Feature Selection and Representation

Bùi Tiến Lên

2022

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Contents

Notation

symbol	meaning
$a, b, c, N \dots$	scalar number
$\boldsymbol{w}, \boldsymbol{v}, \boldsymbol{x}, \boldsymbol{y} \dots$	column vector
$\boldsymbol{X}, \boldsymbol{Y} \dots$	matrix
$\mathbb{R}$	set of real numbers
$\mathbb{Z}$	set of integer numbers
$\mathbb{N}$	set of natural numbers
$\mathbb{R}^D$	set of vectors
$\mathcal{X}, \mathcal{Y}, \dots$	set
$\mathcal{A}$	algorithm

operator	meaning
$\boldsymbol{w}^\intercal$	transpose
$\boldsymbol{X}\boldsymbol{Y}$	matrix multiplication
$\boldsymbol{X}^{-1}$	inverse

Motivation

Why Should We Select Features?

- Some problems are defined by **100 or even 1000 input features**
- Most Machine Learning models have to attribute parameters to handle these features (often at least linearly as much)
- Hence, **capacity** is determined by the number of features
- If most features are **noise**, then most of the parameters will be useless $\rightarrow$ capacity is wasted
- Worse, the algorithm might find **false regularities** in the input features of the training data and use the wasted capacity to represent them!
- Other problem: **curse of dimensionality**.
- Finally: for more **interpretability** and **efficiency**.

Classes of Feature Selection Methods

Broad classes of feature selection methods:

- **Filter Methods**:
  - Select the best features according to a **reasonable criterion**
  - The criterion is **independent** of the real problem
- **Wrapper Methods**:
  - Select the best features according to the **final criterion**
  - For each subset of features, try to solve the problem
- In any case, there are

$$\sum_{p=1}^{N} \binom{p}{N} = \sum_{p=1}^{N} \frac{N!}{p!(N-p)!} \textbf{ combinations}$$

- Alternative: **weighting methods**.

Filters

Filter Methods

- **Basic idea**: select the best features according to some prior knowledge
- **Examples** of prior knowledge:
  - if we accept to **transform** the features...
  - features should be **uncorrelated** $\rightarrow$ perform a **PCA** and keep only the eigenvectors corresponding to $x\%$ of the variance.
  - similar ideas: linear discriminant analysis (**LDA**), independent component analysis (**ICA**)
  - features should have **strong correlation with the target** $\rightarrow$ select the k features most linearly correlated to the target
  - select the k features with highest mutual information with the target:

$$I(x,y) = \sum_i \sum_j p(x=i, y=j) \log \left[\frac{p(x=i, y=j)}{p(x=i)p(y=j)} \right]$$

Wrappers

Wrapper Methods

- Basic (**naive**) algorithm:
  - For each subset of features, solve the problem.
  - Select the best subset.
- Impossible because the problem is exponentially long!
- Alternatives: **greedy** heuristics such as **forward** selection or **backward** elimination

Forward Selection

1. let $\mathcal{P} = \emptyset$ be the **current** set of selected features
2. let $\mathcal{Q}$ be the **full** set of features
3. while size of $\mathcal{P}$ smaller than a given constant
   3.1 for each $v \in \mathcal{Q}$
      - set $\mathcal{P}' \leftarrow \{v\} \cup \mathcal{P}$
      - train the model with $\mathcal{P}'$ and keep the **validation** performance

   3.2 set $\mathcal{P}' \leftarrow \{v^*\} \cup \mathcal{P}$ where v^* corresponds to the **best** validation performance obtained in step 3.1
   3.3 set $\mathcal{Q} \leftarrow \mathcal{Q} \setminus \{v^*\}$
   3.4 keep the validation performance obtained with current $\mathcal{P}$
4. return the **best set** $\mathcal{P}$

Backward Elimination

1. let $\mathcal{P}$ be the **full** set of features
2. while size of $\mathcal{P}$ smaller than a given constant
   2.1 for each $v \in \mathcal{P}$
   - set $\mathcal{P}' \leftarrow \mathcal{P} \setminus \{v\}$
   - train the model with $\mathcal{P}'$ and keep the **validation** performance

   2.2 set $\mathcal{P}' \leftarrow \mathcal{P} \setminus \{v^*\}$ where v^* corresponds to the **worst** validation performance obtained in step 2.1
   2.3 keep the validation performance obtained with current $\mathcal{P}$
3. return the **best set** $\mathcal{P}$

Feature Weighting

Feature Weighting Methods

- Instead of **selecting** a subset of features, which is a **combinatorial** problem, why not simply **weight** them?
- Most feature weighting methods are based on the **wrapper** approach
- **Heuristics** for feature weighting:
  - **gradient descent** on the input space $\rightarrow$ train with all features, then fix the parameters and estimate the importance of each input, and loop
  - **AdaBoost** when each model is trained on one feature only ($\rightarrow$ final solution is a linear combination)

Principal Component Analysis

Introduction

Concept 1

Principal Component Analysis (PCA) is an **unsupervised** dimension-reduction tool that can be used to reduce a large set of variables to a small set that still contains most of the information in the large set.

Algorithm

- **Input**: Data $\mathcal{D} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}, \boldsymbol{x}_i \in \mathbb{R}^D$
- **Output**: projection matrix W

1. Construct the mean vector $\boldsymbol{\mu}$

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i$$

2. Construct the covariance matrix S.

$$S = \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{\mu}_i)(\boldsymbol{x}_i - \boldsymbol{\mu}_i)^\mathsf{T}$$

Algorithm (cont.)

3. Decompose the covariance matrix into its eigenvectors and eigenvalues.

$$\{\boldsymbol{w}_1, ..., \boldsymbol{w}_D\} \text{ and } \{\lambda_1, ..., \lambda_D\}$$

4. Sort the eigenvalues by decreasing order to rank the corresponding eigenvectors.

$$\{\boldsymbol{w}_1, ..., \boldsymbol{w}_D\} \text{ where } \lambda_1 \geq ... \geq \lambda_D$$

5. Select k eigenvectors which correspond to the k largest eigenvalues, where k is the dimensionality of the new feature subspace ($k \leq D$).

$$\{\boldsymbol{w}_1, ..., \boldsymbol{w}_k\} \text{ where } \lambda_1 \geq ... \geq \lambda_k$$

6. Construct a projection matrix W from the "top" k eigenvectors.

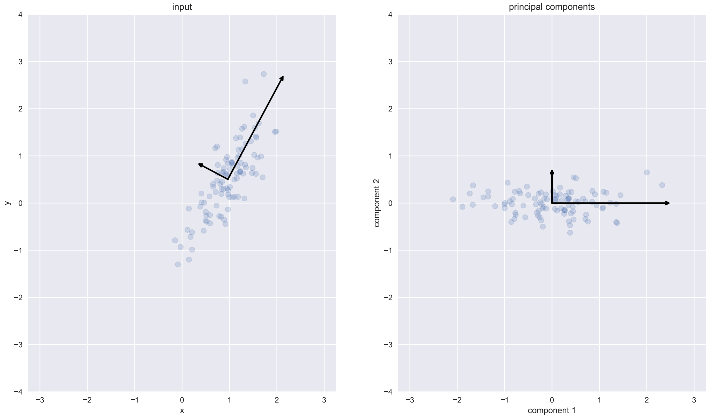
$$W = \begin{bmatrix} \boldsymbol{w}_1 & \dots & \boldsymbol{w}_k \end{bmatrix}^\intercal$$

Algorithm (cont.)

- Transform the D-dimensional input dataset $\mathcal{D}$ using the projection matrix W to obtain the new k-dimensional feature subspace.

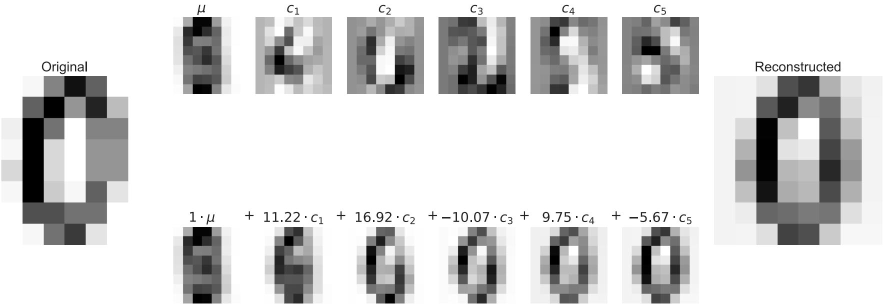
Example

- Project data

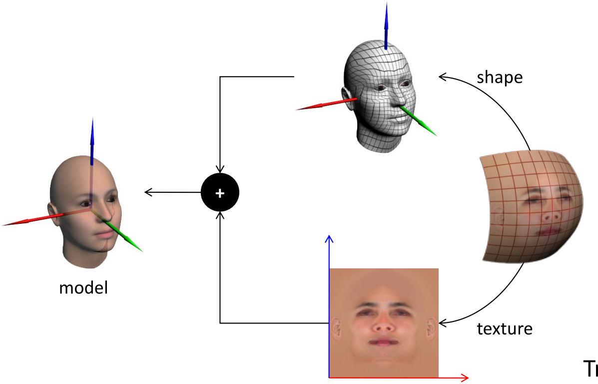


Example

- Reconstruct data



3D Head Model



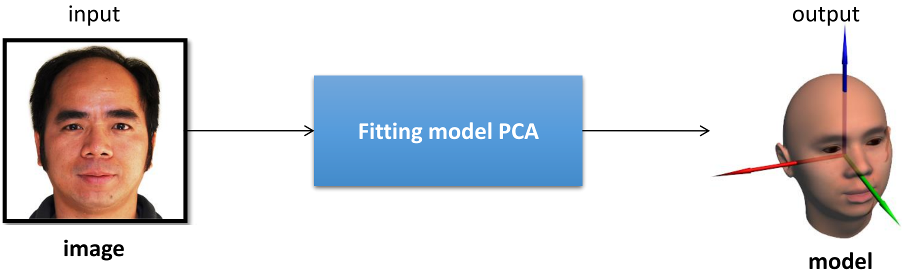
$$\text{Shape } S = \begin{pmatrix} x_1 & x_2 & \dots & x_k \\ y_1 & y_2 & \dots & y_k \\ z_1 & z_2 & \dots & z_k \end{pmatrix}$$

$$\updownarrow \quad \updownarrow \quad \updownarrow$$

$$\text{Texture } T = \begin{pmatrix} r_1 & r_2 & \dots & r_k \\ g_1 & g_2 & \dots & g_k \\ b_1 & b_2 & \dots & b_k \end{pmatrix}$$

$$\text{Triangle List } L = \{t_1, \dots, t_n\}$$

Fitting 3D Head Model



Linear Discriminant Analysis

Introduction

Concept 2

Linear Discriminant Analysis (LDA) is a **supervised** dimension-reduction tool that the goal is to find the feature subspace that optimizes class separability.

Algorithm

- **Input**: Data $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\}, \boldsymbol{x}_i \in \mathbb{R}^D, y_i \in \{c_1, ..., c_k\}$
- **Output**: projection matrix W

1. For each class, compute the D dimensional mean vector.
2. Construct the between-class scatter matrix S_B and the within-class scatter matrix S_W.
3. Compute the eigenvectors and corresponding eigenvalues of the matrix $S_W^{-1} S_B$.
4. Sort the eigenvalues by decreasing order to rank the corresponding eigenvectors.
5. Choose the k eigenvectors that correspond to the k largest eigenvalues to construct a $D \times D$-dimensional transformation matrix W; the eigenvectors are the columns of this matrix.

Algorithm (cont.)

- Transform the D-dimensional input dataset $\mathcal{D}$ using the projection matrix W to obtain the new k-dimensional feature subspace.

References

Goodfellow, I., Bengio, Y., and Courville, A. (2016).
Deep learning.
MIT press.

Lê, B. and Tô, V. (2014).
Cở sở trí tuệ nhân tạo.
Nhà xuất bản Khoa học và Kỹ thuật.

Russell, S. and Norvig, P. (2021).
Artificial intelligence: a modern approach.
Pearson Education Limited.