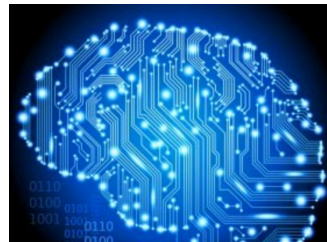




# Lecture 17: Learning 6

[http://cs.nju.edu.cn/yuy/course\\_ai16.ashx](http://cs.nju.edu.cn/yuy/course_ai16.ashx)



# Previously...



## Learning

Decision tree learning

Neural networks

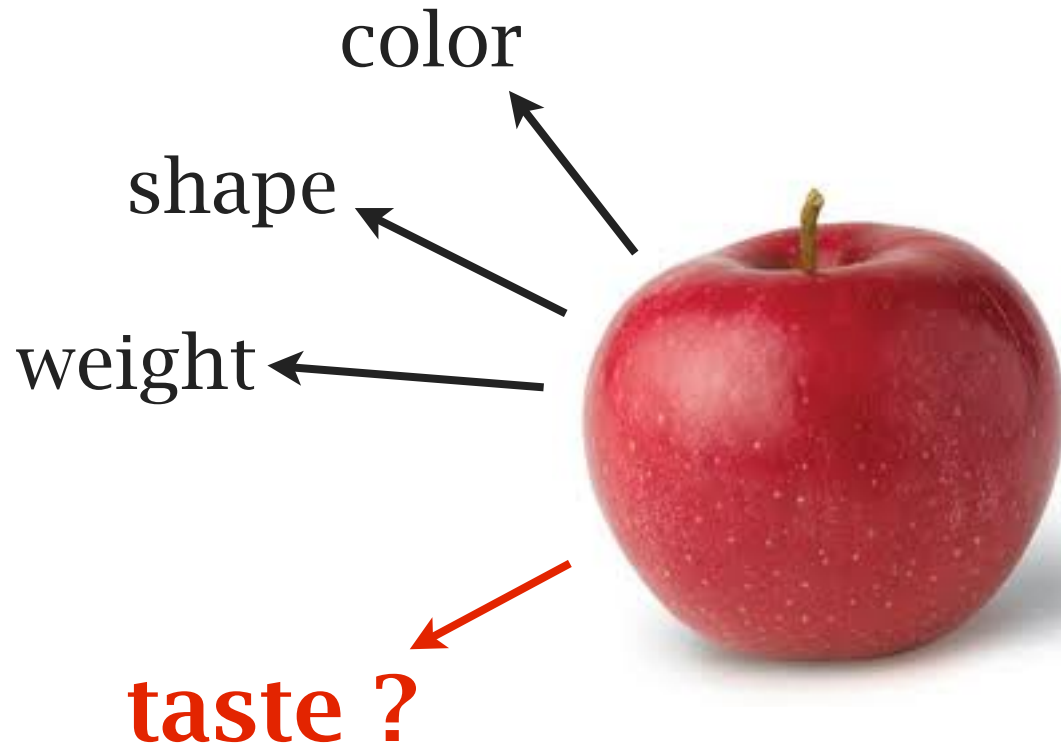
Why we can learn

Linear models

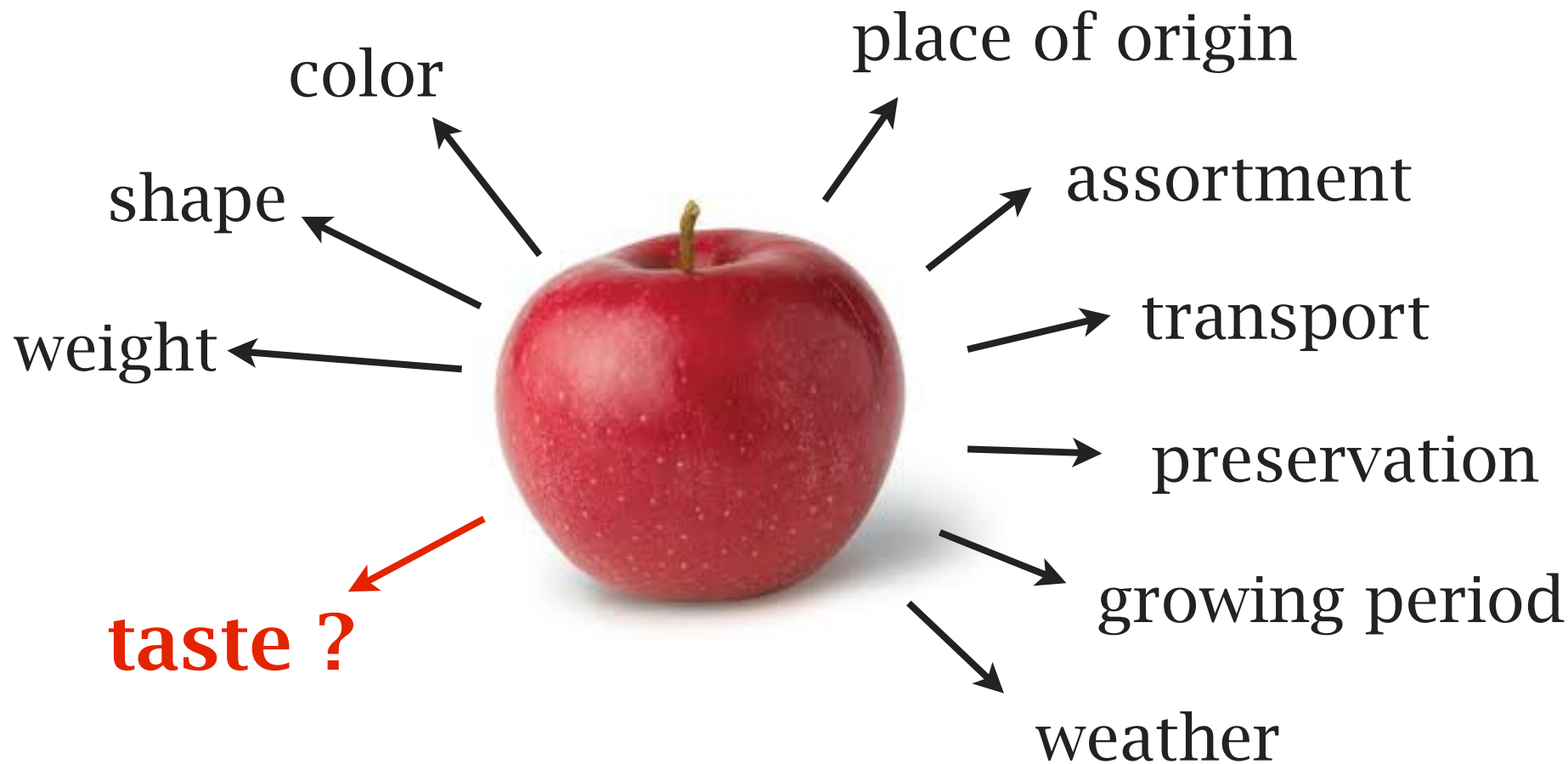
Nearest neighbor classifier

Native Bayes classifier

# The importance of features



# The importance of features

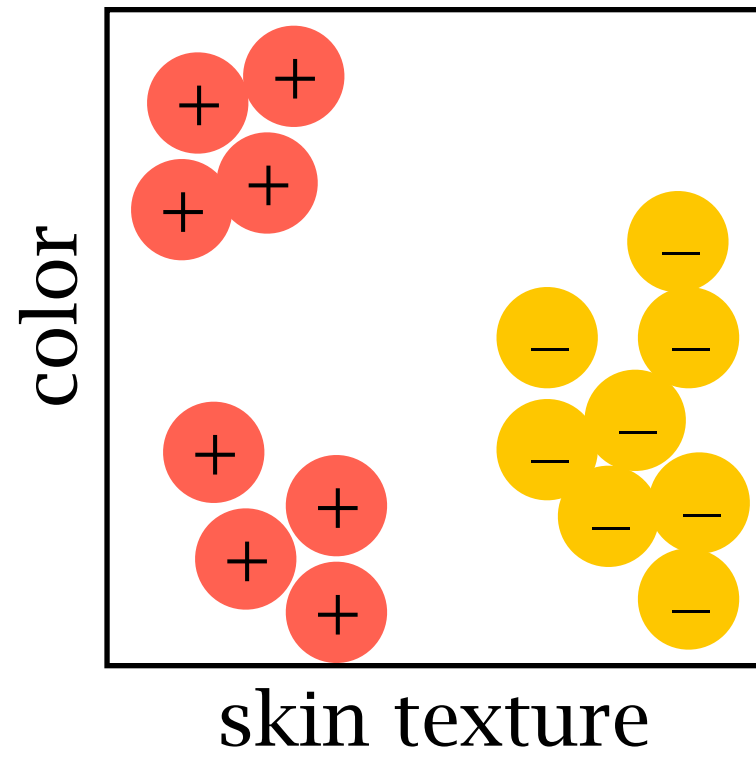
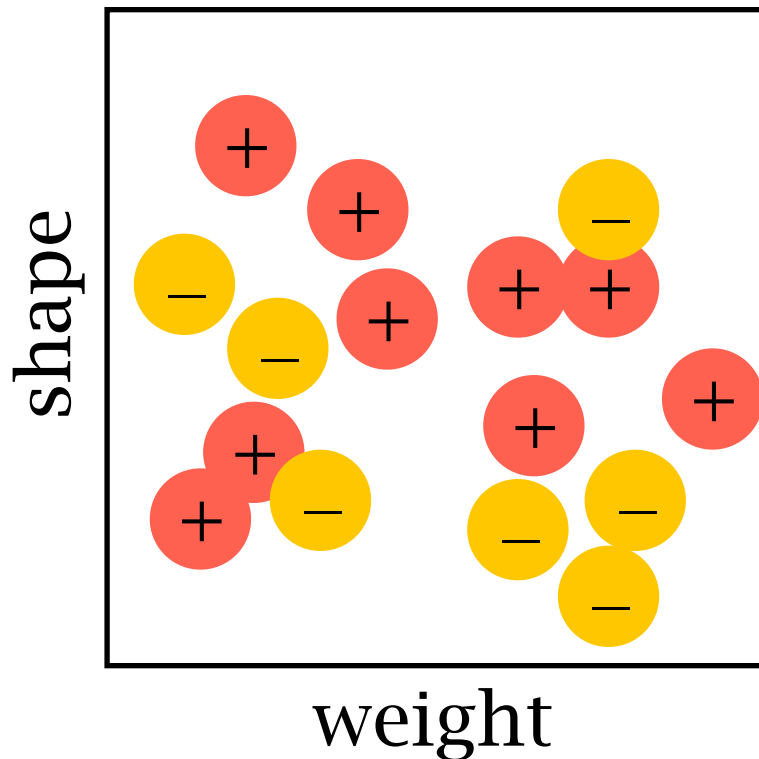
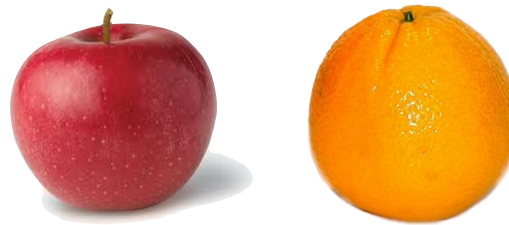


# The importance of features



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

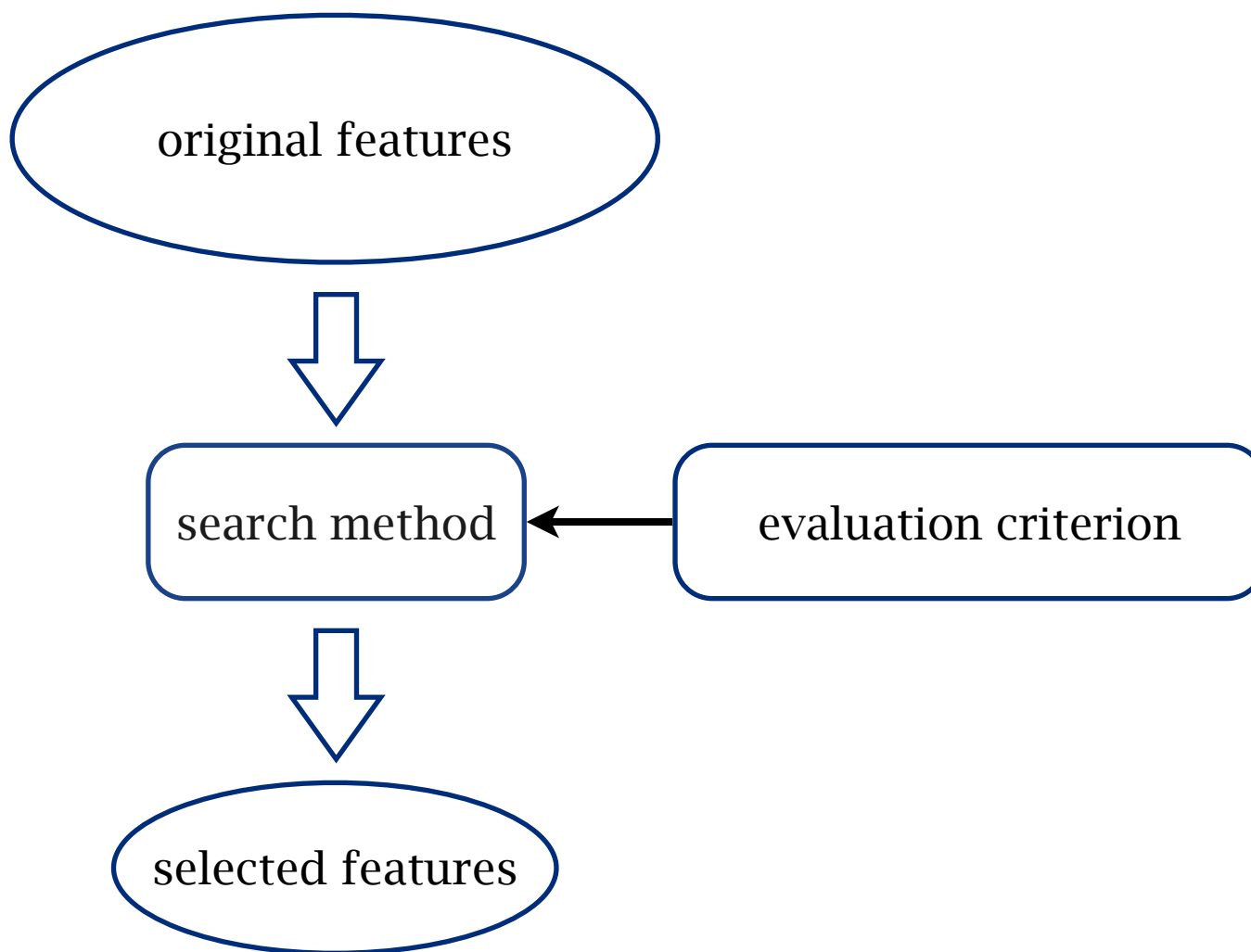
Reduce the time/space complexity of learning

Improve the interpretability

Better data visualization

Saving the cost of observing features

# Feature selection

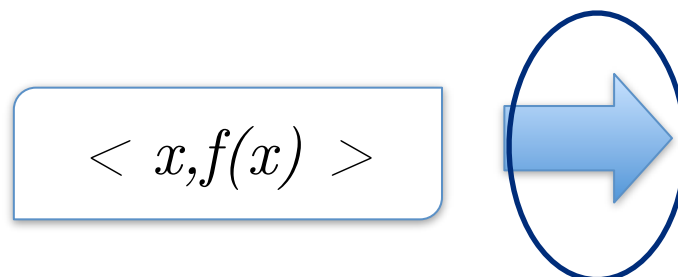




# Evaluation criteria



classifier independent



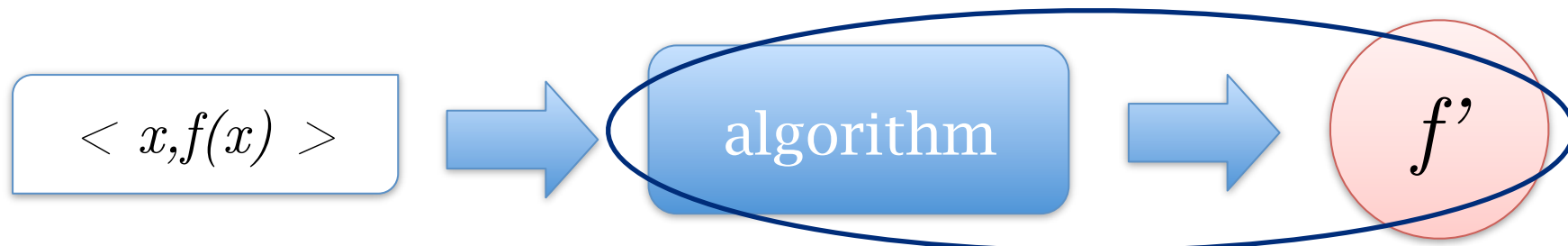
dependency based criteria

information based criteria

distance based criteria

classifier internal weighting

classifier dependent



# Dependency based criteria

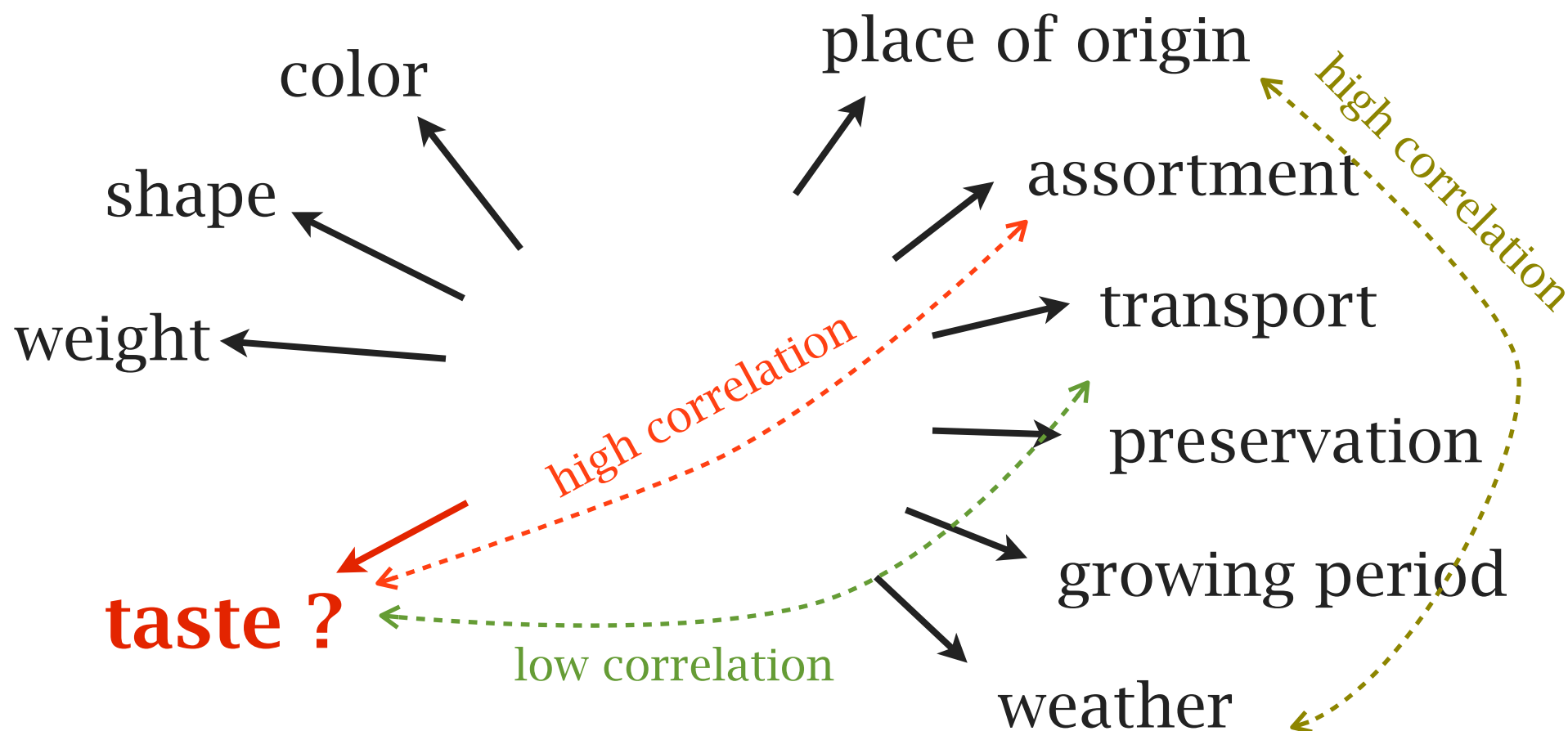


## How a feature set is related with the class

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:

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

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

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

# A simple forward search



sequentially add the next best feature

- 1:  $F$  = original feature sets,  $C$  is the class label
- 2:  $S = \emptyset$
- 3: **loop**
- 4:      $a$  = the best correlated/informative feature in  $F$
- 5:      $v$  = the correlation/IG of  $a$
- 6:     **if**  $v < \theta$  **then**
- 7:         **break**
- 8:     **end if**
- 9:      $F = F / \{a\}$
- 10:      $S = S \cup \{a\}$
- 11: **end loop**
- 12: **return**  $S$

# A simple forward search



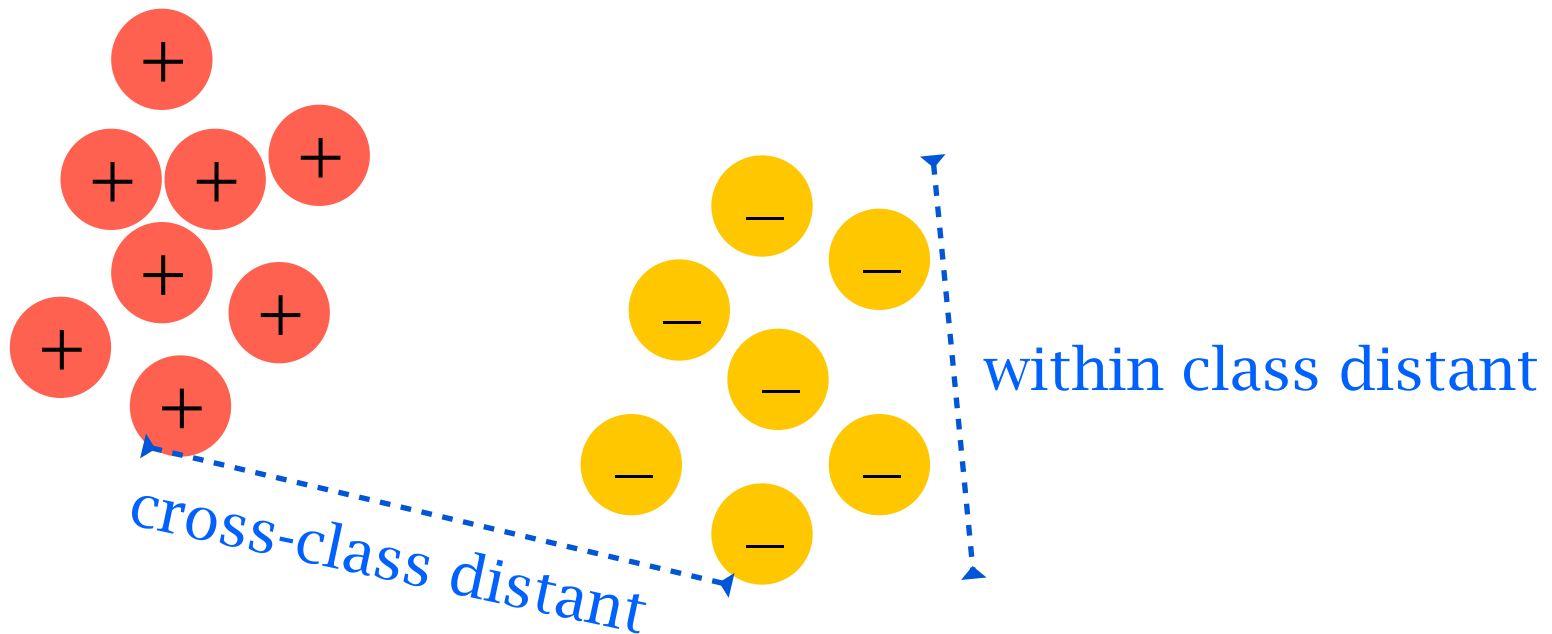
```
1:  $F$  = original feature sets,  $C$  is the class label
2:  $S = \emptyset$ 
3: loop
4:    $a$  = the best correlated/informative feature in  $F$ 
5:    $v$  = the correlation/IG of  $a$ 
6:   if  $v < \theta$  then
7:     break
8:   end if
9:    $F = F / \{a\}$ 
10:   $S = S \cup \{a\}$ 
11:  for  $a' \in F$  do
12:     $v'$  = the correlation/IG of  $a'$  to  $a$ 
13:    if  $v' > \alpha \cdot v$  then  $F = F / \{a'\}$ 
14:    end if
15:  end for
16: end loop
17: 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

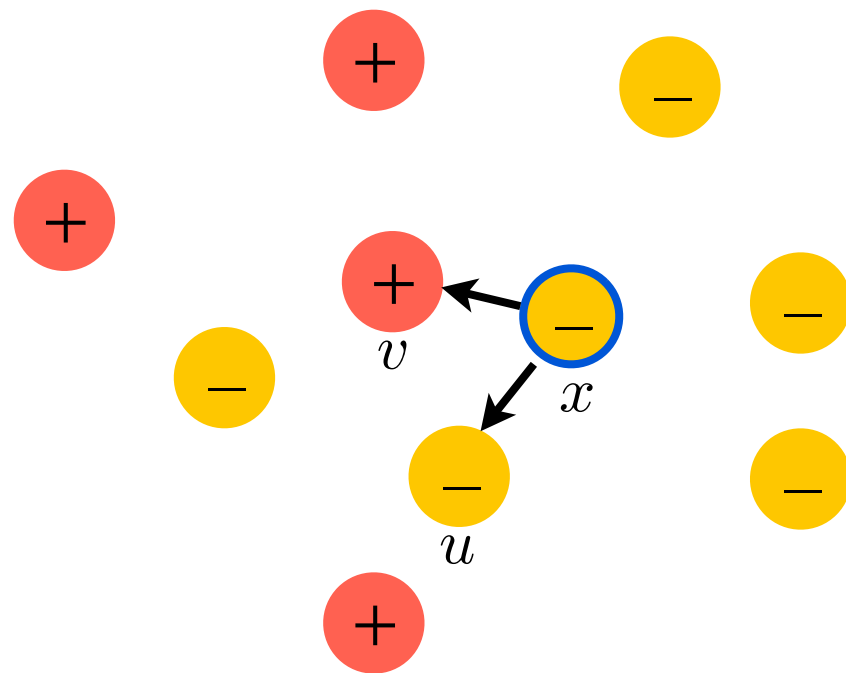
# Distance based criteria



## Relief: feature weighting based on distance

$$w = 0$$

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.  $w = w - |x - u| + |x - v|$
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

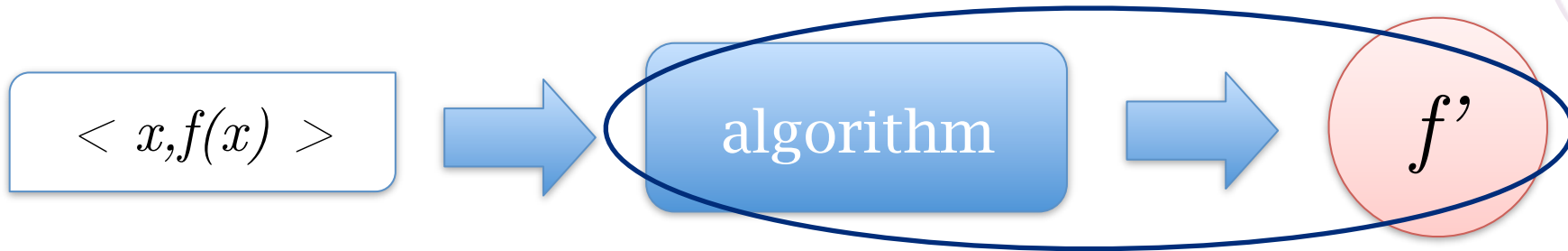
**linear model:** a natural feature weighting

**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**

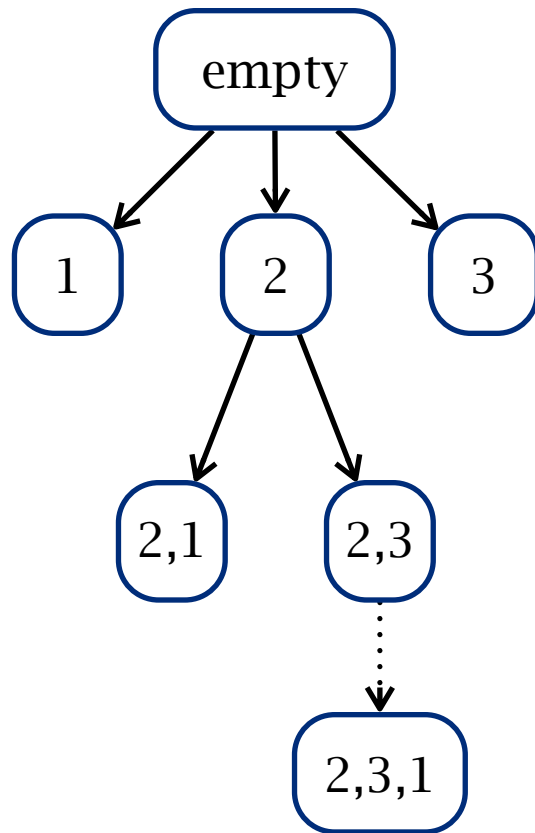
$F = F/\{ma\}$

perf-so-far =  $mv$

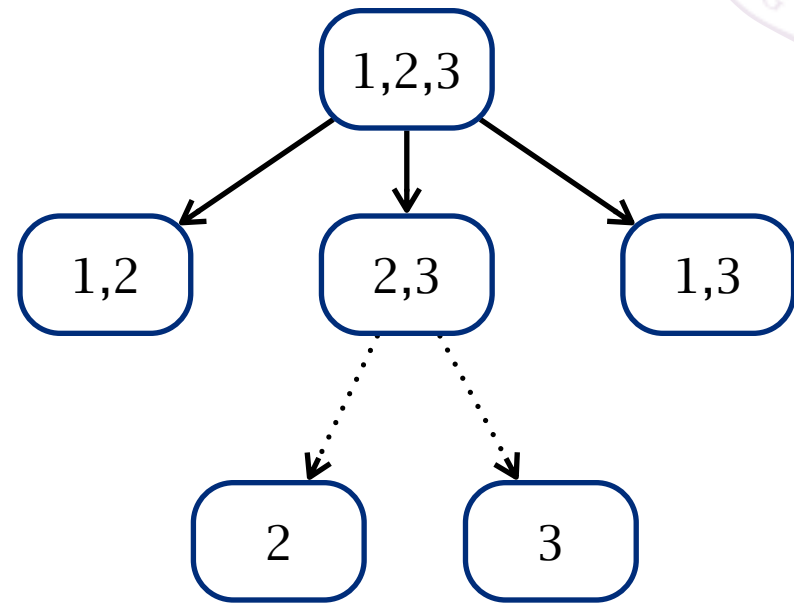
**end loop**

**return**  $S$

# Classifier dependent feature selection



forward  
faster

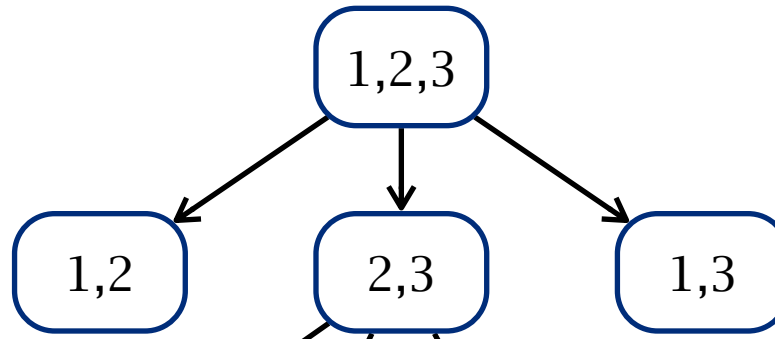


backward  
more accurate

# Classifier dependent feature selection



random init



backward

forward



backward



...

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:

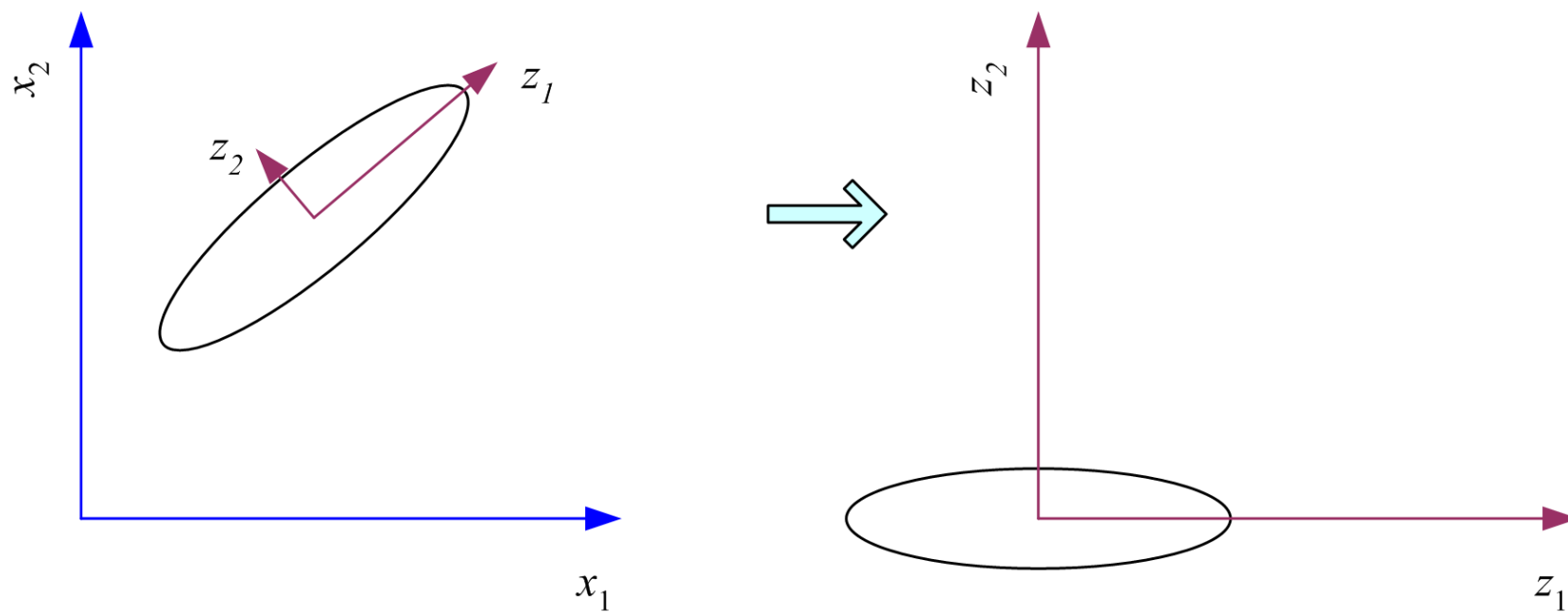
1	2	3	4	0	1
2	3	4	0	1	2
3	4	0	0	4	1
3	1	0	0	2	2
2	0	1	2	3	3
3	3	4	4	1	0

# Linear methods



## Principal components analysis (PCA)

rotate the data to align the directions of the variance

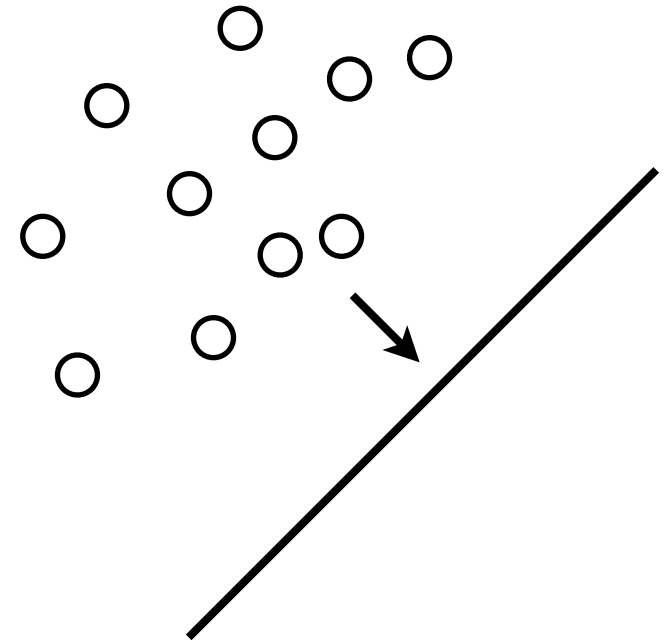


# Linear methods



Principal components analysis (PCA)

the first dimension = the largest variance direction





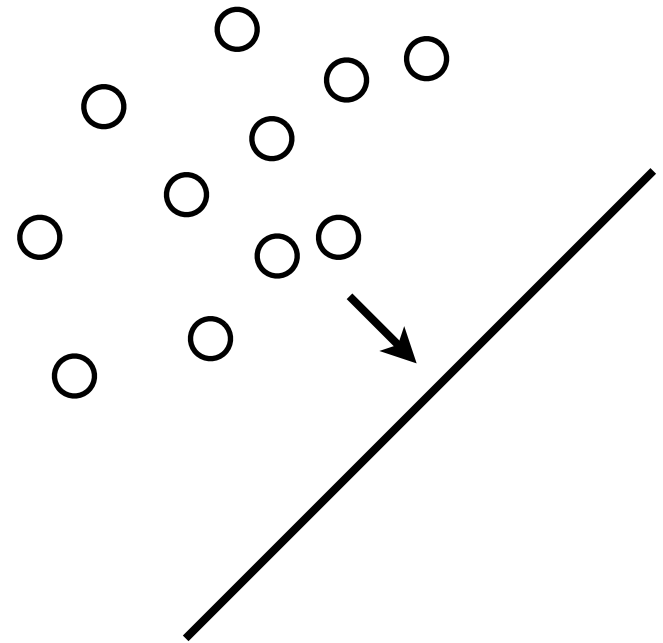
# Linear methods



## Principal components analysis (PCA)

the first dimension = the largest variance direction

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



# Linear methods

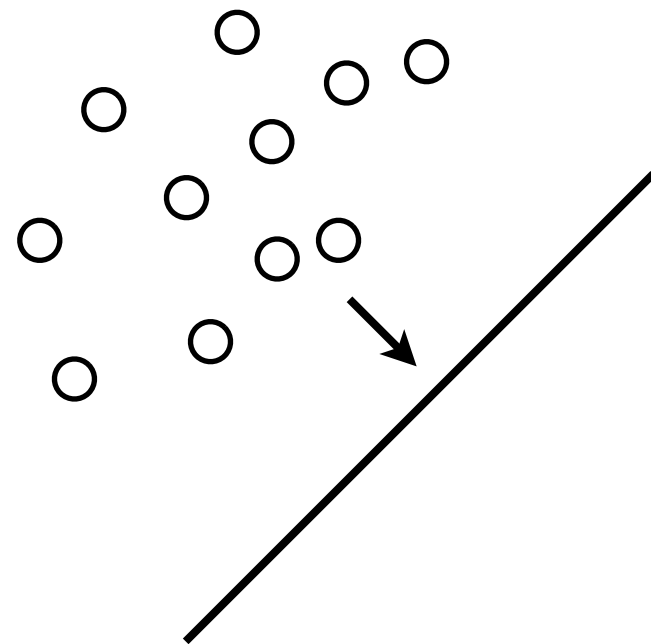


## Principal components analysis (PCA)

the first dimension = the largest variance direction

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

$$\text{Var}(z_1) = \mathbf{w}_1^T \Sigma \mathbf{w}_1$$



# Linear methods



## Principal components analysis (PCA)

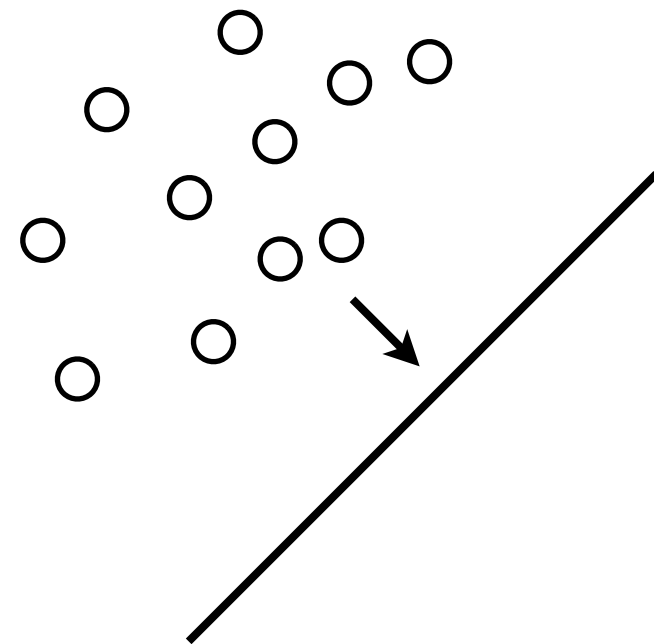
the first dimension = the largest variance direction

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

$$\text{Var}(z_1) = \mathbf{w}_1^T \Sigma \mathbf{w}_1$$

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

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



# Linear methods



## Principal components analysis (PCA)

the first dimension = the largest variance direction

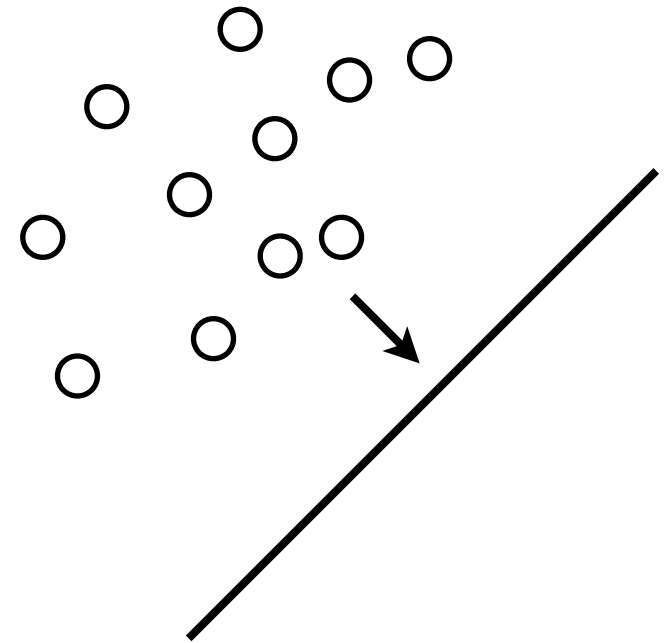
$$z = \mathbf{w}^T \mathbf{x}$$

$$\text{Var}(z_1) = \mathbf{w}_1^T \Sigma \mathbf{w}_1$$

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

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

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



# Linear methods



## Principal components analysis (PCA)

the first dimension = the largest variance direction

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

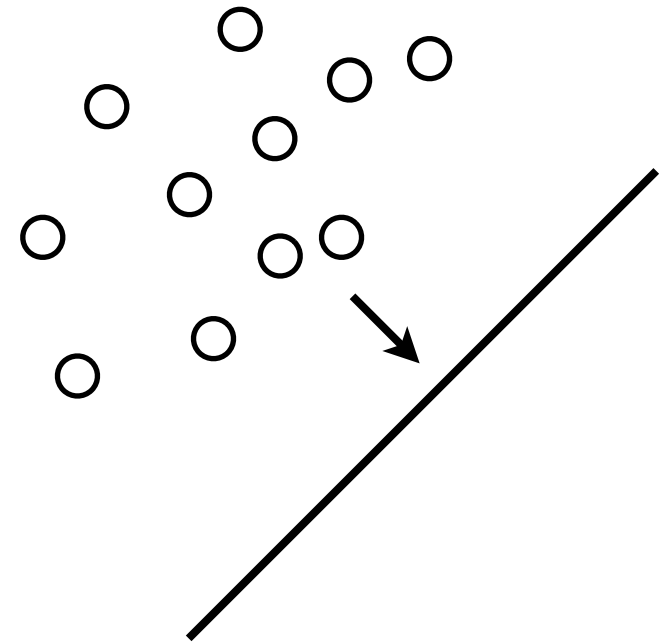
$$\text{Var}(z_1) = \mathbf{w}_1^T \Sigma \mathbf{w}_1$$

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

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

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

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



# Linear methods



## Principal components analysis (PCA)

the first dimension = the largest variance direction

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

$$\text{Var}(z_1) = \mathbf{w}_1^T \Sigma \mathbf{w}_1$$

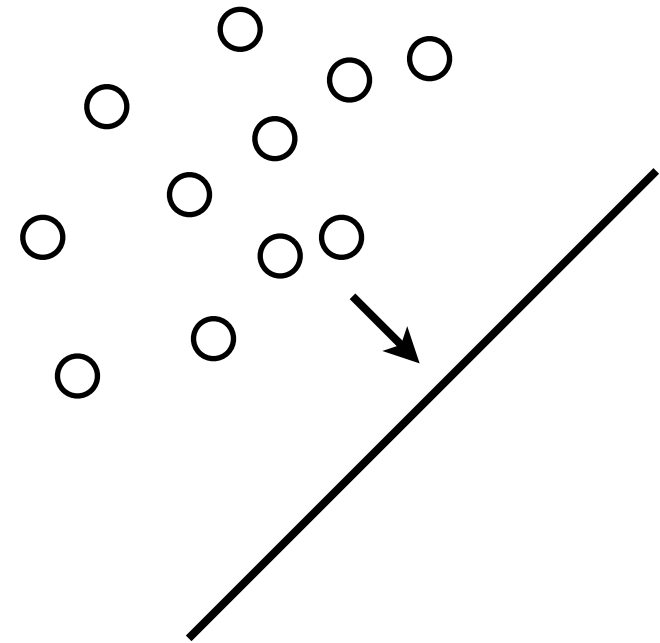
find a unit  $\mathbf{w}$  to maximize the variance

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

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

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

$\mathbf{w}$  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

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



# Linear methods



## Principal components analysis (PCA)

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

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

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

# Linear methods



## Principal components analysis (PCA)

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

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

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

$$\beta = 0 \quad \Sigma \mathbf{w}_2 = \alpha \mathbf{w}_2$$

# Linear methods



## Principal components analysis (PCA)

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

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

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

$$\beta = 0 \quad \Sigma \mathbf{w}_2 = \alpha \mathbf{w}_2$$

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

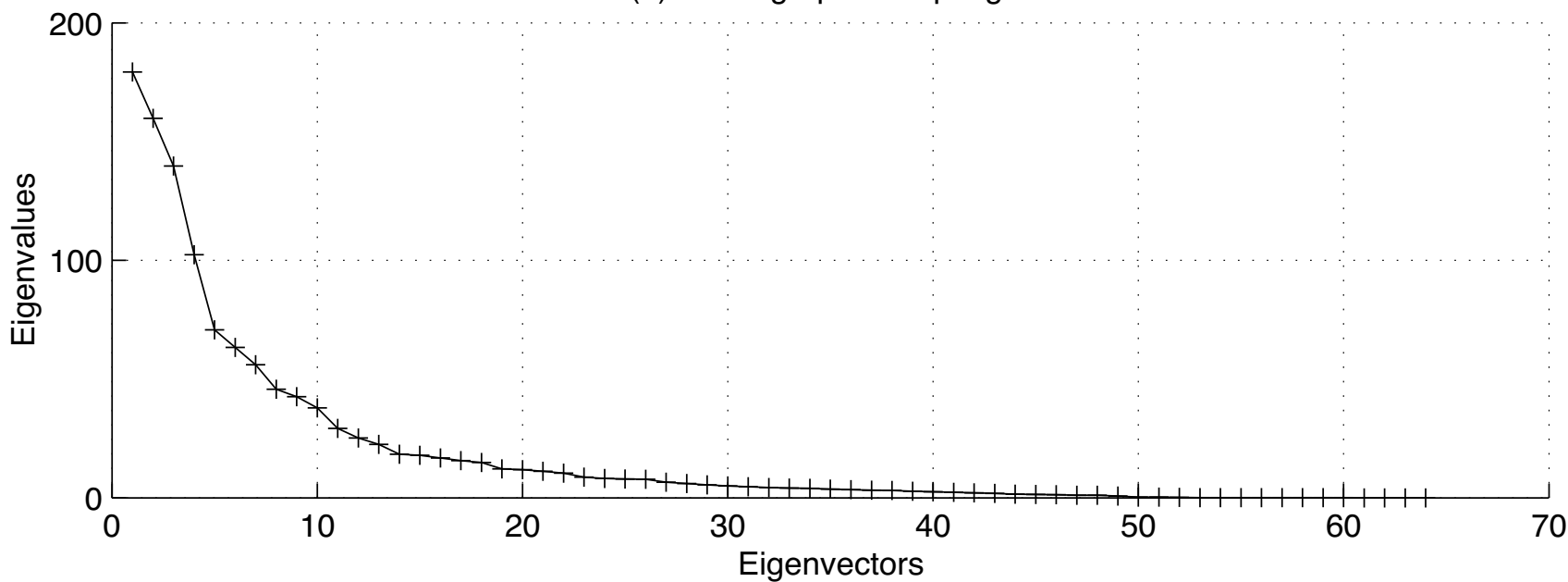
# Linear methods



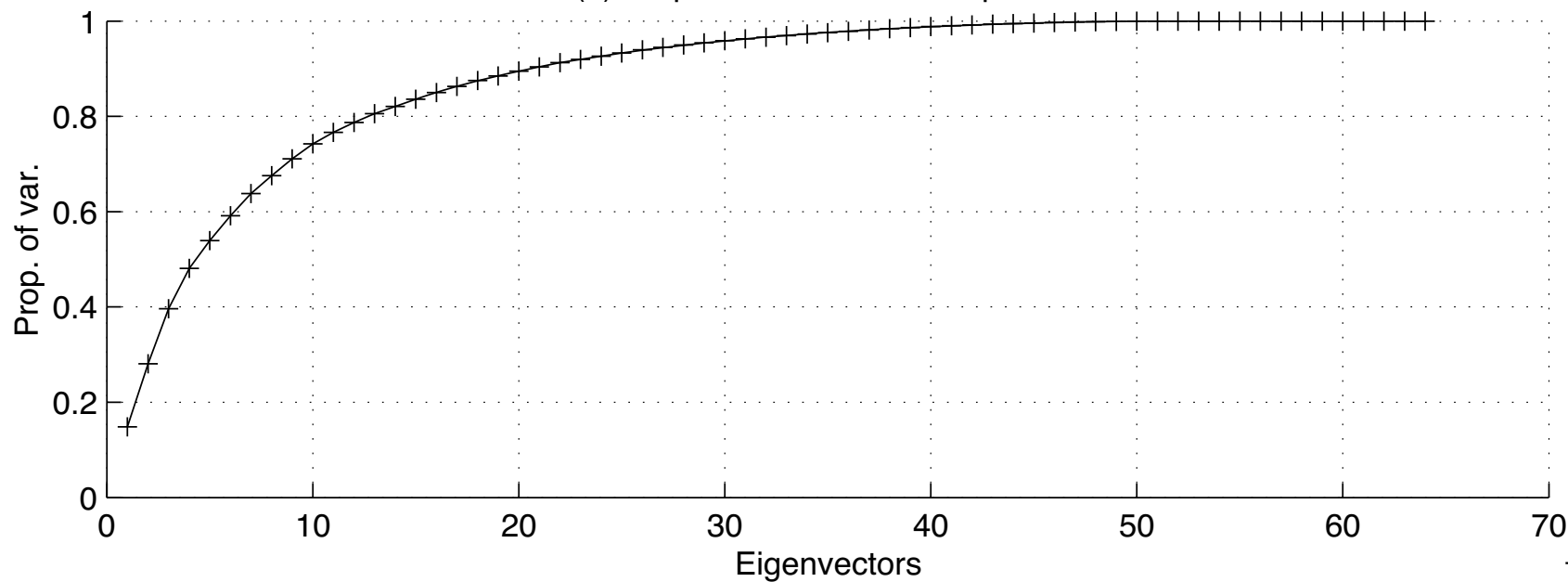
# Linear methods



(a) Scree graph for Optdigits



(b) Proportion of variance explained



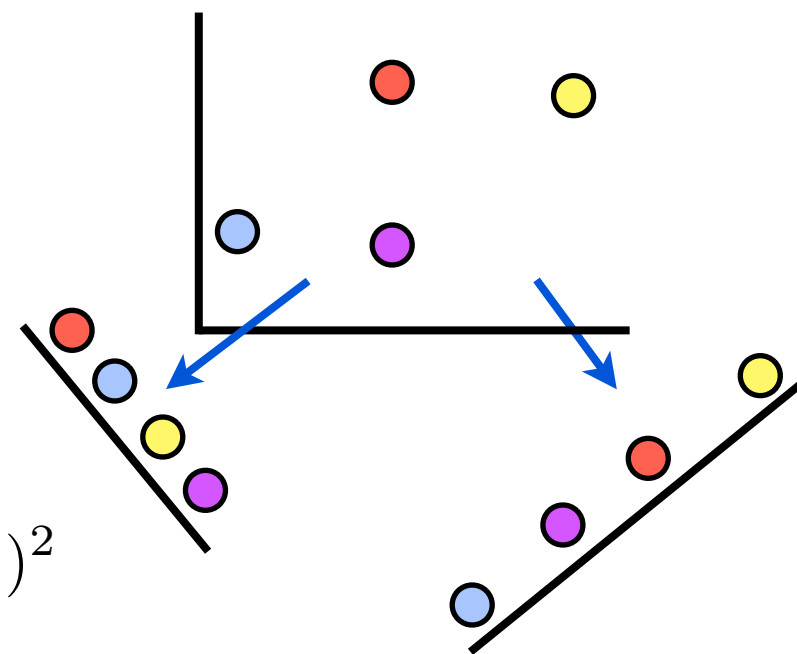
# Linear methods



## Multidimensional Scaling (MDS)

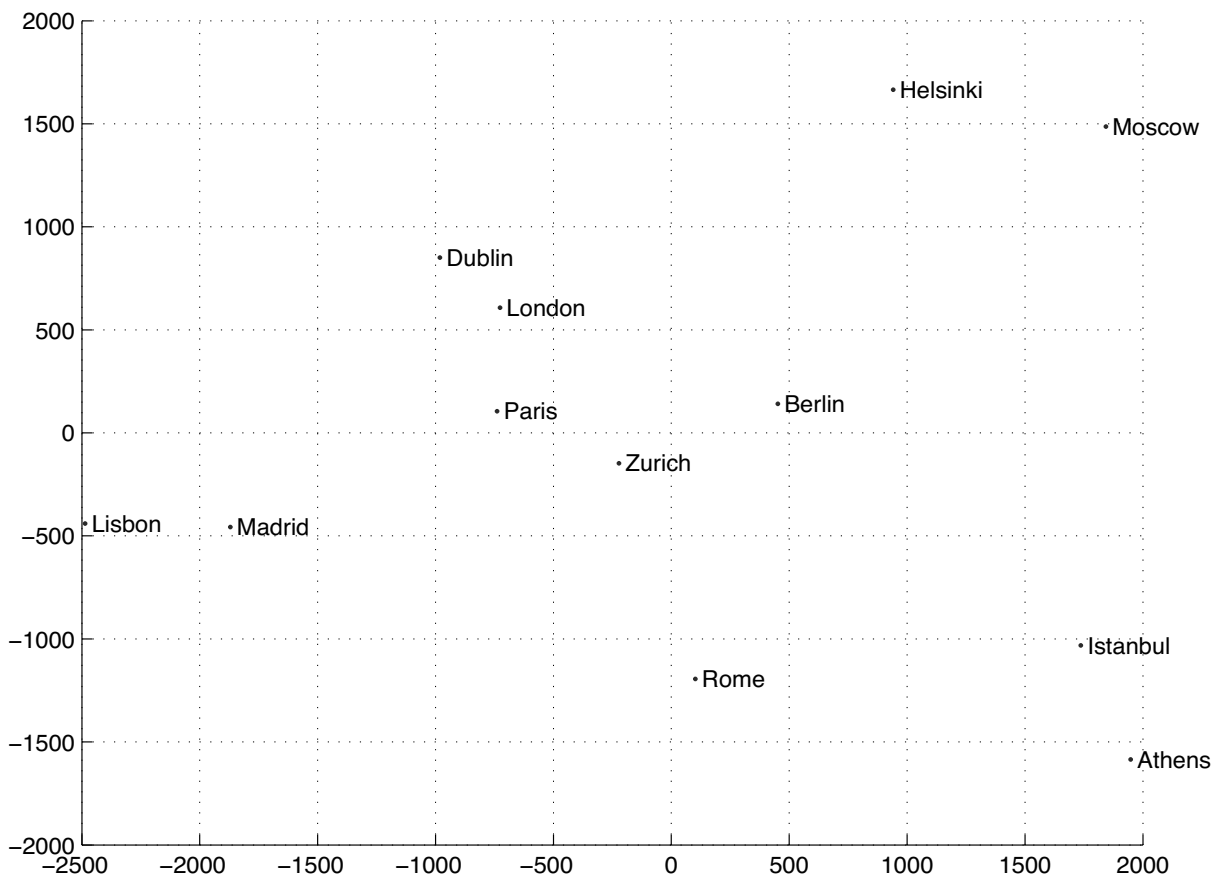
keep the distance into a lower dimensional space

for linear transformation,  
 $W$  is an  $n \times k$  matrix



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

# Linear methods



# Linear methods



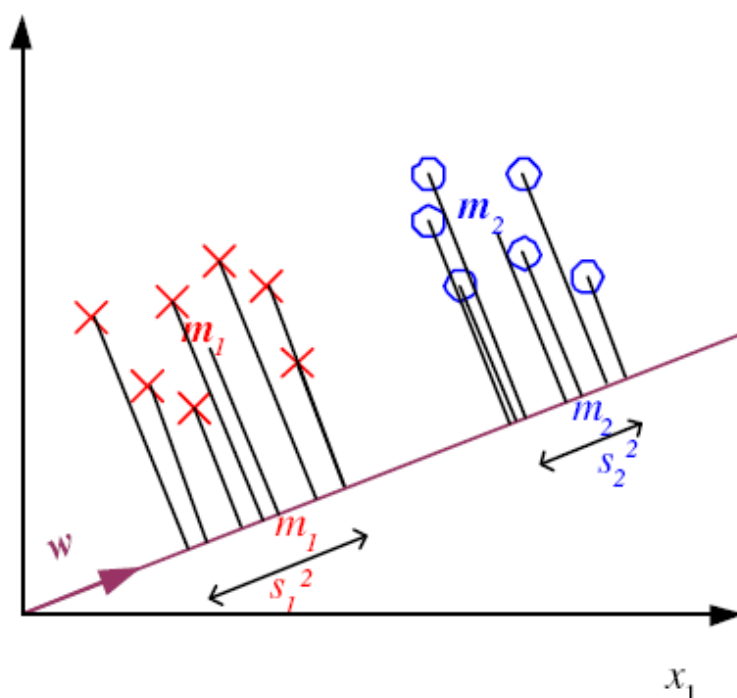
## Linear Discriminant Analysis (LDA)

find a direction such that the two classes are well separated

$$Z = \mathbf{w}^T \mathbf{x}$$

$m$  be the mean of a class

$s^2$  be the variance of a class



maximize the criterion

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



# Linear methods

## Linear Discriminant Analysis (LDA)



# Linear methods



## Linear Discriminant Analysis (LDA)

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

# Linear methods



## Linear Discriminant Analysis (LDA)

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

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

# Linear methods



## Linear Discriminant Analysis (LDA)

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

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

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

# Linear methods



## Linear Discriminant Analysis (LDA)

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

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

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

The objective becomes:

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

# Linear methods



## Linear Discriminant Analysis (LDA)

The objective becomes:

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

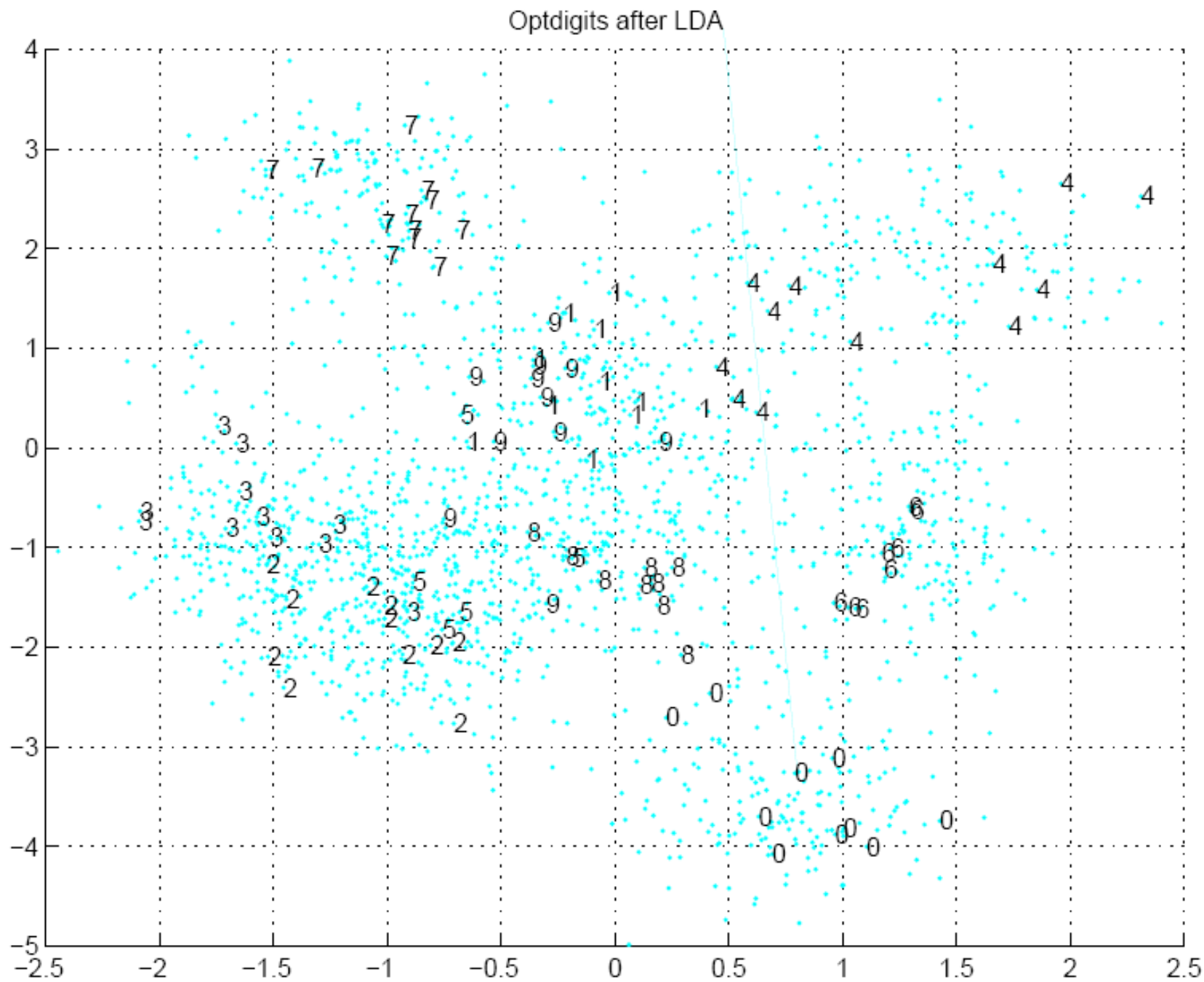
$$\frac{\mathbf{w}^T (\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \left( 2(\mathbf{m}_1 - \mathbf{m}_2) - \frac{\mathbf{w}^T (\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \mathbf{S}_W \mathbf{w} \right) = 0$$

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

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

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

# Linear methods



from [Intro. ML]

# Example: Face recognition

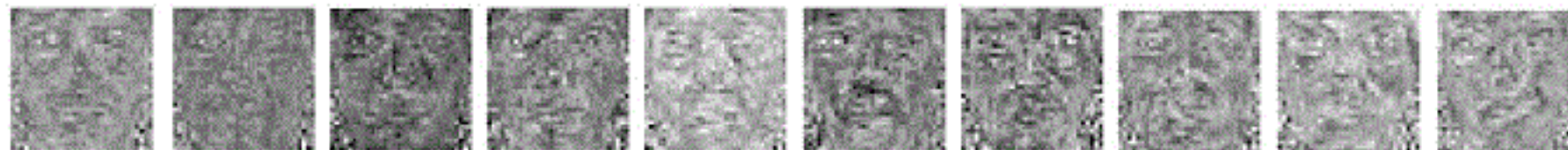


PCA and LDA are commonly used to extract features for face recognition.

Basis of eigenface (PCA):



Basis of Fisherface (LDA):





# Manifold learning



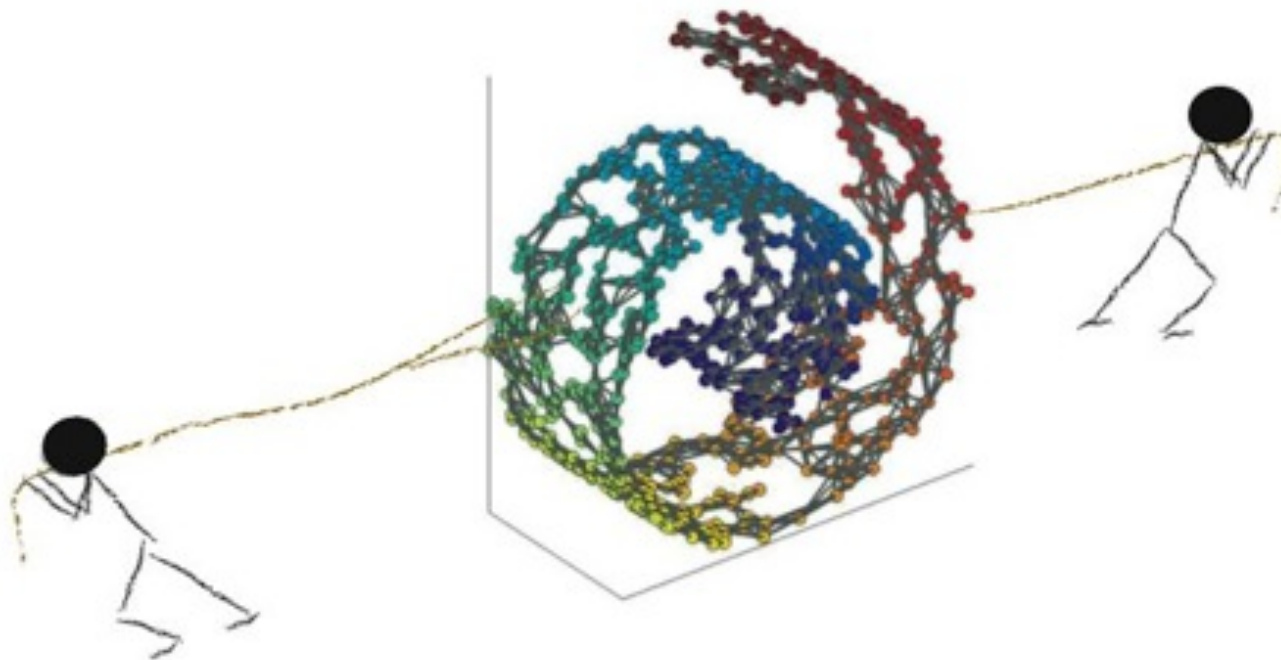


# Manifold learning



A low intrinsic dimensional data embedded in a high dimensional space

cause a bad distance measure

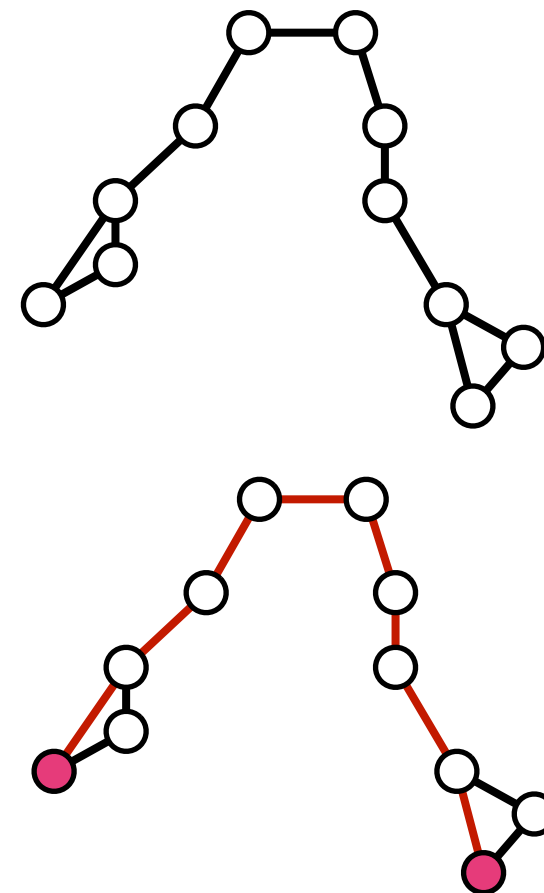


# Manifold learning



## ISOMAP

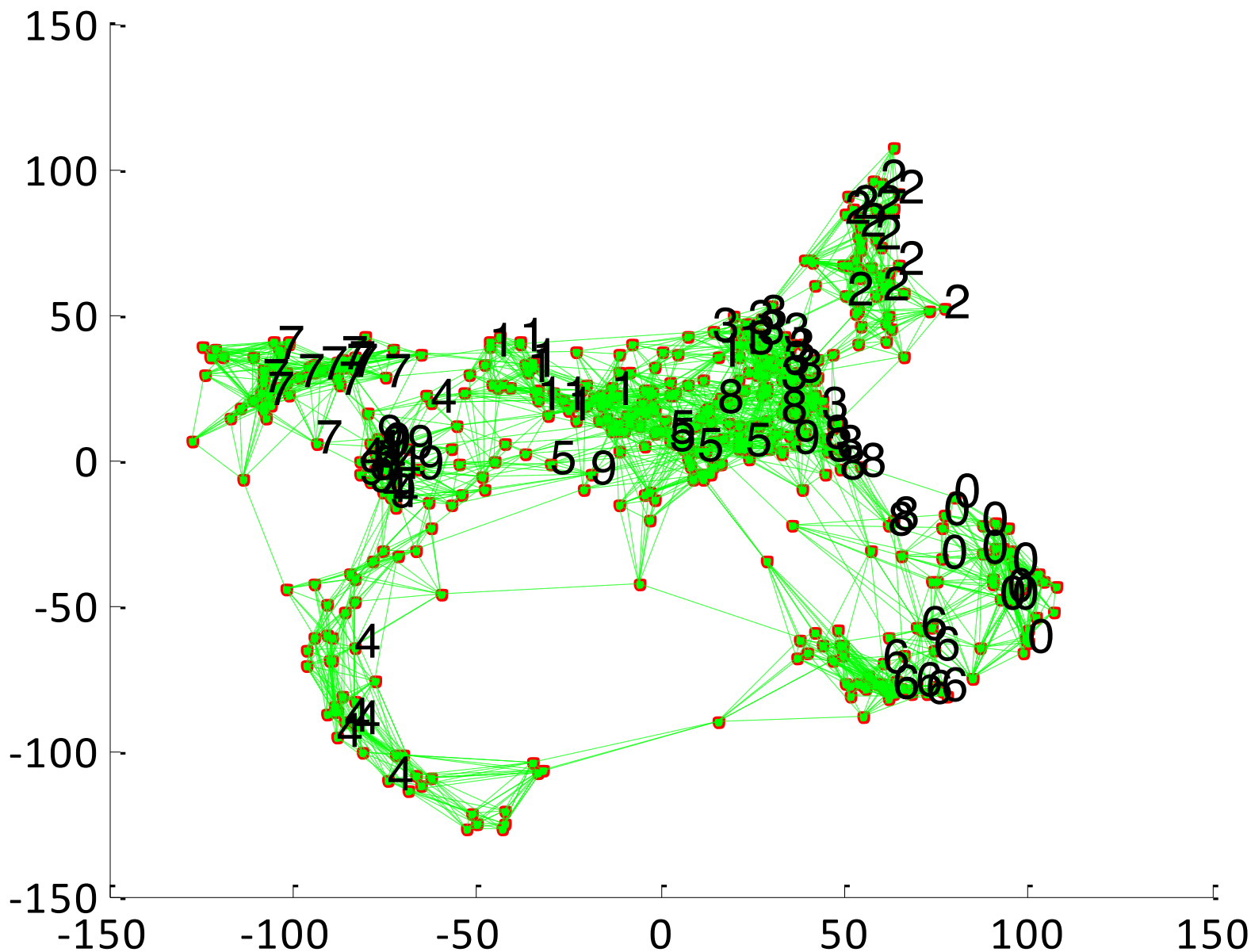
1. construct a neighborhood graph (kNN and  $\epsilon$ -NN)
2. calculate distance matrix as the shortest path on the graph
3. apply MDS on the distance matrix



# Manifold learning



Optdigits after Isomap (with neighborhood graph).



# Manifold learning



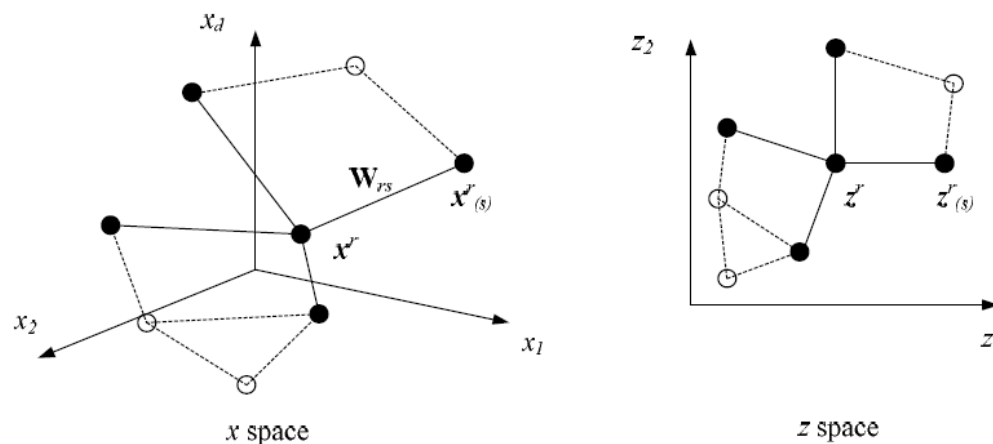
Local Linear Embedding (LLE):

1. find neighbors for each instance
2. calculate a linear reconstruction for an instance

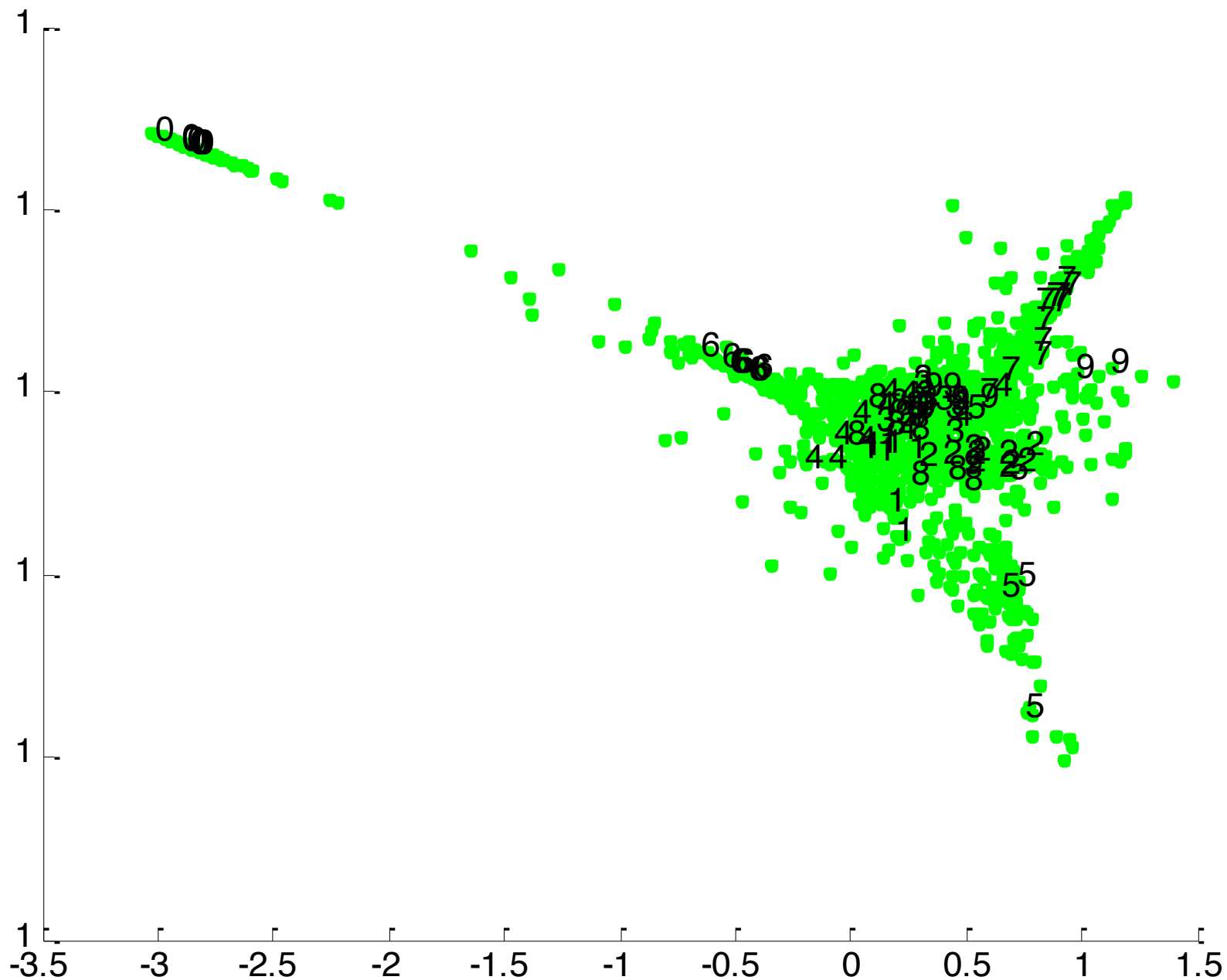
$$\sum_r \left\| \mathbf{x}^r - \sum_s \mathbf{W}_{rs} \mathbf{x}^{(s)} \right\|^2$$

3. find low dimensional instances preserving the reconstruction

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



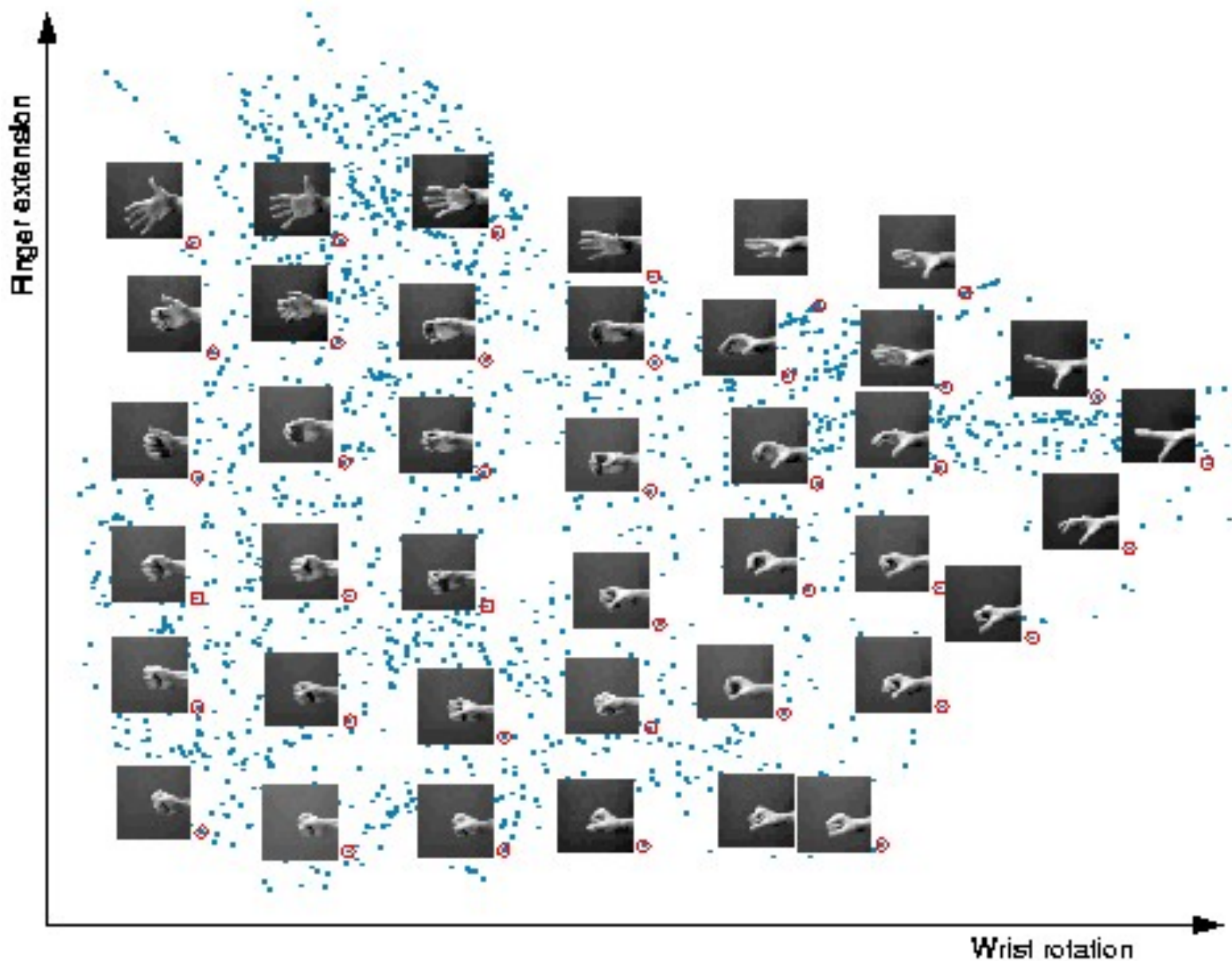
# Manifold learning



from [Intro. ML]

# Manifold learning

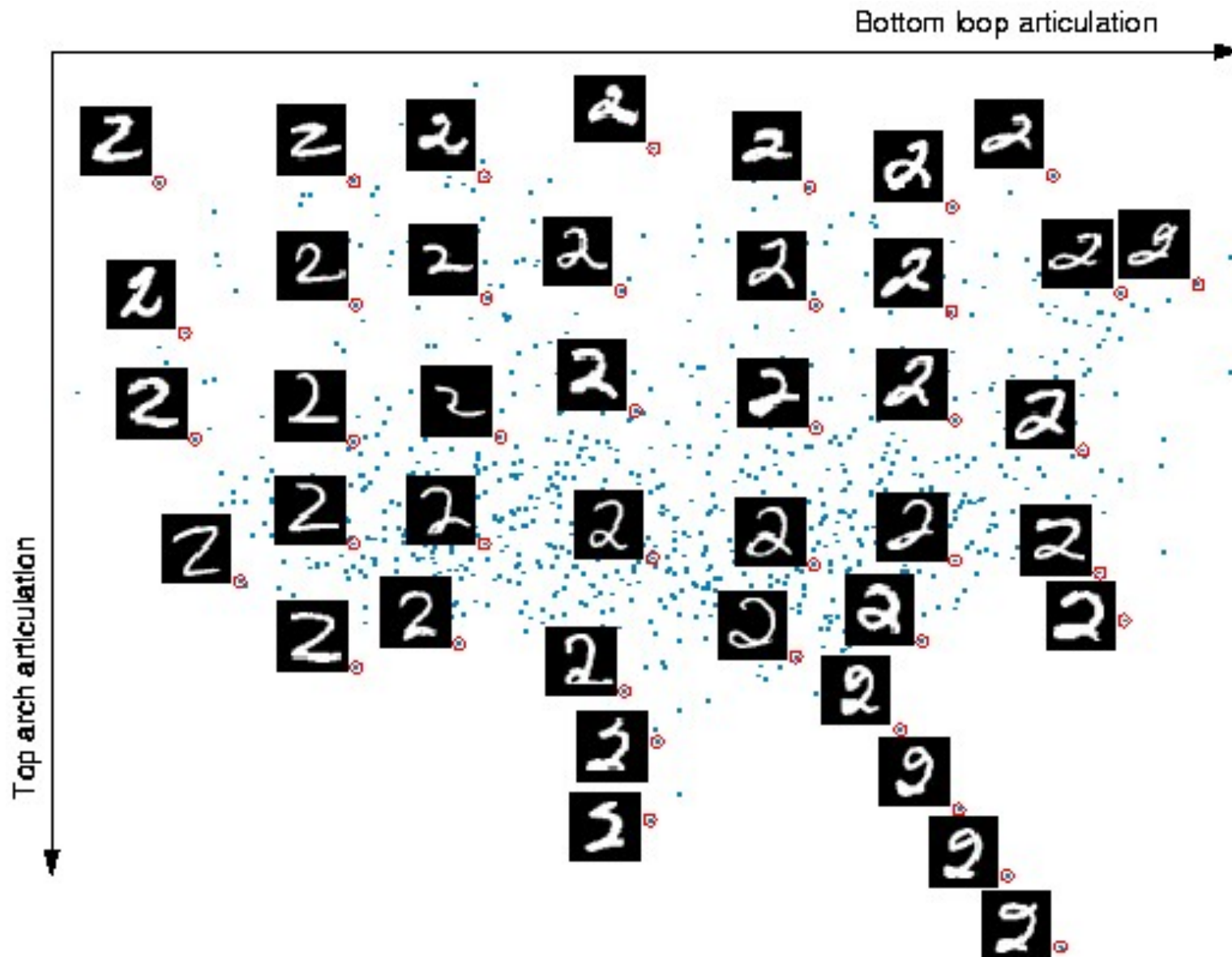
more manifold learning examples





# Manifold learning

more manifold learning examples

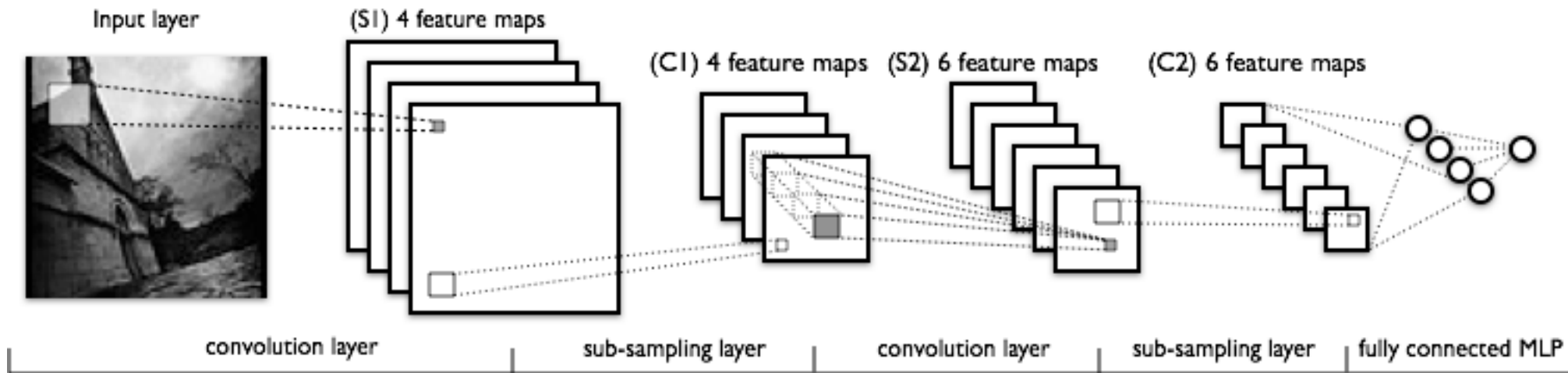




# Other feature extraction methods

Most feature extractions are case specific

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



# A summary of approaches

