CFLG 2020: Obrada lingvističkih podataka korištenjem SSF API-a

Domagoj Ševerdija*, Ivan Židov†

24. rujna 2020.

1 Kako obraditi lingivističke podatke koristeći Python i SSF?

Za demonstraciju ove radionice možete se prijaviti na JupyterHub platformu sa korisničkim računom `clgf` i lozinkom `clgf`.

Unesimo potrebne pakete.

[2]:
```
import os
import sys

# NLP paketi
import nltk
import random

# ML NLP paketi
import spacy
from spacy import displacy

# SSF za hr
from ssf import POSTagger,SSF


# vizualizacija podataka
import numpy as np
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('ggplot')

from sklearn.decomposition import PCA
```

*dseverdi@mathos.hr
†izidov@thanks.hr

2 Definiranje HR korpusa

Učitajmo tekst i spremimo ga koristeći `NLTK` modul za definiranje vlastitog kategoriziranog korpusa. Naš korpus će imati 2 kategorije: SZ i NZ. Uočimo kako će postupak provesti tokenizaciju teksta na riječi, rečenice i odjeljke.

[3]:
```
from nltk.corpus.reader.plaintext import CategorizedPlaintextCorpusReader
biblija = CategorizedPlaintextCorpusReader('corpora/biblija','.*',cat_pattern =␣
 ↪r'(.*)[/]')
```

Ispitajmo dostupne kategorije.

[4]:
```
biblija.categories()
```

[4]:
```
['NZ', 'SZ']
```

Izvucimo čisti tekst iz korpusa i razlomimo u rečenice.

[5]:
```
nltk.sent_tokenize(biblija.raw('NZ/04_Ivan.txt'))[:10]
```

[5]:
```
['Evanđelje po Ivanu\r\n\r\n\r\n\r\nPROSLOV\r\n\r\n1 U početku bijaše Riječ\r\ni
Riječ bijaše u Boga\r\n\t i Riječ bijaše Bog.',
 'Ona bijaše u početku u Boga.',
 'Sve postade po njoj\r\n\t i bez nje ne postade ništa.',
 'Svemu što postade\r\nu njoj bijaše život\r\n\t i život bijaše ljudima
svjetlo;\r\ni svjetlo u tami svijetli\r\n\t i tama ga ne obuze.',
 'Bi čovjek poslan od Boga,\r\n\t ime mu Ivan.',
 'On dođe kao svjedok\r\n\t da posvjedoči za Svjetlo\r\n\t da svi vjeruju po
njemu.',
 'Ne bijaše on Svjetlo,\r\n\t nego - da posvjedoči za Svjetlo.',
 'Svjetlo istinsko\r\n\t koje prosvjetljuje svakog čovjeka\r\n\t dođe na
svijet;\r\nbijaše na svijetu\r\n\t i svijet po njemu posta\r\n\t i svijet ga ne
upozna.',
 'K svojima dođe\r\n\t i njegovi ga ne primiše.',
 'A onima koji ga primiše\r\n\t podade moć\r\n\t da postanu djeca Božja:\r\n\t
onima koji vjeruju u njegovo ime,\r\nkoji su rođeni\r\n\t ne od krvi,\r\n\t ni
od volje tjelesne,\r\n\t ni od volje muževlje,\r\n\t nego - od Boga.']
```

Korpusni čitač prilikom učitavanja korpusa već napravi tokenizaciju na riječi i rečenice.

[6]:
```
biblija.sents('NZ/04_Ivan.txt')[:5]
```

[6]:
```
[['Evanđelje', 'po', 'Ivanu'],
 ['PROSLOV'],
 ['1',
  'U',
  'početku',
  'bijaše',
```

```
 'Riječ',
 'i',
 'Riječ',
 'bijaše',
 'u',
 'Boga',
 'i',
 'Riječ',
 'bijaše',
 'Bog',
 '.'],
['Ona', 'bijaše', 'u', 'početku', 'u', 'Boga', '.'],
['Sve',
 'postade',
 'po',
 'njoj',
 'i',
 'bez',
 'nje',
 'ne',
 'postade',
 'ništa',
 '.']]
```

Nalazi li se specifična riječ u tom korpusu?

[7]:
```
'raj' in biblija.words()
```

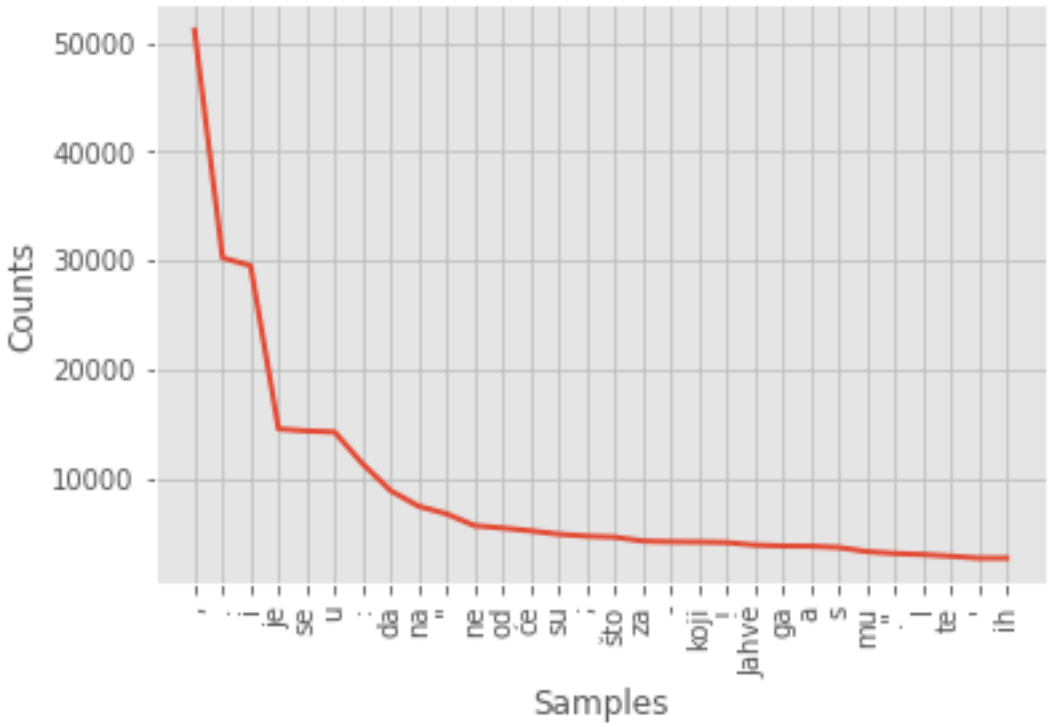
[7]:
```
True
```

2.1 PRIMJENA: Analiza pojavnosti riječi u korpusu.

Koristimo NLTK funkciju za izračun frekvencijske distribucije.

[8]:
```
fd = nltk.FreqDist(biblija.words())
print(fd.most_common(30))
fd.plot(30)
```

```
[(',', 51136), ('.', 30275), ('i', 29555), ('je', 14599), ('se', 14420), ('u',
14322), (':', 11410), ('da', 8951), ('na', 7525), ('"', 6824), ('ne', 5743),
('od', 5528), ('će', 5269), ('su', 4963), (';', 4786), ('što', 4697), ('za',
4336), ('-', 4269), ('koji', 4245), ('!', 4199), ('Jahve', 3965), ('ga', 3879),
('a', 3865), ('s', 3750), ('mu', 3372), ('."', 3193), ('I', 3104), ('te', 2949),
("'", 2774), ('ih', 2771)]
```



[8]: `<matplotlib.axes._subplots.AxesSubplot at 0x7fce18b15c90>`

Primjećujemo veliki broj čestica u tekstu koje nam ne daju neke korisne informacije. Pročistimo onda tekst.

[9]:
```
from string import punctuation

# ucitajmo iz datoteke hr čestice
with open('corpora/hr_stopwords.txt') as f:
    hr_stopwords = f.read().split()

print(hr_stopwords)
```

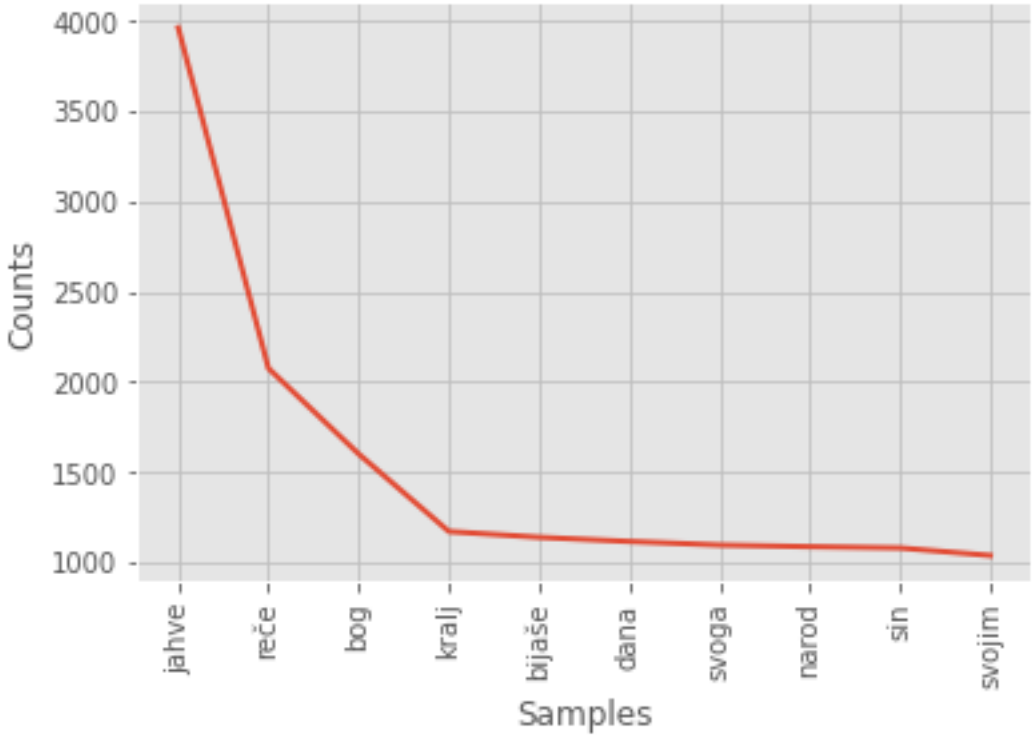
```
['a', 'ako', 'ali', 'bi', 'bih', 'bila', 'bili', 'bilo', 'bio', 'bismo',
'biste', 'biti', 'bumo', 'da', 'do', 'duž', 'ga', 'hoće', 'hoćemo', 'hoćete',
'hoćeš', 'hoću', 'i', 'iako', 'ih', 'ili', 'im', 'iz', 'ja', 'je', 'jedna',
'jedne', 'jedno', 'jer', 'jesam', 'jesi', 'jesmo', 'jest', 'jeste', 'jesu',
'jim', 'joj', 'još', 'ju', 'k', 'ka', 'kad', 'kada', 'kako', 'kao', 'koja',
'koje', 'koji', 'kojima', 'koju', 'kroz', 'li', 'me', 'mene', 'meni', 'mi',
'mimo', 'moj', 'moja', 'moje', 'mu', 'na', 'nad', 'nakon', 'nam', 'nama', 'nas',
'naš', 'naša', 'naše', 'našeg', 'ne', 'nego', 'neka', 'neki', 'nekog', 'neku',
'nema', 'netko', 'neće', 'nećemo', 'nećete', 'nećeš', 'neću', 'nešto', 'ni',
'nije', 'nikoga', 'nikoje', 'nikoju', 'nisam', 'nisi', 'nismo', 'niste', 'nisu',
'njega', 'njegov', 'njegova', 'njegovo', 'njemu', 'njezin', 'njezina',
```

```
'njezino', 'njih', 'njihov', 'njihova', 'njihovo', 'njim', 'njima', 'njoj',
'nju', 'no', 'o', 'od', 'odmah', 'on', 'ona', 'oni', 'ono', 'ova', 'pa', 'pak',
'po', 'pod', 'pored', 'pred', 'prije', 's', 'sa', 'sam', 'samo', 'se', 'sebe',
'sebi', 'si', 'smo', 'ste', 'su', 'sve', 'svi', 'svog', 'svoj', 'svoja',
'svoje', 'svom', 'ta', 'tada', 'taj', 'tako', 'te', 'tebe', 'tebi', 'ti', 'tko',
'to', 'toj', 'tome', 'tu', 'tvoj', 'tvoja', 'tvoje', 'u', 'uz', 'vam', 'vama',
'vas', 'vaš', 'vaša', 'vaše', 'već', 'vi', 'vrlo', 'za', 'zar', 'će', 'ćemo',
'ćete', 'ćeš', 'ću', 'što']
```

[10]:
```
dataset = [word.lower() for word in biblija.words() if word not in punctuation
 ↪and word.isalpha() and word.lower() not in hr_stopwords]

fd = nltk.FreqDist(dataset)
print(fd.most_common(10))
fd.plot(10)
```

```
[('jahve', 3965), ('reče', 2076), ('bog', 1601), ('kralj', 1171), ('bijaše',
1139), ('dana', 1116), ('svoga', 1096), ('narod', 1087), ('sin', 1080),
('svojim', 1038)]
```



[10]: `<matplotlib.axes._subplots.AxesSubplot at 0x7fce1840c590>`

Postupak normalizacije teksta nerijetko zahtjeva segmentaciju (tokenizacija) i lematizaciju riječi. U tu svrhu predstavit ćemo kako nam SSF može koristiti.

3 Syntactic-Semantic Framework (SSF)

Poveznica: Sintaktičko-semantički okvir

Pristup: * Web sučelje (javni pristup i autentifikacija) * API pristup (API ključ potreban)

SSF predstavlja računalni model HR jezika koji u sebi sadrži: 1. tražilicu riječi (Riječi), 2. tražilicu rečenica (Rečenice), 3. stablo gramatičkih i semantičkih obilježja riječi i uzoraka (T-strukture WoS i SoW) 4. rječnik/tezarij (MSY, LEX, MWE, O-strukture).

4 API sučelje

SSF nudi mogućnost pristupa iz drugih alata koristeći SSF API.

4.1 Kategorije API funkcija:

Potpunu dokumentaciju možete pogledati na `funkcije.pdf`

- semantičke funkcije
- statističke funkcije
- morfološke funkcije
- WoS/SoW funkcije
- sintaktičke funkcije
- ...

[28]:
```
recenica = biblija.sents('NZ/04_Ivan.txt')[12]
```

[29]:
```
norm_recenica = (" ".join(recenica).lstrip('1 ').rstrip(' .')+'.').lower()
print(norm_recenica)
nlp_ssf = POSTagger.load('hr')
doc = nlp_ssf(norm_recenica)
#visualize
doc.render()
```

```
i riječ tijelom postade i nastani se među nama i vidjesmo slavu njegovu - slavu
koju ima kao jedinorođenac od oca - pun milosti i istine.
```

```
<IPython.core.display.HTML object>
```

Želimo sada pronaći leme riječi u `recenica`.

[30]:
```
ssf_hr = SSF()
```

[31]:
```
for rijec in recenica:
    print('{:10}   =>  {}'.format(rijec,ssf_hr.lemmatizer(rijec)))
```

```
I           =>  i
Riječ       =>  riječ
tijelom     =>  tijelo
```

```
postade      =>  postati
i            =>  i
nastani      =>  nastati
se           =>  sebe
među         =>  međa
nama         =>  mi
i            =>  i
vidjesmo     =>  vidjeti
slavu        =>  slava
njegovu      =>  njegov
-            =>  -
slavu        =>  slava
koju         =>  koji
ima          =>  imati
kao          =>  kao
Jedinorođenac   =>  Jedinorođenac
od           =>  od
Oca          =>  otac
-            =>  -
pun          =>  pun
milosti      =>  milost
i            =>  i
istine       =>  istina
.            =>  .
```

4.2 SSF sintaktičke funkcionalnosti

Želimo vidjeti koje vrste riječi imamo u našoj rečenici.

[34]:
```
tags = ssf_hr.gets(norm_recenica,'Vrsta riječi')
for rijec,tag in zip(recenica,tags):
    print('{:10} => {} '.format(rijec,tag))
```

```
I          => veznik
Riječ      => imenica
tijelom    => imenica
postade    => glagol
i          => veznik
nastani    => glagol
se         => zamjenica
među       => imenica
nama       => zamjenica
i          => veznik
vidjesmo   => glagol
slavu      => imenica
njegovu    => zamjenica
-          => nlo
slavu      => imenica
```

```
koju       => zamjenica
ima        => glagol
kao        => veznik
Jedinorođenac => nlo
od         => prijedlog
Oca        => imenica
-          => nlo
pun        => pridjev
milosti    => imenica
i          => veznik
istine     => imenica
.          => nlo
```

Za danu rečenicu želimo promijeniti glagolsko vrijeme.

[13]:
```
ssf_hr.change_tense('Vidim plavu kuću','aorist')
```

[13]:
```
'vidjeh plavu kuću'
```

Postoji mogućnost promjene broja za danu rečenicu.

[14]:
```
ssf_hr.plural('Vidim plave kuće')
```

[14]:
```
'vidimo plave kuće'
```

Lematizacija riječi u tekstu.

[19]:
```
ssf_hr.lemmatizer('Poslije ovoga dolaze novi počeci.')
```

[19]:
```
'poslije ovaj dolaziti nov početak .'
```

4.3 SSF semantičke funkcionalnosti

SSF koristi SoW oznake kako bi riječi nadopunio semanatičkim značajkama iz drugih dostupnih baza (CroWN).

[114]:
```
ssf_hr.synonym('raditi')
```

[114]:
```
"['izrađivati', 'uraditi', 'proizvoditi', 'stvoriti', 'proizvesti', 'tvoriti',
'praviti', 'činiti', 'učiniti', 'stvarati', 'izraditi', 'načiniti', 'napraviti',
'izgraditi', 'izgrađivati', 'konstruirati', 'graditi', 'sagraditi', 'biti
uključen', 'biti upaljen', 'obraditi', 'obrađivati', 'fabricirati',
'upravljati', 'dvoriti', 'služiti', 'poslužiti', 'posluživati', 'obnašati
dužnost', 'vršiti dužnost', 'funkcionirati', 'baviti se', 'vršiti', 'obavljati',
'izvršavati', 'provoditi', 'producirati', 'robijati', 'dirinčiti', 'mučiti se',
'crnčiti', 'kulučiti', 'rintati']"
```

Antonimi se mogu detektirati uz pomoć specifičnog kriterija.

[50]:
```
ssf_hr.antonym('mladić','dob')
```

[50]:
```
['starac']
```

[117]:
```
ssf_hr.antonym('mladić','spol')
```

[117]:
```
['djevojka']
```

Pristup kolokacijama ostvaruje se upitom nad MWE dijelom SSFa.

[51]:
```
ssf_hr.collocation('labav')
```

[51]:
```
'["labava carinska unija", "labava federacija", "labava granica", "labava
kompozicija", "labava konfederacija", "labava unija", "labave cijene", "labave
uzde", "labave veze", "labav re\\u017eim", "labava politi\\u010dka suradnja",
"labava veza me\\u0111u koalicijskim strankama"]'
```

Naprednija analiza sintaktičkih uzoraka omogućava prepoznavanje vrste zavisnih rečenica.

[118]:
```
ssf_hr.subordinate("Gdje radi, tamo i spava.")
```

[118]:
```
'mjesna'
```

[119]:
```
ssf_hr.subordinate("Kada budem doma, poslat ću ti poruku.")
```

[119]:
```
'vremenska'
```

U utvrđivanju relacija u rečenici vrlo bitnu ulogu predstavljaju trojke `subjekt-predikat-objekt`. U postupku je implementacija izvlačenja takvih trojki.

[142]:
```
# predikat?
ssf_hr.get_pred('Djed voli unuka.'.lower())
```

[142]:
```
'voli'
```

[143]:
```
# subjekt?
ssf_hr.get_subj('Djed nam je ispričao priču.'.lower())
```

[143]:
```
'djed'
```

[144]:
```
# objekt?
ssf_hr.get_obj('Radnik nam zida kuću.'.lower())
```

[144]:
```
'kuću'
```

U ovoj fazi sustav nudi prepoznavanje imenskih fraza na temelju semantičko-sintaksnih informacija koja riječ ima u SSFu.

```
[145]: ssf_hr.NER('Ivan je odlučio ići u Zagreb.')
```

```
[145]: ['Ivan => Osobno ime ', 'Zagreb => Toponim']
```

Detekcija metafora predstavlja veliki izazov u razumijevanju prirodnog jezika. U ovoj fazi SSF implementira postupak na temelju *jezgrenih glagolima*, drugim riječima, riječ ima specifičan skup glagola uz koje najčešće dolazi.

```
[64]: ssf_hr.detect_metaphor('Njihov predsjednik je snažni lav.')
```

```
[64]: 'predsjednik je lav'
```

SSF nudi mogućnost detekcije metonimija koristeći sličan pristup.

```
[20]: ssf_hr.detect_metonymy('Kruna je dala svoj proglas.')
```

```
[20]: 'Kruna dala'
```

Detekcija *n*-grama se temelji na klizućem prozoru kroz danu rečenice specifične dimenzije.

```
[21]: ssf_hr.Ngrams("Čitala je njegove pjesme kao da prvi put otkriva snagu pjesničke␣
      ↪riječi", 2, 3)
```

```
[21]: "[['Čitala', 'je'], ['Čitala', 'njegove'], ['je', 'njegove'], ['je', 'pjesme'],
      ['njegove', 'pjesme'], ['njegove', 'kao'], ['pjesme', 'kao'], ['pjesme', 'da'],
      ['kao', 'da'], ['kao', 'prvi'], ['da', 'prvi'], ['da', 'put'], ['prvi', 'put'],
      ['prvi', 'otkriva'], ['put', 'otkriva'], ['put', 'snagu'], ['otkriva', 'snagu'],
      ['otkriva', 'pjesničke'], ['snagu', 'pjesničke'], ['snagu', 'riječi'],
      ['pjesničke', 'riječi']]"
```

4.4 SpaCy okvir

U ovom dijelu radionice spomenut ćemo neke vanjske moderne biblioteke za obradu prirodnog jezika koje su uključene (ili barem u postupku) u SSF.

Besplatna biblioteka otvorenog koda za obradu prirodnog jezika spacy.io. Podržava većinu svjetskih jezika, no HR nažalost ne direktno. Model moramo sami izgraditi (dan je u sklopu ove radionice).

Podržava sljedeće mogućnosti:

ime	opis
segmentacija	tokenizacija riječi, rečenica, odlomaka
POS označavanje	označavanje riječi
Parsiranje zavisnosti	pridruživanje zavisničkih ovisnosti
lematizacija	lematizacija
NER	prepoznavanje imenskih fraza
...	ostalo ...

HR model je natreniran s **Croatian UD treebank** temeljen na SETimes-HR korpusu GitHub.

[22]:
```
sentence = 'Maja stavlja knjigu na policu.'
```

4.4.1 SpaCy | Univerzalne zavisnosti

Učitajmo ranije natrenirani model za prepoznavanje univerzalnih zavisnosti za HR u `spacy` sučelje.

[23]:
```
import spacy
# load HR UD model
nlp = spacy.load('models/hr_ud')
```

[24]:
```
doc = nlp(sentence)
# vizualize
displacy.render(doc,style='dep',jupyter=True)
```

```
<IPython.core.display.HTML object>
```

4.5 SpaCy | Prepoznavanje imenskih fraza

[26]:
```
sentence = 'Jučer nam je Marko javio da more nije dobro u Rijeci.'
nlp = spacy.load('models/hr_ner')
doc = nlp(sentence)

# vizualize
displacy.render(doc,style='ent',jupyter=True)
```

```
<IPython.core.display.HTML object>
```

5 Distribuirana semantika | Word2Vec

- Na principima *distribuirane semantike* koja smatra da značenje neke riječi saznajemo iz riječi s kojima često dolazi razvili su se **neuronski modeli jezika**.

- Modeli su za proizvoljnu riječ posljedično iz vokabulara pridruživali *vektore riječi*.
- Vektorskim operacijama nad tim vektorima može se pokazati da su ovi modeli mogu modelirati neka semantička svojstva.

Vratimo se sada na naš `biblija` korpus. Izgradit ćemo vektore riječi za taj korpus.

[29]:
```
biblija.sents()
```

[29]:
```
[['Evanđelje', 'po', 'Marku'], ['I', '.', 'PRIPRAVA', 'ZA', 'ISUSOVO',
'DJELOVANJE'], ...]
```

[30]:
```
'raj' in biblija.words()
```

[30]:
```
True
```

Normalizacija i čišćenje teksta.

[31]:
```
from string import punctuation

dataset = [[word.lower() for word in sent if word not in punctuation and word.
 ↪isalpha()] for sent in biblija.sents()]
```

Word2Vec reprezentacija riječi.

[58]:
```
from gensim.models import Word2Vec

# definicija modela
model = Word2Vec(dataset, min_count=5, size=200) # zanemari rijeci koji se␣
 ↪pojavljuju < 5, dim. vokabualara 200
model.save('models/word2vec/bible_hr')
# ucitaj napravljeni model
#model = Word2Vec.load('models/word2vec/bible_hr')
word_vectors = model.wv
# spremi vektore
word_vectors.save_word2vec_format('models/word2vec/bible_hr_org','models/
 ↪word2vec/bible_hr_voc')
```

Sličnost pojmova.

[59]:
```
word_vectors.most_similar(['bog'],topn=5)
```

[59]:
```
[('tvoj', 0.9154353141784668),
 ('izraelov', 0.9141589403152466),
 ('naš', 0.9028884172439575),
 ('jahve', 0.9011515974998474),
 ('vaš', 0.9009150266647339)]
```

[60]:
```
word_vectors.most_similar(['žena'],topn=5)
```

[60]:
```
[('djevojci', 0.9254284501075745),
 ('muž', 0.9252440929412842),
 ('nerotkinja', 0.9145063161849976),
 ('majka', 0.9016597867012024),
 ('kći', 0.8958909511566162)]
```

[111]:
```
word_vectors.doesnt_match("gospodin bog rijeka svetac".split())
```

```
/opt/anaconda3/lib/python3.7/site-packages/gensim/models/keyedvectors.py:877:
FutureWarning: arrays to stack must be passed as a "sequence" type such as list
```

```
or tuple. Support for non-sequence iterables such as generators is deprecated as
of NumPy 1.16 and will raise an error in the future.
  vectors = vstack(self.word_vec(word, use_norm=True) for word in
used_words).astype(REAL)
```

[111]:
```
'rijeka'
```

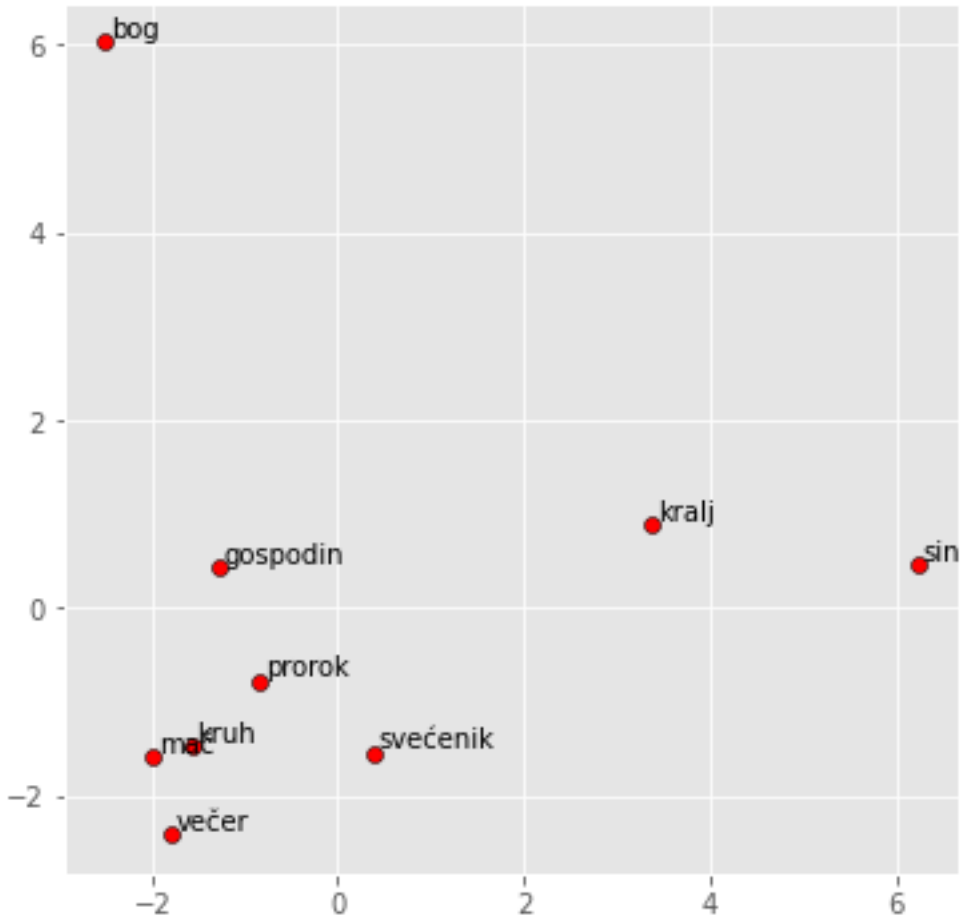
[112]:
```
def display_pca_scatterplot(model, words=None, sample=0):
    if words == None:
        if sample > 0:
            words = np.random.choice(list(model.vocab.keys()), sample)
        else:
            words = [ word for word in model.vocab ]

    word_vectors = np.array([model[w] for w in words])

    twodim = PCA().fit_transform(word_vectors)[:,:2]

    plt.figure(figsize=(6,6))
    plt.scatter(twodim[:,0], twodim[:,1], edgecolors='k', c='r')
    for word, (x,y) in zip(words, twodim):
        plt.text(x+0.05, y+0.05, word)
```

[114]:
```
display_pca_scatterplot(word_vectors,␣
↪words=['kralj','gospodin','prorok','svećenik','kruh','sin','bog','mač','večer'])
```



5.1 Zaključno

SSF za HR jezik nastoji ponuditi cjelovit okvir za sintaksnu i semantičku analizu hrvatskog jezika što uparenu sa modernim metodama obrade prirodnog jezika može uvelike doprinjeti računalnom razumijevanju jezika. Daljnjim razvojem SSFa u smislu integracije dijalektoloških kompoenenti hrvatskog jezika omogućit će se provođenje ovakve korpusne analize i za naše dijalekte.

[ ]: