“Modern and renaissance poetry”

Projektni zadatak iz predmata Istražiavnje podataka 1

Milica Nikolić 89/2018

Januar 2024. godine, Matematički fakultet Univerziteta u Beogradu

Sadržaj

Uvod

Projekat je baziran na podacima iz baze "Poems from poetryfoundation.org" sa sajta "kagle" i sadrži podatke vezane za književna dela. Napravljena je u cilju vežbanja klasifikacije tekstualni podataka. U nastavku projekta biće prikazani Multinomijali Naivni Bajesovski, KNN i SVM algoritam u cilju klasifikacije, KMEANS i hijerarhijsko klasterovanje, kao i pronalaženje pravila pridruživanja Apriori algoritmom.

Analiza podataka

Skup podataka iz baze “Poems from poetryfoundation.org" (u nstavku – skup podataka, ili – podaci) sastoji se od ***573 instance*** i **5 atributa.** Atributi su:

- **author** – autor književnog dela
- **content** – sadržaj ili deo sadržaja književnpg dela
- **poem name** – naziv književnog dela
- **age** – doba književnog dela ('Renaissance' ili 'Modern')
- **type** – tip književnog dela ('Mythology & Folklore', 'Nature', ili 'Love')

Dodatno, kreiran je atribut **age_type** predstavlja kombinacija atributa age i atributa type. Takođe, atribut poem name i author ne smatraju se releveantim za obradu, pa ih nećemo razmatrati.

Što se tiče kategorija za klasifikaciju, fokusiraćemo se na atribut age, ali osvrtaćemo se i na atribute type i age_type. U nastavku su prikazani dijagrami koji prikazuju procentualnu zastupljenost klasa u našim podacima.

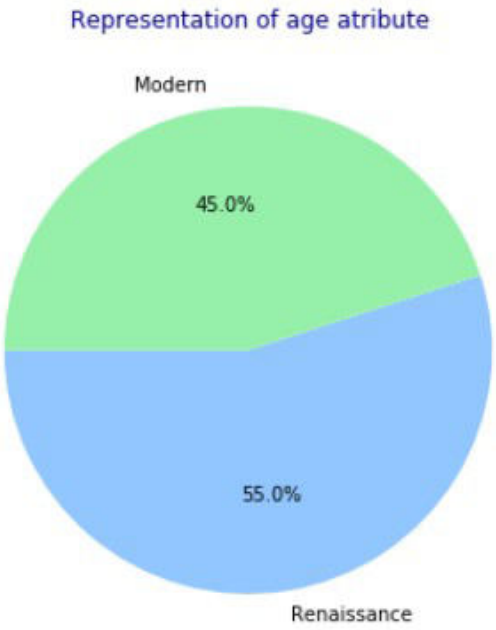


Figure 1: zastupljenost klasa u atributu age

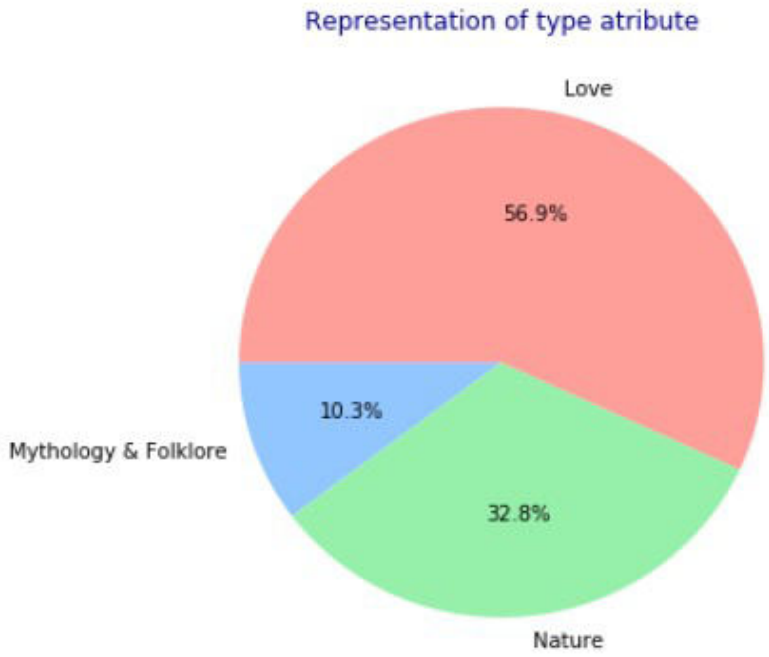


Figure 2: zastupljenost klasa u atributu type

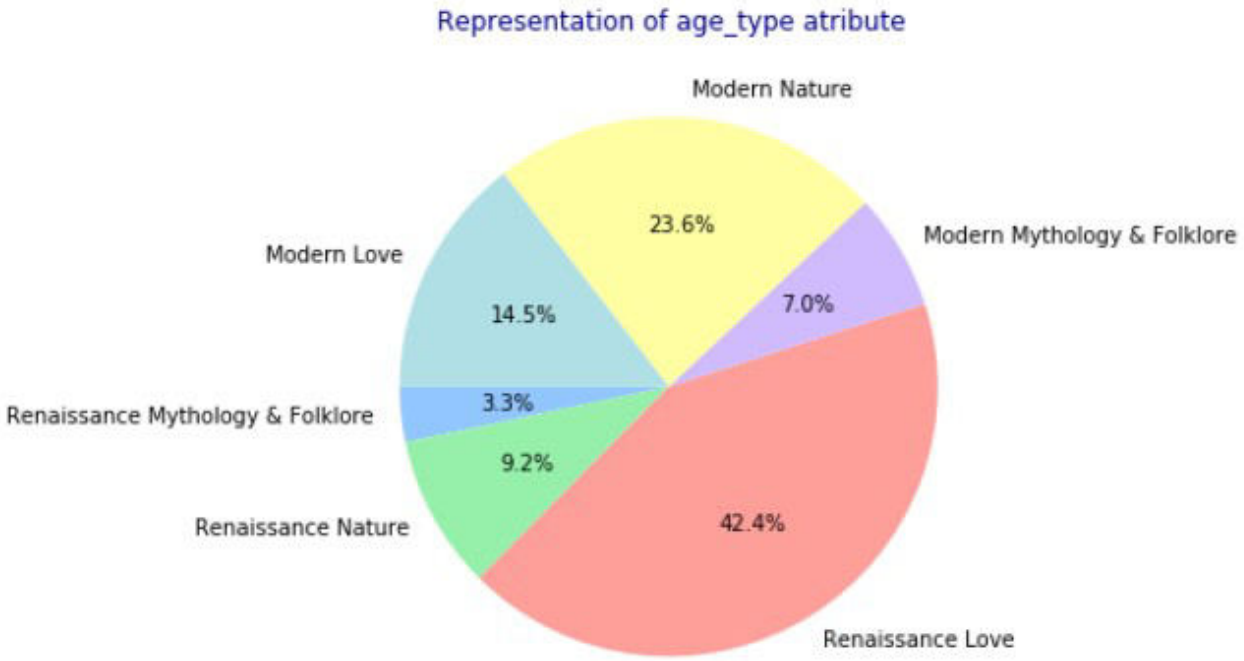


Figure 3: zastupljenost kombinacije klasa u atributima age i type

Kao što je već napomenuto, podatke ćemo prvenstveno klasifikovati u klase iz atributa age. Intuitivno možemo zaključiti, na osnovu prethodnih grafikona (Figure 1, Figure 2) i činjenice da se početni skup podataka sastoji od nešto više od 570 instanci, da je verovatno najlakse napraviti klasifikacioni model za klase atributa age. Broj od 573 instance, koliko ih ima u početnom skupu će se usled analize i pretprocesiranja pottencijalno i smanjiti. Podelu na trening i test podatke izvršićemo u odnosu 70% i 30%. Možemo zaključiti da su klase iz atributa type ***nebalansirane***, te

ih je neophodno posebno razmatrati. Za klase generisane atributom age bismo mogli da smatramo da su ***balansirane***, međutim, kako radimo na relativno malom skupu podataka balansiranost klasa bi mogla drastičnije da utiče na kvalitet modela koje ćemo praviti, pa ćemo iz tog razloga i ove klase smatrati ***blago nebalansiranim*** i po potrebi ih posebno obrađivati. Klase generisane kombinacijom atributa age i type prikazane su znatno nebalansirane, pa s obzirom da je skup podatak mali, njih nećemo dalje razmatrati.

Nedostajuće vrednosti

Skup podataka koji obrađujemo sadrži dve instance kojima nedostaje atribut poem name, ali kao što je ranije naglašeno, taj atribut ne smatramo relevantim, pa je u redu da ove dve instance ostavimo kakve jesu.

```
df.isnull().sum()
```

```
author       0
content      0
poem name    2
age          0
type         0
age_type     0
```

Table 1: broj nedostajućih vrednosti po atributima

Duplirani podaci

Jednu instancu datog skupa podataka smatraćemo duplikatom ako u skupu podataka već postoji instanca čiji je sadržaj atributa content identičan. Duplikate ćemo jednostavno izbaciti iz skupa podataka. Brisanjem duplikata iz početnog skupa dobijamo 506 instanci.

Pretprocesiranje podataka

Najpre pogledajmo kako naš skup podataka izgleda:

```
df.head()
```

	author	content	poem name	age	type	age_type
0	WILLIAM SHAKESPEARE	Let the bird of loudest lay\nOn the sole Arabi...	The Phoenix and the Turtle	Renaissance	Mythology & Folklore	Renaissance Mythology & Folklore
1	DUCHESS OF NEWCASTLE MARGARET CAVENDISH	Sir Charles into my chamber coming in,\nWhen I...	An Epilogue to the Above	Renaissance	Mythology & Folklore	Renaissance Mythology & Folklore
2	THOMAS BASTARD	Our vice runs beyond all that old men saw,\nAn...	Book 7, Epigram 42	Renaissance	Mythology & Folklore	Renaissance Mythology & Folklore
3	EDMUND SPENSER	Lo I the man, whose Muse whilome did maske,\nA...	from The Faerie Queene: Book I, Canto I	Renaissance	Mythology & Folklore	Renaissance Mythology & Folklore
4	RICHARD BARNFIELD	Long have I longd to see my love againe,\nStil...	Sonnet 16	Renaissance	Mythology & Folklore	Renaissance Mythology & Folklore

Table 2: Prvih 5 instanci skupa podataka

Lako zaključujemo da je u pitanju ***obrada teksta***. Naime, na osnovu sadžaja nekog književnog dela,

želimo nešto da zaključujemo o tom delu. Dati tekst je neophodno pretprocesirati, u sledećem smislu:

Primer 1. Razmotrimo narednih nekoliko izjava:

- "Milica voli da igra odbojku, uvek je bila pravi sportista."
- "MILICA je uvek bila pravi sportista! Odbojku voli da igra!"
- "Milica Odbojku Voli. Pravi Sportista!"

Mi intuitivno lako zaključujemo da su sve tri izjave veoma slične. Međutim, ako bismo izjave posmatrali kao string-ove (nizove karaktera), one nisu toliko slične. Zbog toga transformišimo prethodne izjave u niz ***tokena*** (reči) koje smatramo "bitnim":

- [milica voli igra odbojka pravi sportista]
- [milica pravi sportista odbojka voli igra]
- [milica odbojka voli pravi sportista]

Sada bismo i iz "tehničkog" ugla mogli zaključiti da su ove izjave slične. U nastavku ćemo objasniti, šta smo sve to uradili da bismo došli do ovakovg zapisa.

U prethodnom primeru videli smo da dati tekst možemo nekako transformisati i na taj način dobiti "kvalitetniji" skup podataka. Da bismo to uradili, primenili smo neke transformacije i pravila. Istu stvar uradićemo i sa našim skupom podataka.

Transformacije su sledeće:

- sva slova su pretvorena u mala
- svi znaci interpunkcije su uklonjeni
- uklonili smo tzv "stop reči" (neke stop reči za engleski jezik u python-u su: "why", "only", "so", ...)

```
'let bird loudest layon sole arabian treeherald sad trumpet whose sound chaste wings obey thou shrieking harbinger f
oul precurrer fiend augur fevers end troop come thou near session interdictevery fowl tyrant wing save eagle feather
d king keep obsequy strict let priest surplice white defunctive music death divining swan lest requiem lack right th
ou treble dated crow thy sable gender makstwith breath thou givst takst mongst mourners shalt thou go anthem doth co
mmence love constancy dead phoenix turtle fledin mutual flame hence lovd love twainhad essence one two distincts div
ision none number love slain hearts remote yet asunder distance space seentwixt turtle queen wonder love shinethat t
urtle saw rightflaming phoenix sight either others mine property thus appalledthat self single natures double namene
ither two one called reason confounded saw division grow together yet either neither simple well compounded cried tr
ue twainseemeth concordant one love reason reason none parts remain whereupon made threneto phoenix dove co supremes
stars love chorus tragic scene threnosbeauty truth rarity grace simplicity enclosd cinders lie death phoenix nest tu
rtles loyal breastto eternity doth rest leaving posterity twas infirmity married chastity truth may seem cannot beau
ty brag tis truth beauty buried urn let repairthat either true fair dead birds sigh prayer'
```

Figure 4: Sadržaj atributa content jedne od instanci nakon primene transformacija

Kako algoritmi većinom rade sa brojevima, sada je neophodno da u skupu podataka koji imamo, atribute transofrimešmo u numeričke. Najbolji način da to uradimo je kreiranje ***TF-IDF*** matrice.

TF-IDF (*Term Frequencz Inverse Document Frequency*) matrica je tabela čiji su atributi termovi (reči), a vrednosti svakog od atributa t u redu d jeste broj TF-IDF(t,d) koji se odnose na d-ti tekst.

TF-IDF(t) = TF(t) x IDF(t), pri čemu je

TF(t) – broj pojavljivanje terma,

IDF(t) = 1 + log(N/df(t)),

N – broj dokumenata,

df(t) + broj dokumenata koji sadže t.

Bitno je naglasiti da ovo jedan od najjendostavnijih načina kombinovanja metrika TF i IDF. Dakle, ima ih više. Ideja je vrednovati one termove koji nisu uobičajeni globalno (relativno visok IDF), ali ih ima nezanemarljivo mnogo u razmatranom dokumentu (relativno visok TF).

U toku rada, svim modelima će biti prosleđivana TF-IDF matrica, kako za trening tako i za testiranje rada modela. Međutim, bitno je naglasiti da se TF-IDF martica ***kreira na osnovu podataka za testiranje***, jer nikako ne želimo da test podaci imaju bilo kakvog uticaja na kreiranje ove matrice.

U koliko se u nekom od tekstova iz trening skupa pojavi term koji nije u matrici (što se, naravno, u praksi često dešava), jednostavno se ignoriše. Dakle, kao i u bilo kom drugom vidu predviđanja, u klasifikaciji teksta, ***model predviđa isključivo na osnovu podataka iz trening skupa.***

	abandonfrom	abandonment	abashed	abasht	abating	abed	abhor	abhorring	abided	abjure	...	ysicles	yt	ytorne	yvorie	yvory	ywis	zeal	zeno
0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.037868	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.037868	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	

Table 3: prvih 5 redova TF-IDF matrice trening skupa na osnovu atributa content

Neki od zaključaka koje možemo doneti o novoformiranoj matrici je da su podaci ***retki*** i da je broj atributa veliki, pa samim tim treba voditi računa o tzv. ***prokletsvu dimenzionalnosti.*** Dakle, treba imati na umu da će možda biti potrebno redukovanje dimenzija podataka.

Pored prethodno pomenutog načina za predstavljanje tekstualno sadržaja kao numeričkih podataka, gde se pojam terma u TF-IDF matrici vezivao za reči iz teksta, postoji još načina. Jedan od njih je da se tekst podeli u n-grame, gde n-gram predstavlja bilo koju podnisku susednih simbola dužine n. U ovom slučaju tekst bismo podelili u n-grame i njih tretirali kao termove, na osnovu kojih bismo pravili TF ili TF-IDF matricu. Ovakav način[1] je primenjivan u praksi u odredjivanja jezika na kom je tekst napisan.

1 Ideja iz knjige Veštačka inteligencija.

Opis modela

U nastavku su na skup podataka koji obrađujemo primenjeni razni algoritmi u cilju klasifikacije, klasterovanja i određivanja pravila pridruživanja.

Klasifikacija

U nastavku su data tri algoritma za klasifikaciju u klase koje smo smatrali balansiranim. Za svaki je naveden kratak opis i neke mere kvaliteta tog modela. Nakon toga sledi rad sa nebalansiranim klasama i prikazana je naivna strategija generisanja novih instanci slučajnim uzorkovanjem sa ponavljanjem iz postojećeg skupa podataka. Sve je implementirano u programskom jeziku Python i detalji implementacije nisu navođeni.

Algoritam K najbližih suseda (K Nearest Neighbours - KNN)

Algoritam K najbližih suseda (KNN) je algoritam za klasifikaciju podataka ili regresiju, a ujedno i jedan od najjednostavnijih algoritama nadgledanog mašinskog učenja. Osim jednostavnosti, ovaj algoritam se pokazao dobro na manjim skupovima podataka, jer spada u metode "lenjog učenja". Skup podataka na kom radimo ima 506 instanci, pa ga smatramo malim.

Na osnovu sadržaja atributa content, napravićemo model koji korisit KNN allgoritam i uči da klasifikuje podatke u jedno od klasa iz atributa age (Modern ili Renaissance).

Algoritam K najbližih suseda na neredukovanim podacima

Kao vrednost faktora K, uzećemo sqrt(N), gde je N broj instanci. U praksi se izbegavaju parne vrednosti za k. Kako je √506 = 22.49, uzimamo **k = 23**. Zbog toga što su podaci TF-IDF matrice retki, kao meru sličnosti koristimo kosinusnu.

Confusion matrix:
```
[[130  62]
 [  1 208]]
```

Classification report:

	precision	recall	f1-score	support
Modern	0.99	0.68	0.80	192
Renaissance	0.77	1.00	0.87	209
accuracy			0.84	401
macro avg	0.88	0.84	0.84	401
weighted avg	0.88	0.84	0.84	401

Figure 6: Matrica kofuzije i izveštaj klasifikacije na trening skupu

Confusion matrix:
```
[[ 41  25]
 [  0 106]]
```

Classification report:

	precision	recall	f1-score	support
Modern	1.00	0.62	0.77	66
Renaissance	0.81	1.00	0.89	106
accuracy			0.85	172
macro avg	0.90	0.81	0.83	172
weighted avg	0.88	0.85	0.85	172

Figure 5: Matrica kofuzije i izveštaj klasifikacije na test skupu

Vidimo da je accuracy na trening skupu 0.84, a na test skupu 0.85. Kako su ove dve vrednosti blizu jedna druge, možemo zaključiti da nije došlo do preprilagođavanja trening skupu. Takođe, vrednosti AUC skora koje možemo videti na slici (*Figure 7*) su blizu jedinice, pa model smatramo zadovoljavajućim s obzirom da je u pitanju klasifikacija teksta na neveliko skupu podataka.

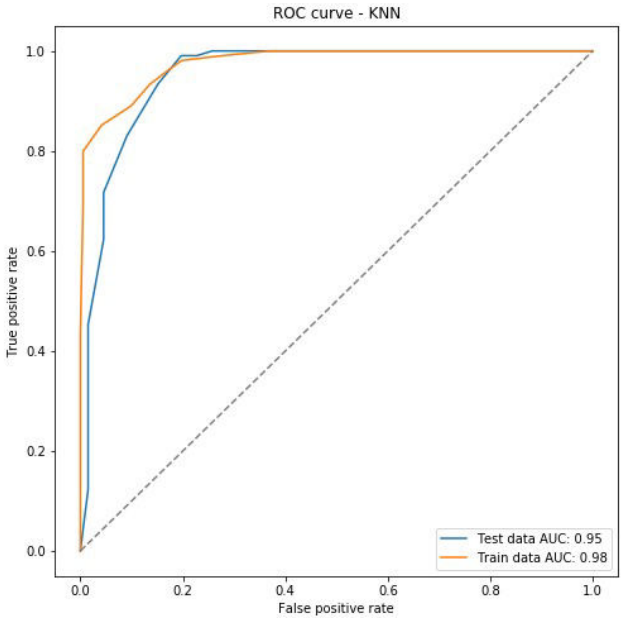


Figure 7: ROC kriva

Algoritam K najbližih suseda na redukovanim podacima

Algoritam KNN zasnovan je na računanju udaljenosti izmedju tačaka. Kako povećanjem dimenzionalnosti podataka raste i raspršenost, odnosno varijansa podataka (tačaka), ovaj algoritam smatramo osetljivim na prokletstvo dimenzionalanosti podataka. Imajući to u vidu, primenićemo algoritam i na redukovane podatke. Kako radimo na redukovanim podacima, kao metriku vise ne koristimo kosinusnu sličnost, vec rastojanje Minkovskog.

Kada podatke redukujemo na 10 komponenti koritsteci dekompoziciju singularnih vrednosti (SVD) i broj suseda postavimo na K = 23, postižemo približno istu tačnost kao u prethodnom slučaju kada smo algoritam primenili na neredukovane podatke.

Multinomijalni Naivni Bajes

Naivni Bajesov algoritam je algoritam za klasifikaciju podataka koji podatak svrastava u onu klasu, koja je za njega najverovatnija, pri pretpostavci da odlike koje se koriste pri odlučivanju nisu zavisne jedna od druge, pa time uslovna verovatnoća svake odlike zavisi samo od klase podatka. Multinomijalni Naivni Bajesov klasifikator se oslanja na multinomijalnu raspodelu te verovatnoće. Pogodan je kada vrednosti odlika predstavljaju broj javljanja nečega, što se uklapa u temu i podatke koje obrađujemo.

Na osnovu sadržaja atributa content, napravićemo model koji korisit Multinomijali Bajesov algoritam i uči da klasifikuje podatke u jedno od klasa iz atributa age (Modern ili Renaissance).

Kako smo i ovaj algoritam primenili u programskom jeziku Python, za određivanje hiperparametara alpha (kontroliše *smoothing* primenjen na procene verovatnoće), class_prior, fit_prior koristili smo ugrađenu funkciju koja pohlepnim algoritmom i unakrsnom validacijom određuje najbolje vrednosti ovih hiperparametara. Na validacionom skupu je najbolja postignuta tačnost bila 0.92. Odatle smo dobili sledeće vrednosti hiperparametara:

- alpha = 0.1
- class_prior = None
- fit_prior = False

Sa slika koje slede u nastavku vidimo vidimo visoke vrednosti f1 skora i accuracy-ja. Površina ispod ROC krive (AUC skor) je blizu jedinice. Ono što takođe možemo primetiti je da se model blago preprilagođava. Ovo možemo pripisati veličini skupa podataka. Ipak, na osnovu svega priloženog, zaključujemo da smo napravili kvalitetan model uzimajući u obzir težinu problema i činjenicu da je skup podataka na kom radimo mali.

```
Confusion matrix:
 [[162   0]
 [  0 192]]

Classification report:
              precision    recall  f1-score   support

      Modern       1.00      1.00      1.00       162
 Renaissance       1.00      1.00      1.00       192

    accuracy                           1.00       354
   macro avg       1.00      1.00      1.00       354
weighted avg       1.00      1.00      1.00       354
```

Figure 9: Matrica kofuzije i izveštaj klasifikacije na trening skupu

```
Confusion matrix:
 [[49 11]
 [ 0 92]]

Classification report:
              precision    recall  f1-score   support

      Modern       1.00      0.82      0.90        60
 Renaissance       0.89      1.00      0.94        92

    accuracy                           0.93       152
   macro avg       0.95      0.91      0.92       152
weighted avg       0.94      0.93      0.93       152
```

Figure 8: Matrica kofuzije i izveštaj klasifikacije na test skupu

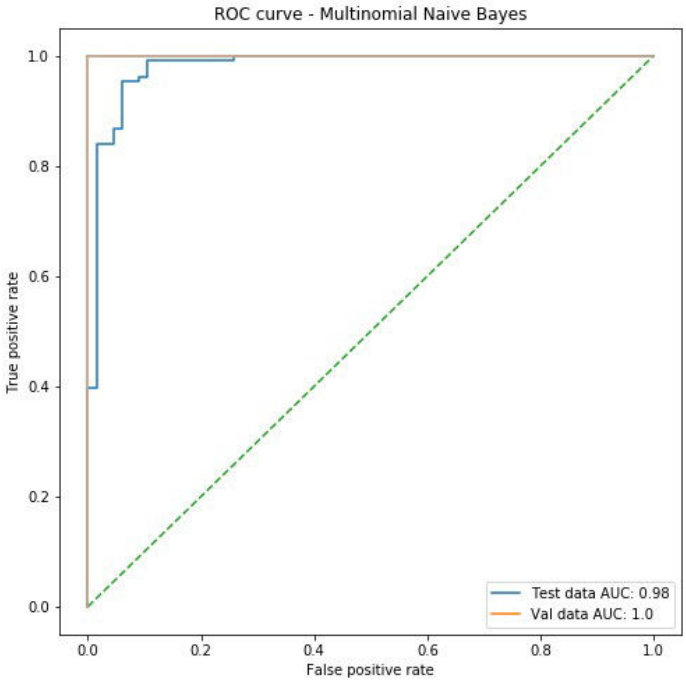


Figure 10: ROC kriva

Pomenuto blago preprilagođavanje, može biti motiv za kreiranje modela koji sadrži hiperparametre za regulariuzaciju i time sprečiti preprilagođavanje.

SVM (Support Vector Machines) metoda

SVM - metoda potpornih vektora je metoda klasifikacije koja generiše model klasifikatora koji predstavlja ***formulu***. U vektorskom prostoru u kom su predstavljeni podaci, pronalazi se ***hiperravan*** koja razdvaja podatke tako da su podaci koji pripadaju istoj klasi, sa iste strane te hiperrravni. Kako ovakvih ravni ima više, bira se ona s najvećom marginom, tj. maksimalnim rastojanjem od trenirajućih podataka. U fazi testiranja, računa se rastojanje od hiperravni koje određuje klasu.

Hiperravan u SVM metodi je potpuno određena trening podacima koje nazivamo ***podržavajući-potporni vektori***, odakle dolazi i ime metode.

Kada koristimo SVM metodu, razlikujemo tri slučaja:

- SVM kada su podaci linearno razvojivi
- SVM sa "mekom" marginom
- SVM sa funkcijama jezgra

Kao i u prethodna dva modela, na osnovu sadržaja atributa content, napravićemo model koji korisiti metodu potpornih vektora i uči da klasifikuje podatke u jedno od klasa iz atributa age (Modern ili Renaissance).

Hiperparamametre ponovo možemo izabrati pohlepnim algoritmom i uz pomoć unakrsne validacije i dobiti optimalni parametar za regularizaciju C = 1 i da možemo koristiti linear za jezgro (tačnost 0.92 na validacionom skupu). Ono što primećujemo primenom ovih optimalnih parametara je da ponovo dolazi do preprilagođavanja. Naime, trening skup nam je tačnosti 1.0, a validacioni 0.92. Ovo rešavamo promenom regularizacionog parametra C. Dakle, u slučaju da pohlepnih algoritmom dobijemo optimalne parametre, ali primetimo pojavu potencijalnog preprilagođavanja, treba testirati i nešto lošije kombinacije hiperparametara u cilju smanjenja potencijalne razlike u kvalitetu klasifikacije na trening i test skupu, odnosno preprilagođavanja. Za C = 0.5 dobijamo rezultate prikazane u nastavku (Figure 12, 13, 14).

```
Confusion matrix:
 [[160   2]
 [  0 192]]

Classification report:
              precision    recall  f1-score   support

      Modern       1.00      0.99      0.99       162
 Renaissance       0.99      1.00      0.99       192

    accuracy                           0.99       354
   macro avg       0.99      0.99      0.99       354
weighted avg       0.99      0.99      0.99       354
```

Figure 12: Matrica kofuzije i izveštaj klasifikacije na trening skupu

```
Confusion matrix:
 [[52  8]
 [ 0 92]]

Classification report:
              precision    recall  f1-score   support

      Modern       1.00      0.87      0.93        60
 Renaissance       0.92      1.00      0.96        92

    accuracy                           0.95       152
   macro avg       0.96      0.93      0.94       152
weighted avg       0.95      0.95      0.95       152
```

Figure 11: Matrica kofuzije i izveštaj klasifikacije na test skupu

Na osnovu prikazanih mera kvaliteta klasifikacije, zaključujemo da smo napravili kvalitetan model.

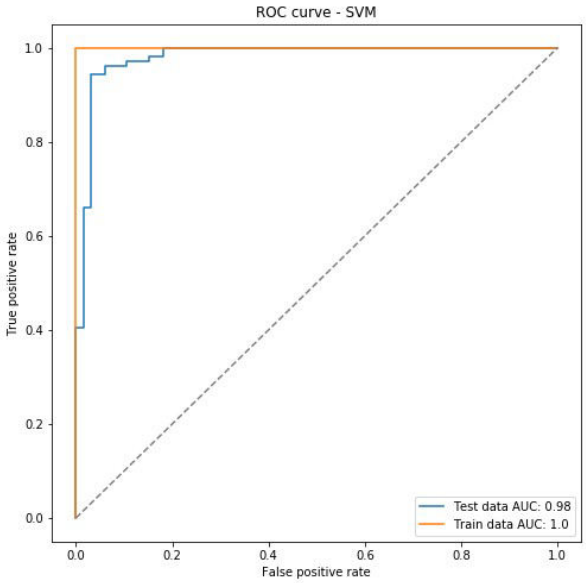


Figure 13: ROC kriva

Ono što želimo sledeće da uradimo je da vizuelizujemo podatke i hiperravan koja ih razdvaja. Kako su nam podaci velikih dimenzija, naknadno redukujemo test skup i prikazujemo hiperravan koju je SVC naučio (Figure 14). Grafikon koji vidimo ne uklapa se u sa kvalitetom modela za koji smo malopre zaključili da je visok. To je posledica toga što model nije treniran na redukovanim, već na podacima velikih dimenzija, koji su na grafikonu prikazani u samo tri dimenzije.

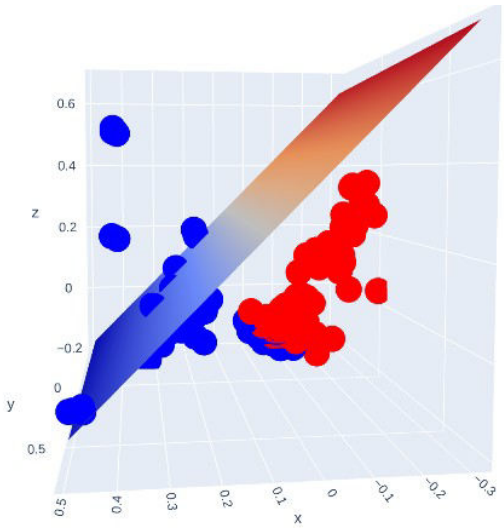


Figure 14: hiperravan SVC-a i naknadno redukovani podaci

SVM nad redukovanim podacima

U nastavku želimo da napravimo SVM model sa linearnim jezgrom i treniramo ga na redukovanim podacima u tri dimenzije. Ovakav pristup nam odgovara jer možemo verodostojno vizualizovati podatke i hiperravan koja ih razdvaja. Pohlepnim algoritmom utvrđeno da bi model bio kvalitetniji korišćenjem RBC jezgra, što je i očekivano, jer promenom dimenzionalnosti podatka menjaju se i rastojanja i gustina podataka. Mi ćemo zadržati linearno jezgro zbog jednostavnije vizuelizacije, a na štetu kvaliteta modela.

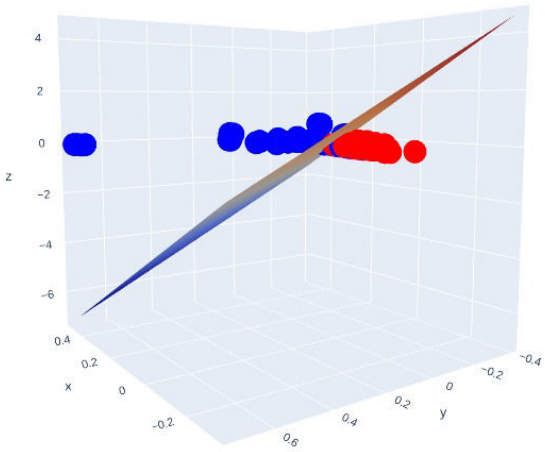


Figure 15: hiperravan SVC-a sa linearnim jezgrom nad redukovanim podacima

U nastavku sledi vizuelni prikaz (Figure 15) hiperravni SVC-a sa linearnim jezgrom i redukovanih podataka iz test skupa. Dostignut accuracy na test skupu je 0.77. Kako smo "nametnuli" linearno jezgro i model koristili u svrhu vizuelizacije, nećemo se dalje baviti unapređivanjem modela. Zaključak je da, kada korigujemo dimenzionalnost podataka, uvek je neophodno da za podatke redukovane dimenzionalnosti ispitamo kako bi model trebalo prilagoditi novodobijenim podacima, jer se te karakteristike često neće poklapati sa onim koje smo primenili nad neredukovanim skupom podataka.

Poređenje modela – SVM, Naivni Bajes i KNN

U nastavku ćemo uprediti kvalitete modela za klasifikaciju koje smo prethodno napravili. Kako smo naš skup podataka odmah u fazi pretprocesiranja podelili na podatke za trening i test, svi modeli su trenirani i testirani nad istim podacima, pa ih ima smisla i uporediti. Poređenje koje sledi u nastavku je nad test skupom podataka. Da nam je cilj bio da pronađemo jedan, najkvalitetniji, model za naš skup podataka i njega koristimo nad test podacima, kvalitet modela bismo poredili na validacionom skupu.

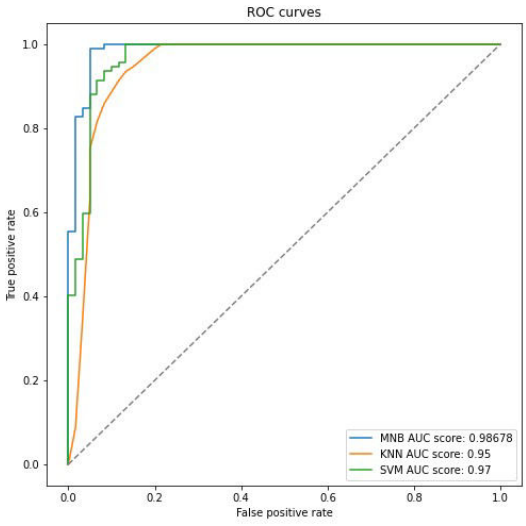


Figure 17: ROC krive

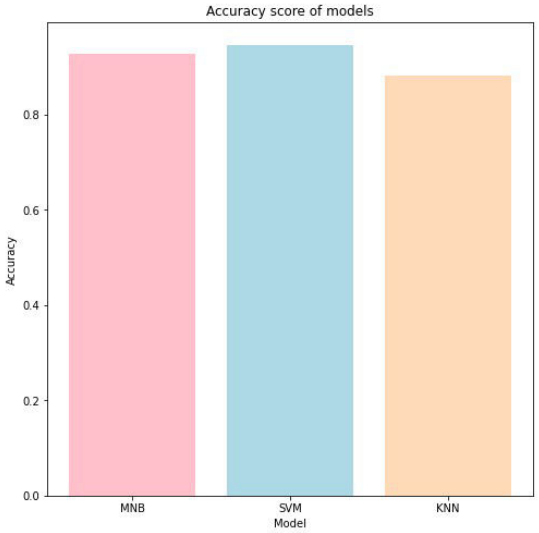
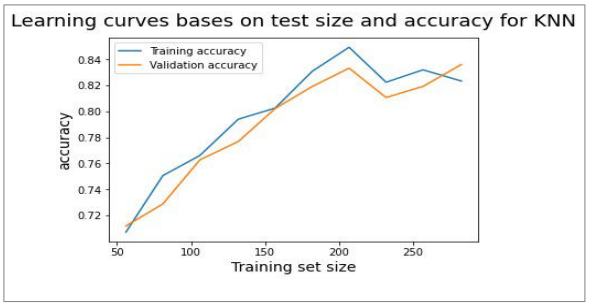
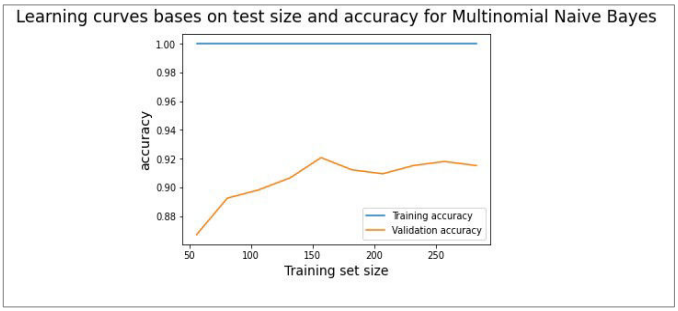


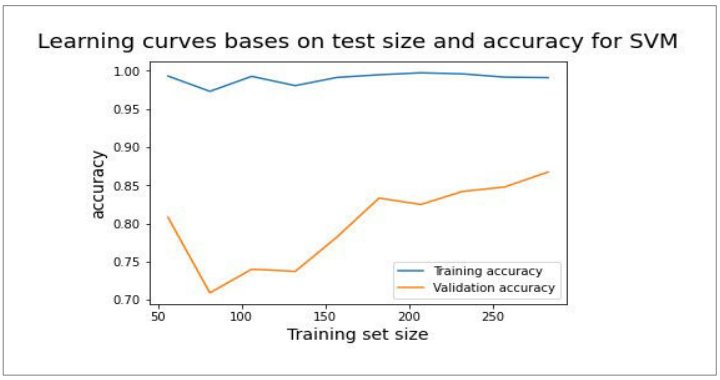
Figure 16: Accuracy modela

Maksimalne dostignute tačnosti modela na test skupu su 0.93, 0.95, 0.84 (Figure 17) redom za Multnomijalni Naivni Bajes, SVM i KNN. Površina ispod ROC krive (AUC skor) je za sva tri modela veoma blizu jedinice (Figure 16).

U nastavku prikazujemo krive učenja za sva tri modela, a kao meru koristimo preciznost. Ono što prvo primećujemo je da krive nisu "glatke". Ovo je posledica toga što radimo na malom skupu podataka, jer što je skup podataka veći, kriva je, naravno, sve više glatka. Na graficima vidimo da je za svaki model preciznost za najmanju veličinu trening skupa manji u odnosu na preciznost za maksimalnu velicinu. To nam govori da sva tri modela, povećanjem trening skupa, bolje uče. To je svakako osobina koju želimo da naši modeli imaju.







Ono što je takođe poželjna osobina je da se krive slično ponašaju na trening i validacionom skupu. To nam sugreiše da je manja šansa da dođe do preprilagođavanja modela. To je veoma dobro postignuto u modelu K najbližih suseda. Razlike koje vidimo na druga dva modela su posledica blagog preprilagođavanja i toga što je trening skup mali, pa je posledično i validacioni skup mali.

Iz svega prethodnog možemo zaključiti da su se SVM i Multinomijalni Naivni Bajes, u našem slučaju, pokazali mnogo bolje u rešavanju problema klasifikacije od algoritma K najbližih suseda (KNN). Kako smo potencijalno preprilagođavanje uspeli da rešimo u SVM modelu, njega smatramo najboljim rešenjem.

Rad sa nebalansiranim klasama

Zaključili smo da u skupu podataka koji obrađujemo, neke klase smatramo ***nebalansiranim***. To su atributi type i age, s tim što je age blago nebalanasirm, pa u tom slučaju ne očekujemo velike promene nakon primena tehnika koje su pomenute u nastaku.

Kada želimo da uradimo klasifikaciju podataka u nebalansirane klase, javljaju nam se brojni problemi i svi proističu iz toga što nemamo dovoljno predstavnika manjinske klase. Ovaj problem se prevazilazi raznim tehnikama koje možemo podeliti u četiri grupe:

- Oversampling - generisanje novih instanci uzorkovanjem manjinske klase
- UnderSampling - smanjivanje broja instanci većinske klase uzorkovanjem.
- Kombinovanje Oversampling-a i UnderSampling-a
- Imbalanced Ensamble – primena ansambla

Bitno je napomenuti da je, pre primene bilo kakvih tehnika, najbolje da pokušamo da pronađemo i dodamo u naš skup podataka više instanci manjinske klase.

Kako nismo u mogućnosti da to uradimo, u nastavku ćemo demonstrirati kako smote over sampling tehnika može uticati na kvalitet klasifikacije. Ovo je tehnika oversampling-a i za tehniku iz ove grupe odlučili smo se jer koristimo relativno mali skup podataka, pa je i intuitivno jasno da undersampling-om ne bismo postigli željene rezultate.

Primenićemo metodu potpornih vektora (SVM) u cilju klasifikacije podataka, na osnovu atributa content, u nebalansirane klase iz atributa type i blago nebalansirane klase atributa age. U prvom slučaju dobijamo veoma loše rezultate na validacionom skupu, a u drugom možemo primetiti da model, iako postiže dobre rezultate, blago favorizuje većinsku klasu. To ćemo u nastavku pokušati da popravimo primenom pomenute tehnike za rad sa nebalansiranim klasama.

Razmotrimo prvo klase iz atributa type. Naime, treniranjem SVM modela na trening skupu dobijamo tačnost od 0.57, što model praktično čini neupotrebljivim. Nakom primene oversampling smote metode, dobijamo skup podataka sa balansiranim klasama (Figure 18) i istim algoritmom postižemo nešto bolju tačnost od 0.75 na trening skupu, odnosno 0.74 na test skupu.

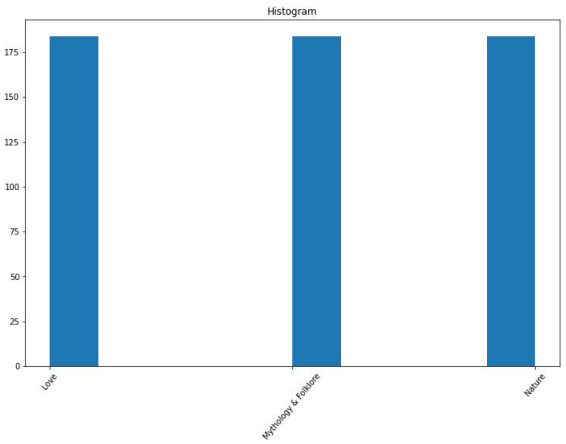


Figure 18

Razmotrimo dalje klase iz atributa age, za koje smo zaključili da su blago nebalansirane. Primenom metode potpornih vektora u cilju klasifikacije u ove klase, dobili smo dobre rezultate. Međutim, ono što smo mogli da primetimo da je model favorizuje većinsku klasu, pa na taj način neke instance lažno klasifikuje u većinsku klasu, ali suprotno ne radi.

Primenom smote metode dobijamo balansirane klase (Figure 19) i primenom modela dobijamo tačnost 0.93 na test skupu. Iako lošije tačnosti, primetimo da ovaj model približno jendako greši u obe klase (Figure 20), pa možemo zaključiti da smo rešili problem favorizovanja većinske klase.

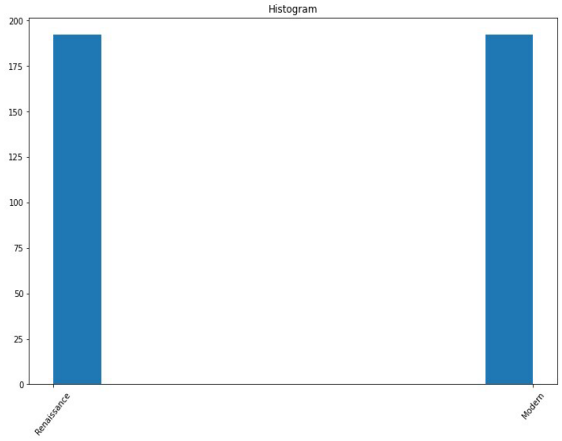


Figure 19

```
Confusion matrix:
 [[54  6]
 [ 5 87]]

Classification report:
              precision    recall  f1-score   support

      Modern       0.92      0.90      0.91        60
 Renaissance       0.94      0.95      0.94        92

    accuracy                           0.93       152
   macro avg       0.93      0.92      0.92       152
weighted avg       0.93      0.93      0.93       152
```

Figure 20

Klasterovanje

U nastavku su opisana dva algoritma korišćena za klasterovanje našeg skupa podataka na osnovu atributa content. Nakon primene svakog od algoritama, procenjena je korektnost klasterovanja unutra[njim i spolja[njim kriterijumima. Kao i u prethodnom slučaju, sve je implementirano u programskom jeziku Python i detalji same implementacije nisu navođeni.

Hopkinsova statistika

Jedna od statističkih metoda čijom se primenom proverava da li postoje klasteri u datim podacima je Hopkinsova statistika.

Ako je dat skup od n tačaka dimenzije d koje klasifikujemo, potrebno je uniformno generisati m<n tačaka na prosotoru koji pokriva početni skup. Nakon toga biramo uzorak od m tačaka iz početnog skupa. Sada za svaku tačku oba ova skupa računamo rastojanje do najbližeg suseda u originalnom skupu. Tada se ***Hopkinsova statistika*** računa po formuli:

$$H = \frac{\sum_{i=1}^{m} u_i^d}{\sum_{i=1}^{m} u_i^d + \sum_{i=1}^{m} w_i^d}$$

gde je u_i predstavlja tačke uzorkovane iz originalnog skupa, a w_i tačke veštački generisanog skupa.

Problem sa ovom statistikom je što će imati veliku vrednost i kada su podaci uniformno raspoređeni, a postoji samo jedan klaster.

Hopkinsova statistika naših podataka približno je 0.1, što nam govori da podaci nemaju tendenciju grupisanja u klastere. Međutim, kako mi imamo i neke stvarne klastere, što bi u našem slučaju bile klase iz atributa age i type, klasterovanje vredi ispitai. Ovako mala vrednost Hopkinsove statistike može značiti i da podaci nisu uniformno raspoređeni.

Algoritam K-sredina

K-sredina (*KMEANS*) je algoritam za klasterovanje podataka koji svakoj od instanci dodeljuje jedan od K klastera. Ovo je algoritam zasnovan na reprezentativnim predstavnicima (centroidima). Podaci se grupišu u klastere po sličnosti. Dakle, svakoj instanci ce biti dodeljen klaster u kom se nalazi njen najbliži centroid. Controidi se menjaju u svakoj iteraciji tako što se uprosečavaju vrenosti elemenata svakog od klastera. Algoritam se zaustavlja kada centroidi prestanu da se manjaju. Kao mera bliskosti instanci korišćeno je Euklidsko rastojanje, iako u obzir dolaze i druge mere, kao što je kosinusna.
Kako je ovaj algoritam veoma osetljiv na izbor početnih centroida, koristićemo napredniji način za njihov odabir. Naime, početni centroidi se biraju u više iteracija i zadržavaju se oni za koje je inercija bila najveća.
Vrednost K se unapred određuje. Postoji više pristupa određivanju broja klastera. Prvo ćemo postaviti da je K = 2 i K = 3 u cilju poređenja klastera sa dve klase iz atributa age i tri klase iz atributa type našeg skupa podataka. K je moguće odrediti i "metodom lakta" (*Elbow method*), što je takođe prikazano u nastavku.
Kao i svaki model koji je zasnovan na udaljenostima ili gustini, algoritam K-sredina smatramo veoma osetljivim na prokletstvo dimenzionalnosti, pa ćemo prikazati kako se on ponaša nad podacima redukovane i neredukovane dimenzionalnosti.

Algoritam K-sredina na neredukovanom skupu podataka

Za početak primenjujemo algoritma K-sredina na neredukovanom skupu podataka, koristeći napredno biranje početnih centroida tehnikom k-means++.

Broja klasera postavljamo na ***K = 2***. Za potrebe crtanja grafikona naknadno smo redukovali skup podataka na dve dimenzije primenom algoritma analize glavnih komponenti (PCA).
Predstavljanjem redukovanih podataka grafikonom (Figure 21) primećujemo da je algoritam K-sredina uspeo da veoma grubo prepozna razliku između klasa iz atributa age.

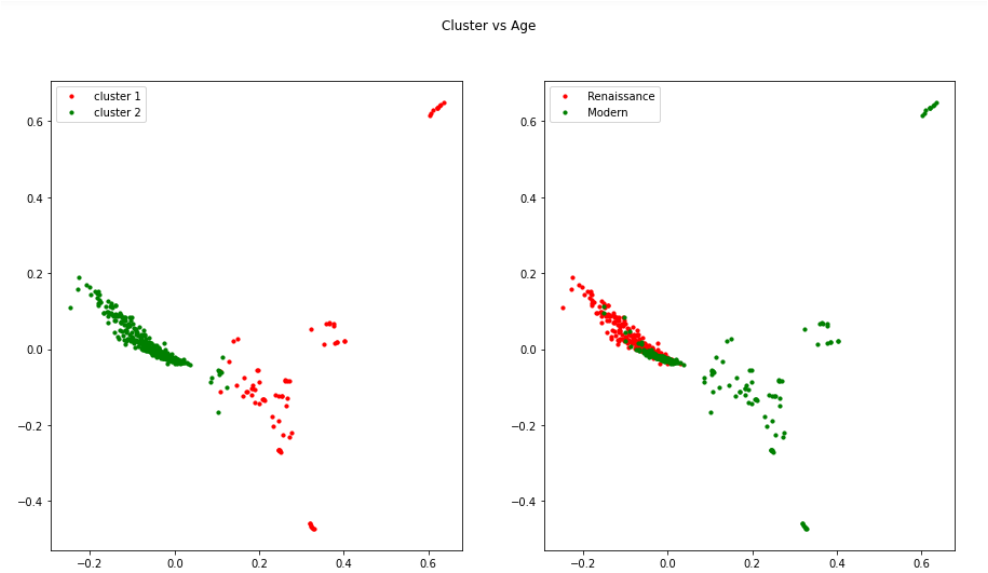


Figure 21: K-sredina za K = 2 i klase atributa age

Dalje, postavljamo broj klastera na ***K = 3*** i primanjujemo isti algoritam, nakon čega ponovo, nad redukovanim podacima, skiciramo grafik i poredimo klastere sa klasama iz atributa type. Na osnovu grafikona (Figure 22) možemo zaključiti da algoritam nije uspeo da prepozna razlike među klasama atributa type.

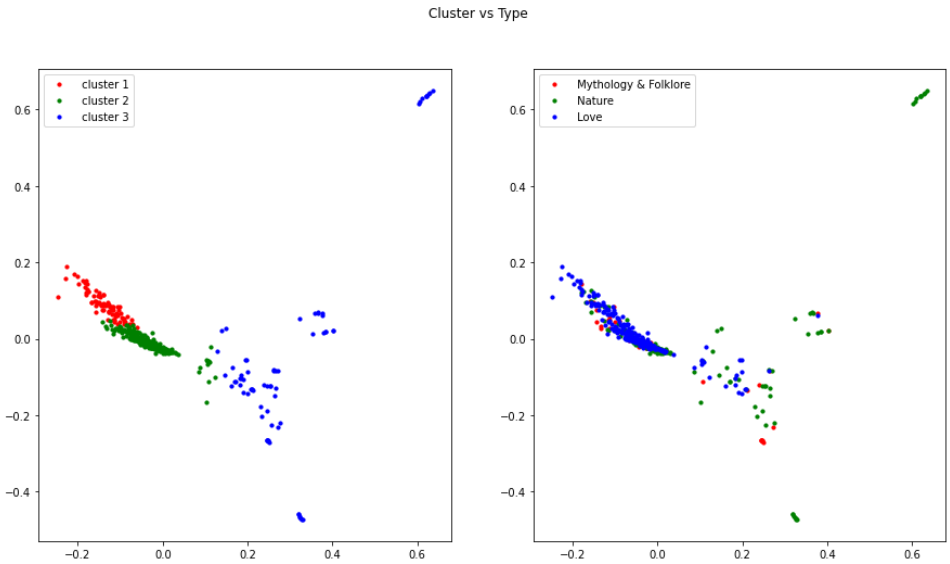


Figure 22: K-sredina za K = 3 i klase atributa type

Prethodno poređenje klastera sa stvarnim labelama/klasama koje su nam dostupne predstavlja ***spoljašnje kriterijume podele*** podataka u klastere. U tom smislu mogu se koristiti i merekao što su matrica konfuzije, preciznost, odziv i slično. Kako nije cilj, i bilo bi veoma teško, napraviti model koji će klasterovati podatke baš u klase koje su bile ciljne u procesu klasifikacije, potrebno je koristiti ***unutrašnje kriterijume*** koji će oceniti kvalitet klasterovanja. Neki od unutrašnjih kriterijuma provere kvaliteta klasteroavanja su silueta koeficijent i inercija.
U nastavku je prikazan grafik zavisnosti inercije (Figure 23 levo) i silueta koeficijenta (Figure 23 desno) u odnosu na broj klastera K. Sa grafikona možemo pročitati da je silueta koeficijent izuzetno mali (približno 0.03) i u najboljem slučaju, kada je K=6. Dakle, zaključujemo da klasterovanje neće biti kvalitetno za bilo koje od ispitanih vrednosti K.

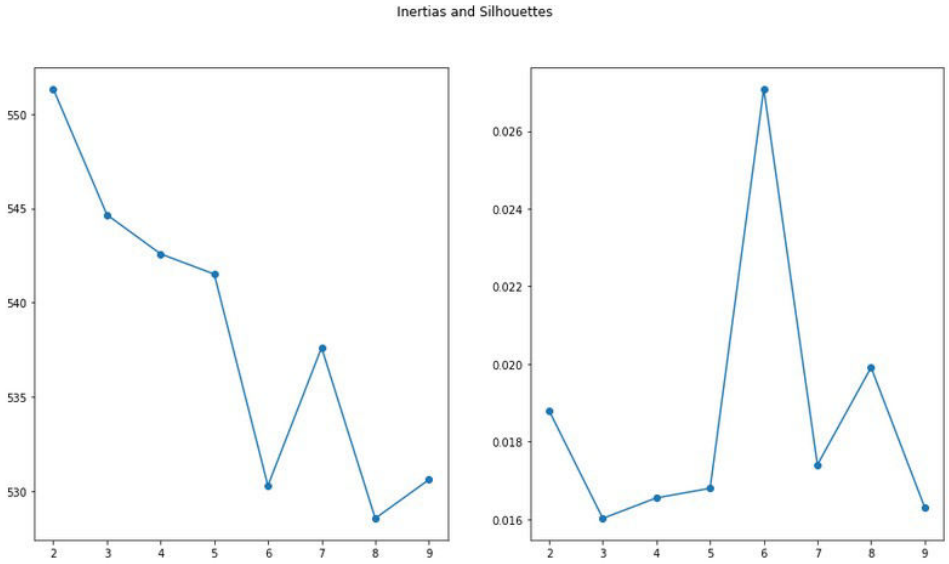


Figure 23: Inercija i Silueta u odnosu na broj klastera

Ovo može biti rezultat toga što su naši podaci izuzetno velike dimenzionalnosti, na šta je algoritam K-sredina veoma osetljiv zbog svoje zasnovanosti na udaljenostima (sličnostima) među podacima. Ovakvo ponašanje algoritma je već pomenuto, kada je primenjen KNN algoritam za klasifikaciju, i naziva se prokletstvo dimenzionalnosti.

Algoritam K-sredina na redukovanom skupu podataka

Kako bismo izbegli pojavu prokletstva dimenzionalnosti, podatke ćemo u nastavku redukovati algoritmom analize glavnih komponenti (PCA).
Najpre podatke redukujemo na dimenziju dva. Na osnovu novog grafikona zavisnosti inercije i siluete u odnosu na broj klastera K (Figure 24) primećujemo da je inercija smanjena, a da silueta koeficijent postiže prihvatljivu vrednost (približno 0.75). Metodom lakta u kombinaciji sa vrednostima silueta koeficijenta zaključujemo da je optimalni broja klastera K = 4, kada radimo sa redukovanim podacima. Ono što dalje radimo je primena algoritma K-sredina na redukovane podatke i sa parametrom K = 4 (Figure 25).

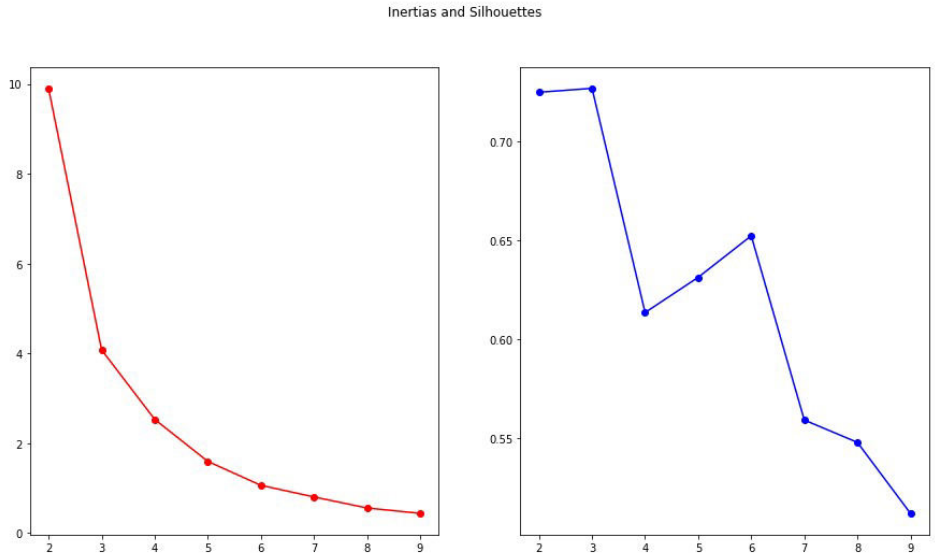


Figure 24: Inercija i Silueta u odnosu na broj klastera - redukovani podaci

Elemenata u prvom klasteru je 422, u drugom 9 i trećem 75.

Odavde možemo zaključiti da bi u ovom slučaju podatke koji se nalaze u klasteru 2 mogli da posmatramo i kao elemente van granica, koje bismo mogli da ignorišemo ili veštački priključimo trećem klasteru.

Priključivanjem drugog klastera trećem klasteru, možemo se vratiti spoljašnjoj analizi klasterovanja i uporediti dobijene klastere sa klasama iz atributa age. Rezultati ovakve spoljašnje analize prikazani su na grafikonu u nastavku (Figure 26).

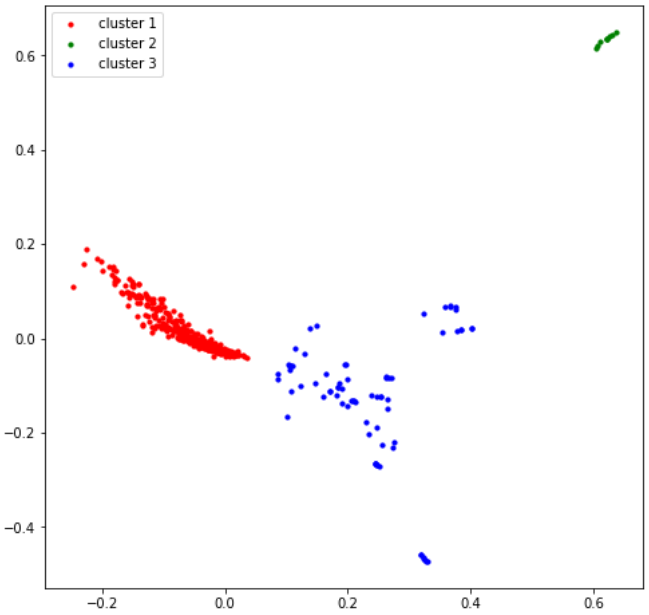


Figure 25: Dobijena 4 klastera nad redukovanim podacima

Tačnost u odnosu na klase atributa age je 0.73, pa možemo i formalno zaključiti da je algortiam donekle zaista prepoznao ove klase. Ono što je takođe zanimljivo je da je sve instance iz klase koje predstavljaju književna dela iz doba renesanse zaista prepoznao i postavio u jedan klaster, ali je takođe i mnoga dela iz modernog doba prepoznao kao renesansna. Da nam je cilj klasterovanja bio bolje upoznavanje podataka u fazi pretprocesiranja za neki vid nadgledanog učenja, ova informacija bi bila veoma korisna. Naprimer, mogli bismo da predpostavimo da će algoritmi za klasifikaciju lakše prepoznavati klasu renesansnih dela, pa u skladu sa tim i prilagoditi algoritam.

```
[[ 84 138]
 [  0 284]]
              precision    recall  f1-score   support

      Modern       1.00      0.38      0.55       222
Renaissance        0.67      1.00      0.80       284

    accuracy                           0.73       506
   macro avg       0.84      0.69      0.68       506
weighted avg       0.82      0.73      0.69       506
```

Figure 26: Spoljašne mere kvaliteta klasterovanja

Hijerarhijsko klasterovanje

Za razliku od algoritma K-sredima, koji je zasnovan na reprezentativnim predstavnicima, osnovna ideja hijerarhijskog klasterovanja je formiranje skupa ugnježdenih klastera koji se organizuju u obliku hijerarhije po nivoima. Rezultati se najčešće prikazuju crtanjem dendograma. Postoje dve osnovne vrste ovih algoritama – sakupljajuće i razdvajajuće klasterovanje. U praksi se mnogo češće sreće sakupljajuće klasterovanje i njime ćemo se i baviti u ovom radu. U tom cilju koriste se razne metode od kojih su neke metod zasnovana na varijansi, Vardov metod, Lens-Vilijamsova formula sličnosti klastera.

Podatke ćemo redukovati na tri dimenzije PCA metodom. Na slici (Figure 27) prikazani su redukovani podaci u 3D prosotoru. Ono što sa slike možemo zaključiti je da imamo kandidate za elemente van granica - outlier-e. Osim grafički, outlier-e možemo detektovati i uz pomoć dendograma. Naime, na dendogramu su potencijalni outlier-i one instance ili grupe instanci kojima je trebalo znatno više vremena da se pridruže nekoj većoj grupi nego ostalim podacima.

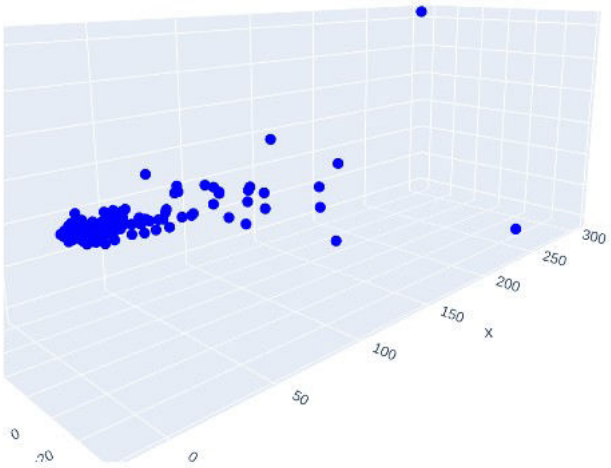


Figure 27: Redukovani podaci u tri dimenzije

Koristćemo Vardovu metodu jer se u praksi pokazala manje osetljivom na outlier-e i šumove, ali ima naklonost ka globularnim klasterima.

Na narednoj slici možemo videti dendogram ograničen na 30 (Figure 28).

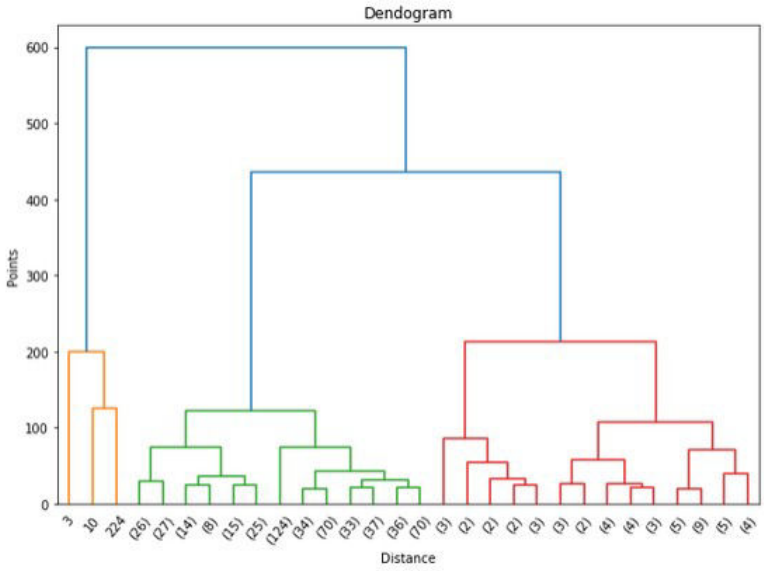
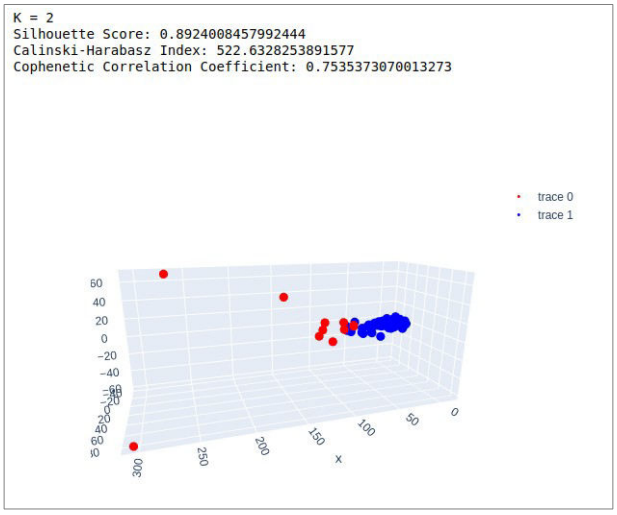
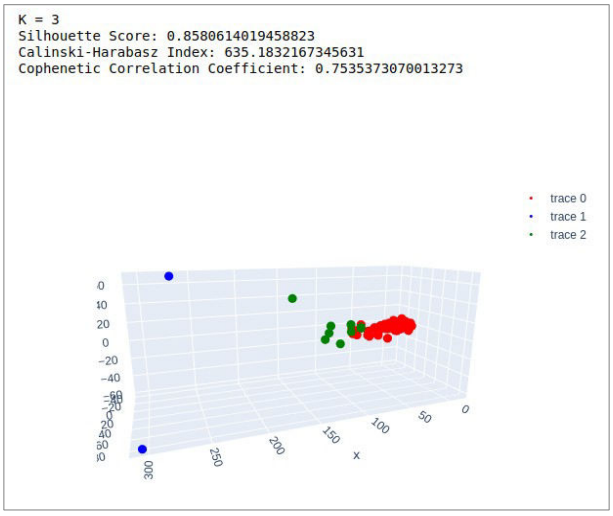
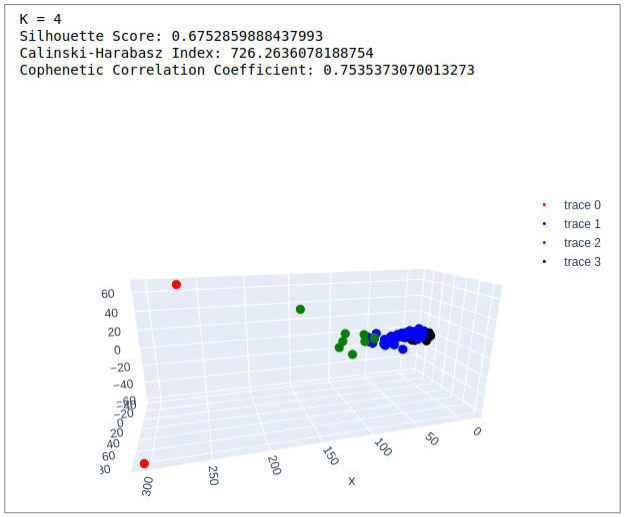


Figure 28: Dendogram

U nastavku primenićemo sakupljajući algoritam za klasterovanje na naš skup podataka za broj klastera K=2, K = 3 i K = 4. Vizuelni prikaz i neki unutrašnji kriterijumi slede u nastavku. Spaljašnjim kriterijumima kvaliteta klasterovanja se u ovom delu nećemo baviti.







Korišćeni su unutrašnji kriterijumi ocene kvaliteta klasterovanja silueta, kriterijum odnosa varijanse (Calinski-Harabasz Index)[2], kofenetski koeficijen korelacije (Cophenetic Correlation Coefficient)[3]. Na osnovu predstavljenih unutrašnjih kriterijuma, zaključujemo da je klasterovanje najuspešnije za K = 2. Ono što bi takođe vredelo razmotriti jesu elementi van granica – outlier-i. Naime, vizuelno već možemo zaključiti da, u slučaju K = 2, elementi prvog klastera se mogu smatrati outliar-ima. Njihovom eliminacijom dobili bismo, potencijalno, kvalitetniji skup podataka koji bismo nadalje koristili.

2 Unutrašnja mera kvaliteta klasterovanja koja se računa po formuli (SSB / SSW) × ((n − k) / (k − 1)), gde SSB predstavlja ukupnu varijansku između klastera, SSW ukupno varijansu između objekata klastera, n broj instanci, a k broj klastera. Što je vrednost veća, to se klasterovanje smatra kvalitetnijim. Prednost je brzo izračunavanje. Obično se koristi u kombinaciji s još nekim kriterijumom.

3 Mera specifična za hijerarhijsko klasterovanje jer meri odnos prividnih sličnosti (udaljenosti) u dendogramu i stvarnih podatka u parovima. Što je vrednost bliža jedinici, klsterovanje se smatra kvalitetnijim.

Pravila pridruživanja

U ovom delu bavimo se određivanjem pravila pridruživanja. Ona predstavljaju skrivene veze među podacima. Motiv nastanka bio je sadržaj potrošačke korpe, odnosno svake transakcije na kasi. Naime, svaka transakcija sastoji se od skupa stavki. Iz ovih transakcija, doneseni su zaključci o tome koji se artikli kupuju zajedno.

U smislu obrade teksta, pravila pridruživanja govore nam o tome koje se to reči pojavljuju zajedno u datim tekstovima. Kako se, u našim podacima, tekstualni sadržaj nalazi u atributu content, na njega ćemo se fokusirati u ovom delu.

Pristup kao u slučaju potrošačke korpe

Pretpostavimo da za cilj imamo određivanje reči koje se zajedno pojavljuju u tekstu, bez obzira na redosled. U ovom slučaju, problem se svodi na problem potrošačke korpe, gde svaki tekst predstavlja jednu transakciju, a reči teksta predstavljaju stavke te transakcije.

Pretprocesiranje

Podsetimo se pretprocesiranja urađenog pre procesa klasifikacije i klasterovanja podatka. Naime, izbacili smo znakove interpukcije iz tekstova, sva slova pretvorili u mala, tekstove smo izdelili u termove (reči) od kojih smo neke (stop reči) potpuno izbacili. Isti proces ponavljamo i u cilju pretprocesiranja podataka za primenu algoritma za određivanje pravila pridruživanja.

```
0      [thus, sole, threneto, doth, augur, seem, near...
1      [mirth, like, coming, thus, thoughts, seen, im...
2      [circes, excels, men, vice, puts, beyond, mean...
3      [bathed, bright, silver, turnings, pas, winged...
4      [never, might, see, day, thus, clouds, sometim...
                           ...
568    [never, heart, hearthe, leaned, flood, shows, ...
569    [publishing, vi, liveright, marc, copyright193...
570    [face, many, beside, mountains, deep, take, sl...
571    [hunger, doors, day, window, sit, wandering, s...
572    [burning, shade, breasts, ecstasy, hillsand, f...
Name: content, Length: 573, dtype: object
```

Figure 29: Tekstovi kao skupovi termova

	document	word
0	0	thus
1	0	sole
2	0	threneto
3	0	doth
4	0	augur
...	...	...
41	572	cypress
42	572	whispered
43	572	sun
44	572	reedsand
45	572	potuia

43994 rows × 2 columns

Figure 30

Sličnosti pretprocesiranja ovde se završavaju. Frekvencije reči u svakom od tekstova nam nisu od značaja, pa sve duplikate termova možemo izbrisati. Dakle, svaki tekst možemo predstaviti kao skup termova (Figure 29).

Pogodniji oblik, sličan onom koji vezujemo za potrošačku korpu, dobićemo ako podatke predstavimo u dve kolone. U prvoj je broj transakcije - odnosno broj dodeljen tekstu, a u drugoj koloni su stavke – termovi/reči (Figure 30).

Apriori algoritam

Apriori algoritam je algoritam koji za određivanje čestih skupova stavki koristi apriori princip[4] i anti-monotonost[5]. Pravi nekoliko prolaza kroz podatke i pretpostavlja da su stavke uređene leksikografski. Zbog toga je prethodno pretprocesirane podatke potrebno i sortirati (Figure 31). Pretprocesiranje podataka odrađeno je u programskom jeyziku Python i ovakav oblik podataka je pogodan za prime nu apriori algoritma u alatu IBM SPSS modeler. U programskom jeziku python za primenu tradicionalno apriori algoritma, podatke je potrebno transformisati u binarnu reprecentaciju (Figure 32). Oba pomenuta oblika imaju nedostatak da ignorišu podatke o količini i ceni stavki. Kako želimo da prikažemo običan "problem potrošačke korpe" i odredimo koje se to reči zajedno pojavljuju u tekstu, ovi nedostaci nam ne predstavljaju problem.

	document	word
111	0	anthem
158	0	appalledthat
103	0	arabian
31	0	asunder
4	0	augur
...	...	...
22	572	thou
40	572	thy
26	572	told
31	572	upon
42	572	whispered

Figure 31

word document	abandonfrom	abandonment	abashed	abasht	abate	abating	abed	abhor	abhorring	abided	...	yvory	ywis	zeal	zealous	zenophontes	zephyr
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
502	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
503	0	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
504	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
505	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	

Figure 32

Primenom apriori algoritma određujemo česte skupove stavki, minimalnu podršku postavićemo na 0.05. Ovo je opravdano time što je skup stavki veliki i povećanjem podrške bismo potencijalno izgubili skupove stavki koji imaju visoku pouzdanost. Neka od pravila dobijena primenom algoritma za kreiranje pravila odlučivanja od čestih skupova stavki prikazana su u nastavku (Figure 33). Ono što bi takođe moglo biti korisno jeste pronaći pravila pridruživanja datih tekstova

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(poems)	(permission)	0.114625	0.154150	0.112648	0.982759	6.375332	0.094979	49.059289	0.952303
1	(publishing)	(permission)	0.059289	0.154150	0.059289	1.000000	6.487179	0.050149	inf	0.899160
2	(reprinted)	(permission)	0.086957	0.154150	0.086957	1.000000	6.487179	0.073552	inf	0.926407
3	(used)	(permission)	0.071146	0.154150	0.065217	0.916667	5.946581	0.054250	10.150198	0.895551
4	(used)	(poems)	0.071146	0.114625	0.061265	0.861111	7.512452	0.053110	6.374704	0.933288
5	(thee)	(thy)	0.154150	0.209486	0.124506	0.807692	3.855588	0.092214	4.110672	0.875612

Figure 33: Pravila pridruživanja

posmatranih kao sekvencijalnih podataka, u kojima mi poredak reči takođe igrao ulogu, ali time se ovde nećemo baviti.

4 Apriori princip: Ako je skup stavki čest, česti su i njegovi podskupovi.

5 Osobina anti-monotonosti: Ako skup stavki nije često, onda ni njegovi nadskupovi nisu česti.

Pravila pridruživanja za kategoričke atribute

Ako bismo izdvojili kategoričke atribute age, type i author iz početnog skupa podataka, iz njih bismo možda mogli da izvučemo neke korisne obrasce. Kako algoritmi za određivanje frekventnih skupova stavki i pravila pridruživanja ne rade direktno sa kategoričkim atributima, pomenute je neophodno nekako transformisati. To možemo uraditi tako što ćemo svaki od kategoričkih atributa zameniti sa onoliko binarnih atributa koliko zamenjeni atribut ima vrednosti (Figure 34).

	Love	Modern	Mythology & Folklore	Nature	Renaissance	archibald macleish	asil bunting	carl sandburg	christopher marlowe	conrad aiken	...	t s eliot	thomas bastard	thomas campion	thomas heywood	thomas lodge	thomas nashe
0	False	False	True	False	True	False	False	False	False	False	...	False	False	False	False	False	False
1	False	False	True	False	True	False	False	False	False	False	...	False	False	False	False	False	False
2	False	False	True	False	True	False	False	False	False	False	...	False	True	False	False	False	False
3	False	False	True	False	True	False	False	False	False	False	...	False	False	False	False	False	False
4	False	False	True	False	True	False	False	False	False	False	...	False	False	False	False	False	False

Figure 34: Binarizacija kategoričkih atributa

Jedan od problema do kojih može doći prilikom ovakve transformacije je dobijanje beskorisnih pravila, kao što je npr. {Love = True} -> {Modern = False}. Ovo se rešava ograničavanjem frekventnih skupova stavki na one atribute koji nisu nastali binarizacijom istog atributa. Još neki problemi koji se mogu javiti su dobijenje prevelikog broja atributa ili atributa koji se prečesto pojavljuju. Prvi možemo rešiti dodatnim grupisanjem atributa, a drugi jednostavnim izbacivanjem atributa koji se prečesto pojavljuje.

Primenom Apriori algoritma za pronalaženje čestih skupova stavki i algoritma za kreiranje pravila prdruživanja izdvajaju se sledeća pravila prikazana na slici ispod (Figure 35).

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(Love)	(Renaissance)	0.568935	0.549738	0.424084	0.745399	1.355916	0.111318	1.768498	0.608937
1	(Renaissance)	(Love)	0.549738	0.568935	0.424084	0.771429	1.355916	0.111318	1.885908	0.582974
2	(Nature)	(Modern)	0.328098	0.450262	0.235602	0.718085	1.594817	0.087872	1.950015	0.555094
3	(william shakespeare)	(Renaissance)	0.123909	0.549738	0.123909	1.000000	1.819048	0.055792	inf	0.513944

Figure 35: Pravila pridruživanja

Dakle, možemo zaključiti da su se u doba Renesanse pisala najčešće ljubavna dela, a i to da su se ljubavna dela generalno, najčešće pisala u doba renesanse. Ako je delo o prirodi, najverovatnije se radi o modernom dobu. Naravno, Šekspir je pisao u doba Renesanse, jer je tada i živeo.

Zaključak

U radu su prikazani različiti algoritmi u cilju klasifikovanja, klasterovanja i određvanja pravila pridruživanja skupa podataka koji sadrži podatke o književnim delima. Kako je fokus bio na rešavanju pomenutih problema u odnosu na sadržaj književnih dela, zaključili smo da problem koji rešavamo spada u probleme obrade teksta. Kao takav, sa sobom donosi određene probleme i olakšice.

Ono što je olakšavajuća okolnost, kada je obrada teksta u pitanju, jeste to da nam nije potrebno nikakvo prethodno domensko predznanje o podacima koji se obrađuju. Pretprocesiranje podataka je intuitivnije. Naime, ako bismo rešavali problem koji uključuje vremenske serije koje opisuju pojmove iz ekonomije, bilo bi nam potrebno domensko znanje iz ekonomije da prepoznamo anomalije, odaberemo karakteristike i slično. U slučaju obrade teksta, to nije neophodno. Ono što smo takođe primetili je da klasterovanje može biti odlična vrsta analize i pretprocesiranja podataka. To naravno, važi i inače, a ne samo u smislu obrade teksta.

Problem do kog će obično doći kada se koristi ovakvo ili slično pretprocesiranje tekstualnih podataka je velika dimenzionalnost podataka. To nam otežava vizuelizaciju podatka, a takođe treba posebno obratiti pažnju na pomenutu pojavu prokletstva dimenzionalnosti. Ukoliko se bavimo klasifikacijom teksta, a primetimo izraženu nebalansiranost klasa na skupu na kom treniramo model, potrebno je, naravno, primeniti neku od tehnika i izbalansirati klase. Pretpostavimo da smo se odlučili za vrstu oversamplinga koja uključuje, ne samo kopiranje, već veštačko kreiranje novih instanci. U ovom slučaju treba voditi računa i o tome da će ove tehnike, koje su pretežno predviđene za skupove podataka sa neprekidnim atributima, možda napraviti instance koje uopše nisu reprezentativni predstavnici klasa. Dakle, iako smo date tekstove transformisali u numerički oblik, uvek treba imati na umu da je ono što zapravo obrađujemo ovde tekst.

Literatura

- https://infoteh.etf.ues.rs.ba/zbornik/2010/radovi/E1/E1-6.pdf
- https://www.analyticsvidhya.com/blog/2021/05/k-mean-getting-the-optimal-number-of-clusters/
- Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar: Introduction to Data Mining, 2nd ed, Pearson Education, 2019.
- https://www.hrpub.org/download/20170830/WJCAT2-13708454.pdf
- Xindong Wu, Vipin Kumar (eds.): The Top Ten Algorithms in Data Mining, CRC Press, 2009.
- Slajdovi sa predavanja prof. Nenada Mitića
- Materijali korišćeni na vežbama u toku semestra
- Mladen Nikolić, Predrag Janičić: Veštačka inteligencija