

# Nenegativna faktORIZACIJA matrica u modeliranju tema i klasteriranju dokumenata

Martina Alilović, Ivan Laković, Tomislav Levanić

30. studenoga 2016.

## Sadržaj

<b>1</b>	<b>Uvod u NFM</b>	<b>3</b>
<b>2</b>	<b>Korištenje NFM-a u klasteriranju</b>	<b>4</b>
<b>3</b>	<b>Algoritam</b>	<b>6</b>
3.1	Blokovni silazak po koordinatama (Block Coordinate Descend - BCD)	6
3.1.1	Konvergencija . . . . .	7
3.1.2	Kriterij zaustavljanja . . . . .	7
3.2	Prorijeđen NFM . . . . .	8
3.3	NFM sa slabim nadzorom . . . . .	8
<b>4</b>	<b>Implementacija</b>	<b>10</b>
4.1	Sakupljanje podataka . . . . .	10
4.2	Priprema podataka . . . . .	12
4.2.1	Čišćenje podataka . . . . .	12
4.2.2	Traženje riječi u rječniku . . . . .	15
4.2.3	Učitavanje u Octave (Matlab) . . . . .	15
4.2.4	Nenegativna faktorizacija matrice, NMF . . . . .	17
4.3	Rezultati . . . . .	18
<b>5</b>	<b>Zaključak</b>	<b>24</b>

## 1 Uvod u NFM

Nenegativna faktorizacija matrica (NFM) je metoda za smanjenje dimenzije i faktORIZACIJU. Ona aproksimira nenegativnu matricu produktom dvije nenegativne matrice niskog ranga. NMF se koristi u klasifikaciji podataka i modeliranju tema.

Neka je dana nenegativna matrica  $A \in \mathbb{R}^{m \times n}$ . Ako je tražena donja dimenzija  $k$  onda je cilj NFM-a pronaći dvije nenegativne matrice  $W \in \mathbb{R}^{m \times k}$  i  $H \in \mathbb{R}^{k \times n}$  takve da vrijedi:

$$A \approx WH$$

Po ovoj formuli, svaka podatkovna točka, koja je reprezentirana stupcem od  $A$ , može biti aproksimirana linearnom kombinacijom nenegativnih baznih vektora (stupci od  $W$ ). Matrice  $W$  i  $H$  tražimo tako da riješavamo optimizacijski problem opisan Frobeniusovom normom (mjera udaljenosti između dvije matrice):

$$\min_{W \geq 0, H \geq 0} f(W, H) = \|A - WH\|_F^2$$

Cilj nam je pokazati primjenu NFM-a u klasteriranju dokumenata i modeliranju tema. Klasteriranje dokumenata je organizacija velikih kolekcija teksta u više semantičkih klastera, cilj mu je olakšati korisnikovo pretraživanje. Modeliranje tema je povezano sa blagim klasteriranjem gdje su dokumenti prikazani kao težinske kombinacije tema prema povezanosti sa svakom od njih.

## 2 Korištenje NFM-a u klasteriranju

Smanjenje dimenzije i klasteriranje usko su povezani. Neka je  $A \in \mathbb{R}_+^{m \times n}$ ,  $W \in \mathbb{R}_+^{m \times k}$ ,  $H \in \mathbb{R}_+^{k \times n}$  i  $k \ll \min(m, n)$ . Stupci od  $A$  reprezentiraju  $n$  dokumenata u  $m$ -dimenzionalnom prostoru, a svaki stupac od  $H$  je  $k$ -dimenzionalna reprezentacija dokumenta. Ako možemo iskoristiti  $H$  da podijelimo  $n$  dokumenata u  $k$  grupa, onda je klasifikacija specijalan tip smanjenja dimenzije.

Jedan primjer je k-means klasifikacija.

$$\min \sum_{i=1}^n \|a_i - w_{g_i}\|_2^2$$

gdje su  $a_1, \dots, a_n$  stupci matrice  $A$ ,  $w_1, \dots, w_k$  središta grupa i  $g_i = j$  kada je  $i$ -ta točka (dokument) pridružena  $j$ -toj grupi. Definirajmo k-means kao problem smanjenja dimenzije.

$$\min_{H \in \{0,1\}^{k \times n}, H^T \mathbf{1}_k = \mathbf{1}_n} \|A - WH\|_F^2$$

gdje su  $\mathbf{1}_k \in \mathbb{R}^{k \times 1}$  i  $\mathbf{1}_n \in \mathbb{R}^{n \times 1}$  vektori čiji su svi elementi 1. U k-means formulaciji, stupci od  $W$  su središta grupa, a jedini element različit od nule u svakom stupcu od  $H$  označava dodjelu grupe.

Još jedan primjer je NFM.

$$\min_{W \geq 0, H \geq 0} f(W, H) = \|A - WH\|_F^2$$

U ovakvoj formulaciji, stupci od  $W$  čine bazu  $k$ -dimenzionalnog prostora, a stupci od  $H$  reprezentiraju  $a_1, \dots, a_n$  u tom prostoru. Sa samo nenegativnošću od  $H$ , ta formulacija može se interpretirati kao rješenje klasifikacije: stupci od  $W$  su  $k$  reprezentanta klastera, a  $i$ -ti stupac od  $H$  sadrži informaciju o pripadanju  $i$ -tog dokumenta svakom od klastera. U slučaju problema klasifikacije i modeliranja tema, bazni vektori u  $W$  reprezentiraju  $k$  tema, a koeficijenti u  $i$ -tom stupcu od  $H$  razmjere relevantnosti tih tema za  $i$ -ti dokument. Ako za svaki dokument želimo odabrati samo jednu temu, samo odaberemo najveći koeficijent u svakom retku od  $H$ .

Pokazali smo da k-means i NFM imaju ekvivalentnu formu ciljane funkcije  $\|A - WH\|_F^2$ . Ipak, svaka od metoda ima svoje uvjete pod kojima se dobro ponaša. K-means pretpostavlja da svaki klaster ima sfernu Gaussovu distribuciju. NMF pak omogućava bolju aproksimaciju nižeg ranga matrice  $A$  od K-means formulacije. Ako je  $k \leq \text{rang}(A)$  onda su stupci od  $W$  linearno nezavisni budući da je  $\text{rang}(A) \leq \text{nenegativni-rang}(A)$  (Nenegativni rang matrice  $X \in \mathbb{R}_+^{m \times n}$  je najmanji broj  $k$  takav da  $X = WH$  gdje je  $W \in \mathbb{R}_+^{m \times k}$  i  $H \in \mathbb{R}_+^{k \times n}$ ). Tako da je NFM bolji kada klasteri odgovaraju linearno nezavisnim vektorima.

Uspjeh NFM-a ovisi o osnovnom skupu podataka i najveći uspjeh postiže u klasifikaciji dokumenata. Ako je matrica  $A$  dokument pojmova, bazni vektori  $w_j$  predstavljaju distribuciju ključnih riječi za svaku temu. Kada su te distribucije linearno nezavisne, što je obično slučaj, NFM može pravilno izvesti klasterne određene oznakama.

U zadnje vrijeme, NFM se počeo uspješno primjenjivati i u modeliranju tema. I klasifikacija dokumenata i modeliranje tema mogu se shvatiti kao problem smanjenje dimenzije. Kada modeliramo teme, stupci dobivene matrice  $W$  predstavljati će teme u odnosu na ključne riječi, a  $i$ -ti stupac dobivene matrice  $H$  prikazati će zastupljenost svake od tema u  $i$ -tom dokumentu.

Jedina razlika između NFM-a i standardnih algoritama za modeliranje tema je da stupci od  $W$  i  $H$  nemaju jediničnu  $L_1$  normu, no to lako ispravimo dodavanjem matrica za skaliranje:

$$A \approx WH = (WD_W)(D_W^{-1}H) = \tilde{W}\tilde{H}$$

gdje je dijagonalna komponenta dijagonalne matrice  $D_W \in \mathbb{R}_+^{k \times k}$  jednaka sumi stupca od  $W$ . Sada je nova matrice  $\tilde{W}$  normalizirana po stupcima.

### 3 Algoritam

NFM je NP-težak problem, pa ćemo kao aproksimativno rješenje tražiti lokalni minimum. Obraditi ćemo algoritam silaska po blokovnim koordinatama.

#### 3.1 Blokovi silazak po koordinatama (Block Coordinate Descend - BCD)

BCD je vrlo popularna metoda rješavanja nelinearnih optimizacijskih problema. On dijeli varijable u disjunktne podgrupe i iterativno minimizira ciljnu funkciju pazeći na varijable iz tih podgrupa. U NFM-u varijable koje moramo riješiti su  $W$  i  $H$  pa, prirodno, podijelimo u podgrupe radimo na način da sve varijable podijelimo u dva bloka koja predstavljaju redom  $W$  i  $H$ .

Naizmjenično rješavamo:

$$W \leftarrow \arg \min_{W \geq 0} f(W, H)$$

$$H \leftarrow \arg \min_{H \geq 0} f(W, H)$$

To možemo zapisati i kao:

$$\min_{W \geq 0} \|H^T W^T - A^T\|_F^2 \quad (1)$$

$$\min_{H \geq 0} \|WH - A\|_F^2 \quad (2)$$

Ove potprobleme nazivamo nenegativno ograničeni problemi najmanjih kvadrata. BCD algoritam:

**Data:** Matrica  $A \in \mathbb{R}^{m \times n}$ , dopuštena greška  $0 < \epsilon \ll 1$ , gornja granica za broj iteracija  $T$

Inicijalizacija matrice  $H$ ;

**repeat**

    Pronađi optimalno rješenje potproblema 1;

    Pronađi optimalno rješenje potproblema 2;

**until** Zadovoljili smo kriterij zaustavljanja na temelju  $W, H$  i  $\epsilon$  ili je broj iteracija dosegao dozvoljenih  $T$ ;

**Result:**  $W, H$

#### Algorithm 1: BCD ALGORITAM

Drukčije inicijalizacije od  $H$  mogu dovesti do drukčijih rješenja. Uobičajena je strategija pokrenuti BCD algoritam za različite nasumične inicijalizacije od  $H$  i odabrati rješenje sa minimalnom vrijednosti ciljne funkcije.

### 3.1.1 Konvergencija

Ciljna funkcija NFM-a je polinom četvrtog reda, dakle nekonveksna funkcija. Za nekonveksni optimizacijski problem većina algoritama garantira stacionarnost točke rješenja, a ne nužno lokalnu minimalnost. U praksi, često pokrećemo NFM algoritam za razne inicijalizacije od  $W$  i  $H$  i odaberemo izlaz sa najmanjom vrijednosti ciljne funkcije.

**Teorem 1.** *Ako je minimum svakog od podproblema 1 i 2 dobiven u svakom koraku, svaka granična točka niza  $\{(W, H)^{(i)}\}$  generirana BCD algoritmom je stacionarna točka od  $\|A - WH\|_F^2$ .*

U stacionarnoj točki rješenja, zadovoljeni su Karush-Kuhn-Tucher (KKT) uvjeti:

$$\begin{aligned} W &\geq 0, H \geq 0, \\ \nabla f_W &= 2WHH^T - 2AH^T \geq 0, \nabla f_H = 2W^TWH - 2W^TA \geq 0, \\ W \cdot * \nabla f_W &= 0, H \cdot * \nabla f_H = 0. \end{aligned}$$

### 3.1.2 Kriterij zaustavljanja

Sve iterativne metode moraju imati kriterij zaustavljanja. Naivan pristup je zaustaviti se kada smanjenje vrijednosti ciljne funkcije postane manje od unaprijed definirane granice.

$$|f(W^{(i-1)}, H^{(i-1)}) - f(W^{(i)}, H^{(i)})| \leq \epsilon$$

Ovaj pristup se često koristi iako može dovesti do krivog rješenja kada smanjenje ciljne funkcije postane malo prije nego što smo došli do stacionarne točke. Imamo i bolje rješenje za kriterij zaustavljanja. Definirajmo projicirani gradijent

$$(\nabla^P f_W)_{ij} = \begin{cases} (\nabla f_W)_{ij} & \text{ako } (\nabla f_W)_{ij} < 0 \text{ or } W_{ij} > 0; \\ 0 & \text{inače} \end{cases}$$

za  $i = 1, \dots, m$  i  $j = 1, \dots, k$ . Sada KKT uvjete možemo zapisati kao:

$$\nabla^P f_W = 0 \text{ i } \nabla^P f_H = 0$$

Označimo projicirane gradijente matrice u  $i$ -toj iteraciji sa  $\nabla^P f_W^{(i)}$  i  $\nabla^P f_H^{(i)}$  i definirajmo

$$\Delta(i) = \sqrt{\|\nabla^P f_W^{(i)}\|_F^2 + \|\nabla^P f_H^{(i)}\|_F^2}$$

Koristeći tu definiciju kriterij zaustavljanja možemo zapisati kao:

$$\frac{\Delta(i)}{\Delta(1)} \leq \epsilon$$

gdje je  $\Delta(1)$  iz prve iteracije od  $(W, H)$ . Taj kriterij garantira stacionarnost točke rješenja.

### 3.2 Prorijeđen NFM

Sa nenegativnošću kao jedinim ograničenjem, dobivena matrica  $H$  NFM-a sadrži razmjerno pridružene vrijednosti koji odgovaraju k klastera koje reprezentiraju stupci od  $W$ . Pokazuje se da ograničenje prorijeđenosti na  $H$  olakšava interpretaciju rezultata NFM-a i poboljšava kvalitetu klasteriranja. Na primjer, pogledajmo dva različita scenarija za stupac od  $H \in \mathbb{R}_+^{3 \times n} : (0.2, 0.3, 0.5)^T$  i  $(0, 0.1, 0.9)^T$ . Očito je potonji bolji indikator da odgovarajuća točka podatka pripada trećem klasteru.

Nova ciljna funkcija sa odgovarajućim ograničenjima:

$$\min_{W, H \geq 0} \|A - WH\|_F^2 + \phi(W) + \psi(H)$$

Ako želimo prorijeđenost od  $H$ , možemo koristiti  $L_1$  normu:

$$\phi(W) = \alpha \|W\|_F^2$$

i

$$\psi(H) = \beta \sum_{i=1}^n \|H(:, i)\|_1^2$$

$L_1$  norma od  $\psi(H)$  promovira prorijeđenost stupaca od  $H$ , a Frobeniusova norma od  $\phi(W)$  je potrebna da  $W$  previše ne naraste. Skalari  $\alpha$  i  $\beta$  se koriste da bi kontrolirali jačinu regularizacije.

Oskudni NFM se može lako računati koristeći BCD. Možemo izmjeniti formulaciju NFM-a i odproblemi od BCD-a postaju:

$$\begin{aligned} \min_{W \geq 0} & \left\| \begin{pmatrix} H^T \\ \sqrt{\alpha} I_k \end{pmatrix} W^T - \begin{pmatrix} A^T \\ 0_{k \times m} \end{pmatrix} \right\|_F^2 \\ \min_{H \geq 0} & \left\| \begin{pmatrix} H^T \\ \sqrt{\beta} 1_k^T \end{pmatrix} H - \begin{pmatrix} A \\ 0_n^T \end{pmatrix} \right\|_F^2 \end{aligned}$$

### 3.3 NFM sa slabim nadzorom

NFM sa slabim nadzorom u običan NFM dodaje razne korisničke ulaze da bi korisnik mogao sam poboljšati klasteriranje i modeliranje tema. Efektivno se koristi kao baza za vizualno analitičko modeliranje tema.

Unosi korisnika prikazuju se u obliku referentne matrice za  $W$  i  $H$ . Te referentne matrice koristimo na način da pokušavamo  $H$  i  $W$  učiniti sličnijim njima. Odnosno, ako su dane matrice  $W_r \in \mathbb{R}_+^{m \times k}$  za  $W$  i  $H_r \in \mathbb{R}_+^{k \times n}$  za  $H$ , dijagonalne težinske matrice  $M_W \in \mathbb{R}_+^{k \times k}$  i  $M_H \in \mathbb{R}_+^{n \times n}$ , matrica podataka  $A \in \mathbb{R}_+^{m \times n}$ , i cijeli broj  $k \ll \min(m, n)$ , NFM sa slabim nadzorom ima dodatne uvjete regularizacije koje penaliziraju razlike između  $H_r$  i  $H$  (stupčano skalirano putem  $D_H$ ) i između  $W_r$  i  $W$  tako da:

$$f(W, H, D_H) = \min_{W, H, D_H} \|A - WH\|_F^2 + \|(W - W_r)M_W\|_F^2 + \|(H - H_r D_H)M_H\|_F^2 \quad (3)$$



za  $W \in \mathbb{R}_+^{m \times k}$  i  $H \in \mathbb{R}_+^{k \times n}$  i dijagonalnu matricu  $D_H \in \mathbb{R}_+^{n \times n}$ .

Kroz ovakvu regularizaciju, NFM sa slabim nadzorem može ukomponirati razne tipove korisnikovog prijašnjeg znanja. Svaki stupac od  $H_r$  specificira članstvo nekog podatka u klasteru. Dijagonalna matrica  $D_H$  računa moguće razlike u skaliranju između  $H_r$  i  $H$ . Npr. vektori (0.1,0.3,0.6) i (0.2,0.6,1.2) se jednako interpretiraju u smislu članstva u klasterima i  $D_H$  omogućuje da se jednako i tretiraju. Također, omogućen je djelomičan nadzor na podskup stupaca u  $H_r$ . To omogućuje težinska matrica  $M_H$  tako da smanji težinu stupaca podataka u  $H_r$  bez prethodnih informacija.

$W_r$  nadzire osnovnu reprezentaciju. U klasteriranju dokumenata i modeliranju tema, stupci od  $W_r$  definiraju reprezentaciju tema u  $W$  po ključnim riječima. Dijagonalna matrica  $M_W$  omogućuje nadzor nad podskupu stupaca u  $W$  tako da smanji težine onim stupcima od  $W_r$  koji nisu pod nadzorom.

Optimizacija od (3) prati BCD tako da iterativno računamo  $W$ ,  $H$  i  $D_H$ . Sa danim početnim vrijednostima,  $W$  računamo:

$$W \leftarrow \arg \min_{W \geq 0} \left\| \begin{bmatrix} H^T \\ M_W \end{bmatrix} W^T - \begin{bmatrix} A^T \\ M_W W_r^T \end{bmatrix} \right\|_F^2$$

Svaki stupac od  $H$  se računa jedan po jedan po formuli:

$$H(:, i) \leftarrow \arg \min_{H(:, i) \geq 0} \left\| \begin{bmatrix} W \\ M_H(i) I_k \end{bmatrix} H(:, i) - \begin{bmatrix} A(:, i) \\ M_H(i) D_H(i) H_r(:, i) \end{bmatrix} \right\|_F^2$$

A  $i$ -ta dijagonalna komponenta  $D_H(i)$  od  $D_H$ :

$$D_H(i) \leftarrow \begin{cases} \frac{H_r(:, i)^T H(:, i)}{\|H_r(:, i)\|_2^2} & \text{ako } M_H(i) \neq 0; \\ 0 & \text{inače} \end{cases}$$

## 4 Implementacija

Nakon proučavanja članka odlučili smo testirati da li je moguće na isti način klasificirati i tekstove pisane na hrvatskom jeziku, te kakve bi rezultate takvo klasificiranje dalo. Prije početka implementacije znali smo da će najveći problem predstavljati postojanje brojnih različitih verzija iste riječi što u engleskom jeziku nije problem jer broj oblika u kojima se riječ može pronaći je mnogo manji. U problem smo se odlučili upustiti nakon otkrivanja rječnika profesora Gorana Igalyja (<http://www.igaly.org/rjecnik-hrvatskih-jezika>). Kako ne postoje neki dogovoreni setovi podataka na kojima se klasifikacija hrvatskih tekstova provjerava odlučili smo se za testirati algoritam na tekstovima Jutarnjeg lista (<http://www.jutarnji.hr>).

### 4.1 Sakupljanje podataka

Pri sakupljanju podataka koristili smo programski jezik Python te modul Scrapy. Scrapy je open source framework za prikupljanje podataka sa web stranica. Osnovna ideja cijelog postupka je otvoriti stranicu, ako postoje, prikupiti naslov te sadržaj članka, potom pronaći sve linkove nepročitanih članaka sa iste domene te ponoviti postupak za svakog od njih. Kratkim analizom nekoliko članaka sa stranice [www.jutarnji.hr](http://www.jutarnji.hr) pronalazimo pravilnosti u HTML kodu stranice te to koristimo za dohvaćanje naslova, sadržaja, linka te tagova članka. Link i tagovi dohvaćaju se radi automatizacije testiranja kvalitete rezultata. To spremanje podataka obavlja sljedeći kod

```
import scrapy

class JutarnjiSpider(scrapy.Spider):
    name = 'JutarnjiSpider'
    allowed_domains = ['www.jutarnji.hr']
    start_urls = ['http://www.jutarnji.hr/']

    custom_settings = {
        'CLOSESPIDER_ITEMCOUNT': 10000
    }

    def parse(self, response):
        for article in response.css('.container > section'):
            # get article content
            content = '\n'.join(article.css(
                'section#CImaincontent > div ::text'
            ).extract())
            content = '\n'.join(list(filter(None,
                content.splitlines()))))

            if content == "":
                continue

            # get article tag list
            tags = []
            for t in article.css('.tags .list-inline ::text').extract():
                if t != '\n':
                    tags.append(t.replace('\n', ''))

            # yield article data
            yield {
                'title': article.css('h1.title ::text').extract_first(),
                'content': content,
                'tags': tags,
                'url': response.url
            }

            # get next batch of links to crawl
            next_pages = response.css('article a ::attr(href)').extract()
            for next_page in next_pages:
                if next_page:
                    yield scrapy.Request(response.urljoin(next_page),
                        callback=self.parse)
```



ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n15/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n16/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n17/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n18/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n19/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n20/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n21/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n22/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n23/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n24/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n25/46 \nObjavljeno:\n 18.07.2016.\nHardwell\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n26/46 \nObjavljeno:\n 18.07.2016.\nHardwell\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n27/46 \nObjavljeno:\n 18.07.2016.\nHardwell\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n28/46 \nObjavljeno:\n 18.07.2016.\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n29/46 \nObjavljeno:\n 18.07.2016.\nHardwell\nZvonimir Barisin / HANZA MEDIA\nZatvori\nFOTOGALERIJA: SPEKTAKULARAN KRAJ ULTRE Pogledajte kakva je bila atmosfera na zatvaranju\nSlika \n30/46 \nObjavljeno:\n 18.07.2016.\nBozidar Vukicevic / HANZA MEDIA\nZatvori\nFOTOGALERIJA:

Najveći problem od svega su nam predstavljale galerije slika. Kao što je vidljivo na slici, za svaku sliku galerije ide jednak tekst i vrijeme objave koji mogu odnos riječi u članku bitno promijeniti.

Popis nepravilnosti u dobivenom tekstu:

1. Kodovi umjesto č, ć, đ, š, ž.
2. Objave slika i videa s Instagrama i Twittera.
3. Galerije slika i videa s portala koji dupliciraju tekst.
4. Ostali kodovi \n, \u00a0, \u0107, \ufe0f, ...
5. Tagovi i url koje smo odlučili izbaciti da ne bi bilo "varanja".
6. Interpunkcija, brojevi te velika i mala slova.

Također, odlučili smo se zanemariti sve riječi koje imaju manje od 3 slova jer te riječi više opisuju osobu koja članak piše nego samu temu članka.

```
function edit_croatian_chars() {

    sed -i 's/\\u0106/ć/g' "$1"
    sed -i 's/\\u0107/č/g' "$1"

    sed -i 's/\\u010c/Č/g' "$1"
    sed -i 's/\\u010d/č/g' "$1"

    sed -i 's/\\u0160/Š/g' "$1"
    sed -i 's/\\u0161/š/g' "$1"

    sed -i 's/\\u017d/Ž/g' "$1"
    sed -i 's/\\u017e/ž/g' "$1"

    sed -i 's/\\u0110/Đ/g' "$1"
    sed -i 's/\\u0111/đ/g' "$1"

    return
}

function edit_multiple_pictures() {

    sed -i 's/\\nSlika.*2016.\\n/ /g' $1
    sed -i 's/\\nZatvori\\nFOTO:/ /g' $1
    sed -i 's/Pogledajte gale.*", "tags":/", "tags":"/g' $1

    sed -i 's/photo posted.*PDT/ /g' $1
    sed -i 's/Fotog.*PDT/ /g' $1

    sed -i 's/ideo.*PDT/ /g' $1
    sed -i 's/@[a-zA-Z0-9]*/ /g' $1

    return
}

function remove_backslash_code() {

    sed -i 's/\\[n"]/ /g' $1
    sed -i 's/\\u[^ ]\\{4\\}/ /g' $1

    return
}
```

#### 4.2.2 Traženje riječi u rječniku

Nakon što smo dobiveni tekst "počistili" potrebno je za svaki članak vidjeti koje sve riječi sadrži. Glavnu ulogu u ovom dijelu igra rječnik profesora Igalyja. Posebnost tog rječnika koji sadrži gotovo sve riječi hrvatskog jezika je ta što sadrži i sve padeže za imenice, vremena za glagole tj. sadrži sve oblike u kojima se riječ može pojaviti i korijenski oblik. Kako sam problem pretraživanja nije težak za implementirati dolazi do poteškoće koje ja prvi pogled nije očita. To je vrijeme pretraživanja svake riječi. Kako naš ulazni skup podataka sadrži 1182 članaka, nakon čišćenja u skupu je ostalo 461,807 riječi za koje je potrebno provjeriti nalaze li se u imeniku, te ako da koji im je izvorni oblik i na kojem se mjestu u rječniku nalaze. Treba još navesti da rječnik sadrži više od 600,000 riječi.

Nakon provođenja tog postupka za svaki članak imamo popis riječi koje se u njemu nalaze, koliko puta se pojavljuju i njihovo mjesto u rječniku što će nam koristiti kod učitavanja podataka u matricu.

#### 4.2.3 Učitavanje u Octave (Matlab)

Matricu smo popunili tako da svaki stupac simbolizira jedan članak, a svaki redak jednu riječ iz rječnika. Nakon popunjavanja matrice velika većina od početnih 126,000+ riječi nikad se nije pojavila stoga nakon izbacivanja tih riječi, ali i riječi koje su se u čitavom tekstu pojavile manje od 3 puta ostali smo na 11812 riječi. Stoga je naša nova matrica, matrica na kojoj ćemo raditi kasifikaciju dimenzija 11812x1182 (ukoliko su brojevi double-ovi to je 100+MB). Kako su i to velike dimenzije važno je primijetiti da je matrica jako rijetka te ju je bolje čuvati kao sparse matricu.

```
function [matrix, dictionary] = process_data(matrix, dictionary)

    % Remove words that appeared less than 3 times in articles.
    [matrix, dictionary] = remove_rows(matrix, dictionary);

    % Export reduced dictionary.
    export_used_words(dictionary);

    % Remove articles that has less than 5 words and normalize each article.
    matrix = remove_normalize_columns(matrix);

end

function [matrix, dictionary] = remove_rows(matrix, dictionary)

    appeared = 0;
    remove_row = -1;
    matrix(1,:) = 0;
```

```
for i=1:size(matrix, 1)

    if sum(matrix(i,:)) < 3
        if appeared == 0
            remove_row = i;
            appeared = 1;
        else
            remove_row (end+1) = i;
        end
    end
end

if appeared == 1
    matrix(remove_row, :) = [];
    dictionary(remove_row, :) = [];
end
end

function [matrix] = remove_normalize_columns(matrix)

    appeared = 0;
    remove_col = -1;
    for i=1:size(matrix, 2)

        % Save suma, we will use it for normalization.
        suma = sum(matrix(:,i));
        if suma < 5
            if appeared == 0
                remove_col = i;
                appeared = 1;
            else
                remove_col(end+1) = i;
            end
        % If we won't remove column we need to normalize it.
        else
            matrix(:,i) = matrix(:,i)/suma;
        end
    end

    if appeared == 1
        matrix(:, remove_col) = [];
    end
end
```



#### 4.2.4 Nenegativna faktORIZACIJA matrice, NMF

Za izvođenje NMF-a odlučili smo se iskoristiti gotov algoritam kojeg je razvio tim s fakulteta Georgia Tech.<sup>1</sup> Prije korištenja NMF normalizirali smo vrijednosti za svaki članak tako da suma svih vrijednosti bude 1. Tako se dobiva privid da su svi članci jednake duljine. Prilikom izvođenja testirali smo algoritam za sparse (rijetke) matrice i popunjene. Za pokretanje algoritma potrebno je i zadati broj klastera na koje želimo rastaviti matricu. Kako je broj realnih tema za naše članke 50 odlučili smo testirati klasifikaciju na sve slučajeve između 2 i 50. Osim dodatnih informacija vezanih za izvođenje nama najvažniji podaci su dobivene matrice  $W$  i  $H$ . Pri formiranju rezultata odlučili smo se na grubo klasteriranje stoga nam je za to potrebna jedino matrica  $H$  u kojoj redci simboliziraju teme, a stupci članke. Kod grubog klasteriranja tražimo samo najveću vrijednost za svaki stupac/članak i kažemo da članak pripada toj temi.

```
function [tema, tema_sparse] do_nmf(matrix, k)

    [W,H,iter,HIS]=nmf(matrix, k);

    for i=1:size(H,2)
        [maks, posit] = max(H(:,i));
        tema(i) = posit;
    end

    matrix = sparse(matrix);
    if issparse(matrix) == 1
        [W,H,iter,HIS]=nmf(matrix, k, 'type', 'sparse');

        for i=1:size(H,2)
            [maks, posit] = max(H(:,i));
            tema_sparse(i) = posit;
        end

    end
end
```

---

<sup>1</sup><http://www.cc.gatech.edu/~hpark/nmfsoftware.php>

### 4.3 Rezultati

Analizom prikupljenih podataka te pripadnih linkova dobivamo podjelu svih članaka na 50 kategorija. Sve članke koje nismo mogli klasificirati tom metodom smo zanemarili (njih 80), te je tako provjera kvalitete klasifikacije provjerena na 1102 članka. Sljedeći kod obavlja analizu tema te izračun NMI rezultata za tu (originalnu) klasifikaciju tema te

```
import json, csv
from urlparse import urlparse
from sklearn.metrics.cluster import normalized_mutual_info_score

class Articles:
    def __init__(self, filename):
        self._data = Articles.read_json(filename)
        self.themes = self._get_themes()

    def _get_themes(self):
        MAX_THEME_LENGTH = 20

        for d in self._data:
            lst = urlparse(d['url']).path.split('/')
            root = child = None
            if (len(lst[1]) < MAX_THEME_LENGTH and
                len(lst[2]) < MAX_THEME_LENGTH and
                lst[1] not in ['config', 'promo']):
                root = lst[1]
                child = lst[2]
                d['theme'] = root + '/' + child
            else:
                d['theme'] = None

        themes = []
        for d in self._data:
            if d['theme'] is not None and d['theme'] not in themes:
                themes.append(d['theme'])

        return themes

    def nmi(self, numbers):
        original = [self.themes.index(n['theme']) for n in self._data
                    if n['theme'] is not None]
        _numbers = [numbers[i] for i in range(len(numbers))
                    if self._data[i]['theme'] is not None]

        return normalized_mutual_info_score(original, _numbers)
```

```

@staticmethod
def read_csv(filename):
    data = []
    with open(filename, 'r') as f:
        c = csv.reader(f, delimiter=',')
        for r in c:
            for s in r:
                data.append(int(s))
    return data

@staticmethod
def read_json(filename):
    data = None
    with open(filename, 'r') as f:
        data = json.load(f)
    return data

```

Sada navodimo NMI rezultate slučaja 2 - 50 spomenutih u prethodnom poglavlju.

```

2 nmi = 0.0655343980115
2_sparse nmi = 0.0651450811895
3 nmi = 0.0942524455946
3_sparse nmi = 0.0894573335087
4 nmi = 0.134824547628
4_sparse nmi = 0.126554275375
5 nmi = 0.165235381277
5_sparse nmi = 0.151189559668
6 nmi = 0.156183728565
6_sparse nmi = 0.162955007348
7 nmi = 0.181016955443
7_sparse nmi = 0.163415537917
8 nmi = 0.179484618514
8_sparse nmi = 0.179351915488
9 nmi = 0.171248975099
9_sparse nmi = 0.19205496426
10 nmi = 0.207669903523
10_sparse nmi = 0.207630112845
11 nmi = 0.191966164057
11_sparse nmi = 0.215381160897
12 nmi = 0.21678848689
12_sparse nmi = 0.244004894969
13 nmi = 0.231326732559
13_sparse nmi = 0.246003958304
14 nmi = 0.240957825004

```

```
14_sparse nmi = 0.246338992716
    15 nmi = 0.246986599662
15_sparse nmi = 0.247174061356
    16 nmi = 0.264315292421
16_sparse nmi = 0.259681018764
    17 nmi = 0.249376406801
17_sparse nmi = 0.271247592633
    18 nmi = 0.261607663606
18_sparse nmi = 0.263958921566
    19 nmi = 0.272843889474
19_sparse nmi = 0.282943137187
    20 nmi = 0.280706149283
20_sparse nmi = 0.276595440614
    21 nmi = 0.286741103654
21_sparse nmi = 0.295900441375
    22 nmi = 0.310207717391
22_sparse nmi = 0.283863356052
    23 nmi = 0.282180880452
23_sparse nmi = 0.291334246485
    24 nmi = 0.296646329799
24_sparse nmi = 0.292723841228
    25 nmi = 0.304475075617
25_sparse nmi = 0.295114320652
    26 nmi = 0.303996904087
26_sparse nmi = 0.289096497021
    27 nmi = 0.300372449629
27_sparse nmi = 0.300305093038
    28 nmi = 0.304608893617
28_sparse nmi = 0.309182735792
    29 nmi = 0.307155782584
29_sparse nmi = 0.308845929434
    30 nmi = 0.318816951849
30_sparse nmi = 0.309549258343
    31 nmi = 0.310005050985
31_sparse nmi = 0.327482345781
    32 nmi = 0.297548010137
32_sparse nmi = 0.323206389411
    33 nmi = 0.333780744908
33_sparse nmi = 0.318345815116
    34 nmi = 0.315718465272
34_sparse nmi = 0.338594706533
    35 nmi = 0.322761364237
35_sparse nmi = 0.325737941337
    36 nmi = 0.340002580721
36_sparse nmi = 0.325123037507
    37 nmi = 0.335586063592
```

```
37_sparse nmi = 0.336227618834
    38 nmi = 0.36047906722
38_sparse nmi = 0.339709017051
    39 nmi = 0.335036923674
39_sparse nmi = 0.334623714366
    40 nmi = 0.33554773561
40_sparse nmi = 0.351335582385
    41 nmi = 0.338521714501
41_sparse nmi = 0.340053554794
    42 nmi = 0.320660201843
42_sparse nmi = 0.359404538774
    43 nmi = 0.348212734764
43_sparse nmi = 0.335778874251
    44 nmi = 0.343608825116
44_sparse nmi = 0.354010383638
    45 nmi = 0.332029521967
45_sparse nmi = 0.352874530427
    46 nmi = 0.350283396918
46_sparse nmi = 0.354453184905
    47 nmi = 0.357760194379
47_sparse nmi = 0.346477654093
    48 nmi = 0.358719359466
48_sparse nmi = 0.350193466416
    49 nmi = 0.362238853651
49_sparse nmi = 0.370390771453
    50 nmi = 0.366952574539
50_sparse nmi = 0.356695427323
```

U nastavku ispisujemo najvažnije riječi za svaku temu kod klasifikacije teksta u 49 tema.

1 fotografija	26 nov
2 ali	27 film
3 kuca	28 jadrans
4 vrijeme	29 grad
5 prvi	30 glas
6 poznat	31 pas
7 bih	32 bilo
8 sustav	33 panda
9 nov	34 igra
10 zbog	35 kako
11 automobil	36 izbor
12 godina	37 model
13 imati	38 hrvatska
14 split	39 koji
15 koji	40 predsjednik
16 sd	41 kao
17 reci	42 htjeti
18 video	43 cesta
19 fotografija	44 and
20 moci	45 kuna
21 sto	46 zagrebacki
22 selo	47 biti
23 festival	48 vlada
24 korisnik	49 posto
25 proizvod	



## 5 Zaključak

U ovom seminaru prikazali smo korištenje NFM-a u klasifikaciji dokumenata. Nakon što smo definirali NFM i objasnili njegovu primjenu u klasifikaciji odlučili smo algoritam testirati na tekstovima na hrvatskom jeziku i to na člancima Jutarnjeg lista. Rezultate koje smo dobili nešto su lošiji od rezultata dobivenih u [1]. Takvi rezultati su i očekivani s obzirom na to da su naši testni podaci ručno sakupljeni, nestandardni te rječnik koji smo koristili pri klasifikaciji nije namijenjen takvom korištenju. Da bi popravili dobivene rezultate potrebno je u rječnik uvesti još mnoge nestandardne riječi koje ovise o podacima (poput trenutno aktualnih imena ljudi, gradova, političkih stranaka).



## Literatura

- [1] Da Kuang and Jaegul Choo and Haesun Park *Nonnegative matrix factorization for interactive topic modeling and document clustering*.
- [2] <http://www.cc.gatech.edu/hpark/nmfsoftware.php>.
- [3] [http://scikit-learn.org/stable/modules/generated/sklearn.metrics.normalized\\_mutual\\_info\\_score.html](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.normalized_mutual_info_score.html).
- [4] <https://doc.scrapy.org/en/1.2/>.