Basics

Generalization

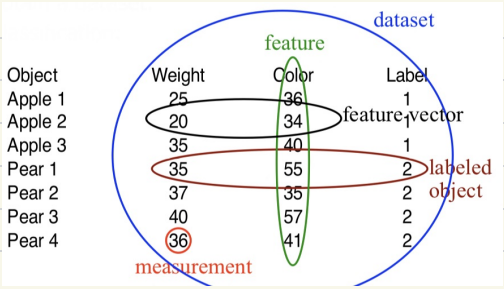
training/test set

Features

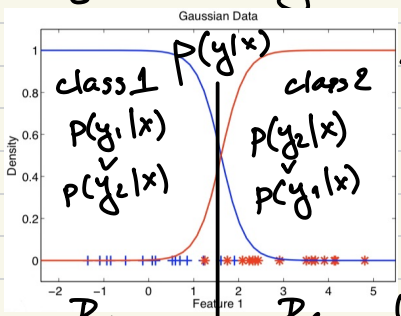
measurable data encoded in vector $x = [x_1, x_2, \dots, x_n]^T$

Dataset

features × label table



$p(x,y)$ - probability density over feature space



$\sum_i p(y_i|x) = 1$

decision boundary $p(y_1|x) = p(y_2|x)$

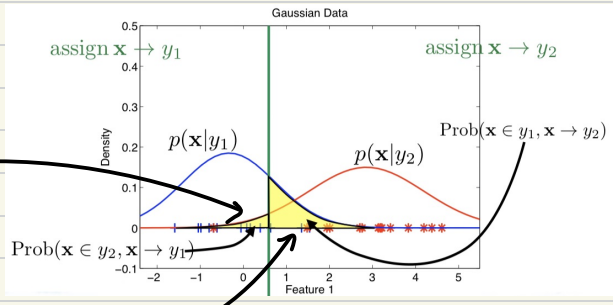
Bayes Theorem

posterior prob. (conditional) class distribution

$$p(y|x) = \frac{p(x|y)\,p(y)}{p(x)}$$

(unc.) data distribution, class prior

Error of decision boundary



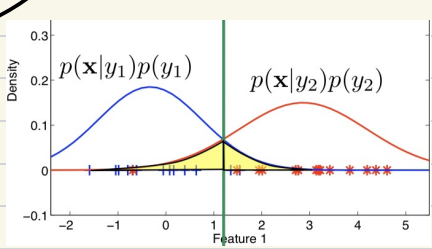
type I

$$\varepsilon_1 = \int_{R_2} p(x|y_1)\,dx$$

type II

$$\varepsilon_2 = \int_{R_1} p(x|y_2)\,dx$$

$$p(\text{error}) = \sum_{i=1}^{2} p(\text{error}|y_i)\,p(y_i)$$



Bayes (minimum) error ε^* (us. > 0)

Misclassification

mistake (true) y_j for y_i -> λ_{ij}

dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$

estimated label $\hat{y}_i$

empirical risk $R = \frac{1}{N} \sum_{i=1}^{N} \lambda_{y_i \hat{y}_i}$

conditional risk $l^i(x) = \sum_{j=1}^{C} \lambda_{ji} p(y_j|x)$

(assign x to class y_i)

average risk over region $r^i = \int_{R_i} l^i(x) p(x) dx$

overall risk $r = \sum_{i=1}^{C} r^i = \sum_{i=1}^{C} \int_{R_i} \sum_{j=1}^{C} \lambda_{ji} p(y_j|x) p(x) dx$

$x \in R_i$ if $\sum_{j=1}^{C} \lambda_{ji} p(y_j|x) \leq \sum_{j=1}^{C} \lambda_{jk} p(y_j|x)$ $k = 1, \ldots, C$ $\lambda_{y_i, y_j} = 0$

two-class problem $i=1$ $\lambda_{21} p(y_2|x)$ $i=2$ $\lambda_{12} p(y_1|x)$

4. Parametric Density-based Classifiers

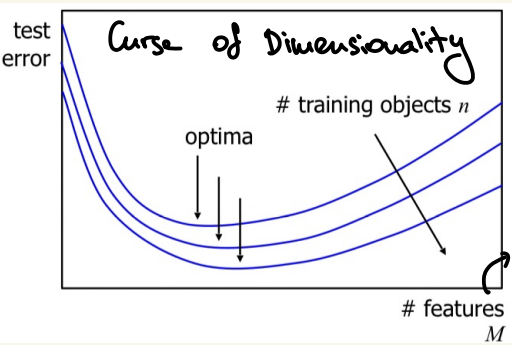
Bayes: $p(y_1|x) = \frac{p(x|y_1)p(y_1)}{p(x)} \overset{?}{=} \frac{p(x|y_2)p(y_2)}{p(x)} = p(y_2|x)$

Model - approximate $p(y|x)$ or $p(x|y), p(y)$

parametric vs. non-parametric

Discriminative vs. Generative

$\hat{p}(y|x)$ → directly approximate $p(y|x) \propto p(x|y)p(y)$



Gaussian Distribution

$\mu = 0, \sigma^2 = 1$, $[\mu - 2\sigma, \mu + 2\sigma] = 95\%$ data

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

$\sigma^2 = \Sigma^1$ (covariance matrix)

$p(x) = \frac{1}{\sqrt{(2\pi)^M \det(\Sigma)}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$

$= \mathcal{N}(x|\mu, \Sigma)$

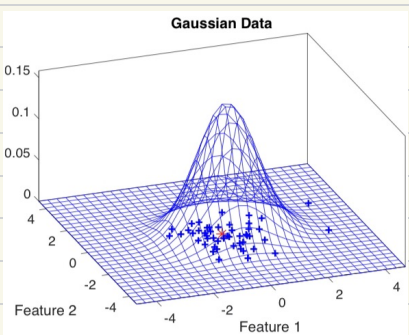
2D:

$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$

$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$

$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$

$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} (x_i - \hat{\mu})(x_i - \hat{\mu})^T$



Quadratic Classifier

$g(x) = \log p(y_1|x) - \log p(y_2|x) = x^T W x + w^T x + w_0$

$\log(\hat{p}(y_i|x)) = -\frac{M}{2}\log(2\pi) - \frac{1}{2}\log(\det\Sigma_i) - \frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i) + \log p(y_i) - \log p(x)$

$W = \frac{1}{2}\left(\Sigma_2^{-1} - \Sigma_1^{-1}\right)$

$w^T = \mu_1^T\Sigma_1^{-1} - \mu_2^T\Sigma_2^{-1}$

$w_0 = -\frac{1}{2}\log\det\Sigma_1 - \frac{1}{2}\mu_1^T\Sigma_1^{-1}\mu_1 + \log p(y_1)$

$+ \frac{1}{2}\log\det\Sigma_2 + \frac{1}{2}\mu_2^T\Sigma_2^{-1}\mu_2 - \log p(y_2)$

$\hat{\Sigma} = \frac{1}{C}\sum_{k=1}^{C} \hat{\Sigma}_k$ ← assume shared covariance

$g_i(x) = -\frac{1}{2}\log(\det\hat{\Sigma}) - \frac{1}{2}(x-\mu_i)^T \hat{\Sigma}^{-1}(x-\mu_i) + \log p(y_i)$

$\hookrightarrow g_i(x) = -\frac{1}{2}\mu_i^T \hat{\Sigma}^{-1}\mu_i + \mu_i^T \hat{\Sigma}^{-1}x + \log p(y_i)$

LDA

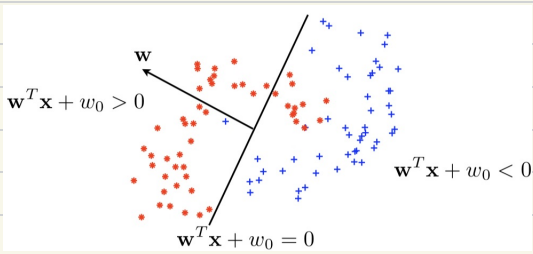
$g(x) = w^T x + w_0$ linear normal-based classifier

$w = \hat{\Sigma}^{-1}(\hat{\mu}_1 - \hat{\mu}_2)$

$w_0 = \frac{1}{2}\mu_2^T \hat{\Sigma}^{-1}\hat{\mu}_2 - \frac{1}{2}\hat{\mu}_1^T \hat{\Sigma}^{-1}\hat{\mu}_1 + \log p(y_1) - \log p(y_2)$

$x \to y_1$ if $w^T x + w_0 \geq 0$

$\hookrightarrow y_2$ if $w^T x + w_0 < 0$



Nearest Mean

$\hat{\Sigma} = \sigma^2 I$ meaning $\begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$

$g_i(x) = -\frac{1}{\sigma^2}\left(\frac{1}{2}\mu_i^T\mu_i - \mu_i^T x\right) + \log(p(y_i))$

$w = \hat{\mu}_1 - \hat{\mu}_2$

$w_0 = \frac{1}{2}\hat{\mu}_2^T\hat{\mu}_2 - \frac{1}{2}\hat{\mu}_1^T\hat{\mu}_1 - \sigma^2 \log\frac{p(y_1)}{p(y_2)}$

14. Non-parametric Density-based Classifiers

Histogram density estimation $\hat{p}(x) = \frac{1}{w}\frac{k_B}{N}$ — k_B → objects in bin; N → # of objects; w ← width of bin

Parzen (window) / Kernel Density Estimation

Parzen probability density $\hat{p}(x) = \frac{1}{w}\sum_{i=1}^{w} K\left(\frac{x - x_i}{w}\right)$

$p(x|y_i) = \frac{1}{w_i}\sum_{j=1}^{w_i} N(x|x_j^{(i)}, w\Sigma)$ ← Gaussian kernel, Σ as covariance

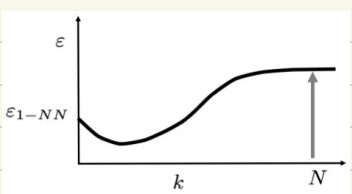
K-Nearest Neighbours

find k nearest neighbours to point

$\hat{p}(x|y_i) = \frac{k_i}{w_i V_k}$ → volume of sphere of k nearest neighbours

$\hat{p}(y_i) = \frac{w_i}{w}$

$\Rightarrow \hat{p}(x|y_i)\hat{p}(y_i) > \hat{p}(x|y_j)\hat{p}(y_j) \equiv k_i > k_j$



$k \xrightarrow{\text{large}}$ classify as most probable class

small → unstable decision boundaries

ties → odd k (solution for 2-class)
- → random
- → greater prior
- → nearest (1-nn)

Distance

euclidean $D(x, x') = \sqrt{\sum_d |x_d - x'_d|^2}$

manhattan $D(x, x') = \sum_d |x_d - x'_d|$

hamming $D(x, x') = \sum_d \mathbb{1}_{x_d \neq x'_d}$

Naive Bayes

$p(x_1, x_2, \dots, x_d | y)$

assumption: independent features

conditional independence: $p(B, S|H) = p(B|H)\,p(S|H)$

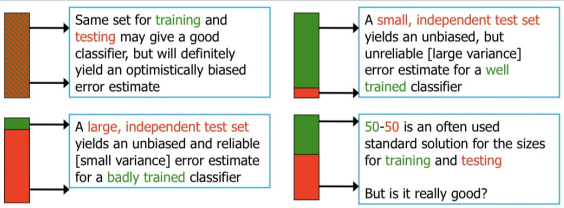
$\Rightarrow p(x|y) = \prod_{i=1}^{d} p(x_i|y)$

$p(x|y) \neq 0$ instead $\frac{num(x,y) + \varepsilon \rightarrow error}{num(y) + K\varepsilon \rightarrow \#\ classes}$

Classifier Evaluation

$\hat{\varepsilon} = \frac{1}{N}\sum_{i=1}^{N} z_i \qquad z_i = \begin{cases} 0 & \text{if } x_i \text{ correctly classified} \\ 1 & \text{if } x_i \text{ incorrectly classified} \end{cases}$

$\sigma_{\hat{\varepsilon}}^2 = Var(\hat{\varepsilon} \mid \text{test set size } N) = \frac{\varepsilon(1-\varepsilon)}{N} \qquad \sigma_{\hat{\varepsilon}} = \sqrt{\frac{\varepsilon(1-\varepsilon)}{N}}$



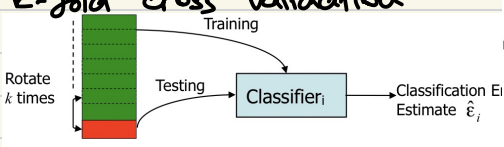
Bootstraping

randomly draw N/N objects with replacement

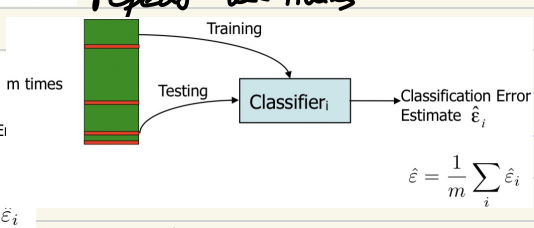
left-over -> testing

repeat m times

k-fold cross validation

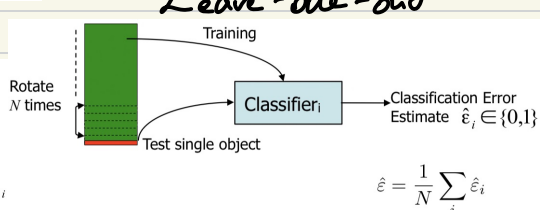


$$\hat{\varepsilon} = \frac{1}{k}\sum_i \hat{\varepsilon}_i$$



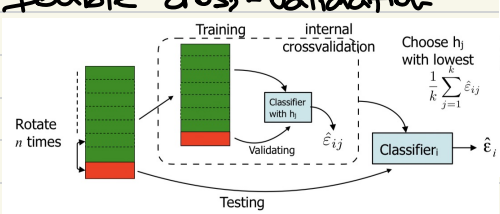
$$\hat{\varepsilon} = \frac{1}{m}\sum_i \hat{\varepsilon}_i$$

Leave-one-out

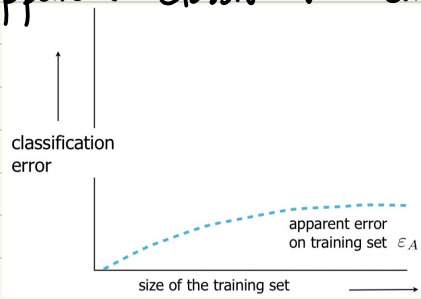


$$\hat{\varepsilon} = \frac{1}{N}\sum_i \hat{\varepsilon}_i$$

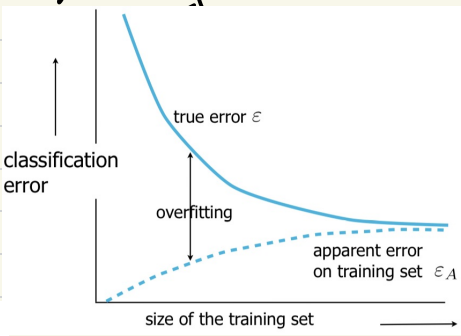
Double cross-validation

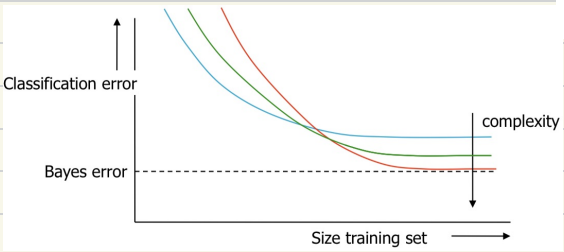


Apparent Classification error



Learning Curve





Larger training set → better classifier
independent test set → unbiased error estimates
Larger test set → more accurate error estimates
more complex classifiers } larger training set
larger feature set size

Squared error: $L(\omega) = E[\|g(x) - y\|^2]$

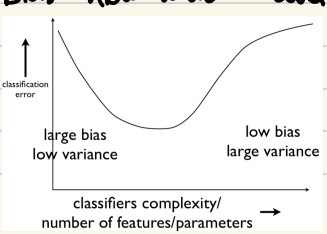
$D = \{(y_i, x_i);\ i = 1, \dots, N\}$

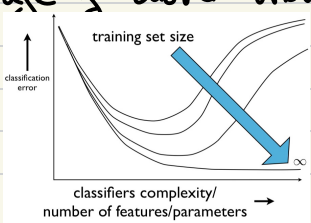
$g(x; D)$ - classifier

$$MSE = E_D[(g(x;D) - E_D[g(x;D)])^2] + E_D[(E_D[g(x;D)] - E[y|x])^2]$$

variance — bias^2

variance - how much g vary over different training sets
bias - how much average g differ from true output





2-class classification

standard error $\varepsilon = \varepsilon_1 p(y_1) + \varepsilon_2 p(y_2)$

weighted error $\varepsilon = \lambda_{12} \varepsilon_1 p(y_1) + \lambda_{21} \varepsilon_2 p(y_2)$

F1-score (harmonic mean) $F_1 = \frac{2 \cdot \text{precission} \cdot \text{recall}}{\text{precission} + \text{recall}}$

error - probability of erroneous classifications

performance / accuracy = 1 - error

sensitivity of class - performance of objects from class

specificity - performance for all objects outside class

precision of class - fraction of correctly assigned to class objects

recall - fraction of correctly classified objects

true positive rate - sensitivity

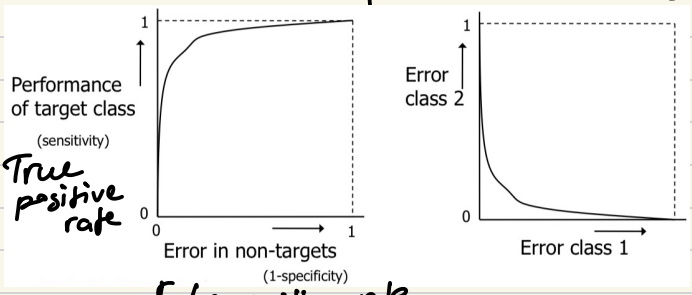
false positive rate - error for all objects outside class

Confusion Matrix

$\Lambda = \begin{bmatrix} \lambda_1 \\ \dots \\ \lambda_N \end{bmatrix}$ real labels $\quad L = \begin{bmatrix} l_1 \\ \dots \\ l_N \end{bmatrix}$ predicted labels

$C = \begin{bmatrix} c_{11} & \dots & c_{1K} \\ \dots & \dots & \dots \\ c_{K1} & \dots & c_{KK} \end{bmatrix} \qquad c_{ij} = N \cdot P[l_j | \lambda_i]$

ROC (Receiver-Operator Characteristic) Analysis



Parzen / Kernel density estimation

-> doesn't assume known distribution
-> estimates probability densities using kernel function
-> uses kernel function of fixed shape and width
-> width matters

K-Nearest Neighbour

Pros

-> simple and flexible
-> good performance
-> simple to adapt complexity

Cons

-> large training sets (stored)
-> compute all distances
-> sensibly scaled features
-> optimized k

Naive Bayes

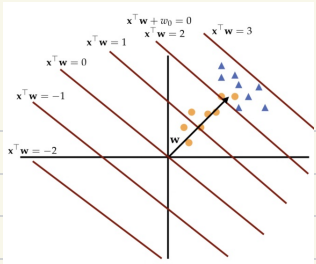
Pros

-> handle high dimensional feature spaces
-> fast training time
-> handle continuous and discrete data

Cons

-> can't deal with correlated features

4. Linear Discriminative Classifiers



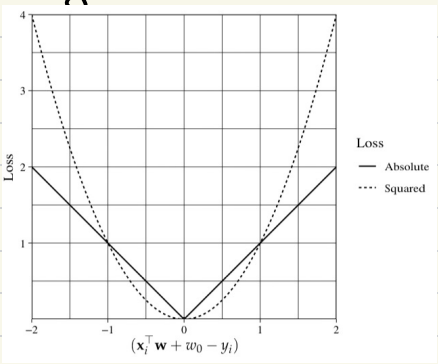
$L(h(x), y)$ → loss/cost function
↳ classifier

$\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} L(h(x_i), y_i)$

$h(x) = x^T w + w_0$
weight vector, bias/intercept

$y = \begin{cases} c_1 & \text{if } h(x) > 0 \\ c_0 & \text{if } h(x) \leq 0 \end{cases}$

decision boundary $h(x) = 0 = w^T x + w_0$



$\min_{w, w_0 \in \mathbb{R}^{D+1}} \frac{1}{N} \sum_{i=1}^{N} L(x_i^T w + w_0, y_i)$

L: absolute $|x_i^T w + w_0 - y_i|$
squared $(x_i^T w + w_0 - y_i)^2$

$\min_{w, w_0 \in \mathbb{R}^{D+1}} \frac{1}{N} \sum_{i=1}^{N} (x_i^T w + w_0 - y_i)^2$

$= \min_{w, w_0} J(w, w_0)$

$w_j^{t+1} = w_j^t - \alpha \left. \frac{\partial J(w, w_0)}{\partial w_j} \right|_{w, w_0 = w^t, w_0^t}$

step size

$\frac{\partial J(w, w_0)}{\partial w_j} = \frac{1}{N} \sum_{i=1}^{N} 2(x_i^T w + w_0 - y_i) x_i^{(j)}$

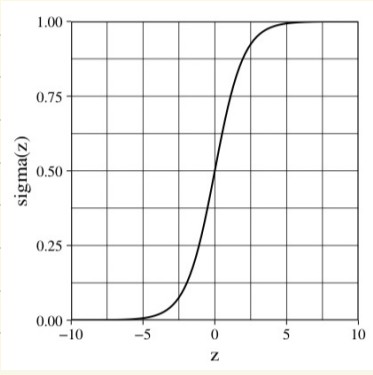
$\frac{\partial J(w, w_0)}{\partial w_0} = \frac{1}{N} \sum_{i=1}^{N} 2(w_0 + x_i^T w - y_i)$

$h(x) \approx P(y = 1 \mid x)$

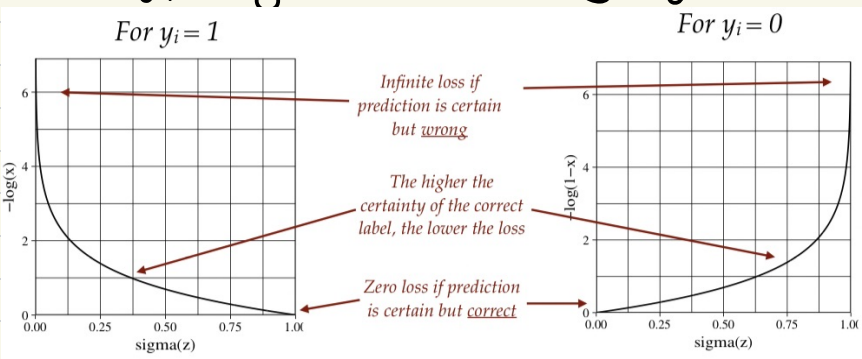
logistic function: $\sigma(z) = \frac{1}{1 + e^{-z}}$

label y_i for i = $\begin{cases} \sigma(h(x_i)), & y_i = 1 \\ 1 - \sigma(h(x_i)), & y_i = 0 \end{cases}$

$L(h) = \prod_{i=1}^{N} \sigma(h(x_i))^{y_i} (1 - \sigma(h(x_i)))^{1 - y_i}$



$$J(h) = -\sum_{i=1}^{N} y_i \log\left(\sigma(h(x_i))\right) + (1-y_i)\log\left(1-\sigma(h(x_i))\right)$$



$$\min_{w,w_0 \in \mathbb{R}^{D+1}} -\sum_{i=1}^{N} y_i \log\left(\sigma(x_i^T w + w_0)\right) + (1-y_i)\log\left(1-\sigma(x_i^T w + w_0)\right)$$

$$w_j^{new} = w_j - \alpha \frac{\partial J(w, w_0)}{\partial w_j} \qquad \frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1-\sigma(z))$$

$$\nabla_w J(w) = -\sum_{i=1}^{N} (y_i - \sigma(x_i^T w)) x_i$$

Support Vector Machines

$x_i^T w + w_0 \geq M \quad y_i = +1$

$x_i^T w + w_0 \leq -M \quad y_i = -1$

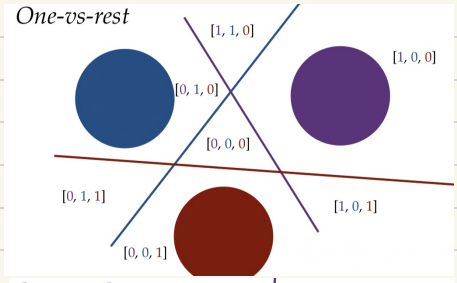
$$\frac{|x_i^T w + w_0|}{\|w\|} = \frac{1}{\|w\|} \Rightarrow \text{margin} = \frac{2}{\|w\|}$$

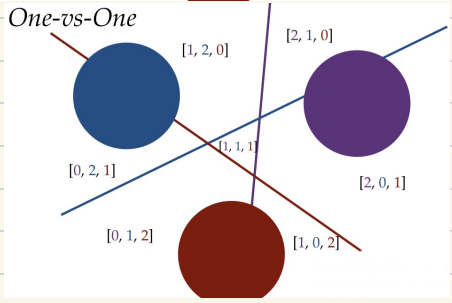
$$\max_{w,w_0} \frac{2}{\|w\|} \rightarrow \min_{w,w_0} \frac{1}{2}\|w\|^2$$

Soft-Margin

$$\min_{w,w_0} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\left(1 - y_i(x_i^T w + w_0)\right)_+$$

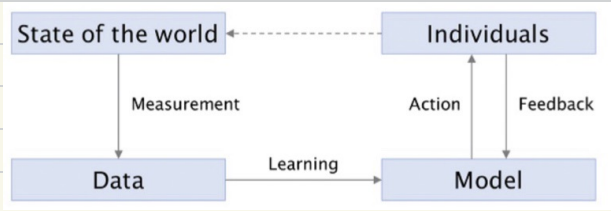
$$P(y=k|x) = \frac{e^{-x^T w_k}}{\sum_{k=1}^{K} e^{-x^T w_k}}$$





V. Ethics

Measurement



race:
- social construct
- changes over time
- different terms

Subjective judgements inherit stereotypes

Learning

Algorithms extract stereotypes, discrimination from data
remove stereotype-identifying information → what's left?

Action

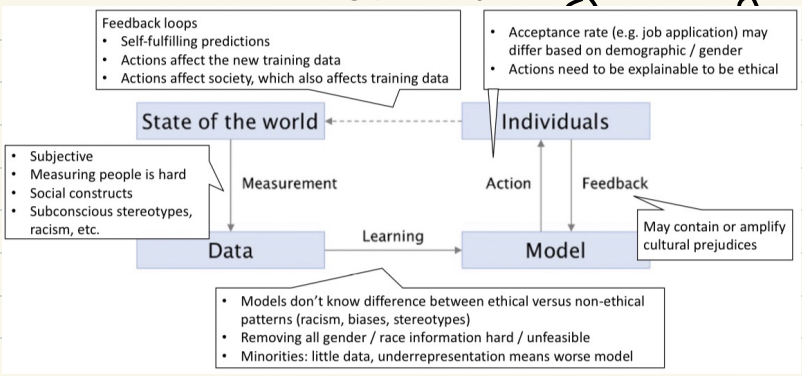
Model reproduces patterns in data
Automated decision → only outcome is important
Ethics → decision process is important

Feedback

Feedback can reflect or amplify cultural prejudices

Loop

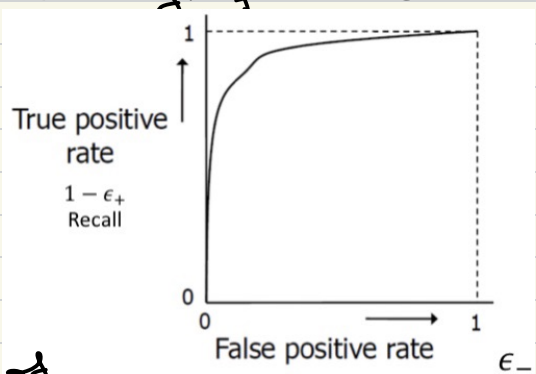
→ Self-fulfilling predictions
→ Predictions that affect the training data
→ Predictions that affect society at large



A - sensitive attribute
X - other features
Y - classification target
R - classifier output R=0 -> reject R=1 -> accept

Criteria 1: Independence
$P(R=1 \mid A=a) = P(R=1 \mid A=b)$
Acceptance rate for each group -> same
Classification rule -> may differ for each group

Criteria 2: Separation
Fraction of mistakes for each group -> same
$\epsilon_+ = \frac{FN}{TP+FN} = \frac{FP}{FP+TN} = \epsilon_-$



Ethics - moral standard of 'right' and 'wrong' that prescribe how we ought to act

Descriptive - explanations of, and factual statements about moral systems people subscribe to, how we act and understand ourselves as moral beings (biology/psychology - anthropology)

Normative - prescriptive accounts on how we ought to act and understand ourselves as moral beings (philosophy)

-> Values (abstract goals or ideas)
-> Norms (rules about actions)
-> Virtues (habits of character)

Normative Ethical Theory - formulation of an account or set of principles that prescribe us how to act
-> consequences of action
-> action itself and norm it's based on
-> agent

Consequentialism
action - morally required is consequences -> maximize happiness
optimize best outcome ↳ minimize suffering
quantification and justification of trade-offs between alternatives

Duty Ethics
never use a person as a mere means to some desired end
person - 'ends in themselves'; always respected
rules for right action irregardless of outcome
absoluteness of basic moral principles

Bias - preference or inclination for or against something
-> Natural / Evolutionary
-> Social / Historical
-> Technologically mediated
Bias -> Prejudice -> Discrimination
↳ preconceived judgements based on inadequate knowledge
discr. -> action that treats someone exclusively based on group

Fairness - given everyone equal consideration and opportunity to benefit in relation to some good or value

VI. Non-Linear Discriminative Classifiers

Decision Tree

split feature space one feature at a time, recursively
-> forms tree structure
-> partition space in "rectangles"
each rectangle/leaf -> value

$$h(x) = \sum_{l \in \text{Leaves}} c_l \, \mathbb{1}(x \in l)$$

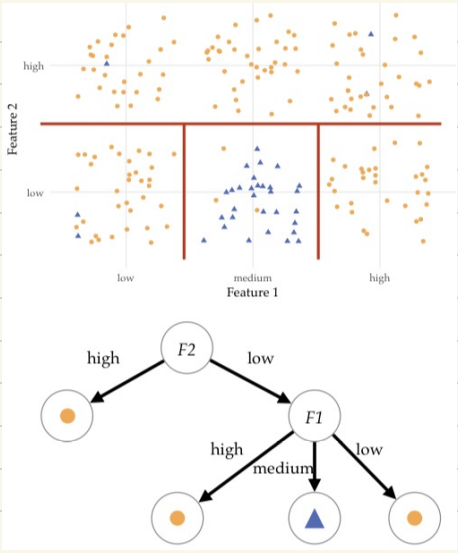
$$\min_{c_l} \sum_{i=1}^{N} \mathbb{1}(h(x_i) \neq y_i)$$

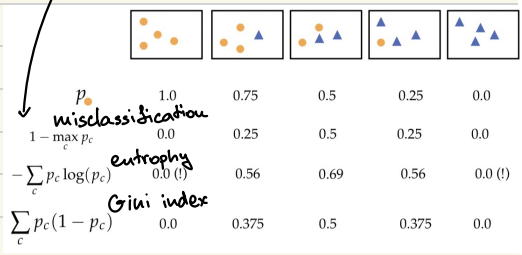
objects in node

$$I(S) - \sum_{v \in \text{values}(F)} \frac{|S_v|}{|S|} I(S_v)$$

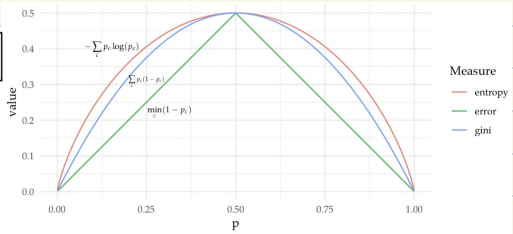
objects value V

feature





p_c	1.0	0.75	0.5	0.25	0.0
misclassification $1 - \max_c p_c$	0.0	0.25	0.5	0.25	0.0
entropy $-\sum_c p_c \log(p_c)$	0.0 (!)	0.56	0.69	0.56	0.0 (!)
Gini index $\sum_c p_c(1 - p_c)$	0.0	0.375	0.5	0.375	0.0



Note: Entropy scaled to have the same maximum

Information Gain

Overal:

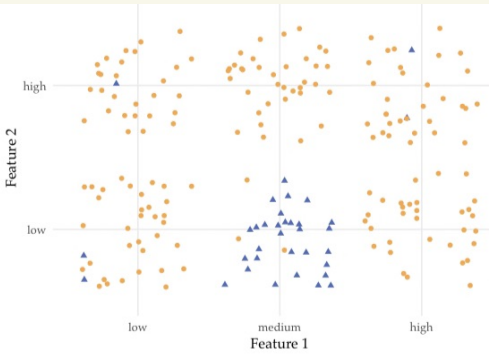
35/200

Feature 1:

Low (3/65), Medium (30/69), High (2/66)

Feature 2:

High (3/99), Low (32/101)



	Node Entropy	Information Gain
	$I(S) = -\sum_c p_c \log(p_c)$	$G(S) - \sum_{v \in Values(F)} \frac{\lvert S_v \rvert}{\lvert S \rvert} G(S_v)$
Overal: 35/200	$-(\frac{35}{200}\log(\frac{35}{200}) + \frac{165}{200}\log(\frac{165}{200})) = 0.464$	
Feature 1:		
Low (3/65)	$-(\frac{3}{65}\log(\frac{3}{65}) + \frac{62}{65}\log(\frac{62}{65})) = 0.187$	
Medium (30/69)	$-(\frac{30}{69}\log(\frac{30}{69}) + \frac{39}{69}\log(\frac{39}{69})) = 0.685$	$0.464 - (\frac{65}{200}0.187 + \frac{69}{200}0.685 + \frac{66}{200}0.136) = 0.122$
High (2/66)	$-(\frac{2}{66}\log(\frac{2}{66}) + \frac{64}{66}\log(\frac{64}{66})) = 0.136$	
Feature 2:		
High (3/99)	$-(\frac{3}{99}\log(\frac{3}{99}) + \frac{96}{99}\log(\frac{96}{99})) = 0.136$	$0.464 - (\frac{99}{200}0.136 + \frac{101}{200}0.624) = 0.081$
Low (32/101)	$-(\frac{32}{101}\log(\frac{32}{101}) + \frac{69}{101}\log(\frac{69}{101})) = 0.624$	

Gain ratio

$$G_G(S) = G(S) - \sum_{v \in Values(F)} \frac{\lvert S_v \rvert}{\lvert S \rvert} G(S_v)$$

$$G_V(S) = -\sum_{v \in Values(F)} \frac{\lvert S_v \rvert}{\lvert S \rvert} \log \frac{\lvert S_v \rvert}{\lvert S \rvert}$$

$$GainRatio(S) = \frac{G_G(S)}{G_V(S)}$$

Criteria

- -> min node size
- -> maximum tree depth
- -> minimum information gain

Regression

- -> squared loss function
- -> Leaf value - average outcome
- -> split minimize variance

Pruning

$$C_\alpha(Tree) = \sum_{t \in Leaves} \lvert S_t \rvert G(S_t) + \alpha \lvert Leaves \rvert$$

Disadvantages

- -> Unstable
- -> cannot model linear relationships efficiently
- -> greedy

Advantages

- -> "Interpretable"
- -> automatic feature selection
- -> easy to incorporate discrete features and missing values
- -> fast

Combining Classifiers

Condorcet's Jury Theorem

If $p > 0.5$, adding voters to the jury increases probability of correct majority vote (assuming voting is independent)

Fixed Rules

hard (majority voting)

soft (mean product)

$$\max_y \sum_{j=1}^{c} \frac{1}{c} h_j(y|x) \qquad \max_y \prod_{j=1}^{c} h_j(y|x) \qquad \min_{P(y|x)} \sum_j D(h_j(y|x), P(y|x))$$

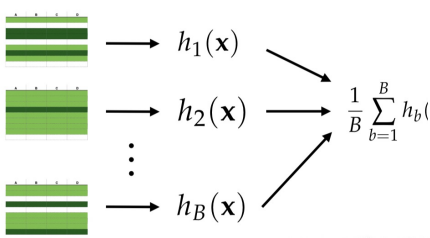
Learned Rules

decision based on set of base classifiers

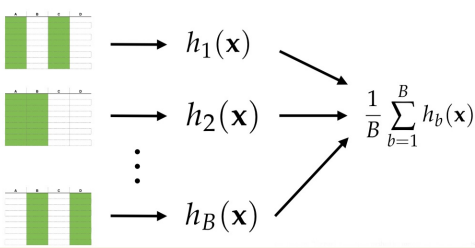
Random Forest - large number of trees using randomly selected objects and features combined

bagging + random subspaces

Bagging (Bootstrap Aggregation)



Random Subspaces



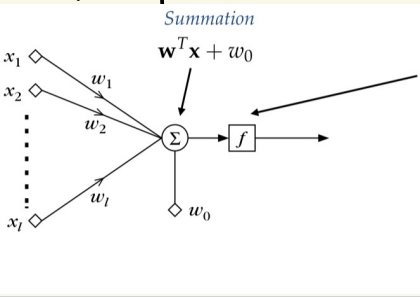
Tree
- -> depth
- -> # leaves
- -> min # object in node
- -> information criterion
- -> pruning

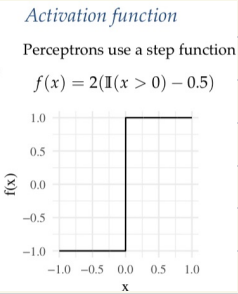
Forest
- -> # trees
- -> size of subspaces

Pros
- -> flexible / low bias
- -> many different types of data
- -> embarrassingly parallel
- -> out-of-bag (OOB) estimates
- -> (scale) invariant
- -> (relatively) few hyperparameters

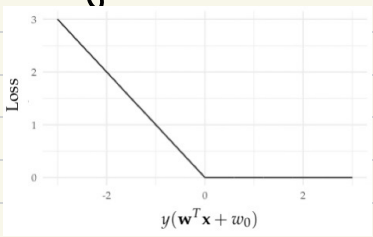
Cons
- -> harder to implement than single trees
- -> computationally expensive

Perceptron



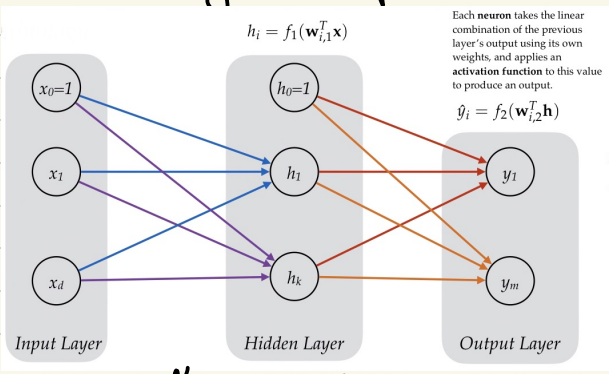


$$\sum_{i=1}^{N} \max(-y_i(w^T x_i + w_0), 0) \qquad y_i \in \{-1, 1\}$$

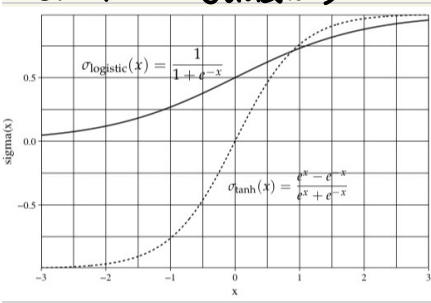


$$\frac{\partial J(w)}{\partial w} = \sum_{y_i(w^T x_i) < 0} -y_i x_i \qquad w^{t+1} = w^t + \alpha_t \sum_{y_i(w^T x_i) < 0} y_i x_i$$

Multi-layer Perceptron



Activation Functions



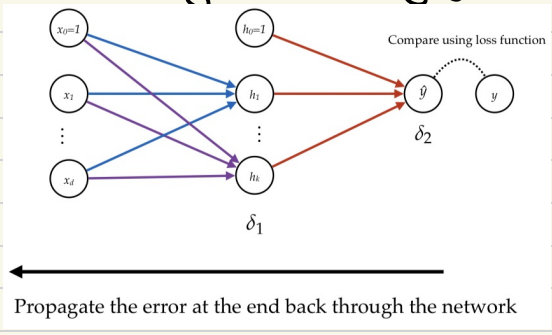
$$J(w) = \sum_{i=1}^{N} L(f_2(w_2^T f_1(w_1 x_i)), y_i)$$

$$x \to a_1 = w_1^T x \to h = f_1(a_1) \to a_2 = w_2^T h \to \hat{y} = f_2(a_2) \to L(\hat{y}, y)$$

Back Propagation

$$w^{t+1} = w^t - \alpha_t \left.\frac{\partial J(w)}{\partial w}\right|_{w=w^t}$$

$$\nabla_w J(w) = \begin{bmatrix} \frac{\partial J(w)}{\partial w_1} \\ \vdots \\ \frac{\partial J(w)}{\partial w_n} \end{bmatrix}$$



$$J(W_2, W_1) = L\left(f_2\left(W_2^T f_1(W_1 x)\right), y\right)$$

$$\left(\nabla_{W_2} J(W_2, W_1)\right)^T = \underbrace{\frac{\partial L(y, \hat{y}_i)}{\partial \hat{y}_i}}_{\delta_2} \cdot \frac{\partial f_2\left(W_2^T f(W_1^T x)\right)}{\partial W_2^T f(W_1^T x)} \cdot \left(f_1(W_1^T x)\right)^T$$

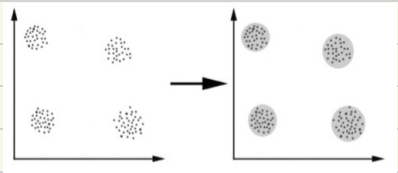
$$\left(\nabla_{W_1} J(W_2, W_1)\right)^T = \underbrace{\frac{\partial L(y, \hat{y}_i)}{\partial \hat{y}_i} \cdot \frac{\partial f_2\left(W_2^T f(W_1^T x)\right)}{\partial W_2^T f(W_1^T x)} \cdot W_2^T \cdot \frac{\partial f_1(W_1^T x)}{\partial W_1^T x}}_{\delta_1 = \delta_2 \cdot W_2^T \cdot f_1'} \cdot \begin{bmatrix} x^T \\ 0^T \\ \dots \\ 0^T \end{bmatrix}$$

Advantages
- -> Flexible model class
- -> Good empirical performance on many (structured) problems
- -> Easily adapted to different learning settings

Disadvantages
- -> Computationally expensive
- -> Lots of hyperparameters
- -> No convergence to a unique optimum
- -> Hard optimization
- -> Hard to interpret

09. Unsupervised Learning

Clustering



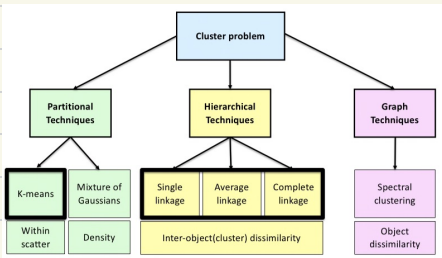
natural groups in data:
- -> items within the group are close
- -> items between groups are far away

Proximity Measure

- -> Similarity measure $s(x_i, x_k)$ ↑ $x_i \sim x_k$
- -> Dissimilarity measure $d(x_i, x_k)$ ↓ $x_i \sim x_k$
- -> Cosine similarity $s_{cos}(x,y) = \frac{x^T y}{\|x\| \|y\|}$
- -> Pearson's correlation coefficient $r_{Pearson}(x,y) = \frac{(x-\mu_x)^T (y-\mu_y)}{\|x-\mu_x\| \|y-\mu_y\|}$

Clustering Evaluation

- Intra-cluster cohesion (compactness)
  - -> how close data points are to cluster's mean
  - -> Sum of squared errors (SSE) us.
- Inter-cluster separation (isolation)
  - -> how far away are different cluster's means



K-means

n data points: $\{x_1, x_2, \dots, x_n\}$

p-dimensional feature vector: $x_i = \{x_{i1}, x_{i2}, \dots, x_{ip}\}$

k clusters

1. k random data points (seeds) for centroids (cluster's mean)
2. Each data point to closest centroid
3. Re-compute centroids using current cluster membership
4. convergence criterion not met, repeat 2&3
   - -> minimum re-assignments of data points to different clusters
   - -> minimum change of centroids
   - -> minimum decrease in the sum of squared errors (SSE)

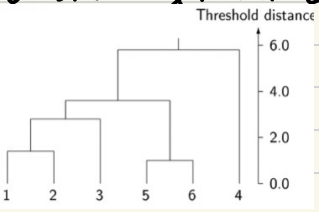
$$J(c, \mu) = \frac{1}{n} \sum_{i=1}^{n} \|x_i - \mu_{c_i}\|^2$$

Initial - run 10000 random, get lowest cost

Advantages
→ fast + simple

Disadvantages
→ only convex (round) clusters
→ sensitive to initialization
→ can get stuck in local minima

Hirerarchical clustering
cluster algorithm can't pick k



top level - all points
bottom level - one cluster per data point

Divisive (top-down)
all points → 1 cluster
split cluster (sensibly)
↳ apply k-means recursively
run k-mean for k=2
for each cluster repeat)

Agglomerative (bottom-up)
each point → cluster
group two closest
sequence of clusters (incr size)
each level, merge two clusters

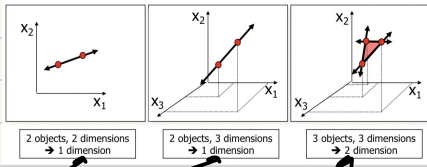
Merging rules
Single linkage - two nearest objects in clusters
$g(R,S) = \min_{ij}\{d(x_i, x_j): x_i \in R, x_j \in S\}$
Complete linkage - two most remote objects in clusters
$g(R,S) = \max_{ij}\{d(x_i, x_j): x_i \in R, x_j \in S\}$
Average linkage - cluster centers
$g(R,S) = \frac{1}{|R||S|}\sum_{ij}'\{d(x_i, x_j): x_i \in R, x_j \in S\}$

Cons
→ Computationally expensive
→ clustering → "hierarchical nesting"

Pros
→ dendogram - overview
→ linkage type - various shapes
→ different dissimilarities

Dimensionality Reduction

- → storage / processing data - easier
- → (visual) discovery of hidden structure
- → redundant and noisy features
- → smaller intrinsic dimensionality



Feature selection

subset of original dimensions $\{X_1, X_2, \dots, X_d\}$
domain knowledge + statistics-based selection methods

Feature extraction

new set of dimensions $E_i = f(X_1, \dots, X_d)$
(linear) combinations of original

Principal Component Analysis

linear subspace → maximized variance
principal components:

1 - direction of greatest variability
2 - perpendicular to 1 + greatest variability
d - original dimensionality

first m components → m new dimensions

$$\max_{\|w\|^2=1} \text{var}(w^T x) \qquad \text{var}(x) = \frac{1}{N}\sum_{n=1}^{N}\left(x_n - \frac{1}{N}\sum_{n=1}^{N} x_n\right)^2 \quad \text{(mean(x))}$$

covariance matrix

$$M = \begin{bmatrix} E[(x_1-\mu_1)(x_1-\mu_1)] & \dots & E[(x_1-\mu_1)(x_D-\mu_D)] \\ \vdots & \ddots & \vdots \\ E[(x_D-\mu_D)(x_1-\mu_1)] & \dots & E[(x_D-\mu_D)(x_D-\mu_D)] \end{bmatrix}$$

zero-mean: $M = \frac{1}{n} X X^T$

↳ assume: $\text{var}(w^T X) = [w^T X X^T w] = w^T M w$

Lagrange multiplier: $\max_{\|w\|^2=1} \text{var}(w^T X) = \max_{w,\lambda} w^T M w - \lambda(1 - w^T w)$

gradient respect to w: $Mw - \lambda w = 0 \Rightarrow Mw = \lambda w$

$Me = \lambda e$ { M-square matrix, λ-constraints, e-non-zero column vector
λ-eigenvalue of M, e-eigenvector of M for eigenvalue λ

1st principal component → eigenvector with highest eigenvalue

$Me = \lambda e$

$(M - \lambda I)e = 0$

$\Rightarrow \det(M - \lambda I) = 0$

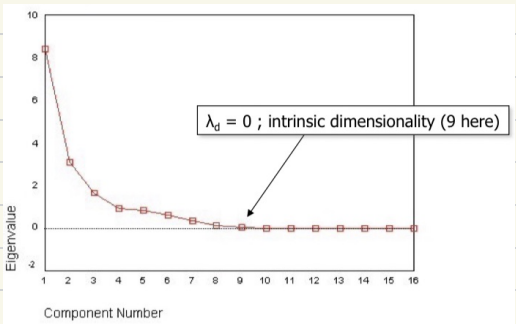
↳ n-degree polynomial $\Rightarrow$ n eigenvalues of M

unit vector x_0

$x_{k+1} := \frac{Mx_k}{\|Mx_k\|}$

converged: $\lambda_1 = x^T M x$

second pair: $M^* = M - \lambda_1 x x^T$



PCA:

X matrix - (zero-mean) points in Euclidean space

covariance XX^T + eigenpairs

E matrix - eigenvector columns, largest eigenvalue first

$X^T E$ → new coordinate space

1st axis (largest eigenvalue) - most significant

E_k - first k columns of E

$X^T E_k$ - k-dimensional X

Issues

Covariance - sensitive to large values

→ normalize dimension to zero mean $x' = \frac{x - \mu}{\sigma}$

Pros

→ reflects intuition about data

→ reduction in data size

Cons

→ expensive for many applications

→ assumptions behind methods