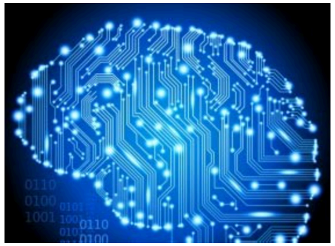
Lecture 18: Learning 6

http://cs.nju.edu.cn/yuy/course_ai15.ashx

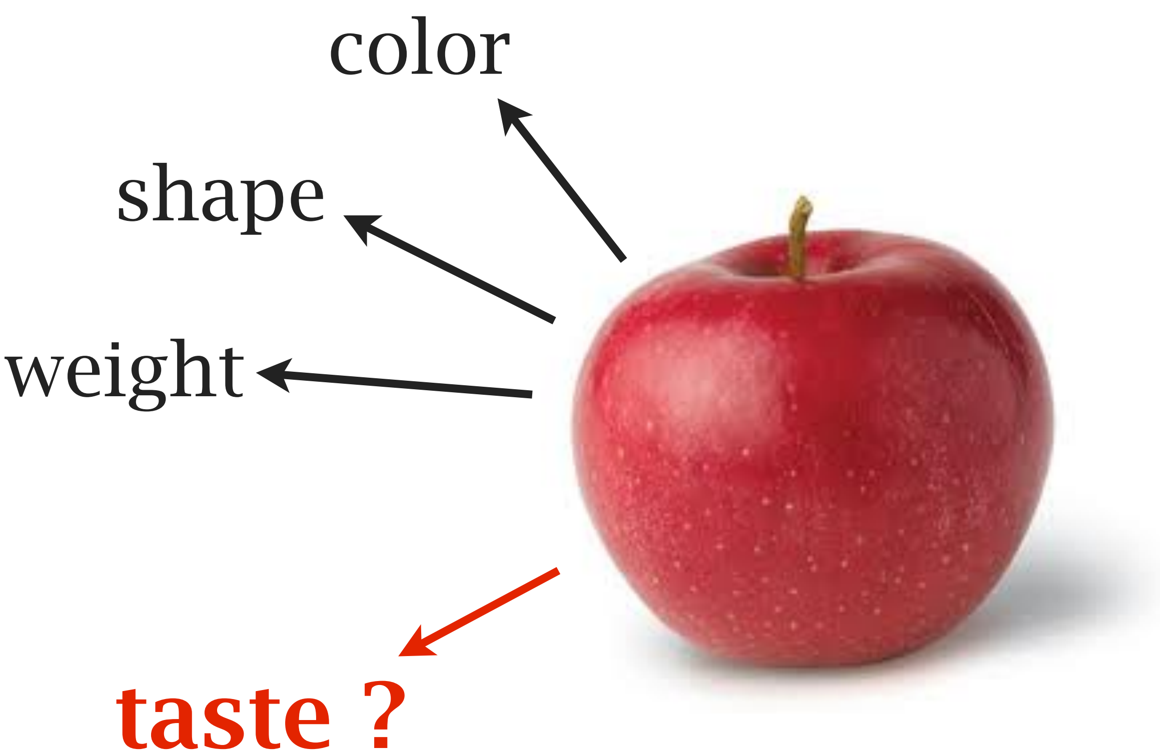


Previously...

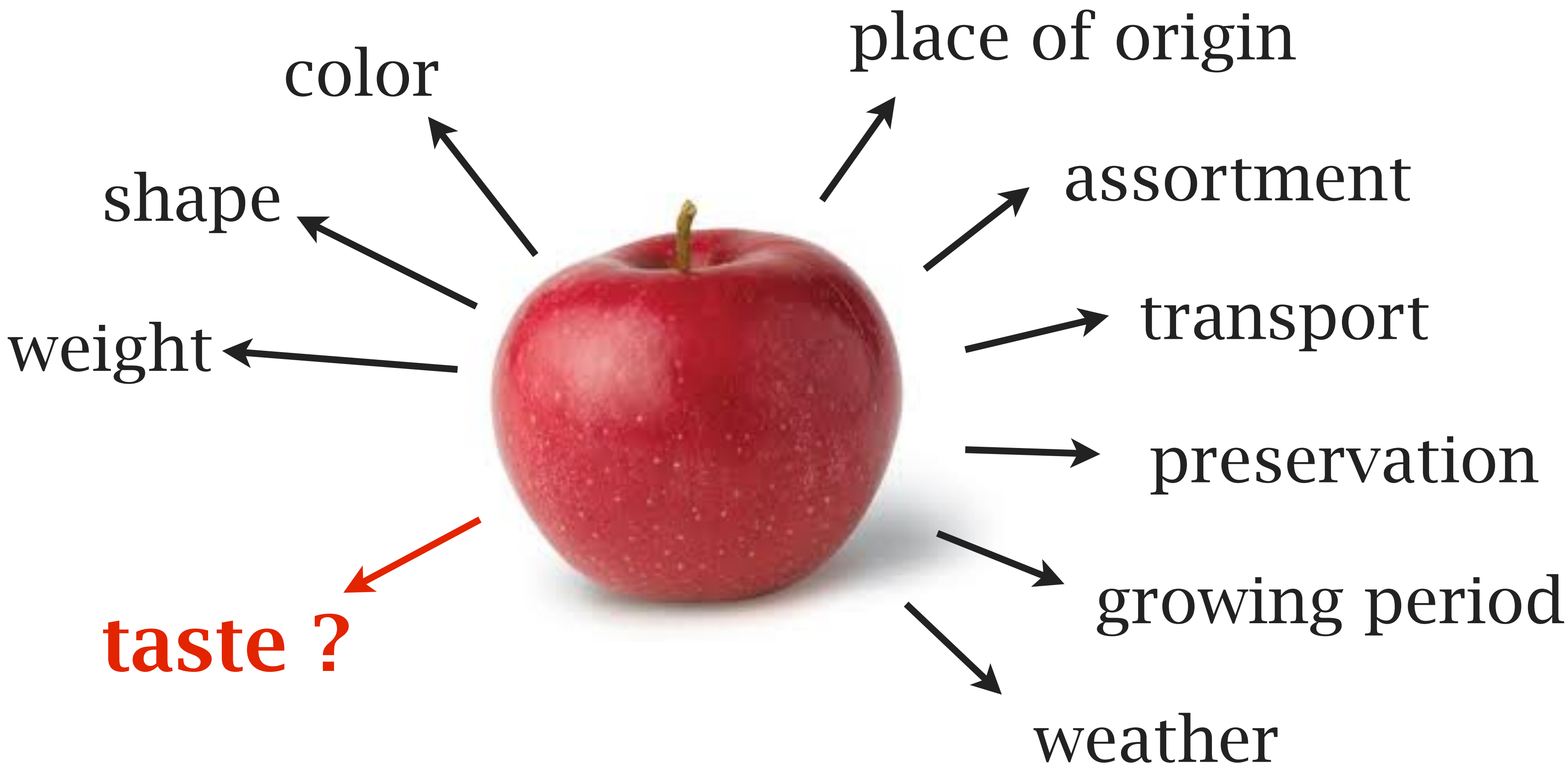
Learning

- Decision tree learning
- Neural networks
- Why we can learn
- Linear models
- Nearest neighbor classifier
- Native Bayes classifier
- Ensemble learning

The importance of features



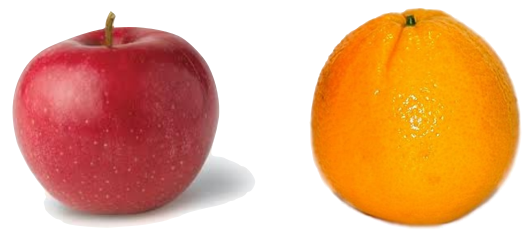
The importance of features

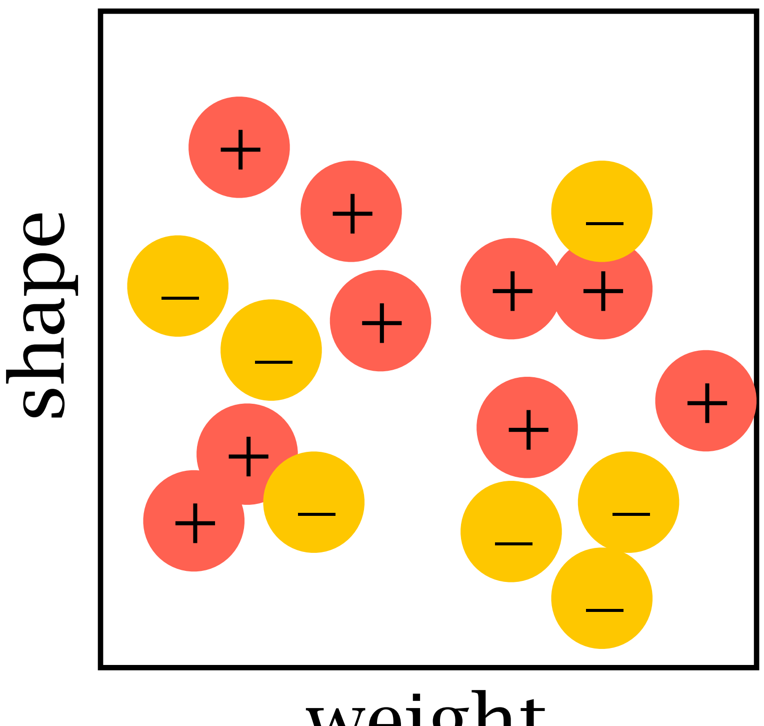


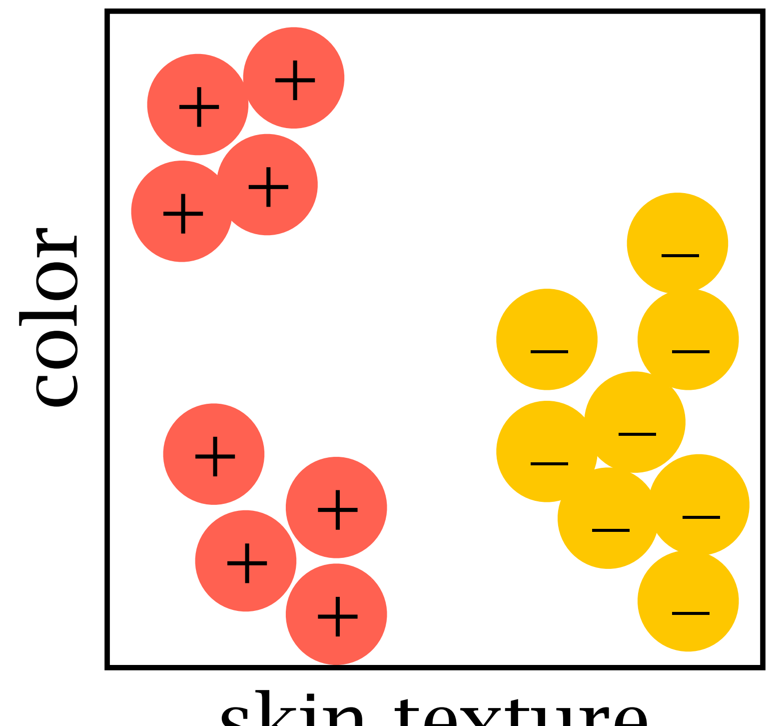
The importance of features

features determine the instance distribution

good features lead to better learning results







Feature processing

a good feature set is more important
than a good classifier

feature selection

feature extraction

Feature selection

To select a set of good features from a given feature set

Improve learning performance

reduce classification error

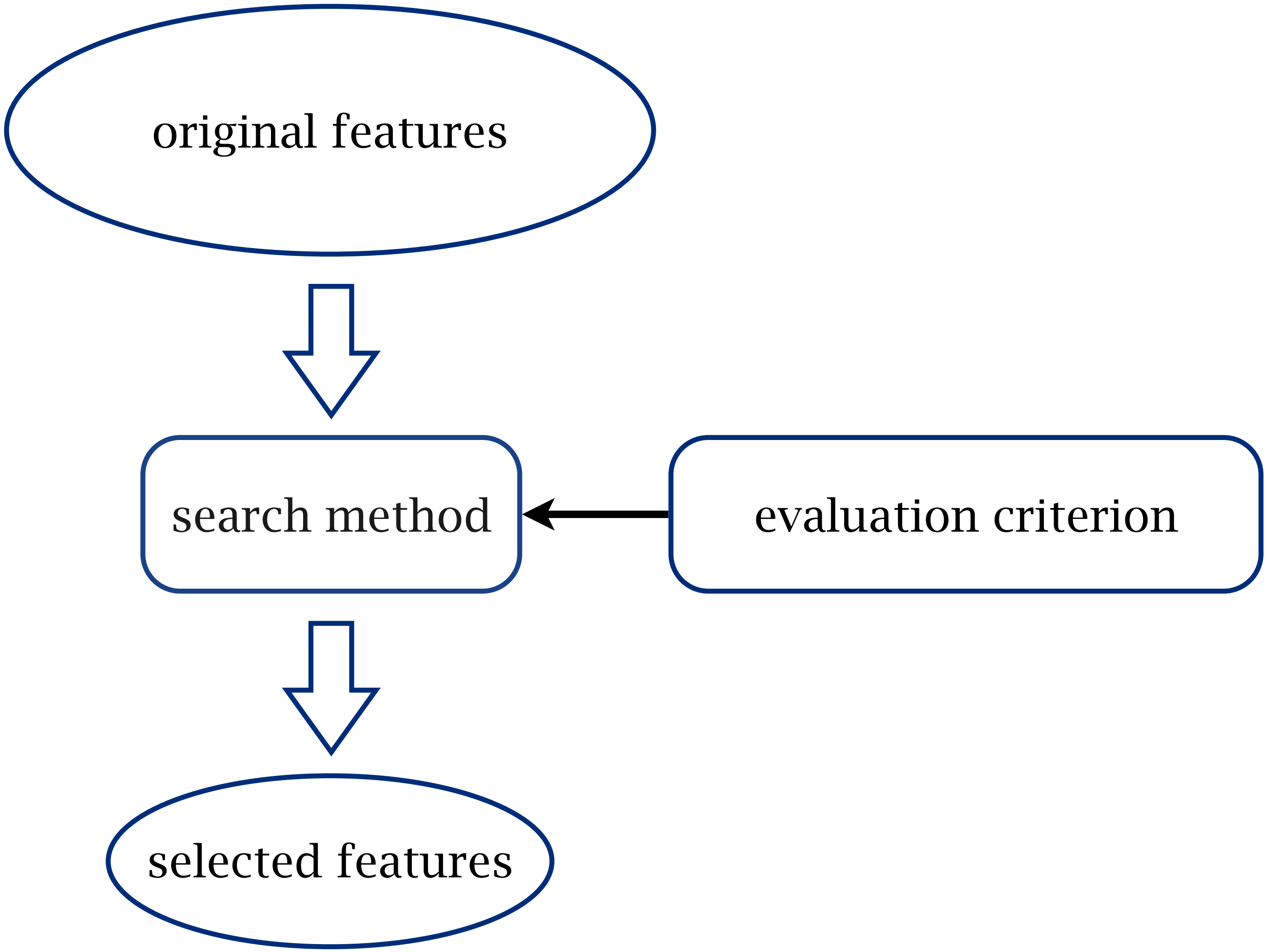
Reduce the time/space complexity of learning

Improve the interpretability

Better data visualization

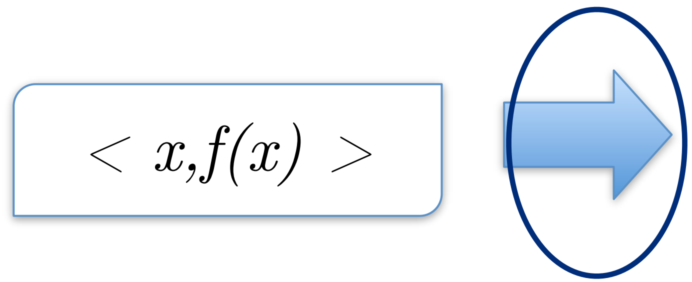
Saving the cost of observing features

Feature selection



Evaluation criteria

classifier independent



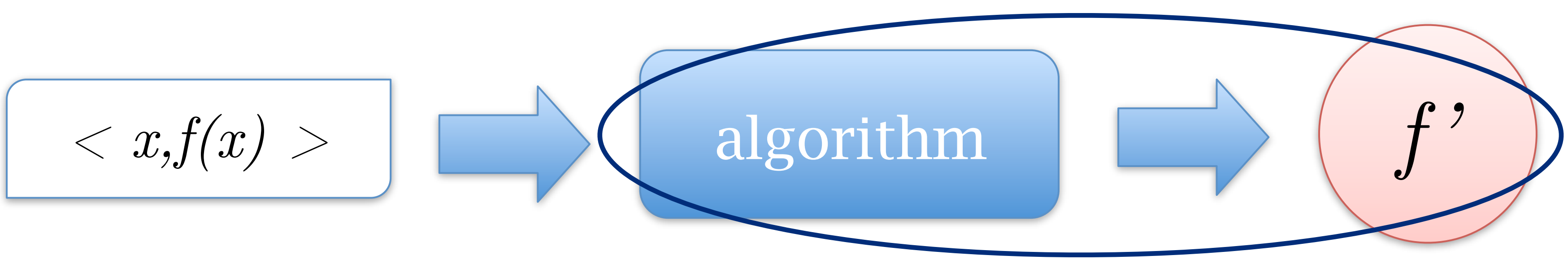
dependency based criteria

information based criteria

distance based criteria

classifier internal weighting

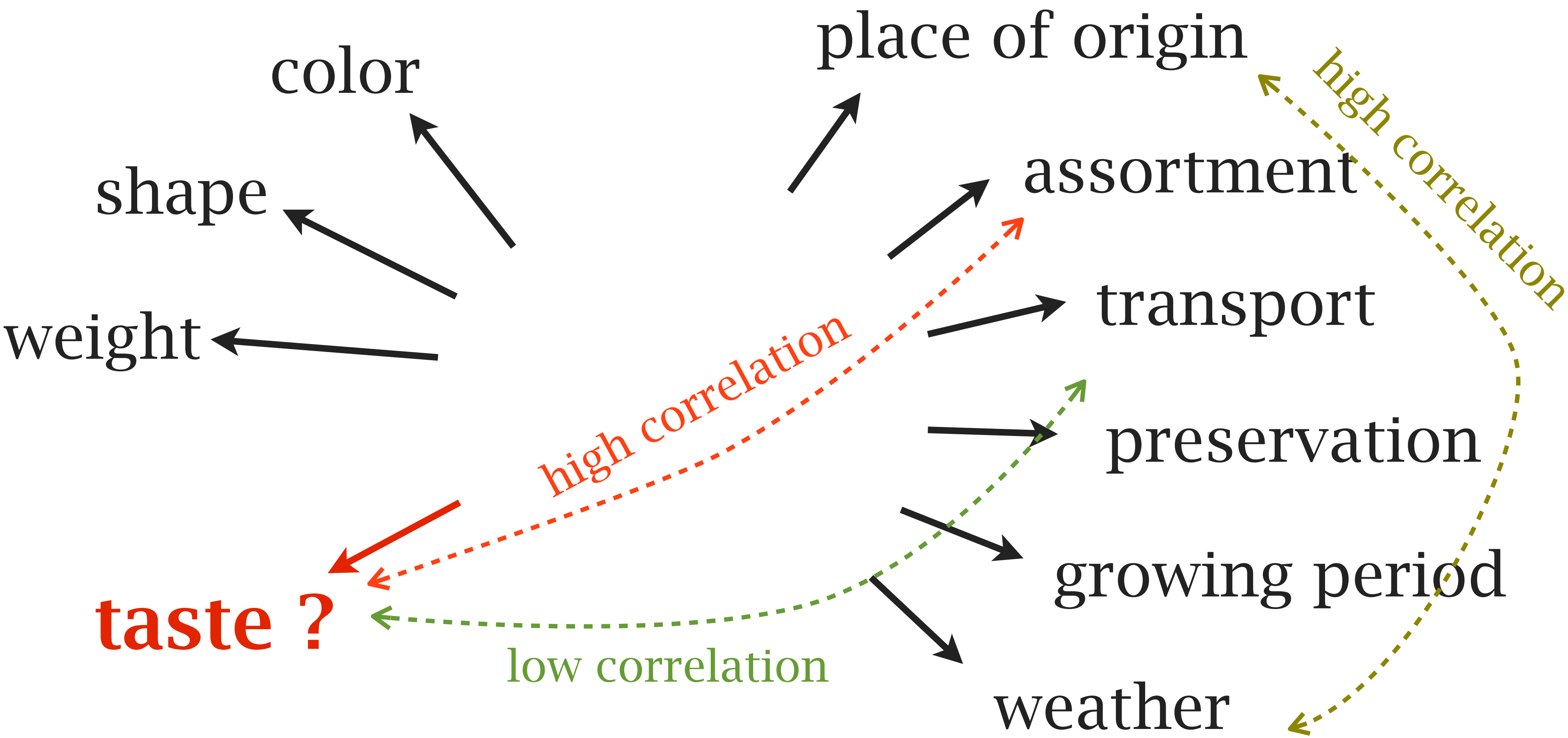
classifier dependent



Dependency based criteria

How a feature set is related with the class

correlation between a feature and the class
correlation between two features
search: select high correlated low redundant features



Information based criteria

How much a feature set provides information about the class

Information gain:

Entropy: $H(X) = -\sum_i p_i \ln(p_i)$

Entropy after split: $I(X;\text{split}) = \sum_j \frac{\#\text{partition } j}{\#\text{all}} H(\text{partition } j)$

Information gain: $H(X)$-$I(X;\text{split})$

A simple forward search

sequentially add the next best feature

F = original feature sets, C is the class label
$S = \emptyset$
loop
    a = the best correlated/informative feature in F
    v = the correlation/IG of a
    if $v < \theta$ **then**
        break
    end if
    $F = F/\{a\}$
    $S = S \cup \{a\}$
end loop
return S

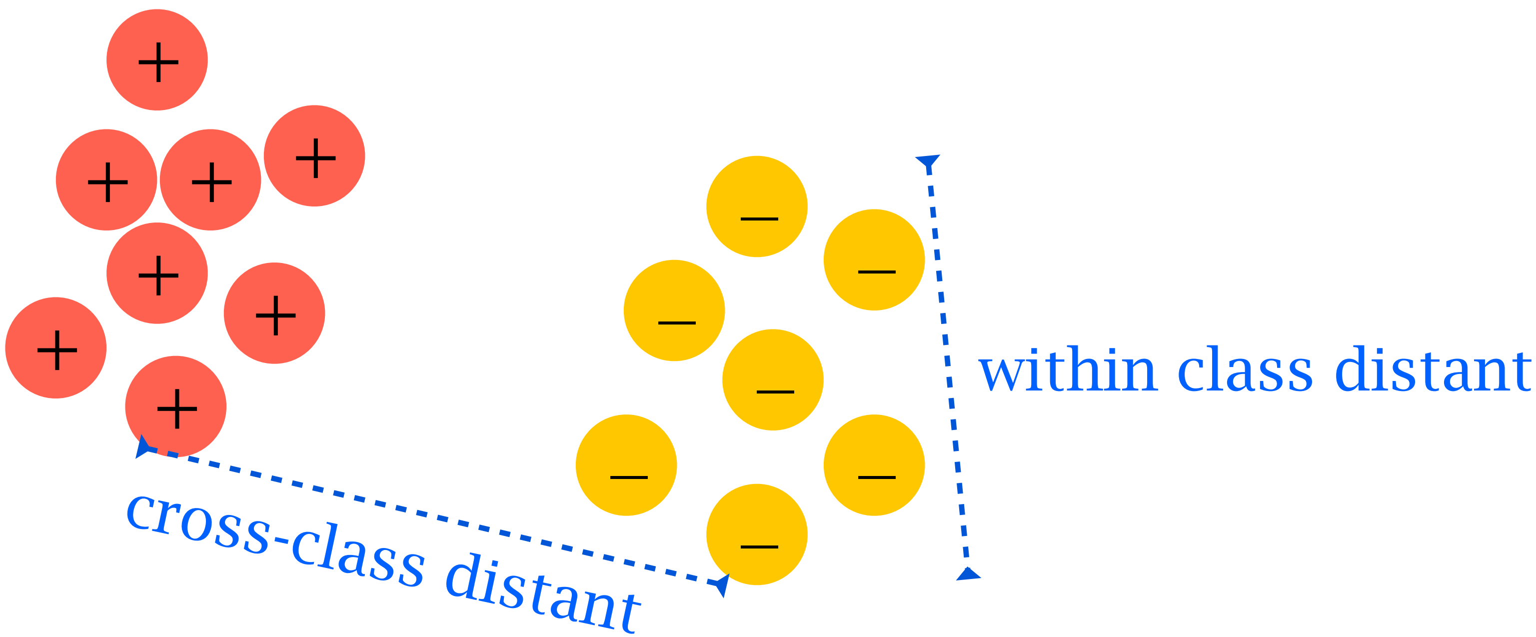
A simple forward search

F = original feature sets, C is the class label
$S = \emptyset$
loop
    a = the best correlated/informative feature in F
    v = the correlation/IG of a
    if $v < \theta$ **then**
        break
    end if
    $F = F/\{a\}$
    $S = S \cup \{a\}$
    for $a' \in F$ **do**
        v' = the correlation/IG of a' to a
        if $v' > \alpha \cdot v$ **then** $F = F/\{a'\}$
        end if
    end for
end loop
return S

remove redundant features

Distance based criteria

Examples in the same class should be near
Examples in different classes should be far



select features to optimize the distance

Distance based criteria

Relief: feature weighting based on distance

$\boldsymbol{w} = 0$

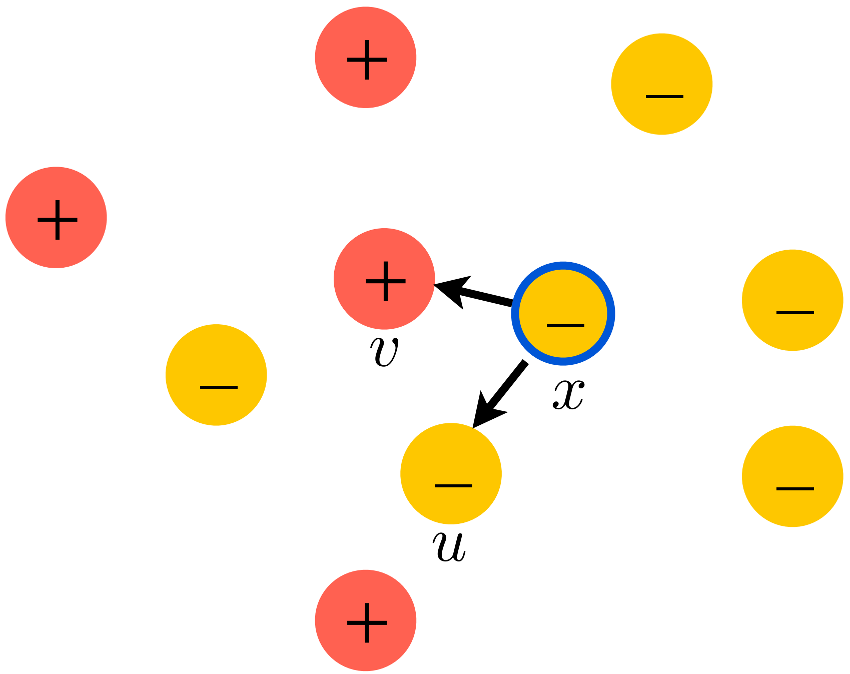
1. random select an instance *x*

2. find the nearest same-class instance u (according to *w*)

3. find the nearest diff-class instance v (according *w*)

4. $\boldsymbol{w} = \boldsymbol{w} - |\boldsymbol{x} - \boldsymbol{u}| + |\boldsymbol{x} - \boldsymbol{v}|$

5. goto 1 for *m* times



select the features whose weights are above a threshold

Feature weighting from classifiers

Many classification algorithms perform feature selection and weighting internally

decision tree: select a set of features by recursive IG

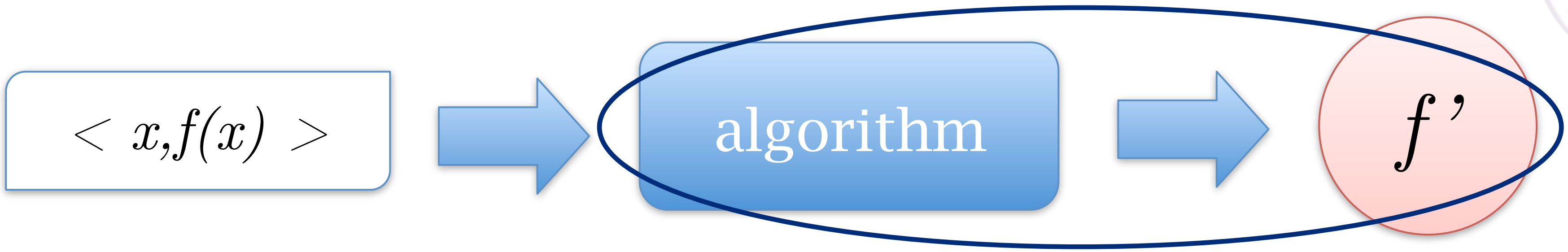
random forest: weight features by the frequency of using a feature

linear model: a natural feature weighting

select features from these models' internal feature weighting

note the difference to FS for classification

Classifier dependent feature selection



select features to maximize the performance of the following learning task

slow in speed
hard to search
hard to generalize the selection results

more accurate learning result

Classifier dependent feature selection

Sequential forward search:
add features one-by-one

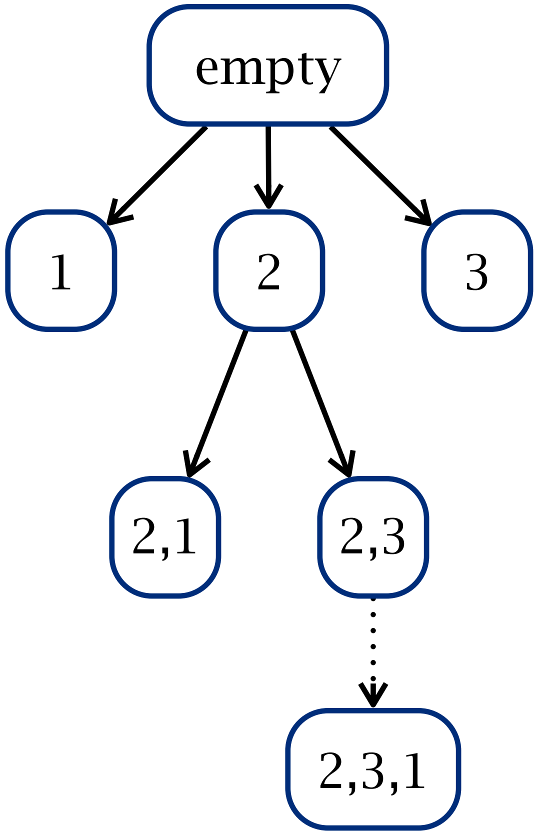
F = original feature set
$S = \emptyset$
perf-so-far = the worst performance value
loop
    for $a \in F$ **do**
        $v(a)$ = the performance given features $S \cup \{a\}$
    end for
    ma = the best feature
    $mv = v(ma)$
    if mv is worse than perf-so-far **then**
        break
    end if
    $S = S \cup ma$
    perf-so-far = mv
end loop
return S

Classifier dependent feature selection

Sequential backward search:
remove features one-by-one

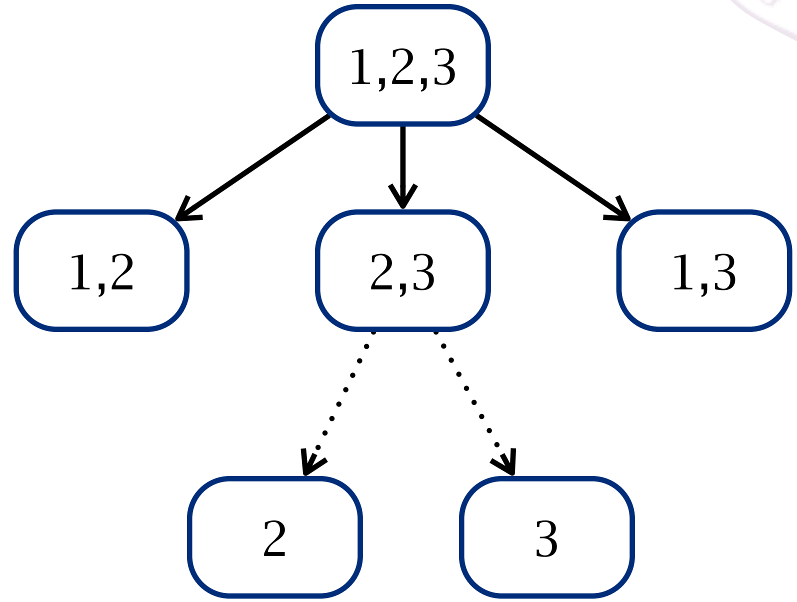
F = original feature set
perf-so-far = the performance given features F
loop
  for $a \in F$ **do**
    $v(a)$ = the performance given features $F/\{a\}$
  end for
  ma = the best feature to remove
  $mv = v(ma)$
  if mv is worse than perf-so-far **then**
    break
  end if
  $F = F/\{ma\}$
  perf-so-far = mv
end loop
return S

Classifier dependent feature selection



forward

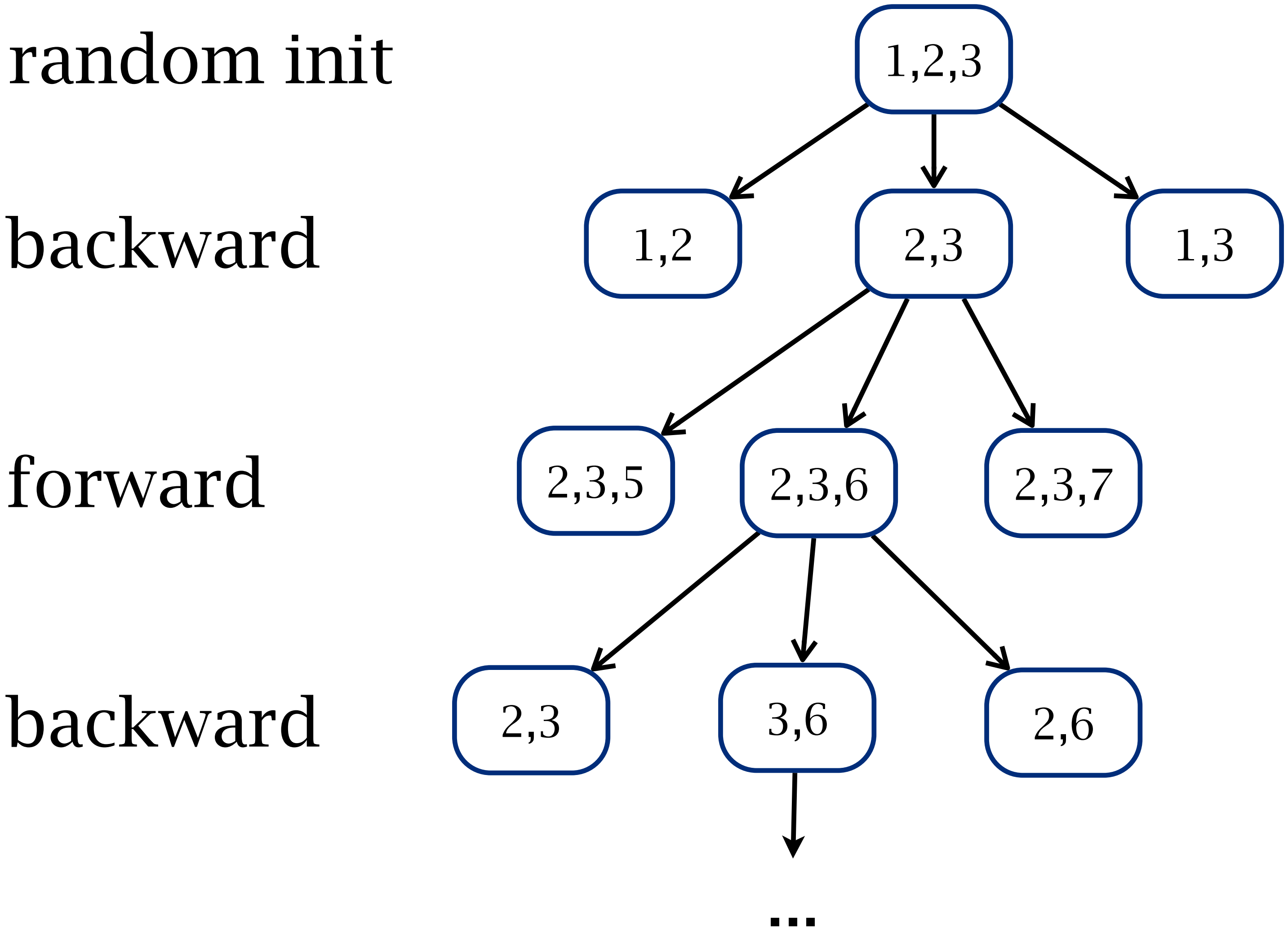
faster



backward

more accurate

Classifier dependent feature selection



combined forward-backward search

Feature extraction

disclosure the inner structure of the data
to support a better learning performance

feature extraction construct new features

commonly followed by a feature selection

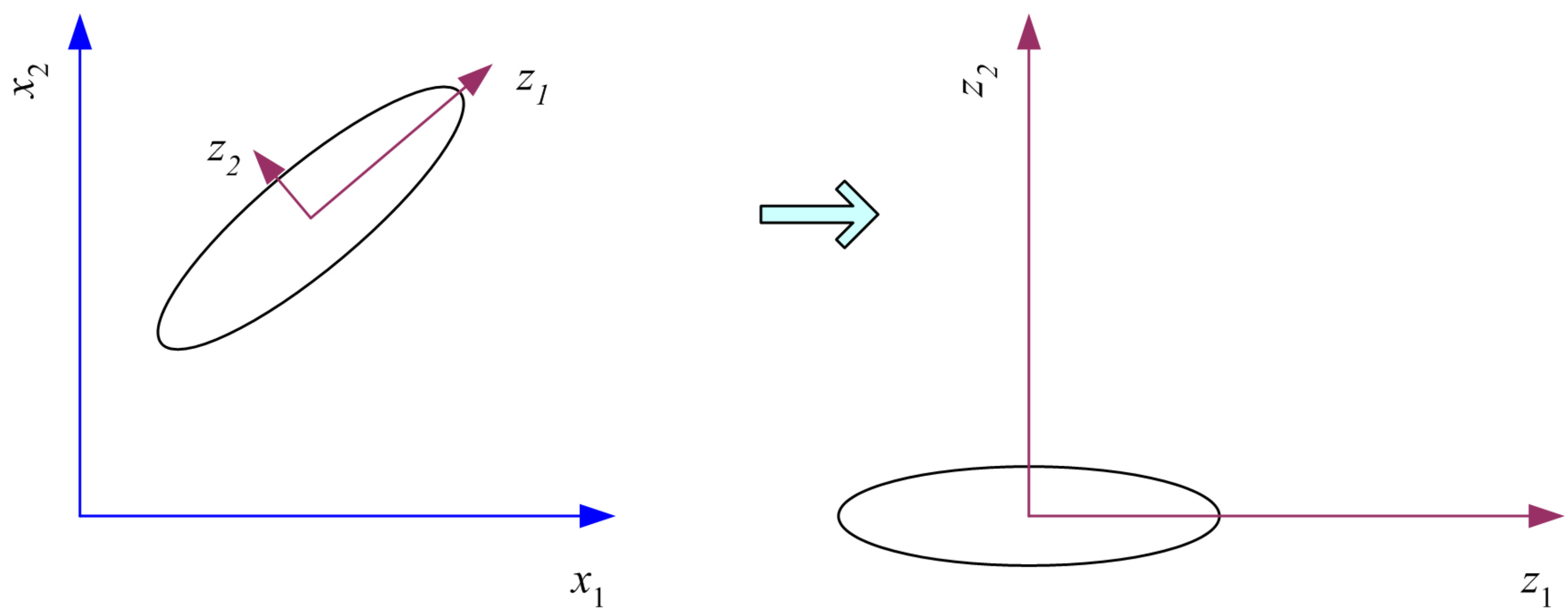
usually used for low-level features

digits bitmap:

Linear methods

Principal components analysis (PCA)

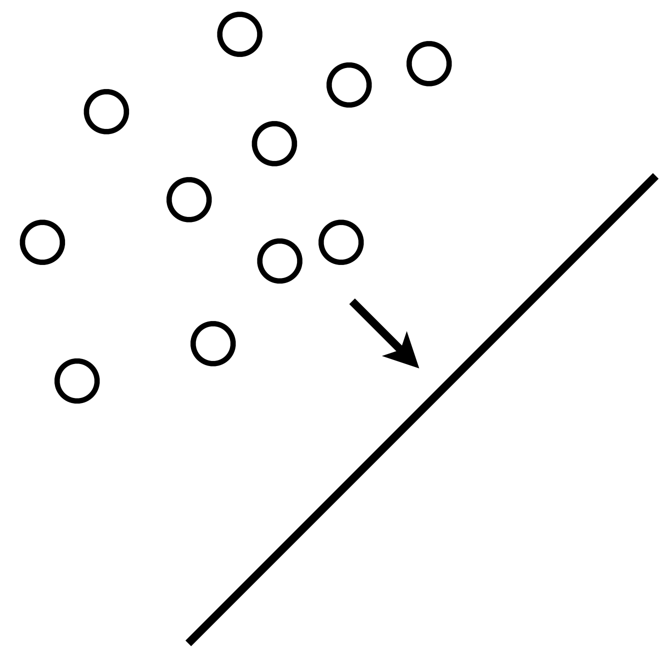
rotate the data to align the directions of the variance



Linear methods

Principal components analysis (PCA)

the first dimension = the largest variance direction

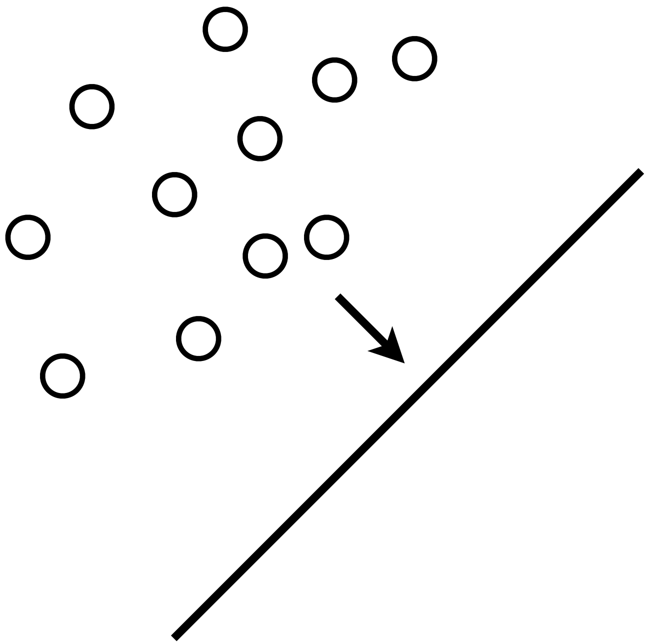


Linear methods

Principal components analysis (PCA)

the first dimension = the largest variance direction

$$z = \boldsymbol{w}^T \boldsymbol{x}$$



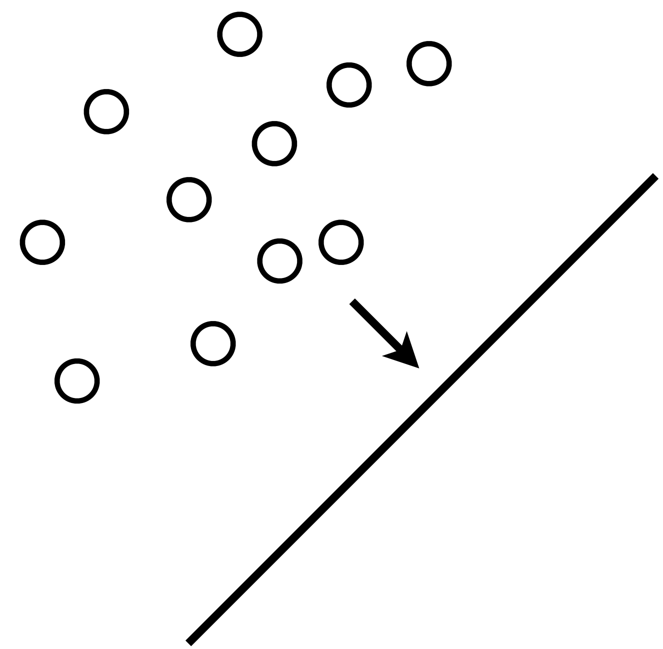
Linear methods

Principal components analysis (PCA)

the first dimension = the largest variance direction

$$z = \boldsymbol{w}^T \boldsymbol{x}$$

$$\mathrm{Var}(z_1) = \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1$$



Linear methods

Principal components analysis (PCA)

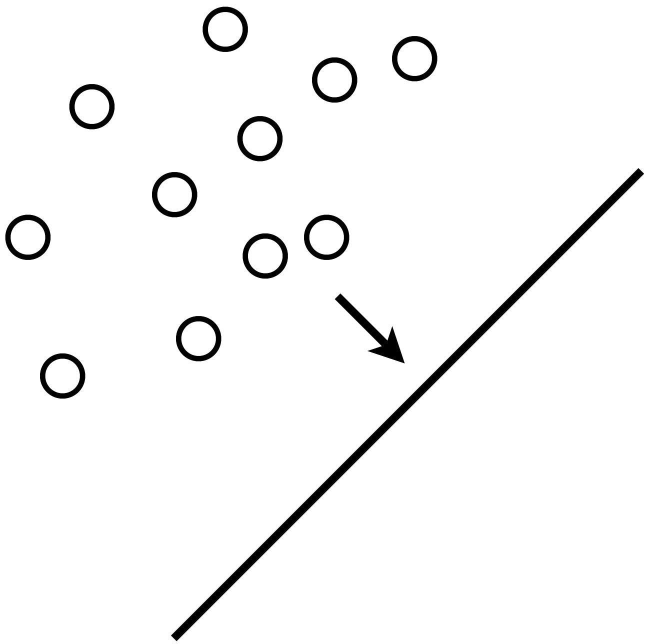
the first dimension = the largest variance direction

$$z = \boldsymbol{w}^T \boldsymbol{x}$$

$$\mathrm{Var}(z_1) = \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1$$

find a unit $\boldsymbol{w}$ to maximize the variance

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



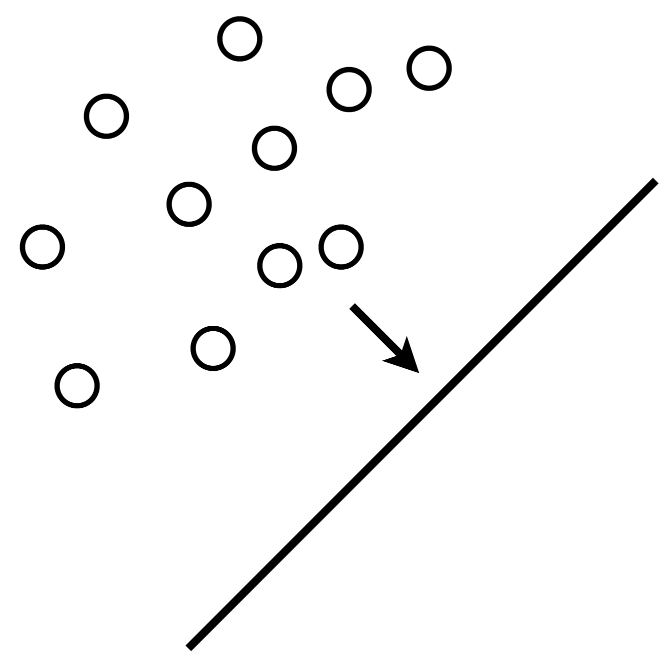
Linear methods

Principal components analysis (PCA)

the first dimension = the largest variance direction

$$z = \boldsymbol{w}^T \boldsymbol{x}$$

$$\mathrm{Var}(z_1) = \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1$$



find a unit $\boldsymbol{w}$ to maximize the variance

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

$2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\alpha\boldsymbol{w}_1 = 0,$ and therefore $\boldsymbol{\Sigma}\boldsymbol{w}_1 = \alpha\boldsymbol{w}_1$

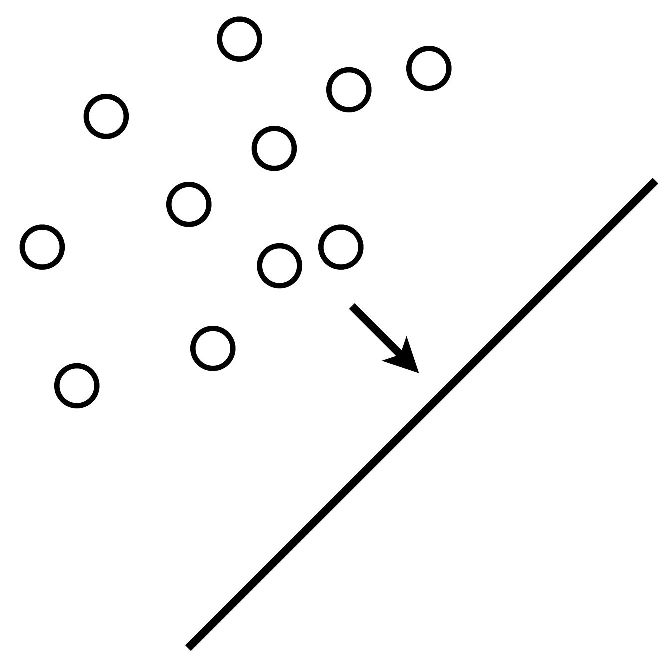
Linear methods

Principal components analysis (PCA)

the first dimension = the largest variance direction

$$z = \boldsymbol{w}^T \boldsymbol{x}$$

$$\mathrm{Var}(z_1) = \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1$$



find a unit $\boldsymbol{w}$ to maximize the variance

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

$2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\alpha\boldsymbol{w}_1 = 0$, and therefore $\boldsymbol{\Sigma}\boldsymbol{w}_1 = \alpha\boldsymbol{w}_1$

$$\boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1 = \alpha \boldsymbol{w}_1^T \boldsymbol{w}_1 = \alpha$$

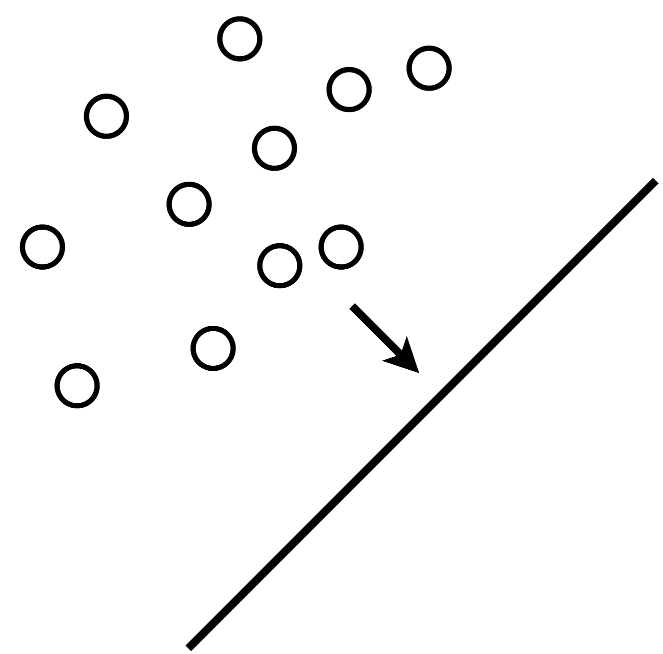
Linear methods

Principal components analysis (PCA)

the first dimension = the largest variance direction

$$z = \boldsymbol{w}^T \boldsymbol{x}$$

$$\mathrm{Var}(z_1) = \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1$$



find a unit $\boldsymbol{w}$ to maximize the variance

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

$2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\alpha\boldsymbol{w}_1 = 0,$ and therefore $\boldsymbol{\Sigma}\boldsymbol{w}_1 = \alpha\boldsymbol{w}_1$

$$\boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1 = \alpha \boldsymbol{w}_1^T \boldsymbol{w}_1 = \alpha$$

$\boldsymbol{w}$ is the eigenvector with the largest eigenvalue

Linear methods

Principal components analysis (PCA)

the second dimension = the largest variance direction orthogonal to the first dimension

Linear methods

Principal components analysis (PCA)

the second dimension = the largest variance direction orthogonal to the first dimension

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

Linear methods

Principal components analysis (PCA)

the second dimension = the largest variance direction orthogonal to the first dimension

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

$$2\Sigma \boldsymbol{w}_2 - 2\alpha \boldsymbol{w}_2 - \beta \boldsymbol{w}_1 = 0$$

Linear methods

Principal components analysis (PCA)

the second dimension = the largest variance direction orthogonal to the first dimension

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

$$2\boldsymbol{\Sigma}\boldsymbol{w}_2 - 2\alpha\boldsymbol{w}_2 - \beta\boldsymbol{w}_1 = 0$$

$$\beta = 0 \qquad \boldsymbol{\Sigma}\boldsymbol{w}_2 = \alpha\boldsymbol{w}_2$$

Linear methods

Principal components analysis (PCA)

the second dimension = the largest variance direction orthogonal to the first dimension

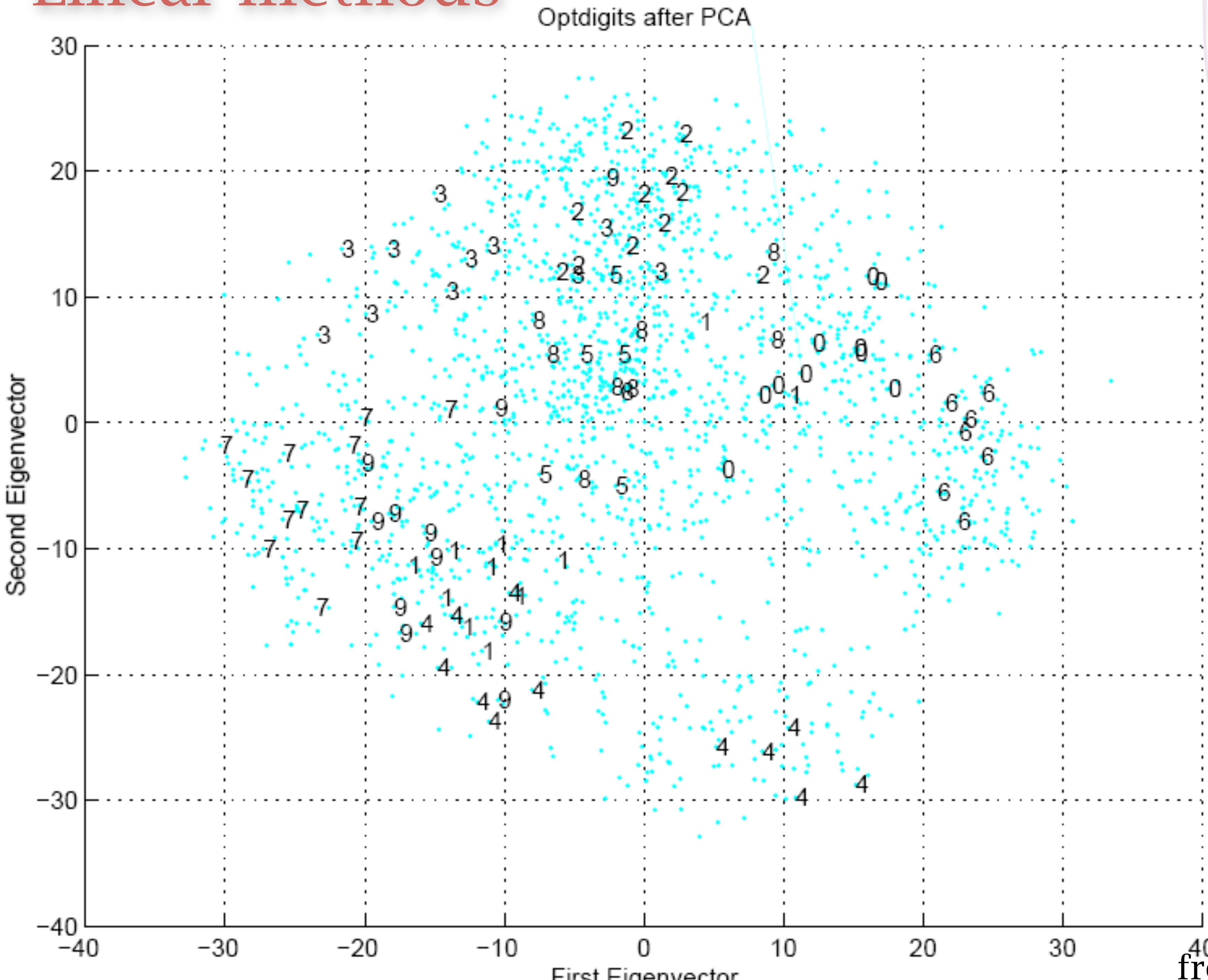
$$\max_{\boldsymbol{w}_2} \boldsymbol{w}_2^T \boldsymbol{\Sigma} \boldsymbol{w}_2 - \alpha(\boldsymbol{w}_2^T \boldsymbol{w}_2 - 1) - \beta(\boldsymbol{w}_2^T \boldsymbol{w}_1 - 0)$$

$$2\boldsymbol{\Sigma}\boldsymbol{w}_2 - 2\alpha\boldsymbol{w}_2 - \beta\boldsymbol{w}_1 = 0$$

$$\beta = 0 \qquad \boldsymbol{\Sigma}\boldsymbol{w}_2 = \alpha\boldsymbol{w}_2$$

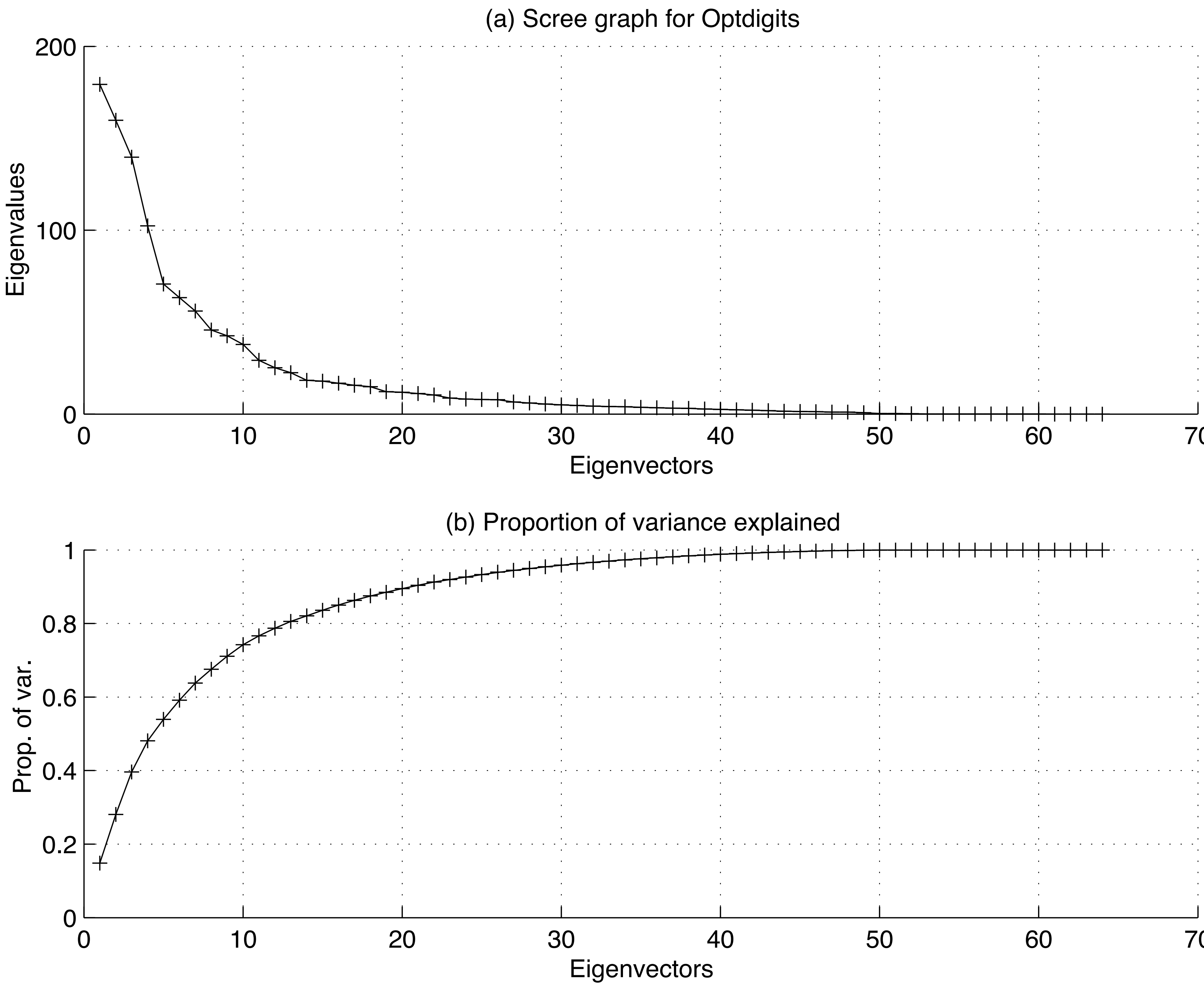
$\boldsymbol{w}$'s are the eigenvectors sorted by the eigenvalues

Linear methods



from [Intro. ML]

Linear methods

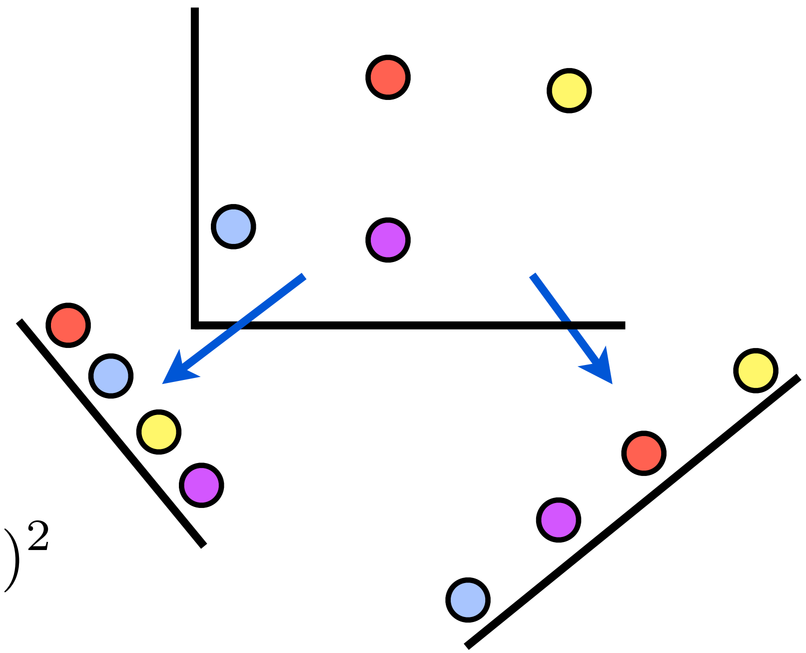


from [Intro. ML]

Linear methods

Multidimensional Scaling (MDS)

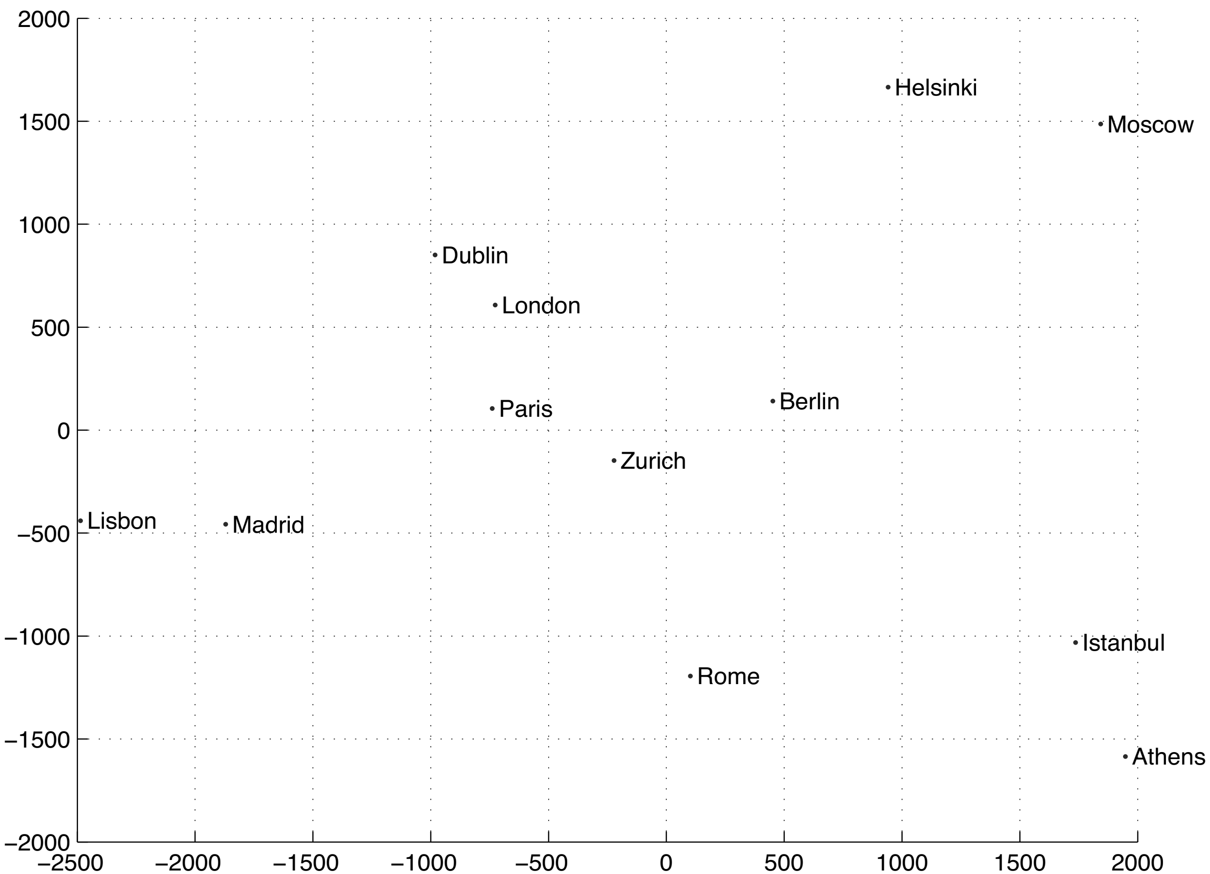
keep the distance into a lower dimensional space



for linear transformation,
W is an n*k matrix

$$\arg\min_W \sum_{i,j} (\|\boldsymbol{x}_i^\top W - \boldsymbol{x}_j^\top W\| - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|)^2$$

Linear methods



from [Intro. ML]

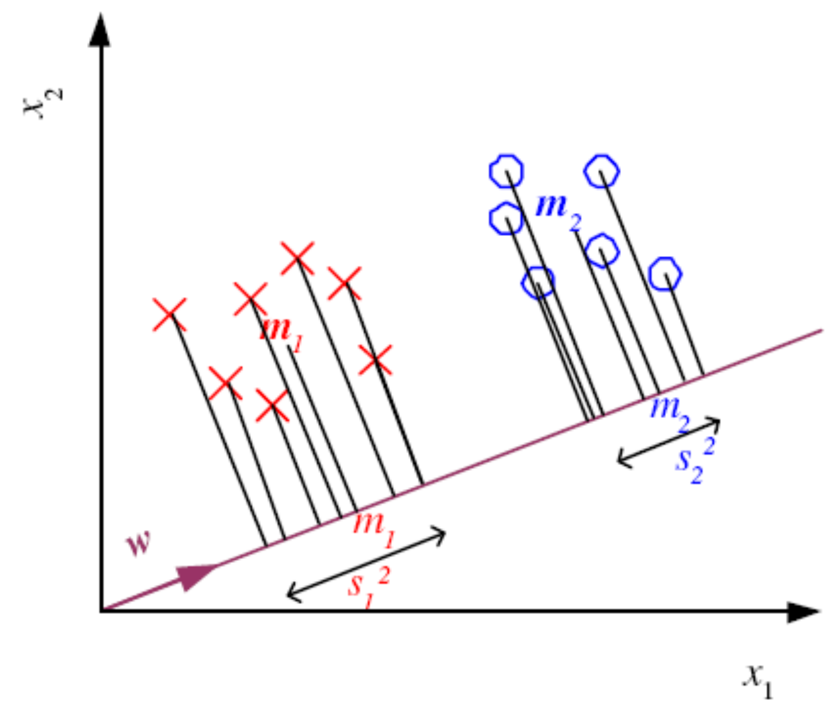
Linear methods

Linear Discriminant Analysis (LDA)

find a direction such that the two classes are well separated

$$z = \boldsymbol{w}^T \boldsymbol{x}$$

m be the mean of a class
s^2 be the variance of a class



maximize the criterion

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

Linear methods

Linear Discriminant Analysis (LDA)

Linear methods

Linear Discriminant Analysis (LDA)

$$
\begin{aligned}
(m_1 - m_2)^2 &= (\boldsymbol{w}^T \boldsymbol{m}_1 - \boldsymbol{w}^T \boldsymbol{m}_2)^2 \\
&= \boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)(\boldsymbol{m}_1 - \boldsymbol{m}_2)^T \boldsymbol{w} \\
&= \boldsymbol{w}^T \mathbf{S}_B \boldsymbol{w}
\end{aligned}
$$

Linear methods

Linear Discriminant Analysis (LDA)

$$
\begin{aligned}
(m_1 - m_2)^2 &= (\boldsymbol{w}^T \boldsymbol{m}_1 - \boldsymbol{w}^T \boldsymbol{m}_2)^2 \\
&= \boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)(\boldsymbol{m}_1 - \boldsymbol{m}_2)^T \boldsymbol{w} \\
&= \boldsymbol{w}^T \mathbf{S}_B \boldsymbol{w}
\end{aligned}
$$

$$
\begin{aligned}
s_1^2 &= \sum_t (\boldsymbol{w}^T \boldsymbol{x}^t - m_1)^2 r^t \\
&= \sum_t \boldsymbol{w}^T (\boldsymbol{x}^t - \boldsymbol{m}_1)(\boldsymbol{x}^t - \boldsymbol{m}_1)^T \boldsymbol{w} r^t \\
&= \boldsymbol{w}^T \mathbf{S}_1 \boldsymbol{w}
\end{aligned}
$$

Linear methods

Linear Discriminant Analysis (LDA)

$$
\begin{aligned}
(m_1 - m_2)^2 &= (\boldsymbol{w}^T \boldsymbol{m}_1 - \boldsymbol{w}^T \boldsymbol{m}_2)^2 \\
&= \boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)(\boldsymbol{m}_1 - \boldsymbol{m}_2)^T \boldsymbol{w} \\
&= \boldsymbol{w}^T \mathbf{S}_B \boldsymbol{w}
\end{aligned}
$$

$$
\begin{aligned}
s_1^2 &= \sum_t (\boldsymbol{w}^T \boldsymbol{x}^t - m_1)^2 r^t \\
&= \sum_t \boldsymbol{w}^T (\boldsymbol{x}^t - \boldsymbol{m}_1)(\boldsymbol{x}^t - \boldsymbol{m}_1)^T \boldsymbol{w} r^t \\
&= \boldsymbol{w}^T \mathbf{S}_1 \boldsymbol{w}
\end{aligned}
$$

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

Linear methods

Linear Discriminant Analysis (LDA)

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

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

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

The objective becomes:

$$J(\boldsymbol{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} = \frac{\boldsymbol{w}^T \mathbf{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}} = \frac{|\boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)|^2}{\boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}}$$

Linear methods

Linear Discriminant Analysis (LDA)

The objective becomes:

$$J(\boldsymbol{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} = \frac{\boldsymbol{w}^T \mathbf{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}} = \frac{|\boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)|^2}{\boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}}$$

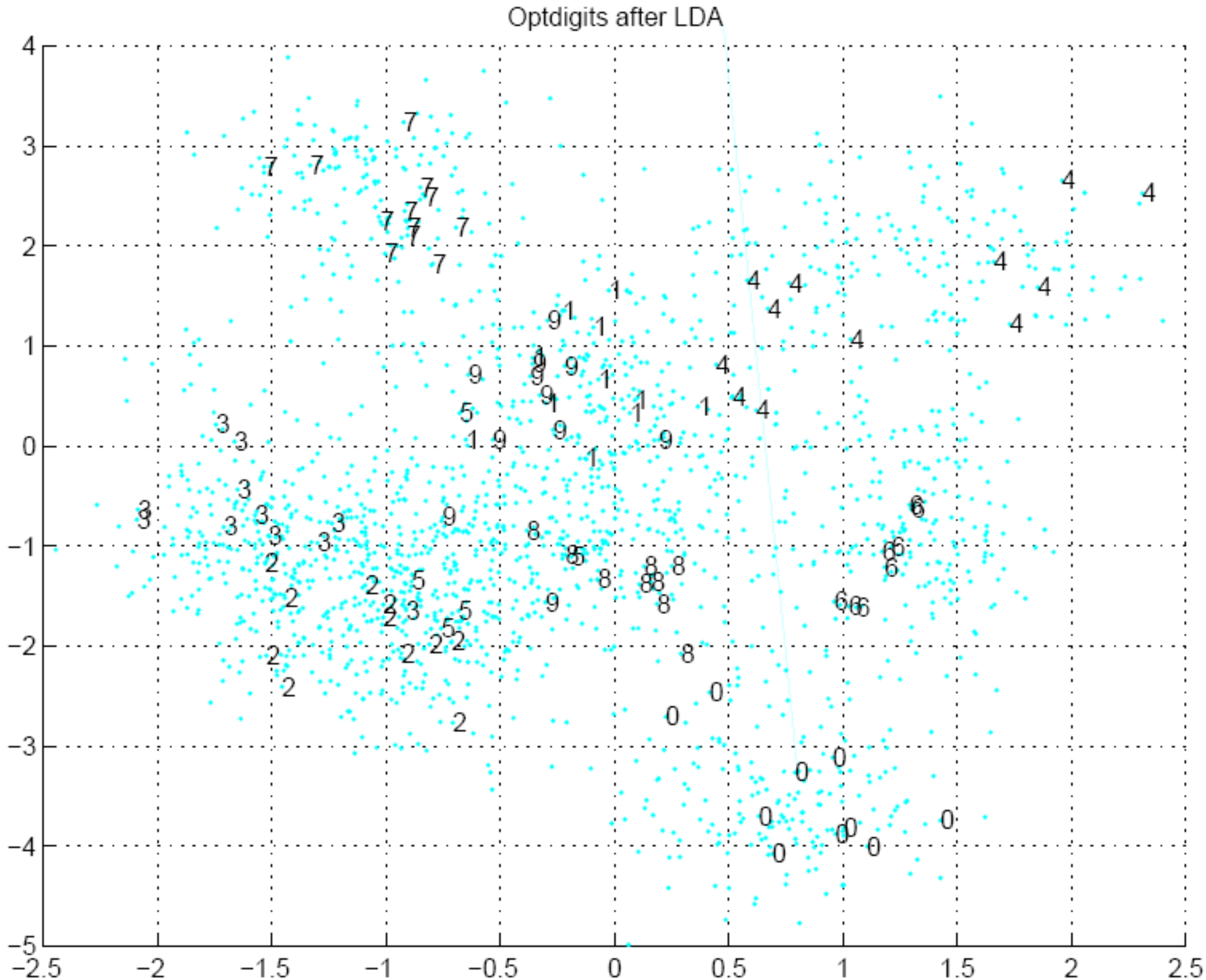
$$\frac{\boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)}{\boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}} \left(2(\boldsymbol{m}_1 - \boldsymbol{m}_2) - \frac{\boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2)}{\boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}} \mathbf{S}_W \boldsymbol{w} \right) = 0$$

Given that $\boldsymbol{w}^T (\boldsymbol{m}_1 - \boldsymbol{m}_2) / \boldsymbol{w}^T \mathbf{S}_W \boldsymbol{w}$ is a constant, we have

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

just take $c = 1$ and find $\boldsymbol{w}$

Linear methods

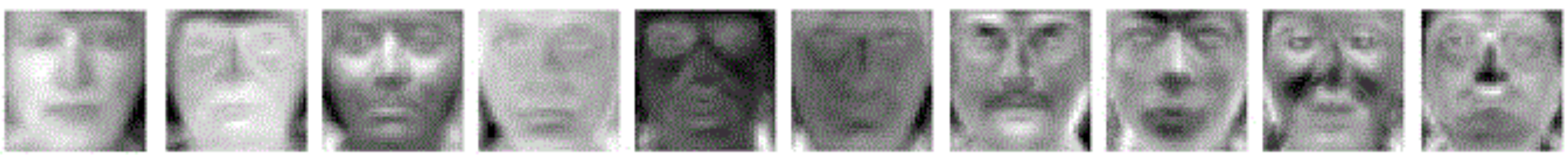


from [Intro. ML]

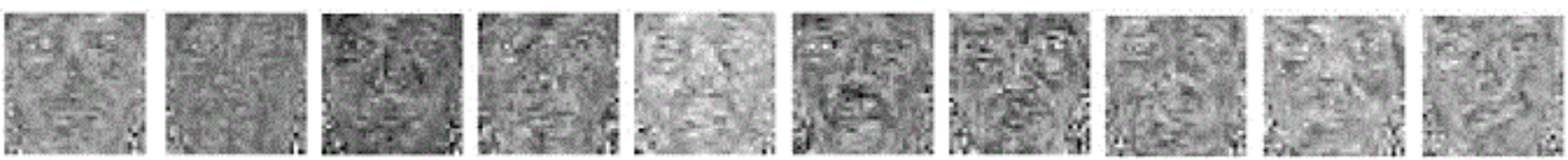
Example: Face recognition

PCA and LDA are commonly used to extract features for face recognition.

Basis of eigenface (PCA):



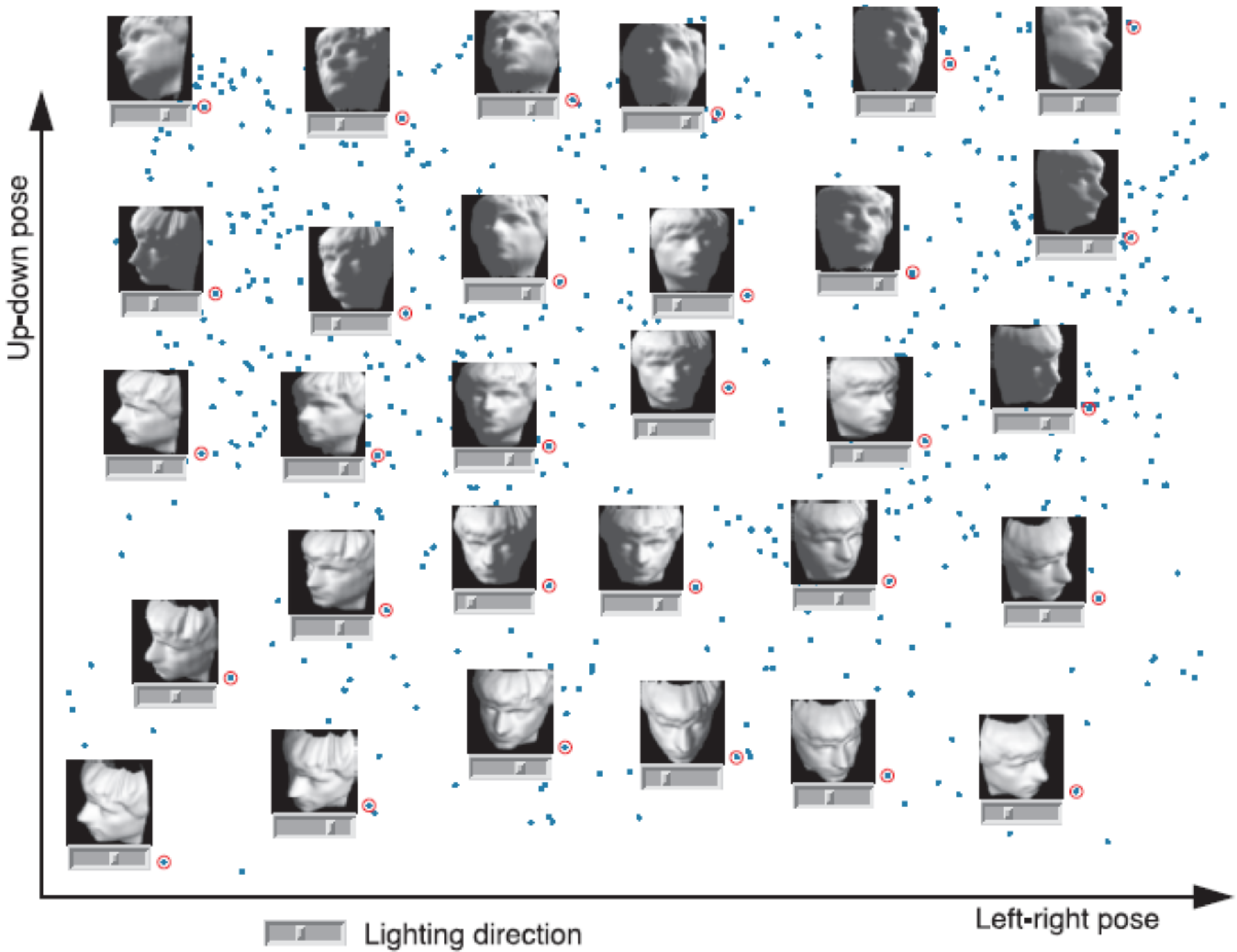
Basis of Fisherface (LDA):



[image from http://commons.wikimedia.org/wiki/File:Fisherface_eigenface_laplacianface.GIF]

Manifold learning

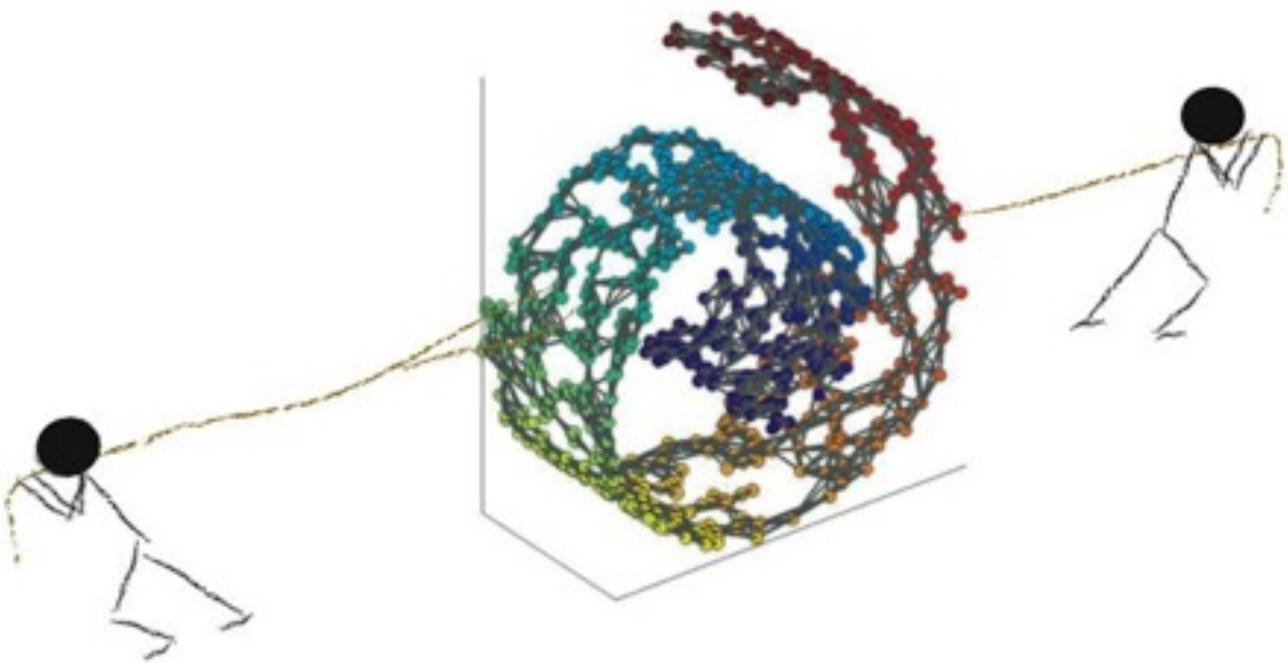
Manifold learning



Manifold learning

A low intrinsic dimensional data embedded in a high dimensional space

cause a bad distance measure



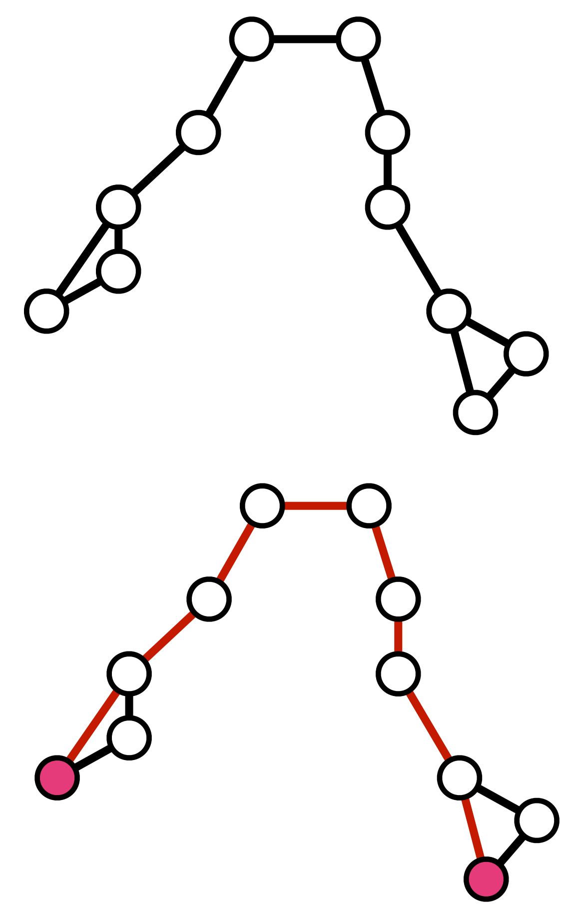
Manifold learning

ISOMAP

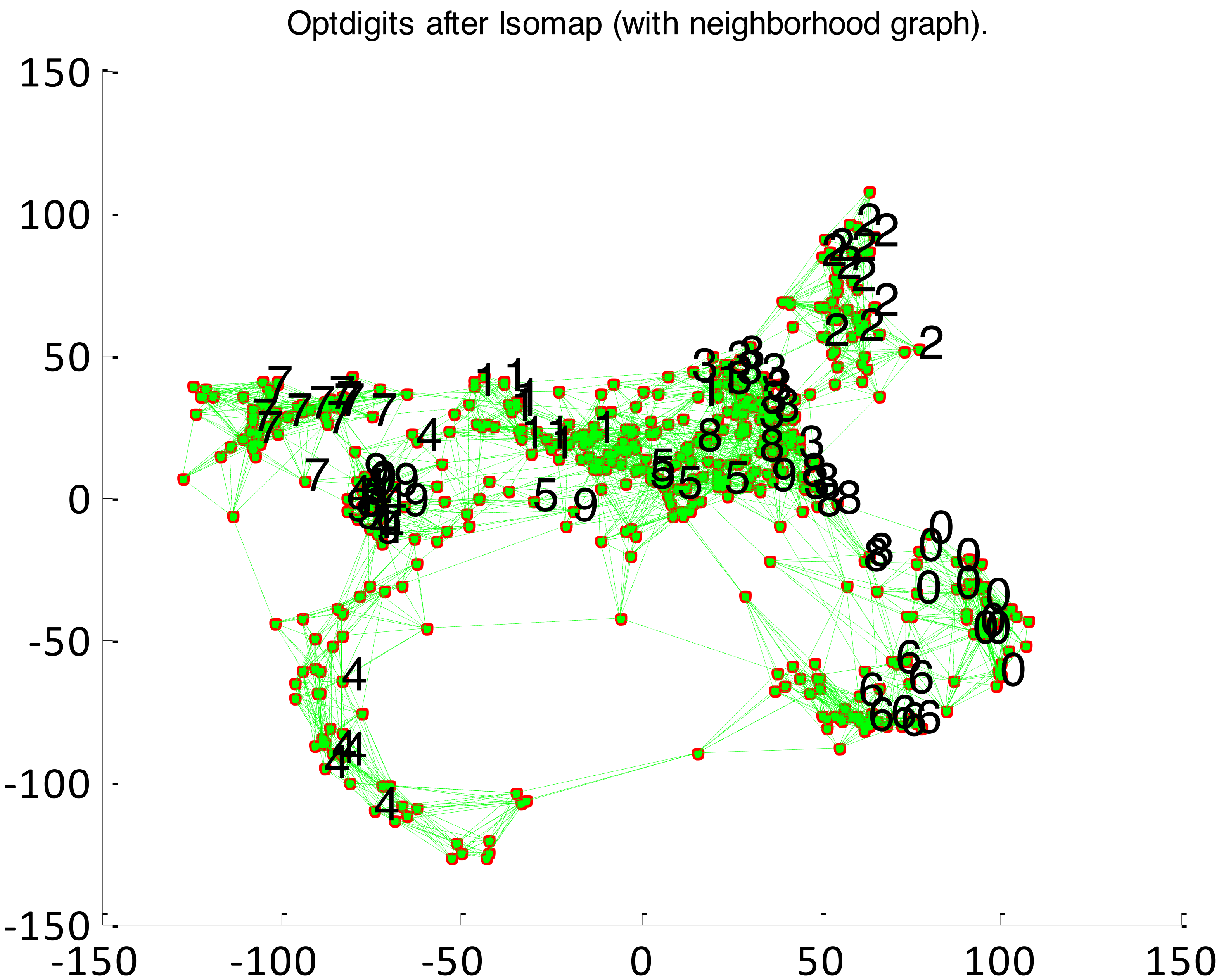
1. construct a neighborhood graph (kNN and ε-NN)

2. calculate distance matrix as the shortest path on the graph

3. apply MDS on the distance matrix



Manifold learning



from [Intro. ML]

Manifold learning

Local Linear Embedding (LLE):

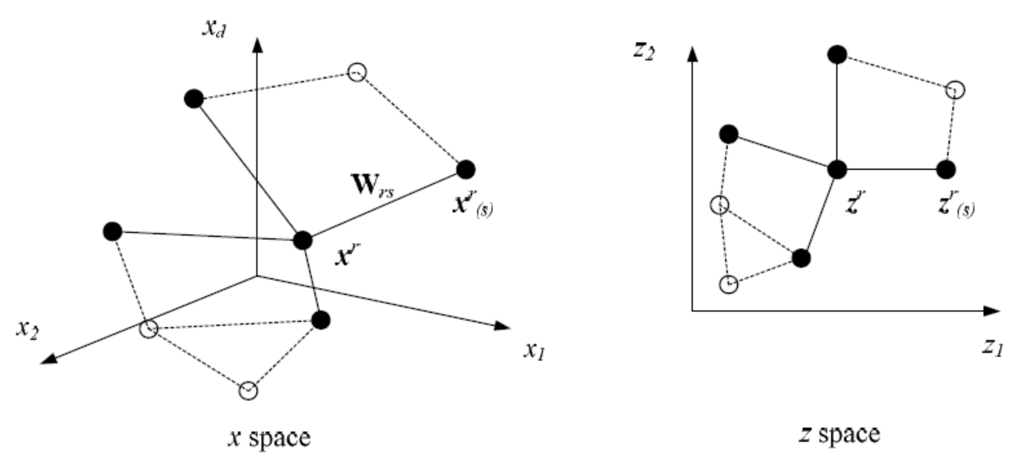
1. find neighbors for each instance

2. calculate a linear reconstruction for an instance

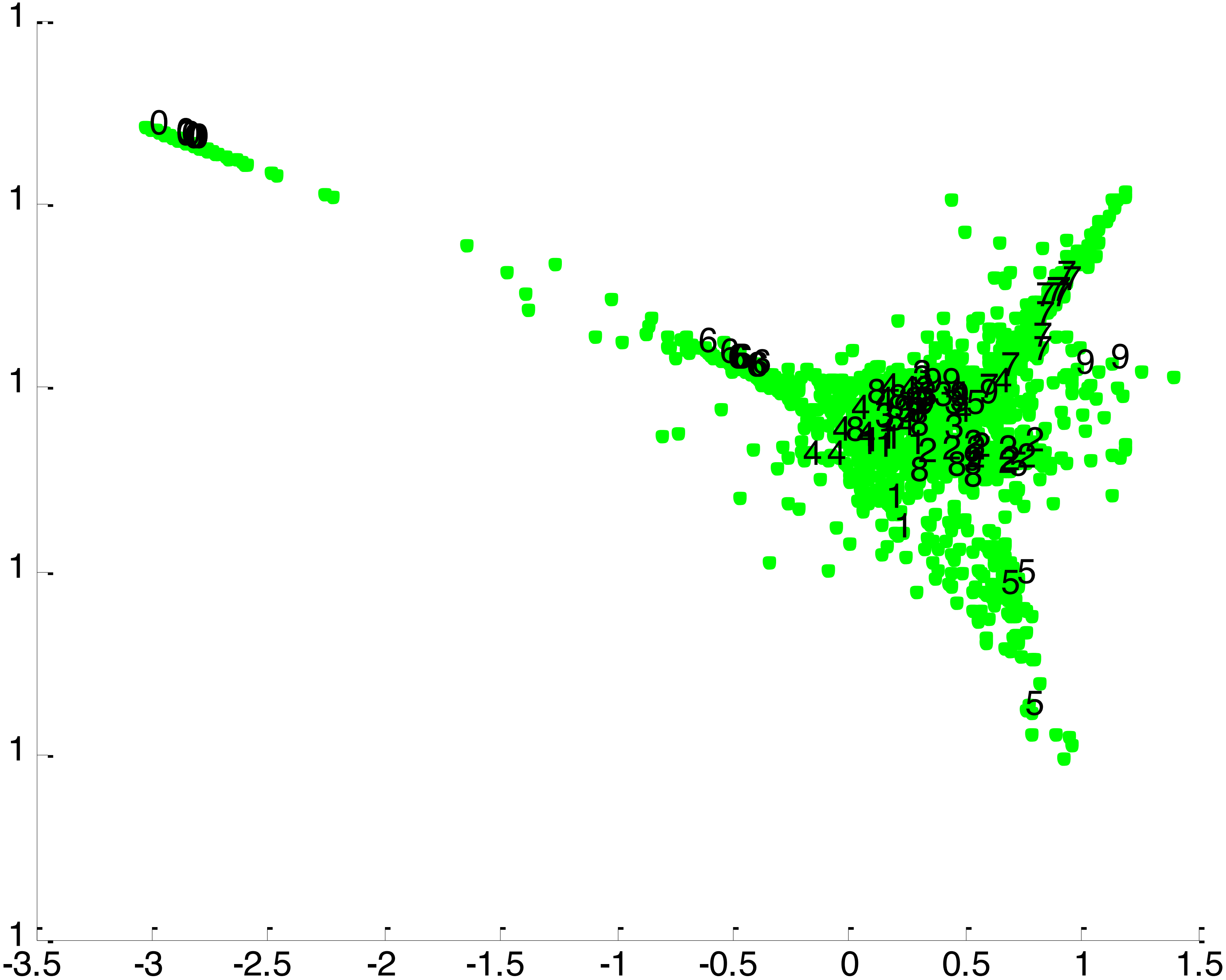
$$\sum_r \| \boldsymbol{x}^r - \sum_s \mathbf{W}_{rs} \boldsymbol{x}^s_{(r)} \|^2$$

3. find low dimensional instances preserving the reconstruction

$$\sum_r \| \boldsymbol{z}^r - \sum_s \mathbf{W}_{rs} \boldsymbol{z}^s \|^2$$



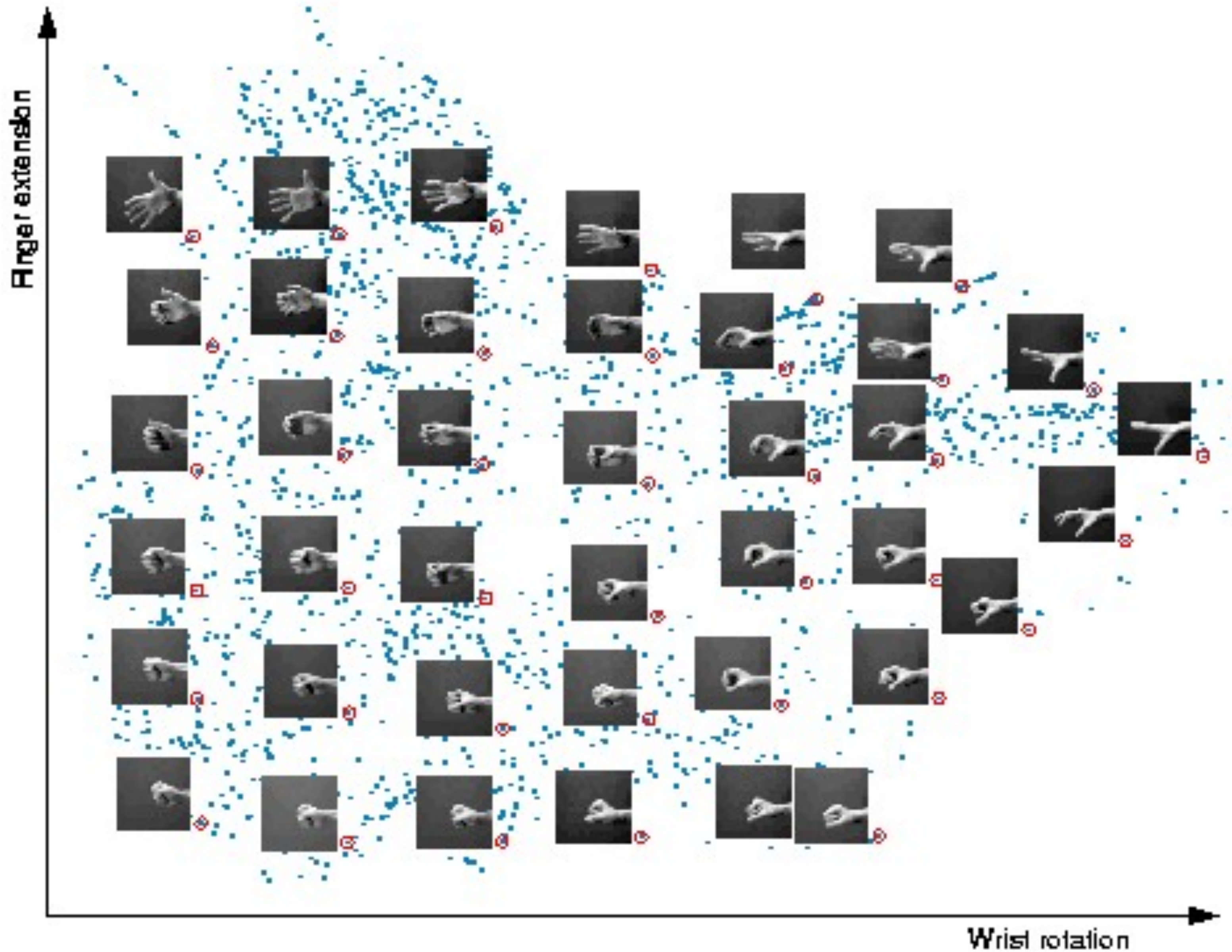
Manifold learning



from [Intro. ML]

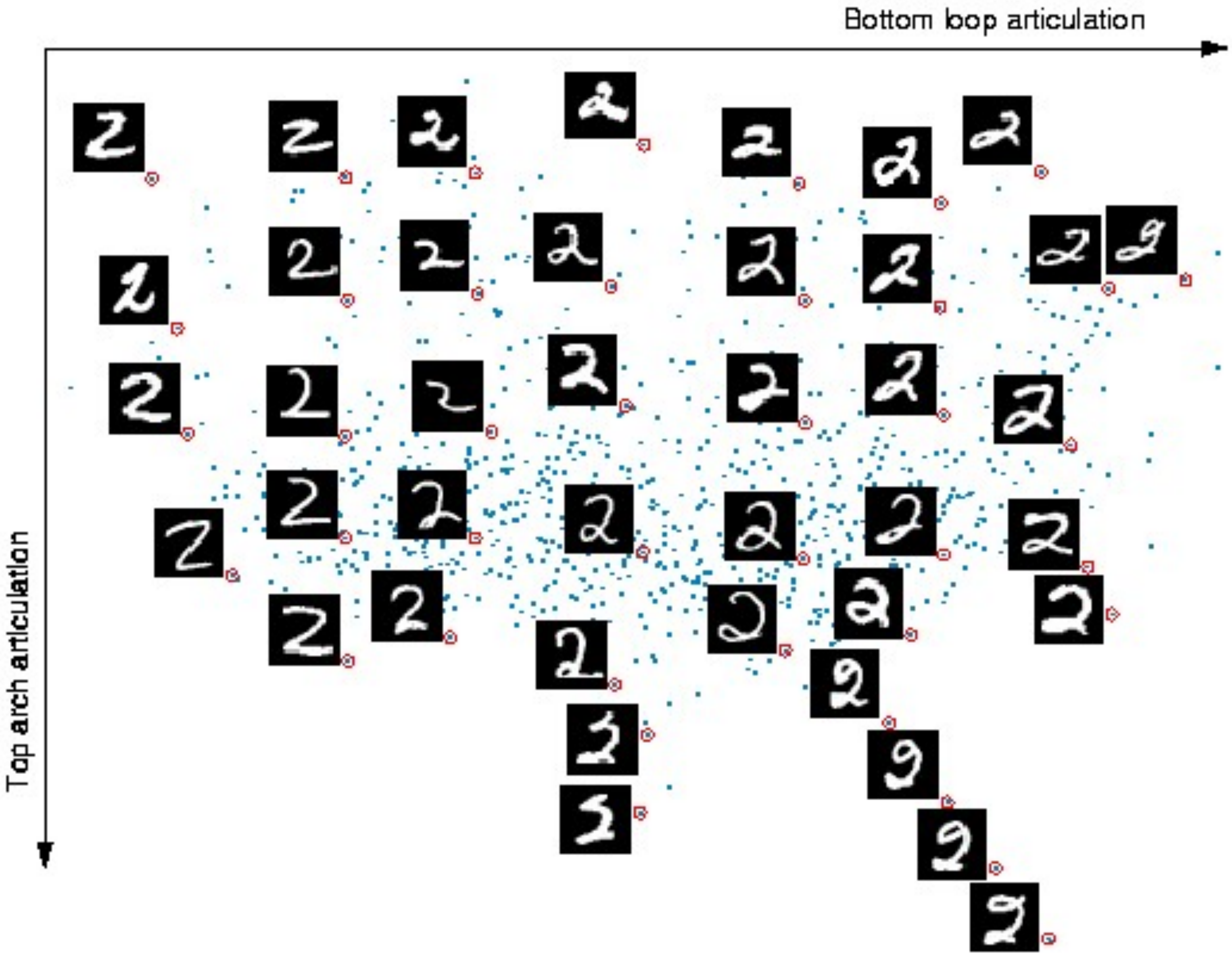
Manifold learning

more manifold learning examples



Manifold learning

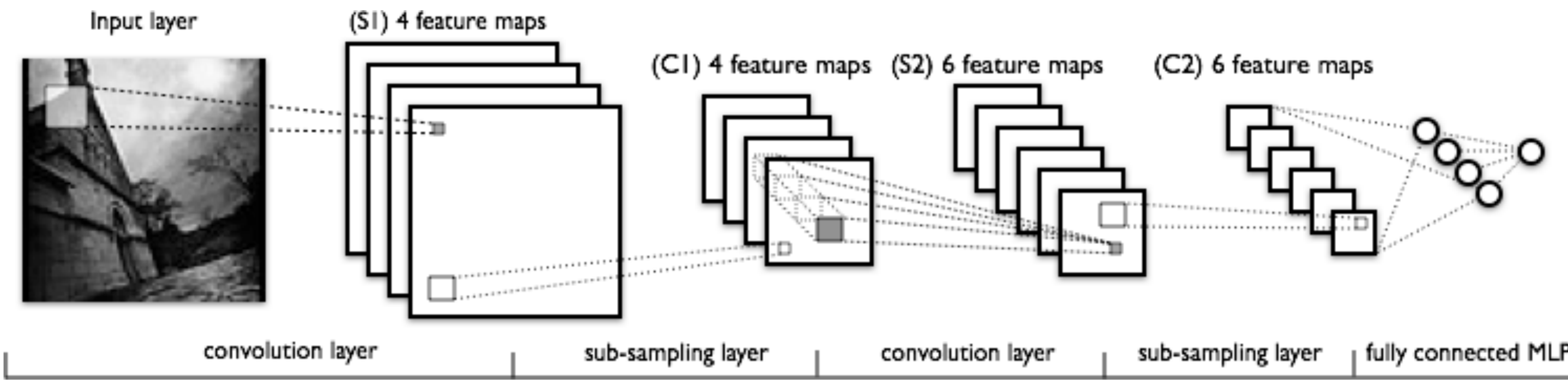
more manifold learning examples



Other feature extraction methods

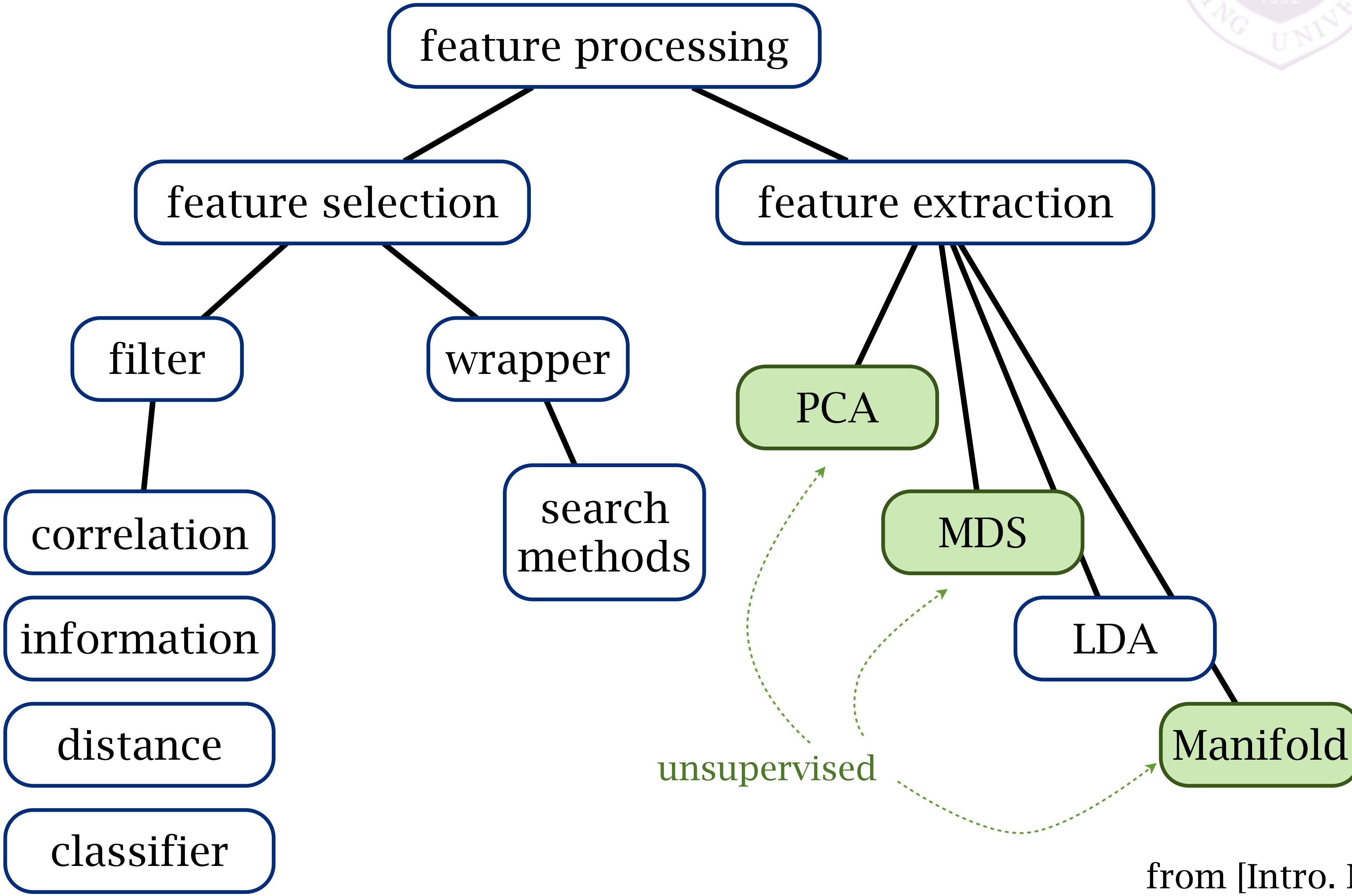
Most feature extractions are case specific

Convolutional Neural Networks (CNN/LeNet)
for general image feature extraction



[image from http://deeplearning.net/tutorial/lenet.html]

A summary of approaches



from [Intro. ML]