Principal Component Analysis

Introduction to Artificial Intelligence with Mathematics
Lecture Notes

Ganguk Hwang

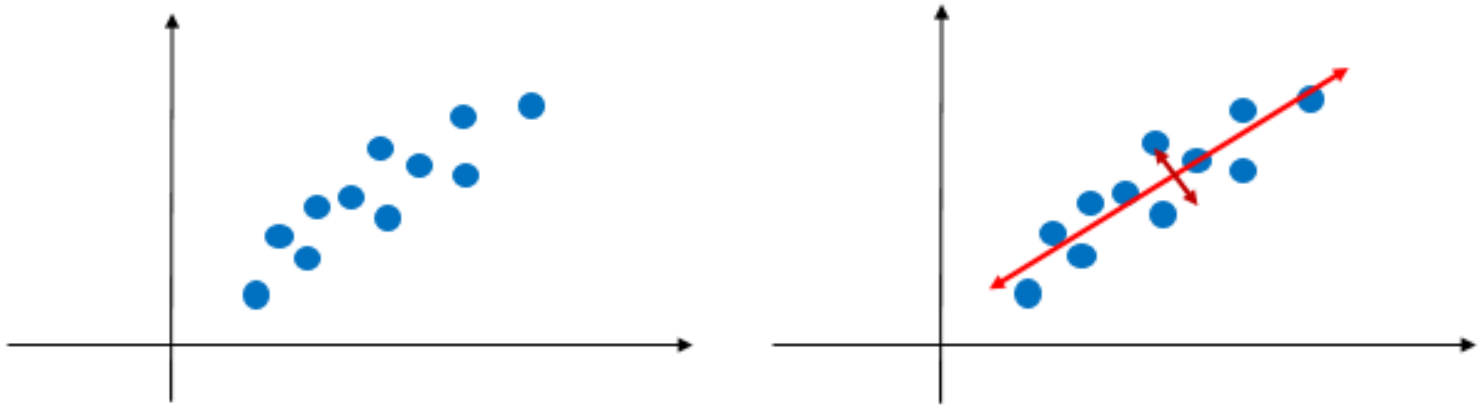
Department of Mathematical Sciences
KAIST

Introduction of Principal Component Analysis

- Principal Component Analysis (PCA) is a technique that is widely used for applications such as dimensionality reduction, lossy data compression, feature extraction, and data visualization.
- Maximum Variance Formulation
- Minimum-error Formulation
- Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2006.

Maximum Variance Formulation

- Consider a data set of observations $\{\mathbf{x}_n\}$ where $n = 1, 2, \cdots, N$ and $\mathbf{x}_i$ are D-dimensional vectors (i.e., $D \times 1$ vectors).
- The main idea of PCA is to project the data onto a space of lower dimension $M (M < D)$, which is called the principal subspace.
- The principal subspace is determined by maximizing the variance of the projected data. Why?



Let

- $\mathbf{u}_i, 1 \leq i \leq M$: the basis vectors of the principal subspace
- Each $\mathbf{u}_i$ is a $D \times 1$ vector and satisfies $\mathbf{u}_i^\top \mathbf{u}_i = 1$.
- $\bar{\mathbf{x}}$: the sample mean of $\{\mathbf{x}_n, n = 1, 2, \cdots, N\}$, i.e.,

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$$

- $\mathbf{S}$: the sample covariance matrix of $\{\mathbf{x}_n, n = 1, 2, \cdots, N\}$, i.e.,

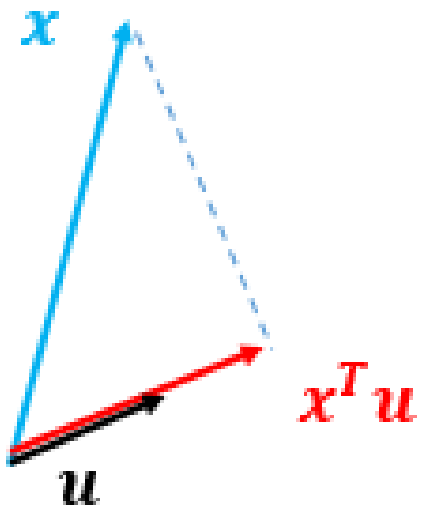
$$\mathbf{S} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top = \frac{1}{N} \sum_{n=1}^{N} \mathbf{z}_n \mathbf{z}_n^\top$$

where $\mathbf{z}_n = \mathbf{x}_n - \bar{\mathbf{x}}$. Note that $\mathbf{S}$ is symmetric.

For a basis vector $\mathbf{u}$, consider the length of the projected data. The variance is given by

$$\frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n^\top \mathbf{u} - \bar{\mathbf{x}}^\top \mathbf{u})^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{z}_n^\top \mathbf{u})^2$$

$$= \frac{1}{N}\sum_{n=1}^{N}\mathbf{u}^\top \mathbf{z}_n \mathbf{z}_n^\top \mathbf{u} = \mathbf{u}^\top \mathbf{S}\mathbf{u}$$



We then now formulate an optimization problem as follows:

$$\max_{\mathbf{u}} \ \mathbf{u}^\top \mathbf{S}\mathbf{u}$$
$$\text{s.t. } \mathbf{u}^\top \mathbf{u} = 1.$$

Using the Lagrange multiplier λ, we make an unconstrained maximiation of

$$J(\mathbf{u}, \lambda) = \mathbf{u}^\top \mathbf{S}\mathbf{u} + \lambda(1 - \mathbf{u}^\top \mathbf{u}).$$

From

$$\nabla_{\mathbf{u}} J = 2(\mathbf{S}\mathbf{u} - \lambda \mathbf{u}) = \mathbf{0}$$

we obtain

$$\mathbf{S}\mathbf{u} = \lambda \mathbf{u}.$$

Moreover, from $\mathbf{Su} = \lambda \mathbf{u}$, by multiplying $\mathbf{u}^\top$ we also obtain

$$\mathbf{u}^\top \mathbf{Su} = \lambda \mathbf{u}^\top \mathbf{u} = \lambda.$$

From the above analysis, we see the following.

- The first principal component (corresponding to the largest variance) is the eigenvector having the largest eigenvalue.

We can define additional principal components in an incremental fashion by choosing each new direction to be that which maximizes the variance of the projected data amongst all possible directions orthogonal to those already considered. That is,

- The principal subspace of dimension M can be obtained with M eigenvectors of the data covariance matrix $\mathbf{S}$ corresponding to the largest M eigenvalues of $\mathbf{S}$.

Minimum-error Formulation

We now consider an alternative formulation of PCA based on projection error minimization.
Let $\mathbf{u}_i, i = 1, 2, \cdots, D$ form an orthornormal basis of the D-dimensional space. Note that $\mathbf{u}_i^\top \mathbf{u}_j = \delta_{ij}$.

Using the orthonormal basis $\mathbf{u}_i, i = 1, 2, \cdots, D$, $\mathbf{x}_n$ can be expressed by

$$\mathbf{x}_n = \sum_{i=1}^{D} \alpha_{ni} \mathbf{u}_i = \sum_{i=1}^{D} (\mathbf{x}_n^\top \mathbf{u}_i) \mathbf{u}_i.$$

Now our objective is to approximate $\mathbf{x}_n$ using a representation in a lower dimension $M(M < D)$ as follows:

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mathbf{u}_i + \sum_{i=M+1}^{D} b_i\mathbf{u}_i.$$

Note that $b_i, i = M+1, \cdots, D$ are the same for all data points $\mathbf{x}_n$.

The objective function is given by

$$J = \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \tilde{\mathbf{x}}_n||^2.$$

Observe that

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=1}^{M}(\alpha_{ni} - z_{ni})\mathbf{u}_i + \sum_{i=M+1}^{D}(\alpha_{ni} - b_i)\mathbf{u}_i$$

and

$$\begin{aligned}
||\mathbf{x}_n - \tilde{\mathbf{x}}_n||^2 &= (\mathbf{x}_n - \tilde{\mathbf{x}}_n)^\top(\mathbf{x}_n - \tilde{\mathbf{x}}_n) \\
&= (\sum_{i=1}^{M}(\alpha_{ni} - z_{ni})\mathbf{u}_i + \sum_{i=M+1}^{D}(\alpha_{ni} - b_i)\mathbf{u}_i)^\top \\
&\qquad \times (\sum_{j=1}^{M}(\alpha_{nj} - z_{nj})\mathbf{u}_j + \sum_{j=M+1}^{D}(\alpha_{nj} - b_j)\mathbf{u}_j \\
&= \sum_{i=1}^{M}(\alpha_{ni} - z_{ni})^2 + \sum_{i=M+1}^{D}(\alpha_{ni} - b_i)^2.
\end{aligned}$$

From

$$J = \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=1}^{M}(\alpha_{ni} - z_{ni})^2 + \sum_{i=M+1}^{D}(\alpha_{ni} - b_i)^2\right),$$

by differntiating J with respect to z_{ni} and b_i and letting them zero we obtain

$$z_{ni} = \alpha_{ni} = \mathbf{x}_n^\top \mathbf{u}_i,$$

$$b_i = \frac{1}{N}\sum_{n=1}^{N}\alpha_{ni} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n^\top \mathbf{u}_i = \bar{\mathbf{x}}^\top \mathbf{u}_i.$$

So J can be expressed as

$$J = \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=M+1}^{D}(\mathbf{x}_n^\top \mathbf{u}_i - \bar{\mathbf{x}}^\top \mathbf{u}_i)^2\right).$$

Moreover, we see that

$$
\begin{aligned}
J &= \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=M+1}^{D}(\mathbf{x}_n^\top \mathbf{u}_i - \bar{\mathbf{x}}^\top \mathbf{u}_i)^2\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=M+1}^{D}((\mathbf{x}_n - \bar{\mathbf{x}})^\top \mathbf{u}_i)^2\right) = \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=M+1}^{D}(\mathbf{z}_n^\top \mathbf{u}_i)^2\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=M+1}^{D}\mathbf{u}_i^\top \mathbf{z}_n \mathbf{z}_n^\top \mathbf{u}_i\right) = \sum_{i=M+1}^{D}\mathbf{u}_i^\top\left(\frac{1}{N}\sum_{n=1}^{N}\mathbf{z}_n\mathbf{z}_n^\top\right)\mathbf{u}_i \\
&= \sum_{i=M+1}^{D}\mathbf{u}_i^\top \mathbf{S}\mathbf{u}_i = \sum_{i=M+1}^{D}\lambda_i.
\end{aligned}
$$

To minimize J, we consider the $D - M$ smallest eigenvalues. In other words, for approximation we take the M eigenvectors corresponding to the largest eigenvalues of $\mathbf{S}$.

Examples:

1. The examples are based on `https://github.com/rickiepark/hanson-ml`.
2. They require python libraries, **scikit-learn, numpy, matplotlib, pandas**. For implementation, it is recommended to install those libraries.

The iris data

PCA is widely used for a data whose feature dimension is huge. To check this, we deal with two examples of the classification problem which cannot be visualized.

The first one is the well-known iris data. It has 4 features with 150 data. For model test, we split our data set into a training set (112) and a test set (38).

Here, we compare the test accuracies of the logistic regression models which use

1. the whole data (4 features),
2. reduced data via PCA extraction (2 features), and
3. reduced data via heuristic selection (2 features) where we select to use petal width and length.

The iris data

Before the comparison, we visualize our two reduced data.

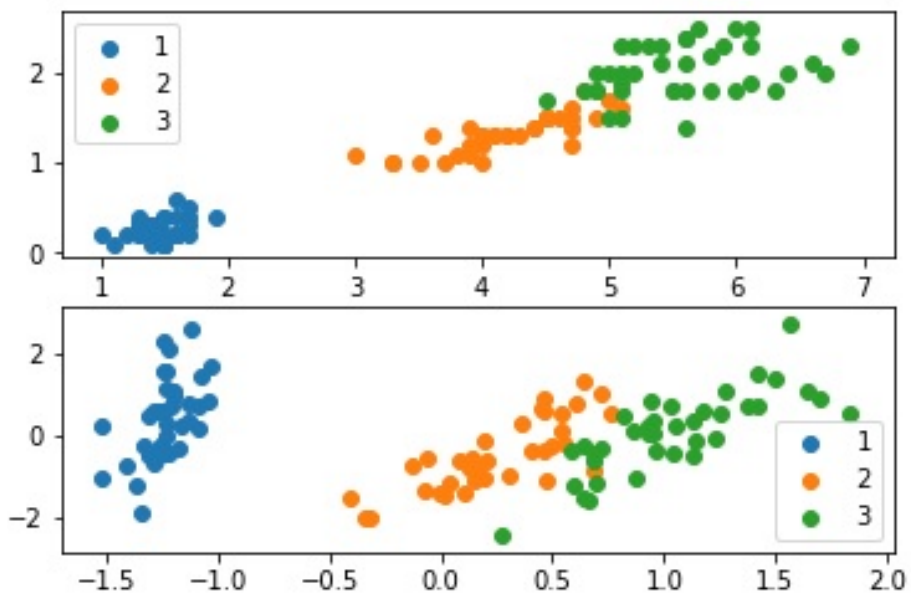


Figure: Upper : Visualization of reduced data via heuristic selection, Lower : Visualization of reduced data via PCA extraction

Without using a feature extraction method like PCA, the original data already has two important features for classification.

	test accuracy	computation time
The whole data	97.37%	0.001s
Reduced data (heuristic)	97.37%	< 0.001s
Reduced data (PCA)	94.74%	< 0.001s

Table: The comparision of three logistic regression models

From the previous figure, we can say that **as the original data already has key features for classification, the effect of PCA is small**. Moreover, as the training data is small (112), we cannot check the amount of time that we can save by using the PCA.

The hand written digits

We now deal with a data which is much bigger than the iris data. It is a hand written digits data, which is a simplified version of the MNIST data.

It has $64 = 8 \times 8$ features with 1797 data. Similar with the iris data, we split the data set into a training (about 75%)and a test set (about 25%). In this data, we extract $6 \times 6 = 36$ features from the original 64 features. The following figure shows the result.

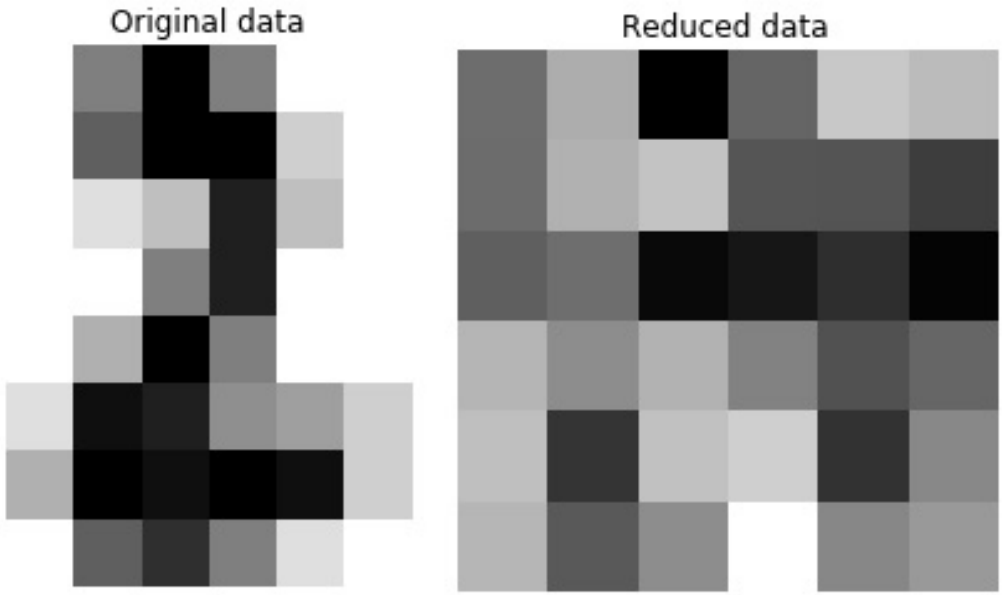


Figure: The original data and reduced data of 2

We again build two logistic regression models which use the original data and reduced data, respectively, for classification.

	test accuracy	computation time
The whole data	95.1%	0.165s
Reduced data (PCA)	**96%**	**0.066s**

Table: The comparision of two logistic regression models

Here, we can check that by using PCA, we can significantly decrease the computation time for training the model. Moreover, as the data has complicate features, a proper feature extraction can lead to better training.

Example: Face Decomposition and Approximation

We apply PCA to decompose a face dataset. Each face picture is a 64×64 matrix.

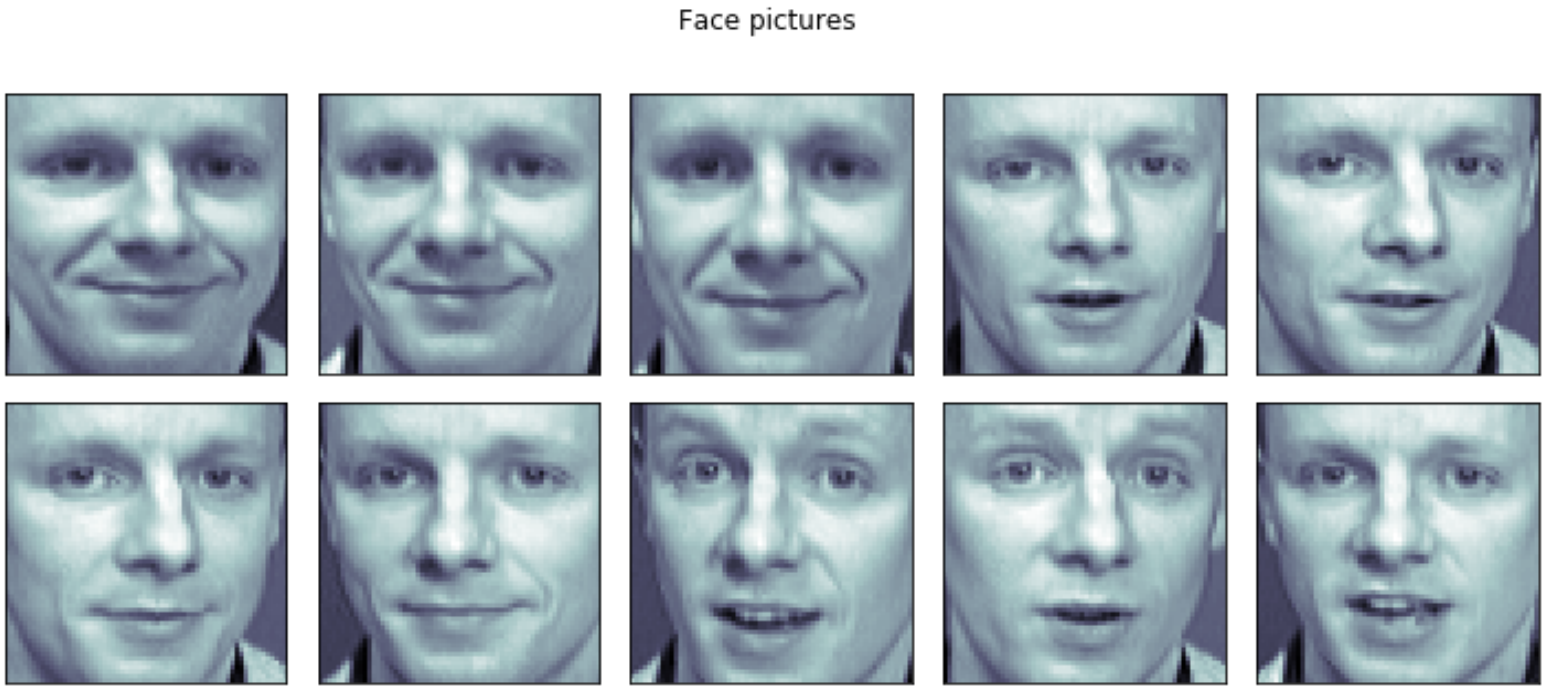


Figure: The original face dataset

We approximate the face dataset with 2 principal components.

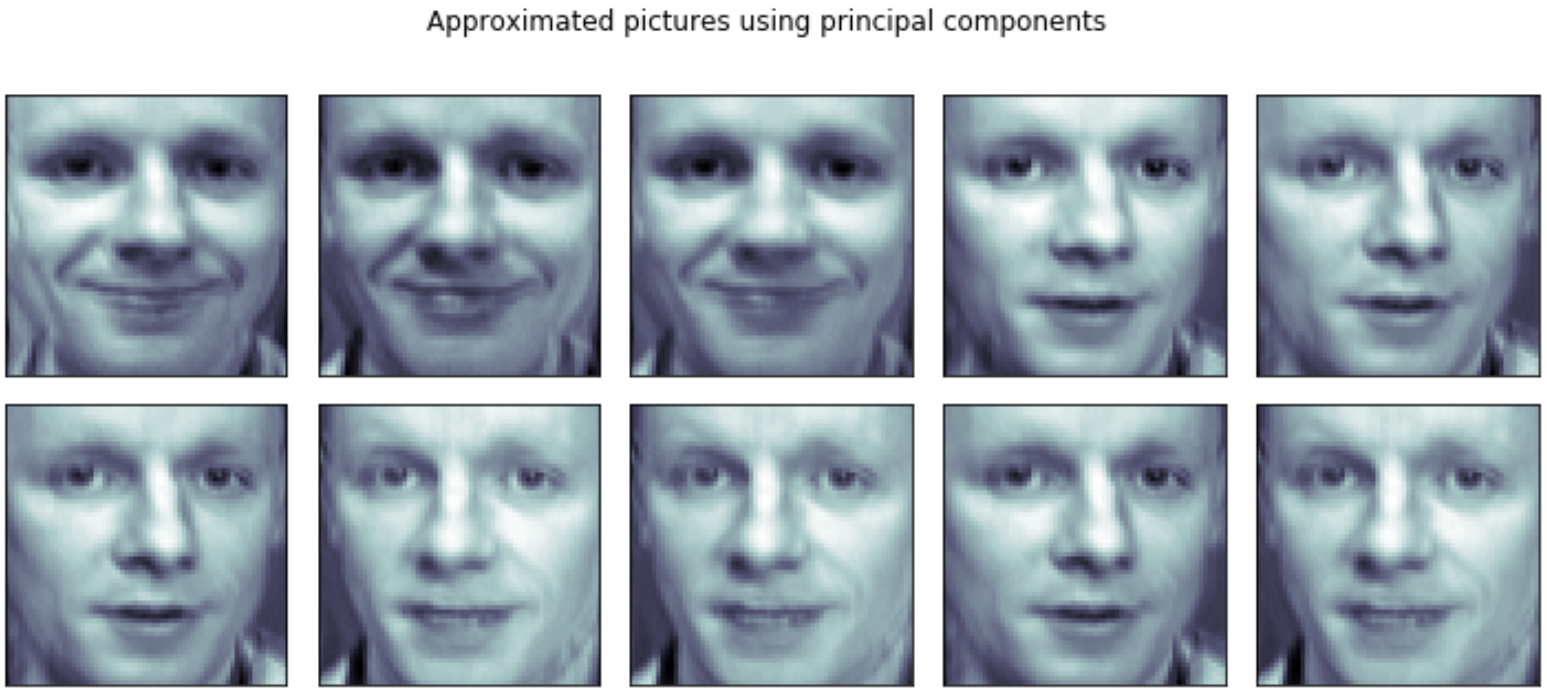


Figure: Approximation with 2 principal componens

The averaged face and two principal components are given as follows.

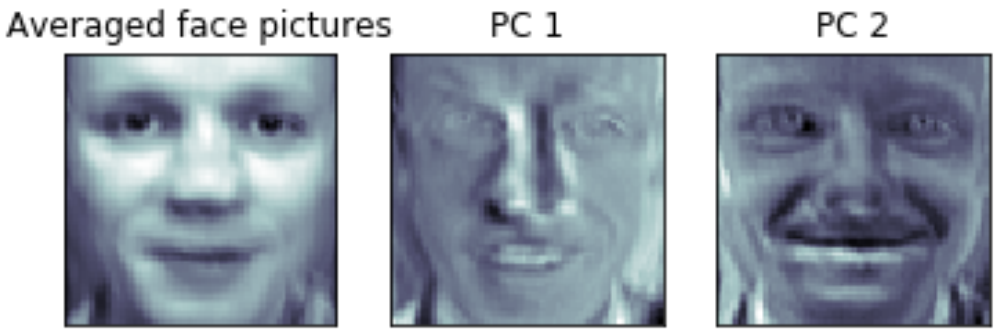


Figure: The averaged face and two principal components

Feature Selection and Feature Extraction

- Feature selection is about selecting a **subset of features** out of the original features in order to reduce model complexity.

$$(x_1, x_2, x_3, \cdots, x_{10}) \longrightarrow (x_2, x_5, x_7)$$

- Feature extraction is about extracting information form the original features **to create a new feature subspace**.