

Module 6

Dimensionality Reduction

Problem with many input variables

The number of input variables or features for a dataset is referred to as its **dimensionality**.

In machine learning classification problems, there are often too many factors on the basis of which the final classification is done.

These factors are basically variables called features. The **higher** the number of **features**, the **harder** it gets to **visualize the training set and then work on it**.

Sometimes, most of these **features** are **correlated**, and hence **redundant**. This is where dimensionality reduction algorithms come into play.

What is Dimensionality Reduction?

More input features often make a predictive modeling task more challenging to model, more generally referred to as the **curse of dimensionality**.

A feature set could be a dataset with a hundred columns (i.e features) or it could be an array of points that make up a large sphere in the three-dimensional space.

Dimensionality reduction is bringing the number of columns down to say, twenty or converting the sphere to a circle in the two-dimensional space.

Dimensionality Reduction

Dimensionality reduction is simply, the process of reducing the dimension of your feature set.

Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables.

It is commonly used in the fields that deal with high-dimensional data, such as **speech recognition, signal processing, bioinformatics, etc.** **It can also be used for data visualization, noise reduction, cluster analysis, etc.**

Dimensionality Reduction

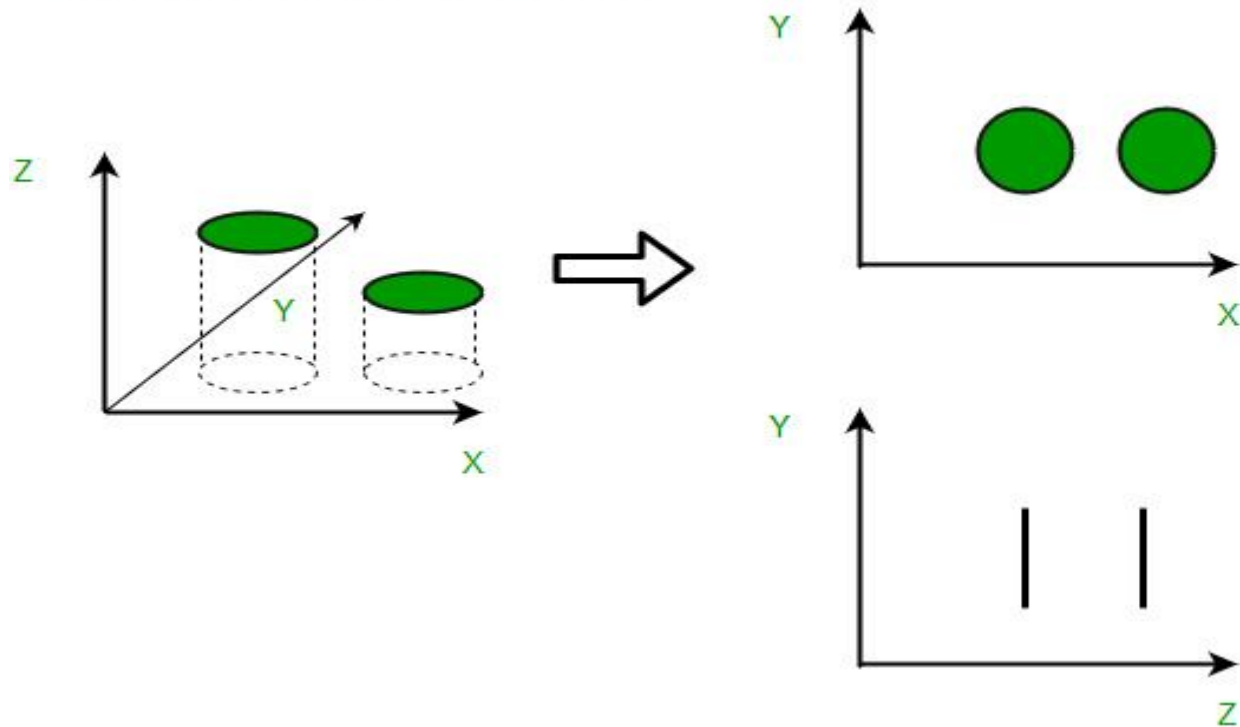
Example:

Consider e-mail classification problem, where we need to classify whether the e-mail is spam or not. This can involve a large number of features, such as whether or not the e-mail has a generic title, the content of the e-mail, whether the e-mail uses a template, etc. However, some of these features may overlap.

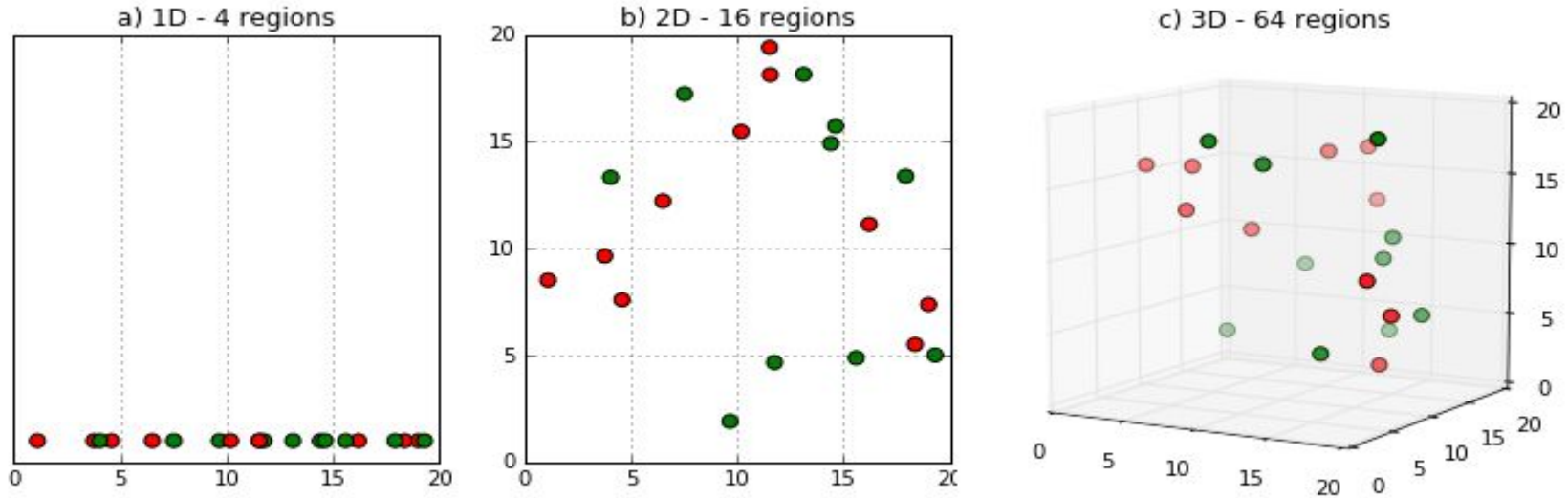
A 3-D classification problem can be hard to visualize, whereas a 2-D one can be mapped to a simple 2 dimensional space, and a 1-D problem to a simple line. The below figure illustrates this concept, where a 3-D feature space is split into two 1-D feature spaces, and later, if found to be correlated, the number of features can be reduced even further.

Dimensionality Reduction

Dimensionality Reduction



Dimensionality Reduction



One dimension need only 4 spaces for describing any of the points.

In 2-dimensional, the number of spaces increases to 16

And for 3 dimensional, the number of spaces rises to 64.

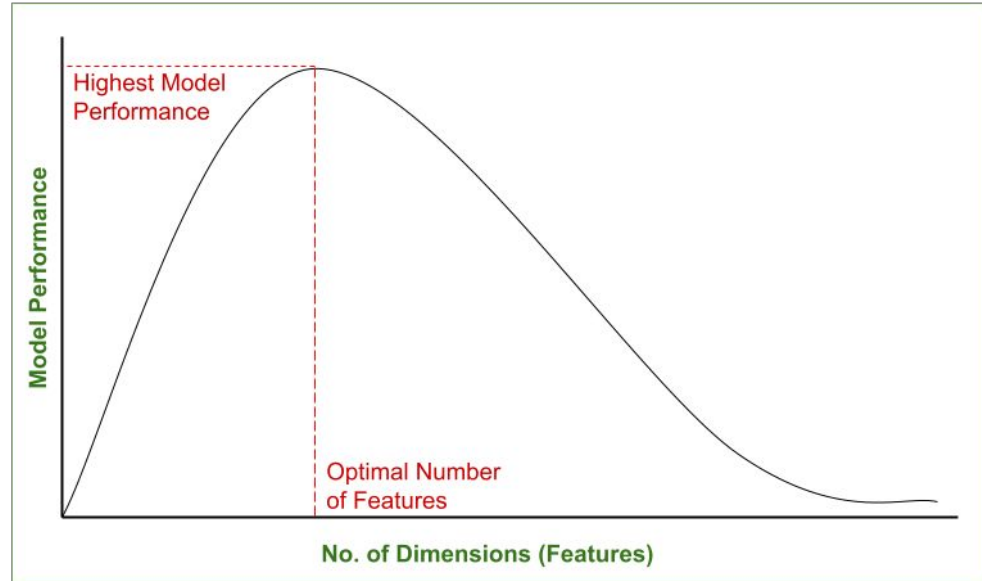
The Curse of Dimensionality

The curse of dimensionality refers to all the problems that arise when working with data in the higher dimensions, that did not exist in the lower dimensions.

As the number of features increase, the number of samples also increases proportionally. The more features we have, the more number of samples we will need to have all combinations of feature values well represented in our sample.

The Curse of Dimensionality

As the number of features increases, the model becomes more complex. The more the number of features, the more the chances of overfitting. A machine learning model that is trained on a large number of features, gets increasingly dependent on the data it was trained on and in turn overfitted, resulting in poor performance on real data, beating the purpose.



Advantages of Dimensionality Reduction

1. Less misleading data means model accuracy improves.
2. Less dimensions mean less computing. Less data means that algorithms train faster.
3. Less data means less storage space required.
4. Less dimensions allow usage of algorithms unfit for a large number of dimensions
5. Removes redundant features and noise.
6. Visualization: projection of high-dimensional data onto 2D or 3D.

Disadvantages of Dimensionality Reduction

- It may lead to some amount of data loss.
- PCA tends to find linear correlations between variables, which is sometimes undesirable.
- PCA fails in cases where mean and covariance are not enough to define datasets.
- We may not know how many principal components to keep- in practice, some thumb rules are applied.

Approaches of Dimension Reduction

There are two ways to apply the dimension reduction technique, which are given below:

1) Feature selection

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high accuracy. In other words, it is a way of selecting the optimal features from the input dataset.

In feature selection, we are interested in finding k of the total of n features that give us the most information and we discard the other $(n-k)$ dimensions. We are going to discuss subset selection as a feature selection method.

Three methods are used for the feature selection:

1. Filters Methods

2. Wrappers Methods

3. Embedded Methods:

Feature selection

Example

Suppose we are building a model that predicts the height of a building.

We have a dataset with various features including the number of windows, number of apartments, color of the building, etc.

In this dimensionality reduction example, the feature of **color** is hardly a deciding factor in the height of a building, and hence we can deselect this feature to simplify our dataset.

2) Feature extraction

- This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions
- In feature extraction, we are interested in finding a new set of k features that are the combination of the original n features. These methods may be supervised or unsupervised depending on whether or not they use the output information.
- A simple way to understand the difference between feature selection and feature extraction is this – feature selection can reduce (x, y, z) to (x, y) ; feature extraction can extract $(2x-3y)$.
- The best known and most widely used feature extraction methods are Principal Components Analysis (PCA) and Linear Discriminant Analysis (LDA), which are both linear projection methods, unsupervised and supervised respectively

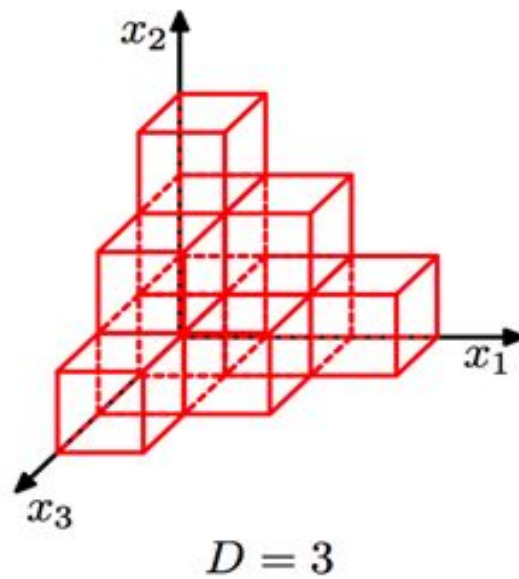
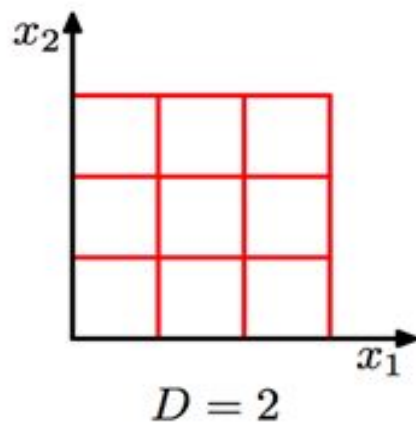
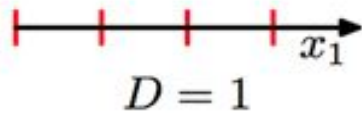
Dimensionality reduction methods

The methods used for dimensionality reduction include:

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)

Principal Component Analysis (PCA)

- Data preprocessing
- feature extraction method
- Principal Component Analysis (PCA) is an unsupervised, non-parametric statistical technique primarily used for dimensionality reduction in machine learning.
- High dimensionality means that the dataset has a large number of features.
- The primary problem associated with high-dimensionality in the machine learning field is model overfitting, which reduces the ability to generalize beyond the examples in the training set.
- Curse of Dimensionality



PCA

- This method was introduced by Karl Pearson. It works on a condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, the variance of the data in the lower dimensional space should be maximum
- The ability to generalize correctly becomes exponentially harder as the dimensionality of the training dataset grows.
- Models also become more efficient as the reduced feature set boosts learning rates and diminishes computation costs by removing redundant features.

PCA steps:

Step 1: Normalize the data

This is done by subtracting the respective means from the numbers in the respective column. So if we have two dimensions X and Y, all X become \bar{x} and all Y become \bar{y} . This produces a dataset whose mean is zero.

Step 2: Calculate the covariance matrix

For 2-dimensional, this will result in a 2x2 Covariance matrix.

$$\text{Matrix}(\text{Covariance}) = \begin{bmatrix} \text{Var}[X_1] & \text{Cov}[X_1, X_2] \\ \text{Cov}[X_2, X_1] & \text{Var}[X_2] \end{bmatrix}$$

Note that $\text{Var}[X_1] = \text{Cov}[X_1, X_1]$ and $\text{Var}[X_2] = \text{Cov}[X_2, X_2]$.

Mean

$$\bar{X} = \frac{\sum_{i=1}^n x_i}{n}$$

PCA steps:

Step 3: Calculate the eigenvalues and eigenvectors

Next step is to calculate the eigenvalues and eigenvectors for the covariance matrix. The same is possible because it is a square matrix. λ is an eigenvalue for a matrix A if it is a solution of the characteristic equation:

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

Where, I is the identity matrix of the same dimension as A which is a required condition for the matrix subtraction as well in this case and '**det**' is the determinant of the matrix. For each eigenvalue λ , a corresponding eigen-vector v , can be found by solving:

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

PCA steps:

Step 4: Choosing components and forming a feature vector:

order the eigenvalues from largest to smallest so that it gives us the components in order of significance. Here comes the dimensionality reduction part. If we have a dataset with n variables, then we have the corresponding n eigenvalues and eigenvectors. It turns out that the eigenvector corresponding to the highest eigenvalue is the principal component of the dataset and it is our call as to how many eigenvalues we choose to proceed our analysis with. To reduce the dimensions, we choose the first p eigenvalues and ignore the rest. We do lose out some information in the process, but if the eigenvalues are small, we do not lose much.

PCA steps:

Step 5: Forming Principal Components

This is the final step where we actually form the principal components using all the math we did till here. For the same, we take the transpose of the feature vector and left-multiply it with the transpose of scaled version of original dataset.

$$NewData = FeatureVector^T \times ScaledData^T$$

Here,

NewData is the Matrix consisting of the principal components,

FeatureVector is the matrix we formed using the eigenvectors we chose to keep, and *ScaledData* is the scaled version of original dataset

PCA Numerical Problem

[PCA Numerical](#)

Independent Component Analysis (ICA)

Independent Component Analysis (ICA) is a machine learning technique to separate independent sources from a mixed signal. It is a method to extract the factors or components from multidimensional statistical data.

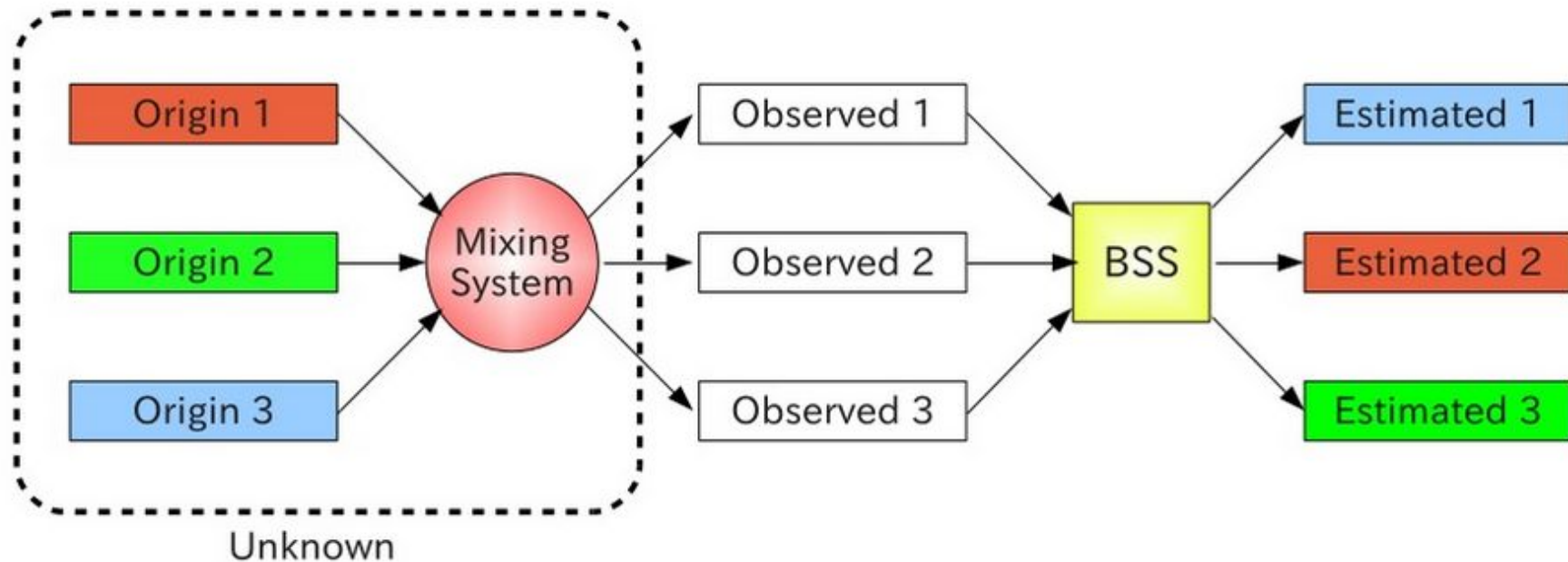
Unlike principal component analysis which focuses on maximizing the variance of the data points, the independent component analysis focuses on independence, i.e. independent components.

ICA is a linear dimension reduction method, which transforms the dataset into columns of independent components.

ICA is used to solve Blind Source Separation or "cocktail party problem"

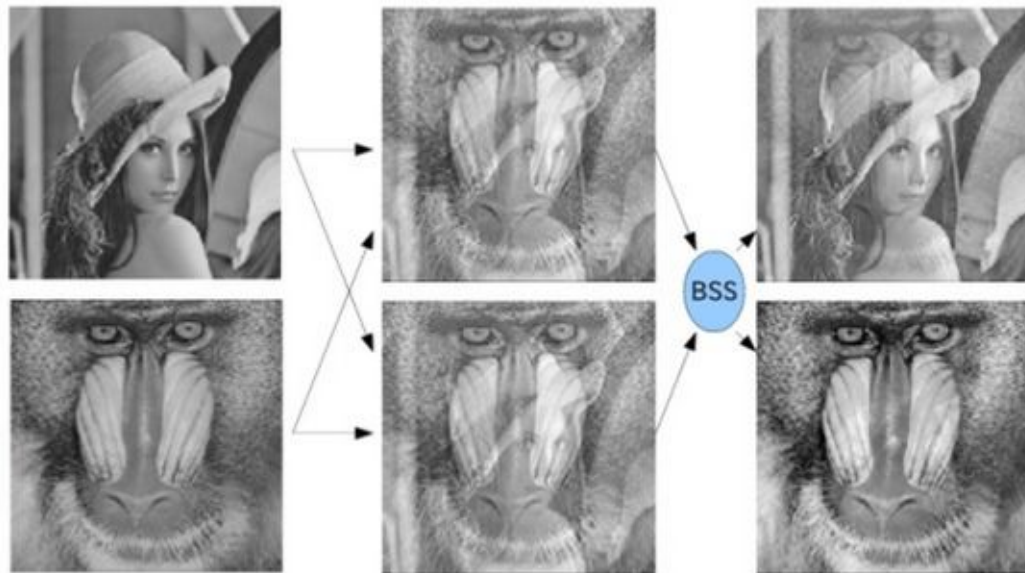
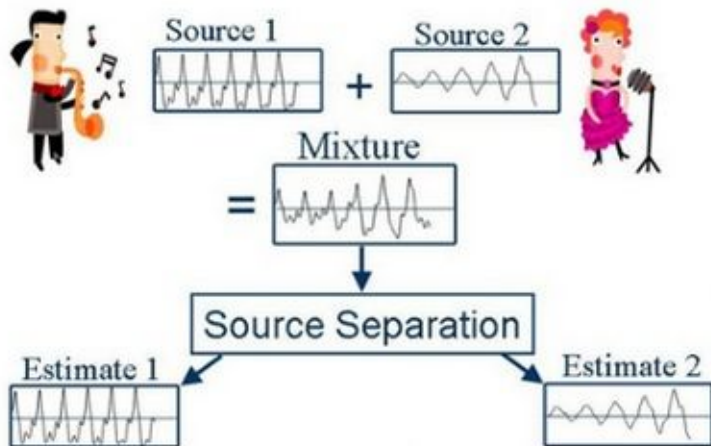
Blind Source Separation

Blind Source Separation is a method to estimate original signals from observed signals which consist of mixed original signals



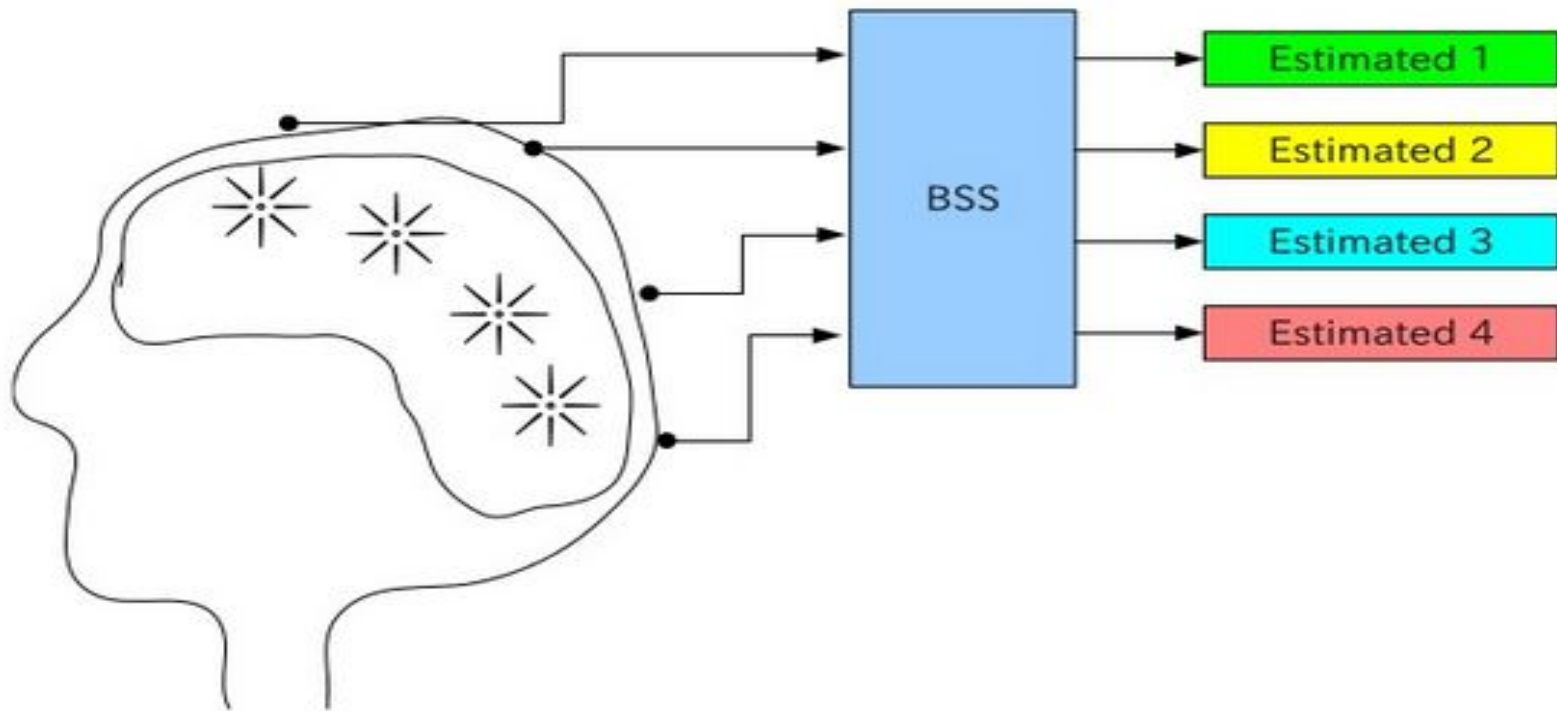
Examples of BSS

BSS is often used for Speech analysis and Image analysis.

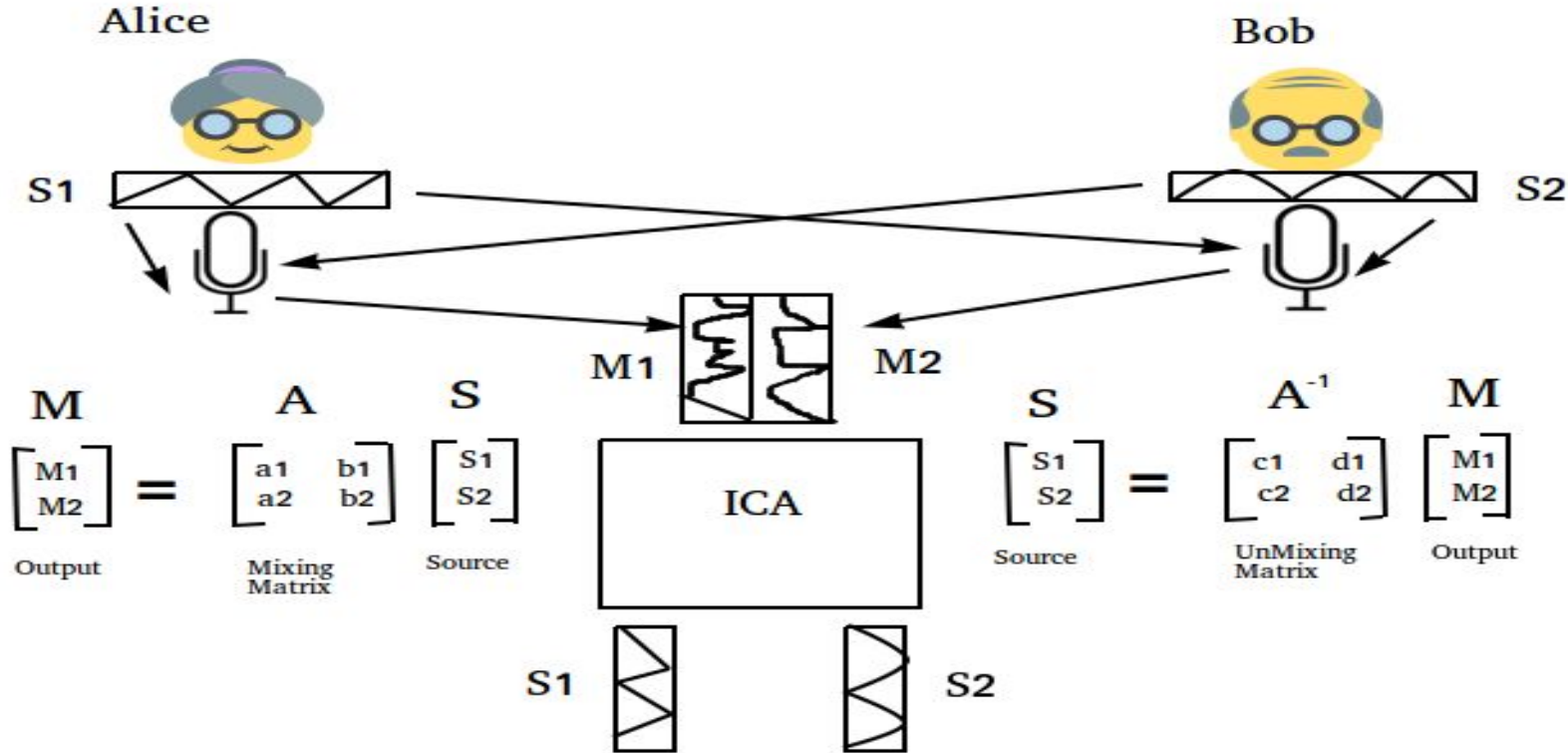


Examples of BSS

BSS is also very important for brain signal analysis.



Blind Source Separation / "cocktail party problem"



Blind Source Separation / "cocktail party problem"

Alice and Bob, both are speaking at the same time.

Both mics receive inputs S_1 & S_2 from Alice and Bob respectively.

ICA assumes that the mixing process is linear i.e. it can be represented as a matrix multiplication.

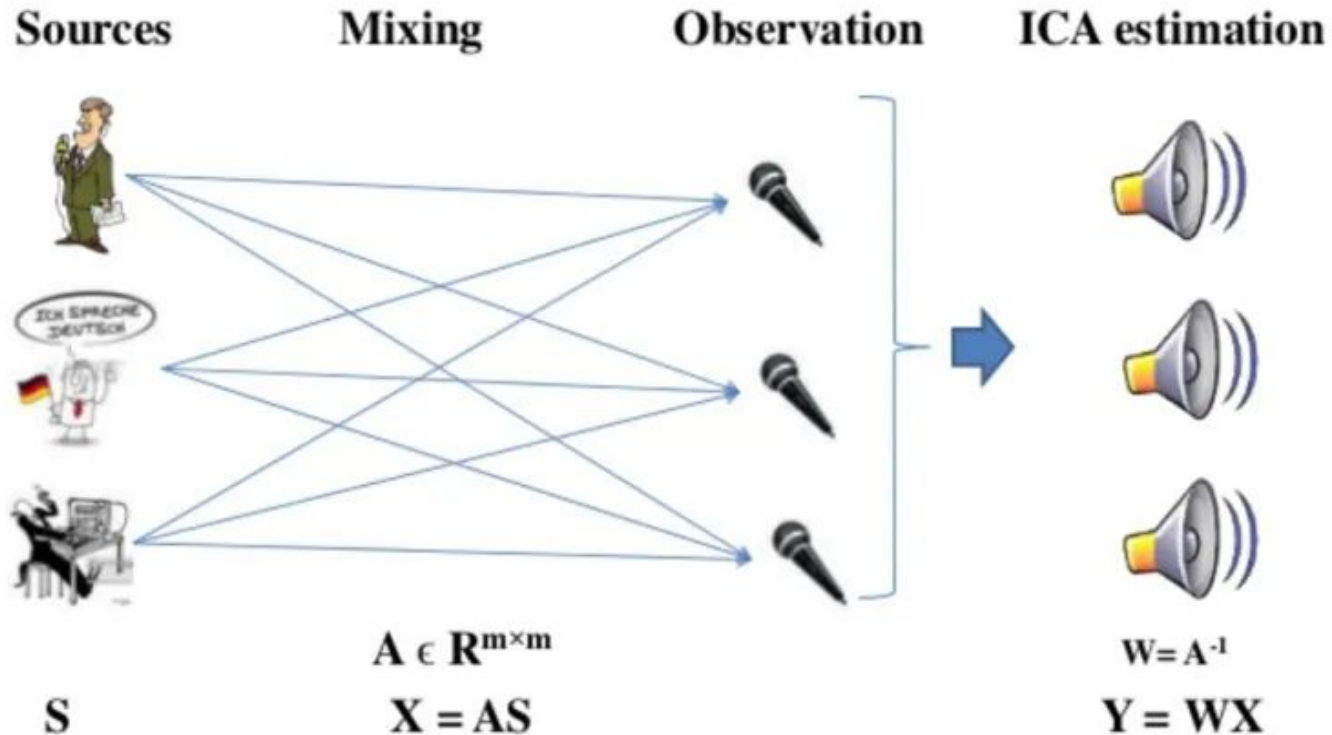
Each mic mixes S_1 & S_2 according to its location and settings which is given by a matrix A .

The matrix operation produces vector M as an output.

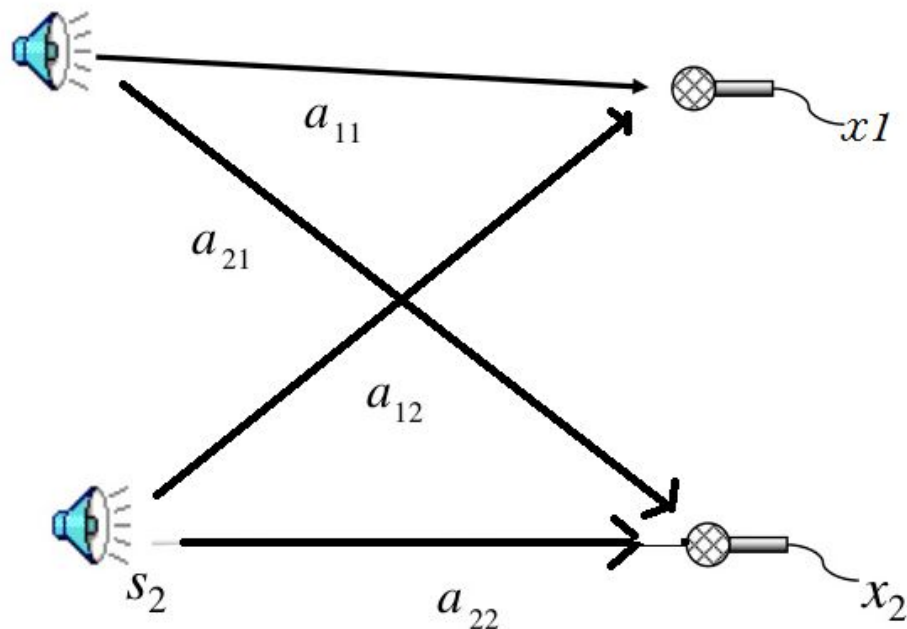
Now, you wish to separate the S_1 & S_2 from M_1 & M_2 .

This is referred to as **cocktail party problem** or **blind source separation**.

Blind Source Separation / "cocktail party problem"



Blind Source Separation / "cocktail party problem"

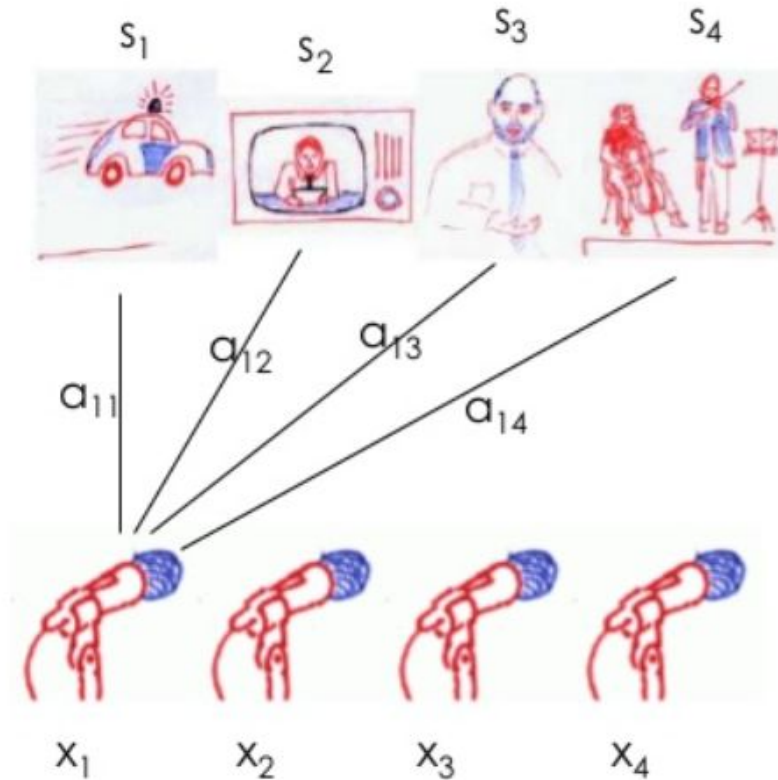


$$\left. \begin{aligned} x_1 &= a_{11}s_1 + a_{12}s_2 \\ x_2 &= a_{21}s_1 + a_{22}s_2 \end{aligned} \right\}$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

Blind Source Separation / "cocktail party problem"



$$x_i(t) = a_{i1}s_1(t) + a_{i2}s_2(t) + a_{i3}s_3(t) + a_{i4}s_4(t)$$

Here, $i=1:4$.

In vector-matrix notation, and dropping index t , this is

$$\mathbf{x} = \mathbf{A} * \mathbf{s}$$

ICA Assumptions

ICA approach to this problem is based on three assumptions. These are:

1. Mixing process is linear.
2. All source signals are independent of each other.
3. All source signals have non-gaussian distribution.

ICA cannot be applied to just any mixed signal. First, the mixed signals must be a **linear combination** of source signals.

Next, ICA assumes that the source signals are **independent**, while the mixed signals are not (as they share the same source signals), and uses this fact for separation.

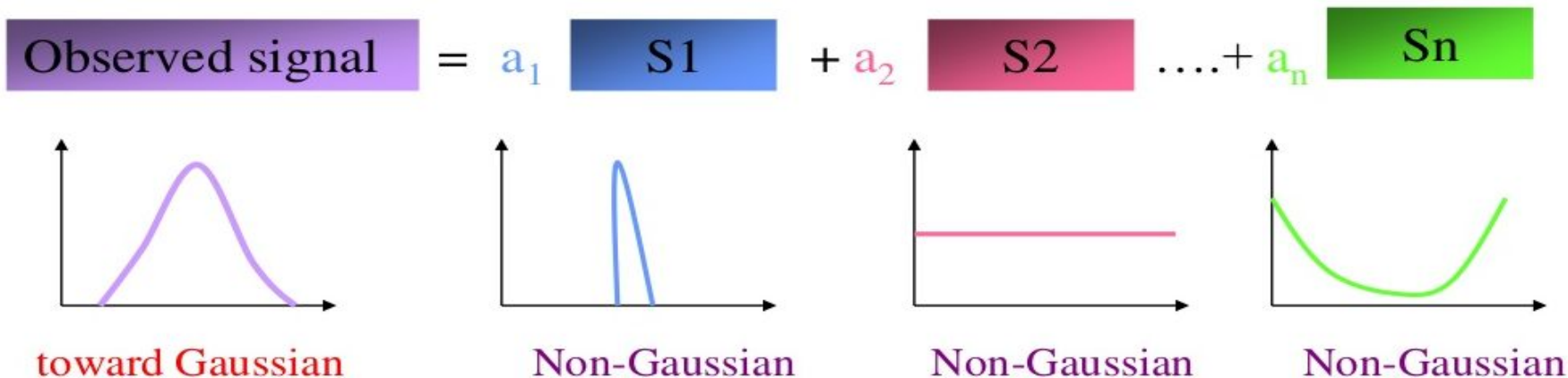
Finally, ICA also assumes the source signals are **non-Gaussian**, and uses the Central Limit Theorem which implies a sum of two signals has a distribution closer to a Gaussian than the two signals on their own.

Non-gaussianity is independent

Central limit theorem

The distribution of a **sum** of independent random variables tends toward a **Gaussian distribution**

According to the Central Limit Theorem, the sum of independent random variables is more Gaussian than the independent variables.



Applications

Image Denoising

Original
image



Noisy
image



Wiener
filtering



ICA
filtering



Applications

Image Denoising

Audio processing

Speech enhancement(noisy speech recognition)

Bio signal processing

Finance

Telecommunication (CDMA)

Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA)

- Most commonly used as a dimensionality reduction technique.
- Goal is to project a dataset onto a lower dimensional space with :-
 - Good Class separability
 - Reduce computational costs.
- Approach is similar to PCA, i.e. in addition to finding the component axes that maximises the variance of the data(PCA) , we are interested in the axes that maximizes the separation between multiple classes (LDA)

Steps used in Linear Discriminant Analysis (LDA)

General steps for performing LDA analysis:-

1. Compute the d-dimensional mean for the different classes from the dataset.
2. Compute the scatter matrices (in between classes and within class scatter matrix).
3. Compute the eigen vectors (e_1, e_2, \dots, e_n) and corresponding eigen values for the scatter matrices.
4. Sort the eigenvectors by decreasing eigen values and choose k eigenvectors with the largest eigen values to form a $d \times k$ dimensional matrix W
5. $d \times k$ eigen vectors matrix to transform the samples onto the new subspace. This can be summarized by matrix multiplication $Y = X \times W$ (where X is a $n \times d$ -dimensional matrix representing the n samples and y are the transformed $n \times k$ -dimensional samples in the new subspace).

LDA Numerical

[LDA Numerical](#)

Numerical

Compute the Linear discriminant projection for the following 2 D dataset

$$X1 = (x1, x2) = ((4,2),(2,4),(2,3),(3,6),(4,4))$$

$$X2 = (x1, x2) = ((9,10),(6,8),(9,5),(8,7),(10,8))$$

Step 1 :- Compute the mean for Class 1 and Class 2

$$M1 = [3 \quad 3.60] \quad M2 = [8.40 \quad 7.60]$$

Numerical

Step 2 :- Compute within class scatter matrix

$$SW = S_1 + S_2$$

Where , S_1 = Covariance matrix of class S1

S_2 = Covariance matrix of class S2

$$\begin{aligned} S_1 &= \sum (X - M_1)(X - M_1)^T \\ S_2 &= \sum (X - M_2)(X - M_2)^T \end{aligned}$$

Numerical

$$S_1 = \begin{bmatrix} 0.80 & -0.40 \\ -0.40 & 2.60 \end{bmatrix} \quad S_2 = \begin{bmatrix} 1.84 & -0.04 \\ -0.04 & 2.64 \end{bmatrix}$$
$$SW = \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 5.28 \end{bmatrix}$$

$$SW = \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 5.28 \end{bmatrix}$$

Numerical

Step 3:- Compute between class scatter matrix

$$\begin{aligned} SB &= (M_1 - M_2)(M_1 - M_2)^T \\ SB &= \begin{bmatrix} 29.16 & 21.60 \\ 21.60 & 16.00 \end{bmatrix} \end{aligned}$$

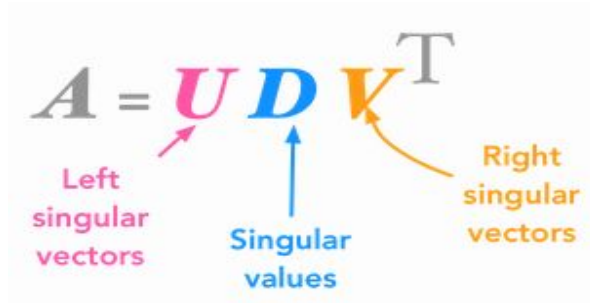
Step 4: - Find best LDA projection vector

$$w^* = SW^{-1}(M_1 - M_2) = [-0.91 \quad -0.39]^T$$

Singular Valued Decomposition (SVD)

Single Valued Decomposition (SVD)

In this method, a matrix is decomposed into three other matrices :-



The diagram illustrates the SVD decomposition equation $A = U D V^T$. The matrix A is shown in grey. The matrix U is pink, with a pink arrow pointing to it from the label "Left singular vectors" below. The matrix D is blue, with a blue arrow pointing to it from the label "Singular values" below. The matrix V is orange, with an orange arrow pointing to it from the label "Right singular vectors" below. The superscript T is grey.

Where, A is $m \times n$ matrix

Single Valued Decomposition (SVD)

Matrix U has left singular vectors as columns

D is a diagonal matrix which contains singular values

V^T has right singular vectors as rows.

In SVD, original data present in coordinate system is expanded. And the covariance matrix is diagonal.

Calculation of SVD

1. Find the eigenvalues and eigenvectors of AA^T and $A^T A$
2. Column of V consists of eigenvectors of A^T
3. Column of U consists of eigenvectors of AA^T
4. The square roots of eigenvalues from AA^T or $A^T A$ represents the singular values in S .
5. The singular values are arranged in descending order and stored as the diagonal entries of S matrix. The singular values are always real numbers.
6. If the matrix A is a real matrix, then U and V are also real .

SVD Numerical

SVD Numerical

SVD Numerical

Find SVD for $A =$

$$\begin{bmatrix} 2 & 2 \\ -1 & 1 \end{bmatrix}$$

Step 1 :- Calculate $A^T A$

$$= \begin{bmatrix} 5 & 3 \\ 3 & 5 \end{bmatrix}$$

SVD Numerical

Step 2: To calculate Eigen vectors V_1, V_2

$$V_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$
$$V_2 = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

Step 3:

To calculate AV_1 and AV_2

$$AV_1 = \begin{bmatrix} 2\sqrt{2} \\ 0 \end{bmatrix}$$
$$AV_2 = \begin{bmatrix} 0 \\ \sqrt{2} \end{bmatrix}$$

SVD Numerical

Step 4: Calculate U1 and U2

$$U_1 = \frac{AV_1}{|AV_1|} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$U_2 = \frac{AV_2}{|AV_2|} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Step 5 : SVD representation

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

SVD

SVD numerical taken from the link :-

<https://medium.com/intuition/singular-value-decomposition-svd-working-example-c2b6135673b5>

Feature selection

1. Filters Methods: which do not involve any learning while reducing variables.

In this method, the dataset is filtered, and a subset that contains only the relevant features is taken. Some common techniques of filters method are:

- **Correlation**
- **Chi-Square Test**
- **ANOVA**
- **Information Gain, etc.**

Feature selection

Wrappers Methods: which involve some learning while reducing variables.

The wrapper method has the same goal as the filter method, but it takes a machine learning model for its evaluation. In this method, some features are fed to the ML model, and evaluate the performance. The performance decides whether to add those features or remove to increase the accuracy of the model. This method is more accurate than the filtering method but complex to work. Some common techniques of wrapper methods are:

- Forward Selection
- Backward Selection
- Bi-directional Elimination

Feature selection

Embedded Methods: which combine feature selection and classifier establishing

Embedded methods check the different training iterations of the machine learning model and evaluate the importance of each feature. Some common techniques of Embedded methods are:

- **LASSO**
- **Elastic Net**
- **Ridge Regression, etc.**

Dimensionality Reduction Techniques

1. Missing Values

If there are data columns with a considerably large number of missing values, the information in the column may not turn out to be of much use.

Such columns are better removed. The more the missing values, the more is the reduction.

2. Low Variance

Data columns with negligible differences in values are not much useful too. Such low variance columns are better eliminated.

Let's think of a scenario where we have a constant variable (all observations have the same value, 5) in our data set. Do you think, it can improve the power of model? Of course NOT, because it has zero variance.

Dimensionality Reduction Techniques

3. Backward Feature Elimination

In this case, we train the model on n input features, then $n-1$ for the next iteration and so on for a total of n times.

The feature that causes the smallest increase in error rate is removed. This leads us with $n-1$, $n-2$ and so on a lesser number of features while not compromising on accuracy.

4. Forward Feature Construction

This is the opposite of backward feature elimination where we go on adding one feature at a time, one that produces maximum improvement in accuracy.

Among all the dimensionality reduction techniques, forward feature construction is one of the costliest.

Dimensionality Reduction Techniques

5.High Correlation

If two or more variables share fairly similar information, they are said to be highly correlated. In such a case, similar or almost duplicate looking data variables can be removed.

Dimensions exhibiting higher correlation can lower down the performance of a model. Moreover, it is not good to have multiple variables of similar information. You can use Pearson correlation matrix to identify the variables with high correlation. And select one of them using VIF (Variance Inflation Factor). Variables having a higher value ($VIF > 5$) can be dropped.