11

$$p(\boldsymbol{x}) \sim \mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Lambda}) \qquad p(\boldsymbol{\mu} \mid \boldsymbol{\Lambda}) \sim \mathcal{N}_d(\boldsymbol{\mu}_0, (1/\kappa_0)\boldsymbol{\Lambda}) \qquad p(\boldsymbol{\Lambda}) \sim \text{Wishart}(\nu_0, \mathbf{V}_0)$$

$$\begin{aligned} p(\mu, \boldsymbol{\Lambda}) &= p(\boldsymbol{\mu} \mid \boldsymbol{\Lambda}) p(\boldsymbol{\Lambda}) \\ &\sim \text{normal-Wishart}(\boldsymbol{\mu}_0, \kappa_0, \nu_0, \mathbf{V}_0) \end{aligned}$$

$$p(\mu, \boldsymbol{\Lambda} \mid \mathcal{X}) \sim \text{normal-Wishart}(\boldsymbol{\mu}_N, \kappa_N, \nu_N, \mathbf{V}_N)$$

$$\begin{aligned} \kappa_N &= \kappa_0 + N \\ \boldsymbol{\mu}_N &= \frac{\kappa_0 \boldsymbol{\mu}_0 + N\boldsymbol{m}}{\kappa_N} \\ \nu_N &= \nu_0 + N \\ \mathbf{V}_N &= \left(\mathbf{V}_0^{-1} + \mathbf{C} + \frac{\kappa_0 N}{\kappa_N} (\boldsymbol{m} - \boldsymbol{\mu}_0)(\boldsymbol{m} - \boldsymbol{\mu}_0)^T\right)^{-1} \end{aligned}$$

$$\begin{aligned} p(\boldsymbol{x} \mid \mathcal{X}) &= \int\int p(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(\boldsymbol{\mu}, \boldsymbol{\Lambda} \mid \mathcal{X}) d\boldsymbol{\mu} d\boldsymbol{\Lambda} \\ &\sim t_{\nu_N - d + 1}\left(\boldsymbol{\mu}_N, \frac{\kappa_N + 1}{\kappa_N(\nu_N - d + 1)} (\mathbf{V}_N)^{-1}\right) \end{aligned}$$

Estimating the Parameters of a Function: Regression

- $r=\mathbf{w}^T\mathbf{x}+\varepsilon$, $p(\varepsilon)\sim N(0,1/\beta)$, and $p(r^t|x^t,\mathbf{w},\beta)\sim N(\mathbf{w}^T\mathbf{x}^t, 1/\beta)$
- Log likelihood

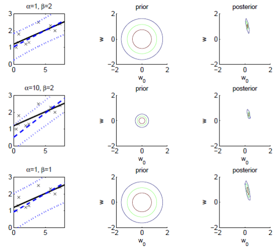
$$L(\mathbf{r}|\mathbf{X},\mathbf{w},\beta)=\log\prod_t p(r^t|\mathbf{x}^t,\mathbf{w},\beta)$$
$$=-N\log\left(\sqrt{2\pi}\right)+N\log\beta-\frac{\beta}{2}\sum_t\left(r^t-\mathbf{w}^T\mathbf{x}^t\right)$$

ML solution $\mathbf{w}_{ML}=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{r}$

- Gaussian conjugate prior: $p(\mathbf{w})\sim N(0,1/\alpha)$
- Posterior: $p(\mathbf{w}|\mathbf{X})\sim N(\mu_N,\Sigma_N)$ where

$$\mu_N=\beta\Sigma_N\mathbf{X}^T\mathbf{r}$$
$$\Sigma_N=(\alpha\mathbf{I}+\beta\mathbf{X}^T\mathbf{X})^{-1}$$

Aka ridge regression/parameter shrinkage/ L2 regularization/weight decay



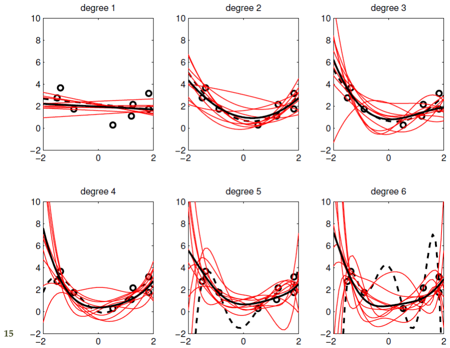
Prior on Noise Variance

$$p(\beta)\sim\text{gamma}(a_0,b_0)\qquad p(\mathbf{w}|\beta)\sim\mathcal{N}(\boldsymbol{\mu}_0,\beta\Sigma_0)$$
$$p(\mathbf{w},\beta)=p(\beta)p(\mathbf{w}|\beta)\sim\text{normal-gamma}(\boldsymbol{\mu}_0,\Sigma_0,a_0,b_0)$$

$$p(\mathbf{w},\beta|\mathbf{X},\mathbf{r})\sim\text{normal-gamma}(\boldsymbol{\mu}_N,\Sigma_N,a_N,b_N)$$

$$\begin{aligned}\Sigma_N &= (\mathbf{X}^T\mathbf{X}+\Sigma_0)^{-1}\\ \boldsymbol{\mu}_N &= \Sigma_N(\mathbf{X}^T\boldsymbol{r}+\Sigma_0\boldsymbol{\mu}_0)\\ a_N &= a_0+N/2\\ b_N &= b_0+\frac{1}{2}(\boldsymbol{r}^T\boldsymbol{r}+\boldsymbol{\mu}_0^T\Sigma_0\boldsymbol{\mu}_0-\boldsymbol{\mu}_N^T\Sigma_N\boldsymbol{\mu}_N)\end{aligned}$$

Markov Chain Monte Carlo (MCMC) sampling



Basis/Kernel Functions

- For new $\mathbf{x}'$, the estimate r' is calculated as

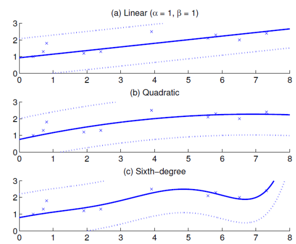
$$r'=(\mathbf{x}')^T$$
$$=\beta(\mathbf{x}')^T\Sigma_N\mathbf{X}^T\mathbf{r}$$
$$=\sum_t\beta(\mathbf{x}')^T\Sigma_N\mathbf{x}^t r^t$$

Dual representation

- Linear kernel
- For any other $\phi(\mathbf{x})$, we can write $K(\mathbf{x}',\mathbf{x})=\phi(\mathbf{x}')^T\phi(\mathbf{x})$

$$r'=\sum_t\beta(\mathbf{x}')^T\Sigma_N\mathbf{x}^t r^t\sum_t\beta K(\mathbf{x}',\mathbf{x}^t)r^t$$

Kernel Functions



What's in a Prior?

- Defining a prior is subjective
- Uninformative prior if no prior preference
- How high to go?

Level I: $p(x|X)=\int p(x|\theta)p(\theta|X)d\theta$

Level II: $p(x|X)=\int p(x|\theta)p(\theta|X,\alpha)p(\alpha)d\theta d\alpha$

- Empirical Bayes: Use one good α^*

Level II ML: $p(x|X)=\int p(x|\theta)p(\theta|X,\alpha^*)d\theta$

Bayesian Model Comparison

- Marginal likelihood of a model:

$$p(X|\mathcal{M})=\int p(X|\theta,\mathcal{M})p(\theta|\mathcal{M})d\theta$$

- Posterior probability of model given data:

$$p(\mathcal{M}|X)=\frac{p(X|\mathcal{M})p(\mathcal{M})}{p(X)}$$

- Bayes' factor:

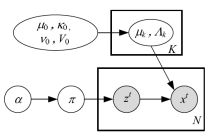
$$\frac{P(\mathcal{M}_1|X)}{P(\mathcal{M}_0|X)}=\frac{P(X|\mathcal{M}_1)}{P(X|\mathcal{M}_0)}\frac{P(\mathcal{M}_1)}{P(\mathcal{M}_0)}$$

- Approximations:

BIC: $\log p(X|\mathcal{M})\approx\text{BIC}\equiv\log p(X|\theta_{ML},\mathcal{M})-\frac{|\mathcal{M}|}{2}\log N$

AIC: $\text{AIC}\equiv\log p(X|\theta_{ML},\mathcal{M})-|\mathcal{M}|$

Mixture Model

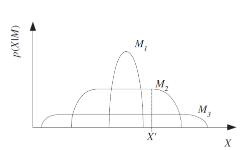


$$p(\mathbf{x})=\sum_{i=1}^k P(\mathcal{G}_i)p(\mathbf{x}|\mathcal{G}_i)$$

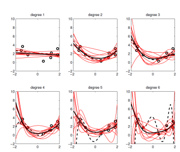
$$\begin{aligned}p(\Phi) &= p(\boldsymbol{\pi})\prod_i p(\boldsymbol{\mu}_i,\Lambda_i)\\ &= \text{Dirichlet}(\boldsymbol{\pi}|\boldsymbol{\alpha})\prod_i\text{normal-Wishart}(\boldsymbol{\mu}_0,\kappa_0,\nu_0,\mathbf{V}_0)\end{aligned}$$

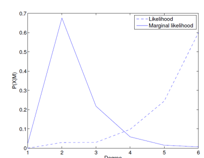
$$\mathcal{Q}(\Phi|\Phi^l)=\sum_t\sum_i h_i^t\log\pi_i+\sum_t\sum_i h_i^t\log p_i(x^t|\Phi^l)+\log p(\boldsymbol{\pi})+\sum_i\log p(\boldsymbol{\mu}_i,\Lambda_i)$$

$$\pi_i^{l+1}=\frac{\alpha_i+N_i-1}{\sum_i\alpha_i+N-k}\qquad\boldsymbol{\mu}_i^{l+1}=\frac{\kappa_0\boldsymbol{\mu}_0+N_i\boldsymbol{m}_i}{\kappa_0+N_i}\qquad\Lambda_i^{l+1}=\left(\frac{\mathbf{V}_0^{-1}+\mathbf{C}_i+\mathbf{S}_i}{\nu_0+N_i+d+2}\right)^{-1}$$



Models in increasing complexity. A complex model can fit more datasets but is spread thin, a simple model can fit few datasets but has higher marginal likelihood where it does (MacKay 2003)





Nonparametric Bayes

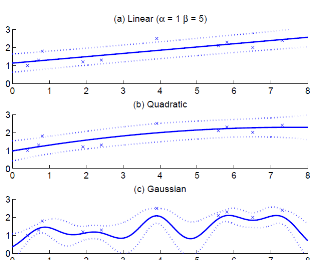
- Model complexity can increase with more data (in practice up to N, potentially to infinity)
- Similar to k-NN and Parzen windows we saw before where training set is the parameters

Gaussian Processes

- Nonparametric model for supervised learning
- Assume Gaussian prior $p(\mathbf{w})\sim N(0,1/\alpha)$
  $\mathbf{y}=\mathbf{X}\mathbf{w}$, where $E[\mathbf{y}]=0$ and $\text{Cov}(\mathbf{y})=\mathbf{K}$ with $\mathbf{K}_{ij}=(\mathbf{x}^i)^T\mathbf{x}^i$
  $\mathbf{K}$ is the covariance function, here linear
- With basis function $\phi(\mathbf{x})$, $\mathbf{K}_{ij}=(\phi(\mathbf{x}^i))^T\phi(\mathbf{x}^i)$
  $r\sim N_N(\mathbf{0},\mathbf{C}_N)$ where $\mathbf{C}_N=(1/\beta)\mathbf{I}+\mathbf{K}$
- With new $\mathbf{x}'$ added as $\mathbf{x}_{N+1}$, $r_{N+1}\sim N_{N+1}(0,\mathbf{C}_{N+1})$

$$\mathbf{C}_{N+1}=\begin{bmatrix}\mathbf{C}_N & \mathbf{k}\\ \mathbf{k} & c\end{bmatrix}$$

where $\mathbf{k}=[K(\mathbf{x}',\mathbf{x}^t)_t]^T$ and $c=K(\mathbf{x}',\mathbf{x}')+1/\beta$.
$p(r'|\mathbf{x}',\mathbf{X},\mathbf{r})\sim N(\mathbf{k}^T\mathbf{C}_{N-1}\mathbf{r},c-\mathbf{k}^T\mathbf{C}_{N-1}\mathbf{k})$



Dirichlet Processes

- Nonparametric Bayesian approach for clustering
- Chinese restaurant process
- Customers arrive and either join one of the existing tables or start a new one, based on the table occupancies:

$$\begin{aligned}\text{Join existing table } i \text{ with } P(z_i=1) &= \frac{n_i}{\alpha+n-1}, i=1,\ldots,k\\ \text{Start new table with } P(z_{k+1}=1) &= \frac{\alpha}{\alpha+n-1}\end{aligned}$$

Nonparametric Gaussian Mixture

- Tables are Gaussian components and decisions based both on prior and also on input x:

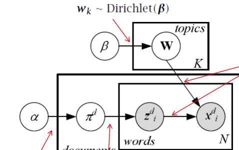
$$\begin{aligned}\text{Join component } i \text{ with } P(z_i^t=1) &\propto \frac{n_i}{\alpha+n-1}p(x^t|X_i), i=1,\ldots,k\\ \text{Start new component with } P(z_{k+1}^t) &\propto \frac{\alpha}{\alpha+n-1}p(x^t)\end{aligned}$$

Latent Dirichlet Allocation

- Bayesian feature extraction

$\mathbf{w}_k\sim\text{Dirichlet}(\boldsymbol{\beta})$

$x_i^d\sim\text{Mult}_M(\mathbf{w}_{z_i^d})$



$\boldsymbol{\pi}^d\sim\text{Dirichlet}_K(\boldsymbol{\alpha})$

$z_i^d\sim\text{Mult}_K(\boldsymbol{\pi}^d)$

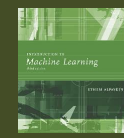
Beta Processes

- Nonparametric Bayesian approach for feature extraction
- Matrix factorization:

$$\mathbf{X} = \mathbf{Z}\mathbf{A} \qquad z_j^t = \begin{cases} 1 & \text{with probability } \mu_j \\ 0 & \text{with probability } 1-\mu_j \end{cases}$$

$$\mu_j \sim \text{beta}(\alpha, 1)$$

- Nonparametric version: Allow j to increase with more data probabilistically
- Indian buffet process: Customer can take one of the existing dishes with prob μ_j or add a new dish to the buffet



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 17:

COMBINING MULTIPLE LEARNERS

Rationale

- No Free Lunch Theorem: There is no algorithm that is always the most accurate
- Generate a group of base-learners which when combined has higher accuracy
- Different learners use different
  - Algorithms
  - Hyperparameters
  - Representations /Modalities/Views
  - Training sets
  - Subproblems
- Diversity vs accuracy

Voting

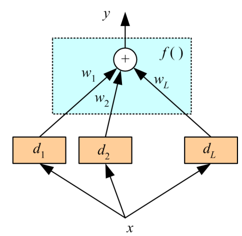
- Linear combination

$$y = \sum_{j=1}^{L} w_j d_j$$

$$w_j \geq 0 \text{ and } \sum_{j=1}^{L} w_j = 1$$

- Classification

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$



- Bayesian perspective:

$$P(C_i \mid x) = \sum_{\text{all models } \mathcal{M}_j} P(C_i \mid x, \mathcal{M}_j) P(\mathcal{M}_j)$$

- If d_j are iid

$$E[y] = E\left[\sum_j \frac{1}{L} d_j\right] = \frac{1}{L} L \cdot E[d_j] = E[d_j]$$

$$\text{Var}(y) = \text{Var}\left(\sum_j \frac{1}{L} d_j\right) = \frac{1}{L^2} \text{Var}\left(\sum_j d_j\right) = \frac{1}{L^2} L \cdot \text{Var}(d_j) = \frac{1}{L} \text{Var}(d_j)$$

Bias does not change, variance decreases by L

- If dependent, error increase with positive correlation

$$\text{Var}(y) = \frac{1}{L^2} \text{Var}\left(\sum_j d_j\right) = \frac{1}{L^2}\left[\sum_j \text{Var}(d_j) + 2 \sum_j \sum_{i<j} \text{Cov}(d_i, d_j)\right]$$

Fixed Combination Rules

Rule	Fusion function $f(\cdot)$
Sum	$y_i = \frac{1}{L} \sum_{j=1}^{L} d_{ji}$
Weighted sum	$y_i = \sum_j w_j d_{ji}, w_j \geq 0, \sum_j w_j = 1$
Median	$y_i = \text{median}_j d_{ji}$
Minimum	$y_i = \min_j d_{ji}$
Maximum	$y_i = \max_j d_{ji}$
Product	$y_i = \prod_j d_{ji}$

	C_1	C_2	C_3
d_1	0.2	0.5	0.3
d_2	0.0	0.6	0.4
d_3	0.4	0.4	0.2
Sum	0.2	**0.5**	0.3
Median	0.2	**0.5**	0.4
Minimum	0.0	**0.4**	0.2
Maximum	0.4	**0.6**	0.4
Product	0.0	**0.12**	0.032

Error-Correcting Output Codes

- K classes; L problems (Dietterich and Bakiri, 1995)
- Code matrix $\mathbf{W}$ codes classes in terms of learners
- One per class $L=K$

$$\mathbf{W} = \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{bmatrix}$$

- Pairwise $L=K(K-1)/2$

$$\mathbf{W} = \begin{bmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix}$$

- Full code $L=2^{(K-1)}-1$

$$\mathbf{W} = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & +1 & +1 & +1 & +1 \\ -1 & +1 & +1 & -1 & -1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{bmatrix}$$

- With reasonable L, find $\mathbf{W}$ such that the Hamming distance btw rows and columns are maximized.
- Voting scheme

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$

- Subproblems may be more difficult than one-per-K

Bagging

- Use bootstrapping to generate L training sets and train one base-learner with each (Breiman, 1996)
- Use voting (Average or median with regression)
- Unstable algorithms profit from bagging

AdaBoost

Generate a sequence of base-learners each focusing on previous one's errors

(Freund and Schapire, 1996)

Training:
- For all $\{x^t, r^t\}_{t=1}^N \in \mathcal{X}$, initialize $p_1^t = 1/N$
- For all base-learners $j = 1, \ldots, L$
  - Randomly draw $\mathcal{X}_j$ from $\mathcal{X}$ with probabilities p_j^t
  - Train d_j using $\mathcal{X}_j$
  - For each (x^t, r^t), calculate $y_j^t \leftarrow d_j(x^t)$
  - Calculate error rate: $\epsilon_j \leftarrow \sum_t p_j^t \cdot 1(y_j^t \neq r^t)$
  - If $\epsilon_j > 1/2$, then $L \leftarrow j-1$; stop
  - $\beta_j \leftarrow \epsilon_j/(1-\epsilon_j)$
  - For each (x^t, r^t), decrease probabilities if correct:
    - If $y_j^t = r^t$ $p_{j+1}^t \leftarrow \beta_j p_j^t$ Else $p_{j+1}^t \leftarrow p_j^t$
  - Normalize probabilities:
    - $Z_j \leftarrow \sum_t p_{j+1}^t; \quad p_{j+1}^t \leftarrow p_{j+1}^t / Z_j$

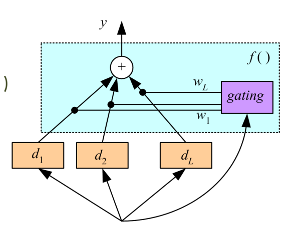
Testing:
- Given x, calculate $d_j(x), j = 1, \ldots, L$
- Calculate class outputs, $i = 1, \ldots, K$:
  - $y_i = \sum_{j=1}^{L} \left(\log \frac{1}{\beta_j}\right) d_{ji}(x)$

Mixture of Experts

Voting where weights are input-dependent (gating)

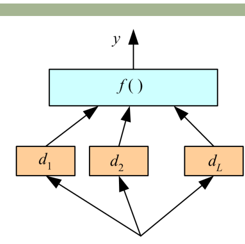
$$y = \sum_{j=1}^{L} w_j d_j$$

(Jacobs et al., 1991)
Experts or gating can be nonlinear



Stacking

- Combiner f () is another learner (Wolpert, 1992)



Fine-Tuning an Ensemble

- Given an ensemble of dependent classifiers, do not use it as is, try to get independence

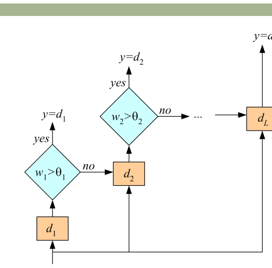
1. Subset selection: Forward (growing)/Backward (pruning) approaches to improve accuracy/diversity/independence
2. Train metaclassifiers: From the output of correlated classifiers, extract new combinations that are uncorrelated. Using PCA, we get "eigenlearners."

- Similar to feature selection vs feature extraction

Cascading

Use d_j only if preceding ones are not confident

Cascade learners in order of complexity

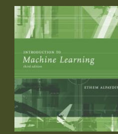


Combining Multiple Sources/Views

15

- Early integration: Concat all features and train a single learner
- Late integration: With each feature set, train one learner, then either use a fixed rule or stacking to combine decisions
- Intermediate integration: With each feature set, calculate a kernel, then use a single SVM with multiple kernels
- Combining features vs decisions vs kernels

i2ml3e-chap18.pdf



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

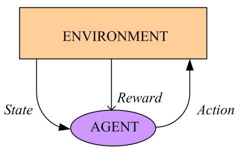
CHAPTER 18:

REINFORCEMENT LEARNING

Introduction

3

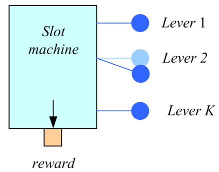
- Game-playing: Sequence of moves to win a game
- Robot in a maze: Sequence of actions to find a goal
- Agent has a state in an environment, takes an action and sometimes receives reward and the state changes
- Credit-assignment
- Learn a policy



Single State: K-armed Bandit

4

- Among K levers, choose the one that pays best
  - Q(a): value of action a
  - Reward is r_a
  - Set Q(a) = r_a
  - Choose a^* if
    - $Q(a^*)=\max_a Q(a)$



- Rewards stochastic (keep an *expected* reward):

$$Q_{t+1}(a)\leftarrow Q_t(a)+\eta[r_{t+1}(a)-Q_t(a)]$$

Elements of RL (Markov Decision Processes)

5

- s_t : State of agent at time t
- a_t: Action taken at time t
- In s_t, action a_t is taken, clock ticks and reward r_{t+1} is received and state changes to s_{t+1}
- Next state prob: $P(s_{t+1} \mid s_t, a_t)$
- Reward prob: $p(r_{t+1} \mid s_t, a_t)$
- Initial state(s), goal state(s)
- Episode (trial) of actions from initial state to goal
- (Sutton and Barto, 1998; Kaelbling et al., 1996)

Policy and Cumulative Reward

6

- Policy, $\pi : S \to \mathcal{A}$ $\quad a_t=\pi(s_t)$
- Value of a policy, $V^\pi(s_t)$
- Finite-horizon:

$$V^\pi(s_t)=E[r_{t+1}+r_{t+2}+\cdots+r_{t+T}]=E\left[\sum_{i=1}^{T} r_{t+i}\right]$$

- Infinite horizon:

$$V^\pi(s_t)=E[r_{t+1}+\gamma r_{t+2}+\gamma^2 r_{t+3}+\cdots]=E\left[\sum_{i=1}^{\infty}\gamma^{i-1} r_{t+i}\right]$$

$0\le\gamma<1$ is the discount rate

$$V^*(s_t)=\max_\pi V^\pi(s_t),\forall s_t$$

$$=\max_{a_t} E\left[\sum_{i=1}^{\infty}\gamma^{i-1} r_{t+i}\right]$$

$$=\max_{a_t} E\left[r_{t+1}+\gamma\sum_{i=1}^{\infty}\gamma^{i-1} r_{t+i+1}\right]$$

$$=\max_{a_t} E[r_{t+1}+\gamma V^*(s_{t+1})]$$ Bellman's equation

$$V^*(s_t)=\max_{a_t}\left(E[r_{t+1}]+\gamma\sum_{s_{t+1}}P(s_{t+1}\mid s_t,a_t)V^*(s_{t+1})\right)$$

$V^*(s_t)=\max_{a_t} Q^*(s_t,a_t)$ Value of a_t in s_t

$$Q^*(s_t,a_t)=E[r_{t+1}]+\gamma\sum_{s_{t+1}}P(s_{t+1}\mid s_t,a_t)\max_{a_{t+1}}Q^*(s_{t+1},a_{t+1})$$

7

Model-Based Learning

8

- Environment, $P(s_{t+1} \mid s_t, a_t)$, $p(r_{t+1} \mid s_t, a_t)$ known
- There is no need for exploration
- Can be solved using dynamic programming
- Solve for

$$V^*(s_t)=\max_{a_t}\left(E[r_{t+1}]+\gamma\sum_{s_{t+1}}P(s_{t+1}\mid s_t,a_t)V^*(s_{t+1})\right)$$

- Optimal policy

$$\pi^*(s_t)=\arg\max_{a_t}\left(E[r_{t+1}\mid s_t,a_t]+\gamma\sum_{s_{t+1}}P(s_{t+1}\mid s_t,a_t)V^*(s_{t+1})\right)$$

Value Iteration

9

```
Initialize V(s) to arbitrary values
Repeat
  For all s ∈ S
    For all a ∈ A
      Q(s,a) ← E[r|s,a] + γ Σ_{s'∈S} P(s'|s,a)V(s')
    V(s) ← max_a Q(s,a)
Until V(s) converge
```

Policy Iteration

10

```
Initialize a policy π arbitrarily
Repeat
  π ← π'
  Compute the values using π by
    solving the linear equations
      V^π(s) = E[r|s,π(s)] + γ Σ_{s'∈S} P(s'|s,π(s))V^π(s')
  Improve the policy at each state
    π'(s) ← arg max_a (E[r|s,a] + γ Σ_{s'∈S} P(s'|s,a)V^π(s'))
Until π = π'
```

Temporal Difference Learning

11

- Environment, $P(s_{t+1} \mid s_t, a_t)$, $p(r_{t+1} \mid s_t, a_t)$, is not known; model-free learning
- There is need for exploration to sample from $P(s_{t+1} \mid s_t, a_t)$ and $p(r_{t+1} \mid s_t, a_t)$
- Use the reward received in the next time step to update the value of current state (action)
- The temporal difference between the value of the current action and the value discounted from the next state

Exploration Strategies

12

- ε-greedy: With pr ε,choose one action at random uniformly; and choose the best action with pr 1-ε
- Probabilistic:

$$P(a\mid s)=\frac{\exp Q(s,a)}{\sum_{b=1}^{\mathcal{A}}\exp Q(s,b)}$$

- Move smoothly from exploration/exploitation.
- Decrease ε
- Annealing $\quad P(a\mid s)=\dfrac{\exp[Q(s,a)/T]}{\sum_{b=1}^{\mathcal{A}}\exp[Q(s,b)/T]}$

Deterministic Rewards and Actions

13

$$Q^*(s_t,a_t)=E[r_{t+1}]+\gamma\sum_{s_{t+1}}P(s_{t+1}\mid s_t,a_t)\max_{a_{t+1}}Q^*(s_{t+1},a_{t+1})$$

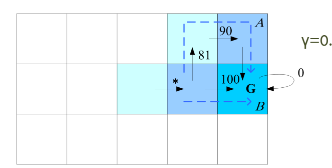
- Deterministic: single possible reward and next state

$$Q(s_t,a_t)=r_{t+1}+\gamma\max_{a_{t+1}}Q(s_{t+1},a_{t+1})$$

used as an update rule (backup)

$$\hat{Q}(s_t,a_t)\leftarrow r_{t+1}+\gamma\max_{a_{t+1}}\hat{Q}(s_{t+1},a_{t+1})$$

Starting at zero, Q values increase, never decrease



Consider the value of action marked by '*':

If path A is seen first, Q(*)=0.9*max(0,81)=73

Then B is seen, Q(*)=0.9*max(100,81)=90

Or,

If path B is seen first, Q(*)=0.9*max(100,0)=90

Then A is seen, Q(*)=0.9*max(100,81)=90

Q values increase but never decrease

14

Nondeterministic Rewards and Actions

15

- When next states and rewards are nondeterministic (there is an opponent or randomness in the environment), we keep averages (expected values) instead as assignments
- Q-learning (Watkins and Dayan, 1992):

$$\hat{Q}(s_t,a_t) \leftarrow \hat{Q}(s_t,a_t) + \eta\left(\boxed{r_{t+1} + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1},a_{t+1})} - \hat{Q}(s_t,a_t)\right)$$

- Off-policy vs on-policy (Sarsa)
backup
- Learning V (TD-learning: Sutton, 1988)

$$V(s_t) \leftarrow V(s_t) + \eta\left(\boxed{r_{t+1} + \gamma V(s_{t+1})} - V(s_t)\right)$$

Q-learning

16

```
Initialize all Q(s,a) arbitrarily
For all episodes
   Initalize s
   Repeat
      Choose a using policy derived from Q, e.g., ε-greedy
      Take action a, observe r and s'
      Update Q(s,a):
         Q(s,a) ← Q(s,a) + η(r + γ max_a' Q(s',a') − Q(s,a))
      s ← s'
   Until s is terminal state
```

Sarsa

17

```
Initialize all Q(s,a) arbitrarily
For all episodes
   Initalize s
   Choose a using policy derived from Q, e.g., ε-greedy
   Repeat
      Take action a, observe r and s'
      Choose a' using policy derived from Q, e.g., ε-greedy
      Update Q(s,a):
         Q(s,a) ← Q(s,a) + η(r + γQ(s',a') − Q(s,a))
      s ← s', a ← a'
   Until s is terminal state
```

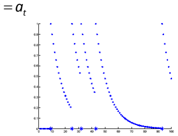
Eligibility Traces

18

Keep a record of previously visited states (actions)

$$e_t(s,a) = \begin{cases} 1 & \text{if } s = s_t \text{ and } a = a_t \\ \gamma\lambda e_{t-1}(s,a) & \text{otherwise} \end{cases}$$

$$\delta_t = r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)$$
$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \eta\delta_t e_t(s,a), \forall s,a$$



Sarsa (λ)

19

```
Initialize all Q(s,a) arbitrarily, e(s,a) ← 0, ∀s,a
For all episodes
   Initalize s
   Choose a using policy derived from Q, e.g., ε-greedy
   Repeat
      Take action a, observe r and s'
      Choose a' using policy derived from Q, e.g., ε-greedy
      δ ← r + γQ(s',a') − Q(s,a)
      e(s,a) ← 1
      For all s,a:
         Q(s,a) ← Q(s,a) + ηδe(s,a)
         e(s,a) ← γλe(s,a)
      s ← s', a ← a'
   Until s is terminal state
```

Generalization

20

- Tabular: Q (*s* , *a*) or *V* (*s*) stored in a table
- Regressor: Use a learner to estimate Q(*s*,*a*) or *V*(*s*)

$$E^t(\boldsymbol{\theta}) = \left[r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right]^2$$
$$\Delta\boldsymbol{\theta} = \eta\left[r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right]\nabla_{\boldsymbol{\theta}_t} Q(s_t,a_t)$$

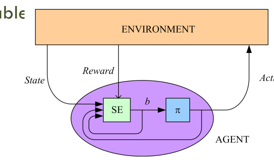
Eligibility

$$\Delta\boldsymbol{\theta} = \eta\delta_t \mathbf{e}_t$$
$$\delta_t = r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)$$
$$\mathbf{e}_t = \gamma\lambda\mathbf{e}_{t-1} + \nabla_{\theta_t} Q(s_t,a_t) \text{ with } \mathbf{e}_0 \text{ all zeros}$$

Partially Observable States

21

- The agent does not know its state but receives an observation $p(o_{t+1} \mid s_t, a_t)$ which can be used to infer a belief about states
- Partially observable MDP



The Tiger Problem

22

- Two doors, behind one of which there is a tiger
- *p*: prob that tiger is behind the left door

$r(A,Z)$	Tiger left	Tiger right
Open left	−100	+80
Open right	+90	−100

- $R(a_L) = -100p + 80(1-p)$, $R(a_R) = 90p - 100(1-p)$
- We can sense with a reward of $R(a_S) = -1$
- We have unreliable sensors

$$P(o_L \mid z_L) = 0.7 \qquad P(o_L \mid z_R) = 0.3$$
$$P(o_R \mid z_L) = 0.3 \qquad P(o_R \mid z_R) = 0.7$$

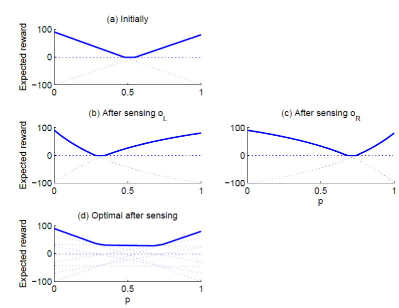
- If we sense o_L, *our belief in tiger's position changes*

$$p' = P(z_L \mid o_L) = \frac{P(o_L \mid z_L)P(z_L)}{P(o_L)} = \frac{0.7p}{0.7p + 0.3(1-p)}$$
$$R(a_L \mid o_L) = r(a_L, z_L)P(z_L \mid o_L) + r(a_L, z_R)P(z_R \mid o_L)$$
$$= -100p' + 80(1-p')$$
$$= -100\frac{0.7p}{P(o_L)} + 80\frac{0.3(1-p)}{P(o_L)}$$
$$R(a_R \mid o_L) = r(a_R, z_L)P(z_L \mid o_L) + r(a_R, z_R)P(z_R \mid o_L)$$
$$= 90p' - 100(1-p')$$
$$= 90\frac{0.7p}{P(o_L)} - 100\frac{0.3(1-p)}{P(o_L)}$$
$$R(a_S \mid o_L) = -1$$

23

$$V' = \sum_j \left[\max_i R(a_i \mid o_j)\right] P(o_j)$$
$$= \max(R(a_L \mid o_L), R(a_R \mid o_L), R(a_S \mid o_L))P(o_L) + \max(R(a_L \mid o_R), R(a_R \mid o_R), R(a_S \mid o_R))P(o_R)$$
$$= \max\begin{pmatrix} -100p & +80(1-p) \\ -43p & -46(1-p) \\ 33p & +26(1-p) \\ 90p & -100(1-p) \end{pmatrix}$$

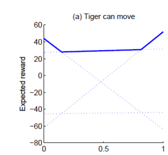
24



25

- Let us say the tiger can move from one room to the other with prob 0.8

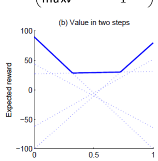
$$p' = 0.2p + 0.8(1-p)$$
$$V' = \max\begin{pmatrix} -100p' & +80(1-p') \\ 33p & +26(1-p') \\ 90p & -100(1-p') \end{pmatrix}$$



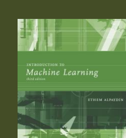
26

- When planning for episodes of two, we can take a_L, a_R, or sense and wait:

$$V_2 = \max\begin{pmatrix} -100p & +80(1-p) \\ 90p & -100(1-p) \\ \max V' & -1 \end{pmatrix}$$



27



Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 19:

DESIGN AND ANALYSIS OF MACHINE LEARNING EXPERIMENTS

Introduction

3

- Questions:
  - Assessment of the expected error of a learning algorithm: Is the error rate of 1-NN less than 2%?
  - Comparing the expected errors of two algorithms: Is *k*-NN more accurate than MLP ?
- Training/validation/test sets
- Resampling methods: K-fold cross-validation

Algorithm Preference

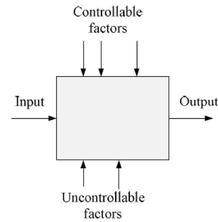
4

- Criteria (Application-dependent):
  - Misclassification error, or risk (loss functions)
  - Training time/space complexity
  - Testing time/space complexity
  - Interpretability
  - Easy programmability
- Cost-sensitive learning

Factors and Response

5

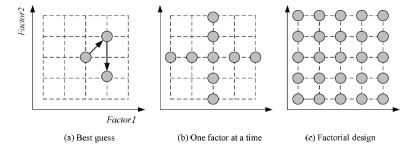
- Response function based on output to be maximized
- Depends on controllable factors
- Uncontrollable factors introduce randomness
- Find the configuration of controllable factors that maximizes response and *minimally affected by uncontrollable factors*



Strategies of Experimentation

6

How to search the factor space?



Response surface design for approximating and maximizing the response function in terms of the controllable factors

Guidelines for ML experiments

7

A. Aim of the study
B. Selection of the response variable
C. Choice of factors and levels
D. Choice of experimental design
E. Performing the experiment
F. Statistical Analysis of the Data
G. Conclusions and Recommendations

Resampling and K-Fold Cross-Validation

8

- The need for multiple training/validation sets $\{X_i,V_i\}_i$: Training/validation sets of fold *i*
- *K*-fold cross-validation: Divide X into *k*, X_i, $i=1,...,K$

$$\mathcal{V}_1 = \mathcal{X}_1 \quad \mathcal{T}_1 = \mathcal{X}_2 \cup \mathcal{X}_3 \cup \cdots \cup \mathcal{X}_K$$
$$\mathcal{V}_2 = \mathcal{X}_2 \quad \mathcal{T}_2 = \mathcal{X}_1 \cup \mathcal{X}_3 \cup \cdots \cup \mathcal{X}_K$$
$$\vdots$$
$$\mathcal{V}_K = \mathcal{X}_K \quad \mathcal{T}_K = \mathcal{X}_1 \cup \mathcal{X}_2 \cup \cdots \cup \mathcal{X}_{K-1}$$

- T_i share *K*-2 parts

5×2 Cross-Validation

9

- 5 times 2 fold cross-validation (Dietterich, 1998)

$$\boxed{\begin{matrix}\mathcal{T}_1 = \mathcal{X}_1^{(1)} & \mathcal{V}_1 = \mathcal{X}_1^{(2)} \\ \mathcal{T}_2 = \mathcal{X}_1^{(2)} & \mathcal{V}_2 = \mathcal{X}_1^{(1)}\end{matrix}}$$
$$\boxed{\begin{matrix}\mathcal{T}_3 = \mathcal{X}_2^{(1)} & \mathcal{V}_3 = \mathcal{X}_2^{(2)} \\ \mathcal{T}_4 = \mathcal{X}_2^{(2)} & \mathcal{V}_4 = \mathcal{X}_2^{(1)}\end{matrix}}$$
$$\vdots$$
$$\boxed{\begin{matrix}\mathcal{T}_9 = \mathcal{X}_5^{(1)} & \mathcal{V}_9 = \mathcal{X}_5^{(2)} \\ \mathcal{T}_{10} = \mathcal{X}_5^{(2)} & \mathcal{V}_{10} = \mathcal{X}_5^{(1)}\end{matrix}}$$

Bootstrapping

10

- Draw instances from a dataset *with replacement*
- Prob that we do not pick an instance after N draws

$$\left(1-\frac{1}{N}\right)^N \approx e^{-1} = 0.368$$

that is, only 36.8% is new!

Performance Measures

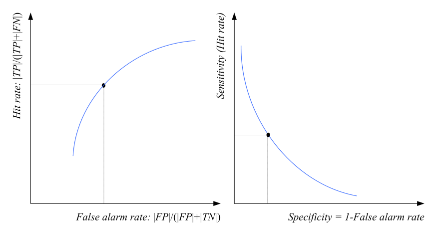
11

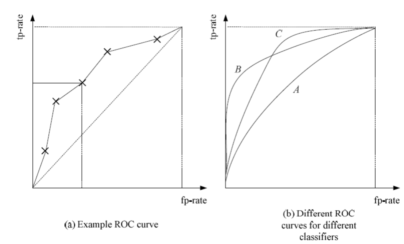
	Predicted class	
True Class	Yes	No
Yes	TP: True Positive	FN: False Negative
No	FP: False Positive	TN: True Negative

- Error rate = # of errors / # of instances = (FN+FP) / N
- Recall = # of found positives / # of positives = TP / (TP+FN) = sensitivity = hit rate
- Precision = # of found positives / # of found = TP / (TP+FP)
- Specificity = TN / (TN+FP)
- False alarm rate = FP / (FP+TN) = 1 - Specificity

ROC Curve

12

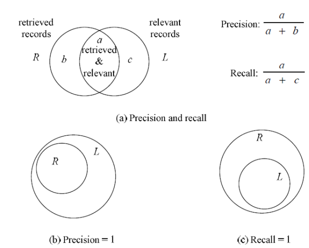




13

Precision and Recall

14



Interval Estimation

15

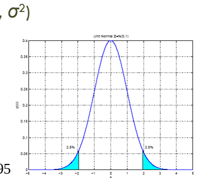
- $X = \{ x^t \}_t$ where $x^t \sim N(\mu, \sigma^2)$
- $m \sim N(\mu, \sigma^2/N)$

$$\sqrt{N}\frac{(m-\mu)}{\sigma} \sim Z$$

$$P\left\{-1.96 < \sqrt{N}\frac{(m-\mu)}{\sigma} < 1.96\right\} = 0.95$$

$$P\left\{m - 1.96\frac{\sigma}{\sqrt{N}} < \mu < m + 1.96\frac{\sigma}{\sqrt{N}}\right\} = 0.95$$

$$P\left\{m - z_{\alpha/2}\frac{\sigma}{\sqrt{N}} < \mu < m + z_{\alpha/2}\frac{\sigma}{\sqrt{N}}\right\} = 1-\alpha$$



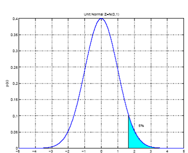
100(1- α) percent confidence interval

$$P\left\{\sqrt{N}\frac{(m-\mu)}{\sigma} < 1.64\right\} = 0.95$$

$$P\left\{m - 1.64\frac{\sigma}{\sqrt{N}} < \mu\right\} = 0.95$$

$$P\left\{m - z_{\alpha}\frac{\sigma}{\sqrt{N}} < \mu\right\} = 1-\alpha$$

100(1- α) percent one-sided confidence interval



When σ^2 is not known:

$$S^2 = \sum_t (x^t - m)^2/(N-1) \qquad \frac{\sqrt{N}(m-\mu)}{S} \sim t_{N-1}$$

$$P\left\{m - t_{\alpha/2,N-1}\frac{S}{\sqrt{N}} < \mu < m + t_{\alpha/2,N-1}\frac{S}{\sqrt{N}}\right\} = 1-\alpha$$

16

Hypothesis Testing

17

- Reject a null hypothesis if not supported by the sample with enough confidence

$X = \{ x^t \}_t$ where $x^t \sim N(\mu, \sigma^2)$

H_0: $\mu = \mu_0$ vs. H_1: $\mu \neq \mu_0$

Accept H_0 with level of significance α if μ_0 is in the 100(1- α) confidence interval

$$\frac{\sqrt{N}(m-\mu_0)}{\sigma} \in (-z_{\alpha/2}, z_{\alpha/2})$$

Two-sided test

	Decision	
Truth	Accept	Reject
True	Correct	Type I error
False	Type II error	Correct (Power)

- One-sided test: H_0: $\mu \leq \mu_0$ vs. H_1: $\mu > \mu_0$

Accept if $\frac{\sqrt{N}(m-\mu_0)}{\sigma} \in (-\infty, z_\alpha)$

- Variance unknown: Use *t*, instead of *z*

Accept H_0: $\mu = \mu_0$ if

$$\frac{\sqrt{N}(m-\mu_0)}{S} \in (-t_{\alpha/2,N-1}, t_{\alpha/2,N-1})$$

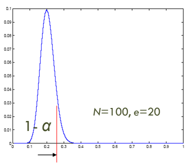
18

Assessing Error: $H_0: p \le p_0$ vs. $H_1: p > p_0$

19

- Single training/validation set: Binomial Test

  If error prob is p_0, prob that there are e errors or less in N validation trials is



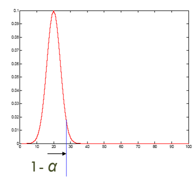
$$P\{X \le e\} = \sum_{j=1}^{e} \binom{N}{j} p_0^{\,j} \left(1 - p_0^{\,j}\right)^{N-j}$$

Accept if this prob is less than 1- α

Normal Approximation to the Binomial

20

- Number of errors X is approx N with mean Np_0 and var $Np_0(1-p_0)$



$$\frac{X - Np_0}{\sqrt{Np_0(1-p_0)}} \sim Z$$

Accept if this prob for $X = e$ is less than $z_{1-\alpha}$

Paired t Test

21

- Multiple training/validation sets
- $x^t_i = 1$ if instance t misclassified on fold i
- Error rate of fold i:

$$p_i = \frac{\sum_{t=1}^{N} x^t_i}{N}$$

- With m and s^2 average and var of p_i, we accept p_0 or less error if

$$\frac{\sqrt{K}(m - p_0)}{S} \sim t_{K-1}$$

  is less than $t_{\alpha,K-1}$

Comparing Classifiers: $H_0: \mu_0 = \mu_1$ vs. $H_1: \mu_0 \neq \mu_1$

22

- Single training/validation set: McNemar's Test

e_{00}: Number of examples misclassified by both	e_{01}: Number of examples misclassified by 1 but not 2
e_{10}: Number of examples misclassified by 2 but not 1	e_{11}: Number of examples correctly classified by both

- Under H_0, we expect $e_{01} = e_{10} = (e_{01} + e_{10})/2$

$$\frac{\left(|e_{01} - e_{10}| - 1\right)^2}{e_{01} + e_{10}} \sim \mathcal{X}^2_1$$

Accept if $< \mathcal{X}^2_{\alpha,1}$

K-Fold CV Paired t Test

23

- Use K-fold cv to get K training/validation folds
- p_i^1, p_i^2: Errors of classifiers 1 and 2 on fold i

  $p_i = p_i^1 - p_i^2$: Paired difference on fold i
- The null hypothesis is whether p_i has mean 0

$$H_0: \mu = 0 \text{ vs. } H_0: \mu \neq 0$$

$$m = \frac{\sum_{i=1}^{K} p_i}{K} \qquad s^2 = \frac{\sum_{i=1}^{K}(p_i - m)^2}{K-1}$$

$$\frac{\sqrt{K}(m-0)}{s} = \frac{\sqrt{K} \cdot m}{s} \sim t_{K-1} \text{ Accept if in } \left(-t_{\alpha/2,K-1}, t_{\alpha/2,K-1}\right)$$

5×2 cv Paired t Test

24

- Use 5×2 cv to get 2 folds of 5 tra/val replications (Dietterich, 1998)
- $p_i^{(j)}$: difference btw errors of 1 and 2 on fold j=1, 2 of replication i=1,...,5

$$\bar{p}_i = \left(p_i^{(1)} + p_i^{(2)}\right)/2 \qquad s_i^2 = \left(p_i^{(1)} - \bar{p}_i\right)^2 + \left(p_i^{(2)} - \bar{p}_i\right)^2$$

$$\frac{p_1^{(1)}}{\sqrt{\sum_{i=1}^{5} s_i^2/5}} \sim t_5$$

Two-sided test: Accept H_0: $\mu_0 = \mu_1$ if in $(-t_{\alpha/2,5}, t_{\alpha/2,5})$

One-sided test: Accept H_0: $\mu_0 \le \mu_1$ if $< t_{\alpha,5}$

5×2 cv Paired F Test

25

$$\frac{\sum_{i=1}^{5}\sum_{j=1}^{2}\left(p_i^{(j)}\right)^2}{2\sum_{i=1}^{5} s_i^2} \sim F_{10,5}$$

Two-sided test: Accept H_0: $\mu_0 = \mu_1$ if $< F_{\alpha,10,5}$

Comparing L>2 Algorithms: Analysis of Variance (Anova)

26

$$H_0: \mu_1 = \mu_2 = \cdots = \mu_L$$

- Errors of L algorithms on K folds

$$X_{ij} \sim \mathcal{N}(\mu_j, \sigma^2),\ j = 1,\ldots,L,\ i = 1,\ldots,K$$

- We construct two estimators to σ^2 .

  One is valid if H_0 is true, the other is always valid.

  We reject H_0 if the two estimators disagree.

27

If H_0 is true:

$$m_j = \sum_{i=1}^{K} \frac{X_{ij}}{K} \sim \mathcal{N}(\mu, \sigma^2/K)$$

$$m = \frac{\sum_{j=1}^{L} m_j}{L} \qquad S^2 = \frac{\sum_j (m_j - m)^2}{L-1}$$

Thus an estimator of σ^2 is $K \cdot S^2$, namely,

$$\hat{\sigma}^2 = K \sum_{j=1}^{L} \frac{(m_j - m)^2}{L-1}$$

$$\sum_j \frac{(m_j - m)^2}{\sigma^2/K} \sim \mathcal{X}^2_{L-1} \qquad SSb \equiv K \sum_j (m_j - m)^2$$

So when H_0 is true, we have

$$\frac{SSb}{\sigma^2} \sim \mathcal{X}^2_{L-1}$$

28

Regardless of H_0 our second estimator to σ^2 is the average of group variances S_j^2 :

$$S_j^2 = \frac{\sum_{i=1}^{K}(X_{ij} - m_j)^2}{K-1} \qquad \hat{\sigma}^2 = \sum_{j=1}^{L} \frac{S_j^2}{L} = \sum_j \sum_i \frac{(X_{ij} - m_j)^2}{L(K-1)}$$

$$SSw \equiv \sum_j \sum_i (X_{ij} - m_j)^2$$

$$(K-1)\frac{S_j^2}{\sigma^2} \sim \mathcal{X}^2_{K-1} \qquad \frac{SSw}{\sigma^2} \sim \mathcal{X}^2_{L(K-1)}$$

$$\left(\frac{SSb/\sigma^2}{L-1}\right) \Big/ \left(\frac{SSw/\sigma^2}{L(K-1)}\right) = \frac{SSb/(L-1)}{SSw/(L(K-1))} \sim F_{L-1,L(K-1)}$$

$$H_0: \mu_1 = \mu_2 = \cdots = \mu_L \text{ if } < F_{\alpha,L-1,L(K-1)}$$

ANOVA table

29

Source of variation	Sum of squares	Degrees of freedom	Mean square	F_0
Between groups	$SS_b \equiv K\sum_j (m_j - m)^2$	$L-1$	$MS_b = \frac{SS_b}{L-1}$	$\frac{MS_b}{MS_w}$
Within groups	$SS_w \equiv \sum_j \sum_i (X_{ij} - m_j)^2$	$L(K-1)$	$MS_w = \frac{SS_w}{L(K-1)}$	
Total	$SS_T \equiv \sum_j \sum_i (X_{ij} - m)^2$	$L \cdot K - 1$		

If ANOVA rejects, we do pairwise posthoc tests

$$H_0: \mu_i = \mu_j \text{ vs } H_1: \mu_i \neq \mu_j$$

$$t = \frac{m_i - m_j}{\sqrt{2}\sigma_w} \sim t_{L(K-1)}$$

Comparison over Multiple Datasets

30

- Comparing two algorithms:

  Sign test: Count how many times A beats B over N datasets, and check if this could have been by chance if A and B did have the same error rate
- Comparing multiple algorithms

  Kruskal-Wallis test: Calculate the average rank of all algorithms on N datasets, and check if these could have been by chance if they all had equal error

  If KW rejects, we do pairwise posthoc tests to find which ones have significant rank difference

Multivariate Tests

31

- Instead of testing using a single performance measure, e.g., error, use multiple measures for better discrimination, e.g., [fp-rate,fn-rate]
- Compare p-dimensional distributions
- Parametric case: Assume p-variate Gaussians

$$H_0: \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 \text{ vs. } H_1: \boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2$$

Multivariate Pairwise Comparison

32

- Paired differences: $\boldsymbol{d}_i = \boldsymbol{x}_{1i} - \boldsymbol{x}_{2i}$

$$H_0: \boldsymbol{\mu}_d = \mathbf{0} \text{ vs. } H_1: \boldsymbol{\mu}_d \neq \mathbf{0}$$

- Hotelling's multivariate T^2 test

$$T'^2 = K\boldsymbol{m}^T \mathbf{S}^{-1} \boldsymbol{m}$$

- For p=1, reduces to paired t test

Multivariate ANOVA

33

- Comparsion of L>2 algorithms

$$H_0 : \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \cdots = \boldsymbol{\mu}_L \text{ vs.}$$
$$H_1 : \boldsymbol{\mu}_r \neq \boldsymbol{\mu}_s \text{ for at least one pair } r, s$$

$$\mathbf{H} = K \sum_{j=1}^{L} (\boldsymbol{m}_j - \boldsymbol{m})(\boldsymbol{m}_j - \boldsymbol{m})^T$$

$$\mathbf{E} = \sum_{j=1}^{L} \sum_{i=1}^{K} (\boldsymbol{x}_{ij} - \boldsymbol{m}_j)(\boldsymbol{x}_{ij} - \boldsymbol{m}_j)^T$$

$$\Lambda' = \frac{|\mathbf{E}|}{|\mathbf{E} + \mathbf{H}|}$$

is Wilks's Λ distributed with $p, L(K-1), L-1$ degrees of freedom