Calcul Numeric

Curs 13

2020

Anca Ignat

Principal Component Analysis (PCA)

"a way of identifying patterns in data, and expressing the data in such a way to highlight their similarities and differences"

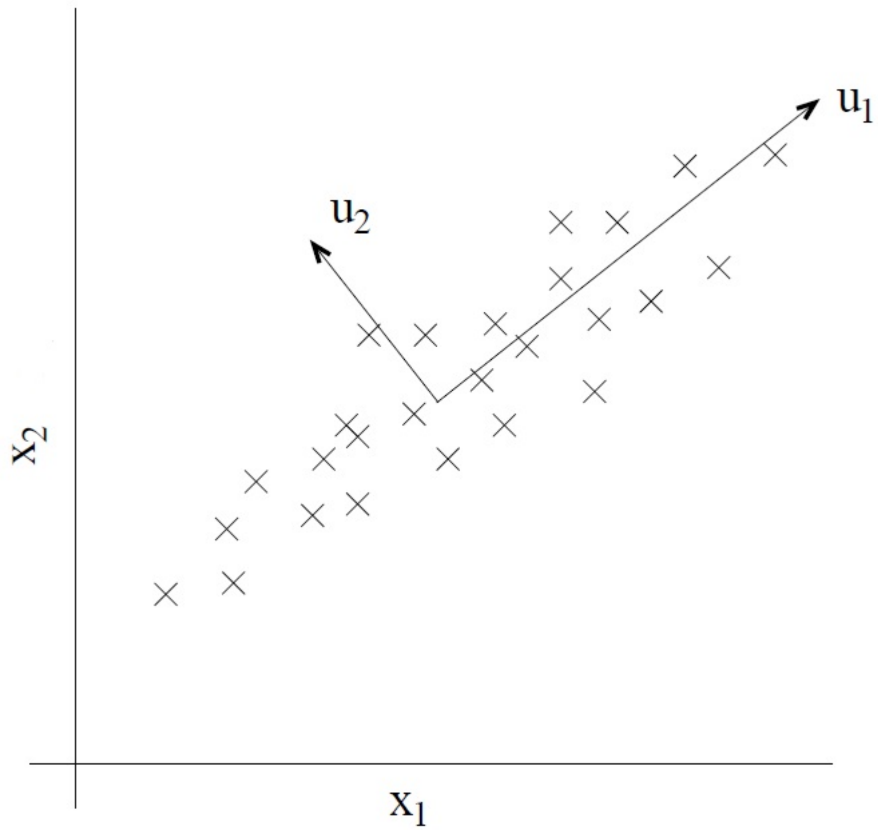
$\boldsymbol{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^p\}, \boldsymbol{x}^k \in \mathbb{R}^n$ - a set of observations

$\boldsymbol{X} \subseteq \boldsymbol{Y} \subset \mathbb{R}^n$, $\mathbf{dim}\boldsymbol{Y} = \boldsymbol{k} \ll \boldsymbol{n}$ - dimension reduction

(removing redundancies in data)

- some components of the vectors $\boldsymbol{x}^k$ may be linearly dependant one with the other

($\boldsymbol{x_i}$ =grades of a student, $\boldsymbol{x_j}$ =credits obtained by the student)



$\boldsymbol{u_1}$ – the principal direction of the data

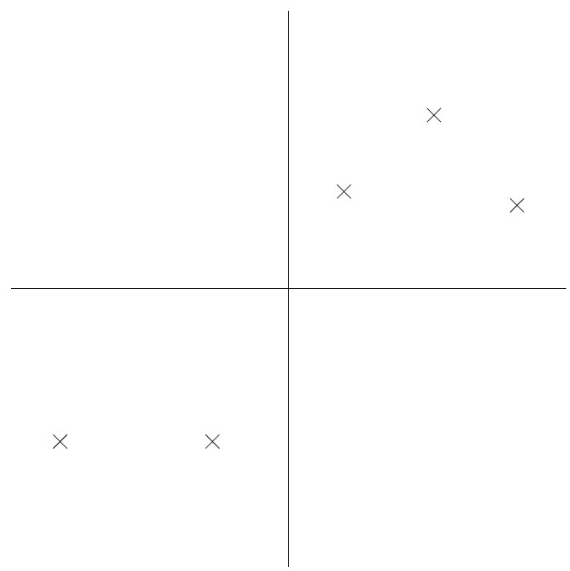
How to automatically compute $\boldsymbol{u_1}$?

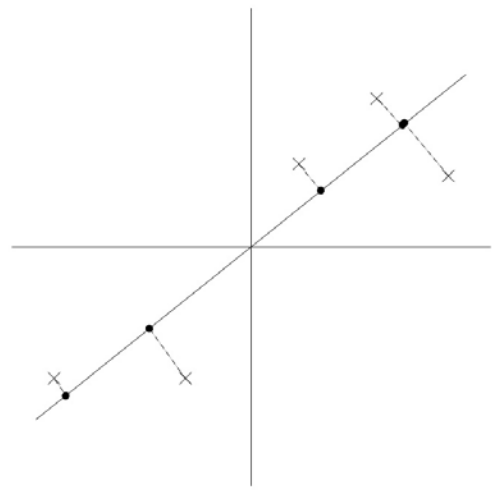
Pre-processing of data

1. Compute the mean value $\boldsymbol{\mu} = \frac{1}{p}\sum_{k=1}^{p} \boldsymbol{x}^{k}$
2. Subtract mean from data: replace $\boldsymbol{x}^{\boldsymbol{k}}$ with $\boldsymbol{x}^{\boldsymbol{k}}$-$\boldsymbol{\mu}$
3. Compute $\boldsymbol{\sigma}_i^2 = \frac{1}{p}\sum_{k=1}^{p}\left(\boldsymbol{x}_i^k\right)^2 , \; \boldsymbol{i} = \mathbf{1},\ldots,\boldsymbol{n}$
4. Replace each $\boldsymbol{x}_i^k$ with $\frac{1}{\boldsymbol{\sigma}_i}\boldsymbol{x}_i^k$

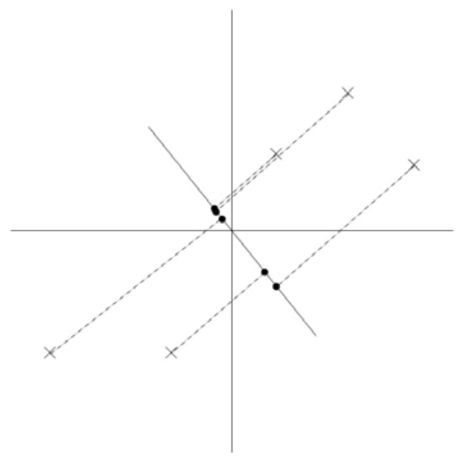
(1)+(2) – creates data with zero mean
(3)+(4) – scale normalization (when the components were computed with different measure units, marks ↔ credits)

Compute the “major axis of variation” – ***u*** – the direction that describes the data positioning/ alignment.
Find the unit vector ***u*** such that projecting the data onto the direction given by ***u*** one obtains new data with maximized variance.





“good” direction



“bad” direction

The length of the projection of vector $\boldsymbol{x}$ onto direction $\boldsymbol{u}$ is given by the inner product $\boldsymbol{x^T u = (x,u)}$.
To maximize the variance of projections one needs to solve the optimization problem:

$$\max\{\frac{1}{p}\sum_{k=1}^{p}\left(\left(x^k\right)^T u\right)^2 = \frac{1}{p}\sum_{k=1}^{p} u^T x^k \left(x^k\right)^T u =$$

$$u^T\left(\frac{1}{p}\sum_{k=1}^{p} x^k\left(x^k\right)^T\right)u \;\; ; \;\; u \in \mathbb{R}^n, \|u\|_2 = 1\}$$

The solution to this problem is the **unit eigenvector** corresponding to the **biggest eigenvalue** of the **covariance matrix** ***C*** of the dataset (with zero mean!) ***X***.

$$\boldsymbol{C} = \frac{\mathbf{1}}{\boldsymbol{p}} \sum_{\boldsymbol{k}=\mathbf{1}}^{\boldsymbol{p}} \boldsymbol{x}^{\boldsymbol{k}} \left(\boldsymbol{x}^{\boldsymbol{k}} \right)^{\boldsymbol{T}}$$

Consider the eigenvalues of the covariance matrix in decreasing order:

$$\lambda_{\mathbf{1}} \geq \lambda_{\mathbf{2}} \geq \cdots \geq \lambda_{\boldsymbol{n}}$$

The covariance matrix being symmetric, one can compute an orthonormal basis of eigenvectors $\boldsymbol{u}^{\mathbf{1}}, \boldsymbol{u}^{\mathbf{2}}, \ldots, \boldsymbol{u}^{\boldsymbol{n}}$ - this are the principal components for dataset $\boldsymbol{X}$. One can consider only the first most important $\boldsymbol{r}$ principal components, and project the data on the space generated by the first $\boldsymbol{r}$ eigenvectors $\boldsymbol{u}^{1}, \boldsymbol{u}^{2}, \ldots, \boldsymbol{u}^{r}$.

To represent $\boldsymbol{x}^k$ on this basis we compute the vector $\boldsymbol{y}^k \in \mathbb{R}^r$ thus achieving dimensionality reduction.

$$\boldsymbol{y}^k = \begin{pmatrix} (\boldsymbol{u}^1)^T \boldsymbol{x}^k \\ (\boldsymbol{u}^2)^T \boldsymbol{x}^k \\ \vdots \\ (\boldsymbol{u}^r)^T \boldsymbol{x}^k \end{pmatrix} \in \mathbb{R}^r$$

$\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^r$ - the principal components

PCA algorithm

1. Compute the mean value $\boldsymbol{\mu} = \frac{1}{p}\sum_{k=1}^{p} \boldsymbol{x}^{k}$

2. Subtract mean from data: replace $\boldsymbol{x}^k$ with $\boldsymbol{x}^k\text{-}\boldsymbol{\mu}$

/* 3. + 4. can be skipped if data have the same scale

3. Compute $\boldsymbol{\sigma}_i^2 = \frac{1}{p}\sum_{k=1}^{p}\left(\boldsymbol{x}_i^k\right)^2$, $i = 1,\ldots,n$

4. Replace each $\boldsymbol{x}_i^k$ with $\frac{1}{\boldsymbol{\sigma}_i}\boldsymbol{x}_i^k$

5. Compute new $\boldsymbol{X} = \left[\, \boldsymbol{x}^1 \; \boldsymbol{x}^2 \cdots \boldsymbol{x}^p \,\right]$

6. Compute the covariance matrix $\boldsymbol{C} = \frac{1}{p}\boldsymbol{X} * \boldsymbol{X}^T$

7. Compute the eigenvalues and eigenvectors

(orthonormal!) of the covariance matrix

8. Sort the eigenvalues (and eigenvectors) in decreasing order

9. Choose first $\boldsymbol{r}$ eigenvectors $\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^r$

$$\boldsymbol{U} = \left[\boldsymbol{u}^1 \ \boldsymbol{u}^2 \ \ldots \ \boldsymbol{u}^r \right] \in \mathbb{R}^{n \times r}$$

10. Project data on principal components

$$\boldsymbol{Y} = \boldsymbol{U}^T * \boldsymbol{X} \in \mathbb{R}^{r \times p}, \boldsymbol{Y} = \left[\boldsymbol{y}^1 \ \boldsymbol{y}^2 \cdots \boldsymbol{y}^p \right]$$

Andrew Ng - *Principal Component Analysis*
http://cs229.stanford.edu/notes/cs229-notes10.pdf

Lindsay I. Smith - *A tutorial on principal components analysis*, 2002.
https://ourarchive.otago.ac.nz/bitstream/handle/10523/7534/OUCS-2002-12.pdf?sequence=1

Jonathon Shlens - *A tutorial on principal component analysis*, *arXiv preprint arXiv:1404.1100* (2014).
https://arxiv.org/abs/1404.1100

Face Recognition – Eigenfaces

(Sorry, the code is “narrative” ☺)

```
cale_att='./ATTfaces/';
loc_images=dir(strcat(cale_att,'*.pgm'));

X=[];
i_X=1;
X_test=[];
i_X_t=1;
Nume=zeros(length(loc_images),1);
Num_poza=zeros(length(loc_images),1);
for i=1:length(loc_images)
    numef=strcat(cale_att,loc_images(i).name);
    I=imread(numef);
    [m, n]=size(I);
```

```
 % label processing
 nume=loc_images(i).name;
 ind=strfind(nume,'.');
 nume=nume(2:ind(1)-1);
if(str2double(nume) < 100)
   label=str2double(nume(1));
   nr_poza=str2double(nume(2:end));
elseif str2double(nume(2:3)) == 10 && length(nume) == 3
   label=str2double(nume(1));
   nr_poza=str2double(nume(2:end));
else
   label=str2double(nume(1:2));
   nr_poza=str2double(nume(3:end));
end;
Nume(i)=label;
Num_poza(i)=nr_poza;
```

```
  %linearize images
   vI=double(I(:));
   % build data matrix test/training
   if nr_poza == 1 || nr_poza == 2 || nr_poza == 3
       X_test=[X_test, vI];
       Label_test(i_X_t)=label;
       i_X_t=i_X_t+1;
   else
       X=[X, vI];
       Label(i_X)=label;
       i_X=i_X+1;
   end;
end;
```

```
% compute average image
ave_im=mean(X,2);
% transform data in data with zero mean
X=X-repmat(ave_im,1,size(X,2));
disp(size(X));
[u,s,ve]=svd(X);
%compute the covariance matrix
disp(' Matricea de covarianta ');
tic;
covx=cov(X');
% compute eigenvalues and eigenvectors
disp(' Calcul valori si vectori proprii');
[V,D] = eig(covx);
saveV=V;
eigval = diag(D);
```

```
% sort eigenvalues (rearrange eigenvectors)
eigval = eigval(end:-1:1);
V = fliplr(V);
[eigval1,ind]=sort(-diag(D));
eigval1=-eigval1;
V1=saveV(:,ind);
disp('Elapsed time '); disp(toc);
% https://en.wikipedia.org/wiki/Eigenface
% compute the number of principal components
sum(eigenvalues_pc)=95%*sum(all eigenvalues)
eigsum = sum(eigval);
csum = 0;
for i = 1:length(eigval)
    csum = csum + eigval(i);
    tv = csum/eigsum;
    if tv > 0.95
```

```
        k95 = i;
        break;
    end;
end
disp(k95);



% https://blog.cordiner.net/2010/12/02/eigenfaces-face-recognition-matlab/
num_eigenfaces=10;
image_dims=[112,92];
figure;
for n = 1:num_eigenfaces
    subplot(2, ceil(num_eigenfaces/2), n);
    eig_face = reshape(V(:,n), image_dims);
    imshow(eig_face,[ ]);
```

```
end

% fast eigenfaces computation
tic;
xtx=X'*X/size(X,2);
[Vf,Df]=eig(xtx);
eigval_f=diag(Df);
[eigval_f,ind]=sort(-eigval_f);
eigval_f=-eigval_f;
Vf=Vf(:,ind);
Eigen_faces=X*Vf;
disp('Elapsed time '); disp(toc);
```

```
%https://en.wikipedia.org/wiki/Eigenface
% compute the number of principal components
sum(eigenvalues_pc)=95%*sum(all eigenvalues)

eigsum = sum(eigval_f);
csum = 0;
for i = 1:length(eigval)
   csum = csum + eigval_f(i);
   tv = csum/eigsum;
   if tv > 0.95
      k95_f = i;
      break;
   end;
end
disp(k95_f);
```

```
%figure

num_eigenfaces=10;
image_dims=[112,92];
figure;
for n = 1:num_eigenfaces
   subplot(2, ceil(num_eigenfaces/2), n);
   eig_face = reshape(Eigen_faces(:,n), image_dims);
   imshow(eig_face,[ ]);
end
```

```
%Face recognition with 1NN
k95=100;
Eigenfete=V(1:k95,:);
%Eigenfete=Eigen_faces(:,1:k95_f)';
%Project data on eigenfaces
features_training=Eigenfete*X;
features_test=Eigenfete*(X_test-repmat(ave_im,1,size(X_test,2)));

features_training=features_training';
features_test=features_test';
knn_model=fitcknn(features_training,Label);
predicted_labels=predict(knn_model, features_test);
disp('Recognition Error');
disp(sum(predicted_labels~=Label_test')/length(Label_test)*100);
```

Web search – Google's PageRank

- informaţii nestructurate, 'self-organized'
- fără standarde, recenzenţi, 'paznici' ai conţinutului
- informaţie volatilă, eterogenă şi foarte diversă (conţinut, structură, format, limbi, alfabete, scop)
- 1.872.963.735 – pagini web active
- $\approx 45*10^9$ – pagini web indexate (probabil mai multe) ???

Căutarea

- *roboţi web* – parcurg permanent graful Web pentru informaţii
- *depozitul de pagini* - memorează temporar paginile web (paginile foarte populare sunt stocate perioade mai lungi)
- *modulul de indexare* – se extrage din paginile depozitate informaţia esenţială (cuvinte-cheie, fraze-cheie, descriptori) → se obţine o versiune comprimată a paginii; în funcţie de popularitatea paginii aceasta este fie ştearsă, fie păstrată în depozitul de pagini

- informaţiile esenţiale memorate despre orice pagină Web
    - *index de conţinut* - cuvinte sau fraze cheie, titlu (se construieşte inverted file - similar indexului unei cărţi)
    - *index de structură hyperlink*
    - *index de informaţii speciale* – imagini, pdf

- *modul de interogare* – transformă cererea din limbaj natural într-un limbaj pentru motorul de căutare (de obicei numere); se consultă informaţiile esenţiale memorate; se obţin paginile relevante cererii (care conţin termeni din interogare).
- *modulul de ‘ranking’(evaluare a relevanţei)* – prelucrează paginile relevante furnizate de modulul de interogare şi le ordonează după anumite criterii (funcţie de motorul de căutare); acest modul trebuie să discearnă care pagină Web răspunde cel mai bine interogaţiei şi le ordonează;

PageRank este sistemul de evaluare a paginilor web din punctul de vedere al importanţei, folosit de motorul de căutare Google. Nota furnizată de PageRank este unul din factorii folosiţi pentru a ordona paginile web atunci cand se face o căutare.

"**PageRank** is an important factor. It is **one out of 200** signals but still it is an **important** one for a large number of queries." (Matt Cutts – head of Google's Webspam team)

Google dă cel puţin 2 ‘note’/’scoruri’ fiecărei pagini Web:

a. popularitate
b. conţinut – modul de calcul este secret – termenii din interogare sunt în titlu sau în interiorul paginii, de câte ori termenii apar în pagină, cât de aproape unii de alţii sunt termenii din interogaţiile cu cuvinte multiple, cum apar termenii de interogare (bold, subtitluri…), conţinutul paginilor vecine

Nota/scorul finală cu care se face ordonarea este o medie ponderată între notele de mai sus, componenta principală fiind popularitatea.

PageRank (Google) – S. Brin, L. Page (popularitate)

Sergey Brin, Lawrence Page, R. Motwami, Terry Winograd – '*The PageRank citation ranking: bringing order to the Web*' – Technical Report 1999-0120, Computer Science Department, Stanford University, 1999

Ideea: ***O pagină este importantă dacă este referită (citată) de alte pagini importante.***

Importanţa unei pagini (PageRank-ul ei) este calculată sumând PageRank-urile tuturor paginilor care citează pagina respectivă. Dacă o pagină importantă citează mai multe pagini, atunci importanţa ei se împarte egal paginilor pe care le citează.

Graful Web – cu $\boldsymbol{n}$ noduri.

$$\boldsymbol{I}_{ij} = \begin{cases} \mathbf{1} & \text{dacă pagina } \boldsymbol{j} \text{ citează pagina } \boldsymbol{i} \\ \mathbf{0} & \text{altfel} \end{cases}$$

$\boldsymbol{c_j}$ = numărul de citări ale paginii $\boldsymbol{j}$ (outlink-urile paginii $\boldsymbol{j}$)

$\boldsymbol{r_i}$ = PageRank-ul paginii $\boldsymbol{i}$

$$r_i = (1-d) + d\sum_{j=1}^{n}\frac{I_{ij}}{c_j}r_j \ , \ d > 0 \ , \ d =_{?} 0.85 \qquad (1)$$

Constanta ***d*** asigură faptul că fiecare pagina Web are un Pagerank de cel puţin ***(1-d)***.

Relaţia (1) poate fi scrisă matricial astfel:

$$r = (1-d)e + d\, e^T I\, C^{-1} r \qquad (2)$$

$$\boldsymbol{e} = \begin{pmatrix} \mathbf{1} \\ \mathbf{1} \\ \vdots \\ \mathbf{1} \end{pmatrix} \in \mathbb{R}^{n}, \; \boldsymbol{I} = \left(\boldsymbol{I}_{ij} \right) \in \mathbb{R}^{n \times n}, \; \boldsymbol{C} = \operatorname{diag}\left(\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_n \right) \in \mathbb{R}^{n \times n}$$

Deoarece suma tututror Pagerank-urilor este ***1*** putem scrie:

$$\sum_{i=1}^{n} \boldsymbol{r}_i = \boldsymbol{e}^T \boldsymbol{r} = \mathbf{1}.$$

$$\boldsymbol{A} = \frac{\mathbf{1} - \boldsymbol{d}}{\boldsymbol{n}} \boldsymbol{e}\boldsymbol{e}^T + \boldsymbol{d}\, \boldsymbol{I}\, \boldsymbol{C}^{-1} \in \mathbb{R}^{n \times n}$$

Relaţia (2) se poate scrie:

$$\boldsymbol{r} = \boldsymbol{A}\boldsymbol{r}$$

Această relaţie ne spune că matricea $\boldsymbol{A}$ (matrice de dimensiuni foarte mari şi rară) are valoarea proprie ***1*** iar vectorul de Pagerank-uri este vectorul propriu asociat. Se poate folosi metoda puterii pentru determinarea lui $\boldsymbol{r}$.