

# Introduction to Audio Content Analysis

## Module 3.7.4: Feature Dimensionality Reduction

alexander lerch

## corresponding textbook section

### Section 3.7.4

#### ■ lecture content

- problems of dimensionality
- feature selection
- feature transformation/mapping

#### ■ learning objectives

- describe potential challenges with high-dimensional feature spaces
- discuss advantages and disadvantages of various methods for feature selection
- summarize PCA as feature transformation method



## corresponding textbook section

### Section 3.7.4

#### ■ lecture content

- problems of dimensionality
- feature selection
- feature transformation/mapping

#### ■ learning objectives

- describe potential challenges with high-dimensional feature spaces
- discuss advantages and disadvantages of various methods for feature selection
- summarize PCA as feature transformation method



# introduction

## dimensionality reduction

### ■ problem

- many ML approaches cannot cope with large amounts of irrelevant features
- ML algorithms might degrade in performance

### ■ advantages

- reducing storage requirements
- reducing training complexity
- defying the “curse of dimensionality”

### ■ disadvantages

- additional workload for reduction
- adding an additional layer of model complexity

# introduction

## dimensionality reduction

### ■ problem

- many ML approaches cannot cope with large amounts of irrelevant features
- ML algorithms might degrade in performance

### ■ advantages

- reducing storage requirements
- reducing training complexity
- defying the “curse of dimensionality”

### ■ disadvantages

- additional workload for reduction
- adding an additional layer of model complexity

# introduction

## dimensionality reduction

### ■ problem

- many ML approaches cannot cope with large amounts of irrelevant features
- ML algorithms might degrade in performance

### ■ advantages

- reducing storage requirements
- reducing training complexity
- defying the “curse of dimensionality”

### ■ disadvantages

- additional workload for reduction
- adding an additional layer of model complexity

# introduction

## dimensionality issues

problems of high-dimensional data:

- increase in run-time
- overfitting
- curse of dimensionality
- required amount of training samples

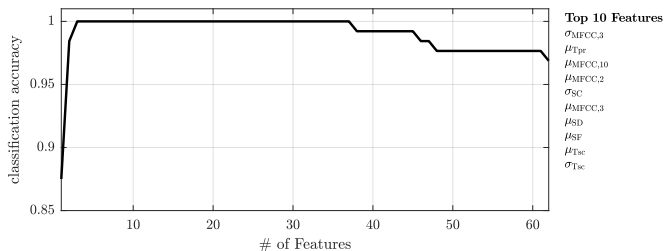
# introduction

## dimensionality issues

problems of high-dimensional data:

- increase in run-time
- overfitting
- curse of dimensionality
- required amount of training samples

⇒ increasing number of input features may *decrease* classification performance





# dimensionality issues

## overfitting

### ■ overfitting:

- lack of training data
- overly complex model

⇒ model cannot be estimated properly

- required training set size depends on
  - ▶ classifier and its parametrization
  - ▶ number of classes
  - ▶ ...

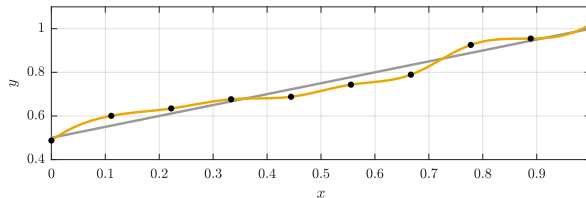
- *rule of thumb:*  
don't bother with training sets smaller than  $\mathcal{F}^2$

# dimensionality issues

## overfitting

### ■ overfitting:

- lack of training data
  - overly complex model
- ⇒ model cannot be estimated properly



- required training set size depends on
  - ▶ classifier and its parametrization
  - ▶ number of classes

# dimensionality issues

## overfitting

### ■ overfitting:

- lack of training data
- overly complex model

⇒ model cannot be estimated properly

- required training set size depends on
  - ▶ classifier and its parametrization
  - ▶ number of classes
  - ▶ ...

- *rule of thumb:*

don't bother with training sets smaller than  $\mathcal{F}^2$

# dimensionality issues

## curse of dimensionality

### ■ curse of dimensionality:

- increasing dimensionality leads to sparse training data
- neighborhoods of data points become less concentrated
- model tends to be harder to estimate in higher-dimensional space
- applies to distance-based algorithms

# dimensionality issues

## curse of dimensionality

**example** (uniformly distributed data): identify region on axis covering **1% of data**

- 1-D: 1% of x-axis
- 2-D: 10% of x/y-axis
- 3-D: 21.5% of x/y/z-axis
- 10-D: 63%
- 100-D: 95%



# dimensionality reduction

## introduction

### ■ **feature subset selection:**

discard least helpful features

- high “discriminative” or descriptive power
- non-correlation to other features
- invariance to irrelevancies

### ■ feature space transformation:

map feature space

# dimensionality reduction

## introduction

### ■ feature subset selection:

discard least helpful features

- high “discriminative” or descriptive power
- non-correlation to other features
- invariance to irrelevancies

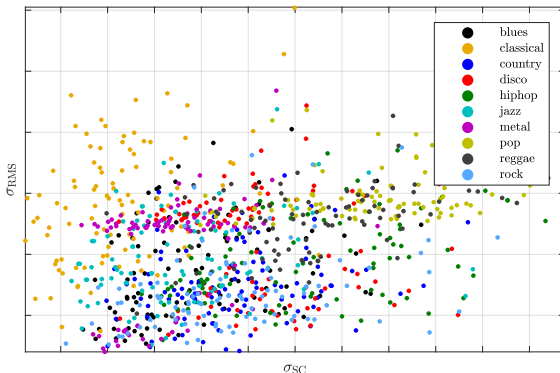
### ■ feature space transformation:

map feature space

# feature subset selection

## manual feature selection

example scatter  
plots of pairs of  
features in a  
multi-class  
scenario





# feature subset selection

## introduction

### 1 wrapper methods:

- *description*
  - ▶ use the “classifier” itself to evaluate feature performance
- *advantages*
  - ▶ taking into account feature dependencies
  - ▶ model dependency
- *disadvantages*
  - ▶ complexity
  - ▶ risk of overfitting

### 2 filter methods:

- *description*
  - ▶ use an objective function
- *advantages*
  - ▶ easily scalable
  - ▶ independent of classification algorithm
- *disadvantages*
  - ▶ no interaction with classifier
  - ▶ no feature dependencies

# feature subset selection

## introduction

### 1 wrapper methods:

- *description*
  - ▶ use the “classifier” itself to evaluate feature performance
- *advantages*
  - ▶ taking into account feature dependencies
  - ▶ model dependency
- *disadvantages*
  - ▶ complexity
  - ▶ risk of overfitting

### 2 filter methods:

- *description*
  - ▶ use an objective function
- *advantages*
  - ▶ easily scalable
  - ▶ independent of classification algorithm
- *disadvantages*
  - ▶ no interaction with classifier
  - ▶ no feature dependencies

# feature subset selection

## wrapper methods 1/2

### 1 single variable classification:

- *procedure*
  - ▶ evaluate each feature individually
  - ▶ choose the top  $N$
- *complexity*
  - ▶ subsets to test:  $\mathcal{F}$
- *challenges*
  - ▶ inter-feature correlation is not considered
  - ▶ feature combinations are not considered

### 2 brute force subset selection

- *procedure*
  - ▶ evaluate all possible feature combinations
  - ▶ choose the optimal combination
- *complexity*
  - ▶ subsets to test:  $2^{\mathcal{F}}$

# feature subset selection

## wrapper methods 1/2

### 1 single variable classification:

- *procedure*
  - ▶ evaluate each feature individually
  - ▶ choose the top  $N$
- *complexity*
  - ▶ subsets to test:  $\mathcal{F}$
- *challenges*
  - ▶ inter-feature correlation is not considered
  - ▶ feature combinations are not considered

### 2 brute force subset selection

- *procedure*
  - ▶ evaluate all possible feature combinations
  - ▶ choose the optimal combination
- *complexity*
  - ▶ subsets to test:  $2^{\mathcal{F}}$

# feature subset selection

## wrapper methods 2/2

### 4 sequential forward selection

- *procedure*

- ① init: empty feature subset  $\mathcal{V}_s = \emptyset$
- ② find feature  $v_j$  maximizing objective function

$$v_j = \operatorname{argmax}_{\forall j | v_j \notin \mathcal{V}_s} J(\mathcal{V}_s \cup v_j)$$

- ③ add feature  $v_j$  to  $\mathcal{V}_s$
- ④ go to step 2

### 5 sequential backward elimination

- *procedure*

- ① init: full feature set
- ② find feature  $v_j$  with the least impact on objective function
- ③ discard feature  $v_j$
- ④ go to step 2

# feature subset selection

## wrapper methods 2/2

### 4 sequential forward selection

- *procedure*

- ① init: empty feature subset  $\mathcal{V}_s = \emptyset$
- ② find feature  $v_j$  maximizing objective function

$$v_j = \operatorname{argmax}_{\forall j | v_j \notin \mathcal{V}_s} J(\mathcal{V}_s \cup v_j)$$

- ③ add feature  $v_j$  to  $\mathcal{V}_s$
- ④ go to step 2

### 5 sequential backward elimination

- *procedure*

- ① init: full feature set
- ② find feature  $v_j$  with the least impact on objective function
- ③ discard feature  $v_j$
- ④ go to step 2

# feature space transformation

## PCA introduction

### ■ objective

- map features to new coordinate system

$$\mathbf{u}(n) = \mathbf{T}^T \cdot \mathbf{v}(n)$$

- ▶  $\mathbf{u}(n)$ : transformed features (same dimension as  $\mathbf{v}(n)$ )
- ▶  $\mathbf{T}$ : transformation matrix ( $\mathcal{F} \times \mathcal{F}$ )

$$\mathbf{T} = \begin{bmatrix} \mathbf{c}_0 & \mathbf{c}_1 & \dots & \mathbf{c}_{\mathcal{F}-1} \end{bmatrix}$$

### ■ properties

- $\mathbf{c}_0$  points in the direction of highest *variance*
- variance concentrated in as few output components as possible
- $\mathbf{c}_i$  orthogonal

$$\mathbf{c}_i^T \cdot \mathbf{c}_j = 0 \quad \forall i \neq j$$

- transformation is invertible

$$\mathbf{v}(n) = \mathbf{T} \cdot \mathbf{u}(n)$$

# feature space transformation

## PCA introduction

### ■ objective

- map features to new coordinate system

$$\mathbf{u}(n) = \mathbf{T}^T \cdot \mathbf{v}(n)$$

- ▶  $\mathbf{u}(n)$ : transformed features (same dimension as  $\mathbf{v}(n)$ )
- ▶  $\mathbf{T}$ : transformation matrix ( $\mathcal{F} \times \mathcal{F}$ )

$$\mathbf{T} = \begin{bmatrix} \mathbf{c}_0 & \mathbf{c}_1 & \dots & \mathbf{c}_{\mathcal{F}-1} \end{bmatrix}$$

### ■ properties

- $\mathbf{c}_0$  points in the direction of highest *variance*
- variance concentrated in as few output components as possible
- $\mathbf{c}_i$  orthogonal

$$\mathbf{c}_i^T \cdot \mathbf{c}_j = 0 \quad \forall \quad i \neq j$$

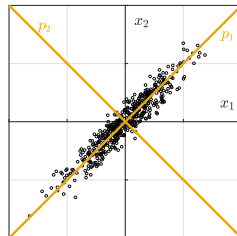
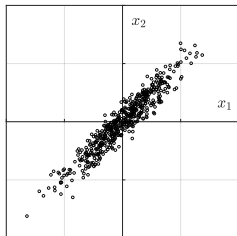
- transformation is invertible

$$\mathbf{v}(n) = \mathbf{T} \cdot \mathbf{u}(n)$$



# feature space transformation

## PCA visualization



calculation of the transformation matrix

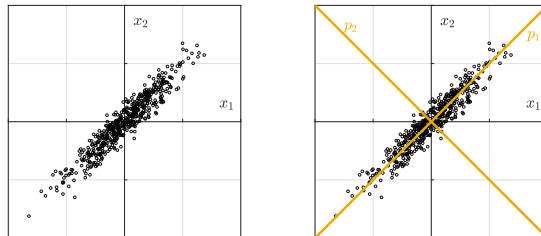
- 1 compute covariance matrix  $R$

$$R = \mathcal{E}\{(V - \mathcal{E}\{V\})(V - \mathcal{E}\{V\})\}$$

- 2 choose eigenvectors as axes for the new coordinate system

# feature space transformation

## PCA visualization



calculation of the transformation matrix

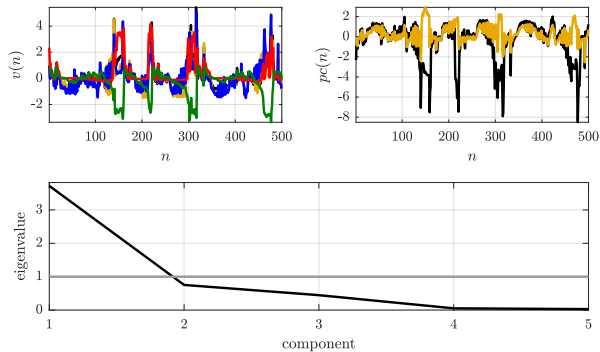
- 1 compute covariance matrix  $R$

$$R = \mathcal{E}\{(V - \mathcal{E}\{V\})(V - \mathcal{E}\{V\})\}$$

- 2 choose eigenvectors as axes for the new coordinate system

# introduction

## PCA example



# introduction

## PCA example

### pca transformation matrix

$$\begin{bmatrix} -0.4187 & 0.3467 & -0.4569 & 0.4143 & -0.1271 & -0.5549 \\ -0.3908 & 0.1815 & 0.8136 & -0.0289 & 0.2060 & -0.3304 \\ -0.4516 & 0.3384 & 0.0859 & 0.2413 & -0.2919 & 0.7285 \\ -0.4337 & 0.1699 & -0.3337 & -0.7243 & 0.3747 & 0.0816 \\ 0.3802 & 0.5599 & -0.0381 & 0.2808 & 0.6622 & 0.1524 \\ 0.3679 & 0.6245 & 0.0956 & -0.4071 & -0.5267 & -0.1495 \end{bmatrix}$$

# introduction

## PCA example

### pca transformation matrix

$$\begin{bmatrix} -0.4187 & 0.3467 & -0.4569 & 0.4143 & -0.1271 & -0.5549 \\ -0.3908 & 0.1815 & 0.8136 & -0.0289 & 0.2060 & -0.3304 \\ -0.4516 & 0.3384 & 0.0859 & 0.2413 & -0.2919 & 0.7285 \\ -0.4337 & 0.1699 & -0.3337 & -0.7243 & 0.3747 & 0.0816 \\ 0.3802 & 0.5599 & -0.0381 & 0.2808 & 0.6622 & 0.1524 \\ 0.3679 & 0.6245 & 0.0956 & -0.4071 & -0.5267 & -0.1495 \end{bmatrix}$$

# summary

## lecture content

### ■ dimensionality problems

- overfitting
- insufficient training data  $\Rightarrow$  sparse feature space

### ■ feature selection

- select subset of features performing “best”
- wrapper methods use classifier itself as objective function
- filter methods use different objective function

### ■ feature transformation

- map feature space into new space minimizing irrelevant information
- still requires computation of all features
- new dimensions commonly lack interpretability

