Dimension reduction: PCA, tSNE

Diane Lingrand

POLYTECH NICE SOPHIA

UNIVERSITÉ CÔTE D'AZUR

SI4

2022 - 2023

Outline

Mean and variance

- n samples of dimension 1 (scalars) : $\{x^0, x^1 \dots x^{n-1}\}$
  - mean $\mu = \frac{1}{n}\sum_j x^j$
  - variance $\text{var}(x) = \sigma^2 = \frac{1}{n}\sum_j (x^j - \mu)^2$

Covariance

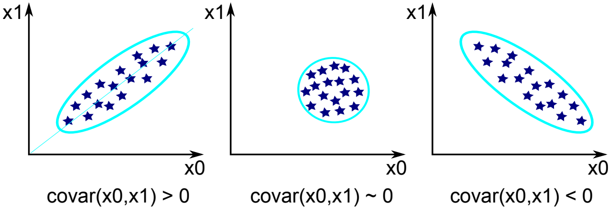
- covariance between 2 variables x_0 and x_1 :
  - measure of the linear relationship between two random variables
  - $\text{covar}(x_0, x_1) = \text{E}[(x_0 - \mu_0)(x_1 - \mu_1)] = \frac{1}{n}\sum_j (x_0^j - \mu_0)(x_1^j - \mu_1)$

Covariance

- covariance between 2 variables x_0 and x_1 :
  - measure of the linear relationship between two random variables
  - $\text{covar}(x_0, x_1) = \text{E}[(x_0 - \mu_0)(x_1 - \mu_1)] = \frac{1}{n}\sum_j (x_0^j - \mu_0)(x_1^j - \mu_1)$
- Example of linearly correlated variables : $x_1^j = \lambda x_0^j$
  - $\text{covar}(x_0, x_1) = \sum_j (x_0^j - \overline{x_0})(x_1^j - \overline{x_1}) = \lambda \, \text{var}(x_0)$
    - high value : means correlation between the 2 variables

Covariance

- covariance between 2 variables x_0 and x_1 :
  - measure of the linear relationship between two random variables
  - $\text{covar}(x_0, x_1) = \text{E}[(x_0 - \mu_0)(x_1 - \mu_1)] = \frac{1}{n}\sum_j (x_0^j - \mu_0)(x_1^j - \mu_1)$
- Example of linearly correlated variables : $x_1^j = \lambda x_0^j$
  - $\text{covar}(x_0, x_1) = \sum_j (x_0^j - \overline{x_0})(x_1^j - \overline{x_1}) = \lambda\, \text{var}(x_0)$
    - high value : means correlation between the 2 variables
- Example of non correlated variables : $E[x_0 * x_1] = E[x_0] * E[x_1]$
  - thus $\text{covar}(x_0, x_1) = 0$



Mean, variance and covariance : dim 2

- n samples of dimension 2 : $\boldsymbol{x}^j = [x_0^j, x_1^j]$ with $0 \leq j < n$

Mean, variance and covariance : dim 2

- n samples of dimension 2 : $\boldsymbol{x}^j = [x_0^j, x_1^j]$ with $0 \leq j < n$
- mean of samples : $\boldsymbol{\mu} = 0.5\,[x_0^0 + x_0^1, x_1^0 + x_1^1]$
- variances :

Mean, variance and covariance : dim 2

- n samples of dimension 2 : $\boldsymbol{x}^j = [x_0^j, x_1^j]$ with $0 \leq j < n$
- mean of samples : $\boldsymbol{\mu} = 0.5\,[x_0^0 + x_0^1, x_1^0 + x_1^1]$
- variances :
  - $\mathrm{var}(x_0) = \mathrm{covar}(x_0, x_0) = \sigma_0^2 = \frac{1}{n}\sum_j (x_0^j - \mu_0)^2$
  - $\mathrm{var}(x_1) = \mathrm{covar}(x_1, x_1) = \sigma_1^2 = \frac{1}{n}\sum_j (x_1^j - \mu_1)^2$
- covariance matrix :

$$\Sigma = \begin{pmatrix} \sigma_0^2 & \mathrm{covar}(x_0, x_1) \\ \mathrm{covar}(x_0, x_1) & \sigma_1^2 \end{pmatrix}$$

- variance : $\mathrm{var}(\boldsymbol{x}) = \mathrm{tr}(\Sigma) = \sigma_0^2 + \sigma_1^2$

Variance-covariance matrix

- original variables of dimension $p \geq 2$: $\mathrm{X^j} = [x_0^j \dots x_{p-1}^j]$
- variance-covariance matrix : symmetric matrix of dim $p \times p$:

$$\Sigma = \begin{pmatrix} \text{var}(x_0) & \text{covar}(x_0, x_1) & \dots & \text{covar}(x_0, x_{p-1}) \\ \text{covar}(x_0, x_1) & \text{var}(x_1) & \dots & \text{covar}(x_0, x_{p-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \text{covar}(x_0, x_{p-1p}) & \text{covar}(x_1, x_{p-1}) & \dots & \text{var}(x_{p-1}) \end{pmatrix}$$

- variance : $\text{var}(\boldsymbol{x}) = \mathrm{tr}(\Sigma) = \sum_i \sigma_i^2$

Outline

PCA (1901 Karl Pearson, 1936 H. Hotelling)

- Unsupervised
- Analysis of variance-covariance matrix
- Reducing the dimension of data
- Visualisation of data of the reduced dimension is 2 or 3
- Interpretation : dependance between variables
- PCA : often as pre-processing

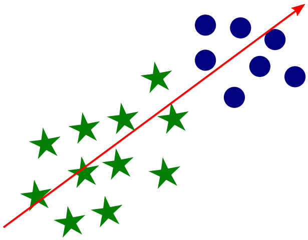
Explained variance

$$\Sigma = \begin{pmatrix} \mathrm{var}(x_0) & \mathrm{covar}(x_0, x_1) & \dots & \mathrm{covar}(x_0, x_{p-1}) \\ \mathrm{covar}(x_0, x_1) & \mathrm{var}(x_1) & \dots & \mathrm{covar}(x_0, x_{p-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{covar}(x_0, x_{p-1p}) & \mathrm{covar}(x_1, x_{p-1}) & \dots & \mathrm{var}(x_{p-1}) \end{pmatrix}$$

- variance $= \mathrm{tr}(\Sigma) = \sum_i \sigma_i^2$
  - symmetric squared matrix : diagonalization is possible !
  - there exists a basis of orthogonal vectors where the covariance matrix is diagonal
    - these vectors are eigenvectors of Σ
    - elements on the diagonal are eigenvalues
    - variance $= \sum_k \lambda_k$
- Idea of PCA
  - diagonalisation of Σ
    - order eigenvalues by decreasing order
  - if 0 is a eigenvalue : the corresponding dimensions can be removed
  - the lower eigenvalues do not contribute a lot to the variance

Geometrical interpretation

- original variables : $x_1, x_2, .., x_p$
- principal components : $c_1, c_2, ...c_k, ..., c_q$ with $q \leq p$
- $c_k = \sum_j a_{jk} x_j$ with :
  - c_k and c_j not correlated
  - maximum variance and
  - decreasing importance



Principal component

- eigenvalues, ordered - eigenvectors
- $tr(\Sigma) = \sigma^2 = \sum_{i=1}^{n} \lambda_i$
- each eigenvalue participates to the global variance

Examples : variance explained

- PCA on Iris dataset :
  - if we perform the PCA on dimension 4 (not very useful) :

```
from sklearn import datasets
from sklearn.decomposition import PCA
X, y = datasets.load_iris(return_X_y=True)
pca4 = PCA(n_components=4)
pca4.fit(X)
X4 = pca4.transform(X)
print("explained variance : ", pca4.explained_variance_ratio_)
```

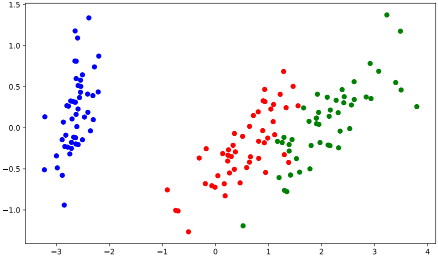
```
explained variance :  [0.92461872 0.05306648 0.01710261 0.00521218]
```

  - with 4 components : 100% of the variance is explained
  - with 3 components : 99.5%
  - with 2 components : 97.8%

Examples : plot the Iris dataset on 2d

```
# PCA transformation
pca2 = PCA(n\_components=2)
pca2.fit(X)
X2 = pca2.transform(X)

#plot
colors = ['b', 'r', 'g']
col = [colors[c] for c in y]
plt.figure(figsize=(10, 6))
plt.scatter(X2[:, 0], X2[:, 1], c=col, marker="o")
```



Links to codes

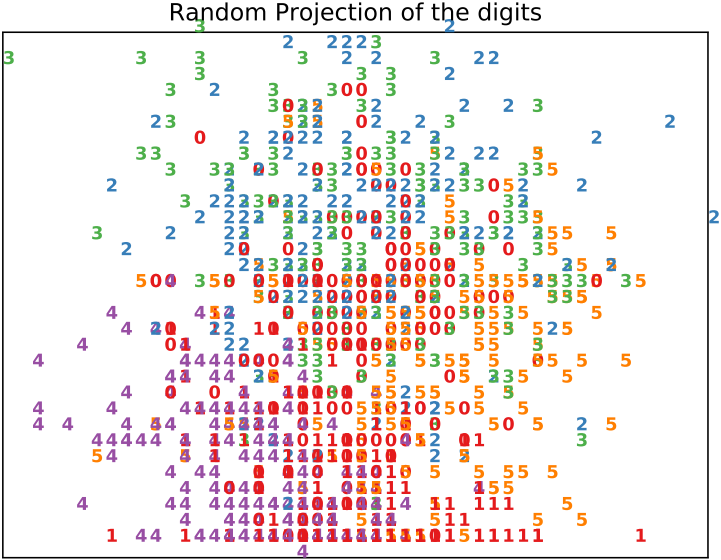
- PCA on Iris dataset :
  - from dimension 4 to dimension 3 for visualisation (https://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_iris.html)
  - from dimension 4 to dimension 2 (https://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html)
  - explained variance ratio (first two components) : [0.92461872 0.05306648]

Practice (part 1)

- With the digit dataset
  - Find the smallest dimension after PCA such that 95% of the variance is explained.
    - hint : `numpy.cumsum` and `numpy.where`
  - What is the proportion of explained variance in dimension 2 ?
  - Plot the digits after a PCA in 2D. Compare with the previous approach.

Outline

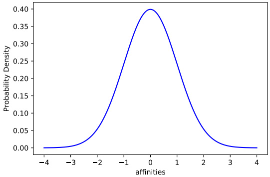
Random projection

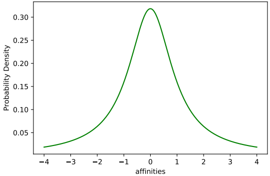


Idea of t-SNE

- Build map in which distances between points reflect similarities in the data
  - typical map dimension : 2 or 3
  - preserving local structures
  - t-SNE : try to avoid all points collapsing
- Non linear dimension reduction
  - converts affinities of data points to probabilities represented by Gaussian joint probabilities
  - affinities in the embedded space are represented by Student's **t**-distributions (heavy tailed)
  - minimisation of Kullback-Leibler divergence of the two distributions (gradient descent) : gives the coordinates in the embedded space
- Exact algorithm of t-SNE is computationally expensive (huge compared to PCA)
- Stochastic algorithm : multiple restarts with different seeds can yield different results

- Why a Gaussian distribution ?
  - In the original space, we want to capture close elements and do no care of distant elements
- Why a Student's t-distribution ?
  - In the embedded space, the samples are initially randomly projected. We need to be able to capture them.



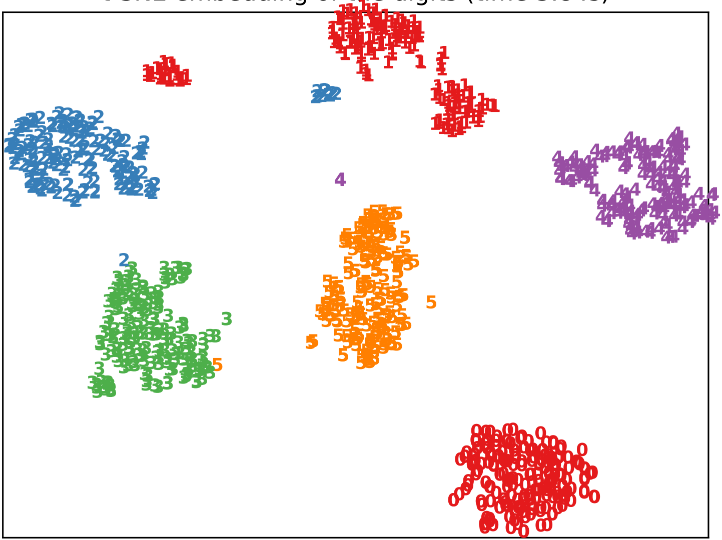


Iterative algorithm

- random initialisation of samples in the embedded space
- iterative minimisation of the KL divergence :
  - compute the distances between embedded points
  - use the t-distribution to transform these values + normalisation
  - compute the gradient of KL and move the samples points in the embedded space

Example using the digit dataset

t-SNE embedding of the digits (time 3.84s)

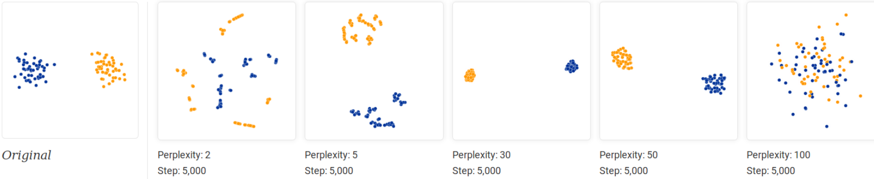


Short intro to t-SNE

https://www.youtube.com/watch?v=NEaUSP4YerM

Parameters of t-SNE

- Perplexity (usually between 5 and 50). Illustration from `https://distill.pub/2016/misread-tsne/`



- Early exaggeration factor : optimization in two steps :
  - exaggeration phase : joint probabilities in the original space are artificially multiplied by a factor
  - final optimization
- Learning rate ϵ : not too small, not too large.
- Maximum number of iterations : 5000 ?
- angle (not used in the exact method)

Barnes-Hut t-SNE

- approximation of t-SNE, more scalable.
  - many of the pairwise interactions between points are similar
- Another parameter : angle :
  - tradeoff between performance and accuracy
  - usual range : from 0.2 to 0.8
    - larger angles imply that we can approximate larger regions by a single point, leading to better speed but less accurate results.
- Limitations :
  - target dimension less than 3. Mostly 2.
  - only for dense dataset (for sparse dataset use exact t-SNE)

Code for visualisation in 2D using t-SNE

```python
from sklearn import manifold
tsne = manifold.TSNE(n_components=2, init='pca', random_state=0)
X_tsne = tsne.fit_transform(X)
```

Practice (part2)

- Compare PCA and tSNE for the visualisation in 2D of the digit dataset
  - compare the speed of transformation
  - compare the plots and play with the parameters