Modellering & Simulatie

Afraz Salim r0439731

Contents

1 Opdracht 2

Hoeveel geheugenruimte is nodig om de volle beoordelingenmatrix full(R) voor te stellen?

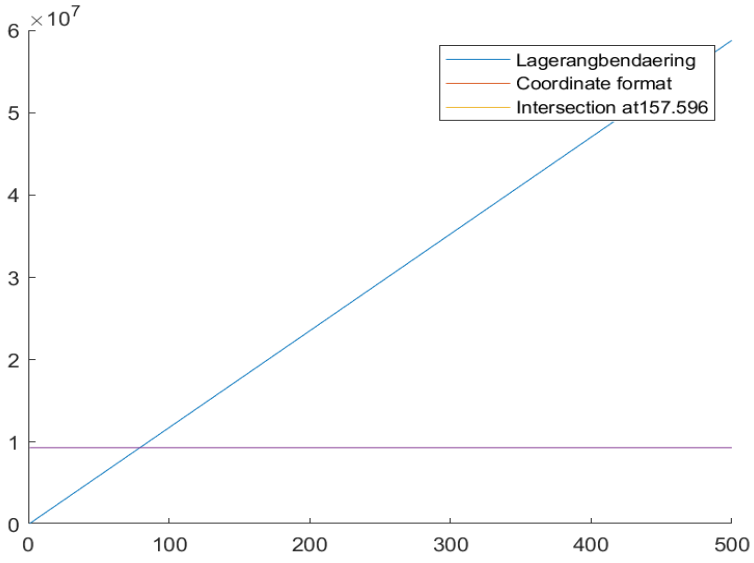
R is een matrix met m rijen en n kolomen en elke beoordeling wordt voorgesteld door 8 bytes. Matrix R neemt $8 \times (m \times n)$ bytes geheugen in beslag. Het is al gegeven dat R 2206 rijen en 12488 kolommen heeft dus R neemt $8 \times (2206 \times 12488)$ bytes in beslag.

Hoeveel geheugenruimte is nodig om de ijle beoordelingenmatrix R voor te stellen in het coordinaatformaat?

Een ijle matrix kan in coordinaatformaat voorgesteld worden door een triplet (i, j, v) waarbij i en j rij en kolom voorstellen en v stelt de beoordeling voor. Rijen en kolommen kunnen voorgesteld worden met geheel getallen terwijl een beoordeling wordt voorgesteld door een dubbele-precisie vlottendekommagetal. Het aantal niet-nul elementen stellen wij voor met nnz. Wij hebben dan $(4 \times nnz) + (4 \times nnz) + (8 \times nnz)$ bytes geheugen nodig. Matlab maakt gebruik van CSC(Compressed Sparse Column) formaat om een ijle matrix voor te stellen.

Hoeveel geheugenruimte is vereist om de rang-r lagerangbenadering FWT uit verg.(2)compact voor te stellen (als functie van r)?

Vergelijking 2 : $R \approx FW^T$.
F is een matrix met m rijen en r kolommen. W is een matrix met n rijen en r kolommen$(F^{m \times r}, W^{n \times r})$.
Wij hebben $(8 \times m \times r) + (8 \times n \times r) \rightarrow (8 \times r) \times (m + n)$ bytes geheugen nodig.



Code voor deze figure kan gevonden worden in 1.

De snijpunt kan bepaald worden door de vergelijking: $(16 \times nnz) = 8 \times r \times (m+n)$ op te lossen. De oplossing van deze vergelijking is:$r = \frac{2 \times nnz}{(m+n)}$.

2 Opdracht 3

Zij $A \in R^{m \times n}$ een matrix van rang $r \leq min\{m,n\}$ met singuliere waardenontbinding $A = \sum_{i=1}^{r} \gamma_i g_i h_i^T$. Je mag veronderstellen dat de singuliere waardenontbinding van A uniek is. Onderstel dat algoritme 1 op A wordt toegepast. Bewijs dat $A - E_k = \sum_{j=1}^{k} \gamma_j g_j h_j^T$ de rang-k met $(k \leq r)$ singulierwaardeontbinding van A is.

De rang van A is gelijk aan r dus de vergelijking:
$A - E_k = \sum_{j=1}^{k} \gamma_j g_j h_j^T$ kunnen wij herschrijven als
$\sum_{i=1}^{r} \gamma_i g_i h_i^T - E_k = \sum_{j=1}^{k} \gamma_j g_j h_j^T$
$\sum_{i=1}^{r} \gamma_i g_i h_i = \sum_{j=1}^{k} \gamma_j g_j h_j^T + E_k$.
Wij moeten nu aantonen dat $E_k = \sum_{i=k+1}^{r} \gamma_i g_i h_i^T$.
Op lijn 5 zien wij dat $E_j \leftarrow E_{j-1} - \sigma_j u_j v_j^T$. Hieuit kunnen wij de waarde van E_k uithalen.
$E_j \leftarrow E_{j-1} - \sigma_j u_j v_j^T \rightarrow \sum_{i=j}^{r} \sigma_i u_i v_i^T - \sigma_j u_j v_j^T \rightarrow \sum_{i=j+1}^{r} \sigma_i u_i v_i^T$ dus
$E_k = \sum_{i=k+1}^{r} \sigma_i u_i v_i^T$. Als wij nu σ_i vervangen door γ_i,u_i door g_i en v_i door h_i dan krijgen wij
$E_k = \sum_{i=k+1}^{r} \gamma_i g_i h_i^T$.

Na hoeveel stappen stopt het algoritme?

De algoritme controleert de 2-de norm van de matrix in elke iteratie en 2-de norm van de matrix is gelijk aan de grootste singulier waarde. Als de matrix A rang-r heeft en als k is gelijk aan r dan zal het algoritm stoppen na r stappen.

3 Opdracht 4

Zij c het laatste cijfer uit je studentennummer. Wat is c? Stel dat we de stappen $4-6$ van algoritme 1 slechts $(c+1)$ maal zouden uitvoeren. Kan men dit algoritme dan toepassen op een 280000×58000 matrix met 27 miljoen niet-nulwaarden (i.e., de volledige MovieLens databank) op een desktop met 8GB werkgeheugen en $2+(c+1)^2$ (in GB) swap geheugen? Verklaar waarom wel of niet?

C is 1 dus wij hebben swapgeheugen: 6 GB.
Wij stellen dubbele-precisie vlottendekommagetallen voor met double. In stap 4 van de algoritme zoeken wij een beste rang-1 benadering. Wij stellen vergelijkingen op voor alle stappen:
Stap 4: $(m+n+1) \times double$
Stap 5: $3 \times (m \times n) \times double$. Wij hebben 3 matrices in stap 5 dus wij hebben deze vergelijking

vermeningvuldigd met 3.
Stap 6: 1×4 bytes.
M en N hebben de waarden $280,000$ en 58000 respectievelijk.
Het geheugen nodig voor stap 4,5 en 6 is de sum van de vergelijkingen van stap 4 + stap 5 + stap 6. Wij berekenen deze sum met de behulp van een matlab script. De script kan gevonden worden in 10.

```
Total available memory: 14 GB.
Memory needed for full matrix: 389.76 GB.
```

Voor sparse matrix:

```
Total available memory: 14 GB.
Needed for sparse matrix: 1.30 GB.
```

Wij kunnen stappen $4-6$ enkel uitvoeren als de matrix geen volle matrix is.

4 Opdracht 5

Schrijf een functie met de hoofding function [X] = SN_sparseModel(Uk,sk,Vk,A) die als uitvoer X de matrix $X = P_\Omega(U_k * diag(S_k) * V_k T)$ geeft.De geheugencomplexiteit van je algoritme in functie van m, n, r en nnz mag hoogstens $O(c)$ bedragen. Hoe heb je deze doelstelling bereikt?

Wij halen eerst de coordinaten van de matrix A waar de elementen niet-nul zijn. Dit wordt gedaan met de behulp van de **find** functie

```
[row,col,val] = find(A).
```

Aangezien wij de waarden van *val* vector niet nodig hebben,de nieuwe waarden worden in *val* vector opgeslagen. Als een element in A op positie (i, j) niet nul is, dan de nieuwe waarde voor deze positie wordt als volgt berekend:

- Wij halen de rij i van matrix U en deze wordt element wise((.∗) in matlab) vermenigvuldigd met de matrix Sk. Wij krijgen dus een nieuwe vector.
- Deze nieuwe vector wordt dan vermenigvuldigd met j-de kolom van de matrix V^T. Als een gevolg van deze vermenigvuldiging , krijgen wij een waarde die in vector *val* op positie van de index van de rij opgeslagen.

De code voor deze opgave kan gevonden worden in 2. Er wordt niet meer geheugen ruimte gebruikt dan het aantal niet-nul elementen in A.

5 Opdracht 7

Schrijf een functie met de hoofding $function[s] = SN\_optimalCoefficients(Uk, Vk, A)$ die als uitvoer $s_k \in \mathbb{R}^k$ optimale oplossing van het kleinste-kwaadraten probleem in verg (7) berekent.

Wij hebben gebruik gemaakt van **lsqr** functie van matlab. Code voor deze opgave kan gevonden worden in 3.

De geheugencomplexiteit van je algoritme in functie van m, n, k en nnz mag hoogstens $O(nnz \times k)$ bedragen je mag hierbij wel veronderstellen dat max$\{m, n\} \leq nnz$. Hoe heb je deze doelstelling bereik?

Voor de functie **SN_sparseModel** geven wij kolomvectoren van de matrix U en V als invoer. Het resultaat van functie **SN_sparseModel** wordt opgeslagen in de sparse matrix B. Matrix B wordt gecreëerd voor $k * nnz$ elementen. In matlab kan dit gedaan worden met

```
B  = spalloc(rows,colU,nnz(A)*k);
```

.

6 Opdracht 8

Schrijf een functie met de hoofding function [mu] = SN_userMeans(A) die als uitvoer een kolomvector van lengte n geeft waarvan het i-de element μ_i gelijk is aan het gemiddelde van alle gekende beoordelingen van gebruiker i.

De kolom vector van een matrix A stelt een gebruiker voor en rijen stellen de movies voor. Wij maken gebruik van **mean** functie en als invoer geven wiij een vector zonder nul-waarden. Code voor deze functie kan gevonden worden in 4.

7 Opdrach 9

Hoeveel bedragen de 3 laagste gemiddelde beoordelingen? Hoeveel gebruikers gaven exact 5 als gemiddelde score

```
Drie laagste waarden zijn
    0.5100

    0.5727

    0.8983

Gebruikers met score 5
    13
```

Code voor deze opdracht kan gevonden worden in 5.

8 Opdracht 10

Schrijf een functie met de hoofding $function[err] = SN\_RMSE(A, B)$ die bij invoer van twee ijle matrices $A, B \in R^{m \times n}$ met eenzelfde ijlheidspatroon Ω als uitvoer de root mean square error als uitvoer err geeft.

Code voor deze opdracht kan gevonden worden in 6.

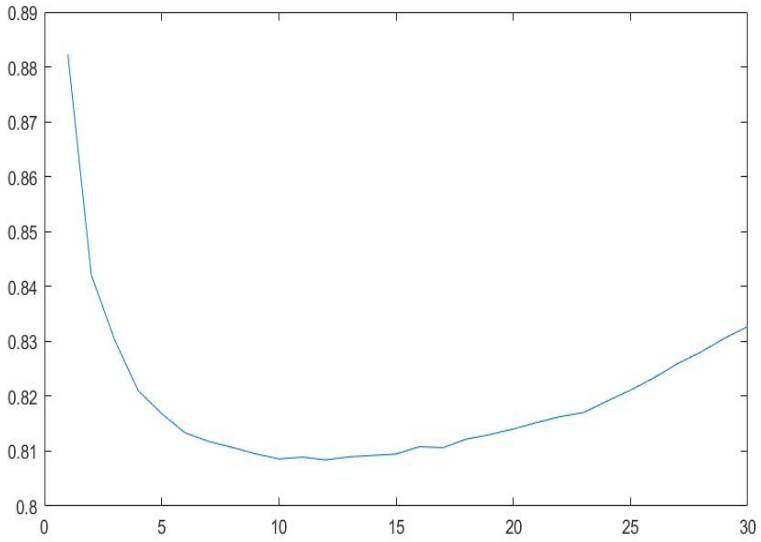
9 Opdracht 11

Zij T de matrix die je in opdracht 1 hebt ingeladen, Ω diens ijlheidspatroon, μ de vector van de gemiddelde gekende beoordelingen van de gebruikers zoals bepaald door de functie SN_userMeans toegepast op de trainingdata R die je in opdracht 1 hebt ingeladen, en zij $1 \in \mathbb{R}^{2206}$ de vector waarin elk element 1 is. Hoeveel bedraagt de RMSE tussen T en de ijle matrix $V = P_\Omega(1\mu^T)$

RMSE tussen T en V is **0.8823**. Code voor deze opgave kan gevonden worden in 7.

10 Opdracht 12

Maak een duidelijke figuur van de evolutie van de RMSE van algoritme 2, zoals deze gegeven wordt door het tweede uitvoerargument rmse van de functie $SN\_rank1MatrixPursuit$. Neem de figuur op in het verslag.



11 Opdracht 13

Bereken $[U20, s20, V20, \ ] = SN\_rank1MatrixPursuit(R, 20, T)$ en duid de factoren van de resulterende lagerangbenadering in de rest van het verslag aan met $U20$, $s20$ en $V20$. Wat zijn de waarden van de vector $s20$, afgerond tot 1 cijfer na de komma?

De waarden van S_{20} zijn :

```
0.9693
806.2371
386.3166
383.1735
284.9237
248.8875
192.6608
199.1986
179.9039
```

```
171.2653
133.1371
152.0562
135.9722
143.7770
121.9643
120.4505
128.7878
116.9602
120.2022
124.7014
```

12 Opdracht 14

Schrijf een functie met de hoofding
function [movieIDs, score] = SN_actualBestMovies(R)

Code voor deze opdracht kan gevonden worden in 8.

13 Opdracht 15

Schrijf een functie met de hoofding
function [movieIDs, score] = SN_predictedBestMovies(Uk,sk,Vk)

.
Code voor deze opdracht kan gevonden worden in 9.
Wij creëren een vector van lengte n om scores te bewaren dus het geheugen complexiteit is maximaal n.

14 Opdracht 16

Bereken met behulp van de functies uit de twee voorgaande opgaves de 25 films met de hoogste gemiddelde beoordelingen (op basis van de gekende beoordelingenmatrix R) en de 25 films met de hoogste gemiddelde voorspelde beoordelingen.

De actuele movies zijn:

```
      ActualBestMovies...
Name                                  Rating        Reviews
Planet Earth II (2016)                 4.4806        387
```

```
Black Mirror: White Christmas (2014)   4.3258        594
Won’t You Be My Neighbor? (2018)       4.2982        57
Sherlock - A Study in Pink (2010)      4.2595        131
Blue Planet II (2017)                  4.2023        131
Over the Garden Wall (2013)            4.1995        218
Whiplash (2013)                        4.1975        357
Human Planet (2011)                    4.1723        177
Inception (2010)                       4.1535        7631
The Night Of (2016)                    4.1467        184
The Jinx: The Life and Deaths of Robert Durst (2015) 4.1421    401
Making a Murderer (2015)               4.1195        113
Piper (2016)                           4.1173        631
Frozen Planet (2011)                   4.1162        198
Whiplash (2014)                        4.1153        3535
Bo Burnham: what. (2013)               4.0789        95
The Handmaiden (2016)                  4.0781        544
Your Name. (2016)                      4.0744        605
Story of Film: An Odyssey, The (2011)  4.0678        59
Laurence Anyways (2012)                4.0649        77
Intouchables (2011)                    4.0617        3331
Spotlight (2015)                       4.0479        2264
Winter on Fire: Ukraine’s Fight for Freedom (2015) 4.0463   54
John Mulaney: New In Town (2012)       4.0419        155
O.J.: Made in America (2016)           4.0399        213
```

Predicted list:

```
        PredictedBestMovies...
Name                               Rating  Reviews
Inception (2010)                   4.1715   7631
Whiplash (2014)                    4.0051   3535
Intouchables (2011)                3.9821   3331
Interstellar (2014)                3.9533   6181
Django Unchained (2012)            3.9409   5851
The Martian (2015)                 3.9308   5268
Arrival (2016)                     3.9219   3508
Gone Girl (2014)                   3.9158   4337
Shutter Island (2010)              3.9053   5505
Grand Budapest Hotel, The (2014)   3.8795   4444
Spotlight (2015)                   3.8762   2264
```

Dark Knight Rises, The (2012)	3.8685	6182
King's Speech, The (2010)	3.859	4225
Ex Machina (2015)	3.859	4412
The Imitation Game (2014)	3.8552	4614
Big Short, The (2015)	3.8377	2886
Edge of Tomorrow (2014)	3.8373	4807
Inside Out (2015)	3.8309	4549
Prisoners (2013)	3.8235	2360
Guardians of the Galaxy (2014)	3.8211	5700
Her (2013)	3.8143	3856
Nightcrawler (2014)	3.8132	3139
Room (2015)	3.8071	1674
Wolf of Wall Street, The (2013)	3.805	5141
Dallas Buyers Club (2013)	3.8046	2747

Wij hebben weinig beoordelingen in de eerste lijst maar de ratings zijn zeer hoog terwijl in tweede lijst zijn er veel beoordelingen dus er is meer zekerheid dat deze movies beste movies zijn. Tweede lijst van de films lijkt meer realistischer dan de eerste lijst .

15 Opdracht 17

Code voor deze opgave kan gevonden worden in 11.

16 Opdracht 18

Gebruik het rang-20 model $R20 = U20 * diag(s20) * VT20$ voor deze opdracht. Bereken de 10 beste films voor de gebruiker met volgnummer 98. Lijst de filmtitels op in het verslag samen met de voorspelde score voor gebruiker 98. Vergelijk deze lijst met de films die gebruiker 98 een score van minstens 4.5 gaf, volgens de testdata T. Neem ook deze lijst met filmtitels en bijhorende scores op in het verslag. Herhaal deze opdracht ook voor gebruiker 10100. Wat vind je van de aanbevelingen?

Best films voor gebruiker 98 op basis van Test data T.

'How to Train Your Dragon (2010)'	5.00
'The Hunger Games (2012)'	5.00
'Dark Knight Rises, The (2012)'	5.00
'Pitch Perfect (2012)'	5.00
'Hobbit: An Unexpected Journey, The (2012)'	5.00
'Way, Way Back, The (2013)'	5.00

'Short Term 12 (2013)'	5.00
'Thor: The Dark World (2013)'	5.00
'Wolf of Wall Street, The (2013)'	5.00
'How to Train Your Dragon 2 (2014)'	5.00

Beste voorgespelde films voor gebruiker 98 en bijbehorende scores.

Deadpool (2016)	5.7072
The Hunger Games (2012)	4.9252
Star Wars: Episode VII - The Force Awakens (2015)	4.8272
Her (2013)	4.6641
The Hunger Games: Mockingjay - Part 1 (2014)	4.5391
Black Swan (2010)	4.4612
Perks of Being a Wallflower, The (2012)	4.1906
Scott Pilgrim vs. the World (2010)	4.1505
Avengers: Age of Ultron (2015)	4.1399
Moonrise Kingdom (2012)	4.0868

Best films voor gebruiker 10100 op basis van Test data T.

'King's Speech, The (2010)'	5.00
'Rise of the Planet of the Apes (2011)'	5.00
'Muppets, The (2011)'	4.50
'Dark Knight Rises, The (2012)'	5.00
'Mission: Impossible-Ghost Protocol (2011)'	5.00
'Hobbit: An Unexpected Journey, The (2012)'	5.00
'Misérables, Les (2012)'	5.00
'Star Trek Into Darkness (2013)'	5.00
'World's End, The (2013)'	5.00
'Thor: The Dark World (2013)'	4.50

Beste voorgespelde films voor gebruiker 10100 en bijbehorende scores.

Captain America: The Winter Soldier (2014)	5.4372
Deadpool (2016)	5.4321
Intouchables (2011)	5.3357
Thor: Ragnarok (2017)	5.3027
Avengers: Infinity War - Part I (2018)	5.2872
Doctor Strange (2016)	5.2846
Big Hero 6 (2014)	5.2215

Logan (2017)	5.1842
John Wick (2014)	5.0459
Spotlight (2015)	4.9656

Voor gebruiker 98 lijkt de voorgespelde lijst realistisch. De meeste films in beide lijsten zijn science-fiction films of computer animated films.

Hoeveel tijd heb je gespendeerd aan het oplossen van de opdrachten? Hoeveel tijd heb je gespendeerd aan het schrijven van het verslag?

Ik heb ongeveer 40 uren in totaal gespendeerd aan deze practicium waarvan 3 of 4 uren om het verslag te schriven.(Het was moeilijk voor mij omdat ik geen goed Nederlands spreek).

In de loop van deze opgave hebben we allerhande veronderstellingen gemaakt om ons nieuw aanbevelingssysteem op te stellen. Wat zijn je bedenkingen hierbij? Vind je de resultaten realistisch? Zou je het ontwikkelde aanbevelingssysteem durven toevoegen aan de lijst van aanbevelingssystemen van MovieLens?

Movies in voorgespelde lijst en de lijst uit de test data zijn verschillend. Om ons algoritme aan MovieLens toe te voegen, moeten we eerst zien hoe goed hun algoritme is.

Welke bedenkingen heb je bij dit practicum? Was de opgave (veel) te gemakkelijk, (veel) te moeilijk of van een gepaste moeilijkheidsgraad? Wat zou je zelf anders aangepakt hebben?

De opgave was niet zeer moeilijk maar het had veel tijd genomen om eerst alle dingen goed te begrijpen.

Appendix : Code

Listing 1: Code voor opdracht 2

```
function r0439731_plots(m,n,nnz,size_per_number)
    r =linspace(1,500,500);
    variable_plot = size_per_number*r*(m+n);
    fixed_plot = repmat((4*m)+(4*n)+(size_per_number*nnz),500);
    hold on;
    plot(r, variable_plot);
    plot(r,fixed_plot);
    intersection  = (2*nnz)/(m+n);
    legend("Lagerangbendaering","Coordinate format", "Intersection at"+ intersection);
    hold off;
end
```

Listing 2: Code voor opdracht 5

```
function [X] = r0439731_sparseModel(Uk, Sk, Vk, A)
    if size(Uk,2) ~= size(Vk,2) || size(Sk,1) ~= size(Vk,2) %% Because for V, we take the transpose.
         throw(MException("Size of the matrices is not equal"));
    end
    [row, col, val] = find(A);
    for index = 1:length(row)
          val(index) = Uk(row(index), :) .* Sk' * Vk(col(index), :)';
          %%For diagonal matrix, we can use elements-wise operations.
    end
    X = sparse(row, col, val);
end
```

Listing 3: Code voor opdracht 7

```
function [s] = r0439731_optimalCoefficients(Uk,Vk,A)
 [rowU,colU] = size(Uk);
 [rowV,unUsed] = size(Vk);
 non_zero_elem = nnz(A);
 rows = rowU*rowV;
 non_zero = non_zero_elem*colU;
 B  = spalloc(rows,colU,non_zero);
 for i = 1:colU
     %%Separate the columns of U and V. Let Sk be 1.
     result = r0439731_sparseModel(Uk(:, i), 1, Vk(:, i), A);
     B(:,i) = result(:);
 end
 s = lsqr(B, A(:));
end
```

Listing 4: Code voor opdracht 8

```
function [mu] = r0439731_userMeans(A)
cols = size(A,2);
mu = zeros(cols,1);
for k = 1:cols
   mu(k) = mean(nonzeros(A(:,k)));
end
end
```

Listing 5: Code voor opdracht 9

```
function number_of_users
 load MovieLens_Subset.mat
 mean_values = r0439731_userMeans(R);
 [vals,~] = sort(mean_values);
 B = length(nonzeros((vals == 5)));
 disp("The three lowest values are ");
 disp(vals(1));
 disp(vals(2));
 disp(vals(3));
 disp("User who rated with 5 stars");
 disp(B);
end
```

Listing 6: Code voor opdracht 10

```
function [err] = r0439731_RMSE(A,B)
 D = find(A);
 N = length(D);
 Div = 1 / sqrt(N);
 err = Div* norm(A - B, 'fro');
end
```

Listing 7: Code voor opdracht 11

```
function [err] =  r0439731_root
   load MovieLens_Subset.mat
   avg = r0439731_userMeans(R);
   [row,col] = find(T);
   result = sparse(row,col,avg(col));
   err = r0439731_RMSE(T,result);
end
```

Listing 8: Code voor opdracht 14

```
function [movieIDs, score] = r0439731_actualBestMovies(R)
[score, movieIDs] = sort(r0439731_userMeans(R'), 'descend');
end
```

Listing 9: Code voor opdracht 14

```
function [movieIDs, score] = r0439731_predictedBestMovies(Uk,sk,Vk)
     row_Uk = size(Uk, 1);
     row_Vk = size(Vk, 1);
     score = zeros(row_Vk,1);
   for x = 1:row_Vk
          total = 0;
     for r = 1:row_Uk
          total = total + (Uk(r, :) * (sk .* transpose(Vk(x, :))));
     end
     score(x) = total / row_Uk;
   end
   [score, movieIDs] = sort(score, 'descend');
end
```

Listing 10: Code voor opdracht 4

```
first = 4;
double = 8;
m = 280000;
n = 58000;
nnz_values = 27000000;
totalMemory = (8+6) * 10^9;
one = (m+n+1)*double;
two = 3*m*n*double;
three = 1 * first;
totalMemoryUsage = one + two + three;
fprintf('Total available memory: %i GB', totalMemory / 10^9);
disp("");
fprintf('Memory needed for full matrix: %.2f GB', totalMemoryUsage / 10^9);
disp("");

one = (m+n+1)*double;
two = 3*(2*nnz_values*int + nnz_values*double);
three = 1 * int;
totalMemoryUsage = one + two + three;
fprintf('For sparse matrix: %.2fGB ', totalMemoryUsage / 10^9)
```

Listing 11: Code voor opdracht 17

```
function [movieIDs, score] = r0439731_predictedBestMoviesForUser(R,Uk,sk,Vk,j)
        [row,~] = size(R);
        n = size(Vk, 1);
        count = 0;
        for i = 1:row
           if(R(i,j) == 0)
               count = count+1;
           end
        end
        disp(count);
        score = zeros(count, 1);
        for i = 1:row
           if(R(i,j) == 0)
               movie = Uk(i,:);
               total = 0;
               total = total + (movie*(sk .* Vk(j,:)'));
               score(i) = total;
           else
               score(i) = -1; %Fill the rest with -1. It will be easy to extract the movies.
           end

        end
       [score, IDs] = sort(score, 'descend');
       disp("Displaying movies");
       for i = 1:10
           disp(score(i))
       end
       score = score(score~=-1);
       rows = size(score,1);
       movieIDs = zeros(rows,1);
       for i = 1 :rows
           movieIDs(i) = IDs(i);
       end
end
```