

Seminarski rad u okviru kursa Istraživanje Podataka 1

Detekcija lažnih oglasa za posao

Aleksandar Šmigić

mi19028@alas.matf.bg.ac.rs

Matematički Fakultet, Univerzitet u Beogradu

Sadržaj

Uvod	3
O skupu podataka	3
Eksplorativna analiza	4
Pretprocesiranje	10
Rad sa nedostajućim vrednostima	10
Obrada elemenata van granica	10
Pretprocesiranje za klasifikaciju	10
Pretprocesiranje za klasterovanje	10
Pretprocesiranje za pravila pridruživanja	11
Klasifikacija	11
Komplementarni Naivni Bajes	11
Odabir hiper-parametara	11
Rezultati	12
Gradijentno Pojačavajuća Drveta Odlučivanja	12
Odabir hiper-parametara	12
Poređenje	14
Klasterovanje	15
K sredina	15
Hijerarhijsko klasterovanje	17
Pravila pridruživanja	19
Zaključak	20

Uvod

Snažnom ekspanzijom internet poslovanja dolazi do promene u načinu oglašavanja poslodavaca. Korišćenjem internet oglasa značajno se smanjuje vreme potrebno za pronalaženje novih kandidata, tako što se oglas prikaže većem broju kandidata nego što bi to bio slučaj sa tradicionalnim metodama. Takvo okruženje je idealno za prevarante, zbog čega je problem detekcije lažnih oglasa važan.

O skupu podataka

Ovaj rad bavi se rešavanjem problema u oblasti prirodnog govora odnosno NLP-a (eng. *Natural Language Processing*) koji se zasnivaju na skupu podataka: “Real or Fake Jobposting” sa sajta Kaggle. Nakon detaljne eksplorativne analize podataka, implementirani su modeli za klasifikaciju zasnovani na Naivnom Bajesu i ansamblu gradijentno pojačavajućih drveta odlučivanja, klasterovanje pomoću K sredina i hijerarhijskih metoda, i pronalaženje pravila pridruživanja pomoću Apriori algoritma.

Skup podataka sadrži 17880 instanci sa sledećim kolonama:

Ime kolone	Tip podatka	Opis
job_id	string	Identifikator posla
title	string	Titula
location	string	Lokacija
department	string	Odsek firme
salary_range	string	Raspon plata
company_profile	string	Profil kompanije
description	string	Opis posla
requirements	string	Zahtevi
benefits	string	Pogodnosti
telecommuting	bool	Rad na daljinu
has_company_logo	bool	Da li ima logo kompanije
has_questions	bool	Da li ima pitanja
employment_type	string	Tip zaposlenja
required_experience	string	Neophodno iskustvo
required_education	string	Neophodno obrazovanje
industry	string	Industrija
function	string	Funkcija
fraudulent	bool	Da li je lažan oglas

Eksplorativna analiza

Delimo dataset u odnosu 2:1 na `training_set` i `test_set` koristeći stratifikaciju. Dalja analiza je izvršena **isključivo** nad `training_set`.

Na Figure 1 i Figure 3 prikazano je prvih nekoliko instanci i kolona skupa podataka, kao i njihove osnovne informacije:

	job_id	title	location	department	salary_range	company_profile	description	requirements	benefits	telecommuting
0	15604	Web Application Developer (NodeJS)	US, OR, Portland	NaN	NaN	Can data be a thing of beauty? We think so. At ...	About Seabourne ConsultingCan data be a thing ...	Responsibilities:The Web Application Developer...	Location: Portland, OR. You must reside in the...	1
1	1946	Graduates: English Teacher Abroad (Conversatio...	US, NY, Ithaca	NaN	NaN	We help teachers get safe & secure jobs ab...	Play with kids, get paid for it :)Love travel...	University degree required. TEFL / TESOL / CEL...	See job description	0
2	7513	English Teacher Abroad (Conversational)	US, CA, Chico	NaN	NaN	We help teachers get safe & secure jobs ab...	Play with kids, get paid for it.Vacancies in A...	University degree required. TEFL / TESOL / CEL...	See job description	0
3	17564	URGENT Job Full Time & Part Time, Cash Pay.	US, CA, Los Angeles	NaN	NaN	NaN	URGENT Job Full Time & Part Time, Cash Pay...	No any experience required.	Perfect for everyone then start immediately.	0
4	8498	Android Developer	DE, BE, Berlin	Development	NaN	NaN	(#URL_0252efddcbc4b8f51969fca7b0545...	You dream in Java and you're proficient in And...	The Web is changing and becoming more interact...	0

Figure 1: `df.head()`

Prilikom klasifikacije nam je cilj da predvidimo binarni atribut `fraudulent`. Na Figure 2 vidimo da je naš skup podataka veoma **nebalansiran** i da lažni oglasi čine samo oko 5% celokupnog skupa podataka.

Real count: 17014 i.e. 95.1566%
Fake count: 866 i.e. 4.8434%

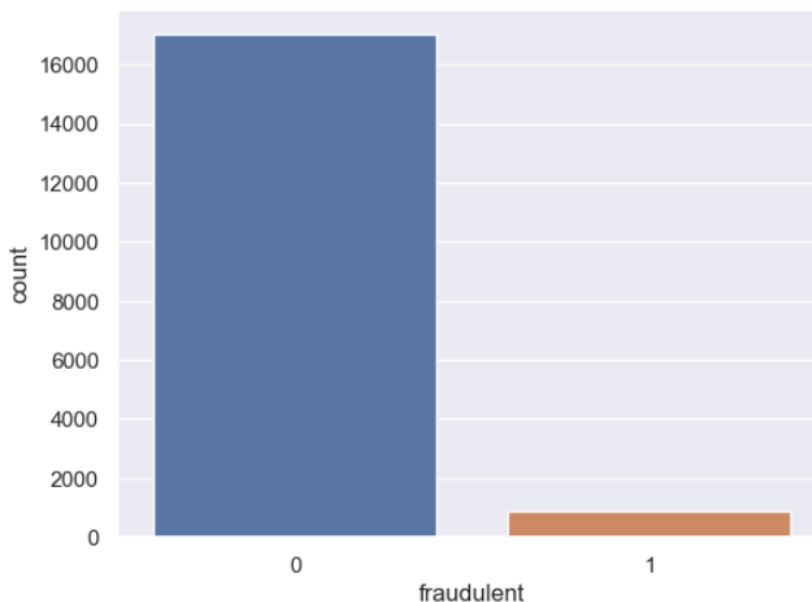


Figure 2: Broj pravih i lažnih oglasa u skupu podataka

Potrebno je napomenuti da definišemo distinkciju između termina kolona i atribut, gde kolone predstavljaju sirove neobrađene podatke, dok atributi predstavljaju podatke pogodne za primenu modela nad njima.

```

RangeIndex: 11979 entries, 0 to 11978
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  -
0   job_id                 11979 non-null  int64
1   title                  11979 non-null  object
2   location               11748 non-null  object
3   department             4264 non-null   object
4   salary_range           1918 non-null   object
5   company_profile        9749 non-null   object
6   description            11978 non-null  object
7   requirements           10187 non-null  object
8   benefits               7118 non-null   object
9   telecommuting          11979 non-null  int64
10  has_company_logo       11979 non-null  int64
11  has_questions          11979 non-null  int64
12  employment_type        9685 non-null   object
13  required_experience     7254 non-null   object
14  required_education     6561 non-null   object
15  industry               8732 non-null   object
16  function               7656 non-null   object
17  fraudulent             11979 non-null  int64

```

Figure 3: df.info()

Na osnovu Figure 3 vidimo da određene kolone imaju značajan broj nedostajućih vrednosti, zbog čega ih na Figure 4 posmatramo.

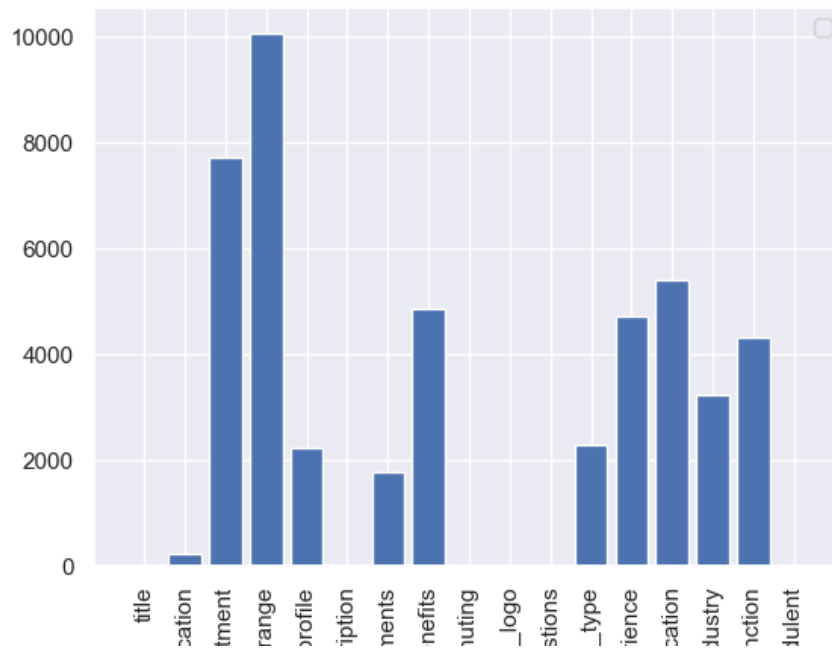


Figure 4: Bar plot broja nedostajućih vrednosti za svaki atribut

Zaključujemo da nedostajuće vrednosti postoje isključivo kod kolona koje predstavljaju sirove tekstualne podatke.

U koloni `location` su enkodirana tri atributa `country_code`, `state_code`, `city` koje razdvajamo. Učitavamo pomoćni dataset `country-list`, dostupan na [adresi](#), koji će nam pomoći u dekodiranju skraćenih naziva za `country_code` i ulepšati prikaz.

Nakon navedenih transformacija, slučajni uzorak od 5 instanci sa kolonama vezanih za lokaciju prikazan je na Figure 5.

	city	state_code	country
7887	Fairfield	CA	United States
7736	Athens	I	Greece
9924	Manchester	MAN	United Kingdom
9898	Boone	NC	United States
375	Fort Mill	SC	United States

Figure 5: Slučajan uzorak transformisanog skupa podataka

Ranije-pomenutu transformaciju smo izvršili kako bismo dobili bolji uvid u naše podatke.

Na narednim graficima Figure 6 i Figure 7, možemo da primetimo da Sjedinjene Američke Države dominiraju i po pitanju broja ukupnog broja oglasa i lažnih oglasa u odnosu na druge države.

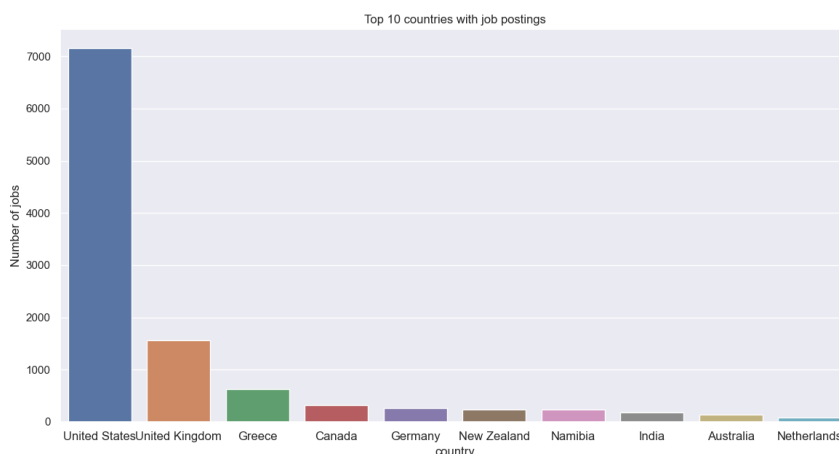


Figure 6: Ukupan broj oglasa po državi

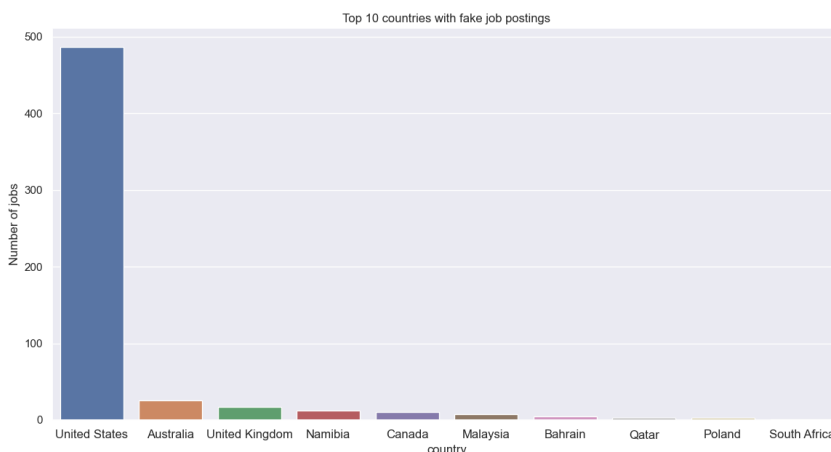


Figure 7: Broj lažnih oglasa po državi

Na Figure 8 vidimo da su Teksas, Kalifornija i Nju Jork tri najčešće države u kojima se javljaju lažni oglasi. Zajedno sa **NONE**, odnosno nedostajućim vrednostima za kolonu **state_code**, predstavljaju skoro tačno polovinu svih lažnih oglasa.

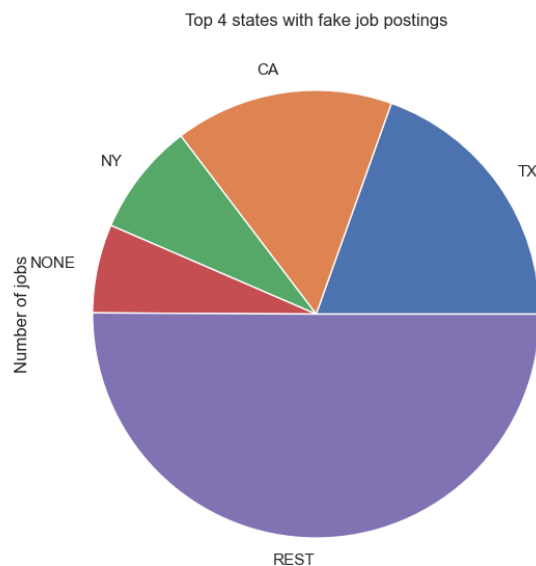


Figure 8: Broj lažnih oglasa po državi u SAD

U jupyter svesci u kojoj je vršena eksplorativna analiza i pretprocesiranje moguće je naći još neke interesantne informacije o samom skupu podataka. Imajući u vidu da ćemo za numeričko enkodiranje sirovog teksta koristiti TF-IDF (eng. *Term Frequency Inverse Document Frequency*) matricu, koja će se ujedno i postarati da u zavisnosti od učestalosti pojavljivanja otežava vrednost reči, nećemo preduzimati bilo kakvo *ad hoc* odbacivanje skupa reči.

S obzirom da velika količina poslovnih titula i termina se sastoji iz dve reči, hoćemo da proverimo koliko se reči obično nalazi u naslovu.

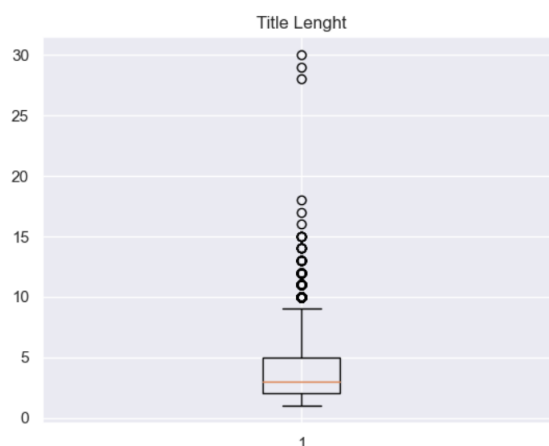


Figure 9: Box plot za broj reči u rečenici

Primećujemo da je medijana skoro 4, što se ne uklapa u našu početnu pretpostavku da će u naslovu oglasa biti samo ime titule.

count	11979.000000
mean	3.907672
std	2.152825
min	1.000000
25%	2.000000
50%	3.000000
75%	5.000000
max	30.000000

Figure 10: Broj lažnih oglasa po državi u SAD

U nastavku će biti prikazani dijagrami određenih interesantnih osobina podataka.

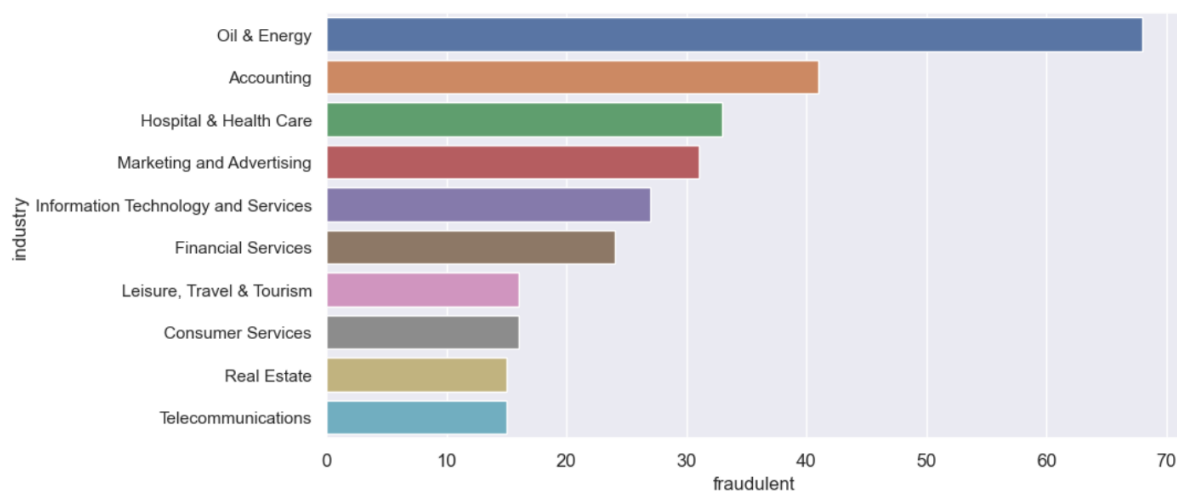


Figure 11: Tip industrije u lažnim oglasim

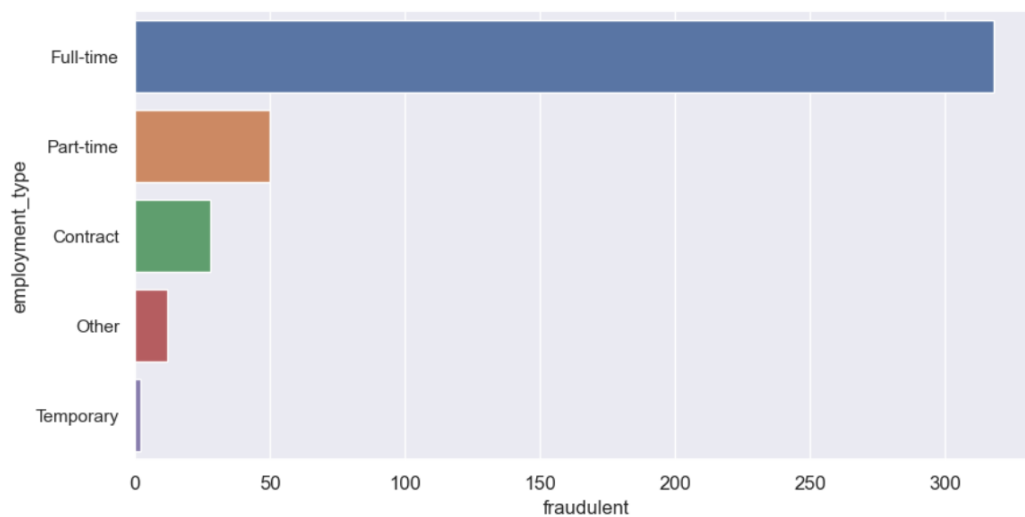


Figure 12: Tip zaposlenja u lažnim oglasim

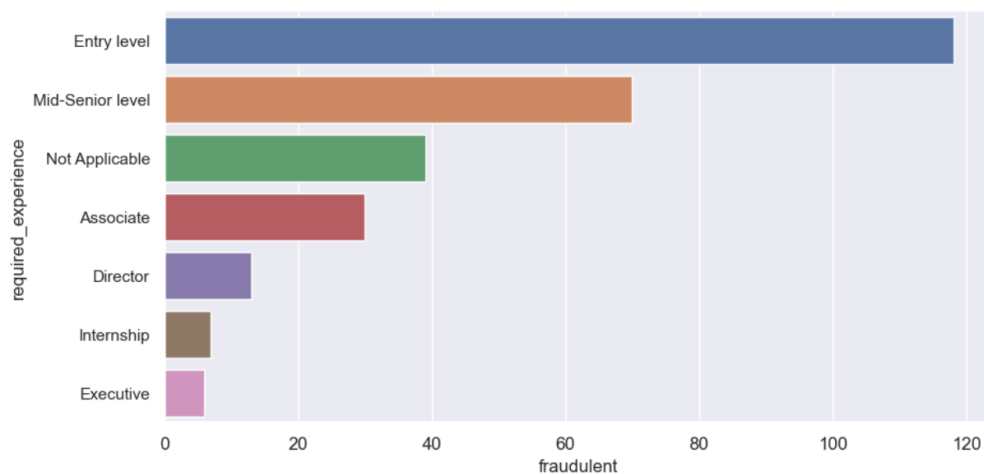


Figure 13: Neophodno iskustvo u lažnim oglasim

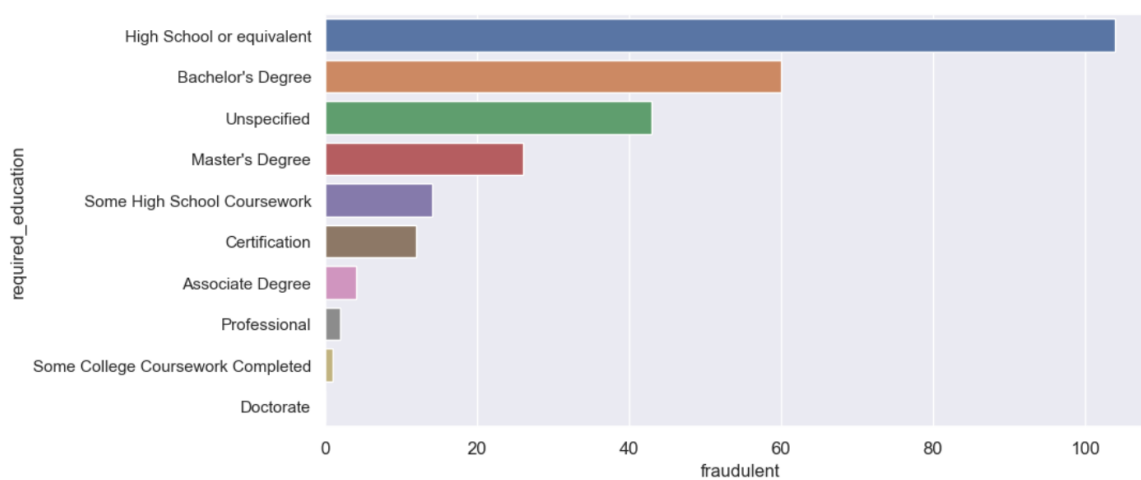


Figure 14: Minimalno obrazovanje u lažnim oglasim

Pretprocesiranje

Rad sa nedostajućim vrednostima

Kolona `job_id` je redundantna, stoga je uklanjamo.

S obzirom da TFIDF matrica predstavlja vreću reči, gde redosled svakako ne postoji, sve tekstualne kolone spajamo u jednu kolonu `text` radi jednostavnije implementacije. Kolone koje spajamo ujedno i popunjavamo praznom niskom u slučaju nedostajućih vrednosti.

Obrada elemenata van granica

Kako bismo naše podatke pripremili za enkodiranje u TFIDF matricu, radimo sledeće transformacije nad njima:

1. Transformisanje u mala slova
 - ugrađen metod `.lower()` u `str`
2. Uklanjanje interpunkcijskih znakova
 - paket `re` za regularne izraze
3. Tokenizacija
 - paket `nltk`
4. Lematizacija
 - paket `nltk`
5. Izbacivanje stop reči
 - paket `nltk`
6. Izbacivanje predugačkih i prekratkih reči
 - zadržane su reči dužine između 3 i 30 karaktera

Pretprocesiranje za klasifikaciju

Kao što je ranije pomenuto koristićemo TFIDF matricu kako bismo enkodirali tekst. Ograničavamo veličinu korpusa na 10^4 najčešćih reči, kako bismo mogli u razumnom vremenu da vršimo dalja računanja na raspoloživom hardveru.

Naizgled deluje da bi bilo korisno koristiti n -grame dužine 2, ali smo se u eksplorativnoj analizi uverili da je medijalni broj reči u naslovu oglasa 4. Radi jednostavnosti ipak posmatramo samo individualne reči prilikom enkodiranja u TFIDF matricu, ali beležimo da je detaljnijom eksplorativnom analizom verovatno moguće načiniti bolji izbor.

Pretprocesiranje za klasterovanje

Prilikom pripreme za klasterovanje spajamo skupove za treniranje i testiranje. Osim što ponavljamo isto pretprocesiranje kao kod klasifikacije, bitno je istaći da sada TFIDF matricu računamo nad **celim skupom podataka**, dok kod klasifikacije to radimo samo nad skupom za trening.

Nakon transformacija, naš skup podataka je oblika (17880, 10000) zbog čega rizikujemo da se susretnom sa *prokletstvom dimenzionalnosti* (eng. *curse of dimensionality*). Kako bismo to izbegli pribegavamo tehnici **skrivenih semantičkih analiza** odnosno LSA (eng. *Latent Semantic Analysis*), koja nam služi za redukovanje broja dimenzija.

Kao i PCA (eng. *Principal Component Analysis*), LSA se zasniva na SVD (eng. *Singular Value Decomposition*), s tim što se LSA primenjuje nad TFIDF matricom.

Nakon primene LSA vršimo normalizaciju kako bismo smanjili uticaj šuma i imali vektore u istoj skali. Kada bismo hteli da zadržimo varijansu od oko 75%, onda bi bilo dovoljno da redukujemo

dimenzije na 1000, što smo empirijski potvrdili. Imajući u vidu da želimo da lepo vizualizujemo rezultate, biramo da redukujemo na 2 dimenzije, čime čuvamo samo 4% originalne varijanse.

Pretprocesiranje za pravila pridruživanja

Slično kao i kod pretprocesiranja za klasterovanje, spajamo i transformišemo trening i test skupove podataka. Originalno smo pokušali da izvezemo kompletnu matricu sa 10000 kolona i 17000+ redova, ali se to ispostavilo kao preveliki zalogaj za dostupan hardver. Umesto korpusa od 10^4 reči, pravimo rečnik od 10^3 reči. Uzorkovanjem značajno smanjujemo drugu dimenziju na oko $6 \cdot 10^3$. Na kraju matricu transformišemo u binarnu reprezentaciju, pogodnu za primenu pravila pridruživanja.

Klasifikacija

Komplementarni Naivni Bajes

Naivni Bajes predstavlja jedan od najčešće korišćenih modela za klasifikaciju teksta. U ovom radu korišćen je Komplementarni Naivni Bajes, objavljen od strane *Rennie et al. (2003)*, koji predstavlja nadogradnju na multinomijalni Naivni Bajes, optimizovan za veoma disbalansirane skupove podataka.

Odabir hiper-parametara

Glavni hiper-parametar ovog klasifikatora je α koje se naziva aditivni parametar ugađivanja koji služi za **regularizaciju**. Korišćena je unakrsna validacija kako bismo našli što optimalniji izbor hiper-parametara. Potrebno je naglasiti da je neophodno biti pažljiv sa odabirom metrike kojom se pretaga vodi, s obzirom da imamo veoma disbalansiran skup podataka. Stoga, podrazumevanu **tačnost** menjamo sa **površinom ispod ROC krive** (eng. *ROC Area Under the Curve*).

Najbolje rezultate na validacionom skupu postizemo sa odabirom parametra $\alpha = 0.01$. Na Figure 15 prikazano je kretanje srednje vrednosti AUC metrike u zavisnosti od α tokom unakrsne validacije.

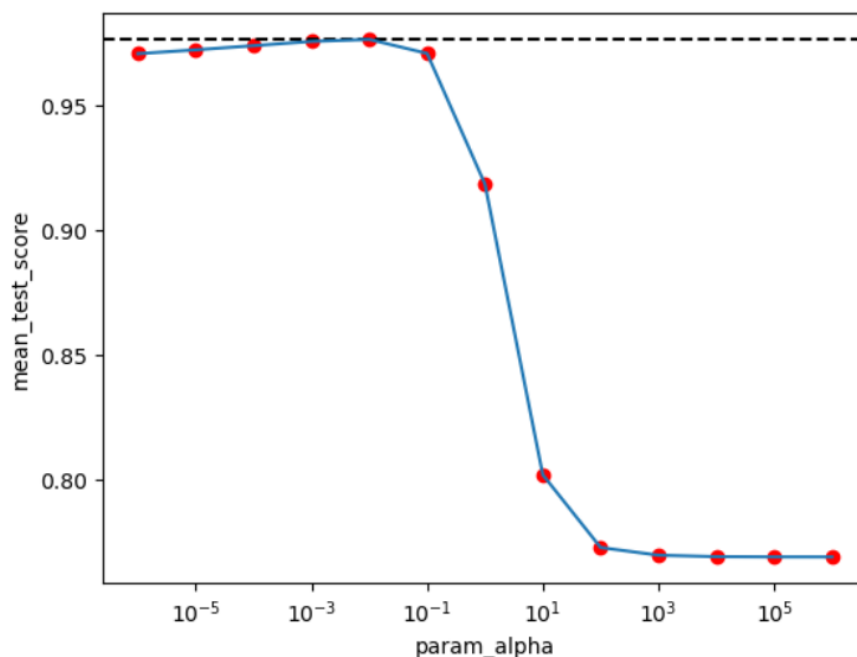


Figure 15: srednja vrednost AUC u zavisnosti α tokom unakrsne validacije

Rezultati

Na osnovu matrice konfuzije na Figure 16 možemo da zaključimo da je ukupan broj pogrešno predviđenih instanci 209.

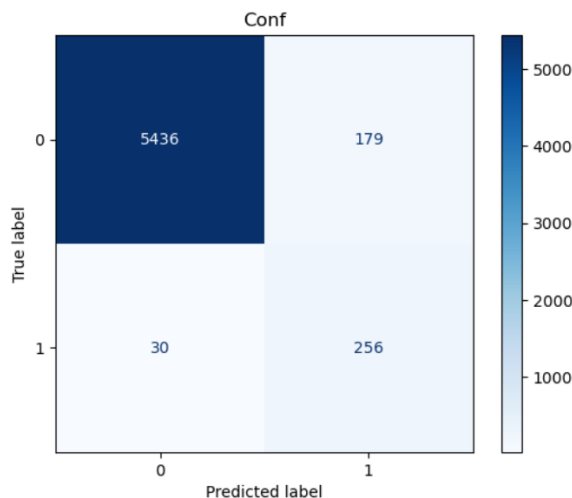


Figure 16: Matrica konfuzije za Komplementarni Naivni Bajes

U navedenoj tabeli prikazane su metrike za evaluaciju nad **test** skupom.

	Preciznost	Odziv	F1 ocena
0	0.99	0.97	0.98
1	0.59	0.90	0.71
tačnost			0.96
makro sredina	0.79	0.93	0.85
težinska sredina	0.97	0.96	0.97

Pre svega na osnovu F1 ocene možemo da zaključimo da model radi zadovoljavajuće.

Gradijentno Pojačavajuća Drveta Odlučivanja

Kao alternativa Naivnom Bajesu koji je pokazao dobre rezultate, koristićemo ansambal drveta odlučivanja. Konkretno, koristićemo gradijentno pojačavajuća drveta odlučivanja bazirana na histogramima (eng. *Histogram-based Gradient Boosted Decision Tree*).

Implementacija u okviru `scikit-learn` biblioteke je inspirisana Majkrosoftovom implementacijom u okviru biblioteke `LightGBM`. Glavna prednost koju implementacija zasnovana na histogramima donosi je značajno brže vreme treniranja za velike skupove podataka (većih od 10^4 , što naš skup podataka jeste) u odnosu na alternativu `GradientBoostingClassifier`.

Odabir hiper-parametara

Kao i kod prethodnog klasifikatora, i ovde koristimo unakrsnu validaciju za pronalaženje optimalnih vrednosti hiper-parametara.

Hiper parametri koje menjamo:

1. `max_iter` - maksimalni broj pojačavanja tj. novih stabala
2. `learning_rate` - stopa učenja novih pojačanih stabala
3. `min_samples_leaf` - najmanji broj uzoraka potreban da bi čvor bio list

Iz nepoznatog razloga `GridSearchCV` nikada ne završi. S obzirom da treniranje podrazumevane konfiguracije klasifikatora traje nekoliko minuta, trebalo bi da i treniranje nekoliko drugačije konfigurisanih klasifikatora traje manje od 5 sati. Umesto istraživanja gde bi greška mogla da se nađe u implementaciji, nastavice sa istraživanjem podataka.

Podrazumevane vrednosti hiper-parametara su `max_iter=100`, `learning_rate=0.1` i `min_samples_leaf=20`.

Na osnovu matrice konfuzije na Figure 17 možemo da zaključimo da je ukupan broj pogrešno predviđenih instanci 71.

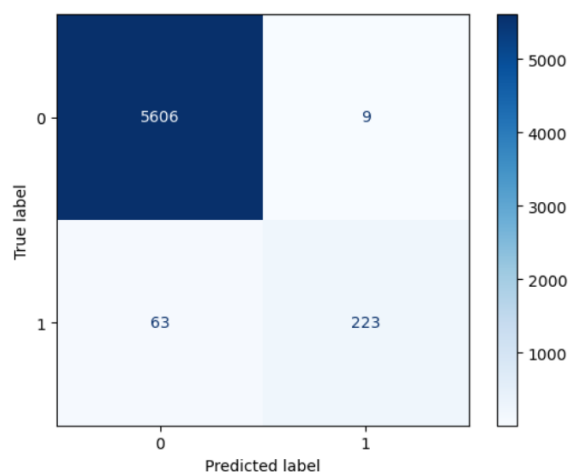


Figure 17: Matrica konfuzije za gradijentom pojačana drveća odlučivanja bazirana na histogramima

U navedenoj tabeli prikazane su metrike za evaluaciju nad **test** skupom.

	Preciznost	Odziv	F1 ocena
0	0.99	1.00	0.99
1	0.96	0.78	0.86
tačnost			0.96
makro sredina	0.98	0.89	0.93
težinska sredina	0.99	0.99	0.99

Poređenje

Na prvi pogled deluje da je ansambl metoda uspešnija, ali kada pogledamo informacije grafički i sračunamo površinu ispod krive ROC (eng. *Receiver Operating Characteristic*) vidimo da je zapravo Komponentni Naivni Bajes balansiraniji klasifikator i po ovoj metrici bolji.

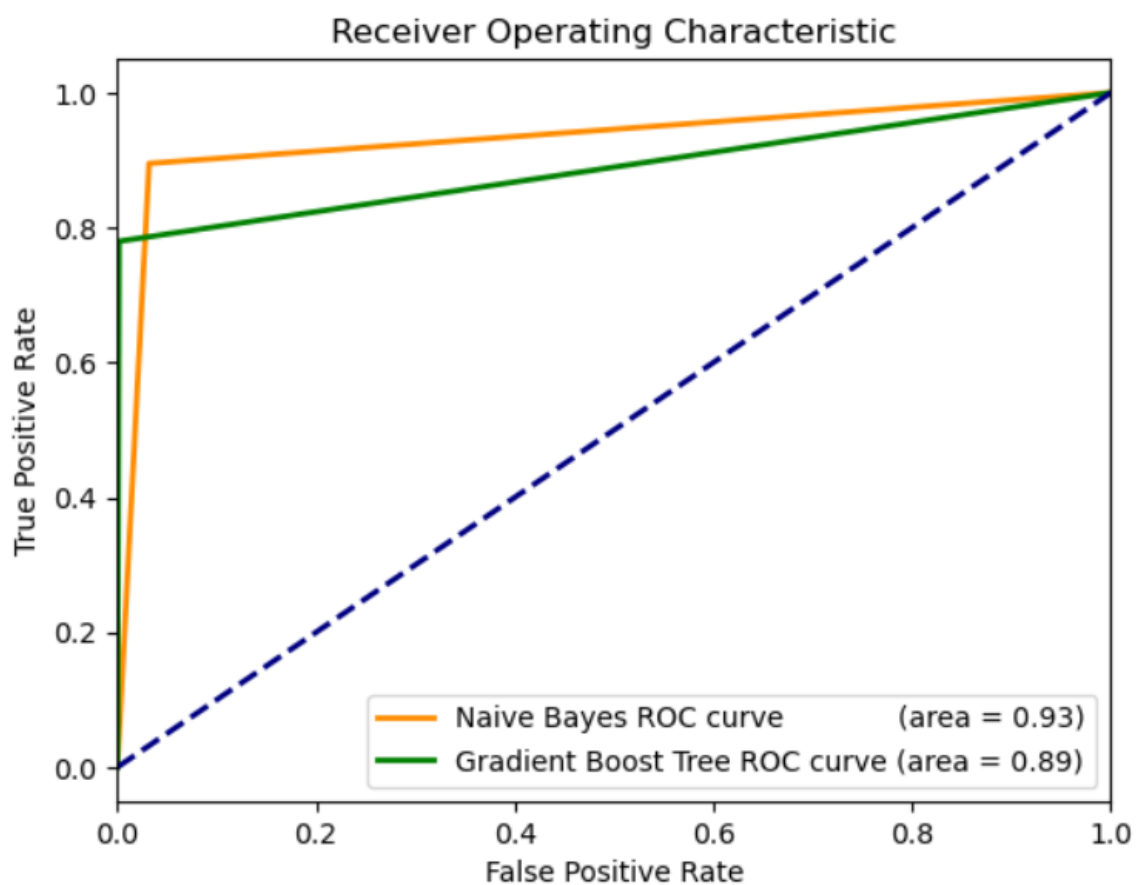


Figure 18: ROC

Klasterovanje

Skup podataka korišćen u ovom radu nije najpogodniji za klasterovanje zbog svoje visoke dimenzionalnosti. U odeljku za preprocesiranje smo obrazložili rezon za redukovanje na 2 dimenzije pomoću LSA.

K sredina

Korišćen je algoritam K sredina sa k-mean++ inicijalizacijom koja ravnomernije raspoređuje centroide po prostoru. Na Figure 19 prikazano je isprobavanje različitih vrednosti hiperparametra k . Plavi kružić označava centroidu klastera.

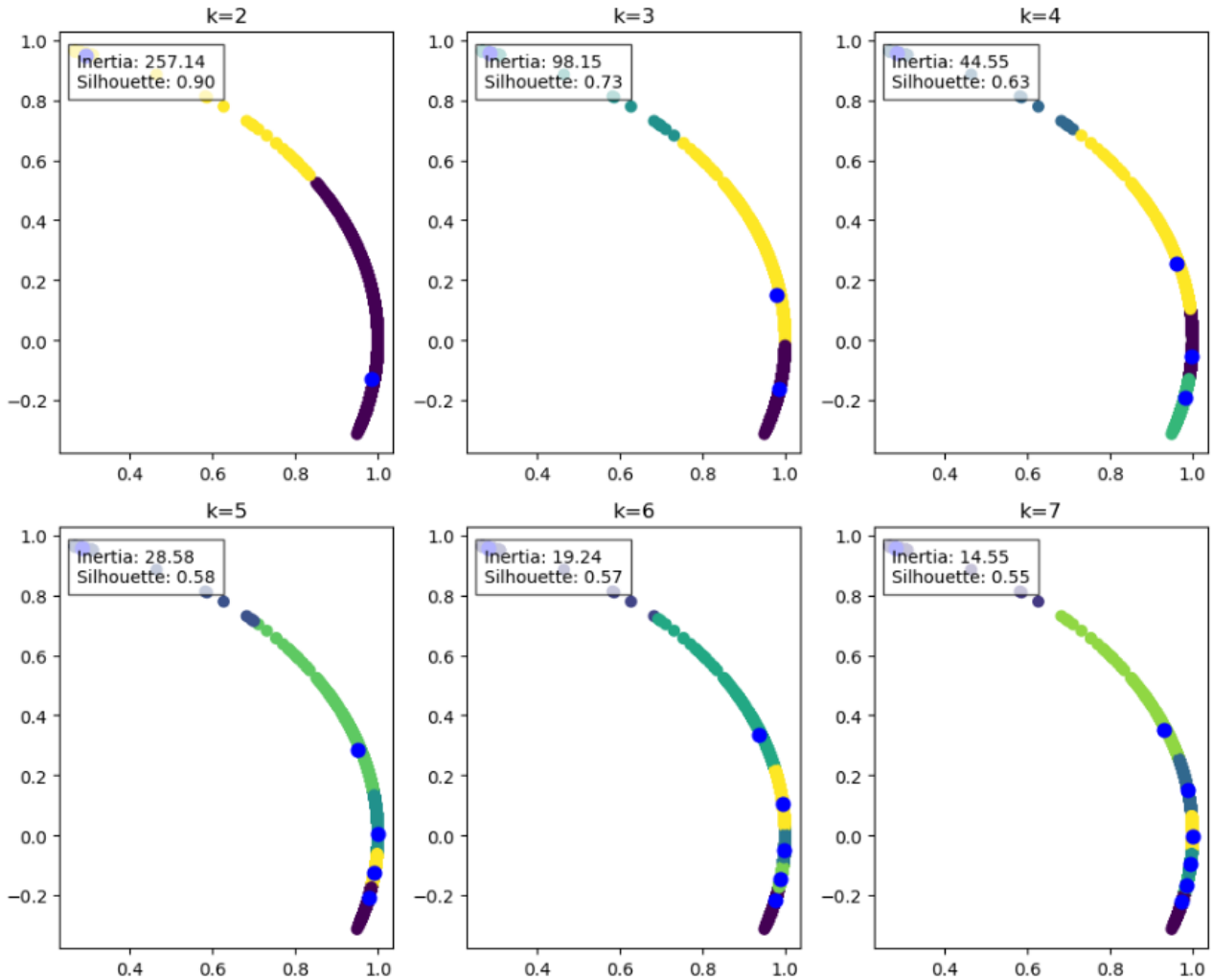


Figure 19: Klasterovanje za različite vrednosti k

Zaključujemo da nam je najbolji model gde je $k = 2$ jer je u tom slučaju silhouette metrika najveća.

Prikazano je i kretanje inercije i siluete u zavisnosti od k

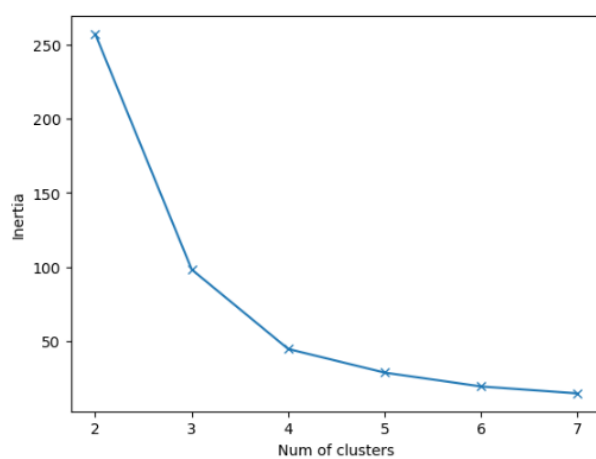


Figure 20: Inercija u zavisnosti od k

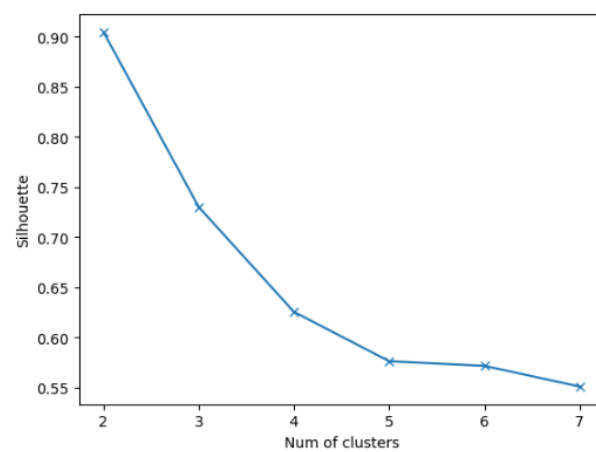


Figure 21: Silueta u zavisnosti od k

Hijerarhijsko klasterovanje

Hijerarhijsko klasterovanje podržava ugnježdene klastere koji se dobijaju sukcesivnim spajanjem. U potrazi za najboljom strategijom za spajanje klastera isprobavamo *complete*, *average*, *ward* i *single* zajedno sa različitim vrednostima za hiper-parametar k .

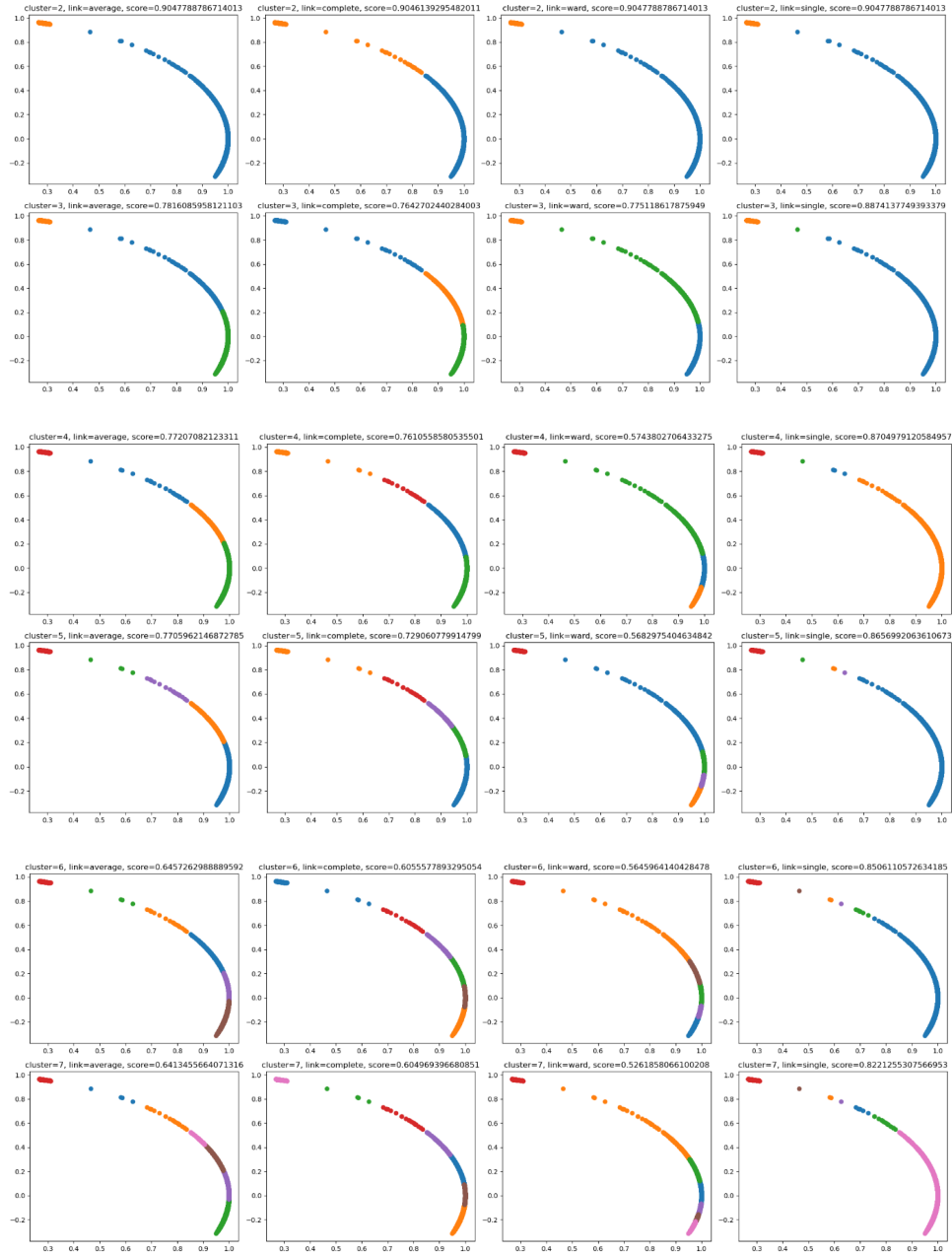


Figure 22: Hijerarhijsko klasterovanje

Najbolji model za siluetu ima rezultat 0.905, $k = 2$ i koristi minimizovanje prosečne udaljenosti između opažanih parova klastera, za spajanje.

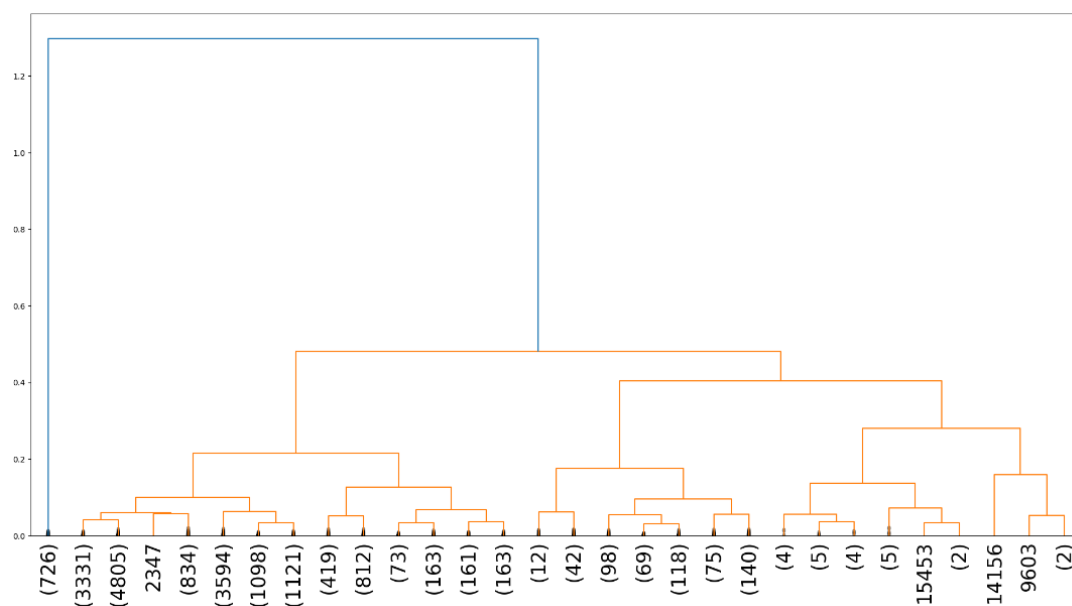


Figure 23: Dendrogram hijerahijskog modela

Na Figure 23 je prikazan dendrogram našeg modela.

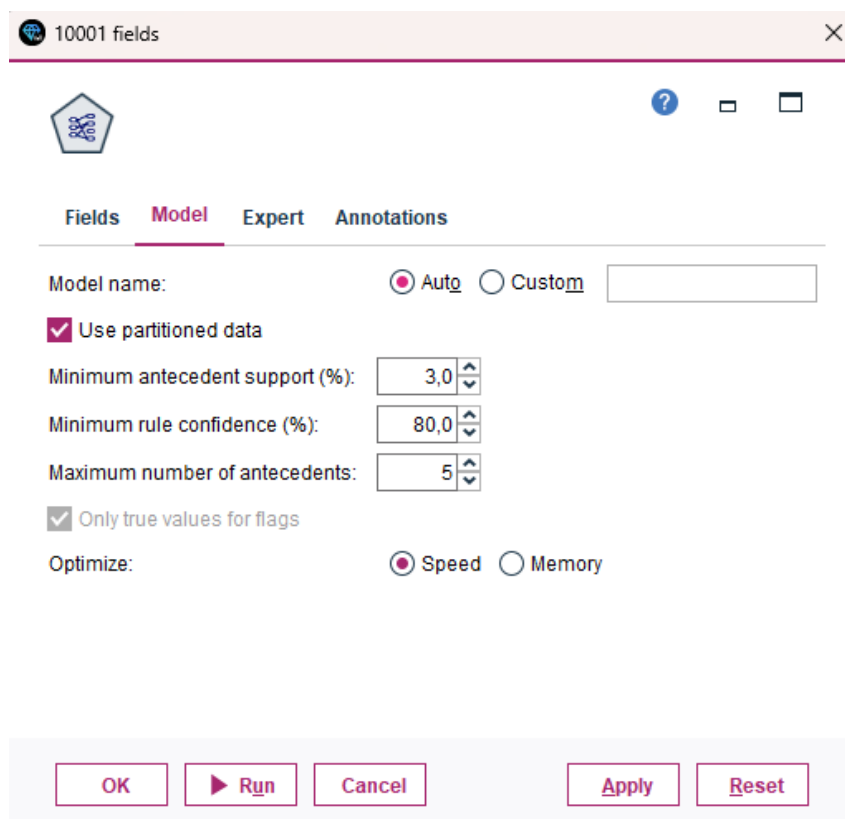
Imajući u vidu veliki gubitak informacija do kog je došlo prilikom dimenzione redukcije, nema previše poente pokušavati da veštački interpretiramo skrivena značenja pronađenih klastera.

Pravila pridruživanja

Korišćen je **apriori algoritam** implementiran u IBM SPSS Modeler kako bismo dobili bolji uvid u to koje reči često idu zajedno.

Prvo je neophodno učitati podatke kao **flag**, zatim postaviti njihovu ulogu na **Both**. Konačno, primenimo algoritam na učitane podatke.

Imajući u vidu da se radi o visokodimenzionim podacima, očekujemo da će nam prag za podršku biti nizak.



The screenshot shows the '10001 fields' dialog box in IBM SPSS Modeler, with the 'Model' tab selected. The 'Fields' tab is also visible. The 'Model name' field is empty, with 'Auto' selected. The 'Use partitioned data' checkbox is checked. The 'Minimum antecedent support (%)' is set to 3,0. The 'Minimum rule confidence (%)' is set to 80,0. The 'Maximum number of antecedents' is set to 5. The 'Only true values for flags' checkbox is checked. The 'Optimize' option is set to 'Speed'. The 'Run' button is highlighted.

Figure 24: df.head()

Pronađeno je čak **483943 pravila**, što i ima smisla s obzirom da je i poenta oglasa za posao da prenese konačan, unapred poznat, skup informacija neophodan kandidatu.

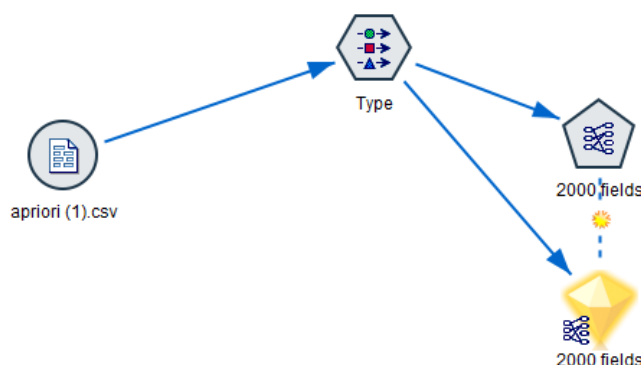


Figure 25: SPSS dijagram

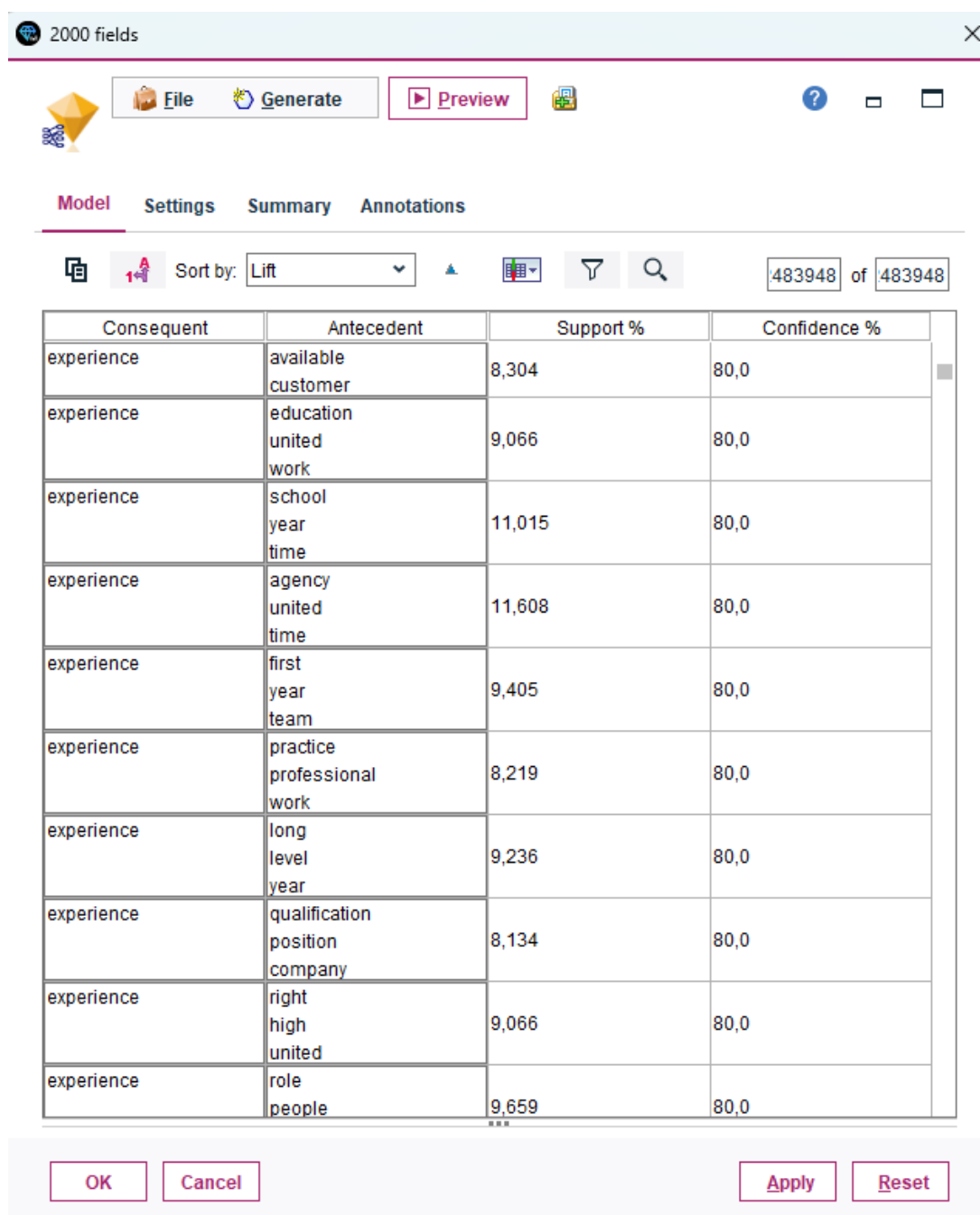


Figure 26: Apriori pravila

Na Figure 26 prikazano je prvih nekoliko najsnažnijih pravila po lift meri.

Zaključak

Ovaj skup podataka se dobro pokazao za ilustriranje klasifikacije i pravila uparivanja, dok zbog svoje visoke dimenzionalnosti nije bio previše pogodan za ilustriranje klasterovanja. Ono što je ovaj ceo proces učinilo interesantnim je praktično neograničen broj tehnika i mogućih pristupa rešavanja problema.