Features Dimensionality Reduction

PLAKSHA UNIVERSITY

Features, More Features, Even More Features

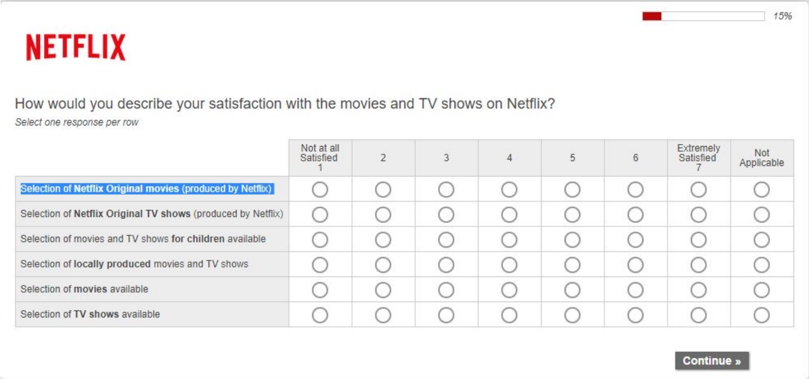
- It's generally a good idea to extract as many diverse features from data as possible. After all, any ML model utilizes the patterns in the features for training. More number of diverse features may provide more number of unique patterns.

- It is also good to extract more features because a fewer number of features may just not be enough for training efficiently any ML model.

- But, having too many features may also pose some problems because the ML model may not work very well on a large dimensional space.

Features, More Features, Even More Features

Thousands of documents
Millions of words!
Millions of contexts between words!



NETFLIX

15%

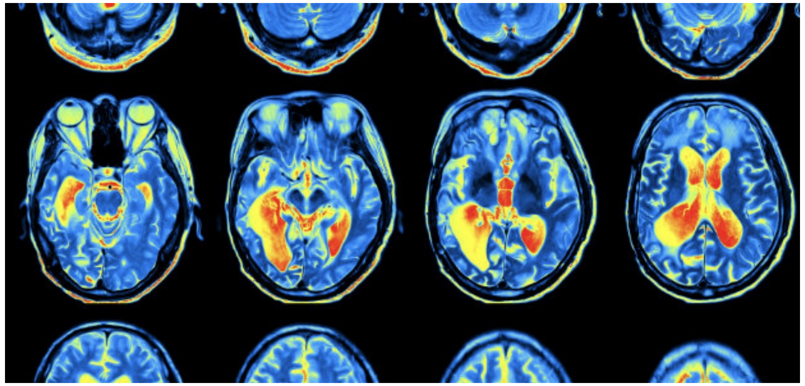
How would you describe your satisfaction with the movies and TV shows on Netflix?
Select one response per row

	Not at all Satisfied 1	2	3	4	5	6	Extremely Satisfied 7	Not Applicable
Selection of **Netflix Original movies** (produced by Netflix)								
Selection of **Netflix Original TV shows** (produced by Netflix)								
Selection of movies and TV shows **for children** available								
Selection of **locally produced** movies and TV shows								
Selection of **movies** available								
Selection of **TV shows** available								

Continue »

Thousands of movies and webseries.
Each with tens of questions in a survey!
Each with hundreds of features based on cast, content, genre, etc.

Features, More Features, Even More Features

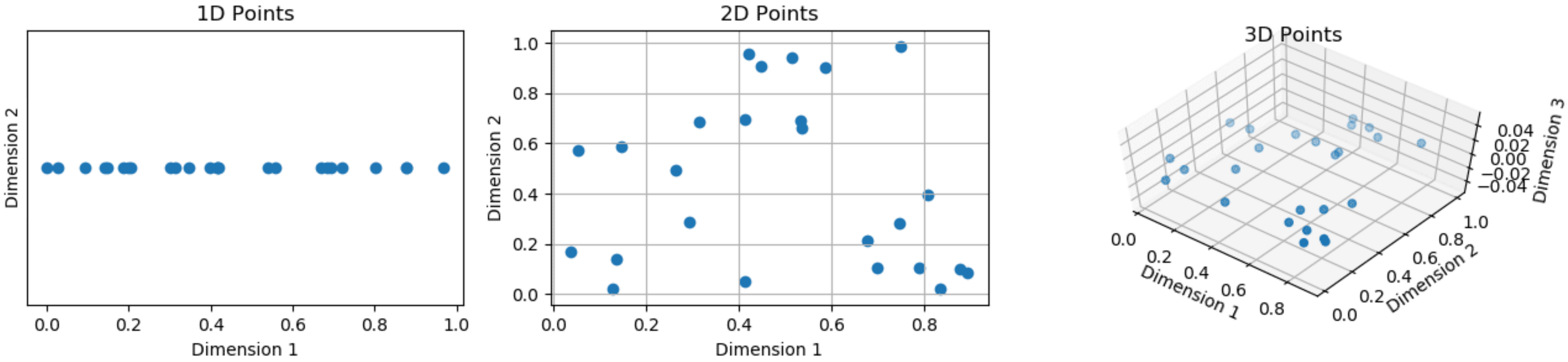


Hundreds of fMRI brain scans.
Each scan with thousands of locations (voxels)!
Many scans over time!

Millions of seconds of financial data.
Each datapoint with many features from multiple stocks!
Each stock with many features from the company's declarations!

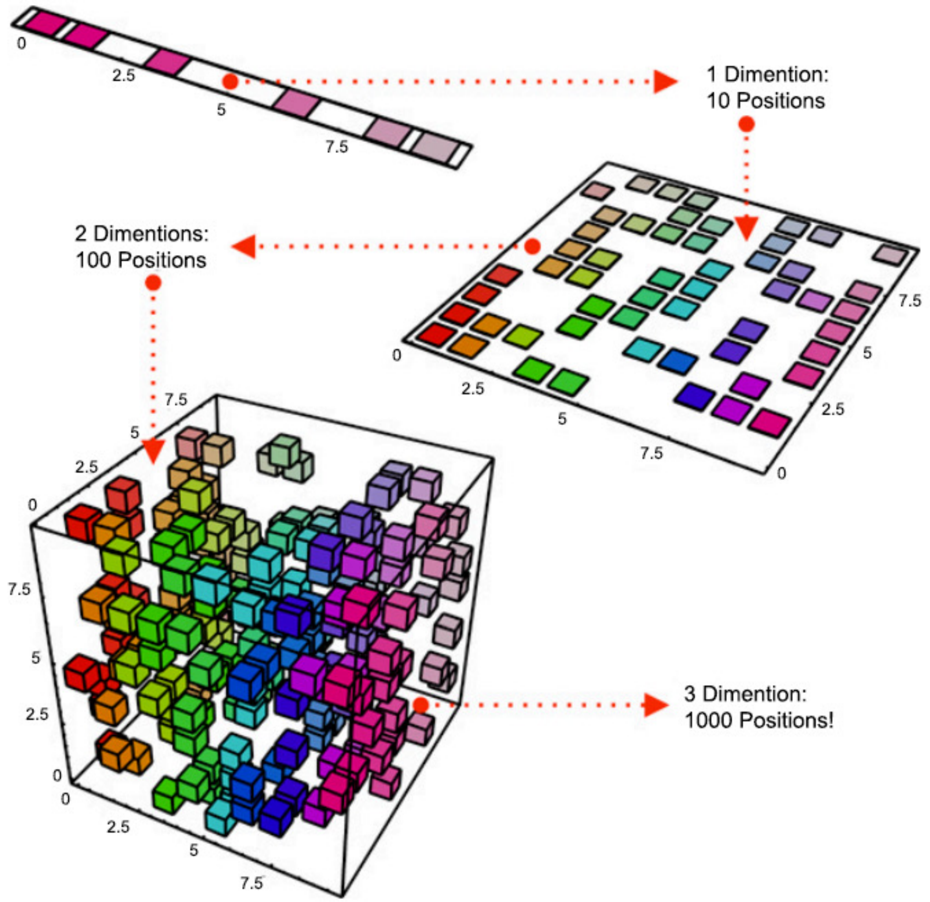
Features, More Features, Even More Features

As we increase the number of features, the data starts to become sparse.



So, we need to go and collect even more data as the number of features increase so that the ML model does not overfit.

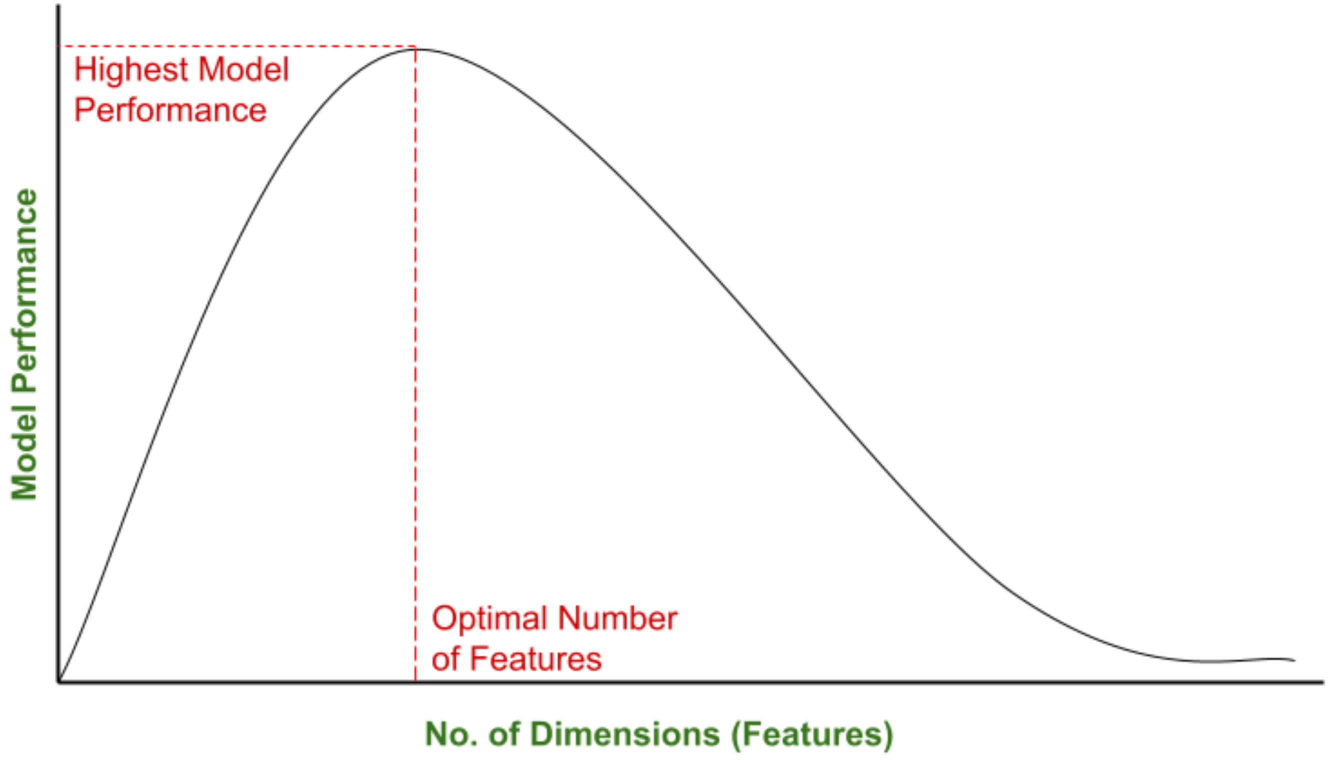
Features, More Features, Even More Features



To have the same average distance between the datapoints, we need to add more and more datapoints (i.e., collect more data) as we increase the number of features i.e., dimensionality.

The Curse of Dimensionality

High number of features may not always be optimal to train an ML model.



In general, the highest model performance may not be achieved by using a large number of features.

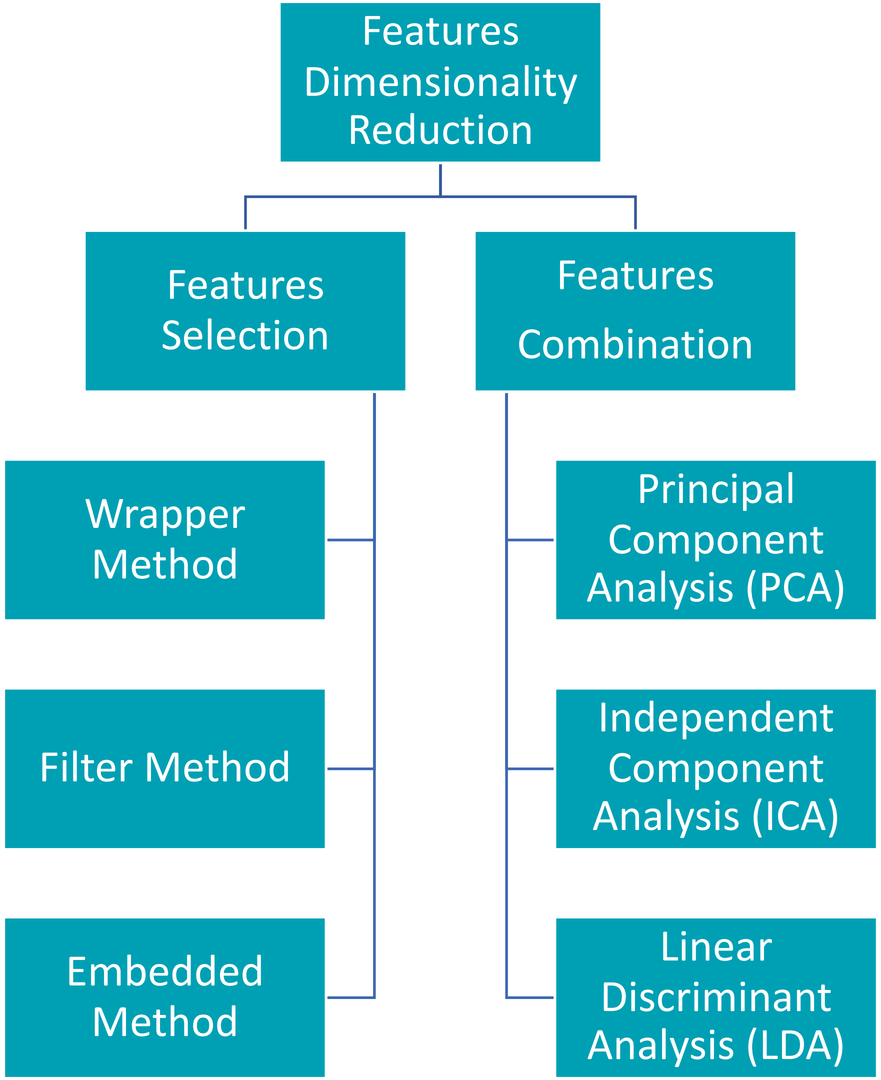
This is the curse of dimensionality.

Training a model with a large number of features is also more time-consuming and may lead to overfitting.

Thus, there is a reduce the number of features for "optimal" model performance.

The Solution

The idea here is:
Choose a few features from many features.

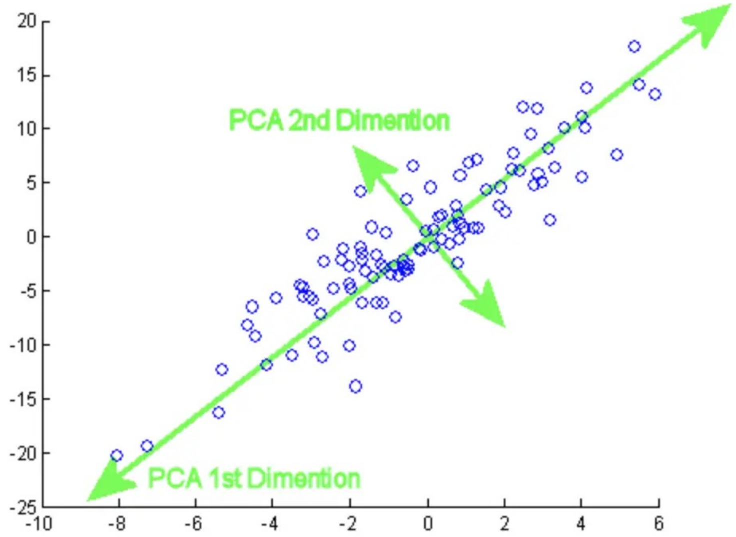


The idea here is:
Combine many features to create a few “superfeatures”.

Principal Component Analysis (PCA)

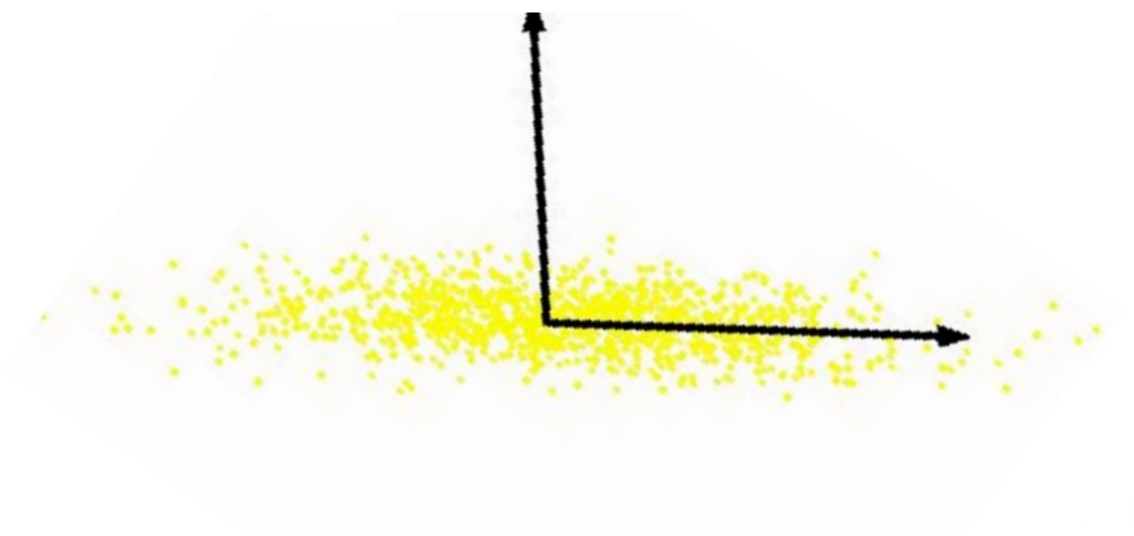
A technique to extract variance structure from high-dimensional data.

PCA is an orthogonal projection of data into a lower-dimensional subspace such that the variance of the projected data is maximized.

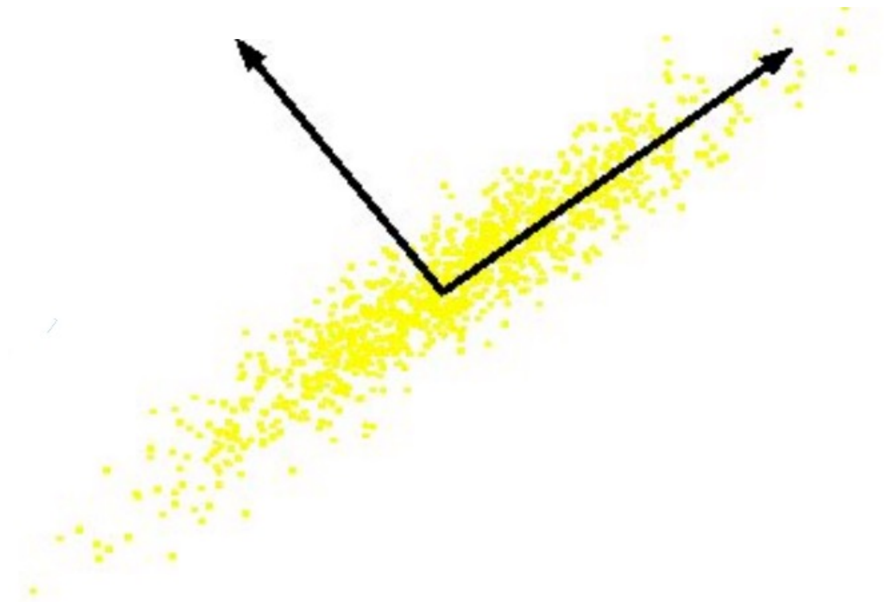


PCA is an unsupervised algorithm!

Principal Component Analysis (PCA)



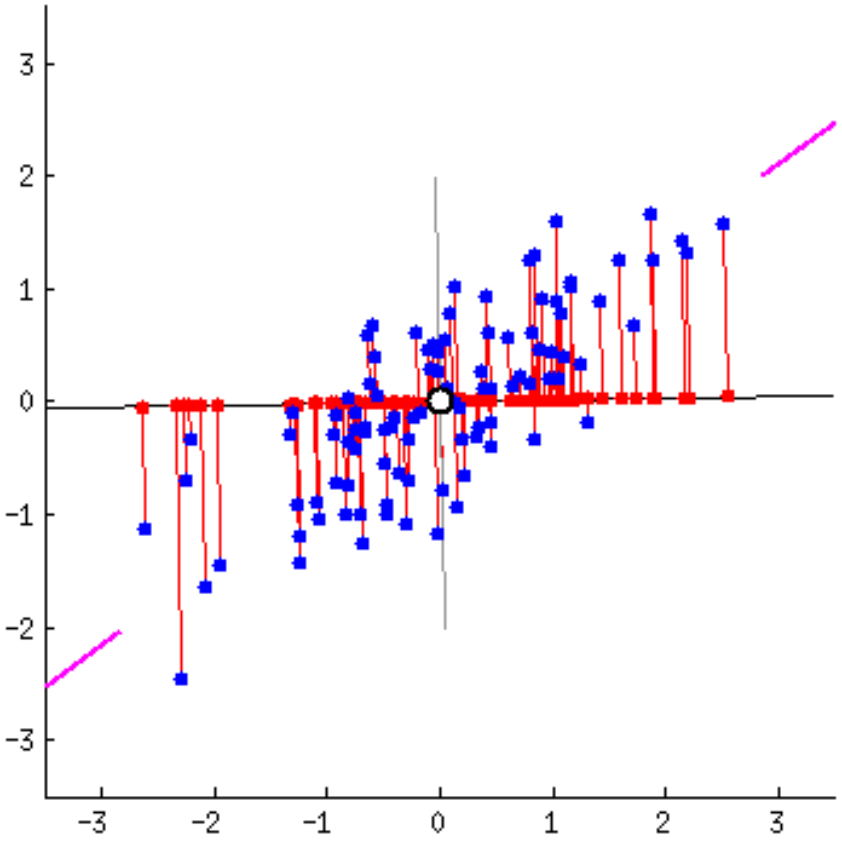
Two features here
Only one is relevant since only one has high variance.



Both features are relevant here.
However, only one coordinate is more relevant since high variance only along one coordinate.

How can we transform the features so that the relevant “feature” along the coordinate is preserved?

How does PCA work?



Principal components are constructed to account for the variance in features.

The first principal component account for the largest possible variance in the dataset.

In this graphic, the first principal component is the line that matches the purple marks.

This is because it is the line that maximizes the variance. The variance is calculated as the average of the squared distances from the projected points (red dots) to the origin.

The second principal component is calculated similarly with the condition that it is orthogonal to the first, and so on!

PCA Algorithm

Step 1: Standardization

$$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation }(x)}$$

#	Emp	Age	Salary
1	Emp1	44	73000
2	Emp2	27	47000
3	Emp3	30	53000
4	Emp4	38	62000
5	Emp5	40	57000
6	Emp6	35	53000
7	Emp7	48	78000
		Mean = 37.42857 Std. Dev. = 6.883876	Mean = 60428.5714 Std. Dev = 10499.7570

Standardization

How to calculate Standardized value?
X = 35, mean = 37.42, Std. Dev. = 6.88
for column Age.
Xstd(for 35) = $\frac{35-37.42}{6.88}$ = −0.3527

Age	Standardized Age	Salary	Standardized Salary
44	0.954611636	73000	1.197306616
27	-1.514927162	47000	-1.278941158
30	-1.079126198	53000	-0.707499364
38	0.083009708	62000	0.149663327
40	0.373543684	57000	-0.326538168
35	-0.352791257	53000	-0.707499364
48	1.535679589	78000	1.673508111
	Mean = 0 Std. dev. = 1		Mean = 0 Std. dev. = 1

Features are scaled such that mean = 0, std = 1.

Good for Gaussian data.

More robust to outliers.

PCA works under the assumption that the data is Gaussian, thus standardization is important so that large differences between range of variables could be minimized.

PCA Algorithm

Step 2: Compute the Covariance matrix

$$Cov\,(x,y) = \frac{\sum_i^n (x_i - \overline{x}) \cdot (y_i - \overline{y})}{N - 1}$$

Sample Covariance for two independent variables (features) x & y

Covariance is positive if both variables (features) increase.

Covariance is negative if when one variable increases, the other decreases.

Covariance is zero, when there is no direct relation.

PCA Algorithm

Step 3: Compute Eigenvalues and Eigenvectors of the Covariance Matrix

For a nxn square matrix A and a non-zero vector X

$$AX = \lambda X$$

for some scalar values λ, known as the Eigenvalues of matrix A.

$$AX - \lambda X = 0$$

$$(A - \lambda I)X = 0$$

where I is an nxn identity matrix. We can solve for

$$|A - \lambda I| = 0$$

to get the Eigenvalues λ, and get corresponding eigenvectors then by solving

$$AX = \lambda X$$

PCA Algorithm

Step 4: Sort the Eigenvalues and corresponding Eigenvectors in descending order

We sort the eigenvalues λ_1, λ_2, λ_3, …, λ_N from largest to smallest and sort the eigenvectors accordingly.

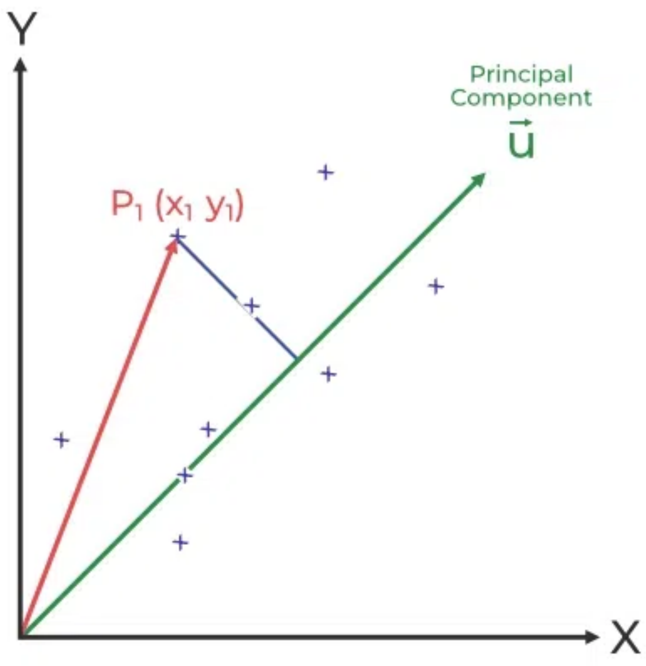
Eigenvectors represent the direction in which the data varies the most, and eigenvalues represent the amount of variation along the corresponding direction.

Since these eigenvalues represent the variance in the data, we can pick P eigenvalues from N that explain most variance.

Generally, we taken P to be as many eigenvalues as explain 95% variance in the data. This P then becomes the P number of principal components.

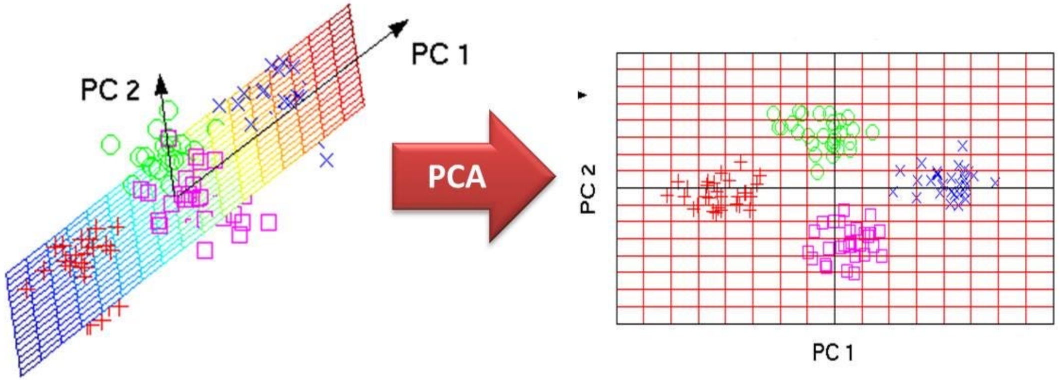
PCA Algorithm

Step 5: Project the data onto the P selected Principal Components

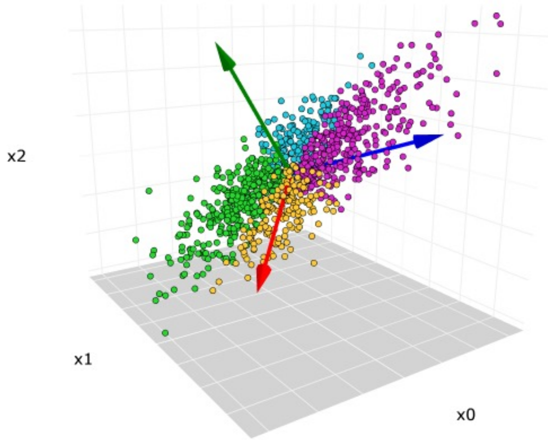


$$\mathrm{Proj}_{P_1}\vec{u} = \frac{P_1 \cdot \vec{u}}{|u|}$$
$$= P_1 \cdot \vec{u} \quad \text{.....}\vec{u} \rightarrow \text{Unit Vector}$$

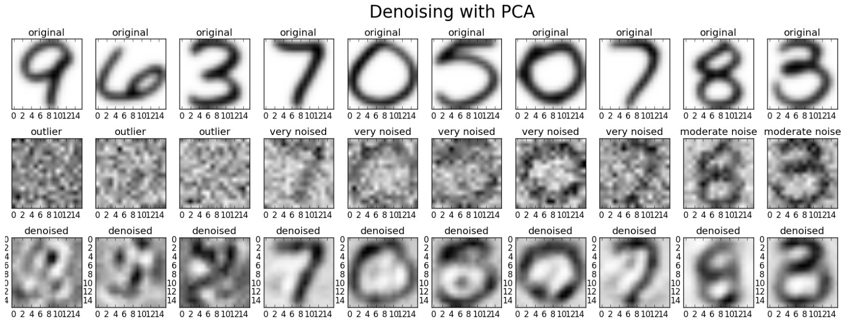
PCA Applications



Dimensionality Reduction



Data Visualization



Denoising (if noise is distributed along PCs with low variance)

Data Compression

PCA Disadvantages

- Interpretation of PCs: It's often difficult to interpret PCs as they are a combination of the original features. Hard to know which features contributed how much for PCs.

- Information Loss: Some information will always be lost when we will choose a few PCs from all. Thus, important to know how many to choose.

- Linear assumption: PCA works by assuming that the features have a linear relationship between them. If it is not linear, PCA may now work well.

- Computational Complexity: This will come into play if the number of variables is very large.

PCA Example

Let's say we have a small dataset of five datapoints with four features.

f1	f2	f3	f4
1	2	3	4
5	5	6	7
1	4	2	3
5	3	2	1
8	1	2	2

We want to go from this four-dimensional dataset to a lower dimensional one.

PCA Example

Step 1: Standardization

	f1	f2	f3	f4
μ =	4	3	3	3.4
σ =	3	1.58114	1.73205	2.30217

f1	f2	f3	f4
-1	-0.63246	0	0.26062
0.33333	1.26491	1.73205	1.56374
-1	0.63246	-0.57735	-0.17375
0.33333	0	-0.57735	-1.04249
1.33333	-1.26491	-0.57735	-0.60812

PCA Example

Step 2: Compute the covariance matrix

	f1	f2	f3	f4
f1	var(f1)	cov(f1,f2)	cov(f1,f3)	cov(f1,f4)
f2	cov(f2,f1)	var(f2)	cov(f2,f3)	cov(f2,f4)
f3	cov(f3,f1)	cov(f3,f2)	var(f3)	cov(f3,f4)
f4	cov(f4,f1)	cov(f4,f2)	cov(f4,f3)	var(f4)

$$Cov\,(x,y) = \frac{\sum_i^n (x_i - \overline{x}) \cdot (y_i - \overline{y})}{N-1}$$

	f1	f2	f3	f4
f1	0.8	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8	0.51121	0.4945
f3	0.03849	0.51121	0.8	0.75236
f4	-0.14479	0.4945	0.75236	0.8

PCA Example

Step 3: Compute Eigenvalues and Eigenvectors of the Covariance Matrix

$$|A - \lambda I| = 0$$

	f1	f2	f3	f4
f1	0.8 - λ	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8- λ	0.51121	0.4945
f3	0.03849	0.51121	0.8 - λ	0.75236
f4	-0.14479	0.4945	0.75236	0.8 - λ

λ = 2.51579324 , 1.0652885 , 0.39388704 , 0.02503121

PCA Example

Step 3: Compute Eigenvalues and Eigenvectors of the Covariance Matrix

$$(A - \lambda I)X = 0$$

$$\begin{pmatrix} 0.800000-\lambda & -(0.252982) & 0.038490 & -(0.144791) \\ -(0.252982) & 0.800000-\lambda & 0.511208 & 0.494498 \\ 0.038490 & 0.511208 & 0.800000-\lambda & 0.752355 \\ -(0.144791) & 0.494498 & 0.752355 & 0.800000-\lambda \end{pmatrix} \times \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = 0$$

e1	e2	e3	e4
0.161960	-0.917059	-0.307071	0.196162
-0.524048	0.206922	-0.817319	0.120610
-0.585896	-0.320539	0.188250	-0.720099
-0.596547	-0.115935	0.449733	0.654547

These are our eigenvectors.

PCA Example

Step 4: Sort the Eigenvalues and corresponding Eigenvectors in descending order

$\lambda = 2.51579324 , 1.0652885$

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

We choose two eigenvalues and the corresponding two eigenvectors.

PCA Example

Step 5: Project the data onto the P selected Principal Components

f1	f2	f3	f4
-1.000000	-0.632456	0.000000	0.260623
0.333333	1.264911	1.732051	1.563740
-1.000000	0.632456	-0.577350	-0.173749
0.333333	0.000000	-0.577350	-1.042493
1.333333	-1.264911	-0.577350	-0.608121

(5,4)

*

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

(4,2)

=

nf1	nf2
0.014003	0.755975
-2.556534	-0.780432
-0.051480	1.253135
1.014150	0.000239
1.579861	-1.228917

(5,2)

PCA allowed us to reduce the dimensionality of the data from four to two!

Independent Component Analysis (ICA)

ICA can find new representation to transform data just like PCA.

It can be used for dimensionality reduction like PCA but instead of finding the components that explain maximum variance, it finds the independent components making up the data. Generally, it is used for separating the multivariate signals into components that are maximally independent.

ICA assumes that the subcomponents of the features are non-Gaussian and are statistically independent.

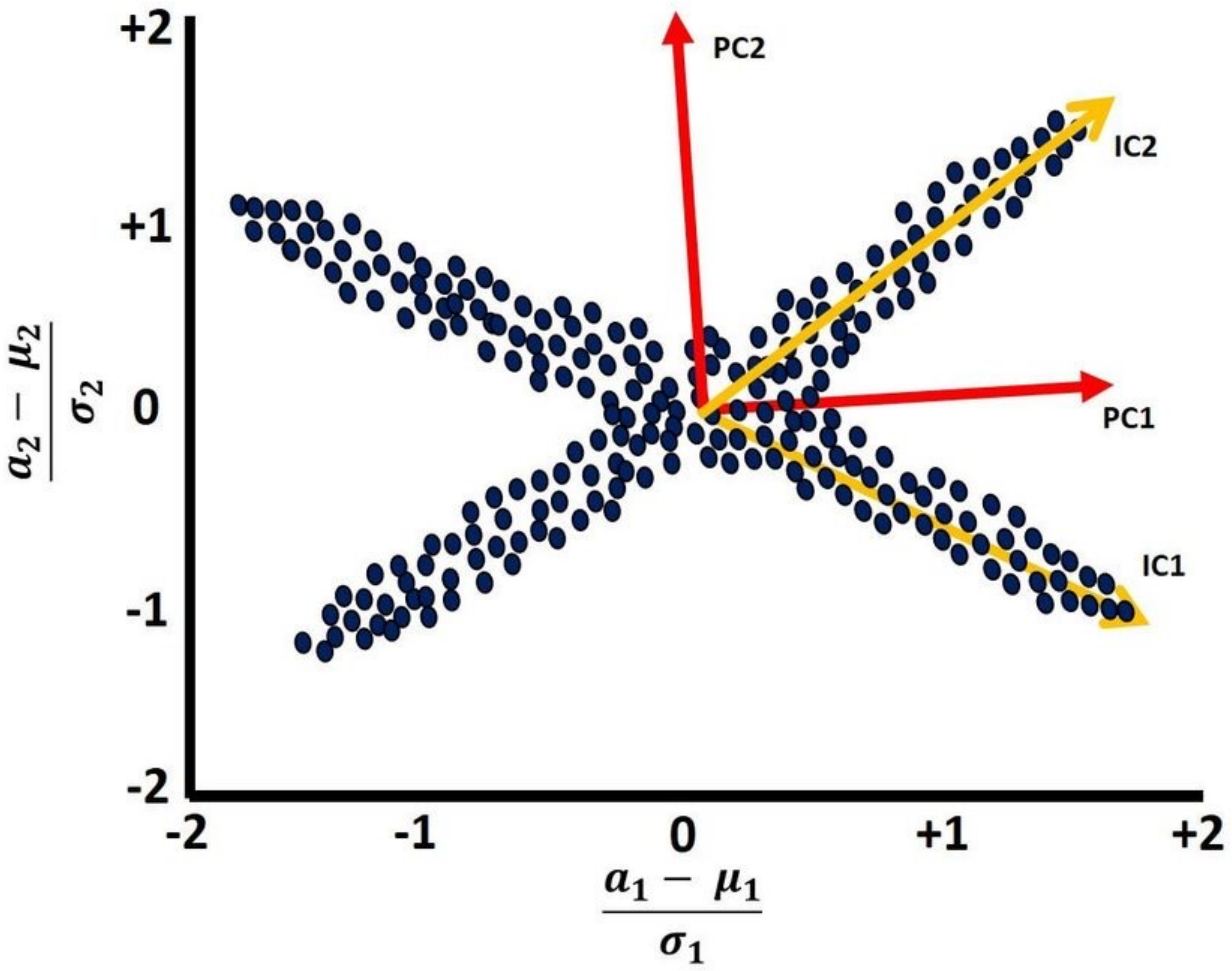
ICA is also an unsupervised method!

PCA vs. ICA

- PCA
  - Focuses on uncorrelated and Gaussian components
  - Second-order statistics (variance)
  - Orthogonal transformation

- ICA
  - Focuses on independent and non-Gaussian components
  - Higher-order statistics (kurtosis)
  - Non-orthogonal transformation

PCA vs. ICA



In PCA, we find the principal components that explain the maximum variance in the data. The PCs are orthogonal to each other.

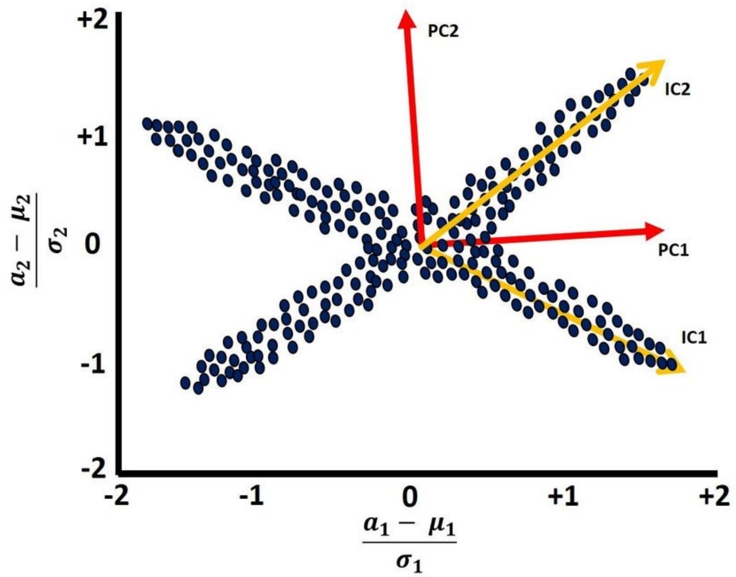
In ICA, we find the independent components that form the data. They are non-orthogonal.

What are independent components?
If one component cannot be estimated from the other component, then it is independent.

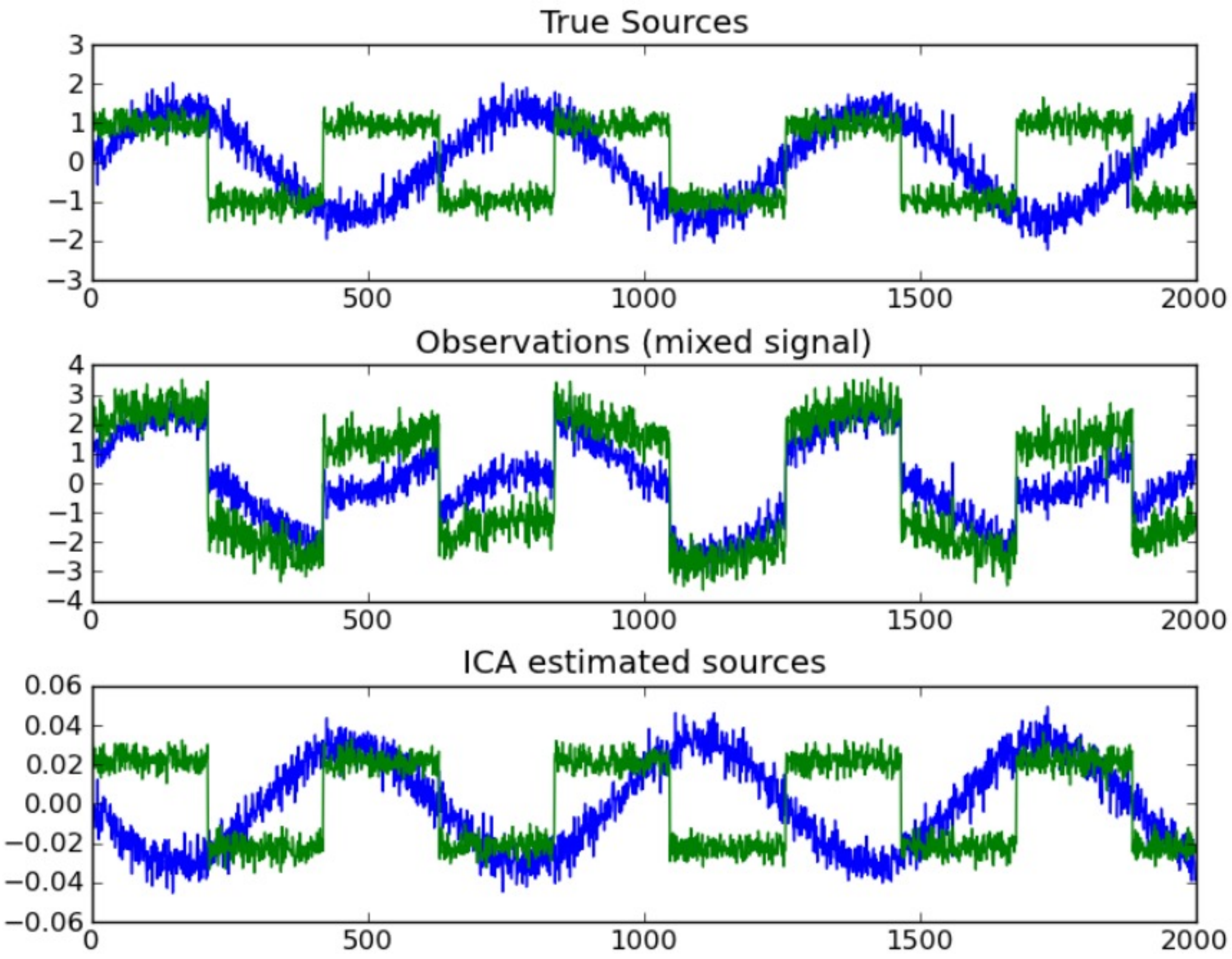
ICA Assumption

By Central Limit Theorem, a sum of two independent random variables is more Gaussian than original variables.

Thus, the distribution of independent components are non-Gaussian.



Blind Source Separation



Blind Source Separation

Consider some signal (data) s that is generated by n independent sources

$$x = As$$

where A is an unknown matrix (called mixing matrix) and x is the received signal

We have repeated observations of x in our dataset $\{\mathrm{x}^{(\mathrm{i})}, i = 1, 2, 3, \ldots, m\}$

Our goal is to recover the original signal $s^{(i)}$

ICA Problem Statement

- We have no prior knowledge of the sources or the mixing matrix A
- Permutation of the sources is ambiguous
- We assume that the original signals are non-Gaussian, how can we recover the n independent sources forming our data s?

ICA Algorithm

Suppose the distribution of each sources s_i is given by a density p_s.

The joint distribution of the sources s_i is given by

$$p(s) = \prod_{i=1}^{n} p_s(s_i).$$

Now, $x = As = W^{-1}s$

$$p(x) = \prod_{i=1}^{n} p_s(w_i^T x) \cdot |W|.$$

The assumption here is that the sources are independent because only then the joint distribution is the product of the individual sources.

We need to find W^{-1} which is an approximation of A i.e., the original mixing matrix.

ICA Algorithm

- We need to specify a cdf which increases from 0 to 1.
- Sigmoid function is a good candidate

$$g(s) = \frac{1}{(1 + e^{-s})}.$$

- This yields, $p_s(s) = g'(s)$

- Given a training set $\{x^{(i)}, i = 1, 2, 3, \ldots, m\}$, the log likelihood for our parameter matrix W is

$$\ell(W) = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \log g'(w_j^T x^{(i)}) + \log |W| \right)$$

ICA Algorithm

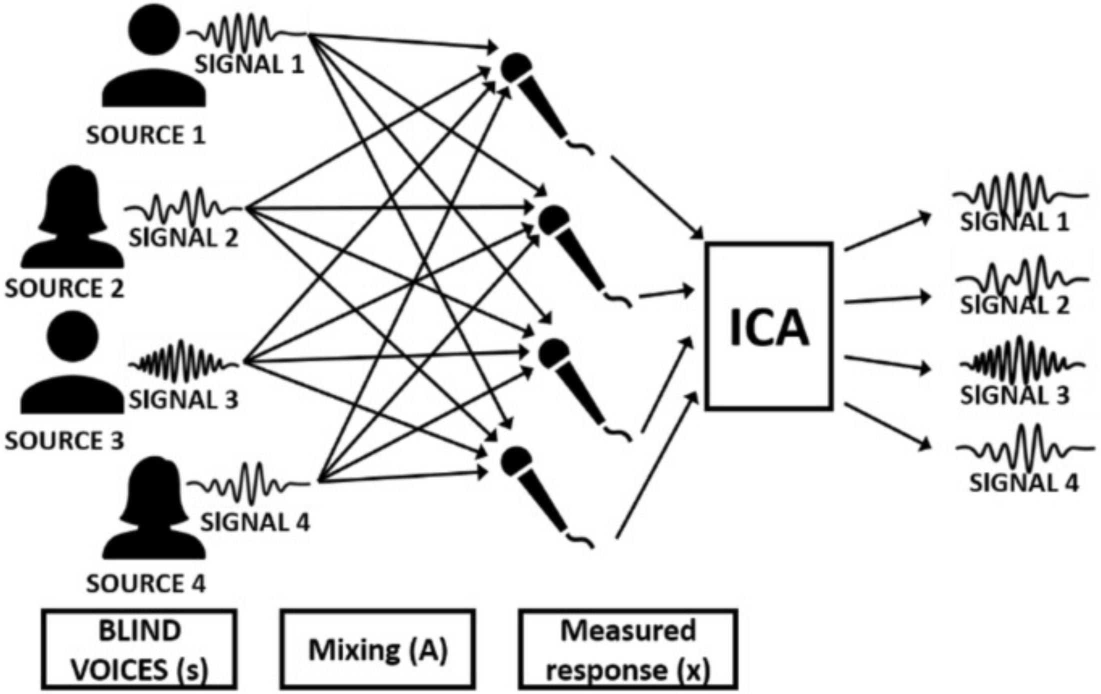
- Maximizing this in terms of *W*, we derive a stochastic gradient ascent learning rule for training example $x^{(i)}$

$$W := W + \alpha \left(\begin{bmatrix} 1 - 2g(w_1^T x^{(i)}) \\ 1 - 2g(w_2^T x^{(i)}) \\ \vdots \\ 1 - 2g(w_n^T x^{(i)}) \end{bmatrix} x^{(i)T} + (W^T)^{-1} \right)$$

where α is the learning rate.

- After the algorithm converges, we get the matrix *W* to estimate the mixing matrix *A*.

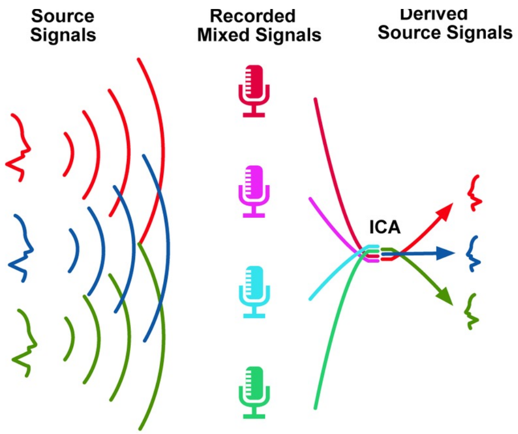
ICA and the Cocktail Party Problem



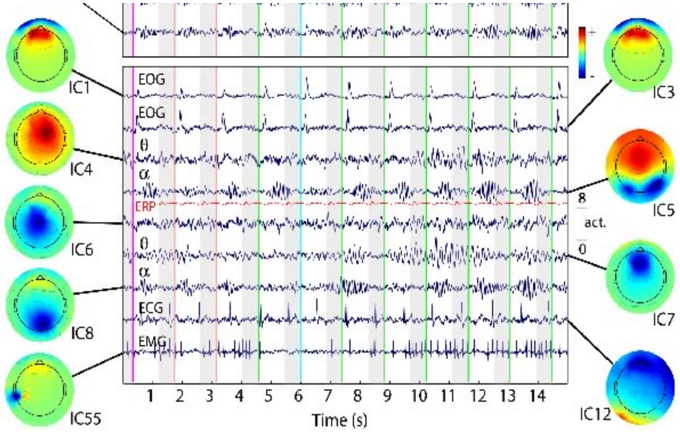
ICA in action

Courtesy:
Dr. Tzyy-Ping Jung
UC San Diego

ICA Applications



Speech source separation



EEG noise removal (remove contamination due to sources such as eye blinks, heart activity, line noise, etc.

Finding hidden factors in Financial Data

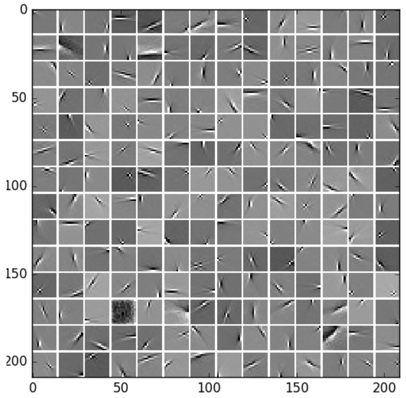


Image Filtering

Summary

This lecture

- PCA algorithm and its applications

- ICA and its applications

Next lecture

- LDA and its applications

- Differences between PCA and LDA, and how LDA could be used for classification.

Questions?