Tema 2 - Metode Numerice

Facultatea de Automatică şi Calculatoare
Universitatea Politehnică Bucureşti

May 6, 2018

Cuprins

1 Introducere

In recunoasterea formelor, selectia si extragerea caracteristicilor reprezinta o alegere decisiva pentru proiectarea oricarui clasificator. Selectia caracteristicilor poate fi vazuta si ca un proces de compresie de date, fiind similara cu o transformare liniara din spatiul initial al observatiilor intr-un spatiu cu mai putine dimensiuni. O astfel de transformare este necesara deoarece poate pastra o mare parte din informatii (prin eliminarea informatiilor redundante sau a celor mai putin semnificative) si permite aplicarea unor algoritmi eficienti doar intr-un spatiu de dimensiuni reduse.

Cele mai multe transformari utilizate pentru selectia caracteristicilor sunt cele liniare, in timp ce transformarile neliniare au o complexitate mai ridicata, sunt mai dificil de implementat, dar pot avea o eficienta mai mare asupra rezultatelor, exprimand mai bine dependenta dintre formele observate si caracteristicile selectate ale acestor forme.

1.1 Descompunerea valorilor singulare

Fiind data o matrice $A \in R^{m*n}$, descompunerea valorilor singulare (DVS, in eng. singular value decomposition - SDV) ale matricei A este data de factorizarea $A = USV^T$, unde:

1. $U \in R^{mxm}$ este o matrice ortonormata
2. $S \in R^{mxn}$ este o matrice diagonala
3. $V \in R^{nxn}$ este o matrice ortonormata

Elementele de pe diagonala principala a lui S sunt intotdeauna numere reale nenegative ($s_{ii} \geq 0$ pentru $i = 1 : min(m, n)$) si se numesc *valorile singulare* ale matricei A. Acestea sunt asezate in ordine descrescatoare, astfel incat $s_{11} \geq s_{22} \geq ... \geq s_{rr} > s_{r+1r+1} = ... = s_{pp} = 0$, unde $p = min(m, n)$.

Coloanele $u_j \in R^m$, $j = 1 : m$ ale lui U se numesc *vectori singulari stanga* ai matricei A. Coloanele $v_j \in R^n$, $j = 1 : n$ ale lui V se numesc *vectori singulari dreapta* ai matricei A.

De exemplu, pentru matricea:

$$A = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix}$$

se obtine urmatoarea descompunere a valorilor singulare:

$$A = USV^T = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{bmatrix}$$

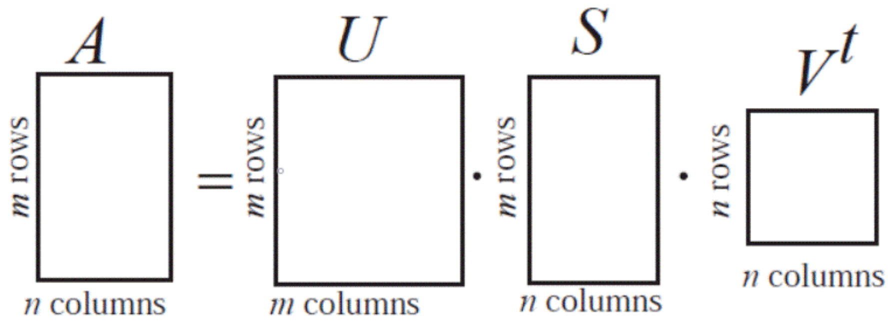

Figura 1: Descompunerea valorilor singulare pentru matricea A de dimensiune m*n, unde $m > n$.

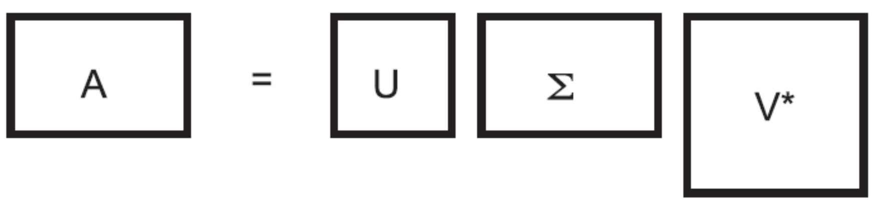

Figura 2: Descompunerea valorilor singulare pentru matricea A de dimensiune m*n, unde $n > m$, $S = \Sigma$, $V^t = V^*$.

2 Compresia imaginilor folosind DVS

Descompunerea redusa a valorilor singulare presupune descompunerea (factorizarea) matricei A astfel: $A \approx A_k = U_k S_k V_k^T$, unde $A_k \in R^{mxn}$, $U_k \in R^{mxk}$, $S_k \in R^{kxk}$, $V_k^T \in R^{kxn}$.

Intuitiv, descompunerea redusa a valorilor singulare semnifica eliminarea valorilor singulare nule sau a valorilor singulare nule si a celor de o valoare mica din matricea S (reprezentand informatia mai putin semnificativa). Acest lucru presupune si eliminarea coloanelor si a liniilor corespunzatoare acestor valori singulare din matricele U, respectiv din V (vezi Figura 3).

In cele ce urmeaza, presupunem ca matricea A reprezinta modelarea matematica pentru o imagine alb-negru clara si matricea A_k este modelarea matematica pentru o imagine alb-negru aproximativa a imaginii clare. Ambele imagini au dimensiune m*n pixeli. Fiecare element (i, j) din matricele A si A_k corespunde intensitatii de gri a pixelului (i, j) din imagine. Prin urmare, elementele matricelor A si A_k au valori cuprinse intre 0 (corespunzatoare culorii negre) si 255 (corespunzatoare culorii albe).

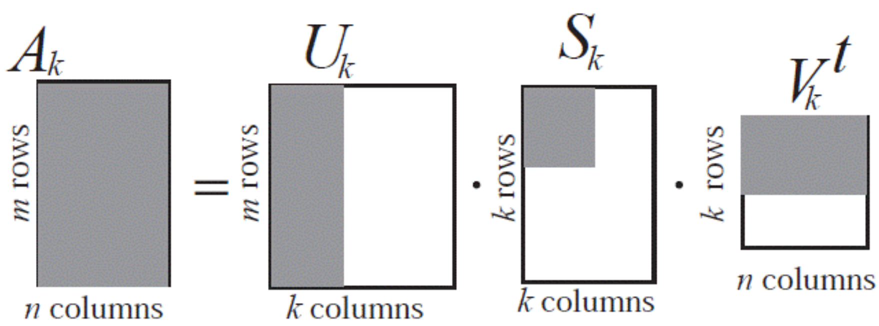

Figura 3: Exemplu de descompunere redusa a valorilor singulare pentru matricea A m*n dimensionala, m>n. Aceasta descompunere presupune eliminarea portiunilor hasurate in alb din matricele U, S, respectiv V^t. Portiunile hasurate in gri (notate U_k, S_k, respectiv V_k^t) din matricele U, S, respectiv V se vor pastra. Astfel, matricea A_k va aproxima matricea initiala A.

2.1 Cerinta 1 [15p]

In cadrul acestei cerinte, va trebui sa scrieti o functie Octave pentru compresia unei imagini folosind descompunerea redusa a valorilor singulare. Semnatura functiei este: $function\ A\_k = cerinta1\ (image,\ k)$, unde $image$ reprezinta calea catre imagine si k numarul de valori singulare. Functia trebuie sa intoarca matricea A_k avand semnificatia de mai sus.

2.2 Cerinta 2 [20p]

Scrieti o functie pentru a obtine urmatoarele 4 grafice pentru o imagine:

- folosind descompunerea valorilor singulare:

1. reprezentati grafic toate *valorile singulare* ale matricei A in ordine descrescatoare.

   - folosind descompunerea redusa a valorilor singulare (cerinta 1), pentru diferite valori ale lui k (de exemplu, k poate fi [1:19 20:20:99 100:30:min(m,n)]):

2. reprezentati grafic k (pe axa OX) si *informatia* data de primele k valori singulare (pe axa OY) calculata dupa formula:

$$\frac{\sum_{i=1}^{k} s_{ii}}{\sum_{i=1}^{min(m,n)} s_{ii}}$$

3. reprezentati grafic k (pe axa OX) si *eroarea aproximarii* pentru matricea A (pe axa OY) calculata dupa formula:

$$\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(A(i,j) - A_k(i,j))^2}{m * n}$$

4. reprezentati grafic k (pe axa OX) si *rata de compresie* a datelor (pe axa OY) calculata dupa formula:

$$\frac{m * k + n * k + k}{m * n}$$

   Justificare formula: Pentru obtinerea imaginii aproximative avem nevoie sa memoram doar matricele U_k, V_k^T si primele k elemente de pe diagonala principala a matricei S_k. In total, m*k+n*k+k elemente trebuie memorate. Astfel, stocarea acestora ocupa memorie mai putina comparativ cu memoria ocupata de matricea A pentru m*n elemente. Tinant cont de faptul ca cea mai mare parte din informatia continuta in matrice este data de primele valori singulare, compresiea datelor folosind descompunerea redusa a valorilor singulare permite o economisire de memorie.

Semnatura functiei este $function\ cerinta2()$.

3 Compresia imaginilor folosind analiza componentelor principale

Scopul analizei componentelor principale (in eng. principal component analysis - PCA), este de a transforma date de tipul $A = [a_1, a_2, ..., a_n]$, dintr-un spatiu dimensional R^m intr-un spatiu dimensional R^k, unde $a_i \in R^m$ si k<m. Acest spatiu este dat de cele k componente principale (PC). Componentele principale sunt ortonormale, necorelate si reprezinta directia variatiei maxime. Prima componenta principala reprezinta directia variatiei maxime a datelor, urmand ca urmatoarele componente principale sa aduca variatii din ce in ce mai mici.

3.1 Cerinta 3 [25p]

Urmatorul algoritm calculeaza componentele principale folosind metoda DVS: Fiind data o matrice $A = [a_1, a_2, ..., a_n] \in R^{m*n}$, unde $a_j \in R^{m*1}, j = 1 : n$:

1. Se calculeaza urmatoarea medie pentru fiecare vector $a_i \in R^{1*n}, i = 1 : m$: $\mu_i = \frac{\sum_{j=1}^{n} a_i(j)}{n}$.
2. Se actualizeaza vectorii $a_i \in R^{1*n}, i = 1 : m$ astfel: $a_i = a_i - \mu_i$.
3. Se construieste matricea $Z = \frac{A^T}{\sqrt{n-1}}, Z \in R^{n*m}$.
4. Se calculeaza DVS pentru matricea Z: $Z = USV^T$.

5. Spatiul k-dimensional al componentelor principale (notat cu W) este dat de primele k coloane din matricea $V = [v_1, v_2, ..., v_m] \in R^{m*m}$: $W = [v_1, v_2, ..., v_k]$ (v_1 este prima PC, v_2 este a doua PC s.a.m.d).

6. Se calculeaza proiectia lui A in spatiul componentelor principale, adica matricea $Y = W^T A$.

7. Se aproximeaza matricea initiala astfel: $A_k = WY + \mu$, unde $\mu \in R^{m*1}$ iar elementele μ_i ale vectorului μ au fost calculate la pasul 1.

Functia Octave care implementeaza acesta cerinta este: function $[A_k\ S] =$ cerinta3(image, k), unde *image* reprezinta calea catre imagine si k numarul de componente principale. Functia intoarce matricele A_k si S cu semnificatia prezentata in acest algoritm.

3.2 Cerinta 4 [15p]

Componentele principale se pot calcula folosind si un algoritm bazat pe matricea de covarianta. Pasii pentru acest algoritm sunt:

Fiind data o matrice $A = [a_1, a_2, ..., a_n] \in R^{m*n}$, unde $a_j \in R^{m*1}, j = 1 : n$:

- Pasii 1-2 sunt aceeasi ca la cerinta 3.
- Se construieste matricea de covarianta $Z = \frac{A*A^T}{n-1}, Z \in R^{m*m}$.
- Se aplica functia *eig* asupra matricei Z: $[V\ S] = eig(Z)$.
- Spatiul k-dimensional al componentelor principale (notat cu W) este dat de primele k coloane din matricea $V = [v_1, v_2, ..., v_m] \in R^{m*m}$: $W = [v_1, v_2, ..., v_k]$.
- Pasii 6-7 sunt aceeasi ca la cerinta 3.

Functia Octave care implementeaza acesta cerinta este: function $[A_k\ S]=$ cerinta4(image, k), unde *image* reprezinta calea catre imagine si k numarul de componente principale. Functia intoarce matricele A_k si S cu semnificatia prezentata in acest algoritm.

3.3 Cerinta 5 [25p]

Folosind *cerinta* 3, scrieti o functie pentru a obtine urmatoarele 4 grafice pentru o imagine:

1. reprezentati grafic vectorul $diag(S)$.

   Pentru diferite valori ale lui k (de exemplu, k poate fi [1:19 20:20:99 100:30:min(m,n)]):

2. reprezentati grafic k (pe axa OX) si *informatia* data de primele k valori singulare (pe axa OY) calculata dupa formula:

$$\frac{\sum_{i=1}^{k} s_{ii}}{\sum_{i=1}^{min(m,n)} s_{ii}}$$

3. reprezentati grafic k (pe axa OX) si *eroarea aproximarii* pentru matricea A (pe axa OY) calculata dupa formula:

$$\frac{\sum_{i=1}^{m} \sum_{j=1}^{n} (A(i,j) - A_k(i,j))^2}{m * n}$$

4. reprezentati grafic k (pe axa OX) si *rata de compresie* a datelor (pe axa OY) calculata dupa formula:

$$\frac{2 * k + 1}{n}$$

   Justificare formula: Matricea A are dimensiunea m*n. Pentru reconstructia matricei A este nevoie de W, Y si μ avand fiecare m liniii si k, k coloane, respectiv 1 coloana. Prin urmare, numarul de coloane m-dimensionale de stocat a fost redus de la n la 2*k+1.

Semnatura functiei este $function\ cerinta5()$.

4 Observatii

1. Imaginile de testare sunt doar in format alb-negru. Pentru a citi o imagine in program folositi functiile *imread* si *double* din Octave.

2. Pentru vizualizarea imaginilor pe care le obtineti folositi functiile *imshow* si *uint*8 din Octave. Aceasta cerinta nu este obligatorie (checker-ul verifica doar datele returnate de functiile obligatorii), dar va ajuta sa observati diferentele pe imaginile modificate.

3. Puteti sa definiti functii auxiliare in cazul in care aveti nevoie de acestea.

4. In rezolvarea temei, aveti voie sa folositi functiile din Octave (inclusiv functiie *svd* si *eig*) cu o singura restrictie: NU folositi functia *princomp* din Octave.

5. Fisierul *Readme* (cu extensia *.pdf*) va contine doua parti: graficele si interpretarea rezultatelor obtinute la cerintele 2 si 5. Cerintele 2 si 5 le testati pentru oricare 2 imagini aflate in directorul *in* din checker. Pentru a desena un grafic continand mai multe subgrafice se foloseste functia *subplot*. Tot in Readme puteti pune si poze cu imaginile obtinute la cerintele 2 si 5 daca considerati ca va ajuta la interpretarea rezultatelor.

6. Checker-ul face testarea automata doar a cerintelor 1, 3 si 4, cerintele 2 si 5 le vom corecta manual.
7. Arhiva *.zip* trebuie sa includa DOAR functiile pe care voi le scrieti, nu includeti in arhiva altceva.

5 Bibliografie

1. Richard L.Burden, J. Douglas Faires, *Numerical Analysis*, Editia 9, Subcapitolul 9.6
2. http://www.cs.utexas.edu/users/inderjit/public_papers/HLA_SVD.pdf
3. https://en.wikipedia.org/wiki/Principal_component_analysis