Data Reduction Techniques

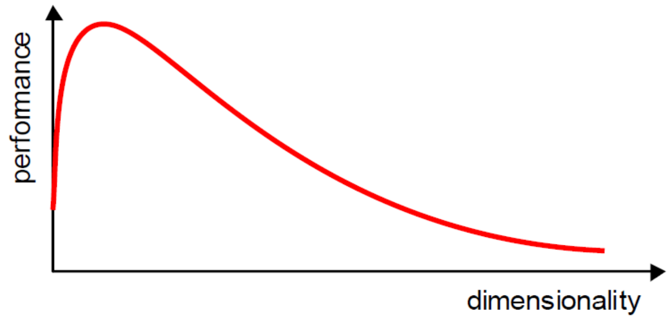
Motivation for Data Reduction

- Why data reduction? Why are more features bad?
  - Redundant features
  - Hard to interpret and visualize
  - Hard to store and process massive amount of data (computationally challenging)
  - Complex data analysis/mining may take a very long time to run on the complete data set
  - Curse of dimensionality

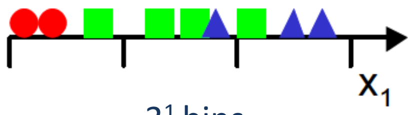
- Data reduction
  - Obtain a reduced representation of the data set that is much smaller in volume but yet produce the same (or almost the same) analytical results

Curse of Dimensionality

- Increasing the number of features will not always improve learning accuracy.



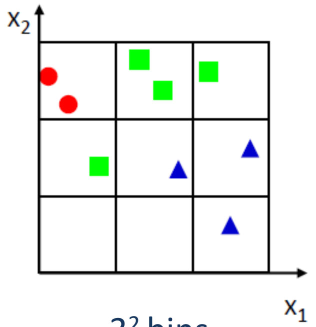
- In practice, the inclusion of more features might actually lead to worse performance.



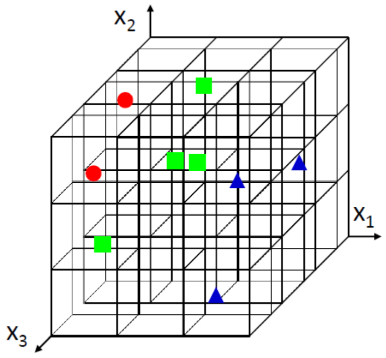
k=3

3^1 bins

- The number of training examples required increases exponentially with dimensionality **d** (i.e., k^d).



3^2 bins

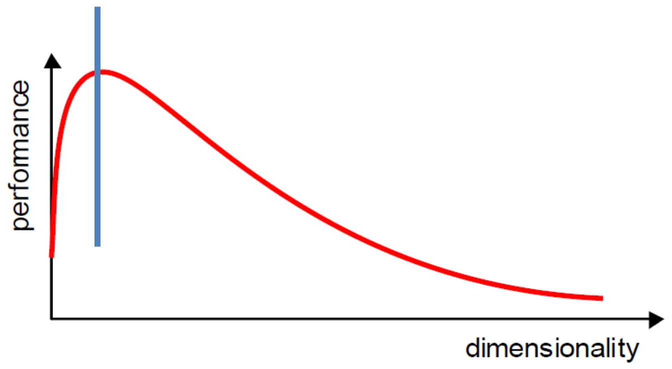


3^3 bins

k: number of bins per feature

Dimensionality Reduction

- What is the objective?
  - Choose an optimum set of features of lower dimensionality to improve classification accuracy.



- Different methods can be used to reduce dimensionality:
  - Feature extraction
  - Feature selection

Data Reduction Strategies

Feature extraction: finds a set of new features (i.e., through some mapping f()) from the existing features.

Feature selection: chooses a subset of the original features.

The mapping f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

K<<N

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ . \\ . \\ x_{i_K} \end{bmatrix}$$

K<<N

Data Reduction Strategies

- **Common Data reduction strategies**
  - Data cube aggregation:
  - Sampling
  - Numerosity reduction — e.g., fit data into models
  - Data Compression
  - Dimensionality reduction — e.g., remove unimportant attributes

Attribute Subset Selection

- **Feature selection** (i.e., attribute subset selection):
  - Select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features
  - reduce # of patterns in the patterns, easier to understand

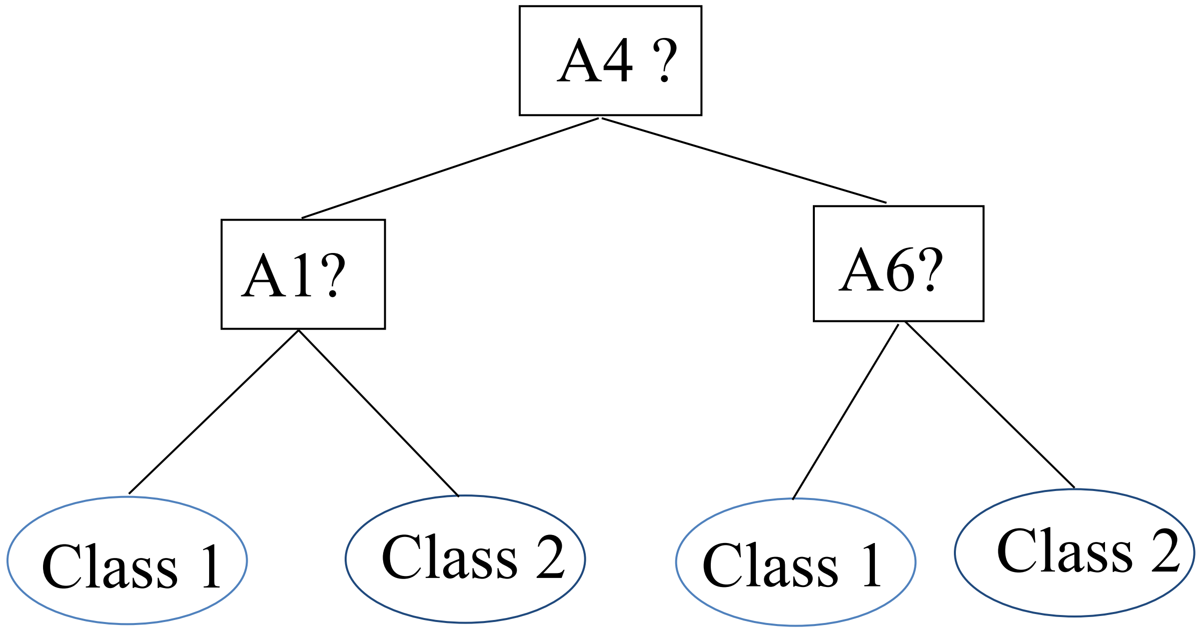
- **Heuristic methods** (due to exponential # of choices):
  - Step-wise forward selection
  - Step-wise backward elimination
  - Combining forward selection and backward elimination
  - Decision-tree induction

Heuristic Feature Selection Methods

- There are 2^d possible sub-features of d features
- Several heuristic feature selection methods:
  - Best single features under the feature independence assumption: choose by significance tests
  - Best step-wise feature selection:
    - The best single-feature is picked first
    - Then next best feature condition to the first, ...
  - Step-wise feature elimination:
    - Repeatedly eliminate the worst feature
  - Best combined feature selection and elimination
  - Optimal branch and bound:
    - Use feature elimination and backtracking

Example of Decision Tree Induction

Initial attribute set:
{A1, A2, A3, A4, A5, A6}

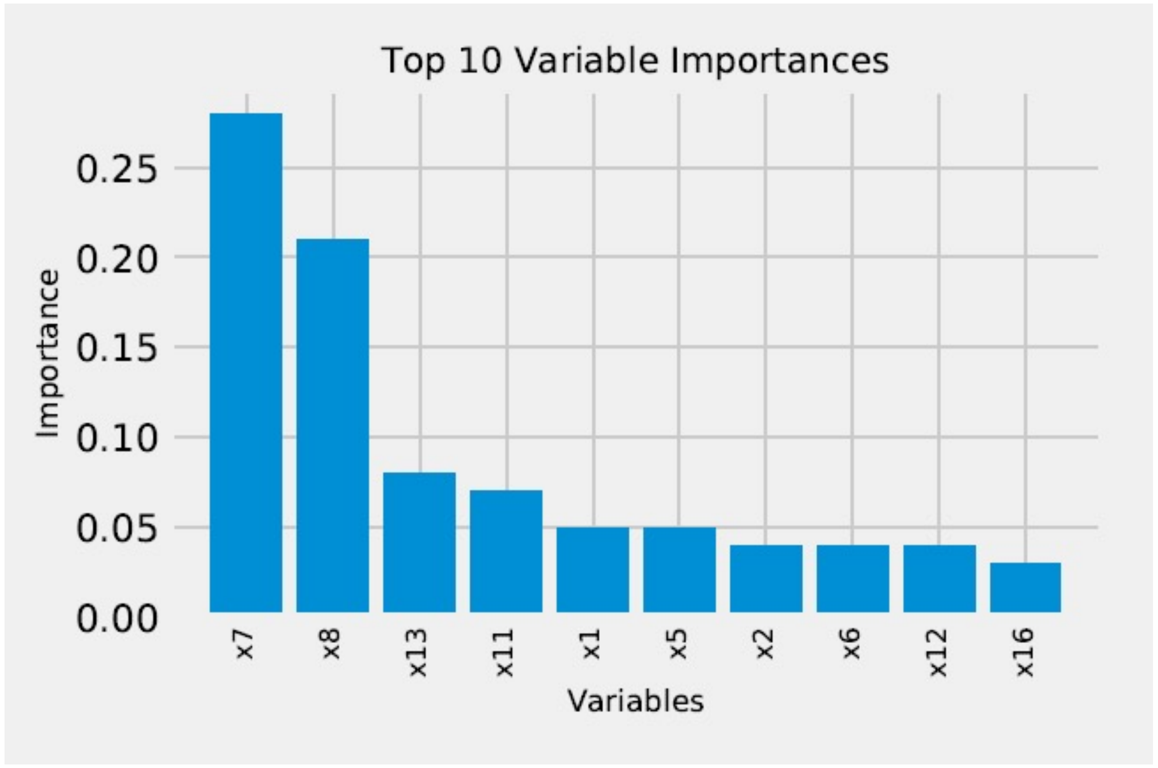


------> Reduced attribute set: {A1, A4, A6}

Random Forest Feature Importancy

Random forests provide two methods for feature selection:

- mean decrease gini impurity
- mean decrease accuracy

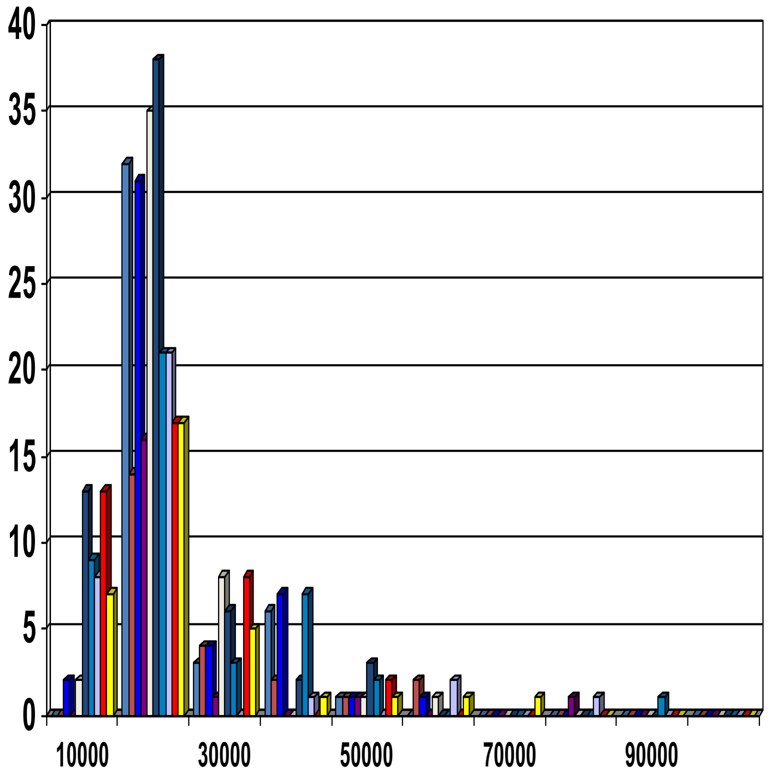


Numerosity Reduction

- Reduce data volume by choosing alternative, smaller forms of data representation
- Parametric methods
  - Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers)
  - Example: Log-linear models—obtain value at a point in m-D space as the product on appropriate marginal subspaces
- Non-parametric methods
  - Do not assume models
  - Major families: histograms, clustering, sampling

Data Reduction Method (2): Histograms

- Divide data into buckets and store average (sum) for each bucket



Data Reduction Method (3): Clustering

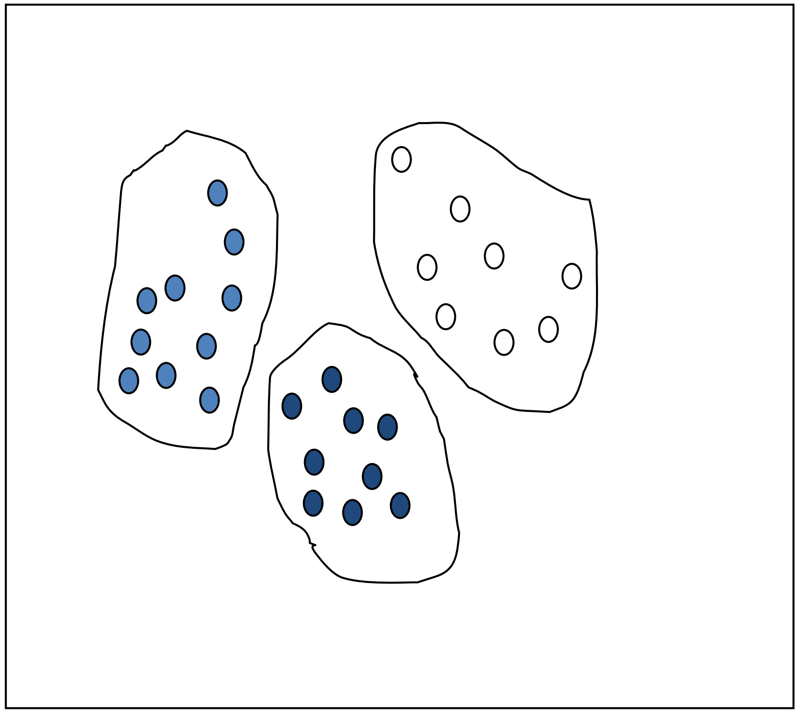
- Partition data set into clusters based on similarity, and store cluster representation (e.g., centroid and diameter) only
- Can be very effective if data is clustered but not if data is “smeared”
- Can have hierarchical clustering and be stored in multi-dimensional index tree structures
- There are many choices of clustering definitions and clustering algorithms

Data Reduction Method (4): Sampling

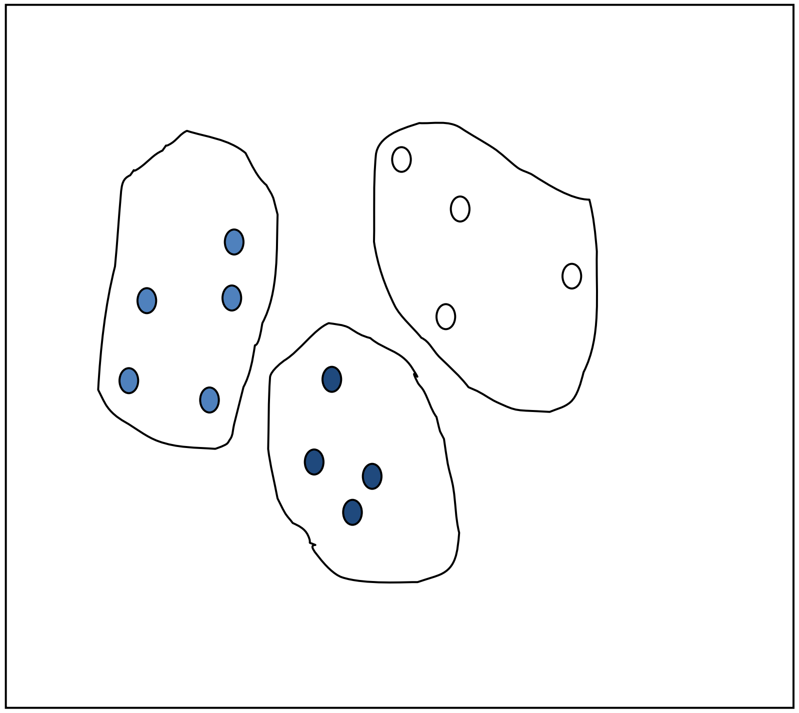
- Sampling: obtaining a small sample *s* to represent the whole data set *N*
- Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data
- Choose a representative subset of the data
  - Simple random sampling may have very poor performance in the presence of skew
- Develop adaptive sampling methods
  - Stratified sampling:
    - Approximate the percentage of each class (or subpopulation of interest) in the overall data
    - Used in conjunction with skewed data

Sampling: Cluster or Stratified Sampling

Raw Data



Cluster/Stratified Sample



Data Reduction Method (5): Dimensionality Reduction:

- Dimensionality reduction
  - Another way to simplify complex high-dimensional data
  - Summarize data with a lower dimensional real valued vector

- Given data points in *d* dimensions
- Convert them to data points in *r<d* dimensions
- With minimal loss of information

Feature Extraction (cont'd)

- From a mathematical point of view, finding an optimum mapping **y**=f(**x**) is equivalent to optimizing an **objective** criterion.

- Different methods use different objective criteria, e.g.,

  - Minimize Information Loss: represent the data as accurately as possible in the lower-dimensional space.

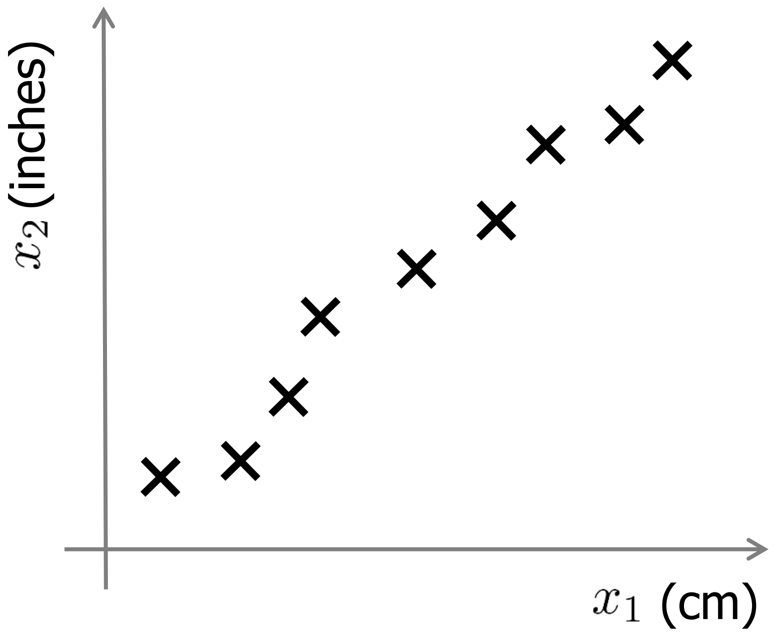
  - Maximize Discriminatory Information: enhance the class-discriminatory information in the lower-dimensional space.

Feature Extraction (cont'd)

- Popular linear feature extraction methods:
  - Principal Components Analysis (PCA): Seeks a projection that **preserves** as much **information** in the data as possible.
  - Linear Discriminant Analysis (LDA): Maximizing the component axes for **class-separation**

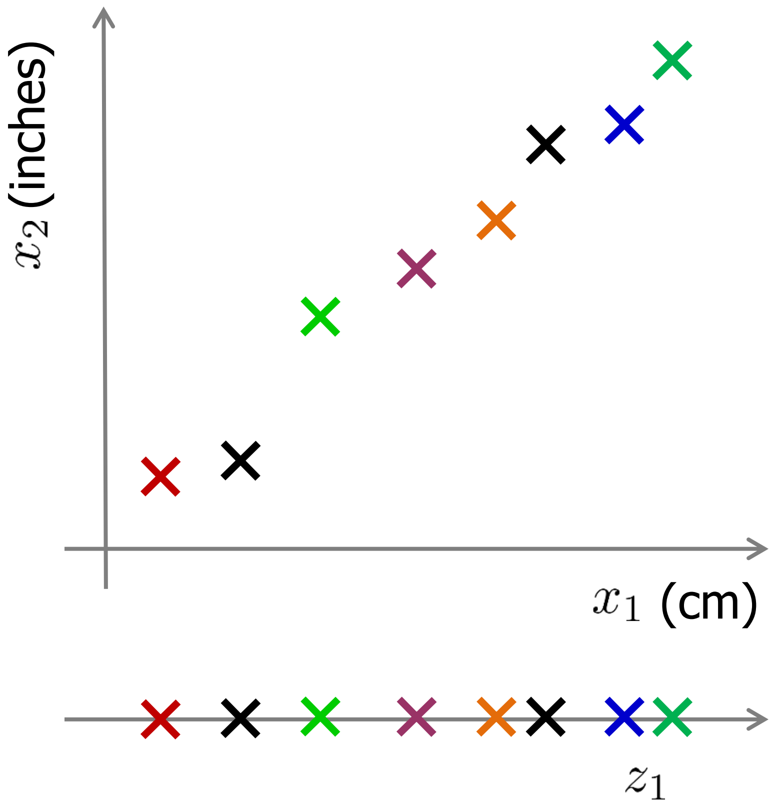
- Many other methods:
  - Independent Component Analysis or ICA-Making features as independent as possible (Very similar to PCA except that it assumes non-Guassian features).
  - Multidimensional Scaling: Find projection that best preserves inter-point distances.
  - Embedding to lower dimensional manifolds (Isomap, Locally Linear Embedding or LLE, T-Distributed Stochastic Neighbour Embedding or tSNE).
  - … etc

Data Compression



Reduce data from 2D to 1D

Data Compression



Reduce data from 2D to 1D

$$x^{(1)} \rightarrow z^{(1)}$$

$$x^{(2)} \rightarrow z^{(2)}$$

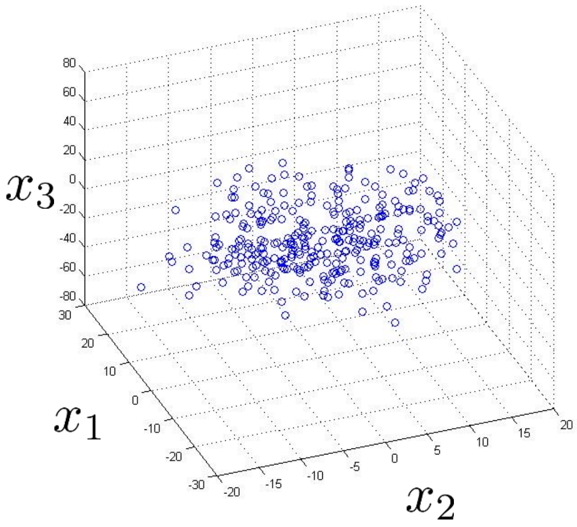
$$\vdots$$

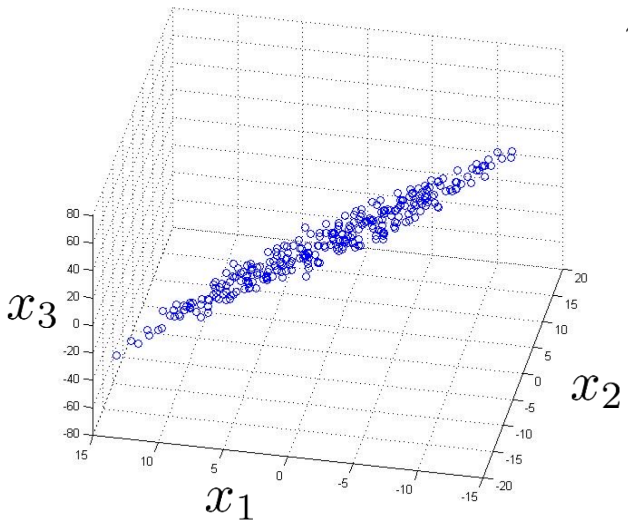
$$x^{(m)} \rightarrow z^{(m)}$$

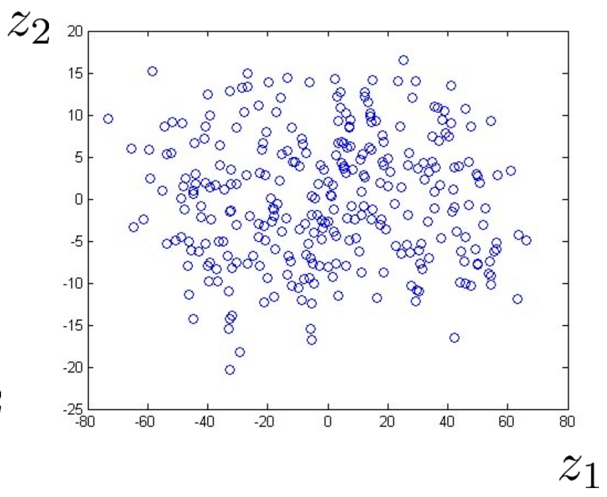
Andrew Ng

Data Compression

Reduce data from 3D to 2D



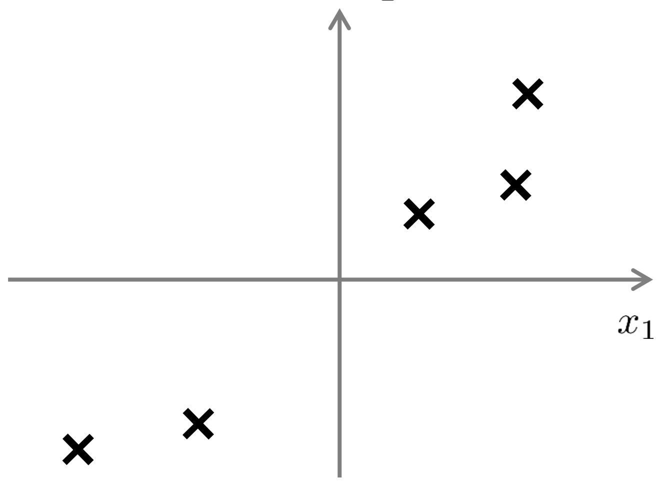


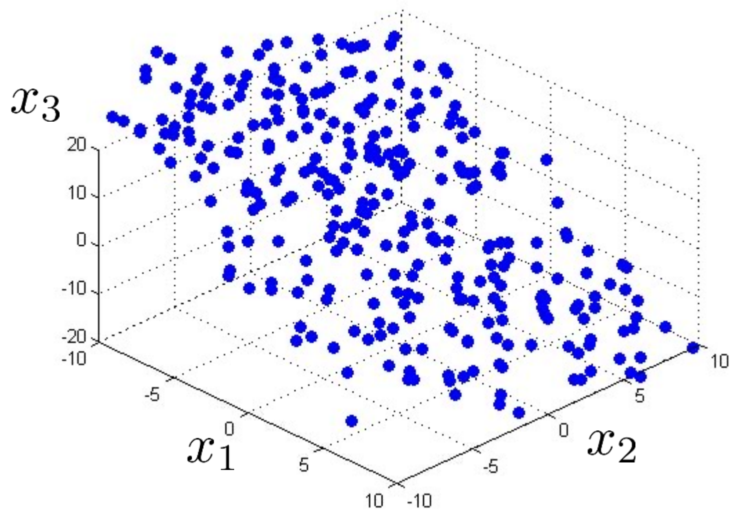


Andrew Ng

Principal Component Analysis (PCA) problem formulation

$3D \to 2D$
$K = 2$





Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

Reduce from n-dimension to k-dimension: Find k vectors $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ onto which to project the data, so as to minimize the projection error.

Andrew Ng

Principal Component Analysis (PCA)

- Given N data vectors from n-dimensions, find $k \le n$ **orthogonal vectors** (*principal components*) that can be best used to **represent data**.
- **PCA maximizes the variance** on transformation.
- **Algorithm Steps**
  - Normalize input data: Each attribute falls within the same range
  - Compute k orthonormal (unit) vectors, i.e., *principal components*
  - Each input data (vector) is a linear combination of the k principal component vectors
  - The principal components are sorted in order of decreasing "significance" or strength
  - Since the components are sorted, the size of the data can be reduced by eliminating the weak components, i.e., those with low variance. (i.e., using the strongest principal components, it is possible to reconstruct a good approximation of the original data
- **Works for numeric data only**
- Used when the **number of dimensions is large**

Principal Component Analysis

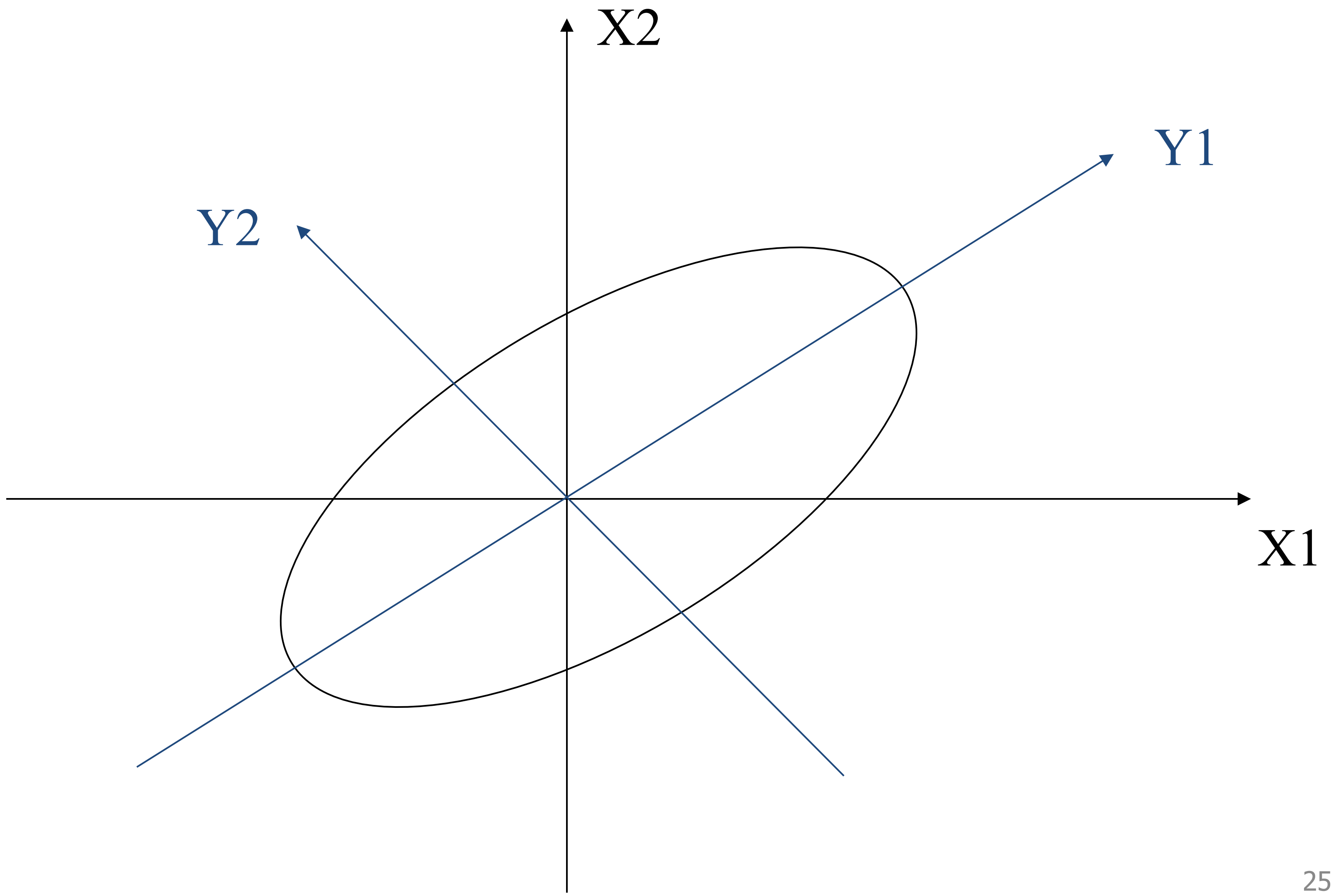
Goal: Find r-dim projection that best preserves variance

1. Compute mean vector μ and covariance matrix Σ of original points
2. Compute eigenvectors and eigenvalues of Σ
3. Select top r eigenvectors
4. Project points onto subspace spanned by them:

$$y = A(x - \mu)$$

   where y is the new point, x is the old one, and the rows of A are the eigenvectors

Principal Component Analysis

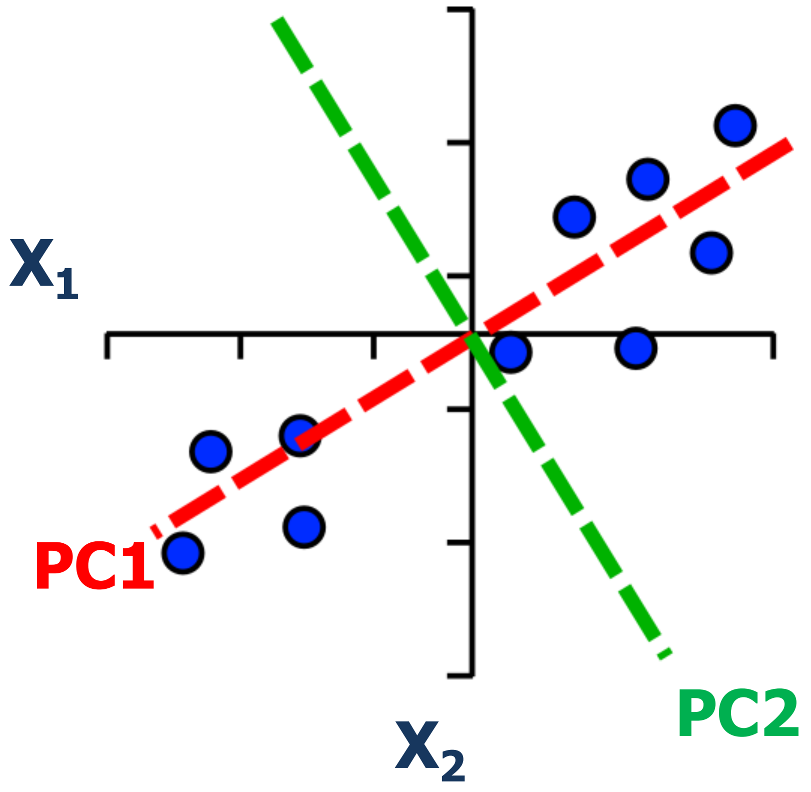


Explaining Variance

- Each PC always explains some proportion of the total variance in the data. Between them they explain everything
  - PC1 always explains the most
  - PC2 is the next highest etc. etc.

- The same idea extends to larger numbers of dimensions (n)
- How do we calculate this?

Explaining variance



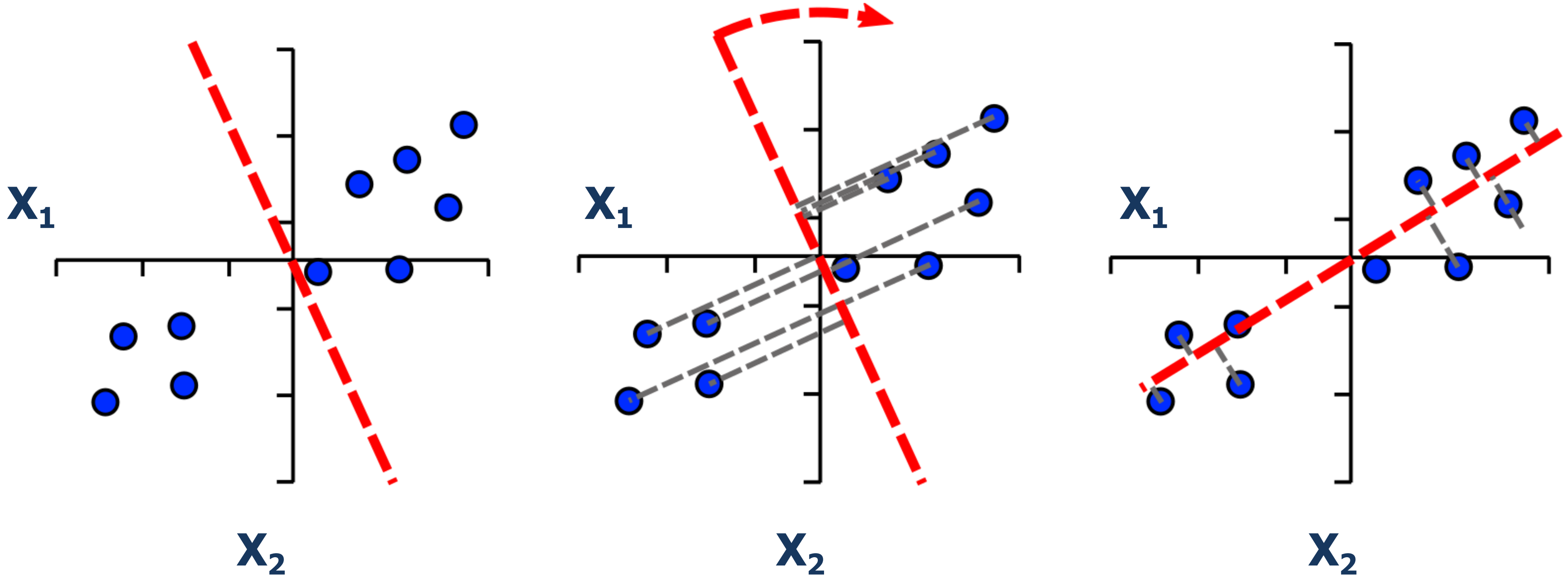
- Project onto PC
- Calculate distance to the origin

- Calculate sum of squared differences (SSD)
  – This is a measure of variance called the 'eigenvalue'

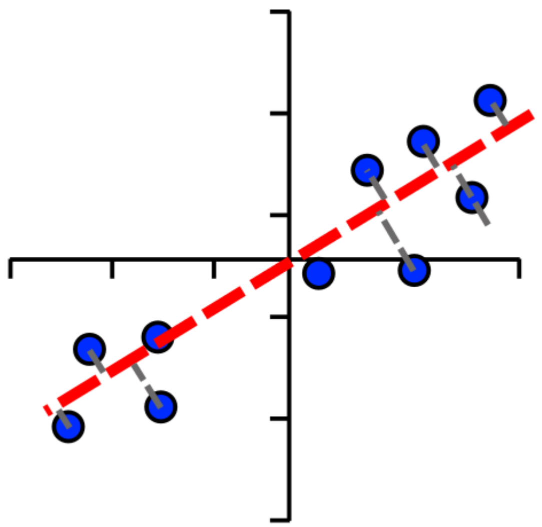
  – Divide by (points-1) to get actual variance

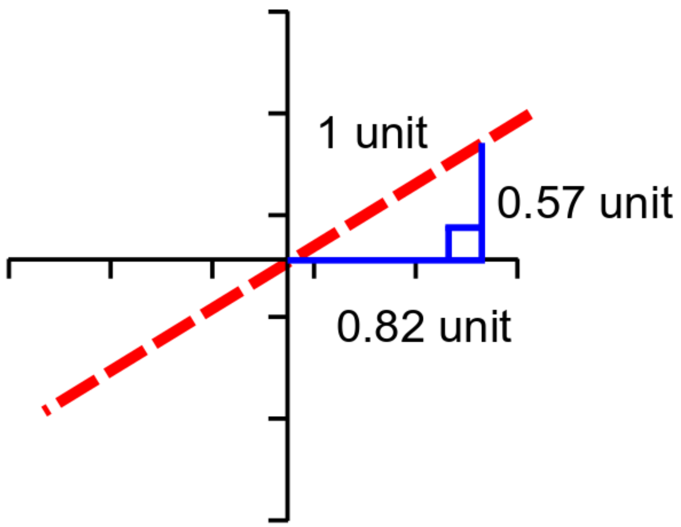
How does PCA work?

- Find line of best fit, passing through the origin



Contributions of Features to PCs





Single Vector or '**eigenvector**'

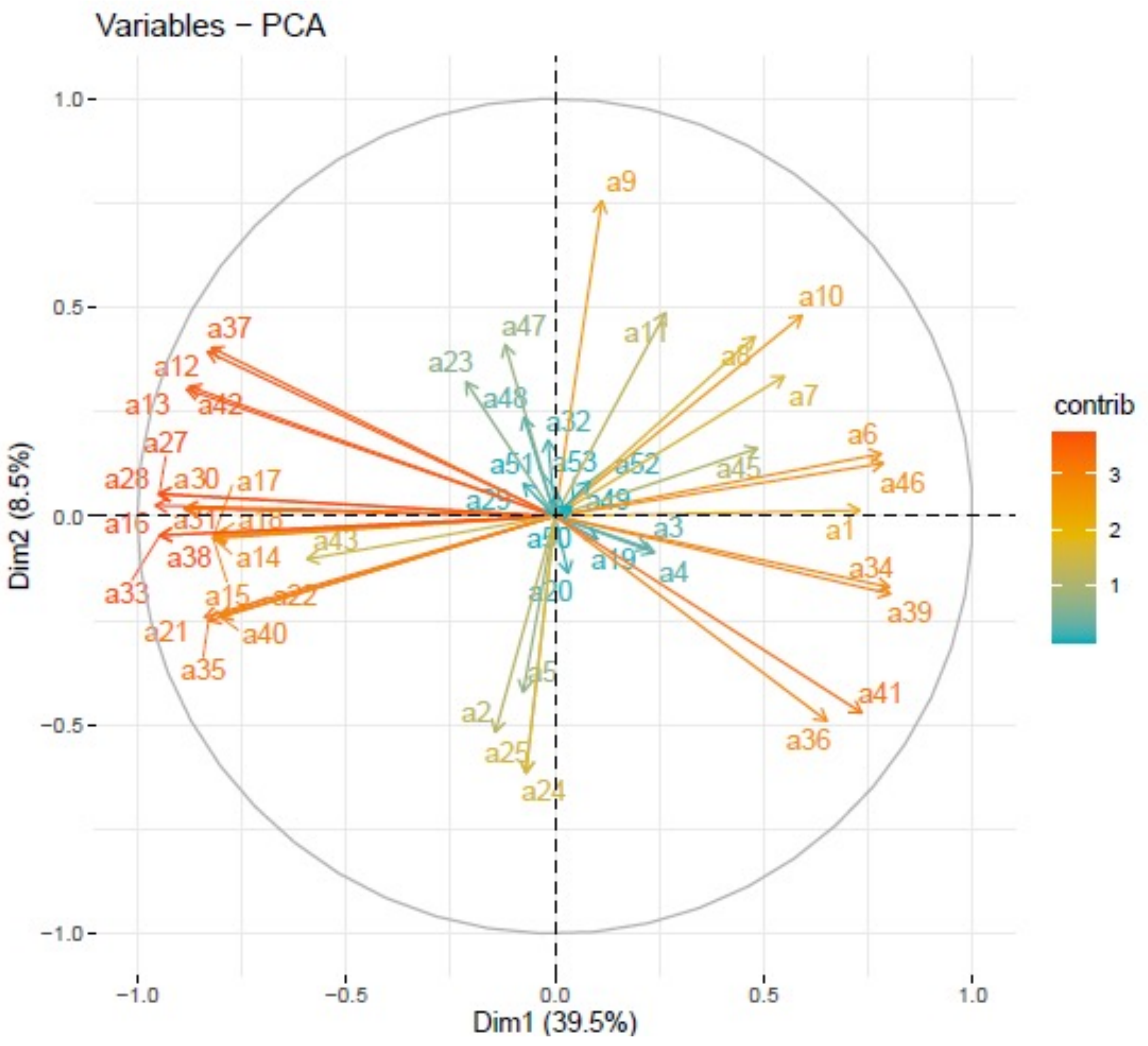
Loadings:

- X1 = 0.82
- X2 = 0.57

Higher loading equals more influence on PC

$PC = (X_1*0.82)+(X_2*0.57)+(X_3*0.12)+\ldots$

Principal Component Analysis



Vector Representation

- A vector **x** ϵ R^n can be represented by n components:

$$\mathbf{x}: \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix}$$

- Assuming the standard base <v_1, v_2, …, v_N> (i.e., unit vectors in each dimension), x_i can be obtained by projecting **x** along the direction of v_i:

$$x_i = \frac{\mathbf{x}^T v_i}{v_i^T v_i} = \mathbf{x}^T v_i$$

- **x** can be "reconstructed" from its projections as follows:

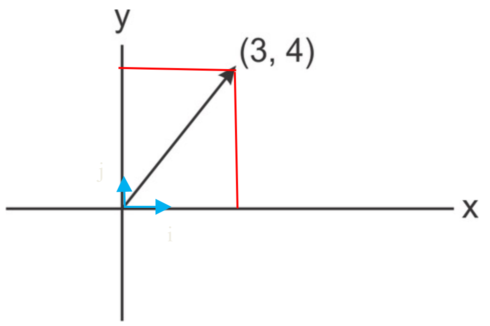
$$\mathbf{x} = \sum_{i=1}^{N} x_i v_i = x_1 v_1 + x_2 v_2 + \ldots + x_N v_N$$

- Since the basis vectors are the same for all x ϵ R^n (standard basis), we typically represent them as a n-component vector.

Vector Representation (cont'd)

- **Example** assuming n=2:

$$\mathbf{x}:\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$



- Assuming the standard base <v$_1$=i, v$_2$=j>, x$_i$ can be obtained by projecting x along the direction of v$_i$:

$$x_1 = \mathbf{x}^T i = \begin{bmatrix} 3 & 4 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = 3$$

$$x_2 = \mathbf{x}^T j = \begin{bmatrix} 3 & 4 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = 4$$

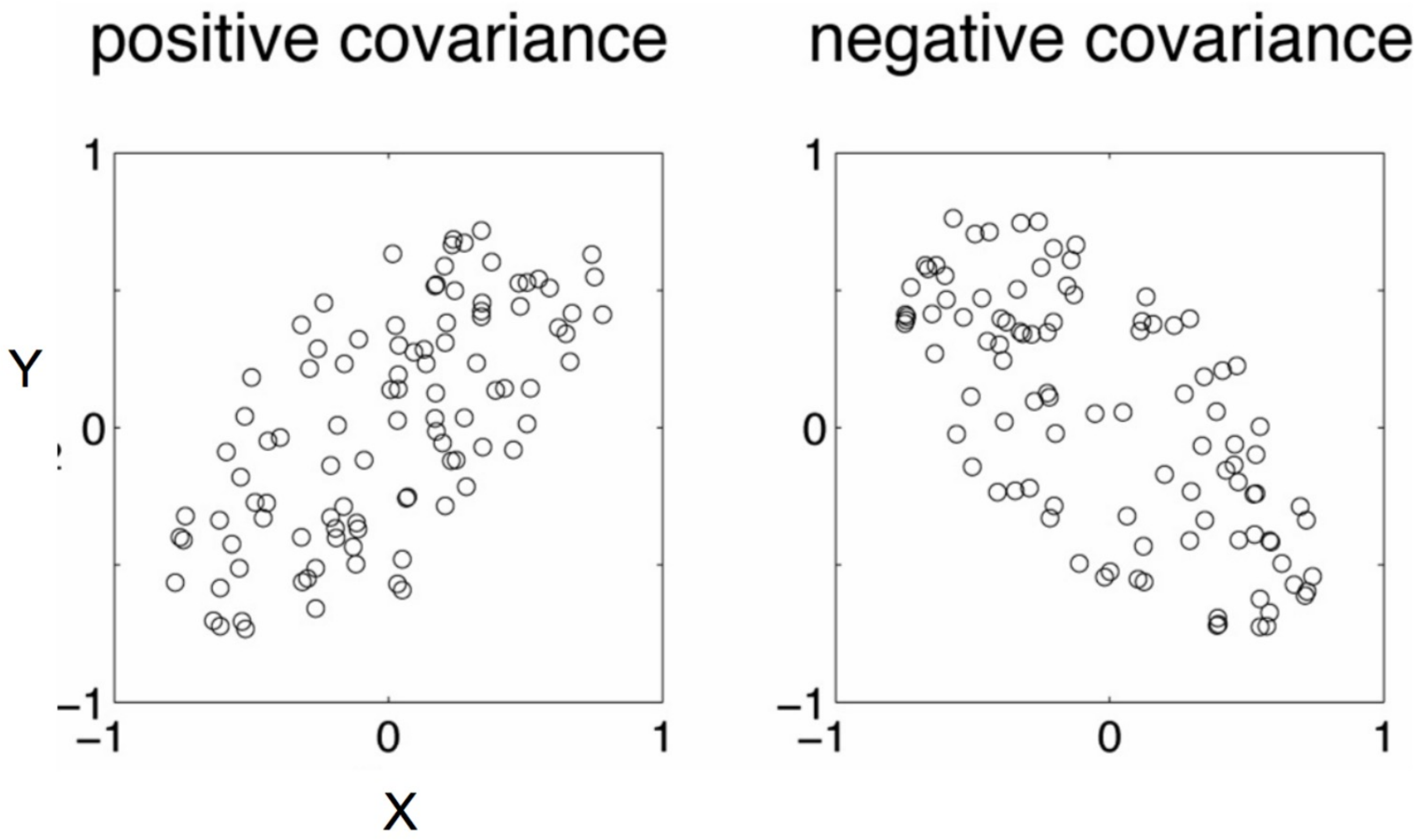
- **x** can be "reconstructed" from its projections as follows:

$$\mathbf{x} = 3i + 4j$$

Covariance

- Variance and Covariance:
  - Measure of the "spread" of a set of points around their center of mass(mean)
- Variance:
  - Measure of the deviation from the mean for points in one dimension
- Covariance:
  - Measure of how much each of the dimensions vary from the mean with **respect to each other**

- **Covariance is measured between two dimensions**
- **Covariance sees if there is a relation between two dimensions**
- **Covariance between one dimension is the variance**



Positive: Both dimensions increase or decrease together **Negative: While one increase the other decrease**

Covariance

- Used to find relationships between dimensions in high dimensional data sets

$$q_{jk} = \frac{1}{N} \sum_{i=1}^{N} (X_{ij} - \underset{\substack{\big\downarrow \\ \text{The Sample mean}}}{E(X_j)})(X_{ik} - E(X_k))$$

Eigenvector and Eigenvalue

$$Ax = \lambda x$$

A: Square Matrix
λ: Eigenvector or characteristic vector
X: Eigenvalue or characteristic value

- ***The zero vector can not be an eigenvector***
- ***The value zero can be eigenvalue***

Eigenvector and Eigenvalue

$Ax = \lambda x$

A: Square Matrix
λ: Eigenvector or characteristic vector
X: Eigenvalue or characteristic value

Example

$$Show\ x = \begin{bmatrix} 2 \\ 1 \end{bmatrix} is\, an\, eigenvector\ for\ A = \begin{bmatrix} 2 & -4 \\ 3 & -6 \end{bmatrix}$$

$$Solution: Ax = \begin{bmatrix} 2 & -4 \\ 3 & -6 \end{bmatrix}\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$But\ \ for\ \ \lambda = 0,\ \ \lambda x = 0\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$Thus, x\, is\, an\, eigenvector\, of\ A, and\, \lambda = 0\ is\, an\, eigenvalue.$$

Eigenvector and Eigenvalue

$$Ax = \lambda x \longrightarrow \begin{array}{l} Ax - \lambda x = 0 \\ (A - \lambda I)x = 0 \end{array}$$

If we define a new matrix B: $\longrightarrow$ $$\begin{array}{l} B = A - \lambda I \\ Bx = 0 \end{array}$$

If B has an inverse: $\longrightarrow$ $x = B^{-1}0 = 0$ ✗ **BUT! an eigenvector cannot be zero!!**

$\Rightarrow$ **x will be an eigenvector of A if and only if B does not have an inverse, or equivalently det(B)=0 :**

$$\boxed{\det(A - \lambda I) = 0}$$

Eigenvector and Eigenvalue

Example 1: Find the eigenvalues of

$$A = \begin{bmatrix} 2 & -12 \\ 1 & -5 \end{bmatrix}$$

$$|\lambda I - A| = \begin{vmatrix} \lambda - 2 & 12 \\ -1 & \lambda + 5 \end{vmatrix} = (\lambda - 2)(\lambda + 5) + 12$$

$$= \lambda^2 + 3\lambda + 2 = (\lambda + 1)(\lambda + 2)$$

two eigenvalues: $-1, -2$

Note: The roots of the characteristic equation can be repeated. That is, $\lambda_1 = \lambda_2 = \ldots = \lambda_k$. If that happens, the eigenvalue is said to be of multiplicity k.

Example 2: Find the eigenvalues of

$$A = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

$$|\lambda I - A| = \begin{vmatrix} \lambda - 2 & -1 & 0 \\ 0 & \lambda - 2 & 0 \\ 0 & 0 & \lambda - 2 \end{vmatrix} = (\lambda - 2)^3 = 0$$

$\lambda = 2$ is an eigenvector of multiplicity 3.

Principal Component Analysis

Input: $\mathbf{x} \in \mathbb{R}^D$: $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Set of basis vectors: $\mathbf{u}_1, \ldots, \mathbf{u}_K$

Summarize a *D* dimensional vector *X* with *K* dimensional feature vector *h(x)*

$$h(\mathbf{x}) = \begin{bmatrix} \mathbf{u}_1 \cdot \mathbf{x} \\ \mathbf{u}_2 \cdot \mathbf{x} \\ \cdots \\ \mathbf{u}_K \cdot \mathbf{x} \end{bmatrix}$$

Principal Component Analysis

$$\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_K]$$

Basis vectors are orthonormal

$$\mathbf{u}_i^T \mathbf{u}_j = 0$$

$$||\mathbf{u}_j|| = 1$$

New data representation *h(x)*

$$z_j = \mathbf{u}_j \cdot \mathbf{x}$$

$$h(\mathbf{x}) = [z_1, \ldots, z_K]^T$$

Principal Component Analysis

$$\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_K]$$

New data representation *h(x)*

$$h(\mathbf{x}) = \mathbf{U}^T \mathbf{x}$$

$$h(\mathbf{x}) = \mathbf{U}^T (\mathbf{x} - \mu_0)$$

Empirical mean of the data $\longrightarrow$ $\mu_0 = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i$

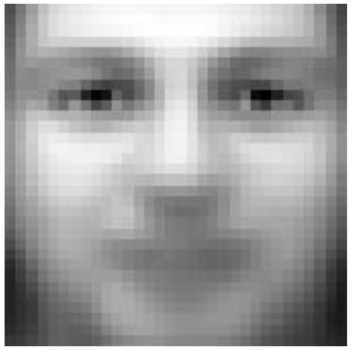
The space of all face images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage
- But very few 10,000-dimensional vectors are valid face images
- We want to effectively model the subspace of face images

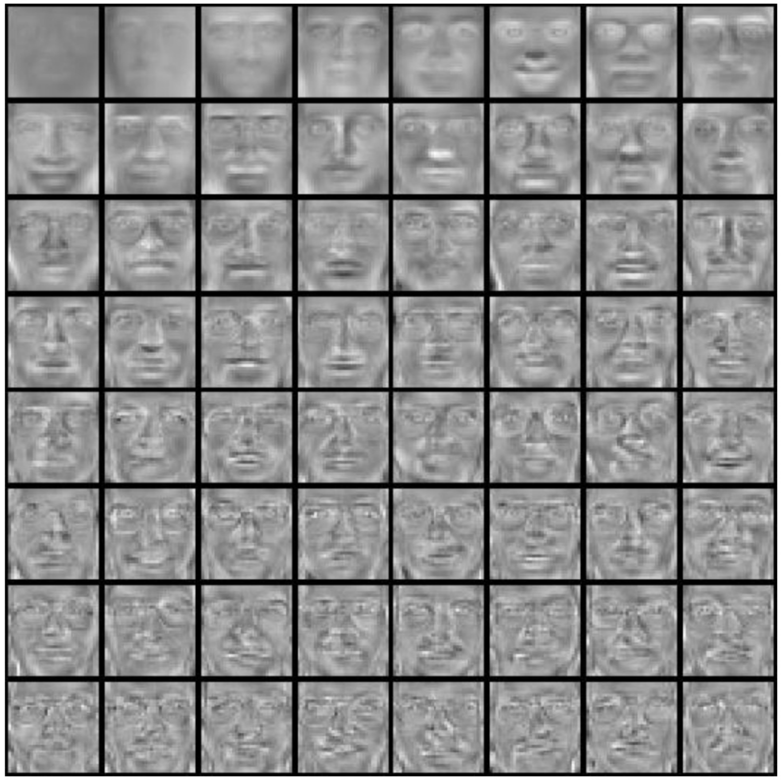
slide by Derek Hoiem

Eigenfaces example

Mean: μ



Top eigenvectors: $u_1, \ldots u_k$



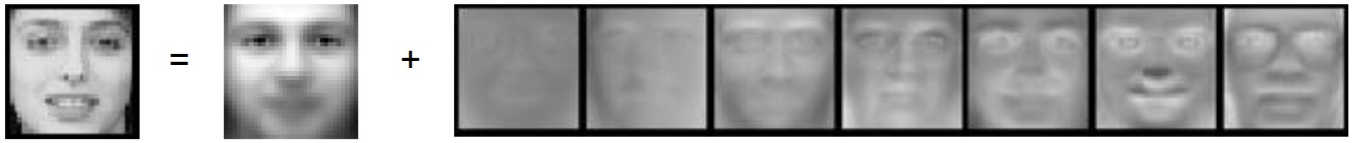
slide by Derek Hoiem

Representation and reconstruction

- Face **x** in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x} - \mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x} - \mu)]$$
$$= \quad w_1, \ldots, w_k$$

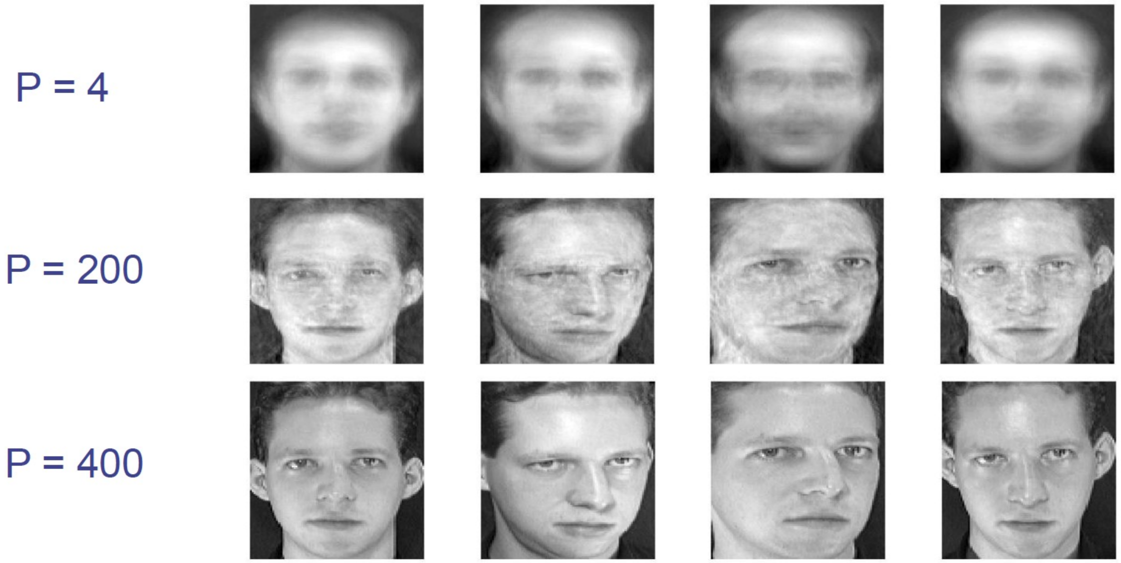
- Reconstruction:



$\hat{x} = \mu + w_1u_1+w_2u_2+w_3u_3+w_4u_4+ \ldots$

slide by Derek Hoiem

Reconstruction



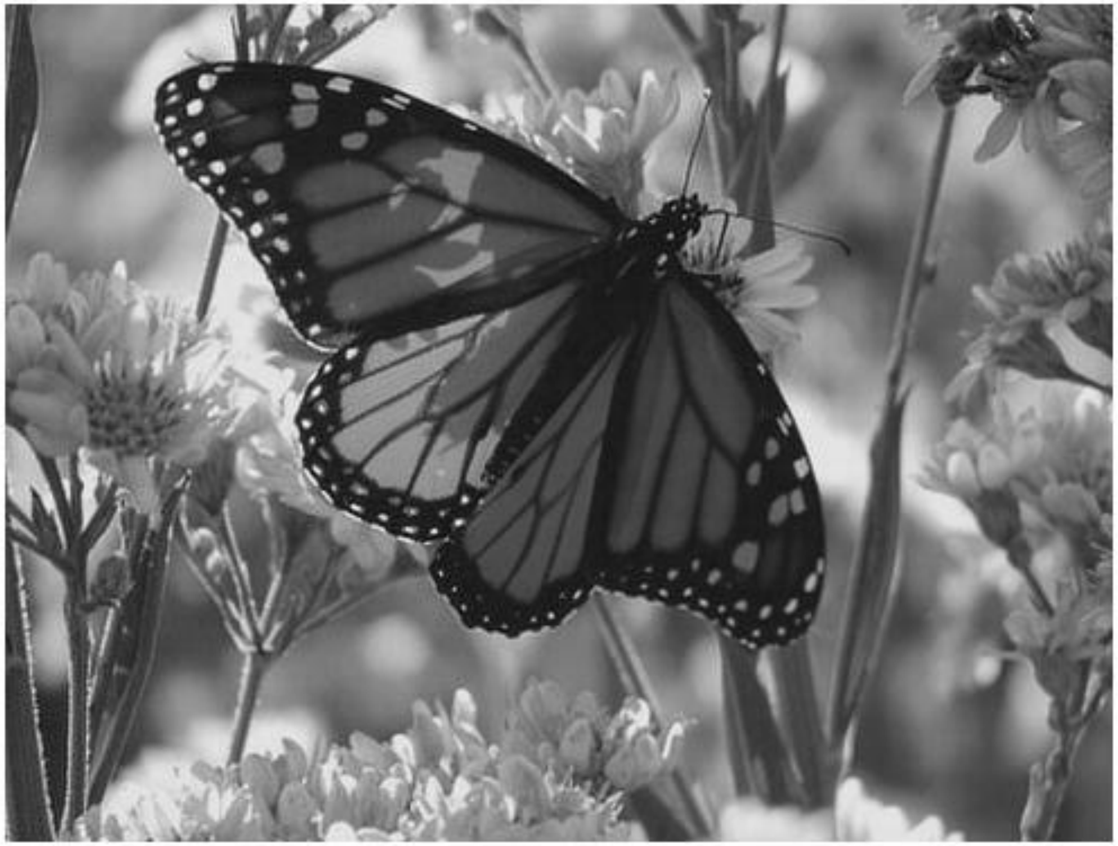
After computing eigenfaces using 400 face images from ORL face database

Application: Image compression

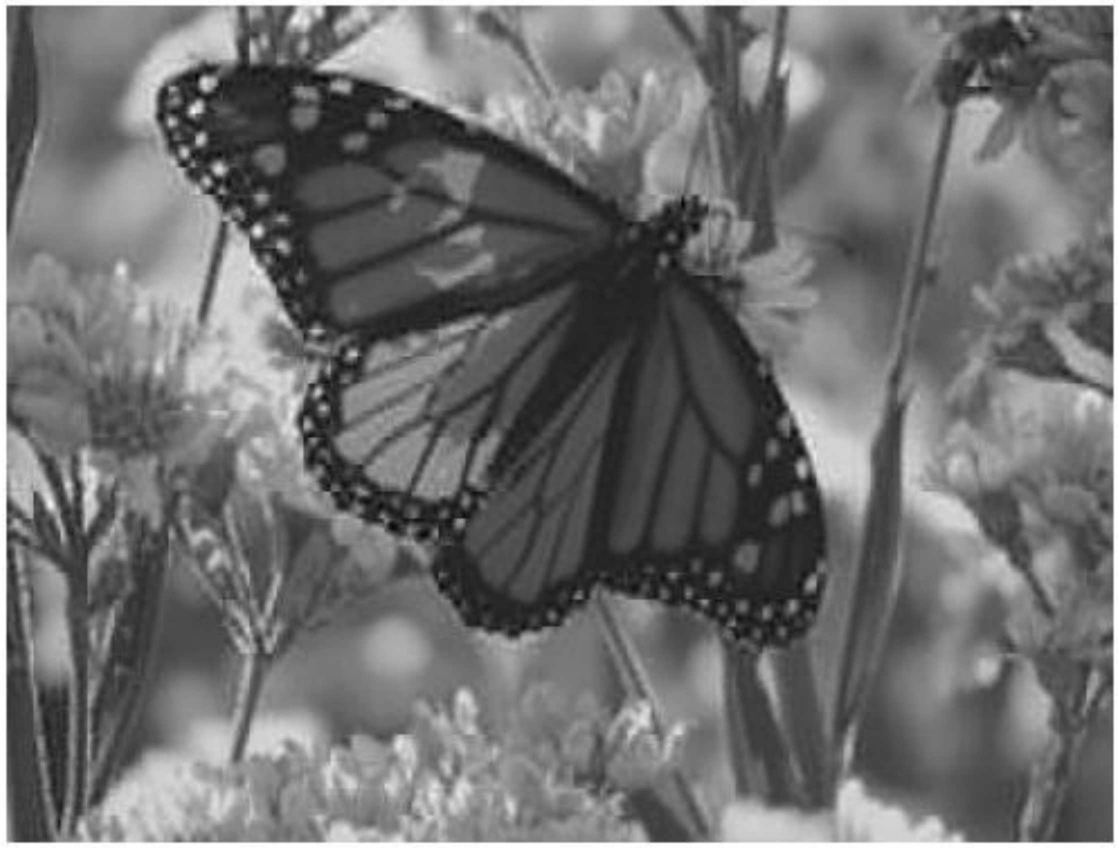
Original Image

- Divide the original 372x492 image into patches:
  - Each patch is an instance that contains 12x12 pixels on a grid
- View each as a 144-D vector

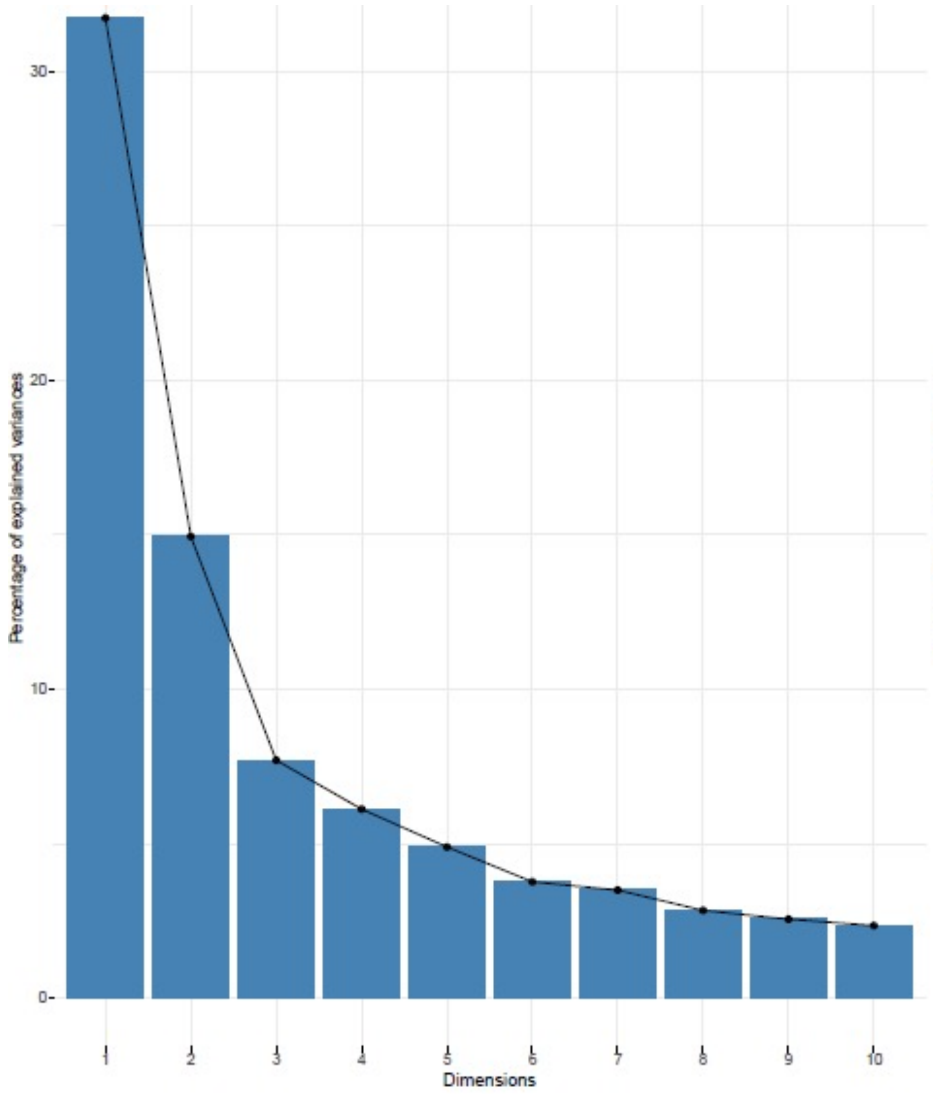
PCA compression: 144D → 60D

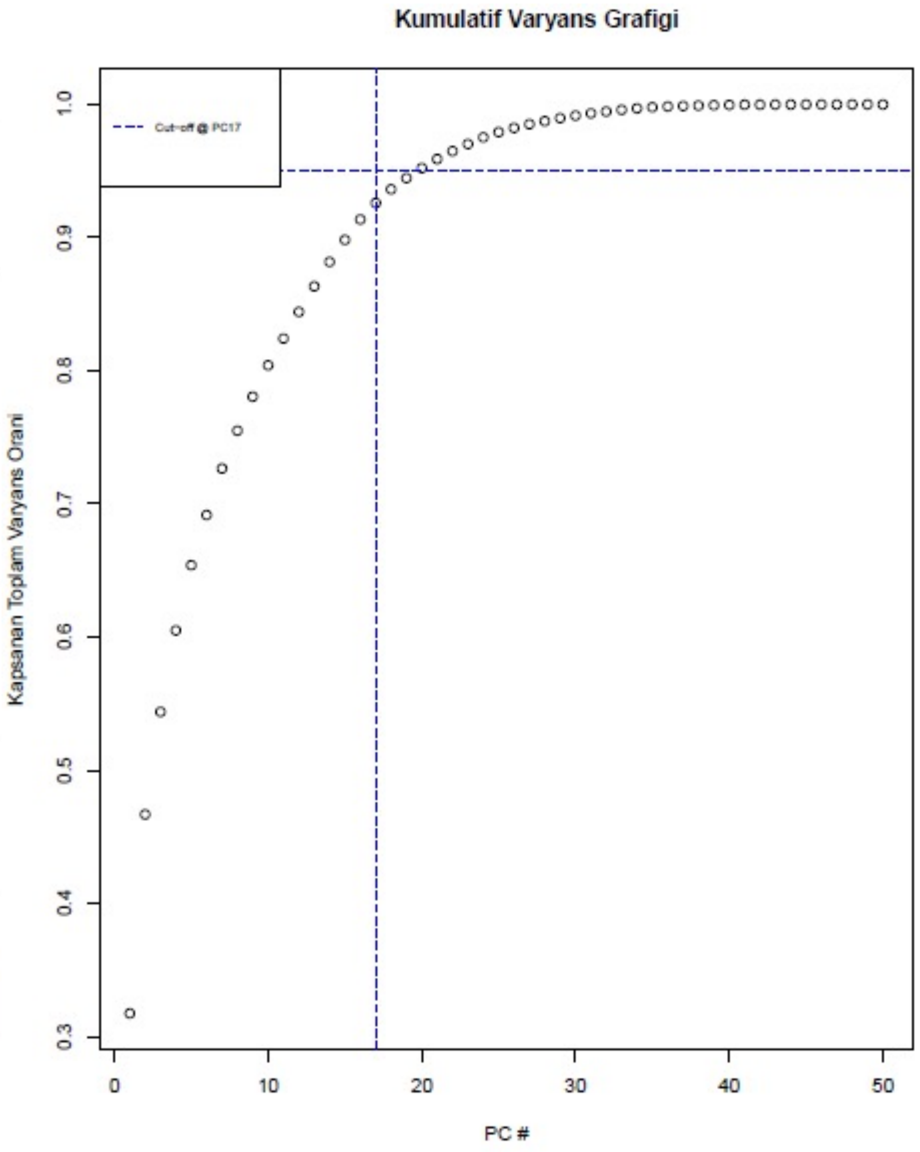


PCA compression: 144D → 16D



How to Choose #of PCs?


Percentage of explained variances
Dimensions


Kumulatif Varyans Grafigi
Cut-off @ PC17
Kapsanan Toplam Varyans Orani
PC #