16강: Independent Components Analysis

Part 12

Independent Components Analysis

다음 토픽은 Independent components analysis(ICA)로 즉 독립 성분 분석이다. PCA와 비슷하게 데이터를 나타내는 새로운 basis를 찾지만 목표에서는 분명한 차이가 존재한다.

(1) motivation

독립 성분 분석에는 칵테일 파티를 예로 들 수 있다. n명의 speaker가 파티에서 말을 하고 있다고 할때 그곳에 있는 microphone으로 n명의 speaker의 겹쳐진 목소리를 녹음하고 있다. n개의 microphone은 speaker와 각각 거리가 다르기 때문에 speaker의 목소리가 혼합된 형태가 각각 다를 것이다.

이렇게 녹음된 micophone을 가지고 n명의 speaker의 speech signal을 어떻게 구분할 수 있을까?

설명을 쉽게 하기 위해서 2명의 speaker가 동시에 말하고 2개의 마이크로 녹음을 하고 있는 상황으로 축소해보자. 그러면 두개의 마이크에 녹음된 신호는 각각 $x_1^{(t)}, x_2^{(t)}$으로, 두사람의 음성 신호는 $s_1(t), s_2(t)$으로 나타낼 수 있다.

이때 녹음된 신호와 음성 신호의 관계는 다음과 같이 선형 방정식으로 표현할 수 있다.

$x_1(t) = a_{11}s_1(t) + a_{12}s_2(t)$

$x_2(t) = a_{21}s_1(t) + a_{22}s_2(t)$

여기서 a는 마이크로 부터 speaker까지의 거리와 관련된 변수라고 한다.

따라서 우리가 원하는 것은 녹음된 신호 $x^{(i)}$[결과]으로 부터 원래 음성 신호 $s^{(i)}$[원본]를 분리해 내는 것이다.

(2) notation

위에서 본 녹음된 신호와 원래 음성신호와의 관계를 행렬로 나타내면 다음과 같다.

◗

A : mixing matrix로 unknown square matrix이다.

또한

$x \in R^n$, $A \in R^{n \times n}$, $s \in R^n$

$x_j^{(i)}$: i 시간에 j번째 마이크로 녹음된 신호

만약 우리가 A를 안다면 아래와 같이 행렬식에 A^{-1}를 곱해서 쉽게 s를 찾을 수 있을 것이다.

$W = A^{-1}$: unmixing matrix

따라서 우리의 목적은 source s[원본]를 구하기 위해 W 행렬이 찾는 것이다.

notation 편의를 위해 w_j^T는 W의 j번째 row로 다음과 같이 W를 나타낼 수 있다.

$$W = \begin{bmatrix} - & w_1^T & - \\ - & w_2^T & - \\ & \vdots & \\ - & w_n^T & - \end{bmatrix}$$

$w_i \in R^n$이고

복원된 j번째 source 는 $s_j^{(i)} = w_j^T x^{(i)}$를 통해 구할 수 있는 것이다

> <notation 정리>
>
> sources :
> $s \in R^n$
>
> $s_j^{(i)}$: speaker j at time i
>
> $x^{(i)} = As^{(i)}, \quad x \in R^n$
>
> Goal : Find
> $W = A^{-1}$
>
> $s^{(i)} = wx^{(i)}$

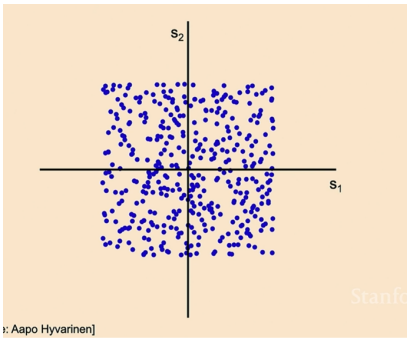
1. ICA ambiguities

그렇다면 우리는 A 행렬도 모르는데 W를 어떻게 찾을 수 있는 것일까?

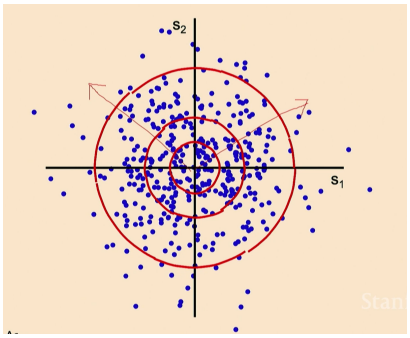
우리는 source , A의 사전정보가 없다면 여기에는 복구가 불가능할수도 있다는 inherent ambiguities가 존재한다.

이렇게 ICA에 source에 대한 ambiguity만 존재하게 하기 위해서 source는 또한 non-gaussian이어야 한다. 만약 가우시안 데이터(정규화됨)라면 분포는 원점을 중심으로 원 형태로 나타나고 density는 rotationally sysmmetric하기 때문에 복구에 필요한 충분한 정보가 존

재하지 않으며 미분 또한 불가능하다. 따라서 non-gaussian인 경우만 ICA에서 사용할수 있다고 한다.



uniform distribution



gaussian distribution

2. Densities and linear transformations

독립 성분 분석에 대한 알고리즘을 이해하기 위해 필요한 마지막 개념은 랜덤변수에 선형변환을 적용했을 때 변환 적용 전 후 변수의 확률밀도(cdf) 간의 간계이다.

확률밀도 함수의 면적의 총 합이 1이 되어야 하기 때문에 어떤 행렬로 랜덤 변수를 선형변환해주면 그 확률밀도함수는 행렬식을 이용해 보정해주어야 한다.

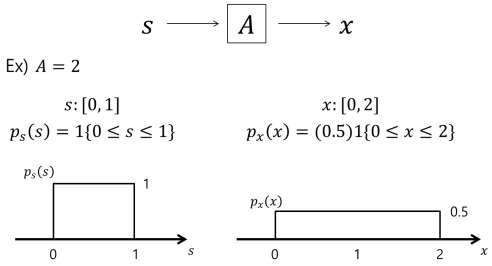
예를 들어

랜덤변수 $s \sim Uniform[0,1]$이라고 할 때 density function은 $p_s(s) = 1\{0 \leq s \leq 1\}$이다.

이때 $A = 2$라고 하고 $x = As$이고, $s = A^{-1}x = Wx$라고 할때,

$p_x(x) = p_s(Wx)$가 아니라

$p_x(x) = p_s(Wx)|W|$이다.



3. ICA algorithm

드디어 ICA algorithm을 유도할 준비가 되었다.

이를 위해 먼저 p(x)를 구해야 한다.

- p(x) 정의

$p(x)$를 계산하는 것은 우리가 실제로 확인할 수 있는 데이터는 s [원본]가 아니라 x [결과] 이기 때문이다.

각 source s_i의 density function을 $p_s(s_i)$라고 하면 모든 source로 구성된 joint distribution은 다음과 같다.

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

((이때 source끼리는 independent하다는 가정을 만족해야 한다.))

이에 이전에 정의했던 $s = A^{-1}x = Wx$라는 관계를 적용시키면 다음과 같아진다.

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

- log likelihood

이제 우리의 training example $\{x^{(1)}, \dots, x^{(m)}\}$의 density function인 p(x)를 통해 log likelihood를 계산해 볼 것이다.

likelihood를 계산하는 이유는 수많은 parameter 중 어떤 parameter를 쓰는게 지금까지 얻은 정보들을 잘 설명할 수 있는지를 알아보기 위함이다.

m 개의 학습 데이터와 n개의 source에 대해 log likelihood를 계산하면 다음과 같다.

$$l(W) = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \log p_s(w_j^T x^{(i)}) + log|W| \right)$$

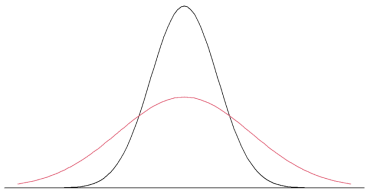
또, 여기서 우리는 p_s에 대한 구체적인 확률밀도함수를 결정해볼 수 있다.

일반적으로 p_s의 CDF로 시그모이드 함수 $g(s) = \frac{1}{(1+e^{-s})}$를 default로 사용한다. 그 이유는 초기에 sigmoid함수를 사용했을 때 알고리즘에 문제가 없어서 주로 이것을 사용해왔기 때문이라고 한다.

▼ 추가

- speaker의 음성신호의 density는 어떻게 선택하는 것이 좋을까?

  여기에서는 주로 cdf함수로 시그모이드 함수를 선택하고 이를 미분한 값을 pdf 분포를 사용한다. gaussian보다 이 pdf를 주로 사용하는 이유는 gaussian 보다 외각 부분이 flatter하기 때문에 극단적인 outlier를 많이 포함시킬 수 있게 되었고 이는 현실에서의 사람의 목소리나 많은 자연적인 현상을 더 잘 나타낼 수 있기 때문이다.



  (만약 source의 density에 대한 사전지식이 있다면 해당 density function을 사용하면 된다.)

pdf: $p_s = g'(s)$이므로 위의 log likelihood function에 적용하면 아래와 같다.

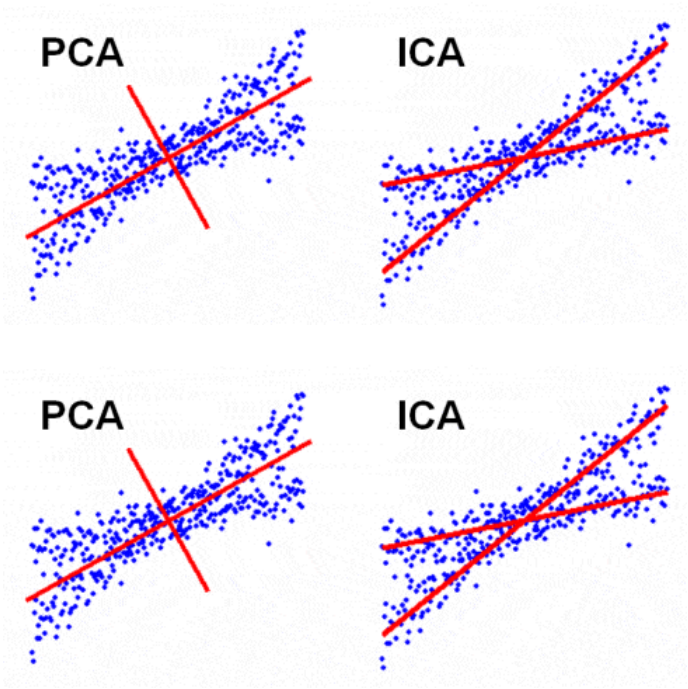
$$l(W) = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \log g'(w_j^T x^{(i)}) + log|W| \right)$$

그렇다면 이제 loglikelihood를 최대화 하는 W를 찾아보자.

1. W에 대해 편미분

$\nabla_W |W| = |W|(W^{-1})^T$라는 식을 이용하여 편미분하면

$$\frac{\partial l}{\partial W} = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \frac{1}{g'(s_j)} \cdot g''(s_j) \cdot x^{(i)^T} + \frac{1}{|W|} |W| (W^{-1})^T \right)$$
$$= \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \frac{1}{g(s_j)(1-g(s_j))} \cdot g(s_j)(1-g(s_j))(1-2g(s_j)) \cdot x^{(i)^T} + (W^{-1})^T \right)$$
$$= \sum_{i=1}^{m} \left(\sum_{j=1}^{n} (1-2g(s_j)) \cdot x^{(i)^T} + (W^{-1})^T \right)$$



2. Gradient ascent 이용하여 W 업데이트

$$W := W + \alpha \frac{\partial l}{\partial W}$$

따라서 각 $j = 1, 2, \cdots, n$에 대해

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

여기서 α는 learning rate이다.

이 알고리즘이 수렴한 후에 얻는 W를 $s^{(i)} = Wx^{(i)}$에 대입하면 원래의 source를 얻을 수 있다.

참고: PCA와 ICA의 차이

PCA와 ICA 모두 공통적으로 주어진 데이터를 대표하는 기저벡터를 찾아준다는 점에서는 같은 역할을 한다고 할 수 있다.

PCA는 feature space에서 직교하는 기저 벡터 집합을 찾아준다. 특히, PCA는 데이터를 정사영했을 때 maximum variance를 얻을 수 있는 벡터를 차례대로 기저벡터로 삼는 특징이 있다.

반면 ICA를 통해 찾은 기저벡터들은 서로 직교하지 않을 수도 있고 ICA를 통해 얻은 기저벡터들에 데이터를 정사영했을 때 그 결과들이 최대한 독립적(non-Gaussian)일 수 있도록하는 벡터들을 기저벡터로 삼는 특징이 있다.