Week 3:
Dimensionality Analysis

CMPS 620 : Machine Learning

Curse of dimensionality

- Many Machine Learning problems involve thousands or even millions of features for each training instance.
- Not only do all these features make training extremely slow, but they can also make it much harder to find a good solution.
  - This problem is often referred to as the **curse of dimensionality**.
- In real-world problems, it is often possible to reduce the number of features considerably, turning an intractable problem into a tractable one.

Curse of dimensionality–cont.

- We will discuss the curse of dimensionality and get a sense of what goes on in high-dimensional space.
- We are so used to living in three dimensions that our intuition fails us when we try to imagine a high-dimensional space.

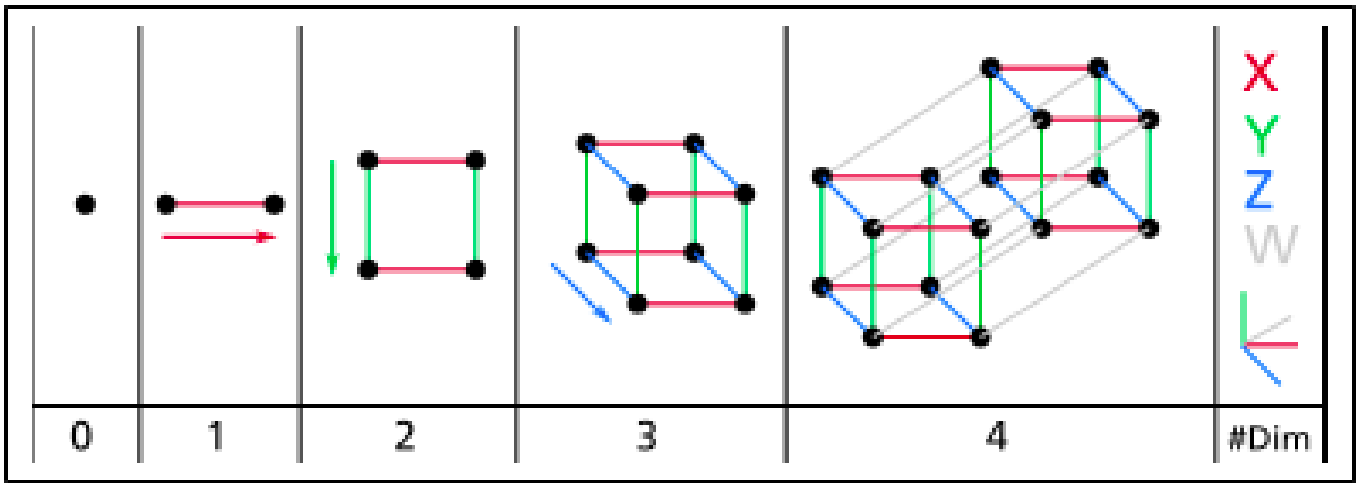

Figure 8-1. Point, segment, square, cube, and tesseract (0D to 4D hypercubes)[2]

- A basic 4D hypercube is incredibly hard to picture in our minds, let alone a 200-dimensional ellipsoid bent in a 1,000-dimensional space.

Example

- Consider the following MNIST images: the pixels on the image borders are almost always white, so you could completely drop these pixels from the training set without losing much information.

Figure 3-1. Digits from the MNIST dataset

- Additionally, if two neighboring pixels are often highly correlated: if you merge them into a single pixel (e.g., by taking the mean of the two pixel intensities), you will not lose much information.

Example– cont.

- Apart from speeding up training, dimensionality reduction is also extremely useful for data visualization (or DataViz).
- Reducing the number of dimensions down to two (or three) makes it possible to plot a condensed view of a high-dimensional training set on a graph and often gain some important insights by visually detecting patterns, such as **clusters**.

Dimensionality Reduction

- Three of the most popular dimensionality reduction techniques are:
  1. Principal Component Analysis (PCA)
  2. Kernel PCA
  3. Locally Linear Embedding (LLE)
- We will however focus on the **PCA**.

Principal Component Analysis (PCA)

- **Principal component analysis (PCA)** refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data.
- PCA is an unsupervised approach, since it involves only a set of features $X_1, X_2, \ldots, X_p$, and no associated response Y.
- Let $\mathbf{X}$ be a data matrix with n rows and p columns. The p elements of each row are measurements on a subject such as height, weight and age.

$$\mathbf{X_{n\times p}} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,r} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,r} & \cdots & x_{2,p} \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ x_{j,1} & x_{j,2} & \cdots & x_{j,r} & \cdots & x_{j,p} \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,r} & \cdots & x_{n,p} \end{bmatrix}$$

Why PCA?

- PCA serves as a great tool for data visualization (visualization of the observations or visualization of the variables).
- Helps identify patterns or structure among objects which could not be visualized otherwise.

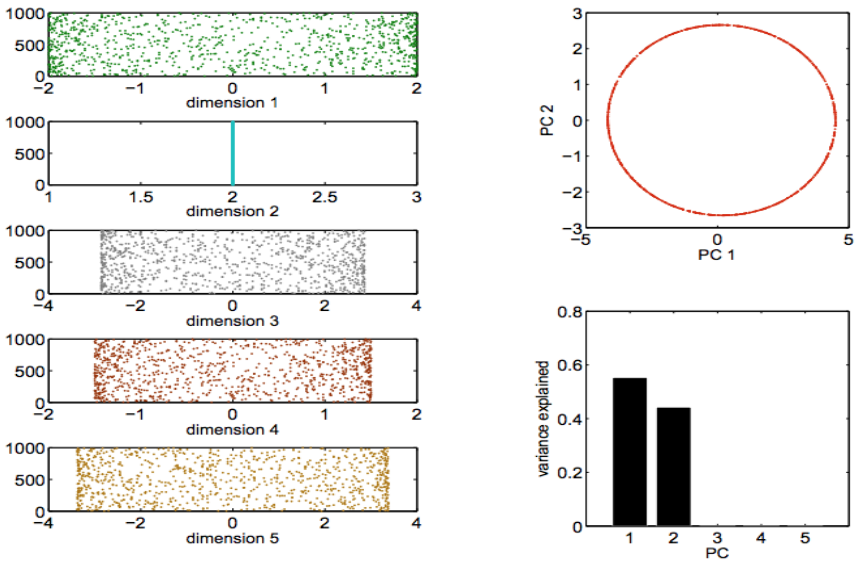

Figure 2: Visualizing low-dimensional data.

Figure: Visualizing low-dimensional data

What Are Principal Components?

- Suppose that we wish to visualize n observations with measurements on a set of p features, $X_1, X_2, \ldots, X_p$ as part of an exploratory data analysis.
- We could do this by examining two-dimensional scatterplots of the data, each of which contains the n observations' measurements on two of the features.
  - There will be $\binom{p}{2} = p(p-1)/2$ such scatterplots; for example, with $p = 10$ there are 45 plots!
  - What happens when p is large??
  - Most likely none of them will be informative since they each contain just a small fraction of the total information present in the data set.

What Are Principal Components?–cont.

- A better method is required to visualize the n observations when p is large.
- We would like to find a low-dimensional representation of the data that captures as much of the information as possible.
  - If we can obtain a two-dimensional representation of the data that captures most of the information, then we can plot the observations in this low-dimensional space.
- PCA provides a tool to do just this.

What Are Principal Components?–cont.

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

First and Second Principal Components

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

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p$$

  that has the largest variance.
  - By normalized, we mean that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$
  - The elements $\phi_{11}, \ldots, \phi_{p1}$ as the **loadings** of the first principal component.
- There is a nice geometric interpretation for the first principal component:
  - The loading vector ϕ_1 with elements $\phi_{11}, \ldots, \phi_{p1}$ defines a direction in feature space along which the data vary the most.
- After the first principal component Z_1 of the features has been determined, we can find the second principal component Z_2:
  - The second principal component is the linear combination of $X_1, X_2, \ldots, X_p$ that has maximal variance out of all linear combinations that are uncorrelated with Z_1.

Subsequent Principal Components

- Each subsequent principal component...
  - is orthogonal to the previous ones, and
  - points in the directions of the largest variance of the residual subspace
- Once we have computed the principal components, we can plot them against each other in order to produce low-dimensional views of the data.
- For instance, we can plot the score vector Z_1 against Z_2, Z_1 against Z_3, Z_2 against Z_3, and so forth.

The Idea

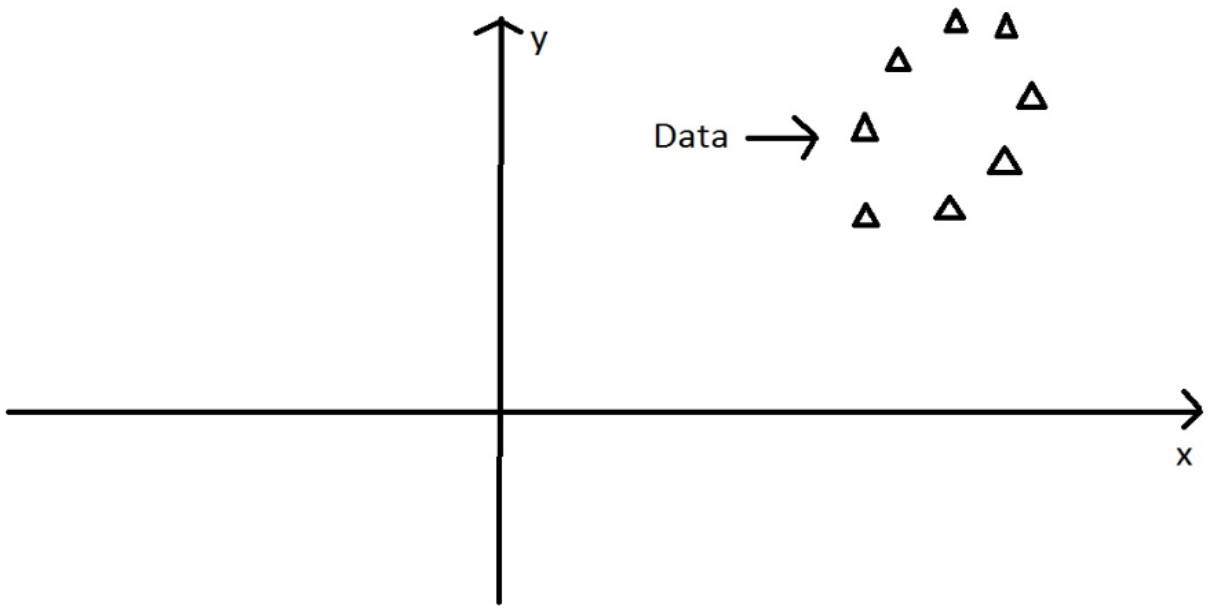

Figure: 2D Data Set

The Idea- Cont.

- If we project the data onto this line, we lose as little information as possible (i.e. we keep as much variance as possible).

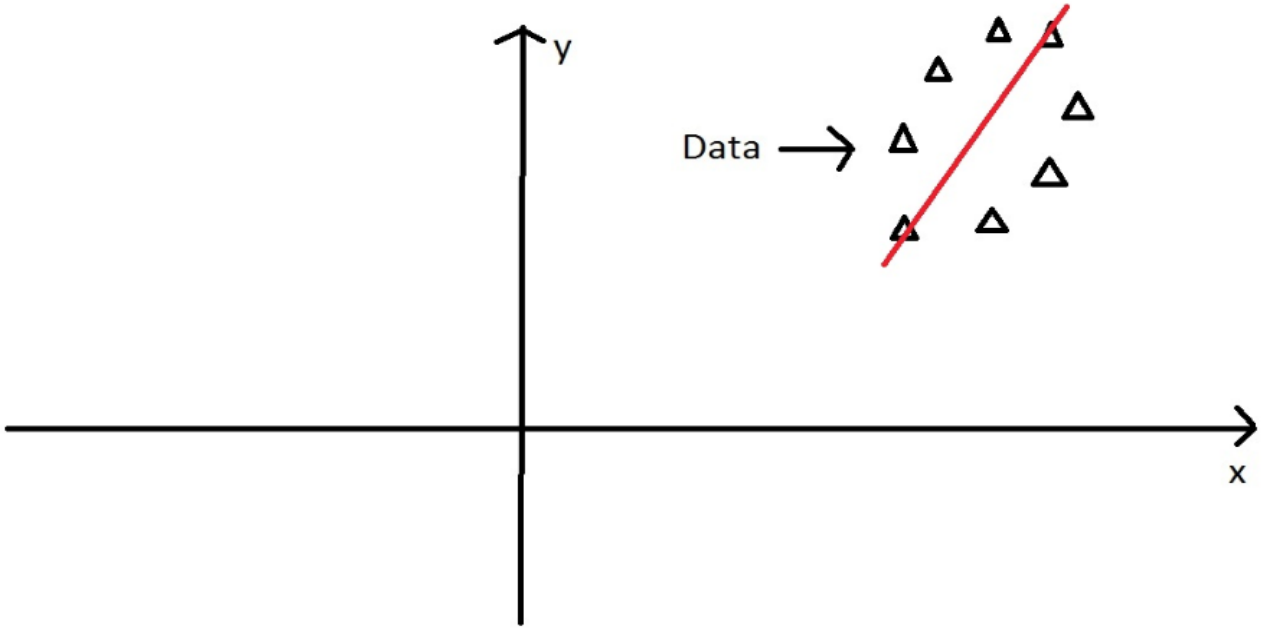

Figure: First PCA axis

The Idea- Cont.

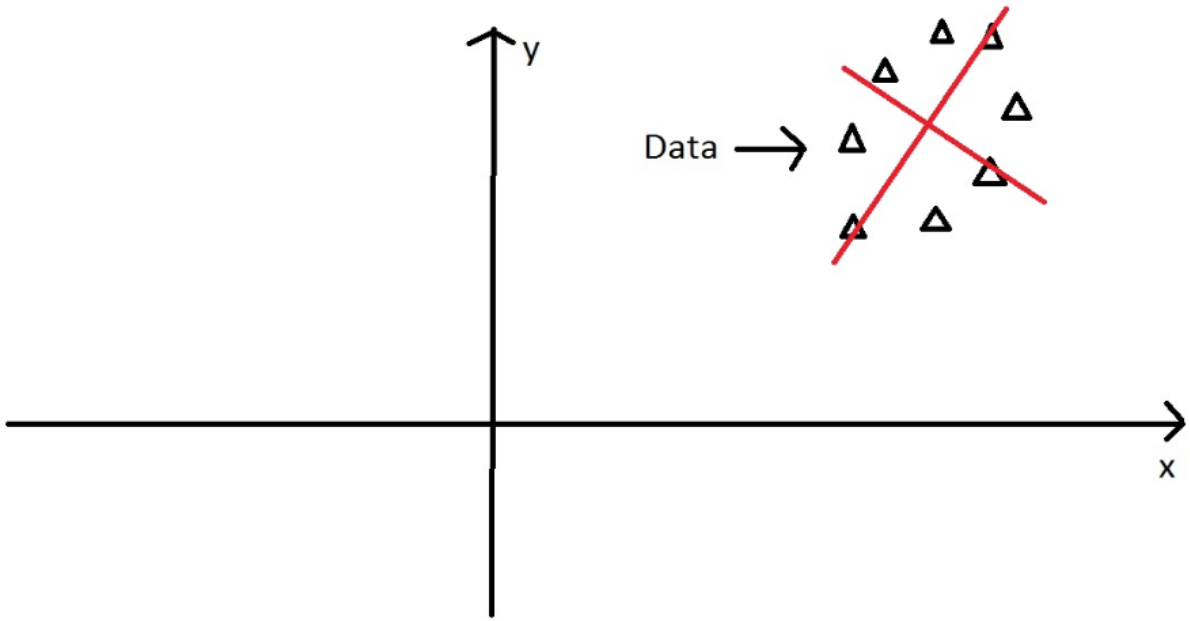

Figure: Second PCA axis

The Idea- Cont.

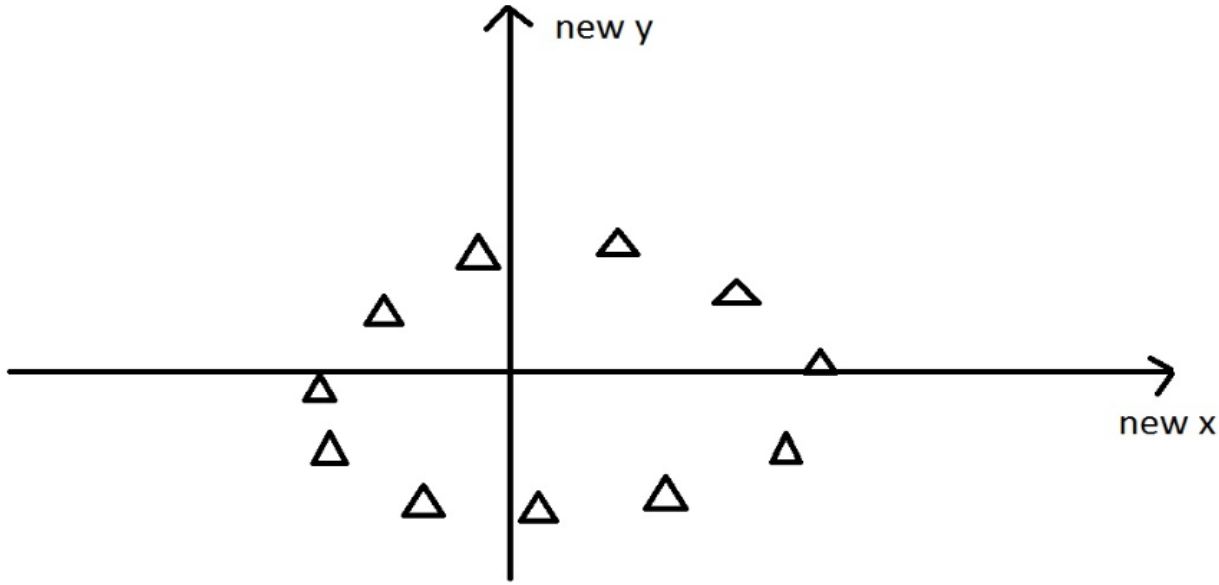

Figure: New axis

Remarks

- PCA is NOT scale invariant!
  - Changing the units of each column will change the results of PCA.
  - Parameters with high variance will influence the principal components more.
  - A good best practice is to center and scale your variables (prcomp in R does not scale automatically)
- Because PCA is based off on orthogonal vectors, the signs for each vector are arbitrary i.e. one could run PCA on the same data set and get all the same numbers except the sign would be flipped.

PCA Algorithm

Let $\mathbf{x}_i$ denote a p-dimensional vectors (data points), $i = 1, \ldots, n$

1. Standardize the data (this step is needed if the variables have different units).
2. Calculate the $p \times p$ covariance matrix:

$$X^T X$$

3. Calculate the eigen decomposition of $X^T X$ i.e. eigen values and corresponding eigenvectors of the covariance matrix.
4. Select r eigenvectors that correspond to the largest r eigenvalues to be the new basis.