Dimensionality Reduction

CS60020 : Foundations of Algorithm Design and Machine Learning

25th March 2025 Scribe - Sarthak Sablania

Contents

Dimensionality Reduction

1 Introduction

Dimensionality Reduction aims to reduce the number of features in a dataset while preserving important information. This process involves transforming data from D dimensions $\{x_1, x_2, \ldots, x_D\}$ to d dimensions $\{x_{k1}, \ldots, x_{kd}\}$, where $1 \leq d \leq D$.

It is essential for improving computational efficiency and addressing the curse of dimensionality in machine learning and data analysis.

2 Feature Selection

Feature Selection focuses on identifying a subset of original features that are most relevant for a given task. A key metric used for evaluating the relevance of features is **Kullback-Leibler (KL) Divergence**. KL Divergence quantifies how much one probability distribution P diverges from a second distribution Q. It is given by:

$$D_{KL}(P, Q) = \sum_i (P(i) \log \frac{P(i)}{Q(i)} + Q(i) \log \frac{Q(i)}{P(i)}). \tag{1}$$

In the context of feature selection, the divergence is used to compare the distribution of the data with and without a feature, helping identify features that contribute the most to the information gain.

The computational complexity of exhaustive search for feature selection is $\binom{D}{d} = O(D^d)$, making heuristic methods like forward and backward selection necessary.

2.1 Best Subset Selection: Forward

Forward Selection starts with an empty set and iteratively adds features that improve the model performance based on a chosen selection criterion, such as **KL Divergence**. The process includes:

1. **Initialization:** Start with an empty feature set $\emptyset$.
2. **Feature Evaluation:** For each remaining feature x_i, compute the change in KL Divergence ΔD_{KL} when adding x_i to the current subset.
3. **Feature Addition:** Add the feature with the highest ΔD_{KL} to the subset.
4. Repeat until the desired subset size d is reached or no significant improvement is observed.

The computational complexity for forward selection is $O(D^d)$.

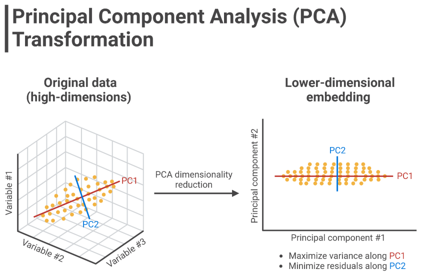
Principal Component Analysis (PCA) Transformation
Original data (high-dimensions)
Variable #1
Variable #2
Variable #3
PC1
PC2
PCA dimensionality reduction
Lower-dimensional embedding
Principal component #2
Principal component #1
PC2
PC1
Maximize variance along PC1
Minimize residuals along PC2

Figure 1: PCA transforming high-dimensional data into a lower-dimensional embedding. The principal components (PC1 and PC2) are chosen to maximize variance along PC1 while minimizing residuals along PC2, enabling effective dimensionality reduction.

2.2 Best Subset Selection: Backward

Backward Selection begins with the full set of features $\{x_1, x_2, \ldots, x_D\}$ and iteratively removes the least useful ones based on the same selection criterion, **KL Divergence**. The process involves:

1. **Initialization:** Start with the full feature set $\{x_1, x_2, \ldots, x_D\}$.
2. **Feature Evaluation:** For each feature x_i in the current subset, compute the change in KL Divergence ΔD_{KL} when removing x_i.
3. **Feature Removal:** Remove the feature with the smallest ΔD_{KL} from the subset.
4. Repeat until the desired subset size d is reached or no significant improvement is observed.

Similar to forward selection, the computational complexity for backward selection is $O(D^d)$.

It is important to note that the solutions obtained via forward and backward search may differ due to their differing strategies, despite using the same selection criterion.

3 Feature Extraction

Feature extraction is the process of creating new features by transforming the original ones, aiming to capture the most critical information in the data.

3.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data. PCA is an unsupervised approach, since it involves only a set of features $X_1, X_2, \ldots, X_p$, and no associated response Y. Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization. It can also be used as a tool for data imputation — that is, for filling in missing values in a data matrix.

We now discuss PCA in greater detail.

3.1.1 What are Principal Components?

PCA finds a low-dimensional representation of a data set that contains as much as possible of the variation. The idea is that each of the n observations lives in p-dimensional space, but not all of these dimensions are equally interesting. PCA seeks a small number of dimensions that are as interesting as possible, where the concept of interesting is measured by the amount that the observations vary along each dimension. Each of the dimensions found by PCA is a linear combination of the p features. We now explain the manner in which these dimensions, or principal components, are found.

The first principal component of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \cdots + \phi_{p1}X_p. \tag{2}$$

that has the largest variance. By *normalized*, we mean that

$$\sum_{j=1}^{p} \phi_{j1}^2 = 1. \tag{3}$$

We refer to the elements $\phi_{11}, \ldots, \phi_{p1}$ as the loadings of the first principal component; together, the loadings make up the principal component loading vector,

$$\boldsymbol{\phi}_1 = (\phi_{11}, \phi_{21}, \ldots, \phi_{p1})^T.$$

We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

Given an $n \times p$ data set X, how do we compute the first principal component? Since we are only interested in variance, we assume that each of the variables in X has been centered to have mean zero (that is, the column means of X are zero). We then look for the linear combination of the sample feature values of the form

$$z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + \cdots + \phi_{p1}x_{ip} \quad \text{subject to} \quad \sum_{j=1}^{p} \phi_{j1}^2 = 1. \tag{4}$$

In other words, the first principal component loading vector solves the optimization problem

$$\max_{\phi_{11},\ldots,\phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^{2} \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \phi_{j1}^{2} = 1. \tag{5}$$

Since $\frac{1}{n}\sum_{i=1}^{n} x_{ij} = 0$, the average of $z_{11}, \ldots, z_{n1}$ will be zero as well. Hence the objective that we are maximizing in (6) is just the sample variance of the n values of z_{i1}. We refer to $z_{11}, \ldots, z_{n1}$ as the scores of the first principal component. Problem (6) can be solved via an eigen decomposition of the variance-covariance matrix of X, as we will see next.

There is a nice geometric interpretation of the first principal component. The loading vector $\boldsymbol{\phi}_1$ with elements $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ defines a direction in feature space along which the data vary the most. If we project the n data points $x_1, \ldots, x_n$ onto this direction, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves.

After the first principal component Z_1 of the features has been determined, we can find the second principal component Z_2. The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance out of all linear combinations that are uncorrelated with Z_1. The second principal component scores $z_{12}, z_{22}, \ldots, z_{n2}$ take the form

$$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \cdots + \phi_{p2} x_{ip} \quad \text{subject to} \quad \sum_{j=1}^{p} \phi_{j1}^{2} = 1. \tag{6}$$

where $\boldsymbol{\phi}_2$ is the second principal component loading vector, with elements $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$. It turns out that constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining direction $\boldsymbol{\phi}_2$ to be orthogonal (perpendicular) to direction $\boldsymbol{\phi}_1$. To find $\boldsymbol{\phi}_2$, we solve a problem similar to (5) with $\boldsymbol{\phi}_2$ replacing $\boldsymbol{\phi}_1$, and with the additional constraint that 2 is orthogonal to 1.

Once we have computed the principal components, we can plot them against each other in order to produce low-dimensional views of the data. Geometrically, this amounts to projecting the original data down onto the subspace spanned by $\boldsymbol{\phi}_1$, $\boldsymbol{\phi}_2$, and $\boldsymbol{\phi}_3$, and plotting the projected points.

An alternative interpretation of principal components can also be useful: principal components provide low-dimensional linear surfaces that are closest to the observations. We expand upon that interpretation here. The first principal component loading vector has a very special property: it is the line in p-dimensional space that is closest to the n observations (using average squared Euclidean distance as a measure of closeness).

3.1.2 How to Find the Principal Components?

PCA aims to maximize the variance in the data by identifying principal components. Suppose that:

- N is the number of data points (samples),
- P is the original feature dimension,
- p is the number of principal components retained ($p \leq P$).

The step-by-step procedure to find them is as follows:

1. **Calculate the Mean:** Compute the mean
$$\bar{X} = (\bar{x_1}, \bar{x_2}, \ldots, \bar{x_P}).$$
2. **Center the Data:** For all $\mathtt{x_i}$, center the data as:
$$\mathtt{x_i'} \leftarrow \mathtt{x_i} - \bar{\mathtt{x_i}}.$$
Or, alternatively, standardize (results will be different).
3. **Covariance Matrix:** Compute the covariance matrix:
$$\mathbf{\Sigma} = \begin{bmatrix} \mathrm{var}(x_1) & \mathrm{cov}(x_1, x_2) \\ \mathrm{cov}(x_1, x_2) & \mathrm{var}(x_2) \end{bmatrix}.$$
For P dimensions, this results in a $P \times P$ matrix.
4. **Eigenvalues and Eigenvectors:** Calculate eigenvalues λ_i and eigenvectors $\boldsymbol{e}_i$ of the covariance matrix. The eigenvector matrix E consists of these eigenvectors as columns:
$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \ldots & \boldsymbol{e}_P \end{bmatrix}.$$
5. **Select Principal Components:** The eigenvectors corresponding to the largest eigenvalues form the principal components. For instance:
$$\boldsymbol{E}_p = \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \ldots & \boldsymbol{e}_p \end{bmatrix}.$$
where the top d components are retained.
6. **Transform the Original Data:** Let X be the data matrix, where each row represents a data sample (after centering). The transformed data is obtained by projecting X onto the selected eigenvectors:
$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{E}_p, \tag{7}$$
where $\boldsymbol{X} \in \mathbb{R}^{N \times P}, \quad \boldsymbol{E}_p \in \mathbb{R}^{P \times p}, \quad \boldsymbol{Z} \in \mathbb{R}^{N \times p}$, and E_p contains the top p eigenvectors as columns.

The resulting Z represents the data in the new principal component space.

3.1.3 Limitations of PCA

1. **Assumes Linearity:** PCA works best for data with linear relationships and may not capture complex, nonlinear structures. Nonlinear techniques like Kernel PCA or t-SNE might be more suitable in such cases.

2. **Interpreting Principal Components is Difficult:** The transformed features are linear combinations of the original variables, making interpretation challenging. Analyzing the loadings (coefficients of original features in PCs) can help understand their contributions.

3.2 Linear Discriminant Analysis (Optional)

Linear discriminant analysis (LDA) is a supervised method for dimensionality reduction for classification problems. We start with the case where there are two classes, then generalize to $K > 2$ classes.

Given samples from two classes C_1 and C_2, we want to find the direction, as defined by a vector $\mathbf{w}$, such that when the data are projected onto $\mathbf{w}$, the examples from the two classes are as well separated as possible. As we saw before,

$$z = \mathbf{w}^T \mathbf{x} \tag{8}$$

is the projection of $\mathbf{x}$ onto $\mathbf{w}$ and thus is a dimensionality reduction from d to 1.

$\mathbf{m}_1$ and m_1 are the means of samples from C_1 before and after projection, respectively. Note that $\mathbf{m}_1 \in \mathbb{R}^d$ and $m_1 \in \mathbb{R}$. We are given a sample $\mathcal{X} = \{\mathbf{x}^t, r^t\}$ such that $r^t = 1$ if $\mathbf{x}^t \in C_1$ and $r^t = 0$ if $\mathbf{x}^t \in C_2$.

$$\mathbf{m}_1 = \frac{\sum_t \mathbf{w}^T \mathbf{x}^t r^t}{\sum_t r^t} = \mathbf{w}^T \mathbf{m}_1 \tag{9}$$

$$\mathbf{m}_2 = \frac{\sum_t \mathbf{w}^T \mathbf{x}^t (1 - r^t)}{\sum_t (1 - r^t)} = \mathbf{w}^T \mathbf{m}_2 \tag{10}$$

The scatter of samples from C_1 and C_2 after projection are

$$s_1^2 = \sum_t (\mathbf{w}^T \mathbf{x}^t - m_1)^2 r^t \tag{11}$$

$$s_2^2 = \sum_t (\mathbf{w}^T \mathbf{x}^t - m_2)^2 (1 - r^t) \tag{12}$$

After projection, for the two classes to be well separated, we would like the means to be as far apart as possible and the examples of classes be scattered in as small a region as possible. So we want $|m_1 - m_2|$ to be large and $s_1^2 + s_2^2$ to be small (see figure 2). Fisher's linear discriminant is $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}. \tag{13}$$

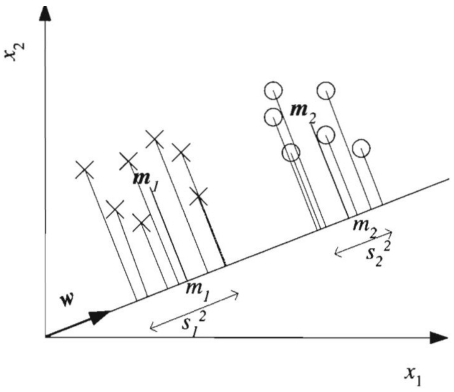

Figure 2: Two-dimensional, two-class data projected on **w**.