Feature Extraction

Ruwen Qin
Stony Brook University

September 8, 2025

Data used in this module:

- California house prices dataset from sklearn.datasets
- Optical recognition of handwritten digits dataset from sklearn.datasets

Python notebook used in this chapter

- Feature_Extraction.ipynb

1 Background

Considering a system characterized by N attributes. We collect a sample data of the attributes, which contains M observations. Denote the sample data as $\mathbf{X} \in \mathbb{R}^{M \times N}$. If N is large, we hope to characterize the system with fewer but informative features. In some cases N is not large, but some attributes may be correlated with each other. We hope to find features that independently characterize the system from unique perspectives.

In this module, we will study principal component analysis and then briefly introduce a few more ML methods for feature extraction and dimension reduction.

2 Principal Component Analysis (PCA)

Define $\mathcal{P}_d$ as the set of N-dimensional rank-d orthogonal projection matrix[1]. We can project the zero-centered sample data $\widetilde{\mathbf{X}}(= \mathbf{X} - \overline{\mathbf{X}})$ onto the d-dimensional linear sub-space using a projection matrix $\mathbf{P} \in \mathcal{P}_d$. We hope to minimize the reconstruction error; that is, the sum of L_2 distances between the original data $\widetilde{\mathbf{X}}$ and the projected data $\widetilde{\mathbf{X}}\mathbf{P}$:

$$\min_{\mathbf{P} \in \mathcal{P}_d} \|\widetilde{\mathbf{X}}\mathbf{P} - \widetilde{\mathbf{X}}\|_F^2, \tag{1}$$

where $\|\cdot\|_F$ is the Frobenius norm for matrices.

The *Principal Component Analysis* (PCA) [2] shows that the solution to the minimization problem in (1), $\mathbf{P}^*$, contains d columns that are top d eigenvectors of $\mathbf{X}$'s covariance matrix Σ:

$$\Sigma = \frac{(\mathbf{X} - \overline{\mathbf{X}})^{\mathrm{T}}(\mathbf{X} - \overline{\mathbf{X}})}{M - 1} = \frac{\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}}}{M - 1} \tag{2}$$

with eigenvalues $\lambda_1 \geq \cdots \geq \lambda_d$[2].

2.1 Principal Components

Principal components (PCs) are normalized linear combinations of the feature vectors:

$$\mathbf{C}_j = \widetilde{\mathbf{X}}\mathbf{p}_j \tag{3}$$

The variance of the jth PC is

$$\mathrm{Var}[\mathbf{C}_j] = (\widetilde{\mathbf{X}}\mathbf{p}_j)^{\mathrm{T}}(\widetilde{\mathbf{X}}\mathbf{p}_j) = \mathbf{p}_j^{\mathrm{T}}\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}}\mathbf{p}_j = \mathbf{p}_j^{\mathrm{T}}\Sigma\mathbf{p}_j, \tag{4}$$

which is the eigenvalue λ_j corresponding to the eigenvector $\mathbf{p}_j$. Therefore,

$$\frac{\lambda_j}{\sum_{i=1}^{d} \lambda_i} \tag{5}$$

is the proportion of variance explained by the jth PC and

$$\frac{\sum_{i=1}^{j} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \tag{6}$$

is the proportion of variance explained by the first j PCs.

[1] an $N \times N$ square matrix that can be written as UU^{T} where $U \in R^{N \times d}$ has d orthogonal columns.

[2] In linear algebra, we also know that the eigenvectors of $\mathbf{X}$'s co-variance matrix are the right singular vectors of $\widetilde{\mathbf{X}}$ and the eignvalues of $\mathbf{X}$'s co-variance matrix are squares of $\widetilde{\mathbf{X}}$'s positive eigenvalues divided by (M-1)

2.2 Derivation of PCA Algorithm

To minimize the reconstruction loss

$$\begin{aligned}\|\widetilde{\mathbf{X}}\mathbf{P}-\widetilde{\mathbf{X}}\|_F^2 &= \mathrm{Tr}((\widetilde{\mathbf{X}}\mathbf{P}-\widetilde{\mathbf{X}})(\widetilde{\mathbf{X}}\mathbf{P}-\widetilde{\mathbf{X}})^{\mathrm{T}}) \\ &= \mathrm{Tr}(\widetilde{\mathbf{X}}\mathbf{P}\mathbf{P}^{\mathrm{T}}\widetilde{\mathbf{X}}^{\mathrm{T}}-2\widetilde{\mathbf{X}}\mathbf{P}\widetilde{\mathbf{X}}^{\mathrm{T}}+\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^{\mathrm{T}}) \\ &= -\mathrm{Tr}(\widetilde{\mathbf{X}}\mathbf{P}\widetilde{\mathbf{X}}^{\mathrm{T}})+\mathrm{Tr}(\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^{\mathrm{T}})\end{aligned} \tag{7}$$

is to maximize $\mathrm{Tr}(\widetilde{\mathbf{X}}\mathbf{P}\widetilde{\mathbf{X}}^{\mathrm{T}})$[3].

Because $\mathbf{P}$ is an N-dimensional rank-d orthogonal projection matrix, $\mathbf{P}=\mathbf{U}\mathbf{U}^{\mathrm{T}}$, where $\mathbf{U}\in\mathbb{R}^{N\times d}$ containing orthogonal columns. Therefore,

$$\mathrm{Tr}(\widetilde{\mathbf{X}}\mathbf{P}\widetilde{\mathbf{X}}^{\mathrm{T}})=\mathrm{Tr}(\widetilde{\mathbf{X}}\mathbf{U}\mathbf{U}^{\mathrm{T}}\widetilde{\mathbf{X}}^{\mathrm{T}})=\mathrm{Tr}(\mathbf{U}^{\mathrm{T}}\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}}\mathbf{U})^{4}=\sum_{j=1}^{d}\mathbf{u}_j^{\mathrm{T}}\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}}\mathbf{u}_j, \tag{8}$$

where $\mathbf{u}_j$ is the jth column of $\mathbf{U}$.

Therefore, we solve the following optimization problem:

$$\begin{aligned}&\max_{\mathbf{u}_1,\ldots,\mathbf{u}_d}\sum_{j=1}^{d}\mathbf{u}_j^{\mathrm{T}}\Sigma\mathbf{u}_j \\ &\text{subject to:}\quad \mathbf{u}_j^{\mathrm{T}}\mathbf{u}_j=1,\quad \text{for } j=1,\ldots,d,\end{aligned} \tag{9}$$

which is equivalent to

$$\max_{\mathbf{u}_1,\ldots,\mathbf{u}_d,\lambda_1,\ldots,\lambda_d}L=\sum_{j=1}^{d}\mathbf{u}_j^{\mathrm{T}}\Sigma\mathbf{u}_j-\sum_{j=1}^{d}\lambda_j(\mathbf{u}_j^{\mathrm{T}}\mathbf{u}_j-1). \tag{10}$$

This quadratic optimization problem is solved by

$$\begin{aligned}\frac{\partial L}{\partial \mathbf{d}_j}&=\Sigma\mathbf{u}_j-\lambda_j\mathbf{u}_j=0\quad \text{for } j=1,\ldots,d. \\ \frac{\partial L}{\partial \lambda_j}&=1-\mathbf{u}_j^{\mathrm{T}}\mathbf{u}_j=0\quad \text{for } j=1,\ldots,d.\end{aligned} \tag{11}$$

That is, λ_j's are eigenvalues of Σ and $\mathbf{u}_j$'s are the corresponding eigenvectors.

Apparently, the objective function is maximized by selecting the d largest eigenvalues and retaining the corresponding eigenvectors.

3 An Example with Two Attributes

3.1 The Data

Let's consider a system with only two attributes (i.e., N=2). We have a sample data of size 1,000 (i.e. M=1,000), $\mathbf{X}\in\mathbb{R}^{1000\times 2}$.

The sample means of the two features are:

$$\overline{\mathbf{X}}=[\overline{x}_1,\overline{x}_2]=[-0.0366,-0.0047]. \tag{12}$$

The sample covariance matrix is

$$\mathbf{\Sigma}=\frac{(\mathbf{X}-\overline{\mathbf{X}})^{\mathrm{T}}(\mathbf{X}-\overline{\mathbf{X}})}{M-1}=\begin{bmatrix}0.7922 & 0.4049\\ 0.4049 & 0.3207\end{bmatrix}. \tag{13}$$

[3]Since $\mathbf{P}$ is an orthogonal projection matrix, $\mathbf{P}=\mathbf{P}^{\mathrm{T}}$ and $\mathbf{P}^2=\mathbf{P}$. $\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^{\mathrm{T}}$ is a constant.
[4]Tr(AB)=Tr(BA) if matrix multiplication AB and BA both exist.

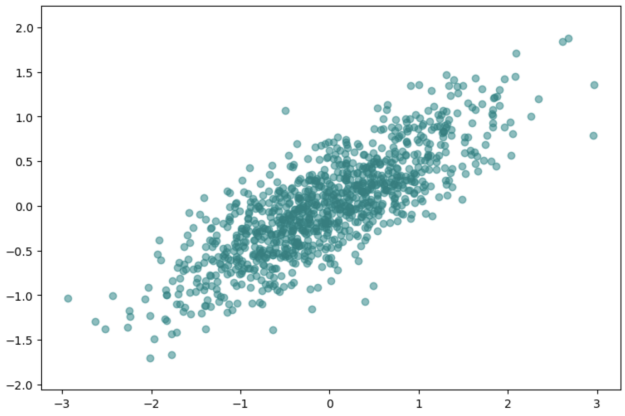


Figure 1: A sample of two-feature data in size 1000

The correlation coefficient is

$$r = \frac{0.4049}{\sqrt{0.7922}\sqrt{0.3207}} = 0.803, \tag{14}$$

which means the two features are strongly correlated with each other.

3.2 Eigen Decomposition of Covariance Matrix

We perform the eigen decomposition for $\mathbf{X}$'s covariance matrix. The eigenvectors are

$$\mathbf{V} = \begin{bmatrix} 0.8670 & -0.4984 \\ 0.4984 & 0.8670 \end{bmatrix} \approx \begin{bmatrix} \cos(\pi/6) & -\sin(\pi/6) \\ \sin(\pi/6) & \cos(\pi/6) \end{bmatrix}. \tag{15}$$

The corresponding eigenvalues in a diagonal matrix are

$$\mathbf{\Lambda} = \begin{bmatrix} 1.0249 & 0 \\ 0 & 0.0879 \end{bmatrix}. \tag{16}$$

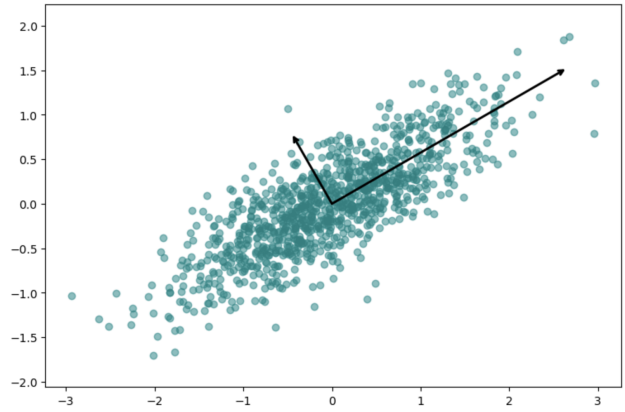


Figure 2: A sample of two-feature data in size 1000

The eigenvectors define two perpendicular directions. As Figure 2 shows, the first direction,

$$\mathbf{v}_1 = \begin{bmatrix} 0.8670 \\ 0.4984 \end{bmatrix} = \begin{bmatrix} \cos(\pi/6) \\ \sin(\pi/6) \end{bmatrix} \tag{17}$$

contains the most variation of the data. The corresponding eigenvalue $\lambda_1 = 1.0249$ measures the amount of variance in this direction.

The second direction is defined by the second eigenvector,

$$\mathbf{v}_2 = \begin{bmatrix} -0.4984 \\ 0.8670 \end{bmatrix} = \begin{bmatrix} -\sin(\pi/6) \\ \cos(\pi/6) \end{bmatrix} \tag{18}$$

The variance in this direction is measured by the eigenvalue $\lambda_2 = 0.0809$.

3.3 PCA with Two Principal Components

Let's perform PCA on the zero-centered data $\widetilde{\mathbf{X}}$. The two projection vectors, which are the eigenvectors of the covariance matrix of data $\mathbf{X}$, defines the projection matrix:

$$\mathbf{P} = [\mathbf{p}_1 \quad \mathbf{p}_2] = \begin{bmatrix} 0.8670 & -0.4984 \\ 0.4984 & 0.8670 \end{bmatrix}. \tag{19}$$

They respectively explain certain amount of variances, measured by their eigenvalues:

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} 1.0249 & 0 \\ 0 & 0.0879 \end{bmatrix}. \tag{20}$$

You must have found that

- the principal components are normalized linear combinations of $\mathbf{X}$'s features using the eigenvectors of $\mathbf{X}$'s sample covariance matrix
- the variances that principal components explain are the eigenvalues of $\mathbf{X}$'s sample covariance matrix.

The variance explained by the first principal component is:

$$\frac{1.0249}{1.0249 + 0.0879} = 92.1\%. \tag{21}$$

Therefore, using the first principal component as a feature seems also to be sufficient.

4 An Example of Feature Extraction using PCA

4.1 California Housing Dataset

Let's practice the feature extraction using PCA. We use the California Housing dataset provided by 'sklearn.dataset'.

```
.. _california_housing_dataset:

California Housing dataset
--------------------------

**Data Set Characteristics:**

    :Number of Instances: 20640

    :Number of Attributes: 8 numeric, predictive attributes and the target

    :Attribute Information:
        - MedInc        median income in block group
        - HouseAge      median house age in block group
        - AveRooms      average number of rooms per household
        - AveBedrms     average number of bedrooms per household
        - Population    block group population
        - AveOccup      average number of household members
        - Latitude      block group latitude
        - Longitude     block group longitude

    :Missing Attribute Values: None

This dataset was obtained from the StatLib repository.
```

```
https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html

The target variable is the median house value for California districts,  expressed in
    hundreds of thousands of dollars ($100,000).

This dataset was derived from the 1990 U.S. census, using one row per census
block group.  A block group is the smallest geographical unit for which the U.S.  Census
    Bureau publishes sample data (a block group typically has a population of 600 to 3,000
    people).

A household is a group of people residing within a home. Since the average number of rooms
    and bedrooms in this dataset are provided per household, these columns may take
    surprisingly large values for block groups with few households and many empty houses,
    such as vacation resorts.

It can be downloaded/loaded using the
:func:`sklearn.datasets.fetch_california_housing` function.

.. topic:: References

    - Pace, R. Kelley and Ronald Barry, Sparse Spatial Autoregressions,
      Statistics and Probability Letters, 33 (1997) 291-297
```

4.2 Data Normalization

Denote the data as $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$. When we check the first few observations of the dataset, we found that those attributes have very different scales.

```
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  Longitude
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88    -122.23
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86    -122.22
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85    -122.24
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85    -122.25
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85    -122.25
```

To eliminate the impact of varied scales of features, we normalize the data through centering (subtracting the sample mean) and scaling (dividing the sample standard deviation):

$$\widetilde{\mathbf{x}_j} = \frac{\mathbf{x}_j - \overline{x}_j}{s_j} \tag{22}$$

for $j = 1, \ldots, N$ before performing PCA. Here,

$$\overline{x}_j = \frac{\sum_{i=1}^{M} x_{i,j}}{M}. \tag{23}$$

is the sample mean of attribute j, and

$$s_j = \sqrt{\frac{\sum_{i=1}^{M} (x_{i,j} - \overline{x}_j)^2}{M-1}} \tag{24}$$

is the sample standard deviation.

4.3 Data Visualization with Two PCs

We perform the PCA on the normalized data. Then, we visualize the data by showing the scatter plot of the first and fourth principal components, shown in Figure 3. The color of the data points is the median house value. The two PCs show the capability for interpreting the median house value.

We also visualize the principal components one by one below. The color of dots is corresponding to the house median price. It shows that the data have limited variation on the last two principal components.

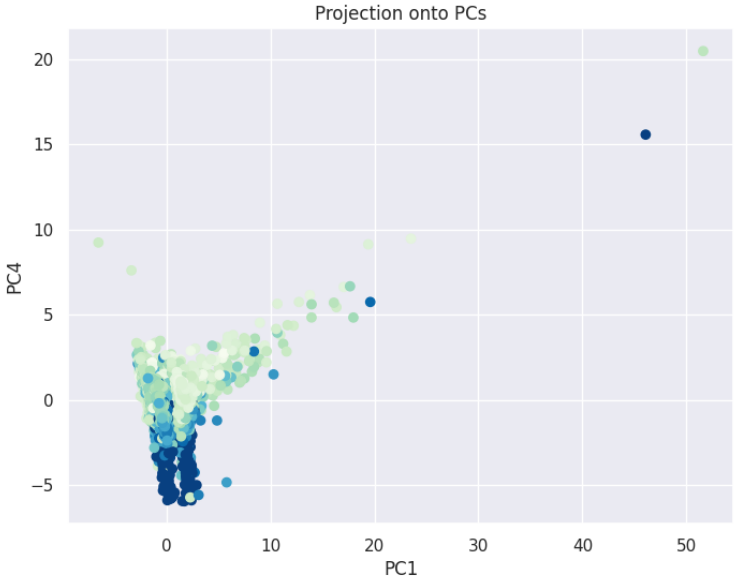


Figure 3: Visualization of the data on the first two principal components

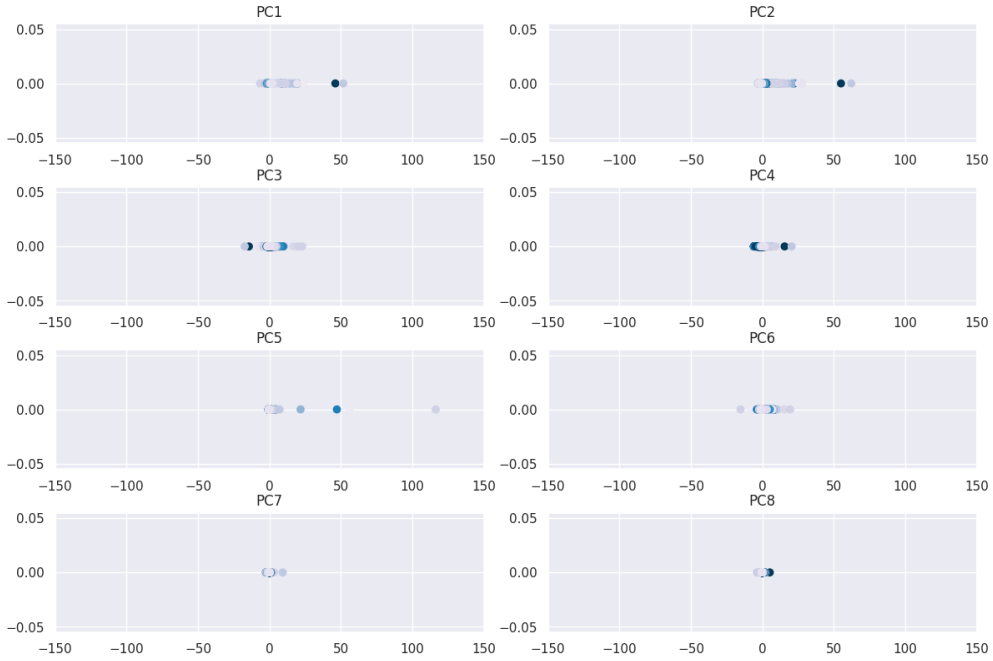


Figure 4: Visualization of the data on each principal component

4.4 Interpret Principal Components

Let

$$[\mathbf{C}_1, \ldots, \mathbf{C}_N] = \widetilde{\mathbf{X}}\mathbf{P} = [\widetilde{\mathbf{X}}\mathbf{p}_1, \ldots, \widetilde{\mathbf{X}}\mathbf{p}_N] \tag{25}$$

be the reconstruction of the normalized data. The jth principal component, $\mathbf{C}_j$, is the normalized linear combination of $\widetilde{\mathbf{X}}$'s N attributes using $\mathbf{p}_j$, the jth eigenvector of $\mathbf{X}$'s covarince matrix:

$$\mathbf{C}_j = \widetilde{\mathbf{X}}\mathbf{p}_j = [\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_N]\mathbf{p}_j = [\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_N] \begin{bmatrix} p_{1,j} \\ \vdots \\ p_{N,j} \end{bmatrix} = p_{1,j}\widetilde{\mathbf{x}}_1 + \cdots + p_{N,j}\widetilde{\mathbf{x}}_N \tag{26}$$

for $j = 1, \ldots, N$, Therefore, we can interpret $\mathbf{p}_j$'s as the loadings of $\widetilde{\mathbf{X}}$ for the linear transformation in Eq. (25). Figure 5 visualizes $\mathbf{p}_j$ for $j = 1, \ldots, 8$ as rows of the heat map. The larger the value of a cell, the higher the correlation between the corresponding principal component and feature. For example, the fifth principal component and the attribute AveOccup are strongly positively correlated. The fourth principal component and the feature Medinc are strongly negatively correlated.

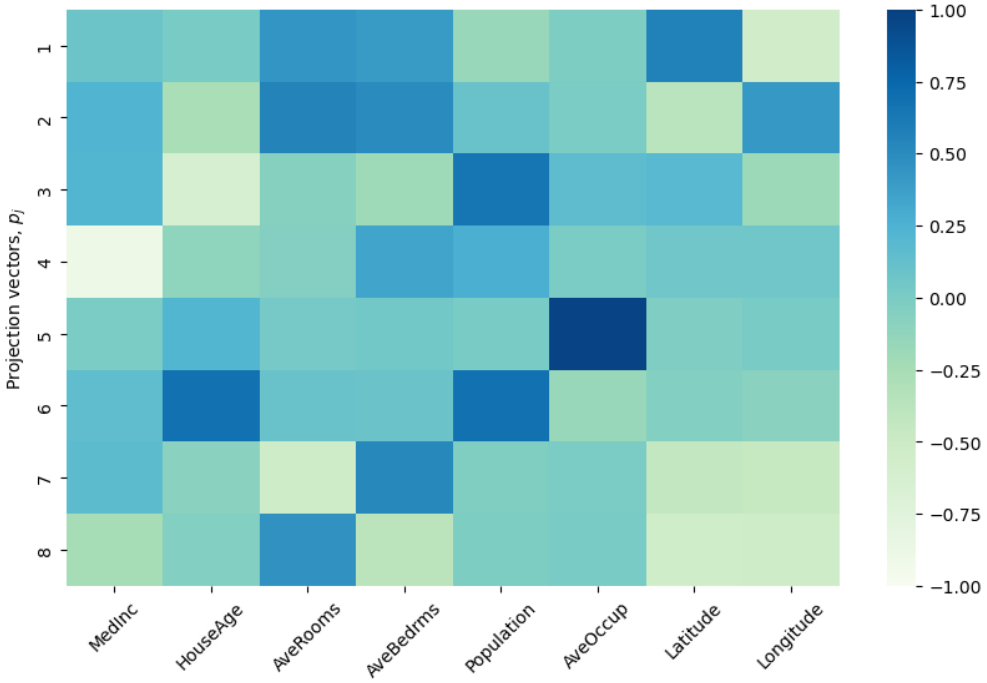
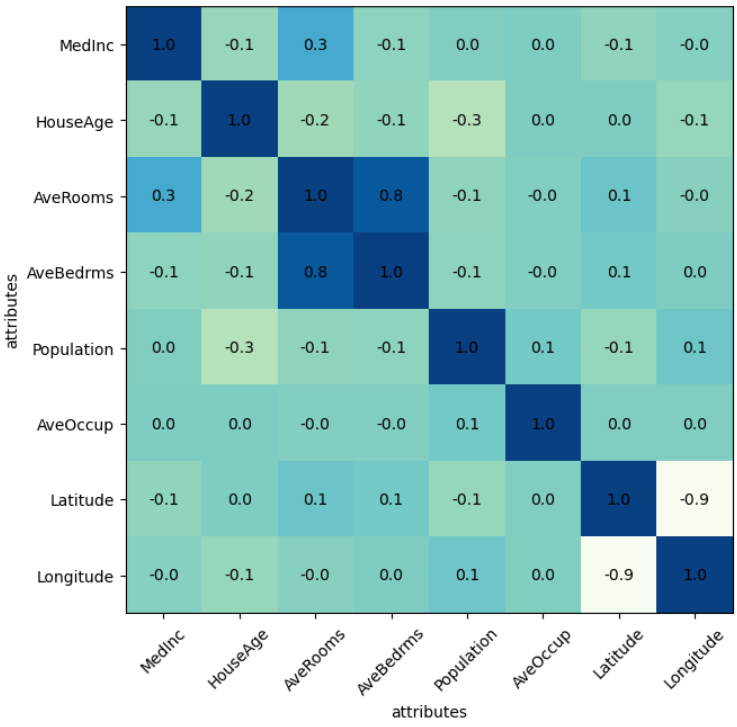
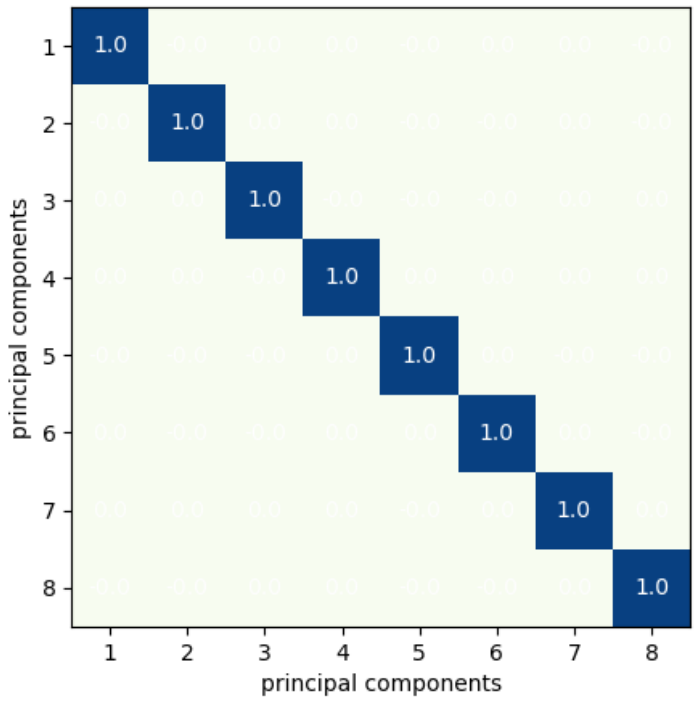


Figure 5: Interpret principal components: project vectors are loadings/weights for linearly aggregating attributes

Fig. 6 visualize the correlation matrices for the attributes and the principal components, respective. Some attributes are apparently correlated. For example, the correlation between AveRooms and AveBedrms is 0.8, and the correlation between MedInc and AveRooms is 0.3. After reconstruction using PCA, the data are represented as principal components that are independent with each other.



(a)



(b)

Figure 6: (a) correlation coefficients between attributes; (b) correlation coefficients between principal components

4.5 Explained Variance by Principal Components

In Figure 7, bars show the proportions of variance explained by individual principal components and the line shows the proportion of variances explained by the top d principal components. The figure shows the first principal component explains 25.34% of total variance. The top 5 principal components explain over 90% of total variance. Therefore, we might extract fewer principal components as new features for predicting the median value of houses. The principal components are independent because their correlation coefficients are zeros. If we build a multiple factor model using principal components for predicting the median house value, the collinearity issue is no longer an issue.

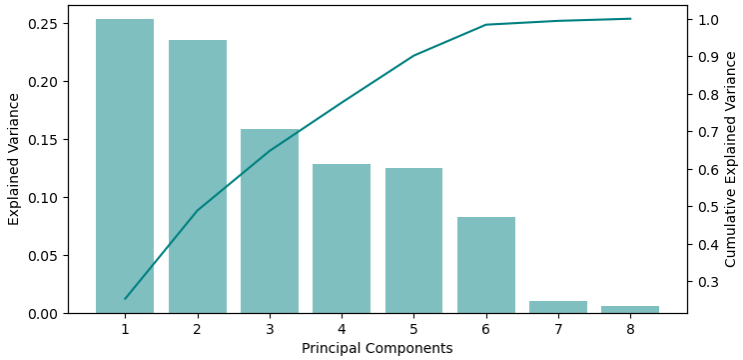


Figure 7: Explained variances by principal components

5 Linear Discriminant Analysis (LDA)

PCA aims to find the most accurate data representation in a lower dimensional space spanned by the maximum variance directions. Such directions may not work well for tasks like classification.

Figure 8 illustrates an example. The left plot shows that the dataset $\mathbf{X} \in \mathbb{R}^{9\times 2}$ contains 9 data points of 2 attributes. $\mathbf{y} \in \mathbb{B}^9$ composes class labels of these data points. There are two classes in the dataset. $y_i \in \{r, g\}$ for $i = 1, \ldots, 9$, which are indicated by the colors. We aim to represent the original two-attribute data with one feature that can maximize the separation of the two classes. That is, we are looking for one direction on which the projection of the data are best separated. The upper right plot illustrates the result of PCA. The direction that maximizes data variance does not separate the two classes of data well. In the bottom right plot, the two classes of data are separated well.

In this section, we introduce the Linear Discriminant Analysis (LDA) method that attempts to preserve discriminatory information between different classes of data.

5.1 The Underlying Mechanism of LDA

- Linear Discriminant Analysis (LDA) is a supervised learning algorithm used for classification tasks. It's an algorithm seeking a linear combination of input attributes/predictors to maximize the separation the classes in a dataset.
- LDA involves searching a lower dimensional space that maximizes the separation between the classes on that space. That is, we maximize the ratio of between-class variance to within-class variance.
- LDA assumes that the data is linear separable. That is, a linear decision boundary can accurately classify the different classes.
- Therefore, LDA is also a supervised learning method commonly used for feature extraction.

LDA assumes that the data of each class has a Gaussian distribution and the covariance matrices of the different classes are equal. It also assumes that the data are linearly separable. If those assumes are not hold, the LDA method is not appropriate.

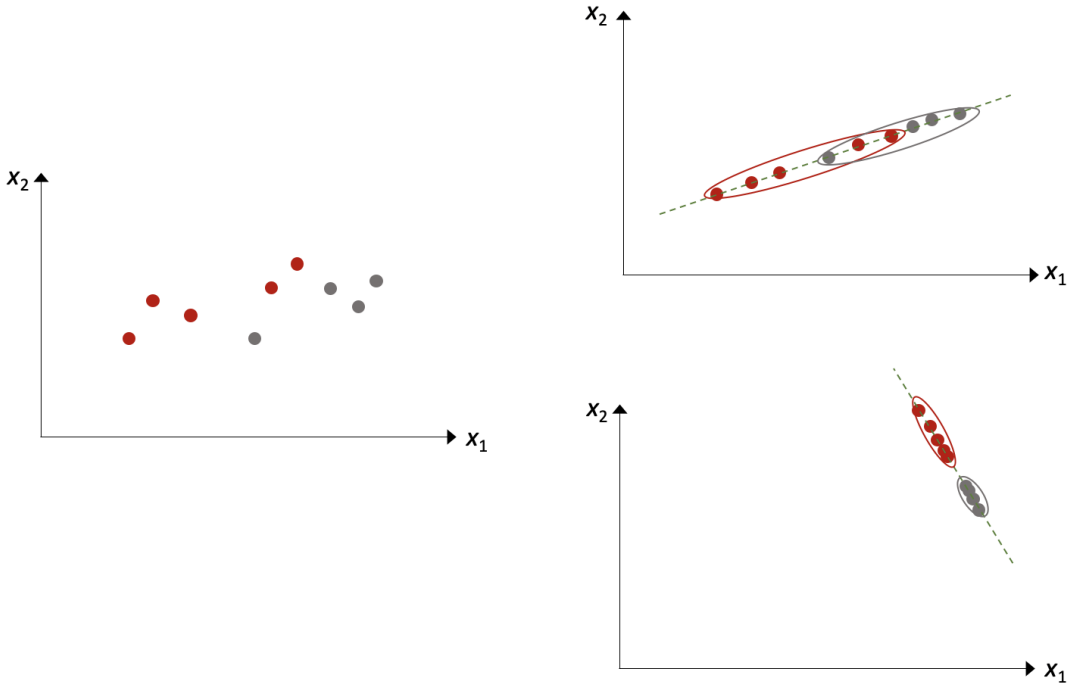


Figure 8: An example of two-class classification problem

5.2 Explanation of LDA using a Simple Example

Further, we describe the LCA approach mathematically based on the simple example illustrated in Figure 8.

Let $\mathbf{v} \in \mathbb{R}^2$ denote the unit vector for projecting the data $\mathbf{X}$ onto a new space with only one dimension. The data on this new one dimensional space, denoted as $\widetilde{\mathbf{X}} \in \mathbb{R}^9$, are

$$\widetilde{\mathbf{X}} = [\widetilde{x}_1, \ldots, \widetilde{x}_9]^{\mathrm{T}} = \mathbf{X}\mathbf{v}. \tag{27}$$

Let $\tilde{c}_r$ and $\tilde{c}_g$ respectively represent the means of the two classes of the data $\widetilde{\mathbf{X}}$:

$$\tilde{c}_r = \frac{\sum_{i=1}^{9} 1\{y_i = r\}\widetilde{x}_i}{\sum_{i=1}^{9} 1\{y_i = r\}}, \tag{28}$$

$$\tilde{c}_g = \frac{\sum_{i=1}^{9} 1\{y_i = g\}\widetilde{x}_i}{\sum_{i=1}^{9} 1\{y_i = g\}}. \tag{29}$$

The within-class variation of the data $\widetilde{\mathbf{X}}$, denoted by $\widetilde{S}_w$, is calculated as

$$\widetilde{S}_w = \sum_{i=1}^{9} 1\{y_i = r\}(\widetilde{x}_i - c_r)^2 + 1\{y_i = g\}(\widetilde{x}_i - c_g)^2, \tag{30}$$

and the between-class variation, $\widetilde{S}_b$, is

$$\widetilde{S}_b = \sum_{i=1}^{9} 1\{y_i = r\}(\tilde{c}_r - \tilde{c})^2 + 1\{y_i = g\}(\tilde{c}_g - \tilde{c})^2, \tag{31}$$

where $\tilde{c} = \sum_i \tilde{x}_i/9$ is the overall mean.

The ratio of between-class variation to within-class variation, $\widetilde{S}_b/\widetilde{S}_w$, measures how well the two classes are separated. It should be mentioned that this ratio is a function of $\mathbf{v}$, the vector we select to generate the lower-dimensional feature,

$$\mathcal{J}(\mathbf{v}) = \frac{\widetilde{S}_b}{\widetilde{S}_w} = \frac{\mathbf{v}^{\mathrm{T}}\mathbf{S}_b\mathbf{v}}{\mathbf{v}^{\mathrm{T}}\mathbf{S}_w\mathbf{v}}, \tag{32}$$

where $\mathbf{S}_b$ and $\mathbf{S}_w$ are the between-class scatter matrix and within-class scatter matrix of data $\mathbf{X}$, respectively.

To maximize $\mathcal{J}(\mathbf{v})$, we take the derivative of $\mathcal{J}(\mathbf{v})$ with respect to $\mathbf{v}$ and make the derivative equal to zero:

$$\frac{d\mathcal{J}(\mathbf{v})}{\mathbf{v}} = 0 \tag{33}$$

Solving this equation is about to find $\mathbf{v}^*$ that solves the following equation:

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{v} = \lambda\mathbf{v}, \tag{34}$$

where

$$\lambda = \frac{\mathbf{v}^{*\mathrm{T}}\mathbf{S}_b\mathbf{v}^*}{\mathbf{v}^{*\mathrm{T}}\mathbf{S}_w\mathbf{v}^*}. \tag{35}$$

That is, the optimal solution $\mathbf{v}^*$ is the eighenvector of $\mathbf{S}_w^{-1}\mathbf{S}_b$ corresponding to the highest eighenvalue.

5.3 Derivation of LDA

Now, let's generalize the description of the LDA method for multiclass data on a higher dimension. Let $\mathbf{X} \in \mathbb{R}^{M\times N}$ be the data composed of M data points with N attributes. The M data points are in K classes, indexed by k, and $\mathbf{y} \in \mathbb{B}^M$ is the vector of class labels for $\mathbf{X}$. Denote $\mathbf{X}_k \in \mathbb{R}^{M_k\times N}$ as the data points of class k in $\mathbf{X}$:

$$\mathbf{X}_k = \{\mathbf{x}_i | y_i = k\}, \tag{36}$$

M_k is the number of data points in $\mathbf{X}_k$, and $\sum_k M_k = M$.

Let $\mathbf{c}_k$ be the mean attributes of $\mathbf{X}_k$,

$$\mathbf{c}_k = \frac{1}{M_k}\sum_{\mathbf{x}_i\in\mathbf{X}_k}\mathbf{x}_i, \tag{37}$$

and $\mathbf{S}_k$ be the scatter matrix of $\mathbf{X}_k$ calculated as

$$\mathbf{S}_k = (\mathbf{X}_k - \mathbf{c}_k)^{\mathrm{T}}(\mathbf{X}_k - \mathbf{c}_k). \tag{38}$$

The total within-class scatter matrix is

$$\mathbf{S}_w = \sum_{k=1}^{K}\mathbf{S}_k. \tag{39}$$

The between-class scatter matrix is

$$\mathbf{S}_b = \sum_{k=1}^{K} M_k(\mathbf{c}_k - \mathbf{c})^{\mathrm{T}}(\mathbf{c}_k - \mathbf{c}), \tag{40}$$

where

$$\mathbf{c} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i = \frac{1}{M}\sum_{k=1}^{K} M_k\mathbf{c}_k \tag{41}$$

is the global mean attributes of $\mathbf{X}$.

Let's find a unit vector $\mathbf{v} \in \mathbb{R}^N$ and project data on that direction. The total within-class variation of the projection is

$$\sum_{k=1}^{K}\mathbf{v}^{\mathrm{T}}\mathbf{S}_k\mathbf{v} = \mathbf{v}^{\mathrm{T}}\mathbf{S}_w\mathbf{v}, \tag{42}$$

and the between-class variation of the projection is

$$\sum_{k=1}^{K} M_k \mathbf{v}^{\mathrm{T}}(\mathbf{c}_k - \mathbf{c})(\mathbf{c}_k - \mathbf{c})^{\mathrm{T}}\mathbf{v} = \mathbf{v}^{\mathrm{T}}\mathbf{S}_b\mathbf{v}. \tag{43}$$

The discriminant function $\mathcal{J}(\mathbf{v})$ is defined as

$$\mathcal{J}(\mathbf{v}) = \frac{\mathbf{v}^{\mathrm{T}}\mathbf{S}_b\mathbf{v}}{\mathbf{v}^{\mathrm{T}}\mathbf{S}_w\mathbf{v}}. \tag{44}$$

The maximization of the discrimination function

$$\max_{\mathbf{v}:\|\mathbf{v}\|_2=1} \mathcal{J}(\mathbf{v}) \tag{45}$$

identifies the unit vector that maximizes the separation of projection by classes. The solution to this optimization problem is an eigenvector of $\mathbf{S}_w^{-1}\mathbf{S}_b$.

The maximum value of the discriminant function is:

$$\mathcal{J}(\mathbf{v}^*) = \frac{\mathbf{v}^{*\mathrm{T}}\mathbf{S}_b\mathbf{v}^*}{\mathbf{v}^{*\mathrm{T}}\mathbf{S}_w\mathbf{v}^*} = \frac{\lambda^*\mathbf{v}^{*\mathrm{T}}\mathbf{S}_w\mathbf{v}^*}{\mathbf{v}^{*\mathrm{T}}\mathbf{S}_w\mathbf{v}^*} = \lambda^*. \tag{46}$$

That is, the maximum value for the discriminant function is the maximum eigenvalue of $\mathbf{S}_w^{-1}\mathbf{S}_b$.

How many discriminatory directions can be found? We just count the number of nonzero eigenvalues of $\mathbf{S}_w^{-1}\mathbf{S}_b$. One can find at most $K-1$ discriminatory directions. Here we skip the proof.

Rigorous proof of LDA method can be found in various references such as [1]. Students interested in this topic are encouraged to read related sections in those textbooks.

5.4 An Example for LDA

We have a dataset $\mathbf{X}$ with six data points in two class one or two:

$$\mathbf{X} = \begin{bmatrix} 1.0 & 2.0 \\ 2.0 & 3.0 \\ 3.0 & 4.9 \\ 2.0 & 1.0 \\ 3.0 & 2.0 \\ 4.0 & 3.9 \end{bmatrix} \tag{47}$$

$$\mathbf{y} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} \tag{48}$$

The mean attributes of class one is

$$\mathbf{c}_1 = [2.0\ 3.3], \tag{49}$$

and those of class two is

$$\mathbf{c}_2 = [3.0\ 2.3]. \tag{50}$$

The within-class scatter matrix is

$$\mathbf{S}_w = \begin{bmatrix} 4.00 & 5.80 \\ 5.80 & 8.68 \end{bmatrix} \tag{51}$$

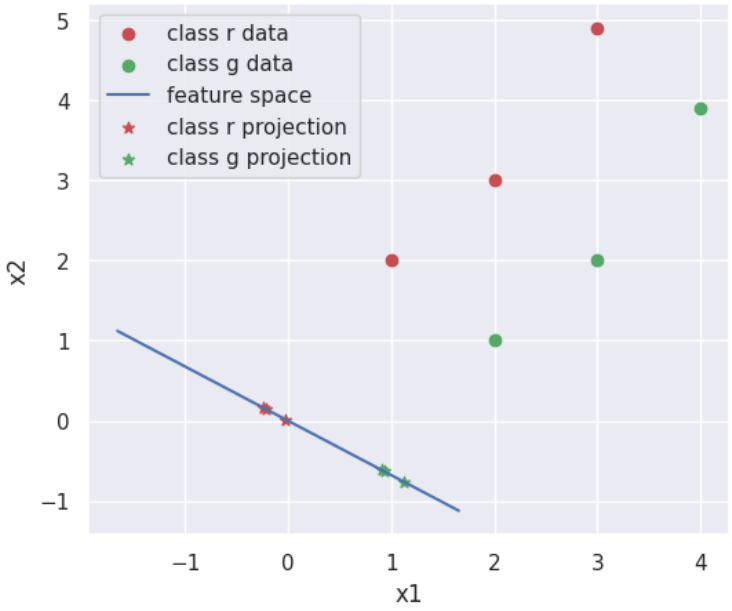


Figure 9: Visualization of the result from the LDA example

The between-class scatter matrix is

$$\mathbf{S}_b = \begin{bmatrix} 1.5 & -1.5 \\ -1.5 & 1.5 \end{bmatrix} \tag{52}$$

.

The eigenvector of $\mathbf{S}_w^{-1}\mathbf{S}_b$ can be found either through eigenvalue decomposition or generalized eigenvalue problem.

$$\mathbf{v} = \begin{bmatrix} -0.828 \\ 0.561 \end{bmatrix} \tag{53}$$

Figure 9 visualizes the result of the example.

6 Some Feature Extraction Methods Developed in Scikit Learn

There are many feature extraction and dimension reduction methods. The scikit learn library [5] has a variety of methods developed for use. Here, we briefly introduce several to broaden the options for use. Students can refer to the documentation of this ML library for the complete discussion.

6.1 Matrix Decomposition

The sklearn.decomposition module includes matrix decomposition algorithms, including PCA, Non-Negative Matrix Factorization (NMF) or Independent Component Analysis (ICA) among others. Most of the algorithms of this module can be regarded as dimensionality reduction techniques.

6.1.1 Incremental PCA

Incremental principal components analysis (IPCA) is about linear dimensionality reduction using Singular Value Decomposition (SVD) of the data, keeping only the most significant singular vectors to project the data to a lower dimensional space. The input data are centered but not scaled for each attribute before applying the SVD. Depending on the size of the input data, this algorithm can be much more memory efficient than a PCA, and allows sparse input.

[5] https://scikit-learn.org/stable/

6.1.2 Kernel PCA

Kernel PCA is an extension of PCA which achieves non-linear dimensionality reduction through the use of kernels. Specifically, it uses a nonlinear kernel function instead of the standard dot product to implicitly perform PCA in a possibly high dimensional space that is nonlinearly related to input space.

6.1.3 Sparse PCA

Sparse PCA is a variant of PCA, which aims to extract the set of sparse components that best reconstruct the data.

6.1.4 Truncated SVD

Truncated SDV performs linear dimensionality reduction by means of truncated SVD. Contrary to PCA, this estimator does not center the data before computing the singular value decomposition. This means it can work with sparse matrices efficiently.

6.1.5 Fast Independent Component Analysis

Independent Component Analysis (ICA) finds the independent components by maximizing the statistical independence of the estimated components. Fast ICA is an algorithm of ICA.

6.1.6 Minibatch Non-negative Matrix Factorization

Minibatch Non-negative Matrix Factorization (Minibatch NMF) finds two non-negative matrices, i.e. matrices with all non-negative elements, (W, H) whose product approximates the non-negative matrix X. This factorization can be used for various purposes including dimensionality reduction, source separation, or topic extraction.

6.2 Manifold Learning

Manifold learning is a class of unsupervised estimators that seeks to describe datasets as low-dimensional manifolds embedded in high-dimensional spaces.

6.2.1 Isomap Embedding

Isomap Embedding is non-linear dimensionality reduction through Isometric mapping.

6.2.2 Locally Linear Embedding

Locally linear embedding (LLE) seeks a lower-dimensional projection of the data which preserves distances within local neighborhoods. It can be thought of as a series of local PCAs which are globally compared to find the best non-linear embedding.

6.3 Discriminant Analysis

Discriminant analysis is statistical technique for classifying observations into non-overlapping groups, based on scores on one or more quantitative predictor variables.

6.3.1 Linear Discriminant Analysis

Linear discriminant analysis (LDA) can perform supervised dimensionality reduction, by projecting the input data to a linear subspace consisting of the directions which maximize the separation between classes. The dimension of the output is necessarily less than the number of classes, so this is in general a rather strong dimensionality reduction, and only makes sense in a multiclass setting. Linear discriminant analysis is a supervised dimensionality reduction technique whereas PCA is unsupervised.

6.3.2 Quadratic Discriminant Analysis

Quadratic Discriminant Analysis (QDA) is used to find a quadratic decision boundary between classes, whereas LDA finds a linear boundary.

6.4 Random Projection

Random projection implements a simple and computationally efficient way to reduce the dimensionality of the data by trading a controlled amount of accuracy (as additional variance) for faster processing times and smaller model sizes. Two types of unstructured random matrix: Gaussian random matrix and sparse random matrix, are introduced below.

6.4.1 Gaussian Random Projection

The Gaussian Random Projection reduces the dimensionality by projecting the original input space on a randomly generated matrix where components are drawn from the following distribution N(0, 1 / # components).

6.4.2 Sparse Random Projection

The Sparse Random Projection reduces the dimensionality by projecting the original input space using a sparse random matrix.

6.5 Neighbors Analysis

The neighors module of sklearn provides functionality for unsupervised and supervised neighbors-based learning methods. Unsupervised nearest neighbors is the foundation of many other learning methods, notably manifold learning and spectral clustering. Supervised neighbors-based learning comes in two flavors: classification for data with discrete labels, and regression for data with continuous labels.

6.5.1 Neighborhood Component Analysis

Neighborhood Components Analysis is a machine learning algorithm for metric learning. It learns a linear transformation in a supervised fashion to improve the classification accuracy of a stochastic nearest neighbors rule in the transformed space.

6.5.2 K Neighbors Classifier

K Neighbors Classifier implements learning based on the nearest neighbors of each query point, where is an integer value specified by the user.

6.6 Comparison of Feature Extraction Methods on the MNIST Dataset

Using the optical recognition of handwritten digits dataset in sklean as an example, we compare the above-discussed methods.

```
.. _digits_dataset:

Optical recognition of handwritten digits dataset
--------------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 1797
    :Number of Attributes: 64
    :Attribute Information: 8x8 image of integer pixels in the range 0..16.
    :Missing Attribute Values: None
    :Creator: E. Alpaydin (alpaydin '@' boun.edu.tr)
    :Date: July; 1998
```

```
This is a copy of the test set of the UCI ML hand-written digits datasets
https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

The data set contains images of hand-written digits: 10 classes where
each class refers to a digit.

Preprocessing programs made available by NIST were used to extract
normalized bitmaps of handwritten digits from a preprinted form. From a
total of 43 people, 30 contributed to the training set and different 13
to the test set. 32x32 bitmaps are divided into nonoverlapping blocks of
4x4 and the number of on pixels are counted in each block. This generates
an input matrix of 8x8 where each element is an integer in the range
0..16. This reduces dimensionality and gives invariance to small
distortions.

For info on NIST preprocessing routines, see M. D. Garris, J. L. Blue, G.
T. Candela, D. L. Dimmick, J. Geist, P. J. Grother, S. A. Janet, and C.
L. Wilson, NIST Form-Based Handprint Recognition System, NISTIR 5469,
1994.

.. topic:: References

  - C. Kaynak (1995) Methods of Combining Multiple Classifiers and Their
    Applications to Handwritten Digit Recognition, MSc Thesis, Institute of
    Graduate Studies in Science and Engineering, Bogazici University.
  - E. Alpaydin, C. Kaynak (1998) Cascading Classifiers, Kybernetika.
  - Ken Tang and Ponnuthurai N. Suganthan and Xi Yao and A. Kai Qin.
    Linear dimensionalityreduction using relevance weighted LDA. School of
    Electrical and Electronic Engineering Nanyang Technological University.
    2005.
  - Claudio Gentile. A New Approximate Maximal Margin Classification
    Algorithm. NIPS. 2000.
```

Figure 10 below shows a sample from the MNIST dataset.

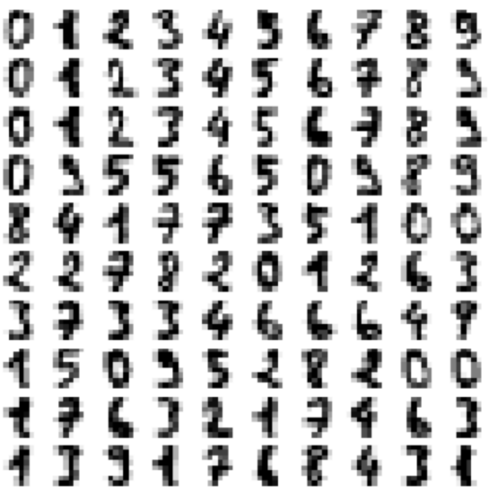


Figure 10: 100 examples from the MNIST dataset

Figure 11 shows reconstructed data using the two features extracted by different methods. The reconstructed data by each method are used as feature inputs to train an KNN classier, with the test accuracy displayed on top of each subplot. The higher the test accuracy, the more informative the data reconstructed. What methods are better than others in this example?

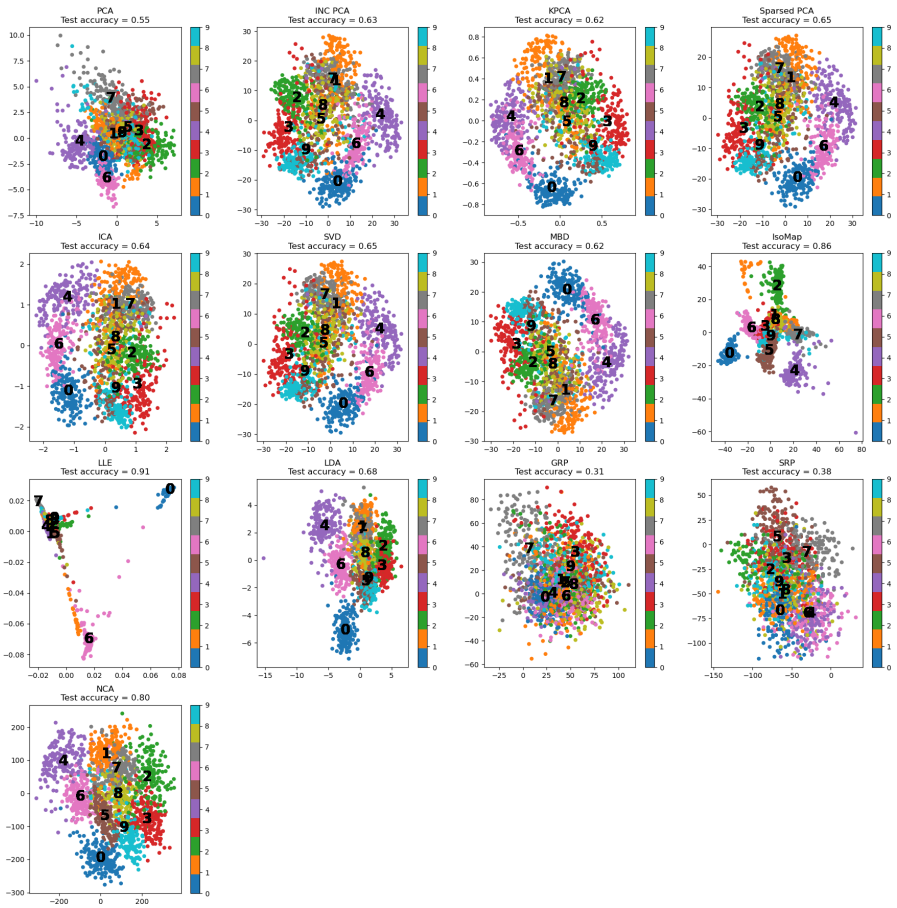


Figure 11: Method Comparison

References

[1] Trevor Hastie. The elements of statistical learning: data mining, inference, and prediction, 2009.

[2] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.