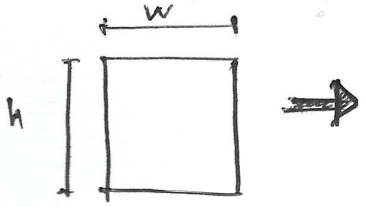
15 Dimensionality Reduction

Let's consider the case in which the latent variable $z_{ij} \in \mathbb{R}$ so they are continuos and we have two cases:

- we can define a <u>linear model</u> of $P(x|z)$ (likelihood)
- we can consider <u>non linear model</u> (iterative)

This is just an overview of what is the topic. Any image dataset has this issue: THE INPUT SPACE HAS THE DIMENSIONALITY of the product given by the height and width of the image.



INPUT SPACE: $w \times h$

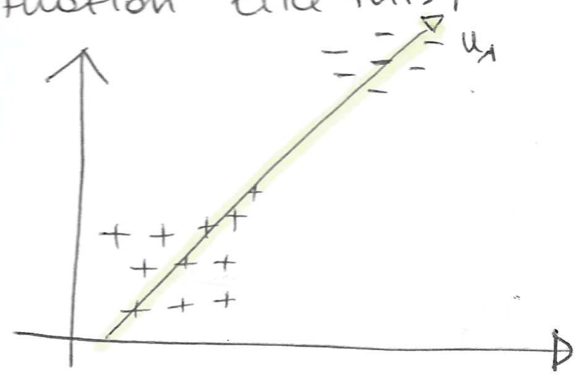
> If you consider all combinations we have many images that are not meaningfull, since if you are interested in classifing cats and dogs you have to consider less images.

THE IDEA IS TO ANALYZE THE INPUT TO UNDERSTAND THE ACTUAL DIMENSIONS THAT MAKE THE VARIABILITY OF OUR DATASET.

If you operate translation and rotation you have 3 degree of freedom: 2D translation + rotation.

CONCLUSION: deal with data with high dimentions by LOOKING FOR LOWER DIMENSIONAL EMBEDDING

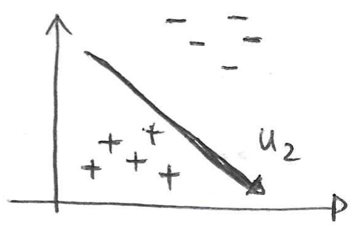
Let's consider a dataset in two dimentions by non-consid. the labels, $D = \{(x_i)_{i=1}^{N}\}$, WE WANT TO UNDERSTAND THE REAL VARIABILITY of THESE DATA. Imagine to have a situation like this:



> In this case we consider also labels. One possibility is TO RECOGNIZE A DIRECTION of these data on WHICH I CAN PROJECT the points, obtaining a transformation of the dataset!
>
> 
>

m dimentional data $\Longrightarrow$ m dimensional data. (2)

I may loose some information if I do not do this ~~reformation~~ transformation properly. Let's consider still the example of before:



if I project everything on u_2, this is not a good choice, because I will have + and − mixed together.

There are two different formulation of the problem that achieve the same solution:

- MAXIMUM VARIANCE
- ERROR MINIMIZATION

$\boxed{\text{VARIANCE MAXIMIZATION}}$

Given data $\{\vec{x}_n\} \in \mathbb{R}^D$.

GOAL: Maximize data variance after projection to some direction $\vec{u}_1$.

$$\boxed{\text{Projected points: } \vec{u}_1^T \vec{x}_n \quad , \text{ where } \vec{u}_1^T \vec{u}_1 = 1}$$

we are interested only in the direction of u_1, it is a <u>versor</u>

We can compute of the data points and the mean of projected data points. We can compute the variance of the projected points.

- Mean of data points: $\boxed{\frac{1}{N} \sum_{n=1}^{N} x_n = \vec{\overline{x}}}$
- Mean of the projected data points: $\hat{u}_1^T \overline{x}$
- Variance of projected data points:

$$\frac{1}{N} \sum_{n=1}^{N} \left[\hat{u}_1^T \vec{x}_n - \hat{u}_1^T \vec{\overline{x}} \right]^2 = \hat{u}_1^T S \hat{u}_1$$

when $S = \frac{1}{N}\sum_{M=1}^{N}(\vec{x}_M - \vec{\bar{x}})(\vec{x}_M - \vec{\bar{x}})^T$ ← MATRIX COMPUTE FROM the dataset

In general this transformation is done in a multimensional space. We want to find some direction in which the variance is maximum, this can be done with optimization techniques by removing the costraints, adding a lagrange multiplier,...

$$\max_{\hat{u}_1} \hat{u}_1^T S \hat{u}_1 \quad \text{s.t.} \quad \hat{u}_1^T \hat{u}_1 = 1$$

by adding Lagrange multipliers:

$$\max_{\hat{u}_1} \hat{u}_1^T S \hat{u}_1 + \lambda_1 (1 - \hat{u}_1^T \hat{u}_1)$$

SOLUTION: $\underbrace{S\hat{u}_1 = \lambda_1 \hat{u}_1}$ (obtained by setting the derivative w.r.t. $\hat{u}_1$ to zero)

This is the definition of eigen vector/eigen values.

u_1 IS AN EINGEN VECTOR OF THE MATRIX S. Since we want to maximize the term, then solution is the eigen value with the maximum value. By left multiplying by $\hat{u}_1^T$ we get

$$\hat{u}_1^T S \hat{u}_1 = \lambda_1$$

VARIANCE IS MAXIMAL WHEN $\hat{u}_1$ IS THE EIGEN VECTOR CORRESPONDING TO THE LARGEST EIGEN VALUE λ_1

This is called the FIRST PRINCIPAL COMPONENT

1. compute S matrix
2. compute eigen values
3. take the maximum eigen values and its eigenvector
4. the direction that correspond to the maximum variance of the projected points

THE SOLUTION IS SIMPLE

(We take M maximum λ_i in a general case)

ERROR MINIMIZATION

We can consider a subspace of vectors that are ortoghonal each other, in particular we define the subspace in this way:

$$\vec{u}_i^T \vec{u}_j = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

ORTHONORMAL SPACE. We can express each point in the dataset as a linear combination of vectors in this new reference system.

$$\boxed{\vec{x}_M = \sum_{i=1}^{D} \alpha_{Mi} \vec{u}_1}$$

With a little bit of arregements we can split the d-components in two parts:

- up TO M, we have coefficient related to the input samples
- FROM M+1 to D we have some coefficients not related to input samples:

$$\tilde{x}_M = \sum_{i=1}^{M} z_{Mi} u_i + \sum_{i=M+1}^{D} b_i \vec{u}_i$$

WE APPROXIMATE $\vec{x}_M$ USING A LOWER DIMENSIONAL REPRESENTATION, we consider only some of the components.

We want to minimize the error, finding the CORRECT z_{Mj} and b_i

ERROR FUNCTION

$$\boxed{J = \frac{1}{N} \sum_{n=1}^{N} \| \vec{x}_M - \tilde{x}_M \|^2}$$

We can compute the difference the points, what is CALLED THE RESIDUAL (components not used in the first M), the D−M remaing has been approximated and the error is due to only to them.

Minimize the error w.r.t to

$\boxed{z_{nj}\text{, we get:}}$	$\boxed{b_j\text{, we get:}}$
$z_{nj} = \vec{x}_n u_j \,, \; j = 1, \dots, M$	$b_j = \bar{x}^T u_j \,, \; j = M+1, \dots, D$

The first M components do not affect the error, it is affected only by the D − M approximated components:

$$x_n - \tilde{x}_n = \sum_{i=M+1}^{D} \left[(x_n - \bar{x}) \vec{u}_i \right] \vec{u}_i$$

(omitted "→" for brevity)

The overall error approximation becomes:

$$J = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=M+1}^{D} \left(\vec{x}_n^T \vec{u}_i - \bar{\vec{x}}^T \vec{u}_i \right)^2 = \sum_{i=M+1}^{D} \vec{u}_i^T S \vec{u}_i$$

Again we can put this to a lagrangian form and the solution will be again the eigen vector and values of S matrix. Now we are going to minimize the error term, so WE WILL TAKE THE D−M SMALLEST EIGEN VALUES.

$$\boxed{S \vec{u}_i = \lambda_i \vec{u}_i}$$

The solution is equivalent to the previous case.

CHOOSING D − M smallest eigen values of S corresponds to find M highest eigen values of S as in the maximum variance formulation.

let's consider:

$f : X \longmapsto Y$, where $X \in \mathbb{R}^D$ and $\{x_i\}_{i=1}^N$, so N is the actual size of the dataset. We may have situations in which $N < D$. For instance:

$$D = \underbrace{640 \times 480 \times 3}_{\text{high resolution color images}} \qquad N = \underbrace{10'000}_{\#\text{images}} \,, \quad N < D.$$

In this situation we have numerical problem, the S matrix is difficult to compute and in this situation we work directly on the data;

Define $\underline{X}$ as the $N \times D$ centered data matrix whose n-th row is $(\vec{x}_n - \vec{\bar{x}})^T$. The Matrix S can be written in this way:

$$\boxed{S = \frac{1}{N} \underline{X}^T \underline{X}}$$

By doing some easy processing with this matrix, we get:

(by left multiplying by X)

$$\frac{1}{N} X^T X u_i = \lambda_i u_i \Rightarrow \frac{1}{N} X X^T \underbrace{(X u_i)}_{v_i} = \lambda_i \underbrace{(X u_i)}_{v_i}$$

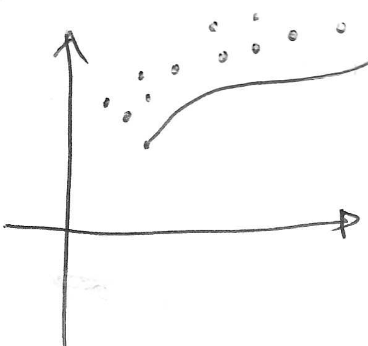
$$\Rightarrow \boxed{\frac{1}{N} X X^T \vec{v}_i = \lambda_i \vec{v}_i}$$

$X X^T$ is an $N \times N$ matrix whose eigen values can be computed efficiently.

Note: $X X^T$ has the same $N-1$ eigenvalues of $X^T X$ (the others are 0)

These methods are called PCA, Principal component Analysis.

Let's see another model that will take into account LATENT VARIABLES. The idea is instead of considering a subspace, we can consider whatever surface:



We may want to project data on this curve.

FIND A TRANSFORMATION OF THE DATASET TO ALLOW BETTER SEPARATION

LINEAR LATENT VARIABLE MODEL

We assume linearity between the latent variables and input variables. We ASSUME a gaussian distribution for latent variables with $\mu = 0$ and $\sigma = I$.

$$\vec{x} = W\vec{z} + \vec{\mu} \quad , \quad P(\vec{z}) = \mathcal{N}(\vec{z}; 0, I)$$

$$P(\vec{x} \mid \vec{z}) = \mathcal{N}(\vec{x}; W\vec{z} + \vec{\mu}, \sigma^2 I)$$

We assume that the posterior distribution is given by a LINEAR GAUSSIAN

The marginal distribution is also a gaussian because it is a combination of gaussian and since we are in the continuos space!

$$P(\vec{x}) = \int P(\vec{x} \mid \vec{z}) P(\vec{z}) d\vec{z} = \mathcal{N}(\vec{x}; \vec{\mu}; C)$$

$$C = WW^T + \sigma^2 I$$

Posterior:

$$P(\vec{z} \mid \vec{x}) = \mathcal{N}(\vec{z}; M^{-1} W^T (\vec{x} - \vec{\mu}), \sigma^2 M)$$

$$M = W^T W + \sigma^2 I$$

Now we find the parameters by maximizing the likelihood:

$$\boxed{\underset{W, \vec{\mu}, \sigma}{\operatorname{argmax}} \ \ln P(X \mid W, \vec{\mu}, \sigma^2)}$$

we set the derivative to zero...

The matrix W depends on the eigenvalues and eigenvectors of S. Maximum likelihood solution for the probabilistic PCA model can be obtained also with EM algorithm.

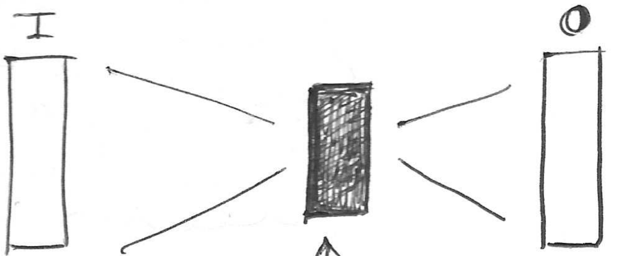
NON LINEAR LATENT VARIABLE MODELS

A non linear model can be relevant for some problems

In the non linear model we do not have a close form solution, so it is easy with iterative solution.

The best way to work with non linear models is to use neural networks as a way to approximate functions.

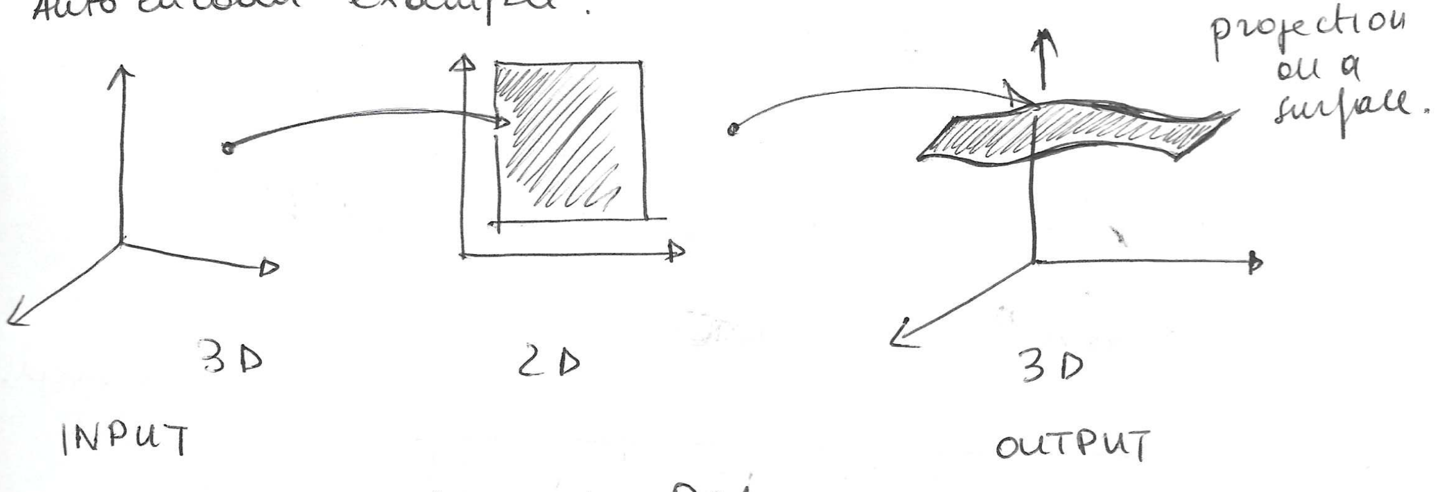
One of the most suitable approach to learn non linear latent variable models is to use AUTO ASSOCIATIVE NN, designed in such a way the input layer and output layer have the same number of units.



You train the network by taking each sample of the dataset on the input and on the output

Interested to compute the parameters in this layer, it is then that the dimensionality reduction happens.

Auto encoder example:



Works better than the PCA.