Machine Learning

Lecture 10

Min-Kuan Chang

minkuanc@nchu.edu.tw

EE, College of EECS

Dimensionality Reduction

Topic 01

Introduction

What Dimensionality Is

- Dimensionality is defined as the number of features in an example in the dataset
  - for example:
    - Iris plants dataset:
      - each example has four attributes, sepal length, sepal width, petal length, petal width
      - its dimensionality is 4
    - Diabetes dataset:
      - each example has age in years, sex, body mass index, average blood pressure, total serum cholesterol, low-density lipoproteins, high-density lipoproteins, total cholesterol / HDL, possibly log of serum triglycerides level, blood sugar level
      - its dimensionality is 10
    - MNIST dataset:
      - each picture contains 28*28 pixels
      - its dimensionality is 784

Why Dimensionality Reduction Needed

- The complexity in most learning algorithms depends on the number of input dimensions
  - decreasing dimension also decreases the complexity of the inference algorithm during testing
- Data can be explained with fewer features
  - a better idea about the process that underlies the data
  - help knowledge extraction
  - visualization
- Save the memory space for storage and speed up the learning algorithm

Why Dimensionality Reduction Needed

- More input features often make a predictive modeling task more challenging to model
  - the curse of dimensionality
- Simpler models are more robust on small datasets
  - fewer input dimensions often mean fewer parameters or a simpler structure
    - less degrees of freedom
    - less chance to overfit the training data
    - better generalization to new data
  - simpler models have less variance
    - they vary less depending on the particulars of a sample, including noise, outliers, and so forth
  - limit the feature dimensionality to maintain the generalization performance

Why Dimensionality Reduction Needed

- Dimensionality reduction refers to techniques for reducing the number of input variables in training data
  - a data preparation technique performed on data prior to modeling
  - project the data to a lower dimensional subspace while preserve the “essence” of the data
- Two main methods for reducing dimensionality
  - feature selection and feature extraction

Dimensionality Reduction – Feature Selection

- Yield a subset of features from the original set of features
  - the selected features are best representatives of the data
- Always done in the context of an optimization problem
- Feature selection methods
  - filter methods
    - features are filtered based on general characteristics (such as correlation) of the dataset
  - wrapper methods
    - the predictive model algorithm exists to identify the most useful features
  - embedded methods
    - feature selection process is embedded in the learning or the model building phase

Dimensionality Reduction – Feature Extraction

- Involve a transformation of the features
  - often is not reversible
  - some information is lost in the process of dimensionality reduction
- A new set of features that are combinations of the original features
- These methods may be supervised or unsupervised depending on whether or not they use the output information
- Feature extraction methods
  - Principal Components Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
  - Factor Analysis (FA)
  - Multidimensional Scaling(MDS)

Pros And Cons Of Dimensionality Reduction

- Advantages
  - alleviate the curse of dimensionality
  - reduction in the space required to store the dataset
  - less computation training time
  - easy to visualizing the data
  - less redundant features
  - less noise in dataset
- Disadvantages
  - information loss due to dimensionality reduction
  - unknown principal components when using PCA dimensionality reduction technique

Illustration of Dimensionality Reduction

	Apple	Pear	Banana	Kiwi
John	10	2	0	0
Bob	2	5	2	2
Mary	3	0	0	0
Sue	0	2	2	2

Illustration of Dimensionality Reduction

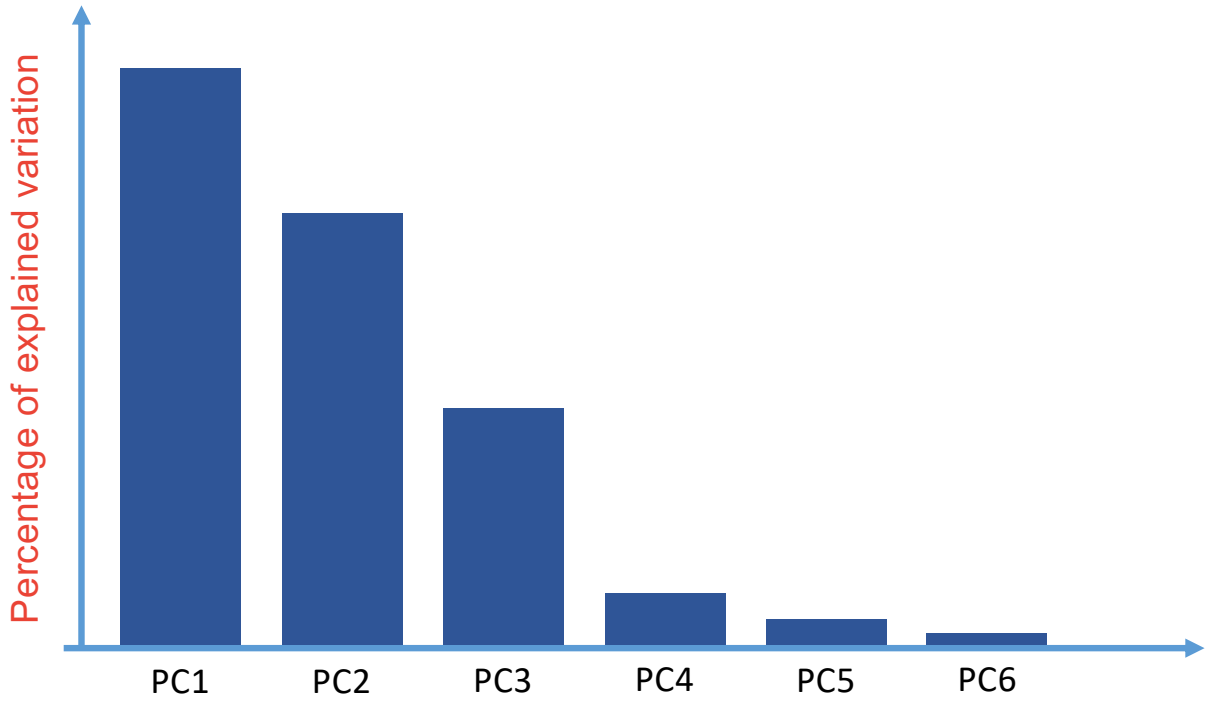
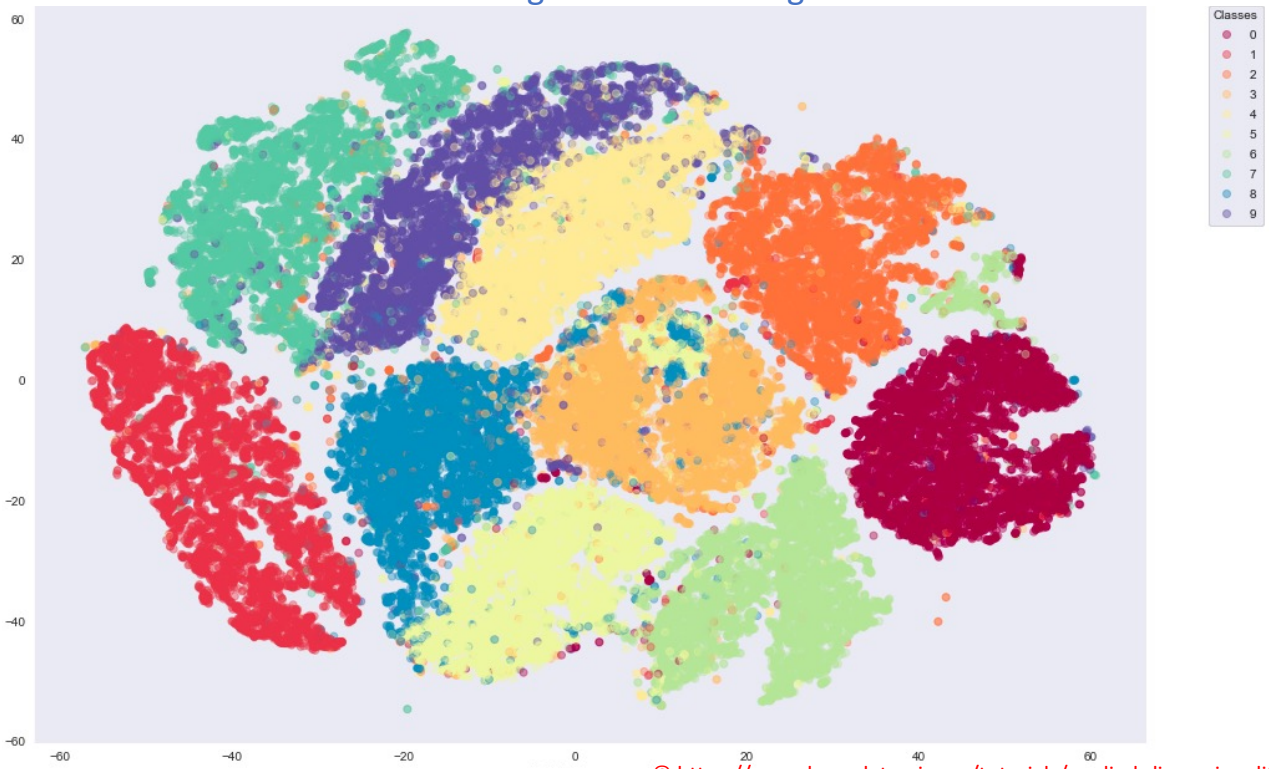


Illustration of Dimensionality Reduction

T-distributed Stochastic Neighbor Embedding on MNIST Dataset



Dimensionality Reduction

Topic 02

Feature Selection

Approaches in Feature Selection

- Feature selection can be viewed as an optimization problem
- Suppose the number of feature is D. The search space for this optimization is 2^D
- The approaches to feature subset selection
  - a filter uses some evaluation functions to perform the feature subset selection
    - this function is independent of the learning algorithm
  - a wrapper uses the same machine learning algorithm for modeling to evaluate the feature subsets
  - a embedded approach performs feature selection during model generation
    - not a pre-processing module before the model generation

Common Search Strategies

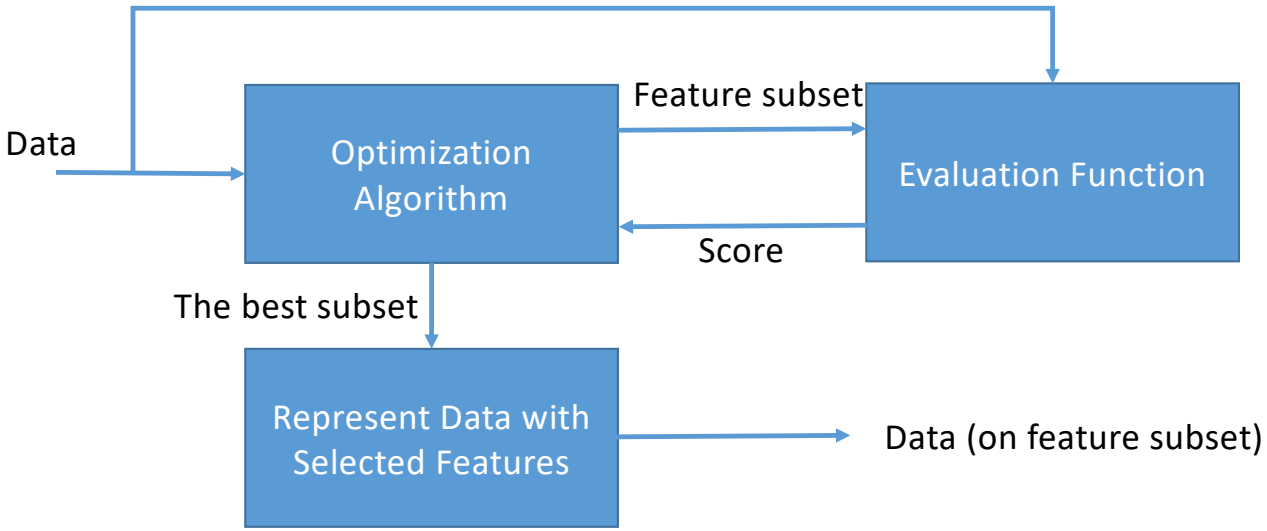
- Forward selection
  - the feature set is set to be empty at the beginning
  - during the process, greedily add one feature at a time
- Backward elimination
  - the feature set contains all the features at the beginning
  - during the process, greedily remove one feature at a time
- Forward stepwise selection
  - the feature set is set to be empty at the beginning
  - during the process, greedily add or remove one feature at a time

Common Search Strategies

- Backward stepwise elimination
  - the feature set contains all the features at the beginning
  - during the process, greedily add or remove one feature at a time
- Random mutation
  - randomly select a subset of features at the beginning
  - add or remove one feature at a time and stop after a certain number of iterations

Filtering Approach

- The flowchart



Filtering Approach

- Approach I
  - backward elimination is adopted to eliminate predefined number of features
  - select a subset of features that keeps the class probability distribution as close as possible to the original distribution
  - the cross entropy is used as the evaluate function to determine the next feature to be deleted

Filtering Approach

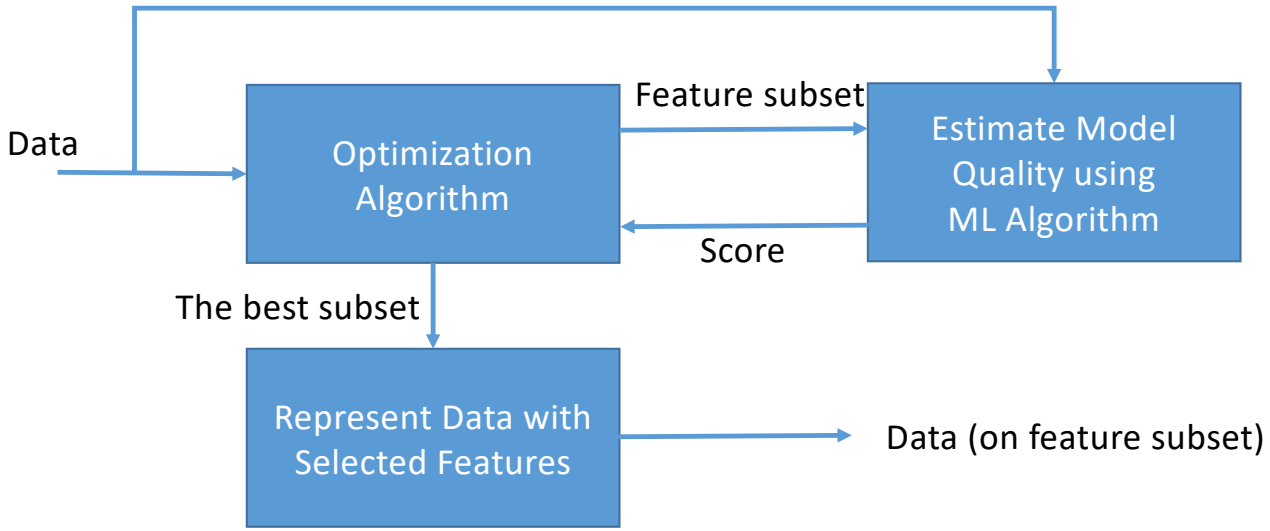
- Approach II
  - use random sampling to search the space of all possible feature subsets
  - the predetermined number of subsets is evaluated
  - the evaluation function is chosen to be the inconsistency rate
    - for all the matching instances without considering their class labels the inconsistency count is the number of the instances minus the largest number of instances of class labels
    - the inconsistency rate is the sum of all the inconsistency counts divided by the total number of instances

Filtering Approach

- Advantages
  - computationally light
  - avoids overfitting
  - do not rely on learning algorithms
    - biased which is equivalent to changing data to fit the learning algorithm
- Disadvantages
  - the selected subset might not be optimal in that a redundant subset might be obtained

Wrapper Approach

- The flowchart



- The wrapper approach can be classified into sequential selection algorithms and heuristic search algorithms

Wrapper Approach

- The sequential feature selection algorithm
  - start with an empty set and adds one feature for the first step
    - this added feature gives the highest value for the objective function
    - the remaining features are added individually to the current subset and the new subset is evaluated
  - the process is repeated until the required number of features are added
- The sequential backward selection algorithm
  - similar to SFS
  - the algorithm starts from the complete set of variables and removes one feature at a time
    - this removed feature gives the lowest decrease in predictor performance

Wrapper Approach

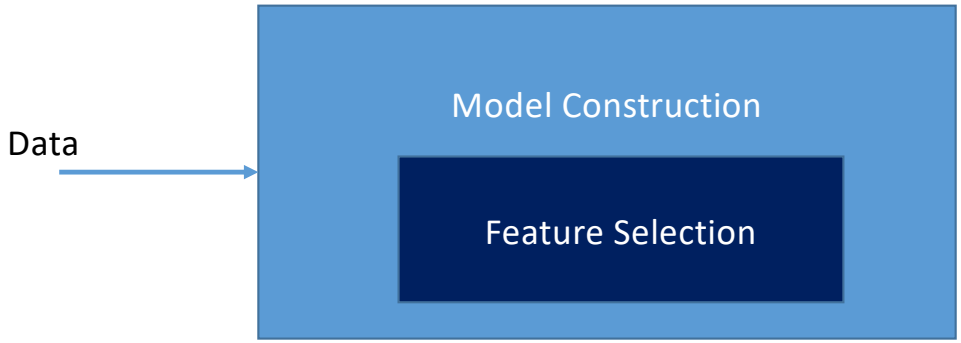
- The sequential floating forward selection
  - more flexible than the sequential feature selection algorithm
  - an additional backtracking step follows the adding step
    - exclude one feature at a time from the subset obtained in the first step and evaluates the new subsets
    - if excluding a feature increases the value of the objective function then that feature is removed and goes back to the adding step with the new reduced subset
    - otherwise, go to the adding step
  - the process is repeated until the required number of features are added

Wrapper Approach

- The main drawback of Wrapper methods is the number of computations
- Another drawback of using the model performance as the objective function is prone to overfitting
  - subset selection can result in a bad feature subset with high performance but poor generalization power
  - to avoid this, a separate holdout test set can be used to guide the performance of the search

Embedded Approach

- The flowchart



Embedded Approach

- Advantages
  - take into consideration the interaction of features like wrapper methods do
  - faster than filter methods
  - more accurate than filter methods
  - find the feature subset for the algorithm being trained
  - much less prone to over-fitting than the wrapper approach

Principal Component Analysis
Topic 01
Fundamentals

- A common problem in statistical pattern recognition is that of feature extraction
  - high-dimensional data is hard to analysis
  - high-dimensional data sometime is overcomplete
    - many dimensions are redundant
    - the redundant dimensions can be explained by a combination of other dimensions
  - storage of data can be expensive
  - feature extraction can help

- Feature extraction
  - a process whereby a data space is transformed into a feature space
  - the data set may be represented by a reduced number of “effective” features
    - most of the intrinsic information content of the original data is preserved
  - the data set undergoes a dimensionality reduction
- Thus, in the problem of feature extraction, we need to find an optimal transformation such that the difference between the transformed data and the original one can be minimal in the mean-square-error sense

- Suppose $\mathbf{X}$ be an D-dimensional random vector and $\mathbb{E}\{\mathbf{X}\} = 0$
  - $\boldsymbol{x}$ is a realization of the random vector $\mathbf{X}$
- Let $\boldsymbol{q}$ be a unit vector of dimension D.
  The projection of $\boldsymbol{x}$ onto $\boldsymbol{q}$ is simple the inner product of them

$$p = \boldsymbol{x}^T\boldsymbol{q} = \boldsymbol{q}^T\boldsymbol{x}$$

  - the mean of p

E{x} 是向量

$$\mathbb{E}\{p\} = \mathbb{E}\{\boldsymbol{x}^T\boldsymbol{q}\} = \mathbb{E}\{\boldsymbol{q}^T\boldsymbol{x}\} = \boldsymbol{q}^T\mathbb{E}\{\boldsymbol{x}\} = 0$$

  - the variance of p

Correlation matrix

$$\sigma_p^2 = Var(p) = \mathbb{E}\{p^2\} = \mathbb{E}\{p^Tp\} = \mathbb{E}\{(\boldsymbol{x}^T\boldsymbol{q})^T(\boldsymbol{x}^T\boldsymbol{q})\} = \boldsymbol{q}^T\,\mathbb{E}\{\boldsymbol{x}\boldsymbol{x}^T\}\boldsymbol{q} = \boldsymbol{q}^T\,\mathbf{R}_\mathbf{X}\boldsymbol{q}$$

$\theta(\boldsymbol{q}) = \sigma_p^2$ **is called a variance probe**

Eigenstructure of PCA

- Next we would like to find the unit vectors such that $\theta(\boldsymbol{q})$ has extremal or stationary values
- The solution to this problem lies in the eigenstructure of the correlation matrix $\mathbf{R_X}$
- If we apply a very small perturbation $\delta\boldsymbol{q}$ to the unit vector $\boldsymbol{q}$, the variance probe becomes $\theta(\boldsymbol{q}+\delta\boldsymbol{q})$

$$\theta(\boldsymbol{q}+\delta\boldsymbol{q}) = (\boldsymbol{q}+\delta\boldsymbol{q})^T\mathbf{R_X}(\boldsymbol{q}+\delta\boldsymbol{q}) = \boldsymbol{q}^T\,\mathbf{R_X}\boldsymbol{q} + 2(\delta\boldsymbol{q})^T\,\mathbf{R_X}\boldsymbol{q} + (\delta\boldsymbol{q})^T\,\mathbf{R_X}\delta\boldsymbol{q}$$

$$\cong \boldsymbol{q}^T\,\mathbf{R_X}\boldsymbol{q} + 2(\delta\boldsymbol{q})^T\,\mathbf{R_X}\boldsymbol{q}$$

Eigenstructure of PCA

- When $\theta(\boldsymbol{q})$ is the extreme value, to the first order approximation, we have

$$\theta(\boldsymbol{q}+\delta\boldsymbol{q}) \approx \theta(\boldsymbol{q})$$

- This leads to

$$(\delta\boldsymbol{q})^T \mathbf{R}_{\mathbf{X}}\boldsymbol{q} = 0$$

under the condition that $\boldsymbol{q}+\delta\boldsymbol{q}$ is also a unit vector

- The fact $\boldsymbol{q}+\delta\boldsymbol{q}$ is a unit vector and $\delta\boldsymbol{q}$ is very small gives us

$$(\delta\boldsymbol{q})^T\boldsymbol{q} = 0$$

Eigenstructure of PCA

- Combing these two equations, we have

$$(\delta \boldsymbol{q})^T \mathbf{R_X} \boldsymbol{q} - \lambda (\delta \boldsymbol{q})^T \boldsymbol{q} = 0$$

$$(\delta \boldsymbol{q})^T (\mathbf{R_X} \boldsymbol{q} - \lambda \boldsymbol{q}) = 0$$

$$\mathbf{R_X} \boldsymbol{q} - \lambda \boldsymbol{q} = 0$$

$$\mathbf{R_X} \boldsymbol{q} = \lambda \boldsymbol{q}$$

兩者都趨近於0 因此有同樣的量級，中間差一個倍率 lambda。

- The last equation tells us for the variance probe to have the extreme values, $\mathbf{R_X} \boldsymbol{q} = \lambda \boldsymbol{q}$ must be fulfilled

Eigenstructure of PCA

- Furthermore, $\mathbf{R_X}\boldsymbol{q} = \lambda\boldsymbol{q}$ tells us that for the variance probe to have the extreme values, $\boldsymbol{q}$ must be the eigenvector if $\mathbf{R_X}$
- Let $\lambda_1, \lambda_2, \cdots, \lambda_D$ be the eigenvalues of the $D \times D$ matrix $\mathbf{R_X}$ and we let $\boldsymbol{q}_i$ be the respective eigenvectors of λ_i with $\|\boldsymbol{q}_i\| = 1$
  - $\lambda_1, \lambda_2, \cdots, \lambda_D$ are arranged in the descending order

$$\lambda_1 > \lambda_2 > \cdots > \lambda_D$$

  - Let $\mathbf{\Lambda}$ be a diagonal matrix whose (i, i) entry is λ_i

$$\mathbf{\Lambda} = \text{diag}[\lambda_1, \lambda_2, \cdots, \lambda_D]$$

  - Let $\mathbf{Q}$ be an $D \times D$ matrix with $\boldsymbol{q}_i$ being in its ith column

$$\mathbf{Q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_D]$$

$$\mathbf{R_X Q} = \mathbf{Q\,\Lambda}$$

Eigenstructure of PCA

- Due to the fact that $\lambda_1 > \lambda_2 > \cdots > \lambda_D$, we have the following properties
  - **Q** is an orthogonal matrix

$$\mathbf{Q}\mathbf{Q}^T = \mathbf{I} \qquad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}$$

  - the orthogonal similarity transformation

$$\mathbf{Q}^T\mathbf{R}_\mathbf{X}\mathbf{Q} = \mathbf{\Lambda} \quad \Rightarrow \quad \boldsymbol{q}_j^T\mathbf{R}_\mathbf{X}\boldsymbol{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases}$$

  - the spectral theorem

$$\mathbf{R}_\mathbf{X} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T = \sum_{i=1}^{m} \lambda_i \, \boldsymbol{q}_i\boldsymbol{q}_i^T$$

Eigenstructure of PCA

- The principle component analysis and the eigendecomposition of $\mathbf{R_X}$ are the same
- Recall the variance probe

$$\theta(\boldsymbol{q}) = \boldsymbol{q}^T \mathbf{R_X} \boldsymbol{q}$$

- For an eigenvector $\boldsymbol{q}_i$ of $\mathbf{R_X}$, the variance probe is

$$\theta(\boldsymbol{q}_i) = \boldsymbol{q}_i^T \mathbf{R_X} \boldsymbol{q}_i = \boldsymbol{\lambda}_i$$

  - the eigenvectors of $\mathbf{R_X}$ can be used to define $\boldsymbol{q}_i$ along which the variance probes have their extremal values
  - the associated eigenvalues represents the extremal values of the variance probes

Basic Data Representations

- Let the data vector $\boldsymbol{x}$ denote a realization of the random vector $\mathbf{X}$
  - $\mathbb{E}\{\mathbf{X}\} = 0$
  - the correlation matrix $\mathbf{R}_{\mathbf{X}}$
    - normalized eigenvectors $\boldsymbol{q}_i$ for $i = 1,2,\cdots,D$
    - the corresponding eigenvalues λ_i for $i = 1,2,\cdots,D$, which is in the descending order
- Let y_i be the project of the data vector $\boldsymbol{x}$ onto $\boldsymbol{q}_i$

$$y_i = \boldsymbol{x}^T\boldsymbol{q}_i = \boldsymbol{q}_i^T\boldsymbol{x} \qquad i = 1,2,\cdots,D$$

  - y_i is called the principal components

每個特徵向量 內積輸入資料x 會得到的那些值稱為 principal components

Basic Data Representations

- Let $\boldsymbol{y} = [y_1, y_2, \cdots, y_m]^T$

$$\boldsymbol{y} = \mathbf{Q}^T \boldsymbol{x}$$

  - we can view this is a transformation
  - the data vector is transformed from the data space to the feature space
- In addition, we can reconstruct $\boldsymbol{x}$ from $\boldsymbol{y}$

$$\boldsymbol{x} = \mathbf{Q}\boldsymbol{y}$$

哇

Principal Component Analysis
Topic 02
The Dimensionality Reduction – Part I

PCA: Eigenvectors' Perspective

- PCA is also known as Karhunen-Loeve transform
- PCA provides an effective techniques for dimensionality reduction
  - discard those dimensions that are combinations of other dimensions
  - obtain effective data representation
- Let $\lambda_1, \lambda_2, \cdots, \lambda_l$ denote the largest l eigenvalues of $\mathbf{R_X}$, we can approximate the data vector $\boldsymbol{x}$ by using the corresponding l eigenvectors

$$\boldsymbol{x} = \sum_{i=1}^{D} y_i \boldsymbol{q}_i \xrightarrow{\text{truncate}} \hat{\boldsymbol{x}} = \sum_{i=1}^{l} y_i \boldsymbol{q}_i \qquad y_i = \boldsymbol{x}^T \boldsymbol{q}_i$$

字幕預留位置

PCA: Eigenvectors' Perspective

- The projection of $\boldsymbol{x}$ onto the first l eigenvectors is to map from the data vector to the feature space

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix}$$

Vector of principal components

Data space

Feature space

PCA: Eigenvectors' Perspective

- Encoder

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_1^T \boldsymbol{x} \\ \boldsymbol{q}_2^T \boldsymbol{x} \\ \vdots \\ \boldsymbol{q}_l^T \boldsymbol{x} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{q}_1^T \\ \boldsymbol{q}_2^T \\ \vdots \\ \boldsymbol{q}_l^T \end{bmatrix}}_{\text{Encoder}} \boldsymbol{x}$$

- Decoder

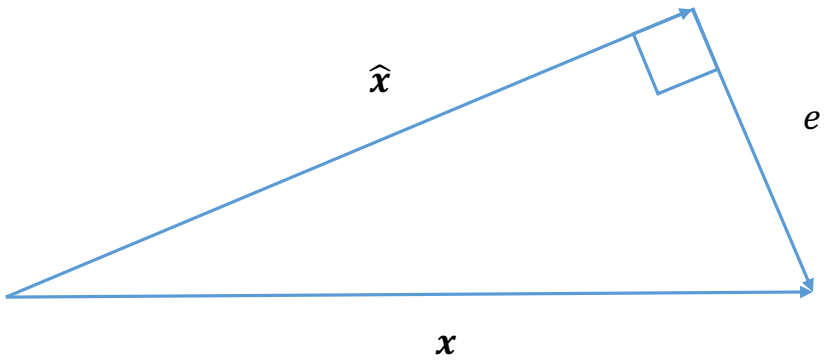
$$\hat{\boldsymbol{x}} = \sum_{i=1}^{l} y_i \boldsymbol{q}_i = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \vdots \\ \hat{x}_l \end{bmatrix} = \underbrace{[\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_l]}_{\text{Decoder}} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix}$$

PCA: Eigenvectors' Perspective

- The approximation error vector

$$e = \boldsymbol{x} - \hat{\boldsymbol{x}} = \sum_{i=1}^{D} y_i \boldsymbol{q}_i - \sum_{i=1}^{l} y_i \boldsymbol{q}_i = \sum_{i=l+1}^{D} y_i \boldsymbol{q}_i$$

- Since $\boldsymbol{q}_i$'s are orthogonal to one another, e is orthogonal to $\hat{\boldsymbol{x}}$
  - the principle of orthogonality



PCA: Eigenvectors' Perspective

- Next, we look at PCA from the perspective of the total variance
  - the total variance of $\boldsymbol{x}$ is defined as

$$\mathbb{E}\{\boldsymbol{x}^T\boldsymbol{x}\} = \mathbb{E}\left\{\left(\sum_{i=1}^{D} y_i\boldsymbol{q}_i\right)^T\left(\sum_{i=1}^{D} y_i\boldsymbol{q}_i\right)\right\} = \sum_{i=1}^{D}\mathbb{E}\{y_i^2\} = \sum_{i=1}^{D}\sigma_{y_i}^2 = \sum_{i=1}^{D}\lambda_i$$

  - the total variance of $\hat{\boldsymbol{x}}$ is defined as

$$\mathbb{E}\{\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}}\} = \mathbb{E}\left\{\left(\sum_{i=1}^{l} y_i\boldsymbol{q}_i\right)^T\left(\sum_{i=1}^{l} y_i\boldsymbol{q}_i\right)\right\} = \sum_{i=1}^{l}\mathbb{E}\{y_i^2\} = \sum_{i=1}^{l}\sigma_{y_i}^2 = \sum_{i=1}^{l}\lambda_i$$

  - the total variance of the approximation error e

$$\sum_{i=l+1}^{D}\sigma_{y_i}^2 = \sum_{i=l+1}^{D}\lambda_i$$

PCA: Eigenvectors' Perspective

- To preserving the information content of the original data, we will need the total variance of approximation error as close to zero as possible
- When performing the dimensionality reduction, we seek for the transformation consisting of those eigenvectors belonging to the dominant eigenvalues
  - this approach is also known as the subspace decomposition
- So far, we still do not answer the question that using the subspace decomposition can achieve the **maximal variance** of the projected data

PCA: Optimization's Perspective

- Consider an i.i.d dataset $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}, \boldsymbol{x}_i \in \mathbb{R}^D$ with mean 0 and the data covariance matrix

$$\mathbf{R}_{\mathbf{X}}^{s} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^T$$

- We assume there exists a low-dimensional representation of x_i

$$\boldsymbol{y}_i = \mathbf{P}^T \boldsymbol{x}_i$$

where $\mathbf{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_l]$ is the $D \times l$ projection matrix and $\boldsymbol{p}_i$'s are of length 1 and orthogonal to one another

PCA: Optimization's Perspective

- We start with the first coordinate of all projected vector $\boldsymbol{y}_i$'s
- The variance of the first coordinate

$$\frac{1}{N}\sum_{i=1}^{N} y_{i1}^2$$

i是sample

where y_{i1} is the first coordinate of $\boldsymbol{y}_i$ and $y_{i1} = \boldsymbol{p}_1^T \boldsymbol{x}_i$

- This variance can be rewritten as

$$\frac{1}{N}\sum_{i=1}^{N} y_{i1}^2 = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{p}_1^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{p}_1 = \boldsymbol{p}_1^T \mathbf{R}_{\mathbf{X}}^s \boldsymbol{p}_1$$

PCA: Optimization's Perspective

- To find the best $\boldsymbol{p}_1$, we formulate the following constrained optimization problem

Maximal variance optimization $\max_{\boldsymbol{p}_1} \boldsymbol{p}_1^T \mathbf{R}_{\mathbf{X}}^s \boldsymbol{p}_1$

最大化variance，
找到variance最大的是最能找到區分項目的(?

subject to $\|\boldsymbol{p}_1\|^2 = 1$

- Using the Lagrange multiplier, this optimization problem becomes

$$\max_{\boldsymbol{p}_1} \boldsymbol{p}_1^T \mathbf{R}_{\mathbf{X}}^s \boldsymbol{p}_1 + \lambda(1 - \boldsymbol{p}_1^T \boldsymbol{p}_1)$$

PCA: Optimization's Perspective

- The optimal $\boldsymbol{p}_1$ should satisfy the two equations listed below
  - the derivative of the objective function with respect to $\boldsymbol{p}_1$ should be a zero vector

$$\frac{\partial}{\partial \boldsymbol{p}_1}[\boldsymbol{p}_1^T \mathbf{R}_{\mathbf{X}}^S \boldsymbol{p}_1 + \lambda(1 - \boldsymbol{p}_1^T \boldsymbol{p}_1)] = \mathbf{0} \longrightarrow \mathbf{R}_{\mathbf{X}}^S \boldsymbol{p}_1 = \boldsymbol{\lambda} \boldsymbol{p}_1$$

  - the derivative of the objective function with respect to λ should be zero

$$1 - \boldsymbol{p}_1^T \boldsymbol{p}_1 = \mathbf{0} \longrightarrow \boldsymbol{p}_1^T \boldsymbol{p}_1 = 1$$

- To yield the largest variance, $\boldsymbol{p}_1$ should be the eigenvector of $\mathbf{R}_{\mathbf{X}}^S$ which corresponds to the largest eigenvalue of $\mathbf{R}_{\mathbf{X}}^S$
  - $\boldsymbol{p}_1$ is called the first principal component

PCA: Optimization's Perspective

- The projected data point of $\boldsymbol{x}_i$ onto $\boldsymbol{p}_1$ is

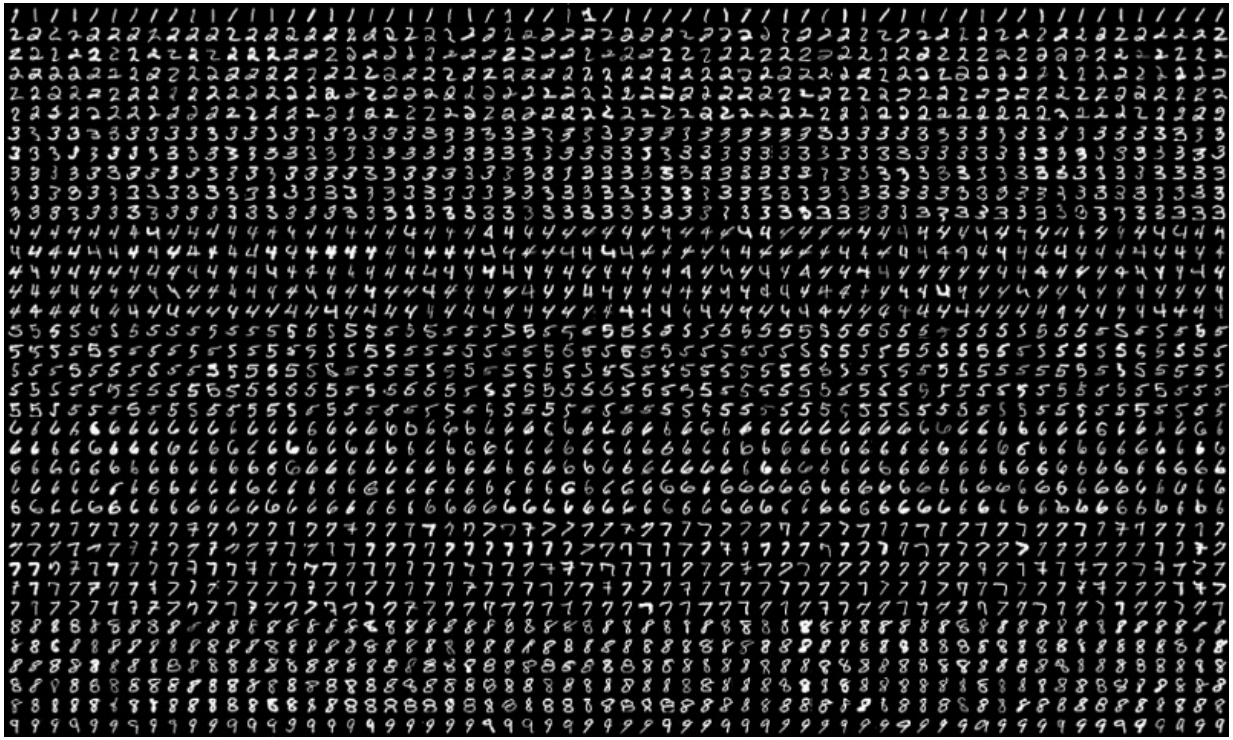
yp = x

$$\underset{\text{y}}{(\boldsymbol{p}_1^T \boldsymbol{x}_i)\boldsymbol{p}_1} = \boldsymbol{p}_1 \boldsymbol{p}_1^T \boldsymbol{x}_i$$

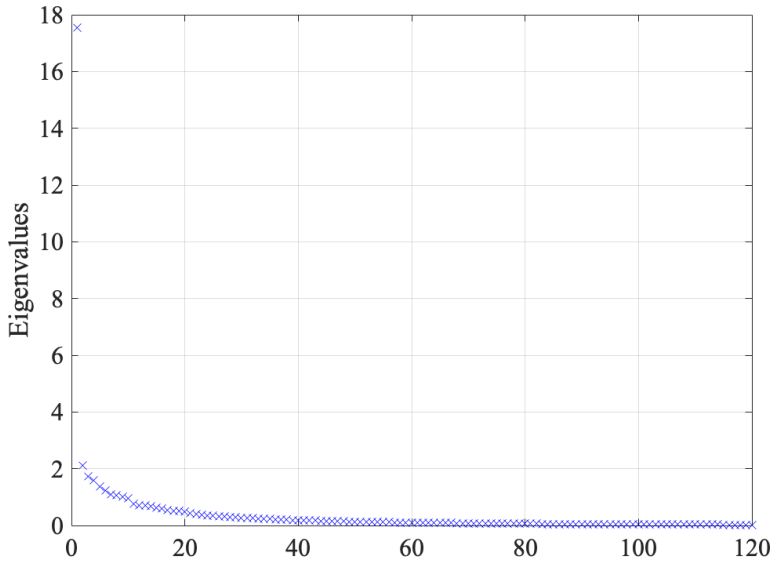
- From this perspective, we can find the first l principal components as the l eigenvectors of $\mathbf{R}_{\mathbf{X}}^{S}$ that are associated with the first l eigenvalues and the projected data point of $\boldsymbol{x}_i$ onto these l eigenvectors is

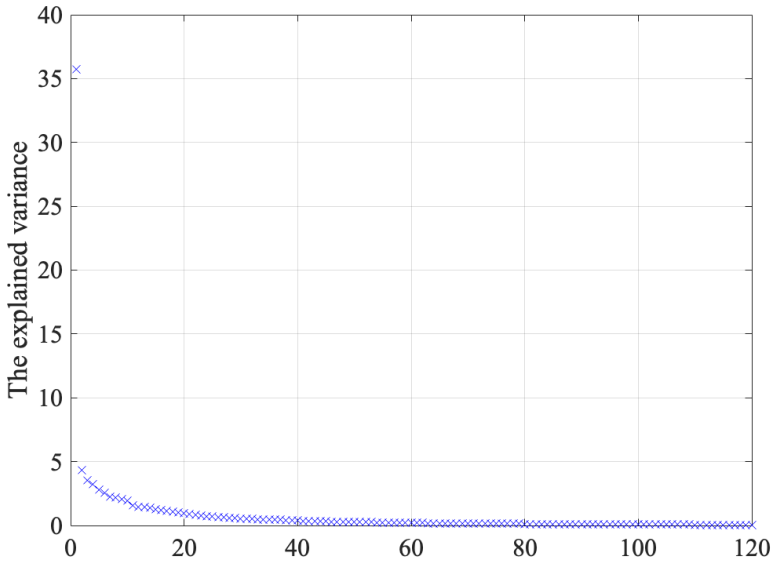
$$\sum_{i=1}^{l} \boldsymbol{p}_i \boldsymbol{p}_i^T \boldsymbol{x}_i$$

Example of MNIST



Example of MNIST





Principal Component Analysis
Topic 03
The Dimensionality Reduction – Part II

PCA: Projection Perspective

- Assume an orthonormal basis $\mathbf{Q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_D]$ of $\mathbb{R}^D$
- Any $\boldsymbol{x} \in \mathbb{R}^m$ can be written as a linear combination of the basis vectors

$$\boldsymbol{x} = \sum_{i=1}^{D} y_i \boldsymbol{q}_i = \sum_{i=1}^{l} y_i \boldsymbol{q}_i + \sum_{i=l+1}^{D} y_i \boldsymbol{q}_i$$

- We are interested in finding

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{l} z_i \boldsymbol{q}_i$$

which is as similar to $\boldsymbol{x}$ as possible

PCA: Projection Perspective

- Suppose $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N], \boldsymbol{x}_i \in \mathbb{R}^D$, the goal is to minimize the reconstruction error

$$\min_{z_{ni}} J = \frac{1}{N} \sum_{n=1}^{N} \|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2$$

where

$$\boldsymbol{x_n} = \sum_{i=1}^{D} y_{ni} \boldsymbol{q}_i \qquad \hat{\boldsymbol{x}}_n = \sum_{i=1}^{l} z_{ni} \boldsymbol{q}_i$$

PCA: Projection Perspective

- The optimal z_{ni} can be found via

$$\frac{\partial J}{\partial z_{ni}} = \frac{\partial}{\partial z_{ni}}\left[\frac{1}{N}\sum_{n=1}^{N}\|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2\right] = 0$$

$$\frac{\partial J}{\partial z_{ni}} = \frac{\partial \hat{\boldsymbol{x}}_n}{\partial z_{ni}}\frac{\partial J}{\partial \hat{\boldsymbol{x}}_n} = 0$$

$$\frac{\partial \hat{\boldsymbol{x}}_n}{\partial z_{ni}} = \boldsymbol{q}_i^T \qquad \frac{\partial J}{\partial \hat{\boldsymbol{x}}_n} = -\frac{2}{N}(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n)$$

PCA: Projection Perspective

- This concludes

$$\boldsymbol{q}_i^T\left(-\frac{2}{N}(\boldsymbol{x}_n-\widehat{\boldsymbol{x}}_n)\right)=0 \quad \Rightarrow \quad \boldsymbol{q}_i^T\boldsymbol{x}_n=\boldsymbol{q}_i^T\widehat{\boldsymbol{x}}_n$$

- From $\boldsymbol{q}_i^T\boldsymbol{x}_n=\boldsymbol{q}_i^T\widehat{\boldsymbol{x}}_n$, we have

$$z_{ni}=\boldsymbol{q}_i^T\boldsymbol{x}_n$$

PCA: Projection Perspective

- $z_{ni} = \boldsymbol{q}_i^T \boldsymbol{x}_n$ tells us
  - the optimal linear project of $\boldsymbol{x}_n$ is an orthogonal projection
  - the coordinates of $\hat{\boldsymbol{x}}_n$ are the coordinates of the orthogonal projection of $\boldsymbol{x}_n$ onto the span of $\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_l$, which is called the principal subspace
  - An orthogonal projection is the best linear mapping to minimize the MSE between $\boldsymbol{x}_n$ and $\hat{\boldsymbol{x}}_n$

- The best basis vectors $\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_l$ of the principal subspace needs to be determined

PCA: Projection Perspective

- To determine the best principal subspace, we first rewrite $\hat{\boldsymbol{x}}_n$ and $\boldsymbol{x}_n$

$$\hat{\boldsymbol{x}}_n = \sum_{i=1}^{l} z_{ni}\boldsymbol{q}_i = \sum_{i=1}^{l} \boldsymbol{q}_i^T \boldsymbol{x}_n \boldsymbol{q}_i = \sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T \boldsymbol{x}_n = \left(\sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T \right) \boldsymbol{x}_n$$

$$\boldsymbol{x}_n = \sum_{i=1}^{D} y_i \boldsymbol{q}_i = \sum_{i=1}^{D} \boldsymbol{q}_i^T \boldsymbol{x}_n \boldsymbol{q}_i = \left(\sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T \right) \boldsymbol{x}_n + \left(\sum_{i=l+1}^{D} \boldsymbol{q}_i \boldsymbol{q}_i^T \right) \boldsymbol{x}_n$$

- The difference vector between $\hat{\boldsymbol{x}}_n$ and $\boldsymbol{x}_n$ is

$$\left(\sum_{i=l+1}^{D} \boldsymbol{q}_i \boldsymbol{q}_i^T \right) \boldsymbol{x}_n$$

PCA: Projection Perspective

- Thus, the projection matrix, which projects $\boldsymbol{x}_n$ onto $\hat{\boldsymbol{x}}_n$ is

$$\sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T = [\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_l][\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_l]^T = \mathbf{P}\mathbf{P}^T$$

- The reconstruction error is

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \mathbf{P}\mathbf{P}^T\boldsymbol{x}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \|(\mathbf{I} - \mathbf{P}\mathbf{P}^T)\boldsymbol{x}_n\|^2$$

PCA: Projection Perspective

- In addition to this observation, since

$$\hat{\boldsymbol{x}}_n = \left(\sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T\right) \boldsymbol{x}_n \text{ and } \boldsymbol{x}_n = \left(\sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T\right) \boldsymbol{x}_n + \left(\sum_{i=l+1}^{D} \boldsymbol{q}_i \boldsymbol{q}_i^T\right) \boldsymbol{x}_n,$$

the reconstruction loss can be also written as

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \left\| \left(\sum_{i=l+1}^{D} \boldsymbol{q}_i \boldsymbol{q}_i^T\right) \boldsymbol{x}_n \right\|^2 = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=l+1}^{D} \left(\boldsymbol{q}_i^T \boldsymbol{x}_n\right)^2$$

PCA: Projection Perspective

- Thus, the reconstruction loss is equivalent to

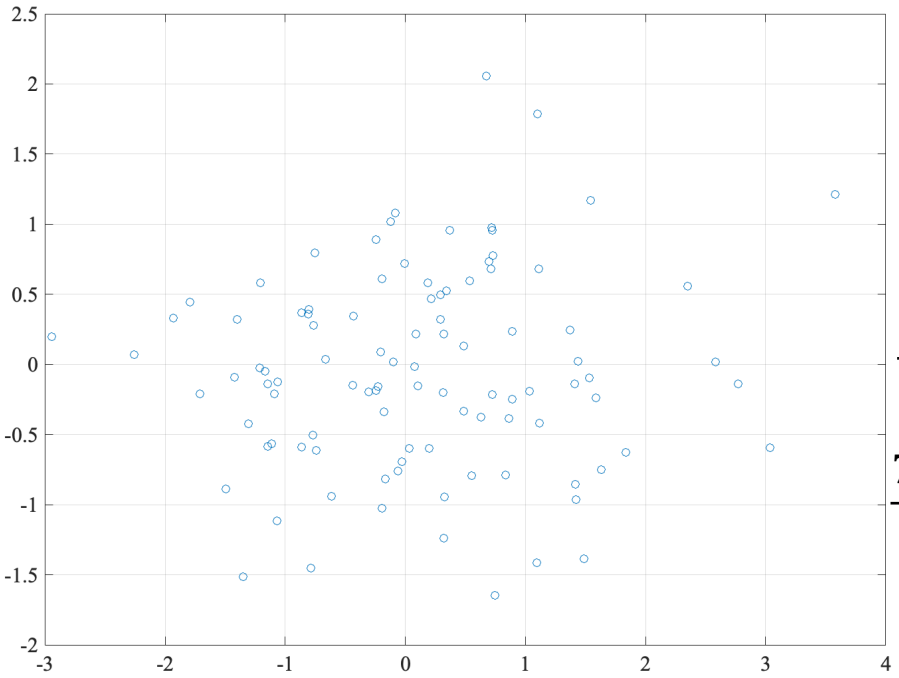
$$J = \sum_{i=l+1}^{D} \boldsymbol{q}_i^T \left(\frac{1}{N} \sum_{n=1}^{N} \sum_{i=l+1}^{D} \boldsymbol{x}_n \boldsymbol{x}_n^T \right) \boldsymbol{q}_i = \sum_{i=l+1}^{D} \boldsymbol{q}_i^T \mathbf{R}_{\mathbf{X}}^{S} \boldsymbol{q}_i = \mathrm{tr}\left(\left(\sum_{i=l+1}^{D} \boldsymbol{q}_i \boldsymbol{q}_i^T \right) \mathbf{R}_{\mathbf{X}}^{S} \right)$$

  - the reconstruction error is the projection of $\mathbf{R}_{\mathbf{X}}^{S}$ onto the orthogonal complement of the principle subspace
  - from $J = \sum_{i=l+1}^{m} \boldsymbol{q}_i^T \mathbf{R}_{\mathbf{X}}^{S} \boldsymbol{q}_i$, minimizing the reconstruction error is equivalent to minimizing the total residual variance of the data
  - thus, maximizing the variance of the projection is the same as the maximum-variance formulation of PCA

Eigenvector Computation

- Two approaches can be adopted to obtain the eigenvalues
  - approach I:
    we compute the eigenvalues of $\mathbf{R}_{\mathbf{X}}^{S} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^T = \frac{1}{N}\mathbf{X}\mathbf{X}^T$ directly
  - approach II
    we can use the singular value decomposition. Since $\mathbf{R}_{\mathbf{X}}^{S}$ is symmetric, the eigenvalues of $\mathbf{R}_{\mathbf{X}}^{S}$ are simply the squared singular values of $\mathbf{X}$ divided by N

Eigenvector Computation



Approach I: the eigenvalues are 1.3564 and 0.4991

Approach II: the singular values are 11.6465 and 7.0645

$$\frac{11.6465^2}{100} = 1.3564$$

$$\frac{7.0645^2}{100} = 0.4991$$

Principal Component Analysis
Topic 04
Singular Value Decomposition

SVD Theorem

If $\mathbf{A}$ is an $m \times n$ matrix and $m \geq n$, then $\mathbf{A}$ has a singular value decomposition. This means that we always can factor $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{UEV}^T$$

where $\mathbf{U}$ is an $m \times m$ orthogonal matrix, $\mathbf{E}$ is an $m \times n$ matrix whose off-diagonal entries are all 0's, and $\mathbf{V}$ is an n$\times n$ orthogonal matrix and whose diagonal elements are nonzero and satisfy

$$E = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$$

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$$

Proof of SVD Theorem

- $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{A}^T\mathbf{A}$ is an $n \times n$ matrix
- $\mathbf{A}^T\mathbf{A}$ is a symmetric matrix and two facts can be observed
  - $\mathbf{A}^T\mathbf{A}$ has n eigenvectors
  - these eigenvectors can form a basis for a n-dimensional space
- Furthermore, the eigenvalues of $\mathbf{A}^T\mathbf{A}$ are nonnegative
  - let $\boldsymbol{v}$ be an eigenvector of $\mathbf{A}^T\mathbf{A}$ and λ is the associated eigenvalue

$$\|\mathbf{A}\boldsymbol{v}\|^2 = (\mathbf{A}\boldsymbol{v})^T(\mathbf{A}\boldsymbol{v}) = \boldsymbol{v}^T\mathbf{A}^T\mathbf{A}\boldsymbol{v} = \boldsymbol{v}^T(\mathbf{A}^T\mathbf{A}\boldsymbol{v}) = \boldsymbol{v}^T(\lambda\boldsymbol{v}) = \lambda\boldsymbol{v}^T\boldsymbol{v} = \lambda\|\boldsymbol{v}\|^2$$

$$\Rightarrow \quad \lambda = \frac{\|\mathbf{A}\boldsymbol{v}\|^2}{\|\boldsymbol{v}\|^2} \geq 0$$

Proof of SVD Theorem

- Let $\mathbf{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n]$, where $\boldsymbol{v}_i$ is the eigenvectors of $\mathbf{A}^T\mathbf{A}$ for $i = 1,2 \cdots, n$
  - with algebraic manipulation, we let $\mathbf{V}$ be an orthogonal matrix
- The eigenvalue of $\boldsymbol{v}_i$ is λ_i. We let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$
- From these eigenvalues, we construct an $n \times n$ diagonal matrix $\tilde{\mathbf{E}}$ whose ith diagonal entry, σ_i, is the square root of λ_i
- Now we only construct $\mathbf{V}$ and $\tilde{\mathbf{E}}$, which is only related to $\mathbf{A}^T\mathbf{A}$ so far. But what are their relationship with $\mathbf{A}$?
- In order to establish that relationship, let us look at the range and the null space of $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}$

Proof of SVD Theorem

- First, we note that the nullspaces of $\mathbf{A}$ and $\mathbf{A}^T\mathbf{A}$ are the same
  - for $\boldsymbol{x}$ in the nullspace of $\mathbf{A}$, $\mathbf{A}\boldsymbol{x} = \mathbf{0}$
  - this implies $\mathbf{A}^T\mathbf{A}\boldsymbol{x} = \mathbf{0}$
  - the nullspaces of $\mathbf{A}$ and $\mathbf{A}^T\mathbf{A}$ are the same
  - nullity of $\mathbf{A}$ and that of $\mathbf{A}^T\mathbf{A}$ are equal
    - nullity is the dimension of the nullspace
- The sum of rank and nullity of either $\mathbf{A}$ or $\mathbf{A}^T\mathbf{A}$ is n
  - the ranks of $\mathbf{A}$ and $\mathbf{A}^T\mathbf{A}$ are equal due the the same nullity
    - rank is the dimension of the range

Proof of SVD Theorem

- Let the rank of either $\mathbf{A}$ or $\mathbf{A}^T\mathbf{A}$ is r
  - the rank of a matrix is associated to the number of non-negative eigenvalues of a matrix
  - $\sigma_{r+1} = \sigma_{r+2} = \cdots = \sigma_n = 0$
- We split $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2]$
  - $\mathbf{V}_1 = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_r]$
    - those eigenvectors correspond to non-negative eigenvalues
    - an orthonormal basis the range of $\mathbf{A}^T\mathbf{A}$
  - $\mathbf{V}_2 = [\boldsymbol{v}_r, \boldsymbol{v}_{r+1}, \cdots, \boldsymbol{v}_n]$
    - those eigenvectors whose eigenvalues are zero
    - an orthonormal basis of the nullspace of $\mathbf{A}^T\mathbf{A}$, and hence, $\mathbf{A}$
    - $\mathbf{A}^T\mathbf{A}\mathbf{V}_2 = \mathbf{0}$ and $\mathbf{A}\mathbf{V}_2 = \mathbf{0}$

Proof of SVD Theorem

- We now look at the relationship between $\mathbf{A}$ and $\mathbf{V}$

$$\mathbf{A} = \mathbf{AI} = \mathbf{A}(\mathbf{V}\mathbf{V}^T) = \mathbf{A}([\mathbf{V_1}, \mathbf{V_2}][\mathbf{V_1}, \mathbf{V_2}]^T) = \mathbf{A}\left(\mathbf{V_1}\mathbf{V_1}^T + \mathbf{V_2}\mathbf{V_2}^T\right)$$

  - the second equality comes from the fact that $\mathbf{V}$ is an orthogonal matrix

$$\mathbf{A}\left(\mathbf{V_1}\mathbf{V_1}^T + \mathbf{V_2}\mathbf{V_2}^T\right) = \mathbf{A}\mathbf{V_1}\mathbf{V_1}^T + \mathbf{A}\mathbf{V_2}\mathbf{V_2}^T = \mathbf{A}\mathbf{V_1}\mathbf{V_1}^T$$

  - the second equality comes from the fact that $\mathbf{AV_2} = \mathbf{0}$

- We have

$$\mathbf{A} = \mathbf{A}\mathbf{V_1}\mathbf{V_1}^T$$

Proof of SVD Theorem

- Now let define $\mathbf{E}$ in SVD

$$\mathbf{E} = \begin{bmatrix} \tilde{\mathbf{E}} \\ \mathbf{0}_{(m-n)\times n} \end{bmatrix}$$

where $\mathbf{0}_{(m-n)\times n}$ is an $(m-n)\times n$ all zero matrix

Proof of SVD Theorem

- We now has shown how to obtain $\mathbf{V}$ and $\mathbf{E}$ in SVD
- How to acquire $\mathbf{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_n]$?
- Let us return to the definition of SVD

$$\mathbf{A} = \mathbf{UEV}^T$$

$$\mathbf{AV} = \mathbf{UE}$$

- With the fact $\sigma_{r+1} = \sigma_{r+2} = \cdots = \sigma_n = 0$, we have

$$\mathbf{A}\boldsymbol{v}_i = \sigma_i \boldsymbol{u}_i, \qquad i = 1,2,\cdots,r$$

Proof of SVD Theorem

- Therefore, the first r columns of $\mathbf{U}$ can be found by

$$\boldsymbol{u}_i = \frac{1}{\sigma_i}\mathbf{A}\boldsymbol{v}_i, \qquad i = 1,2,\cdots,r$$

  - $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_r$ form an orthonormal set so are $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r$
- How about the remaining $\boldsymbol{u}_{r+1}, \boldsymbol{u}_{r+2}, \cdots, \boldsymbol{u}_m$?
  - $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r$ are in the column space of $\mathbf{A}$, which is the range of $\mathbf{A}$
  - since the rank of $\mathbf{A}$ is r, the dimensonality of the column space of $\mathbf{A}$ is r
    - $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r$ form an orthonormal basis of the column space of $\mathbf{A}$
  - $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r\}$ and $\{\boldsymbol{u}_{r+1}, \boldsymbol{u}_{r+2}, \cdots, \boldsymbol{u}_m\}$ are orthogonal
  - the span of $\{\boldsymbol{u}_{r+1}, \boldsymbol{u}_{r+2}, \cdots, \boldsymbol{u}_m\}$ is the orthogonal complement of the range of $\mathbf{A}$
    - the orthogonal complement of the range of $\mathbf{A}$ is the nullspace of $\mathbf{A}^T$
  - $\boldsymbol{u}_{r+1}, \boldsymbol{u}_{r+2}, \cdots, \boldsymbol{u}_m$ form an orthonormal basis of the nullspace of $\mathbf{A}^T$

Procedures to find SVD of a matrix $\mathbf{A}$

1) Calculate $\mathbf{A}^T\mathbf{A}$
2) Find the eigenvalues and eigenvectors of $\mathbf{A}^T\mathbf{A}$ and determine the rank, r, as well
3) Find the singular values of $\mathbf{A}$ by take the square root of the eigenvalues of $\mathbf{A}^T\mathbf{A}$
4) Normalize the eigenvectors of $\mathbf{A}^T\mathbf{A}$ and form the matrix $\mathbf{V}$
5) Find the first r columns of $\mathbf{U}$ via the following equation

$$\mathbf{u_i} = \frac{1}{\sigma_i}\mathbf{A}\mathbf{v_i}.$$

6) Find the rest columns of $\mathbf{U}$ by finding the orthonormal basis of $\mathcal{N}(\mathbf{A}^T)$. There are two approaches to achieved this.
   a) By solving $\mathbf{A}^T\mathbf{x} = \mathbf{0}$.
      Note that the solutions we find is the basis of $\mathcal{N}(\mathbf{A}^T)$. We need to use Gram-Schmidt process to convert them into an orthonormal basis.
   b) By inspection. (This is usually hard to do so!!)

Example

- Find the SVD of $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \rightarrow \boldsymbol{A}^T\boldsymbol{A} = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

- The eigenvalues of $\mathbf{A}$ is 4 and 0
  - the eigenvector corresponds to 4
  - the eigenvector corresponds to 0

$$v_1 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$v_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\rightarrow \mathbf{V} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

Example

- Next let us find $\mathbf{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3]$
- It is easy to find $\boldsymbol{u}_1$

$$\boldsymbol{u}_1 = \frac{1}{2}\mathbf{A}\boldsymbol{v}_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix}$$

- To derive $\boldsymbol{u}_2, \boldsymbol{u}_3$, we just solve $\mathbf{A}^T\boldsymbol{x} = 0$. This gives us

$$\boldsymbol{u}_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \\ 0 \end{bmatrix} \qquad \boldsymbol{u}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Example

- To sum up,

$$\mathbf{V} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \qquad \mathbf{E} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Principal Component Analysis

Topic 05

PCA in Practice

PCA in High Dimensions

- In PCA, we need to compute the data covariance matrix
  - in the D-dimensions, the data covariance matrix is a $D \times D$ matrix
    - for example, if the data are images with 256*256 pixels, the data covariance matrix is a 65536×65536 matrix
  - computing its eigenvalues and the corresponding eigenvectors is computationally expensive
  - PCA is not feasible in very high dimensions
- How to deal with PCA in high dimensions?

PCA in High Dimensions

- We consider the case when the number of data points, N, is much smaller than the dimension, D
- The data covariance matrix, $\mathbf{R}_{\mathbf{X}}^{S}$, is a $D \times D$ matrix and

$$\mathbf{R}_{\mathbf{X}}^{S}\boldsymbol{q}_i = \lambda_i \boldsymbol{q}_i$$

- Since $\mathbf{R}_{\mathbf{X}}^{S} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^T = \frac{1}{N}\mathbf{X}\mathbf{X}^T$, we can rewrite $\mathbf{R}_{\mathbf{X}}^{S}\boldsymbol{q}_i = \lambda_i \boldsymbol{q}_i$ as

$$\mathbf{R}_{\mathbf{X}}^{S}\boldsymbol{q}_i = \frac{1}{N}\mathbf{X}\mathbf{X}^T\boldsymbol{q}_i = \lambda_i \boldsymbol{q}_i \quad \Rightarrow \quad \frac{1}{N}\mathbf{X}^T\mathbf{X}\mathbf{X}^T\boldsymbol{q}_i = \lambda_i \mathbf{X}^T\boldsymbol{q}_i$$

PCA in High Dimensions

- From $\frac{1}{N}\mathbf{X}^T\mathbf{X}\mathbf{X}^T\boldsymbol{q}_i = \lambda_i\mathbf{X}^T\boldsymbol{q}_i$ and let $\boldsymbol{q}_i' = \mathbf{X}^T\boldsymbol{q}_i$, we have a new eigenvector/eigenvalue equation

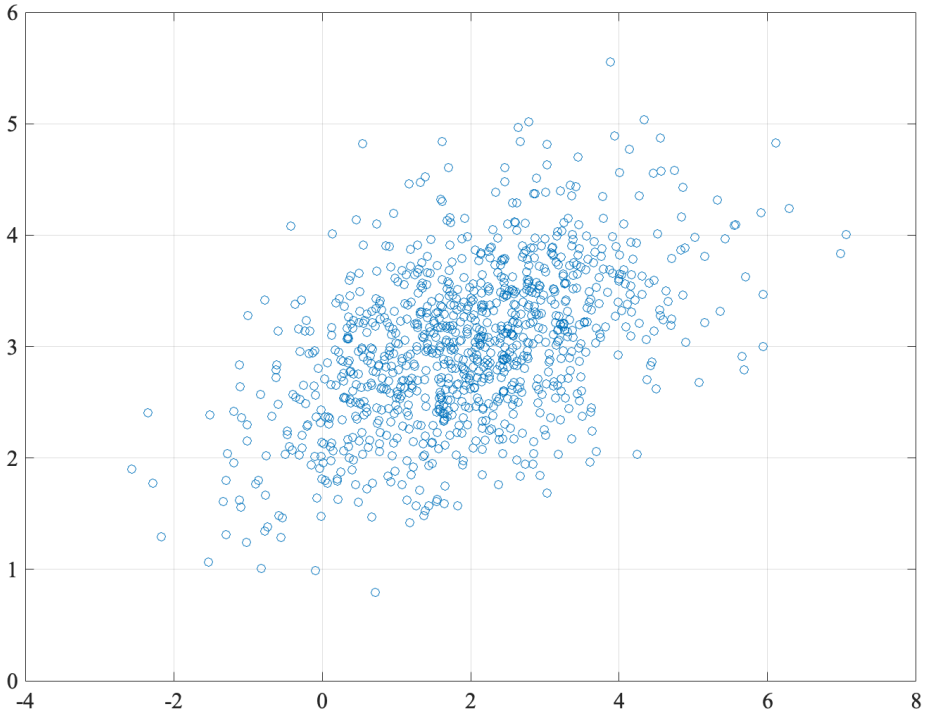
$$\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right)\boldsymbol{q}_i' = \lambda_i\boldsymbol{q}_i'$$

  - we deal with a $N\times N$ matrix, $\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right)$, which has the same eigenvalues of $\mathbf{R}_{\mathbf{X}}^S$
  - to recover the eigenvectors of $\mathbf{R}_{\mathbf{X}}^S$, we left-multiply $\mathbf{X}$ this equation and

$$\mathbf{X}\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right)\boldsymbol{q}_i' = \frac{1}{N}\mathbf{X}\mathbf{X}^T\mathbf{X}\boldsymbol{q}_i' = \mathbf{R}_{\mathbf{X}}^S(\mathbf{X}\boldsymbol{q}_i') = \lambda_i(\mathbf{X}\boldsymbol{q}_i')$$

  - this mean $\mathbf{X}\boldsymbol{q}_i'$ is the eigenvectors of $\mathbf{R}_{\mathbf{X}}^S$

Key Steps of PCA in Practice



Key Steps of PCA in Practice

- Suppose we have a dataset consisting of N data samples

  $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N], \boldsymbol{x}_i \in \mathbb{R}^D$

- Step 1: mean subtraction
  - the mean of the dataset is

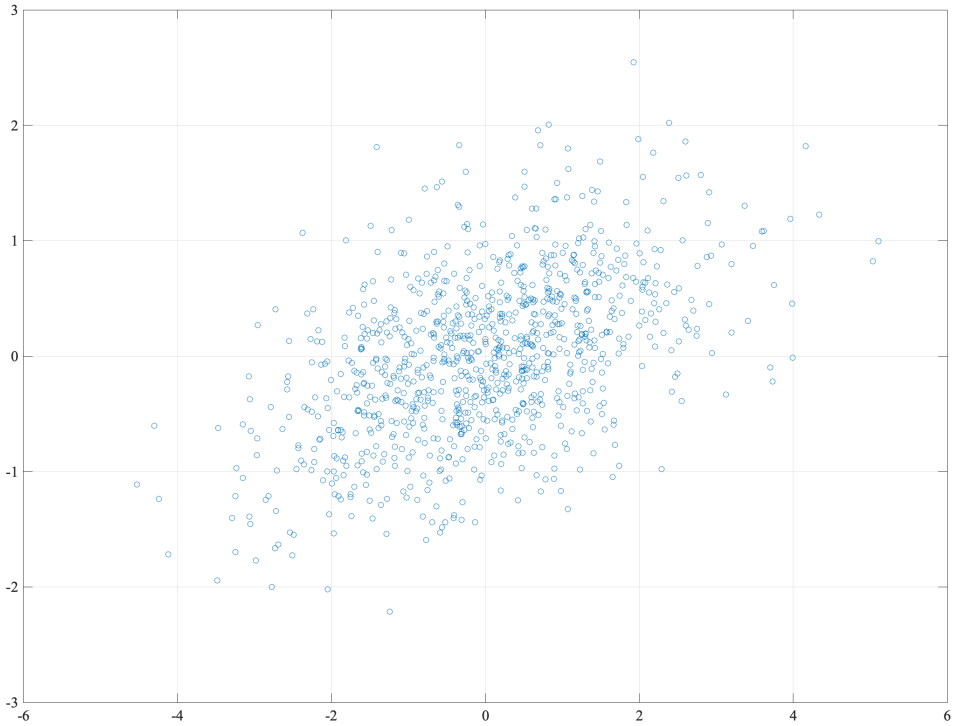
    $$\boldsymbol{m}_X^S = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i$$

  - we subtract $\boldsymbol{m}_X^S$ from every data sample

    $$\boldsymbol{x}_i^o = \boldsymbol{x}_i - \boldsymbol{m}_X^S$$

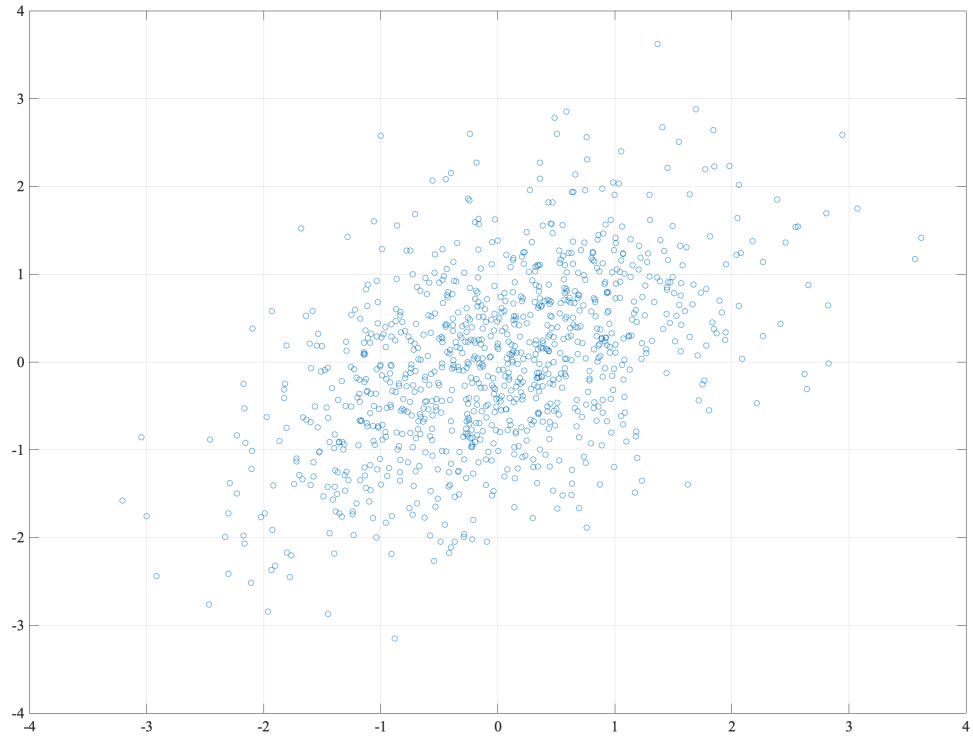
    and

    $$\mathbf{X}_o = [\boldsymbol{x}_1^o, \boldsymbol{x}_2^o, \cdots, \boldsymbol{x}_N^o], \boldsymbol{x}_i^o \in \mathbb{R}^D$$



Key Steps of PCA in Practice

- Step 2: Standardization
  - let $\mathbf{R}_{\mathbf{X}_o}^s = \frac{1}{N}\mathbf{X}_o\mathbf{X}_o^T$
  - we divided the data points by the standard deviation σ_d of the dataset for every dimension $d = 1,2,\cdots,D$
  - σ_d is simply the square root of the dth diagonal element of $\mathbf{R}_{\mathbf{X}_o}^s$



Key Steps of PCA in Practice

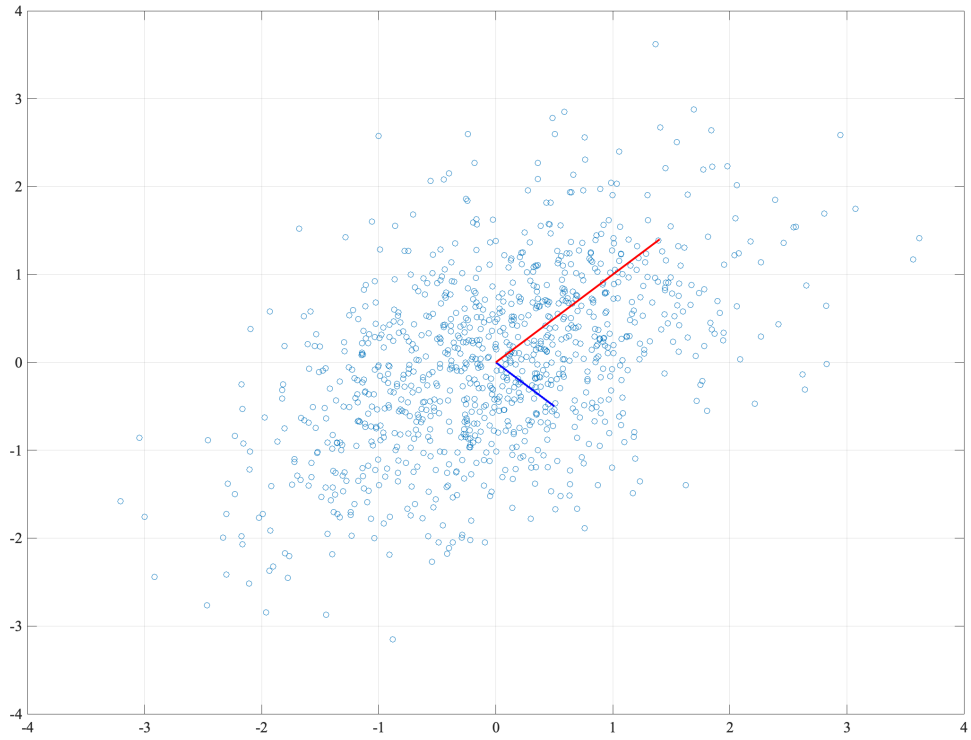
- Step 3: Eigendecomposition of the covariance matrix
  - compute the data covariance

$$\mathbf{R}_{\mathbf{X}_o}^{S} = \frac{1}{N}\mathbf{X}_o\mathbf{X}_o^T$$

  - compute the eigenvalues and the corresponding eigenvectors of $\mathbf{R}_{\mathbf{X}_o}^{S}$

$$\lambda_i, \boldsymbol{q}_i \; i = 1,2,\cdots,D$$

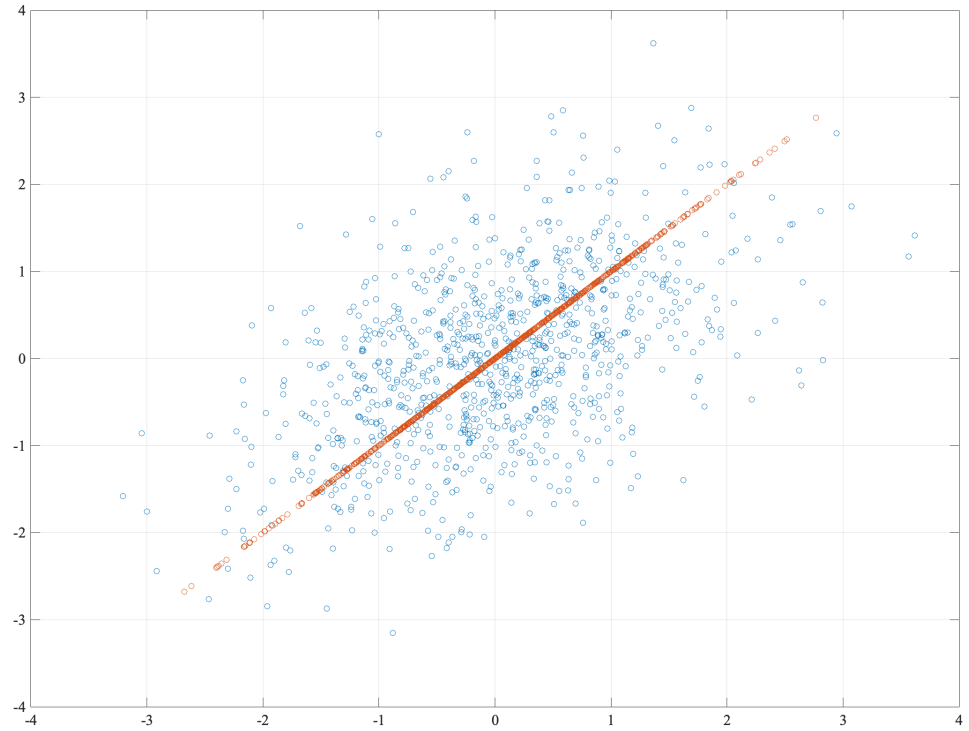
$$\lambda_1 > \lambda_2 > \cdots > \lambda_D$$



Key Steps of PCA in Practice

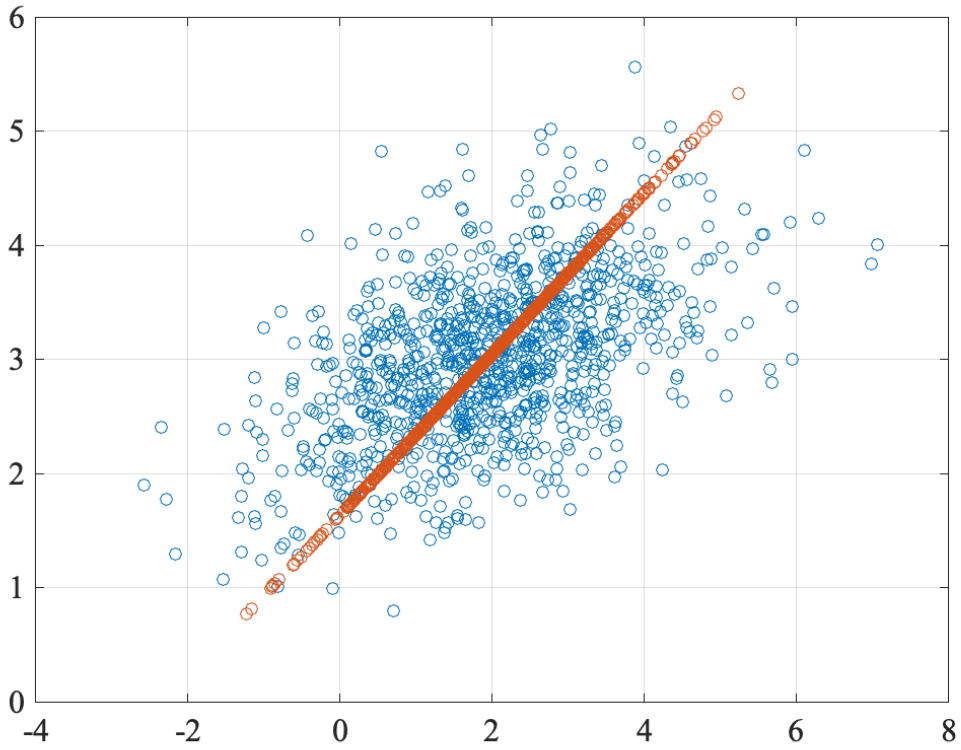
- Step 4: Projection
  - project data point onto the principal subspace

$$\hat{\boldsymbol{x}}_n = \left(\sum_{i=1}^{l} \boldsymbol{q}_i \boldsymbol{q}_i^T\right) \boldsymbol{x}_n^o = \mathbf{P}\mathbf{P}^{\mathbf{T}} \boldsymbol{x}_n^o$$



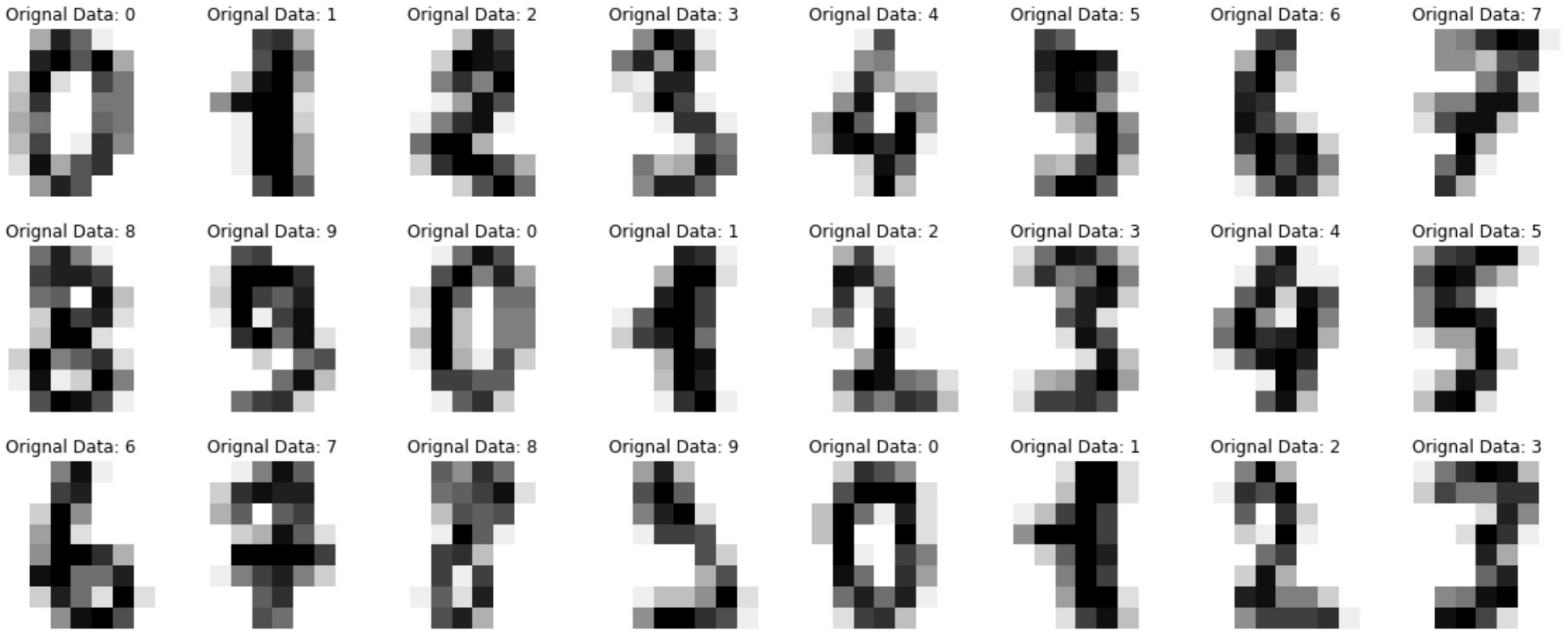
Key Steps of PCA in Practice

- Step 5: Undo the standardization
  - multiply each dimension of $\hat{\boldsymbol{x}}_n$ by σ_d
  - add $\boldsymbol{m}_X^S$ to the scaled $\hat{\boldsymbol{x}}_n$



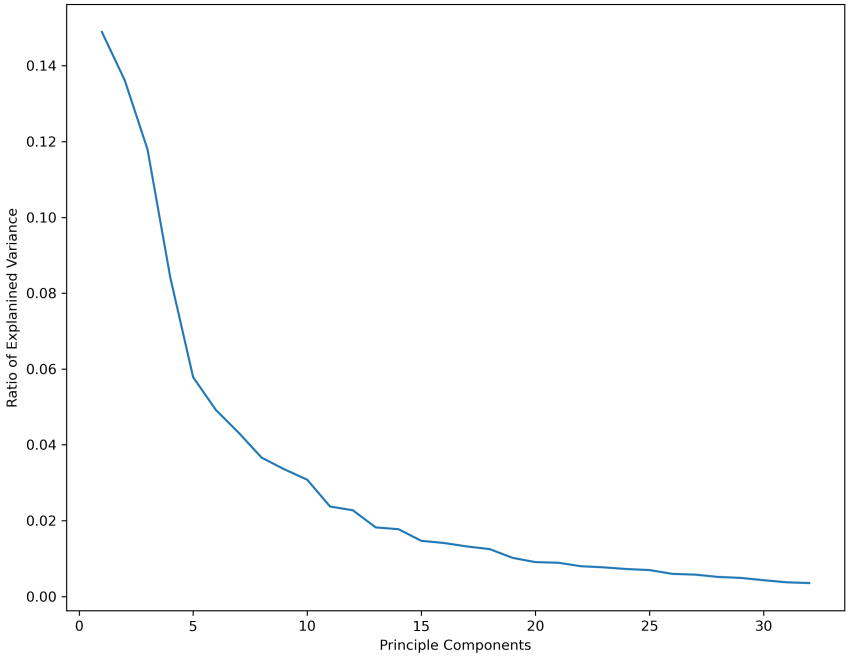
Key Steps of PCA in Practice

- Example: MNIST Dataset (Each image is 8-by-8)



Key Steps of PCA in Practice

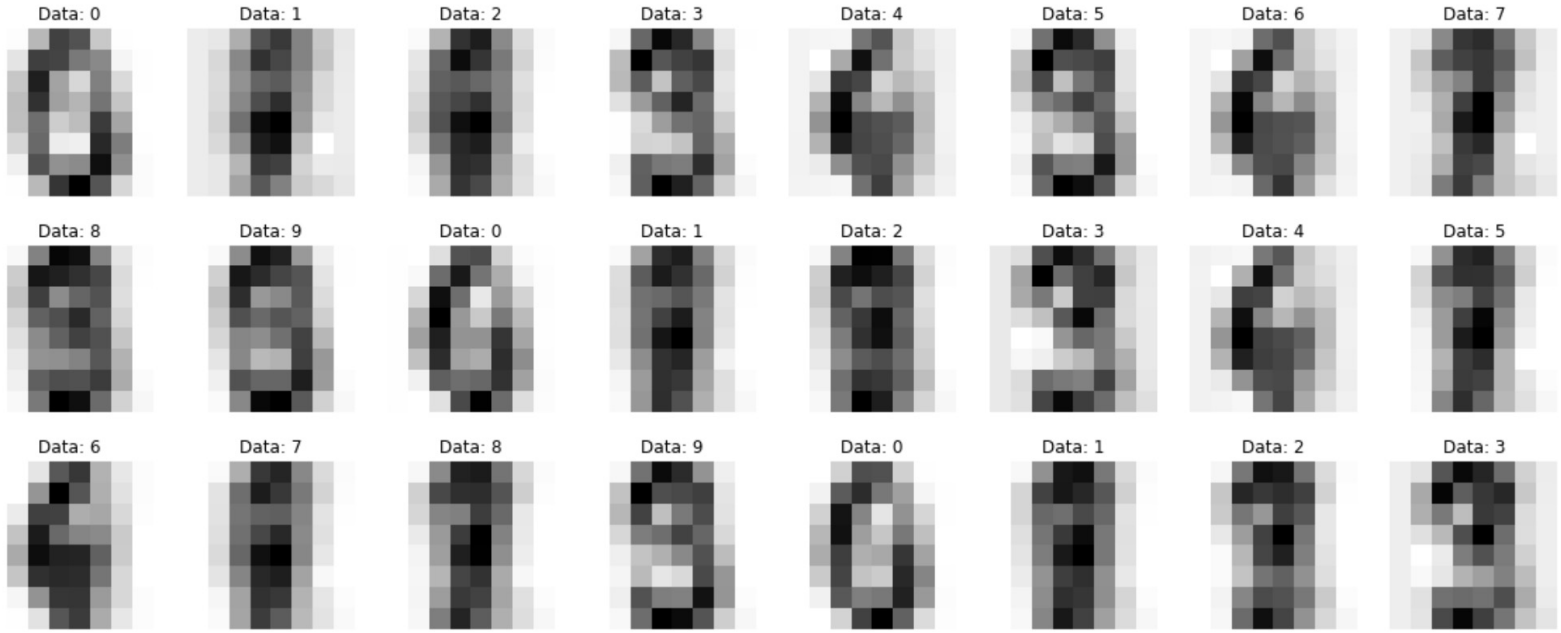
- Example: MNIST Dataset (Each image is 8-by-8)



Key Steps of PCA in Practice

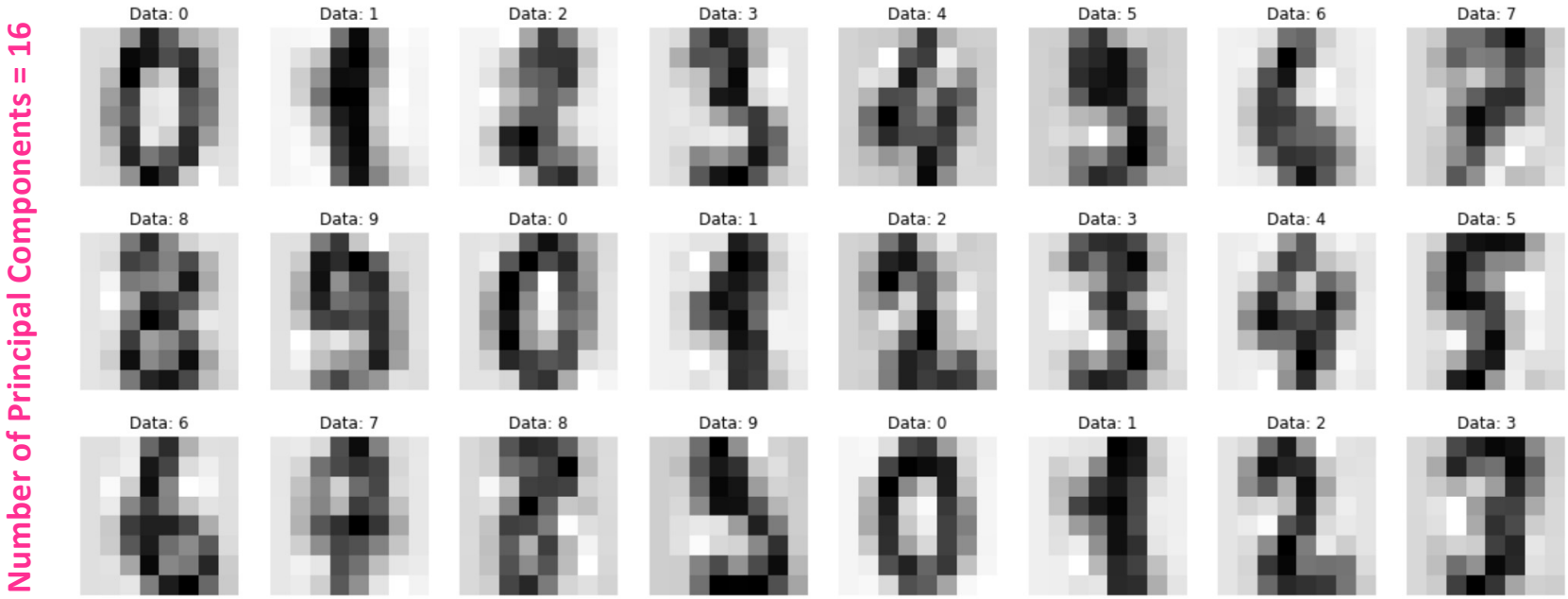
- Example: MNIST Dataset

Number of Principal Components = 2



Key Steps of PCA in Practice

- Example: MNIST Dataset



Key Steps of PCA in Practice

- Example: MNIST Dataset

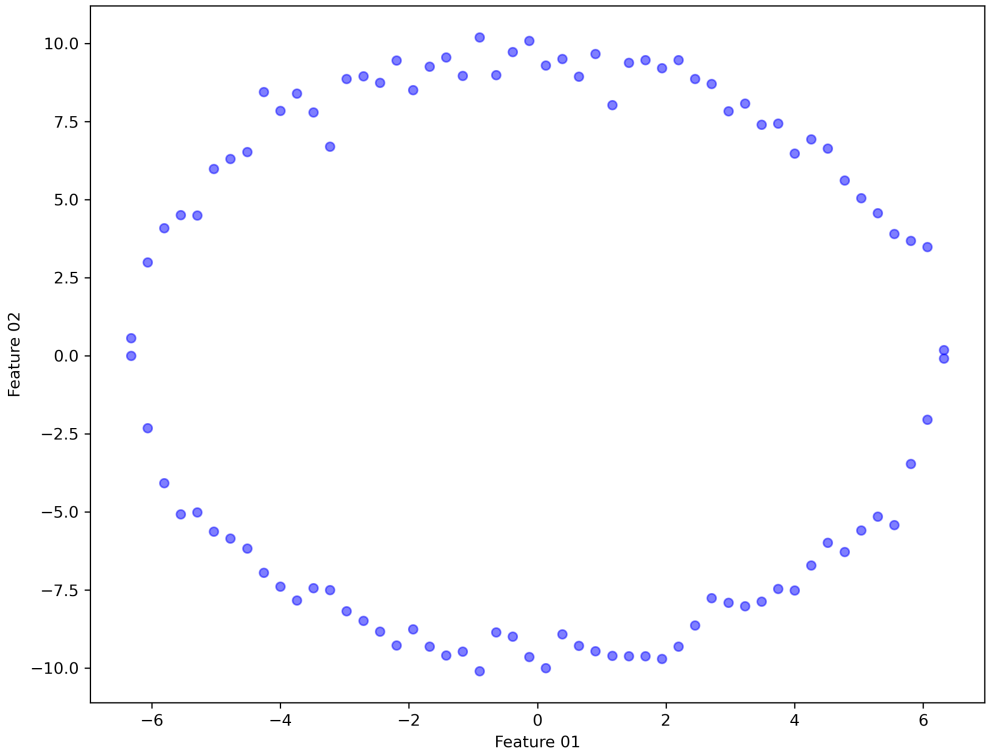
Number of Principal Components = 32

Principal Component Analysis
Topic 06
Implementation Using Scikit-Learn

Step 01: Dataset Preparation

```
import numpy as np
n_samples = 50
xmax = np.sqrt(10*4)
X = np.linspace(-xmax+0.0001,xmax-0.0001,n_samples).reshape(-1,1)
y = np.sqrt(9*(10-X*X/4)).reshape(-1,1)
X = np.append(X,X)
y = np.append(y,-y) + 0.5*np.random.standard_normal(2*n_samples)
Data = np.c_[X,y]
```

Step 01: Dataset Preparation



Step 02: Build/Train ML Model

- 'PCA' is responsible for the principal component analysis
  - SVD is adopted for dimensionality reduction

```
from sklearn.decomposition import PCA
PCA_Model = PCA(n_components=2)
```

  - 'n_components' is the parameter for the desired dimension

Step 02: Build/Train ML Model

- To obtain the information of principal components, we need to let the model to fit the data

```
PCA_Model.fit(Data)
```

- The information of the principal components

```
PCA_Model.components_
```

  - the output looks like

```
array([[-0.01029233, -0.99994703], [-0.99994703, 0.01029233]])
```

Step 02: Build/Train ML Model

- The information of explained variance

```
PCA_Model.explained_variance_
```

  - the output is

```
array([60.94725254, 14.01231312])
```

- The information of singular values

```
PCA_Model.singular_values_
```

  - the output is

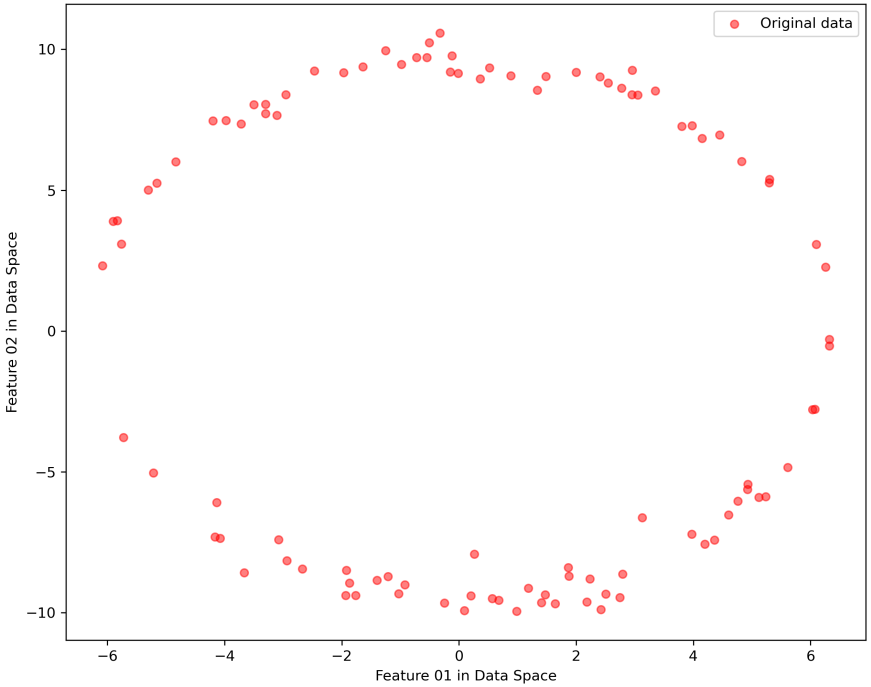
```
array([77.67739698, 37.24538896])
```

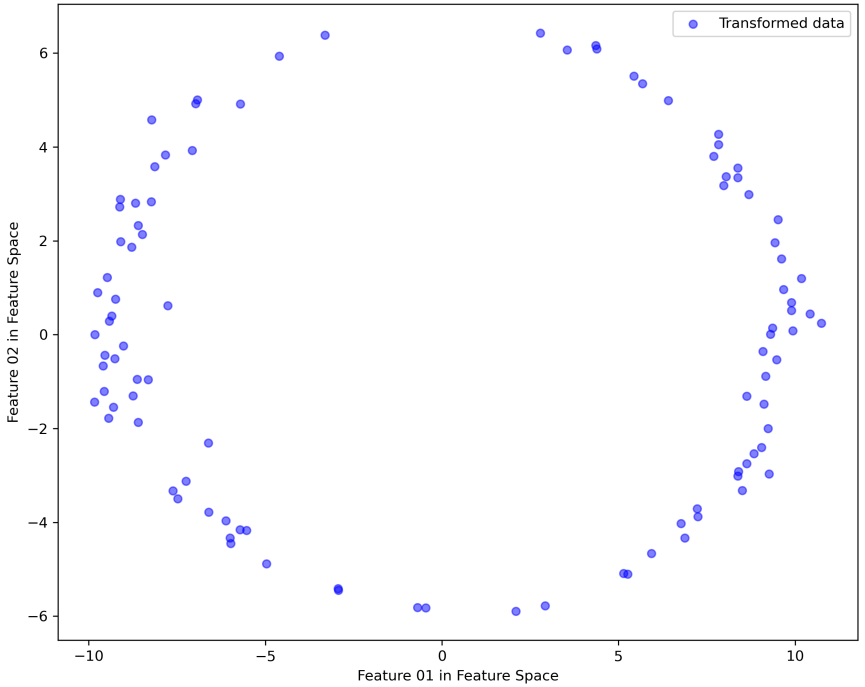
Discussion

- We can transform the data to the feature space spanned by the chosen principal component

```
Data_PCA = PCA_Model.fit_transform(Data)
```

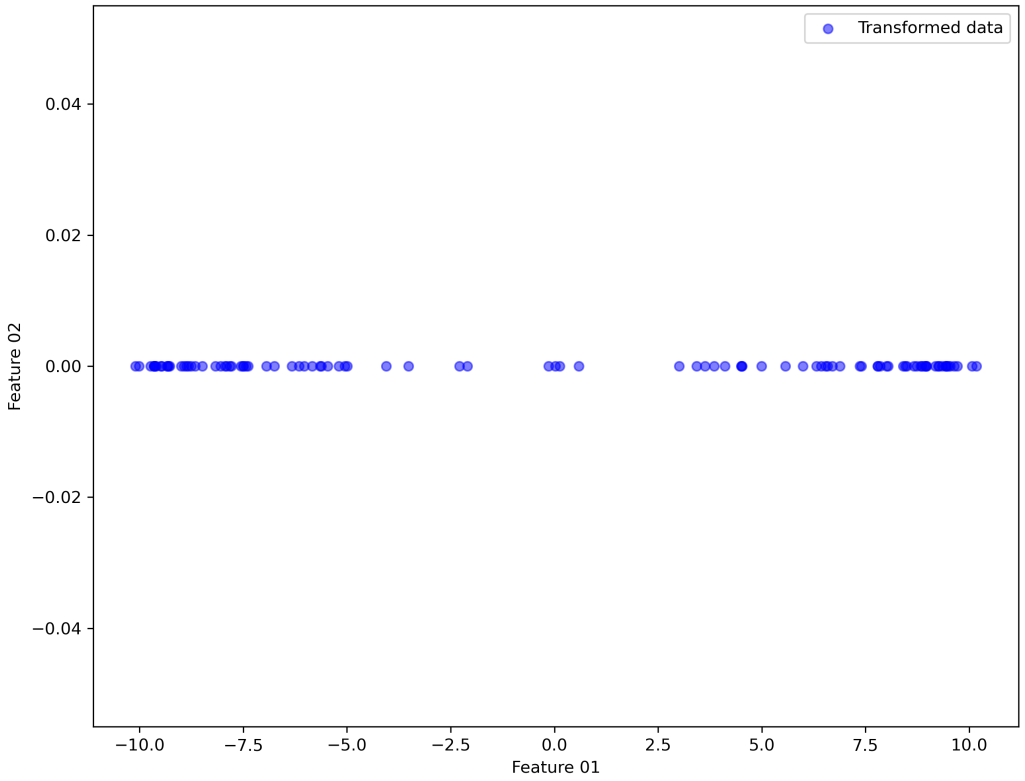
Discussion


Original data
Feature 02 in Data Space
Feature 01 in Data Space
10
5
0
−5
−10
−6
−4
−2
0
2
4
6

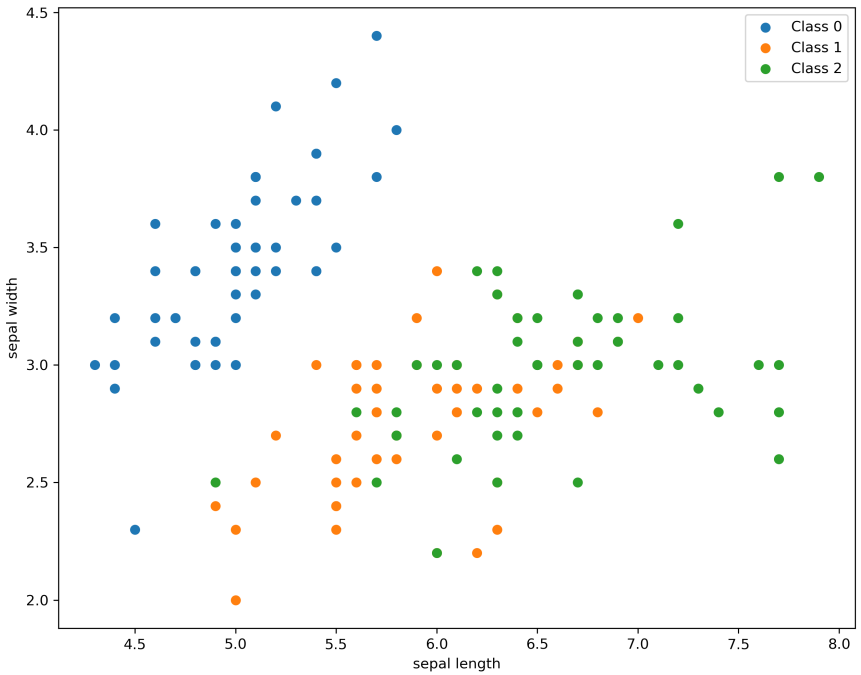

Transformed data
Feature 02 in Feature Space
Feature 01 in Feature Space
6
4
2
0
−2
−4
−6
−10
−5
0
5
10

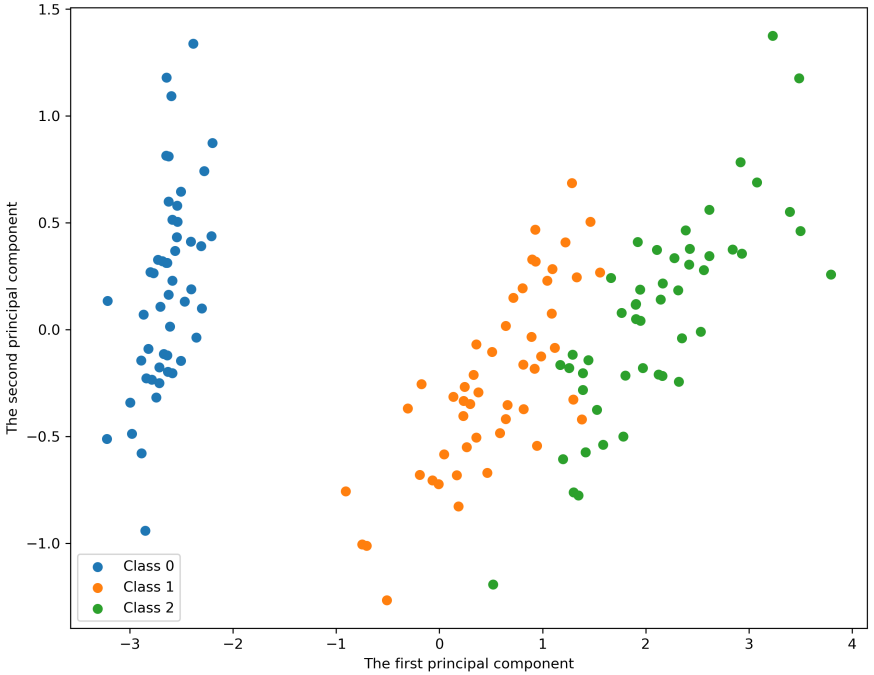
Discussion

```
from sklearn.decomposition import PCA
PCA_Model = PCA(n_components=1)
Data_PCA = PCA_Model.fit_transform(Data)
```



Discussion





Iris Dataset

Q&A