Twitter Sentiment Analysis

Filip Jovanovic

Sadrzaj

Uvod

U nastavku ce biti prikazan rad nad skupom podataka za analizu tweet-ova. Zadatak je da se na osnovu tweet-a zakljuci da dati tweet sadrzi ili ne sadrzi govor mrznje. Na pocetku ce biti govora o pretprocesiranju I obradi teksta nakon cega ce uslediti priprema podataka za algoritme klasifikacije I klasterovanja.

Analiza skupa

Skup podataka je podeljen na test/train podatke medjutim u test.csv fajlu nedostaje atribut “label” koji oznacava da li je dati tweet oznacen kao tweet koji sadrzi govor mrznje, pa se zato dati skup ne korisit vec koristimo skup train.csv koji cemo kasnije podeliti na train/test podatke.

Prikaz prvih 10 instanci naseg skupa:

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is so selfish he drags his kids into his dysfunction. #run
1	2	0	@user @user thanks for #lyft credit i can't use cause they don't offer wheelchair vans in pdx. #disapointed #getthanked
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in urð□□±!!! ð□□□ð□□□□ð□□□ð□□□ ð□□¦ð□□¦ð□□¦
4	5	0	factsguide: society now #motivation
5	6	0	[2/2] huge fan fare and big talking before they leave. chaos and pay disputes when they get there. #allshowandnogo
6	7	0	@user camping tomorrow @user @user @user @user @user @user @user dannyâ¬¦
7	8	0	the next school year is the year for exams.ð□□¯ can't think about that ð□□ #school #exams #hate #imagine #actorslife #revolutionschool #girl
8	9	0	we won!!! love the land!!! #allin #cavs #champions #cleveland #clevelandcavaliers â¬¦
9	10	0	@user @user welcome here ! i'm it's so #gr8 !

Skup ima 31,962 instance.
Iz prilozenog se moze videti da postoje 3 atributa:

- id – Jedinstveni broj tweet-a
- label – Da li je tweet okarakterisan kao tweet sa govorom mrznje ili ne, odnosno vrednost 0 / 1
- tweet – Tekst koji predstavlja tweet odnosno objavu na mrezi Twitter

Kako je atribut “id” irelevantan njega izbacujemo.

Zatim sledi obrada nedostajucih vrednosti. Iako nas skup ne sadrzi “NaN” vrednosti potrebno je proveriti sta zapravo NaN vrednost moze predstavljati u slucaju obrade teksta, pa je potrebno izbaciti prazne tweet-ove u slucaju da oni postoje.

Nakon obradjivanja nedostajucih vrednosti proveravamo da li nas skup podataka sadrzi duplikate, odnosno identicne tweet-ove. I nakon izbacivanja duplikata nas skup se od inicijalnih 31,962 instanci sveo na skup od 29,530 instaci I 2 atributa

Obrada duplikata

```python
print(f'Ima {len(data.loc[data["tweet"].duplicated() == True])} duplikata') if data.duplicated().any() else print('Nema duplikata')
```

Python

```
Ima 2432 duplikata
```

```python
data = data.drop_duplicates(keep='first')
data.shape
```

Python

```
(29530, 2)
```

Naredni korak predstavlja sama obrada tweet-ova. Iz tweet-ova se izbacuju tzv. “stopwords” (npr. “I”, “myself”, “your”, “like”, …), znaci interpunkcije (“,”, “.”, “!”, …), emojis (“:)”, “:(“, “;)”, …) i sl. sto ne moze znacajnije uticati na tekst. Takodje se vrsi uklanjanje brojeva, smanjenje svih slova, izbacivanje ‘@user’ reci koja zapravo predstavlja tagovanje na drustvenim mrezama, lematizacija koja reci svodi na njihov osnovni oblik (npr. [“improve”, “improving”, “improved”] i sl. ce se svesti na “improve”). Takodje pored lematizacije postoji I stemming cija je funkcija uklanjanje poslednjih karaktera koji vode ka pogresnom zakljucku I spelovanju.

Obrada teksta

	label	tweet	tweet_cleaned
0	0	@user when a father is dysfunctional and is so selfish he drags his kids into his dysfunction. #run	father dysfunctional selfish drag kid dysfunction run
1	0	@user @user thanks for #lyft credit i can't use cause they don't offer wheelchair vans in pdx. #disapointed #getthanked	thanks lyft credit use cause offer wheelchair van pdx disapointed getthanked
2	0	bihday your majesty	bihday majesty
3	0	#model i love u take with u all the time in urð□□±!!! ð□□□ð□□□ð□□□ð□□□ ð□□¦ð□□¦ð□□¦	model love u take u time ur
4	0	factsguide: society now #motivation	factsguide society motivation
5	0	[2/2] huge fan fare and big talking before they leave. chaos and pay disputes when they get there. #allshowandnogo	huge fan fare big talking leave chaos pay dispute get allshowandnogo
6	0	@user camping tomorrow @user @user @user @user @user @user @user dannyâ¬¦	camping tomorrow danny
7	0	the next school year is the year for exams.ð□□¯ can't think about that ð□□ #school #exams #hate #imagine #actorslife #revolutionschool #girl	next school year year exam think school exam hate imagine actorslife revolutionschool girl
8	0	we won!!! love the land!!! #allin #cavs #champions #cleveland #clevelandcavaliers â¬¦	love land allin cavs champion cleveland clevelandcavaliers
9	0	@user @user welcome here ! i'm it's so #gr8 !	welcome gr

Nakon obrade teksta moramo opet proveriti da li postoji prazan tweet, jer je moguce da sam tekst bude samo tag ili neka od stopwords ili nesto slicno. Treba napomenuti da smo prilikom analize nedostajucih vrednosti mogli da izvrsimo neku od metoda rada sa nedostajucim vrednostima:

- Popunjavanjem default vrednostima, ovde se postavlja pitanje sta izabrati kao default vrednost, u slucaju da nemamo vrednost za label atribut ako izaberemo 0 tweet-ovi koji sadrze govor mrznje mogu biti okarakaterisani kao tweet-ovi koji ne sadrze govor mrznje, slicno za 1)
- Popunjavanje unapred / unazad
  - ffill() - popunjava null vrednost na osnovu vrednosti koja se nalazi iznad nje
  - bfill() popunjava null vrednost na osnovu vrednosti koja se nalazi ispod nje
- …

Kako nas skup ne sadrzi veliki broj nedostajucih vrednosti mi ih mozemo izbaciti iz skupa.

Nakon obrade tweet-ova potrebno je podeliti nas skup podataka train.csv na prave skupove za trening odnosno za testiranje. Ulazni parametar naseg modela ce predstavljati tweet, dok ce izlazni parametar ce predstavljati odgovor na pitanje da li tweet sadrzi govor mrznje ili ne, odnosno atribut label. Podela se vrsi pozivom funkcije “train_test_split” koja vrsi stratifikovanu podelu (ako postoji ¼ skupa sa labelom 0 I ¾ skupa sa labelom 1 onda ce i train i test podaci imati tu razmeru) pri cemu ce 1/3 skupa biti odvojena za testiranje samog modela.

Podela na train / test skup

```
(X_train, X_test, Y_train, Y_test) = train_test_split(X, Y, stratify=Y, test_size=0.3, random_state=42)
```

Python

```
print(f'Broj instanci trening skupa: {X_train.shape[0]}')
print(f'Broj instanci test skupa: {X_test.shape[0]}')
```

Python

```
Broj instanci trening skupa: 20645
Broj instanci test skupa: 8848
```

Kao parametar za odredjivanje autlajera (outliers) cemo koristiti samu duzinu tweet-a, jer duzina samo teksta moze uticati na nas model prilikom treniranja.

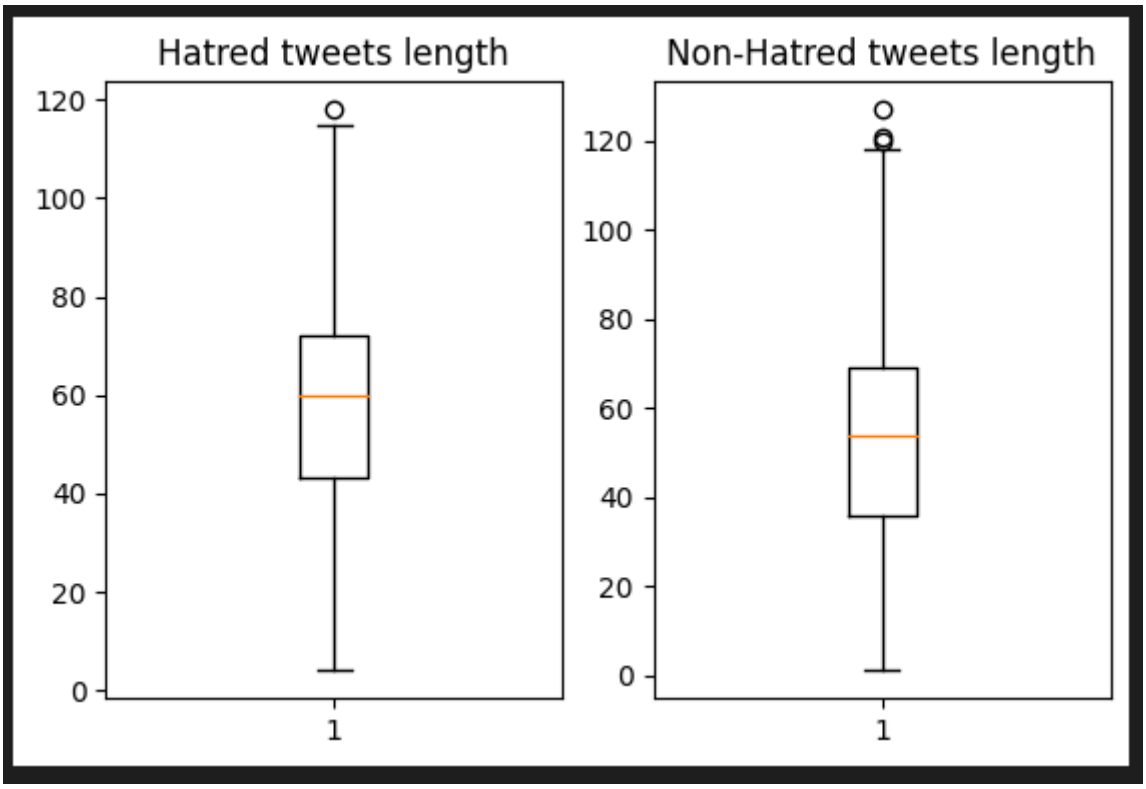
Duzina tweet-a kao parametar za autlajer

	label	tweet	tweet_cleaned	length
0	0	@user when a father is dysfunctional and is so selfish he drags his kids into his dysfunction. #run	father dysfunctional selfish drag kid dysfunction run	53
1	0	@user @user thanks for #lyft credit i can't use cause they don't offer wheelchair vans in pdx. #disapointed #getthanked	thanks lyft credit use cause offer wheelchair van pdx disapointed getthanked	76
2	0	bihday your majesty	bihday majesty	14
3	0	#model i love u take with u all the time in urð□□±!!! ð□□□ð□□□ð□□□ð□□□ ð□□¦ð□□¦ð□□¦	model love u take u time ur	27
4	0	factsguide: society now #motivation	factsguide society motivation	29

Nakon dodavanje date kolone odredjujemo boxplot na osnovu koga mozemo videti kako izgledaju nasi autlajeri

Autlajeri

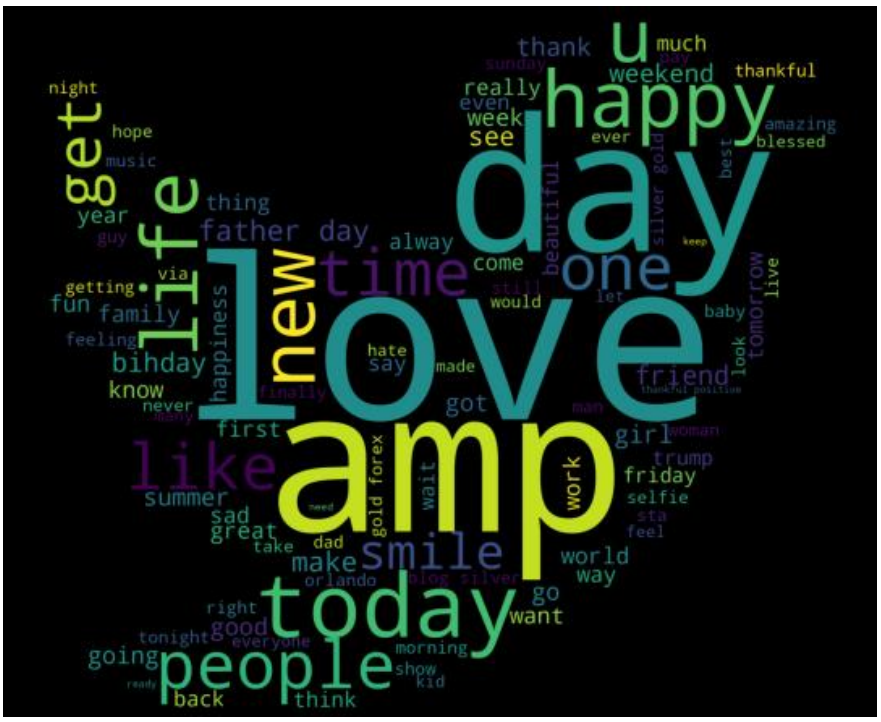
Iz



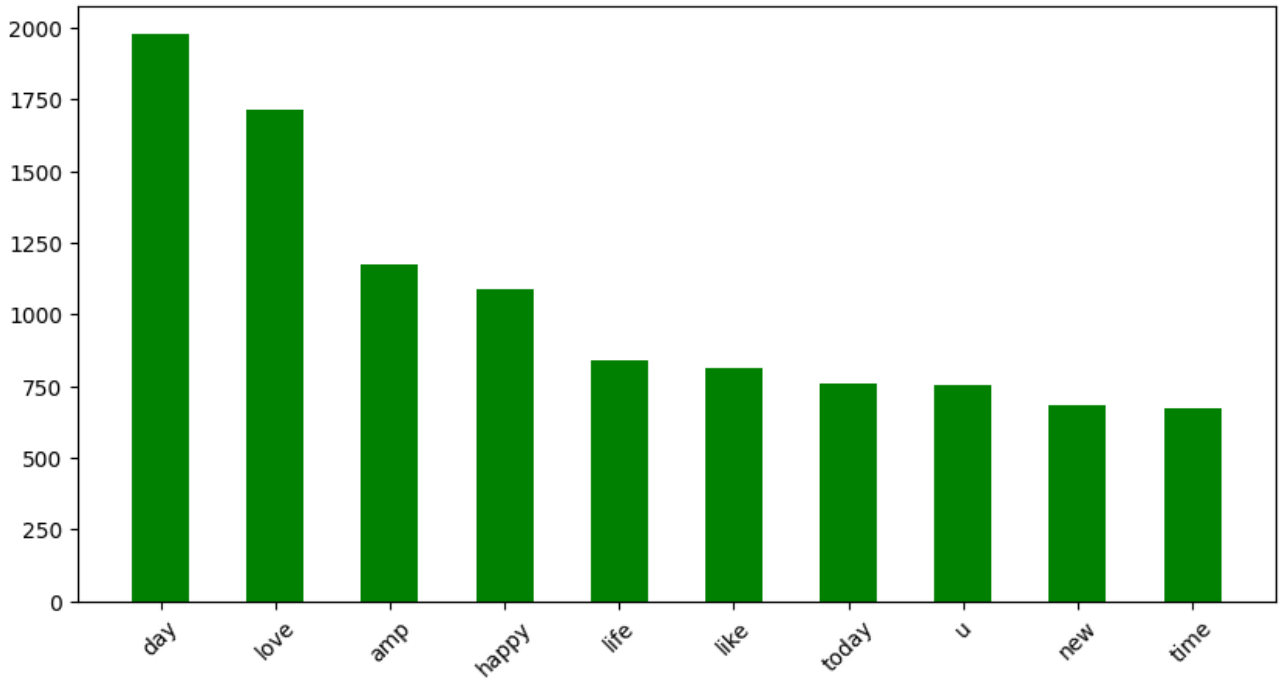
datog skupa izbacujemo autlajere, i to 1 tweet sa govorom mrznje odnosno 3 tweet-a bez govora mrznje

Nakon obrade autlajera odradjen je vizuelni prikaz podataka:

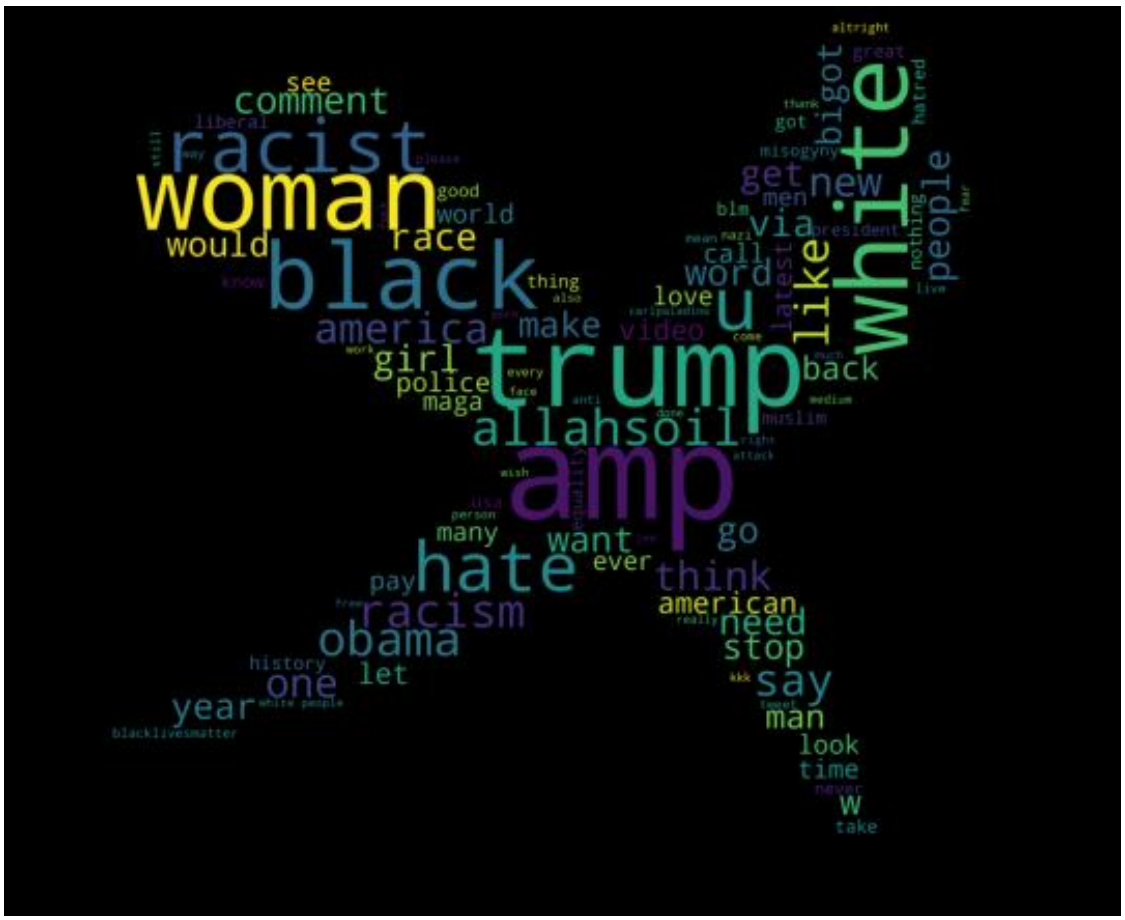
Najzastupljenije reci u trening podacima



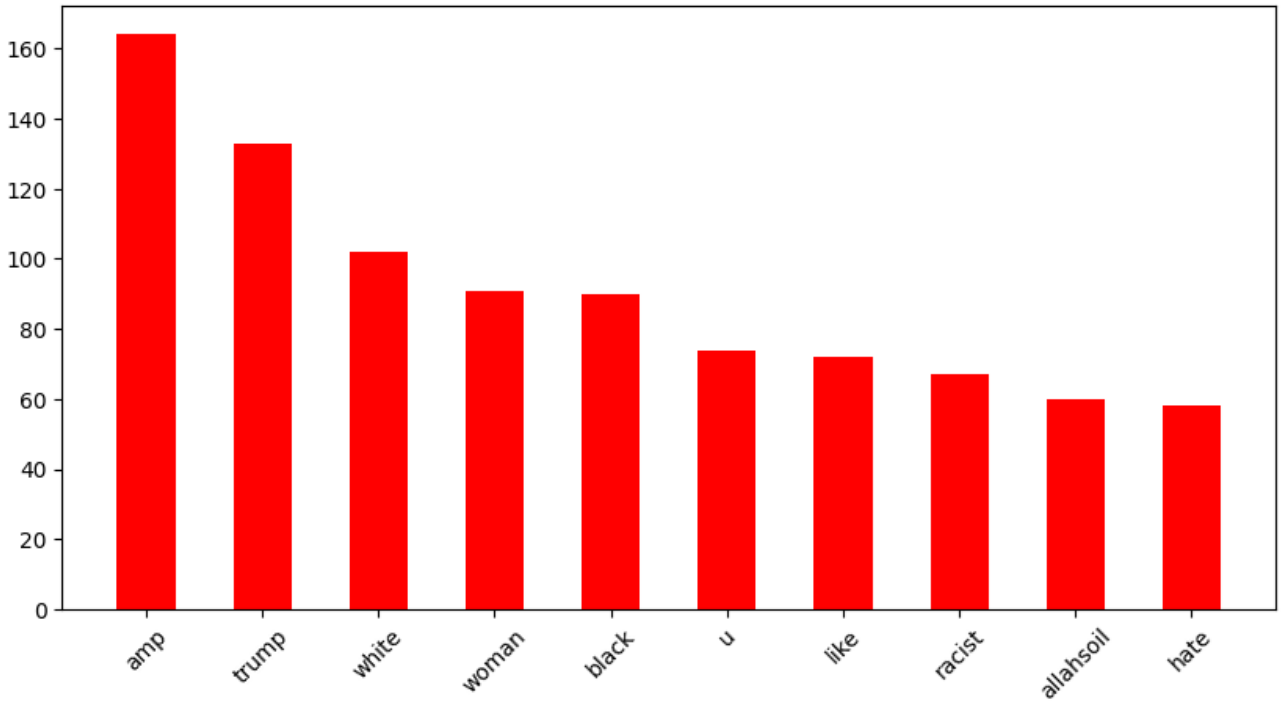
Najzastupljenije reci u trening podacima (graf)



Najzastupljenije reci u tweet-ovima koji sadrze govor mrznje

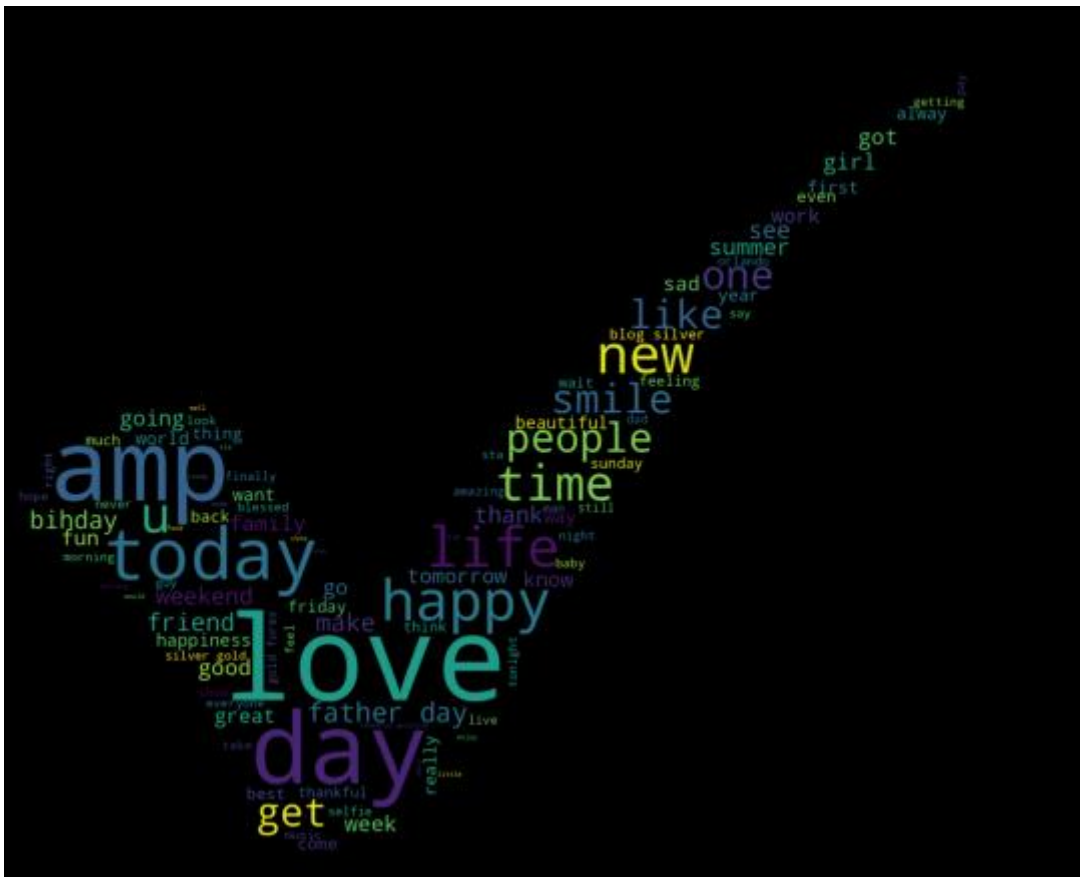


Najzastupljenije reci u tweet-ovima koji sadrze govor mrznje (graf)

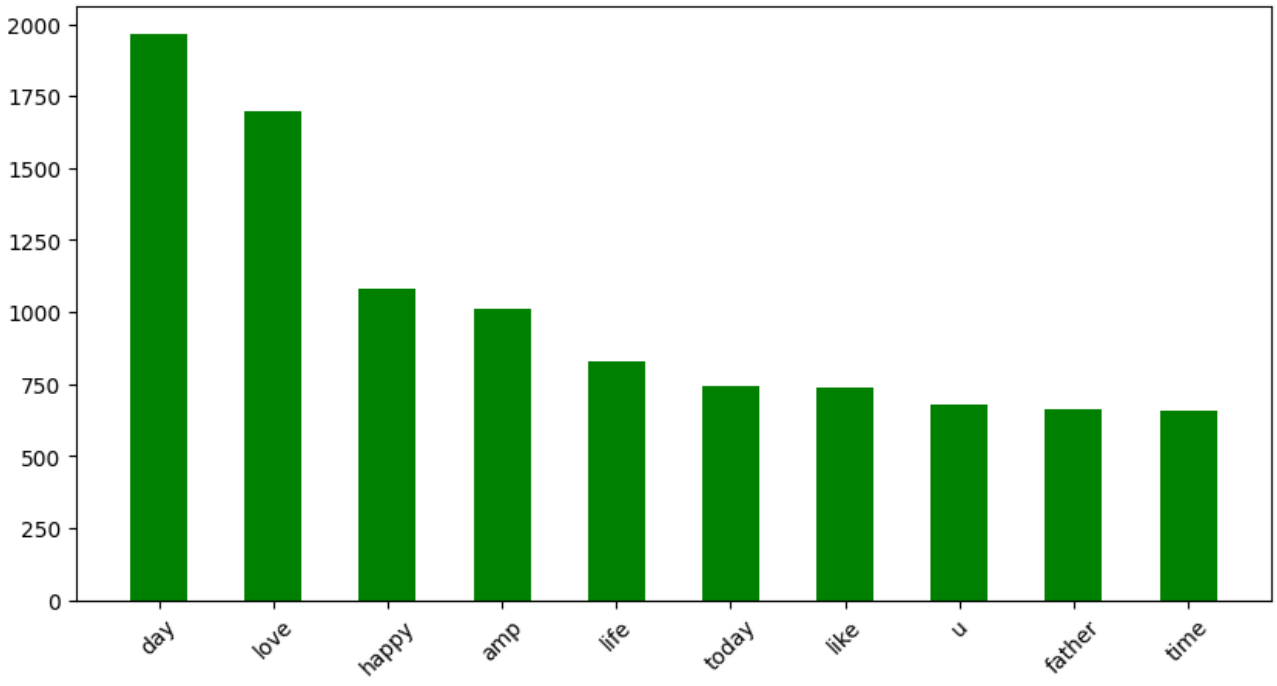


Najzastupljenije reci u tweet-ovima koji ne sadrze govor mrznje

d



Najzastupljenije reci u tweet-ovima koji ne sadrze govor mrznje (graf)



Nakon vizualizacije sledi pregled o odnosu klasa. Kako je odnos klasa 93 % / 7 % u korist tweet-ova koji su okarakterisani kao govor mrznje rec je o jako velikoj nebalansiranosti medju klasama. Problem sa nebalansiranim klasama moze dovesti do toga da nas model preprilagodi podacima koji su okaraksterisani kao tweet-ovi koji ne sadrze govor mrznje I da daje jako dobru preciznost, medjutim kada bi

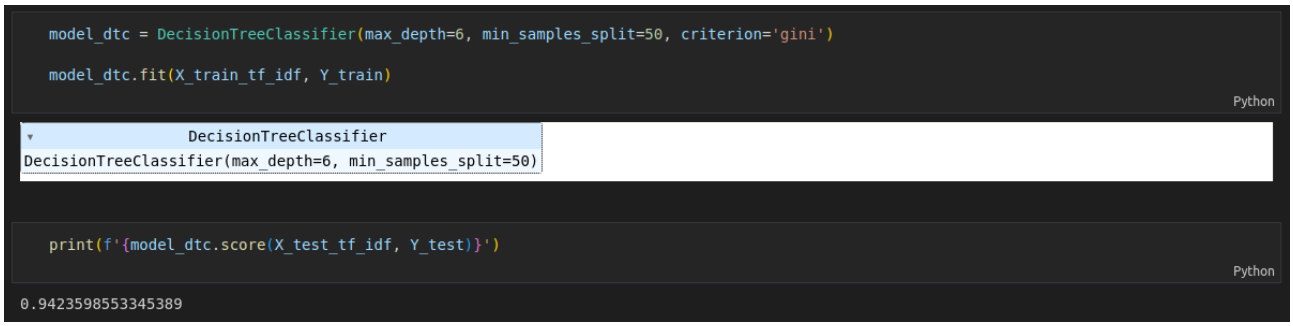
pogledali TF matricu videli bi da model ne vrsi dobru predikciju kada na ulazu dobije hatred tweet.

Neke od tehnika za rad sa nebalansiranim klasama:

- OverSampling (Prilagodjavanje manjeg skupa vecem)
  - RandomOverSampler
  - SMOTE
- UnderSampling (Prilagodjavanje veceg skupa manjem)
  - RandomUnderSampling
  - NearMiss
  - CondenseNearestNeighbour
- Combination Oversampling & Undersampling (Kombinacija prethodna 2)
  - SMOTEENN
- Imbalanced Ensemble
  - BalancedRandomForestClassifier

Pre primene nekog modela potrebno je izvrsiti transformaciju podataka iz tekstualne u numericku kategoriju koristeci “TfidVectorizer” klasu. TF-IDF (Term Frequency - Inverse Document Frequency) matrica koristi frekvenciju pojavljivanja reci da bi odredila koliko je data rec relevantna u datom dokumentu.
Nakon sto smo obradili nase podatke sada ih mozemo proslediti nekom modelu. U ovom primeru cemo iskoristiti kao primer DecisionTreeClassifier da bismo prikazali zasto je potrebno obraditi nebalansiranost podataka.

Treniranje modela



Preciznost naseg modela je 94%, sve je u redu? Za nebalansirane klase mozemo koristiti funkciju “classification_report_imbalanced” koja nam daje bolju sliku modela i njegovih ocena.

Prava slika modela

```
print(classification_report_imbalanced(Y_test, Y_pred))
```

	pre	rec	spe	f1	geo	iba	sup
0	0.95	1.00	0.21	0.97	0.45	0.22	8245
1	0.80	0.21	1.00	0.33	0.45	0.19	603
avg / total	0.93	0.94	0.26	0.93	0.45	0.22	8848

Iz tabele mozemo videti da ocene nisu tako sjajne za nas model.

U ovom primeru smo iskoristili RandomOverSampler kao tehniku koja vrsi prilagodjavanje manje klase vecoj (SMOTE nije mogao da se iskoristi jer je previse zahtevan za lokalni racunar ili Google Colab).

Sledi cuvanje podataka za potrebe algoritama klasifikacije. Obradjene podatke objedinjujemo u 2 skupa (X_train i Y_train u jedan, X_test i Y_test u drugi) i kao takve ih cuvamo u posebne fajlove za prethodno pomenute potrebe.

Zatim sledi cuvanje podataka za potrebe algoritama klasterovanja. Vrsimo objedinjavanje trening i test skupa u jedan skup i uklanjamo atribut label. Cuvamo podatke u posebnom fajlu za klasterovanje.

Algoritmi klasifikacije

Stabla odlucivanja (DecisionTree)

Prethodno sacuvane podatke ucitavamo i delimo na trening i test skup. Nakon toga delimo trening podatke na podatke za treniranje i podatke za validaciju modela. Treniramo nas model i prikazujemo rezultate.

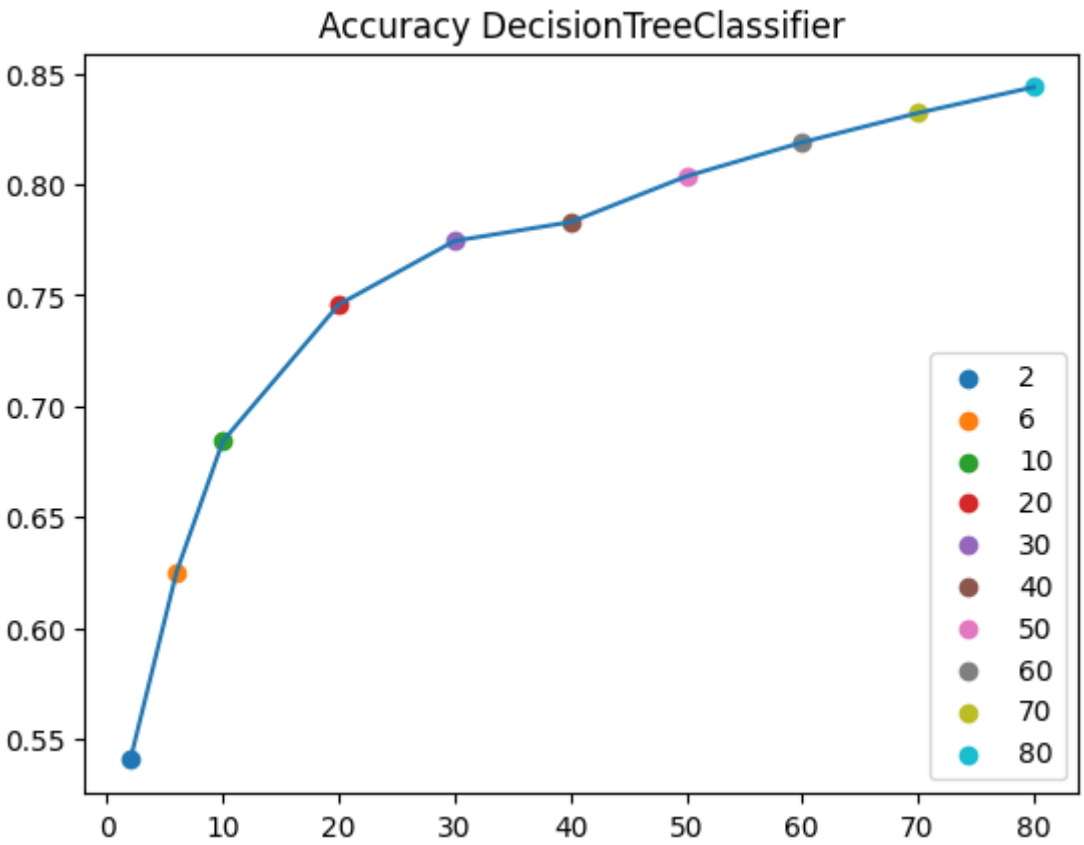
Prikaz rezultata

```
report(dtc_model, X_train, Y_train, 'train')
report(dtc_model, X_test, Y_test, 'test')
```

```
Output exceeds the size limit. Open the full output data in a text editor
Classification report for model DecisionTreeClassifier on train data
--------------------------------------------------------------------------------
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     15391
           1       1.00      1.00      1.00     15391

    accuracy                           1.00     30782
   macro avg       1.00      1.00      1.00     30782
weighted avg       1.00      1.00      1.00     30782

--------------------------------------------------------------------------------
Confusion matrix for model DecisionTreeClassifier on train data
--------------------------------------------------------------------------------
       0      1
0  15355     36
1      6  15385
--------------------------------------------------------------------------------
Classification report for model DecisionTreeClassifier on test data
--------------------------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.97      0.94      0.96      8245
           1       0.44      0.61      0.51       603
...
      0    1
0  7788  457
1   237  366
--------------------------------------------------------------------------------
```

Iako su ocene za trening skup savrsene, podaci nad test skupom nam govore da je doslo do preprilagodjavanja. Jedan od nacina kako da izaberemo pravi model jeste pronalazenje hiper parametra. Takodje vazno je napomenuti da se prilikom trazenja pravog hiperparametra koristi validacioni skup podataka da bi dobili prave ocene modela, inace bi test skup bio kompromitovan, sto ne zelimo. Ispitujemo hiperparametre I biramo najbolji model.

Trazenje hiper parametra



Mozemo videti da preciznost modela raste sa povecanjem hiper parametra, pa cemo kao najbolji parametar uzeti 80 i iskoristiti ga za treniranje najboljeg modela, uz to cemo koristiti kriterijum gini. Takodje vazno je napomenuti da rezultati sa povecanjem hiper parametra iznad 80 nisu bili znacajno bolji pa je zbog toga uzet broj 80 kao hiper parametar.

Najbolji DecisionTree model

```
best_dtc_model = DecisionTreeClassifier(max_depth=80, criterion='gini')
best_dtc_model.fit(X_train, Y_train)

report(best_dtc_model, X_test, Y_test, 'test')
```

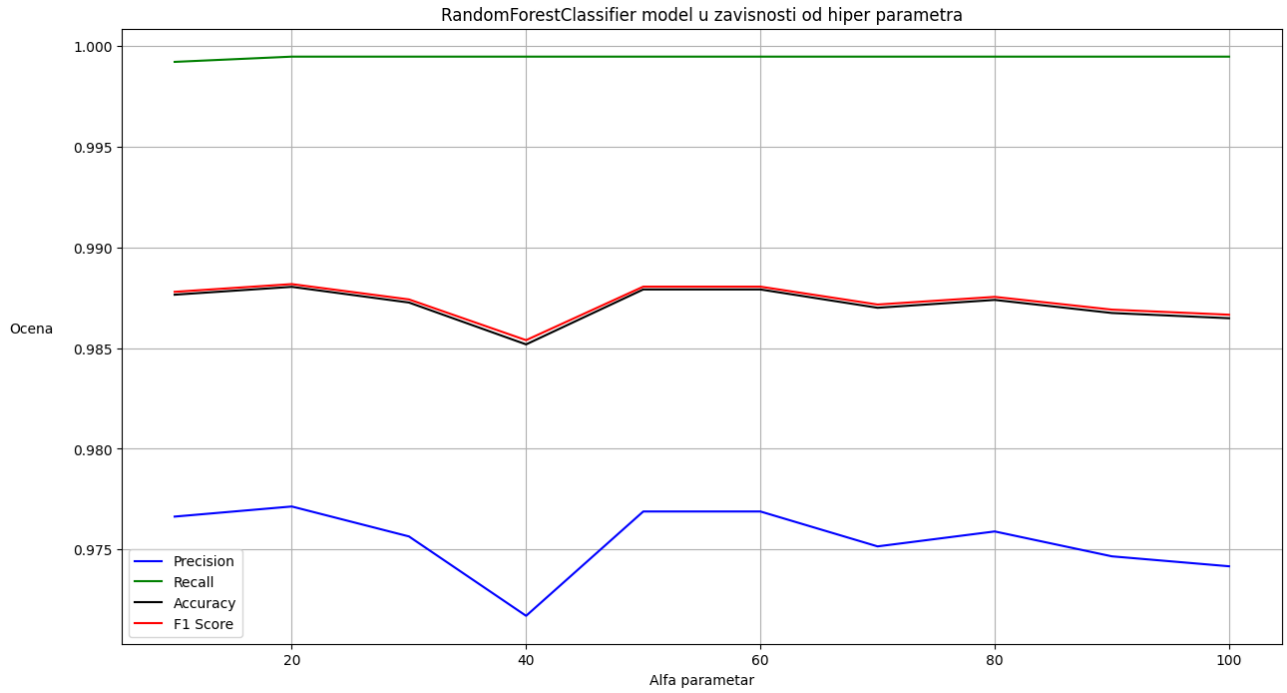
```
Classification report for model DecisionTreeClassifier on test data
--------------------------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.96      0.96      0.96      8245
           1       0.49      0.51      0.50       603

    accuracy                           0.93      8848
   macro avg       0.73      0.74      0.73      8848
weighted avg       0.93      0.93      0.93      8848

--------------------------------------------------------------------------------
Confusion matrix for model DecisionTreeClassifier on test data
--------------------------------------------------------------------------------
      0    1
0  7921  324
1   293  310
--------------------------------------------------------------------------------
```

Mozemo videti da rezultati nisu najbolji ali su znatno bolji nego sto su bili pre tehnike obrade nebalansiranih klasa. Ipak je inicijalni skup podataka sadrzao jako nebalansirane klase pa ne cudi ovako los rezultat.

Slucajne sume (RandomForest)

Slicno kao za prethodno pomenuti algoritam, algoritmu slucajnih suma prosledjujemo parametar n_estimators koji predstavlja broj stabala u sumi. Sledi odabir najboljeg parametra i uporedjivanje modela u zavisnosti od broja stabala u sumi.

RandomForest u zavisnosti od broja stabala



Algoritam RandomForest se najbolje ponasa sa 20 stabala u sumi za nase podatke pa sledi treniranje datog najboljeg modela.

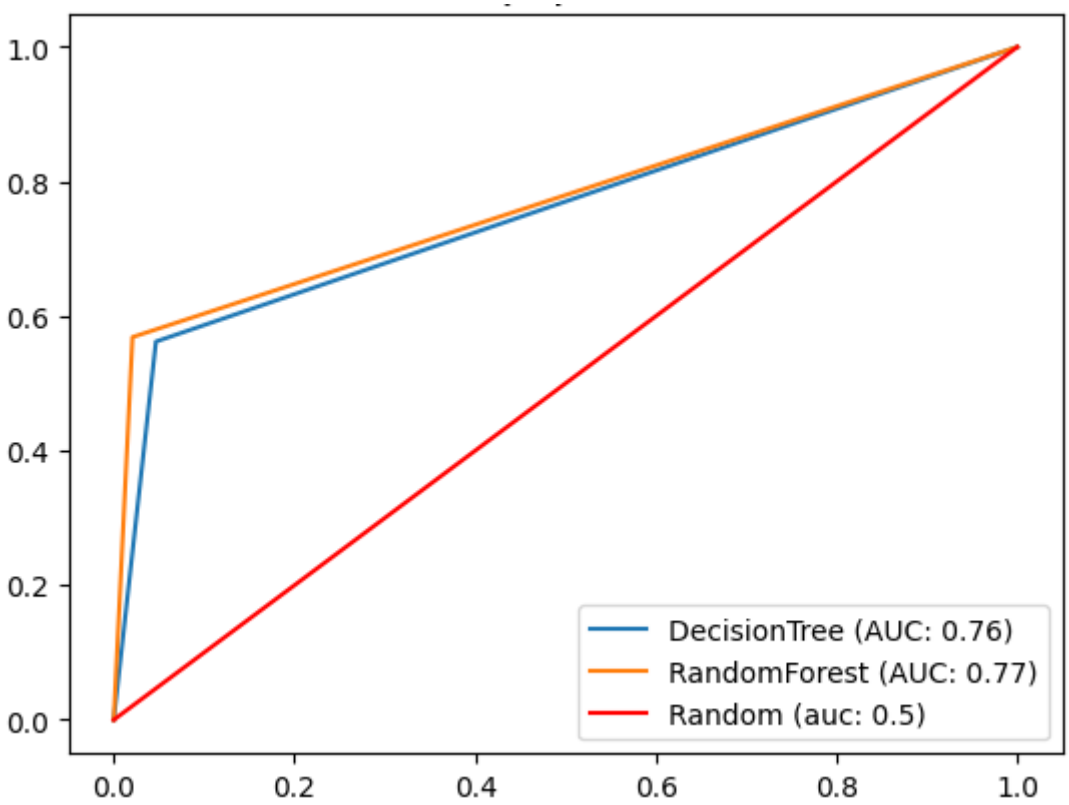
Prikaz rezultata najboljeg modela

```
best_random_forest = RandomForestClassifier(n_estimators=best_estimator)
best_random_forest.fit(X_train, Y_train.values.ravel())

report(best_random_forest, X_test, Y_test, 'test')
```

```
Classification report for model RandomForestClassifier on test data
--------------------------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.97      0.98      0.97      8245
           1       0.67      0.57      0.61       603

    accuracy                           0.95      8848
   macro avg       0.82      0.77      0.79      8848
weighted avg       0.95      0.95      0.95      8848

--------------------------------------------------------------------------------
Confusion matrix for model RandomForestClassifier on test data
--------------------------------------------------------------------------------
      0    1
0  8074  171
1   260  343
--------------------------------------------------------------------------------
```

Nakon obrade oba algoritma sledi poredjenje datih najboljih DecisionTree i RandomForest modela.

Poredjenje modela

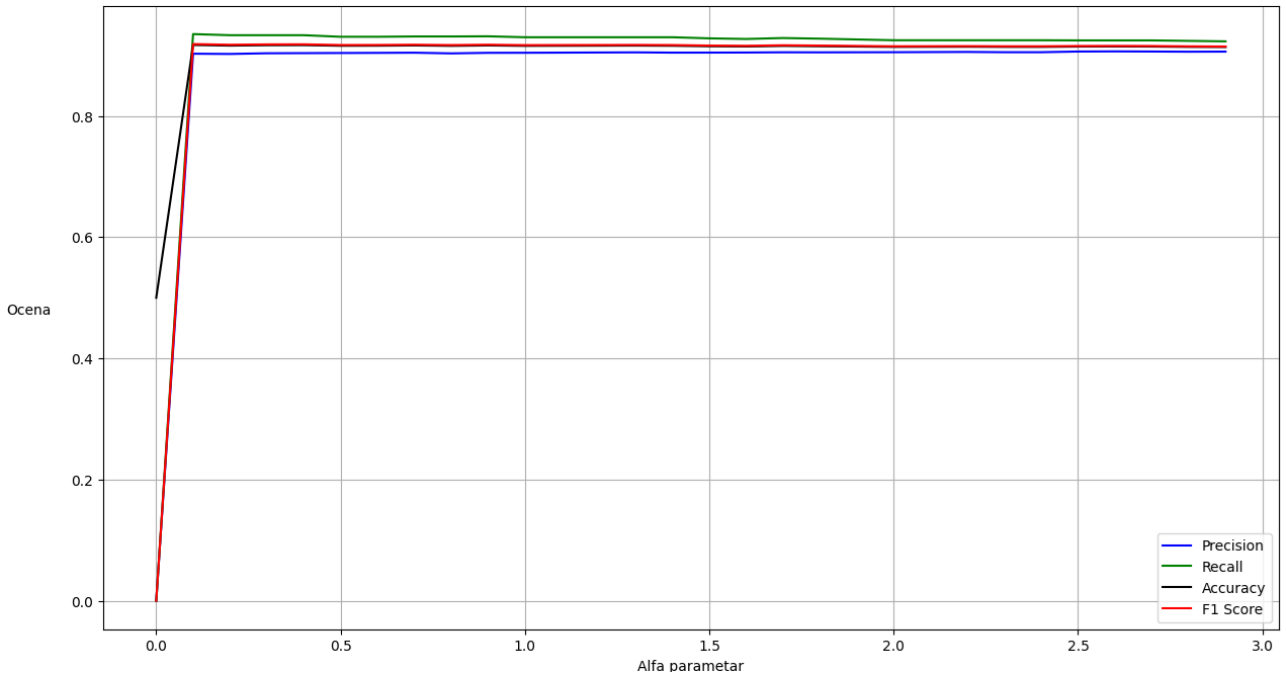


Moze se videti da vecu vrednost AUC-a ima RandomForest algoritam pa cemo njega odabrati kao najbolji model klasifikacije medju odabranim modelima.

Naivni Bajes

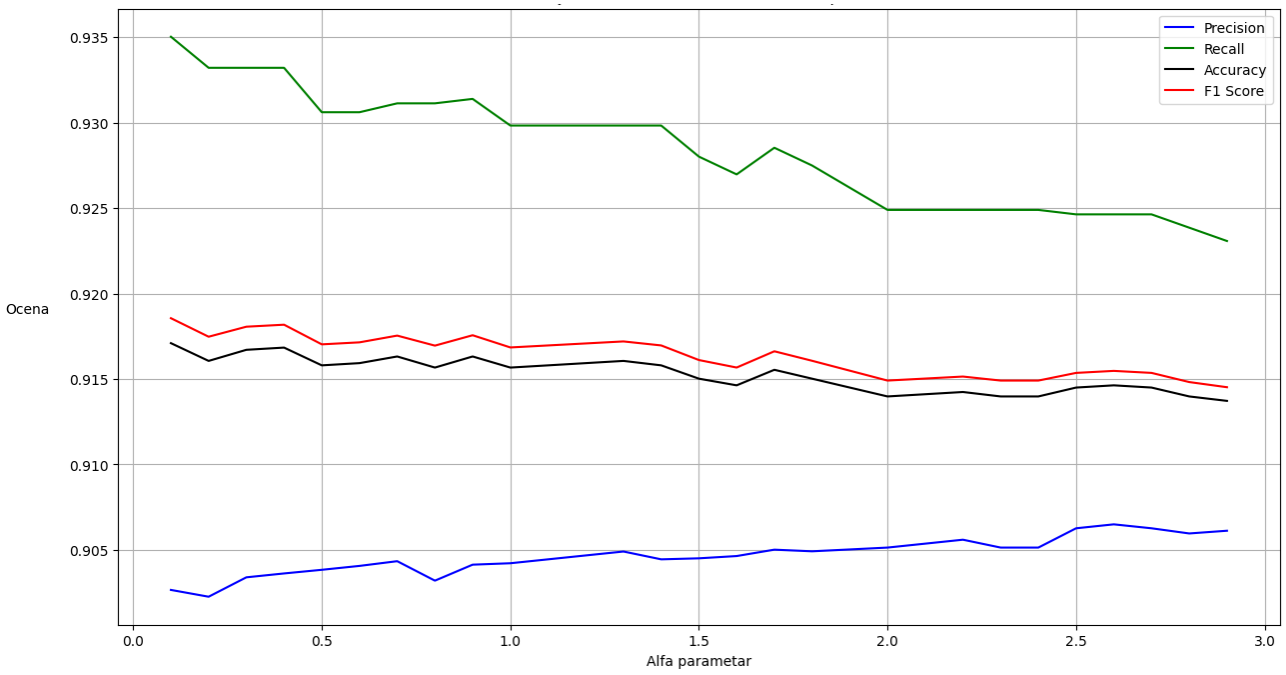
Kao i kod prethodnih algoritama prethodno sacuvane podatke ucitavamo i delimo na trening i test skup. Nakon toga delimo trening podatke na podatke za treniranje i podatke za validaciju modela. Sada za Naivni Bajesov algoritam vrsimo prilagodjavanje modela na osnovu alfa parametra (alfa parametri koji se testiraju su u opsegu od 0 do 3 sa korakom 0.1, tj. 0, 0.1, 0.2, …). Sledi prikaz ponasanja modela u zavisnosti od alfa parametra.

Ocene modela u zavisnosti od alfa parametra



Kako se na grafiku ne vidi bas najjasnije ponasanje ocena modela za alfa > 0.1 uklonicemo ocenu alfa = 0 radi boljeg uvida u podatke.

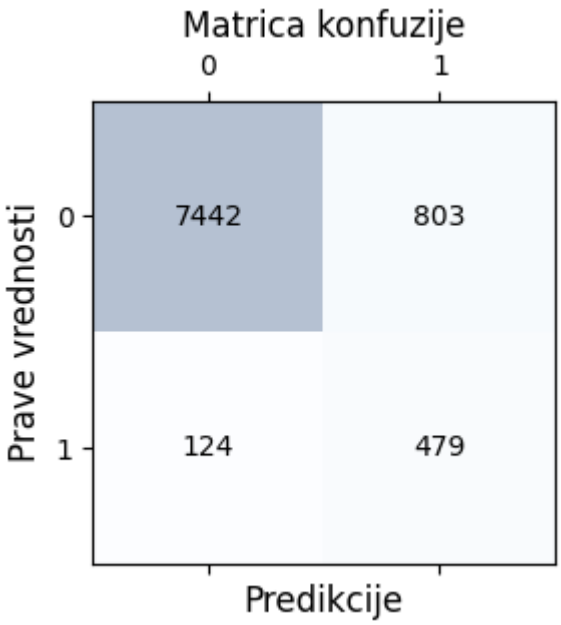
Ocene za alfa >= 0.1



Ovde se jasno vidi da su najbolje ocene za alfa parametar 0.1, pa cemo njega iskoristiti za trening najboljeg modela. Takodje vazno je napomenuti koja predikcija modela je bitnija, FP, ili FN, odnosno da li je bitnije da imamo tweet koji sadrzi govor mrznje a okarakterisan je od strane naseg modela kao tweet koji ne sadrzi govor mrznje ili tweet koji ne sadrzi govor mrznje a okarakterisan je kao tweet koji sadrzi govor mrznje. Ako bi nas model bio taj koji odlucuje doseg tweet-a, odnosno broj ljudi koji bi videli taj tweet, onda bi svakako bilo znacajnije da smanjimo broj tweet-

ova koji sadrze govor mrznje a okarakterisani su kao tweet-ovi koji ne sadrze govor mrznje. U matrici konfuzije se moze videti da je broj gresaak znacajno smanjen.

Matrica konfuzije



Iako je pocetni skup podataka sadrzao veliku nebalansiranost klasa ocene su znatno bolje nego sto su bile pre obrade podataka.

Algoritmi klasterovanja

Kod algoritama klasterovanja nemamo label atribut odnosno nemamo izlaznu promenljivu koja nam govori kojem klasteru instanca pripada, pa algoritmi klasterovanja spadaju u grupu algoritama nenadgledanog ucenja. Ideja klasterovanja jeste da se vrsi grupacija instanci I da se oni dele u grupe odnosno klastere tako da su instance jednog klastera jako slicne (da li je u pitanju Euklidsko rastojanje, max rastojanje, min rastojanje ili neko drugo).

K-means

K-means algoritam deli podatke u K prethodno definisanih klastera i na osnovu sume kvadrata rastojanja izmedju tacaka iz klastera i centroida svrstava instance u klastere.

Nakon ucitavanja podataka a pre instanciranja KMeans-a treba normalizovati podatke. U tu svrhu ce biti koriscen MinMaxScaler. Nakon pokretanja KMeans-a mozemo videti odnos dodeljenih klastera.

Odnos klastera

```
clusters = kmeans.labels_

print(f'Broj instanci koje pripadaju jednom klasteru: {(clusters == 0).sum()}')
print(f'Broj instanci koje pripadaju drugom klasteru: {(clusters == 1).sum()}')
```

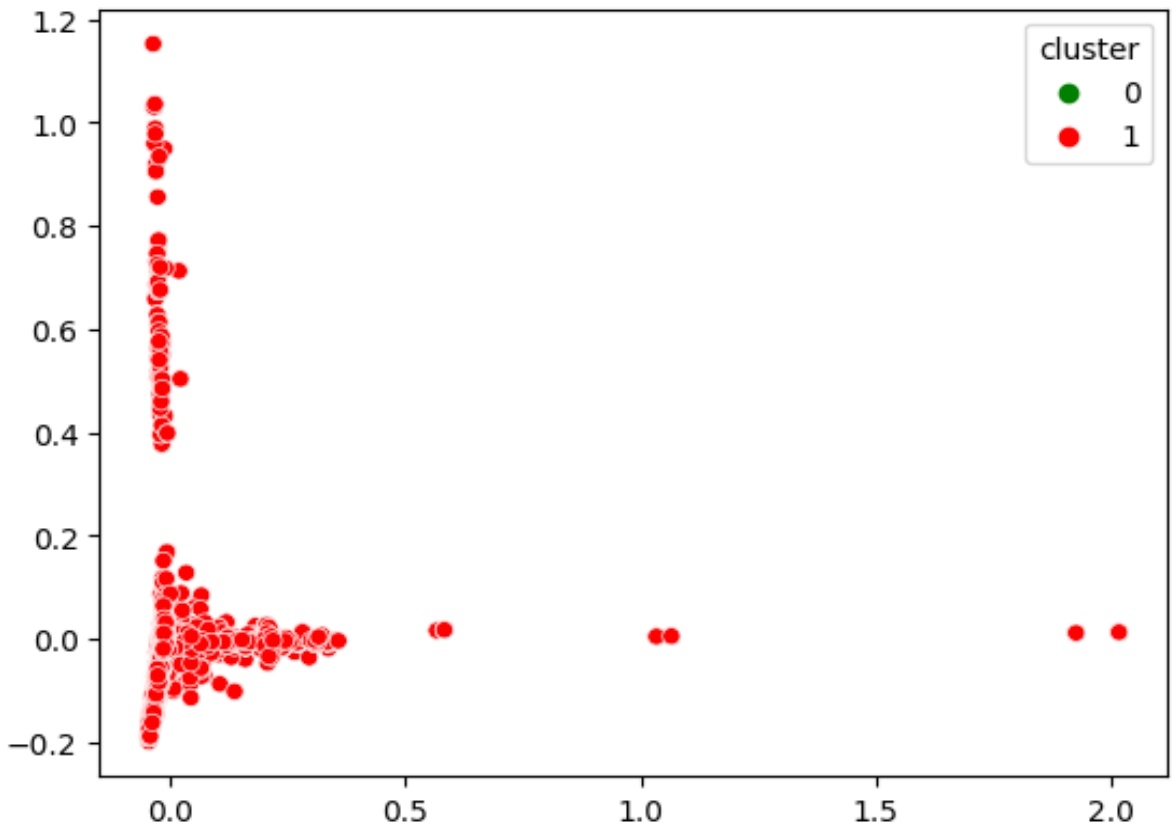
Python

```
Broj instanci koje pripadaju jednom klasteru: 922
Broj instanci koje pripadaju drugom klasteru: 46404
```

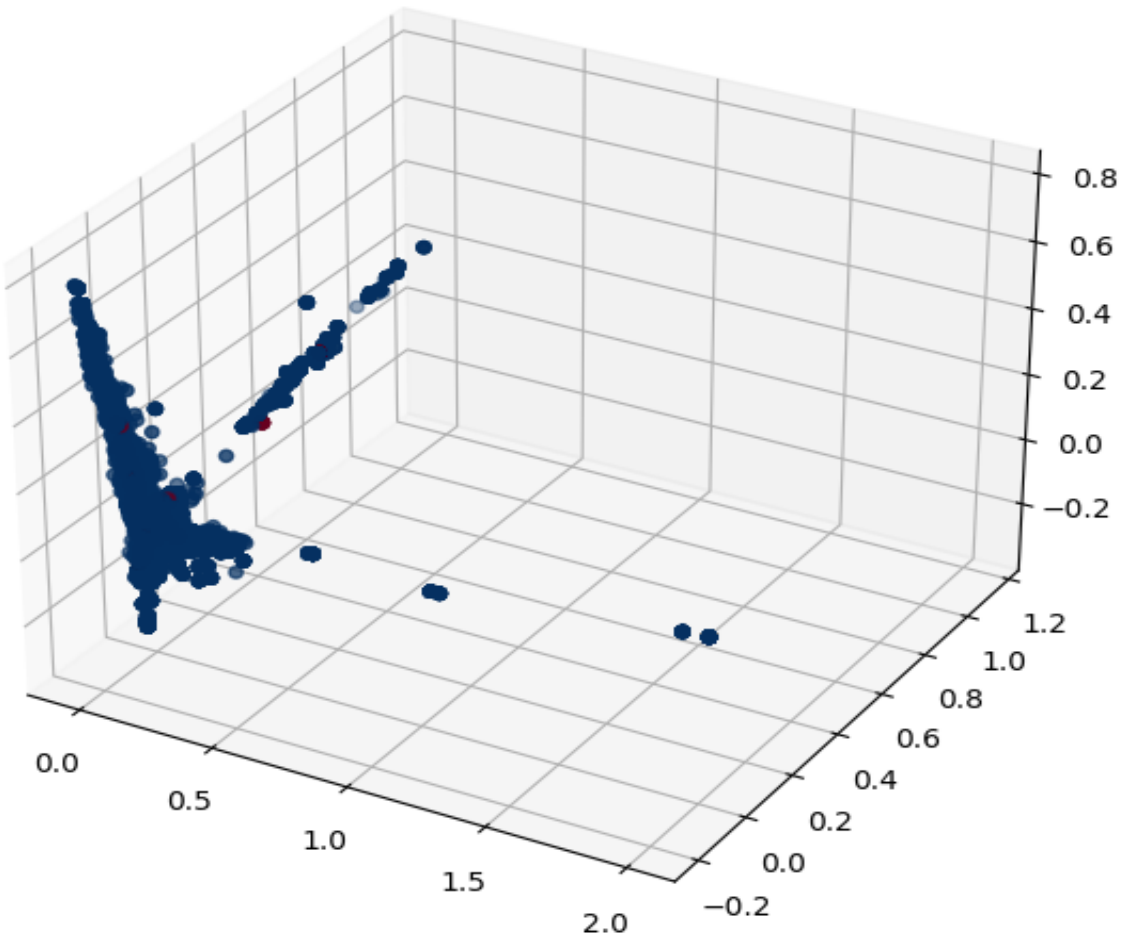
Kao sto mozemo videti broj instanci jednog klastera je 922 dok je broj instanci drugog klastera 46,404 sto bi predstavljao odnos od 2% / 98%, sto je priblizno inicijalnom odnosu klasa. Naravno ne mozemo dovesti zakljucak da su iste instance u datim klasterima kao I u pocetnom skupu podataka.

Da bismo vizualizovali nase podatke potrebno je da smanjimo dimenziju koristeci PCA. Nakon promene kategorije iz tekstualne u numericku sada nasi podaci imaju 3000 atributa u TFIDF matrici pa cemo to smanjiti na 2 i 3, tj. za prikaz u 2D odnosno 3D.

PCA (2D)

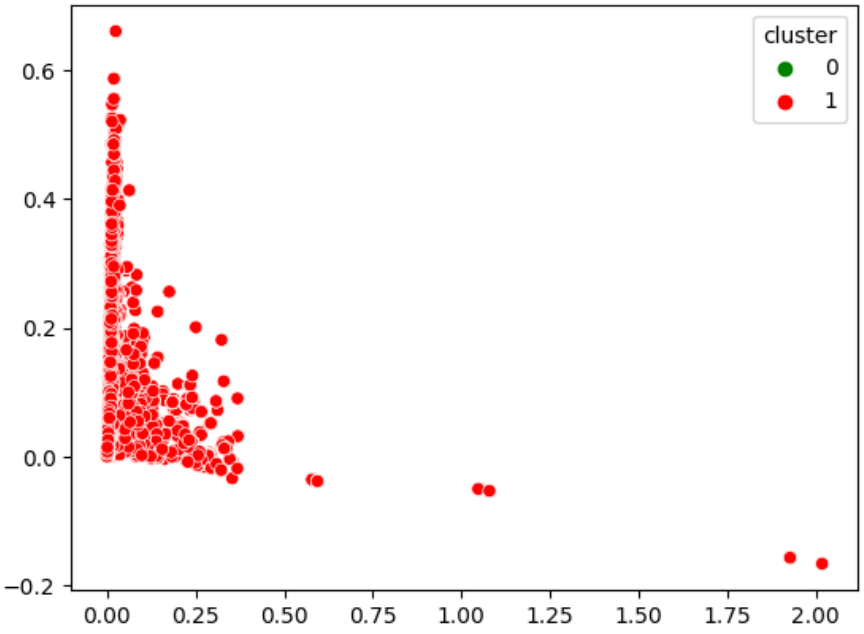


PCA (3D)



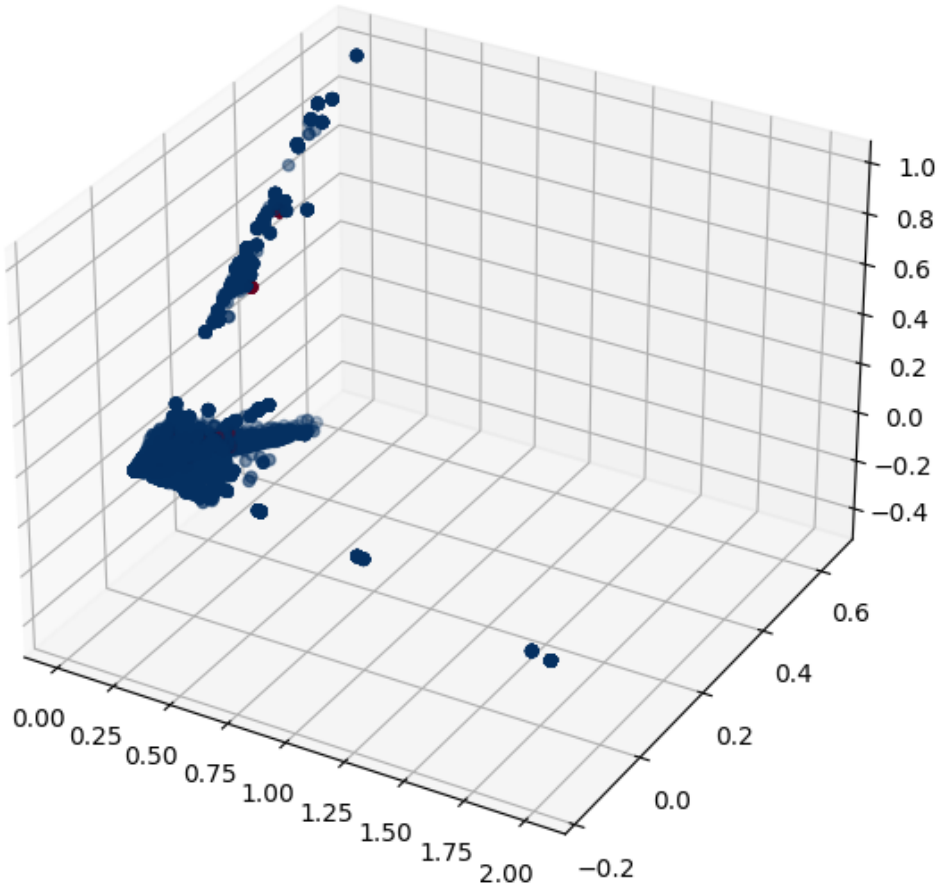
Treba imati na umu da se prilikom smanjenja dimenzionalnosti podataka gube odredjene informacije. Nesto bolju sliku mozemo dobiti koriscenjem TruncatedSVD.

TruncatedSVD (2D)

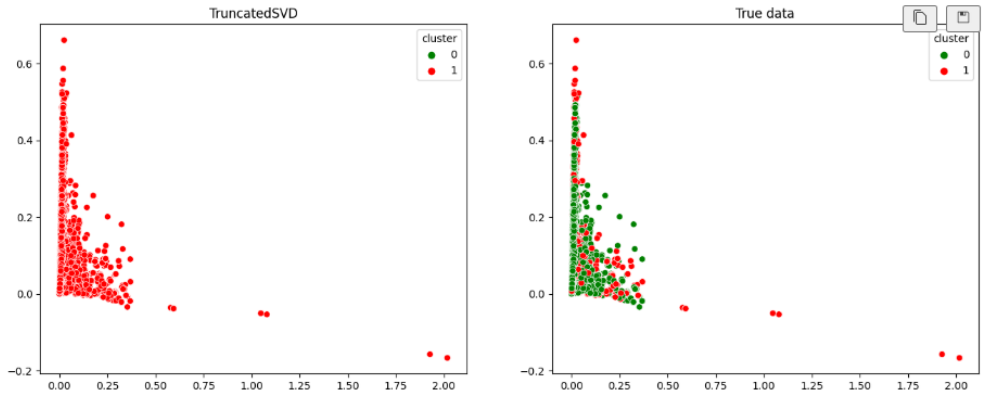


TruncatedSVD (3D)

Nakon

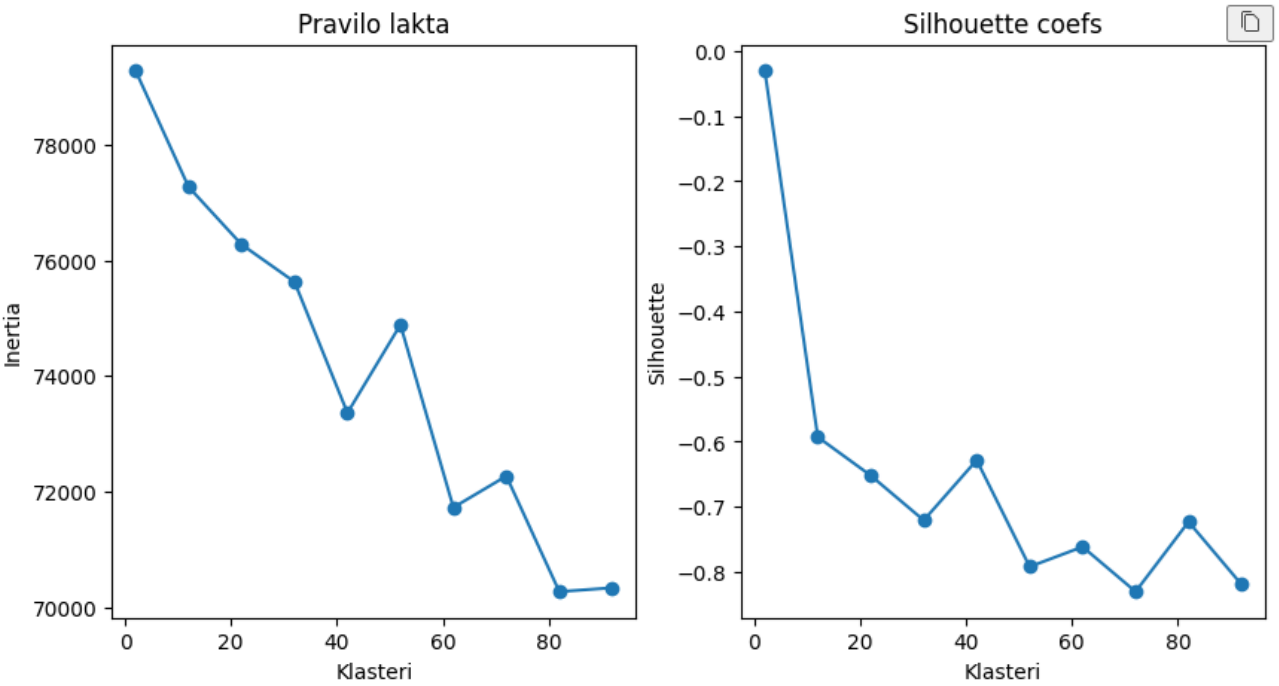


vizualizacije sledi ucitavanja pravih podataka i pravi prikaz kako izgledaju nasi podaci koriscenjem iste tehnike u poredjenju sa onim sto je KMeans dodelio.



Na slici vidimo da je veliki broj instanci koje su okarakterisane da pripadaju klaseru 0 u pravom skupu podataka okarakterisano kao da pripadaju klasteru 1 u KMeans algoritmu. Sledeci korak bi bio pronalazanje pravog broja klastera. Za broj klastera uzimamo vrednosti u opsegu od 2 do 100 sa korakom 10 (tj. 2, 12, 22, …). Prilikom odabira broja klastera koristimo napredniji k-means++ koji koristi nesto drugaciju raspodelu centroida. Prilikom odabira broj klastera posmatramo 2 vrednosti:

- inertia – suma kvadrata rastojanja
- silhouette – koeficijent koji meri bliskost instanci iz istog klastera odnosno razlicitost instanci iz razlicitih klastera



Prilikom odabira broja klastera pravilo lakta bi nam sa prikazanog grafa za broj klastera dao broj 62 jer tada SSE najbrze opada, medjutim, ako pogledamo drugi graf na kom se nalazi silhouette coef mozemo videti da se sa povecanjem broja klastera

opada koherencija unutar klastera kao i separacija izmedju klastera, pa se za broj optimalnih klastera bira broj 2.

Hijerarhijsko klasterovanje

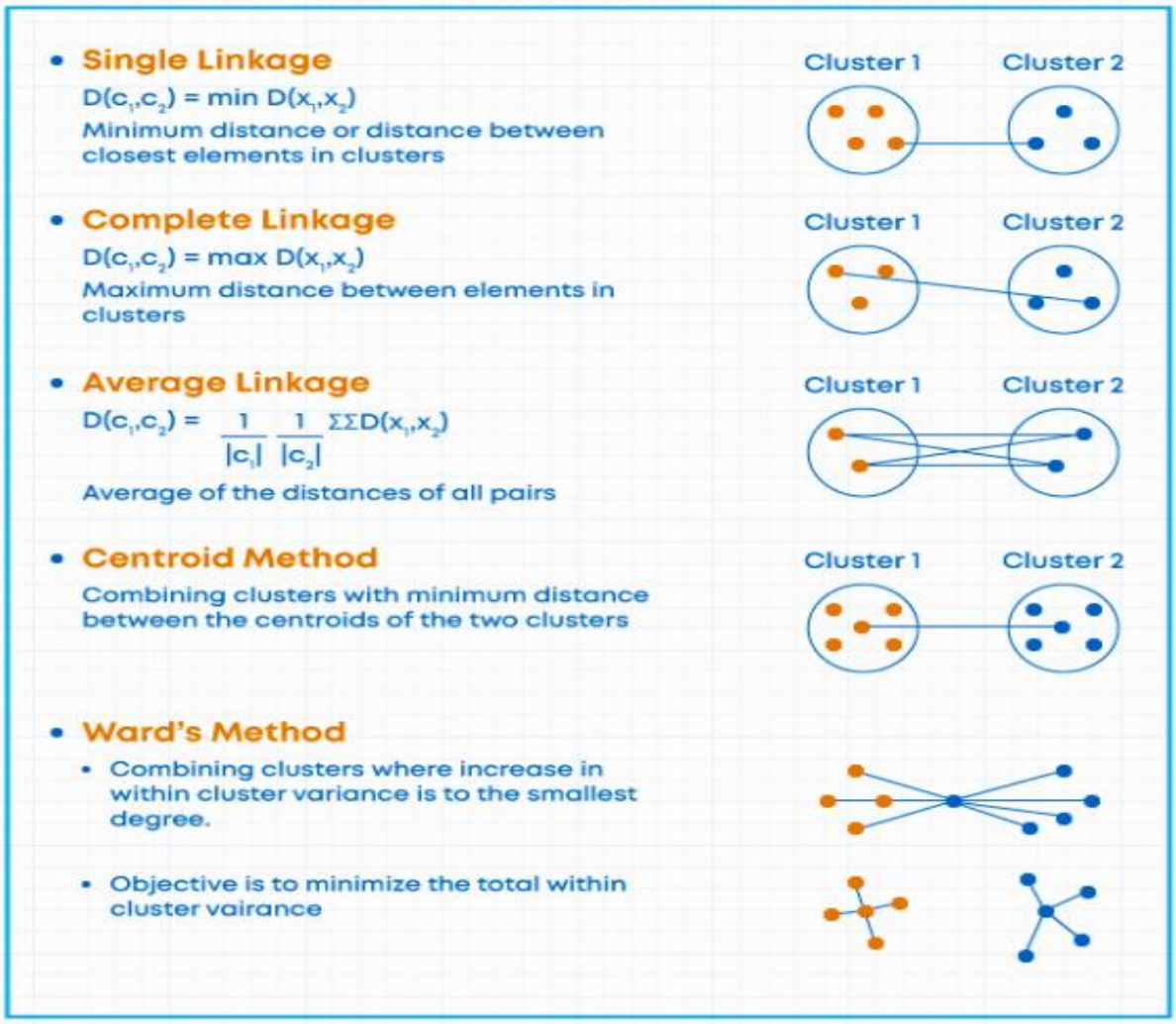
Pre samog rada potrebno je izvrsiti normalizaciju podataka, takodje je potrebno izvrsiti smanjenje dimenzije podataka (u nasem skupu postoji oko 50,000 instanci od kojih svaka ima 3000 atributa). Usled velikog skupa podataka i slozenosti samog algoritma ulazno podaci se smanjuju na broj argumenata od 1500, takodje iz istih razloga koristi se znatno manji skup podataka od obradjenog u pretprocesiranju (vise puta je dolazilo do maksimalnog punjenja RAM-a i pada sistema, kako na lokalnom racunaru tako i na Google Colab-u pa nije bilo moguce raditi sa vecim skupom).

Za smanjenje dimenzionalnosti je koriscen prethodno pomenuti TruncatedSVD koji je pogodniji za rad sa tfidf matricom

Za odabir najboljeg modela su se koristile metrike linkovanja, i to:

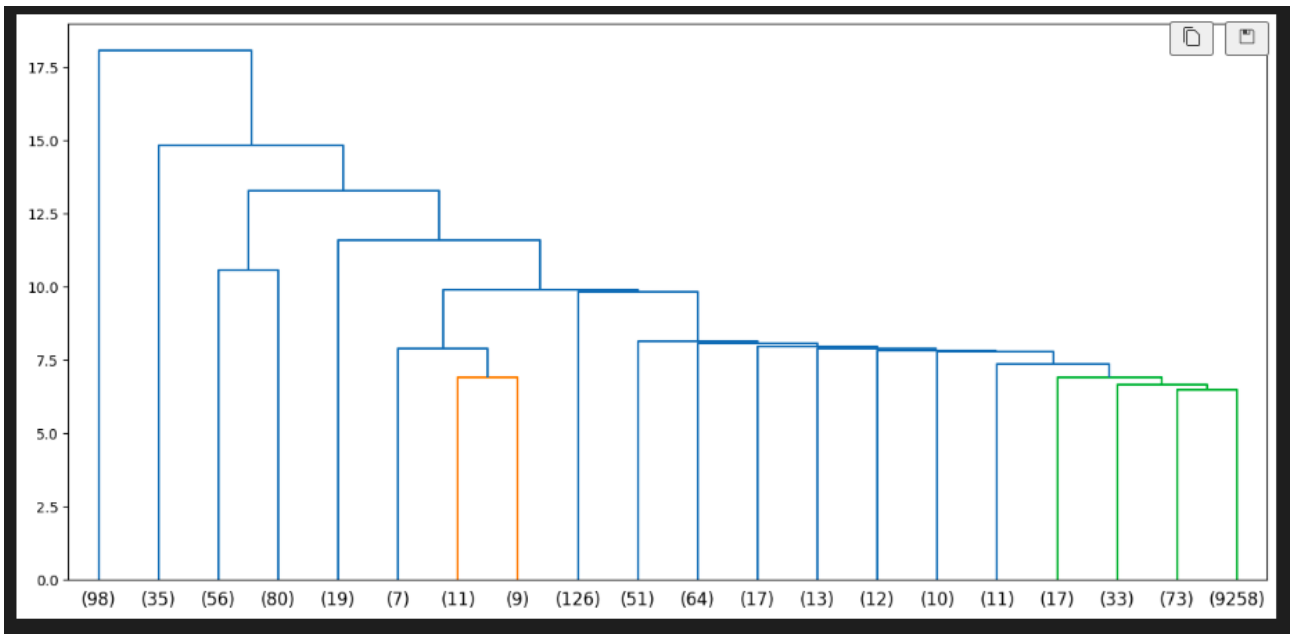
- ward – minimizacija varijanse klastera koji se spajaju
- average – prosek udaljenosti izmedju dva skupa
- complete – maksimalna udaljenost izmedju dva skupa
- single – minimalna udaljenost izmedju dva skupa

Metrike linkovanja



Kako nije bilo mogucnosti da se model AgglomerativeClustering istrenira na originalnim podacima usled pojavljivanja mnogobrojnih problema pre svega hardverskog karaktera, ne mozemo ocekivati da su rezultati znacajno bolji. Jedini model koji je davao neke priblizne rezultate je model koji kao metodu linkovanja koristi 'ward' linkovanje. Za dati model na dendogramu mozemo videti da algoritam vrsi prepoznavanje 3 klastera, i da klaster koji je obojen plavom bojom sadrzi vecinu instanci umanjenog skupa.

Dendogram za najbolji model (ward linkovanje)



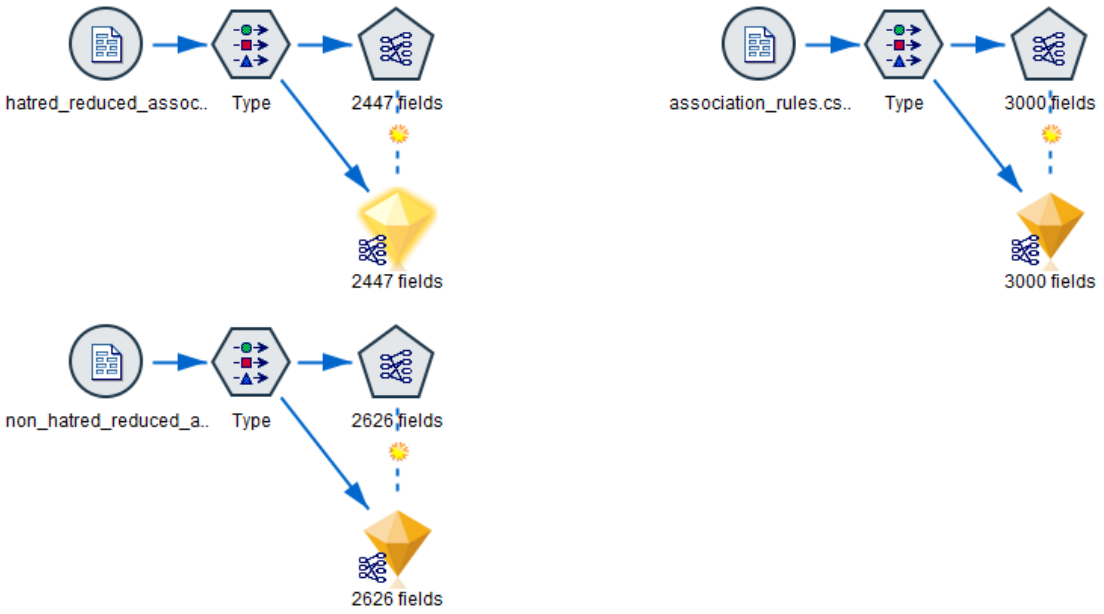
Pravila pridruzivanja

Apriori

Apriori algoritam se koristi za predvidjanje pojavljivanja nekog objekta na osnovu pojavljivanja ostalih objekata. Kao primer se moze navesti {you} → {are}, sto bi znacilo da ukoliko se reci ‘you’ nalazi u tweet-u onda postoji velika verovatnoca da ce se i rec ‘are’ naci u tweet-u. Svrha koriscenja apriori algoritma nad tekstualnim podacima je identifikacija nekih uobicajenih fraza, povezanost odredjenih reci u tekstu i sl.

Za koriscenje Apriori algoritma koriscen je program SPSS Modeler. Sto se tice skupa podataka koriscen je kompletan skup podataka obradjen u pretprocesiranju, a takodje su napravljena i 2 skupa koji su odvojeni na osnovu sadrzaja tweet-a, tj. hatred i non-hatred tweetovi.

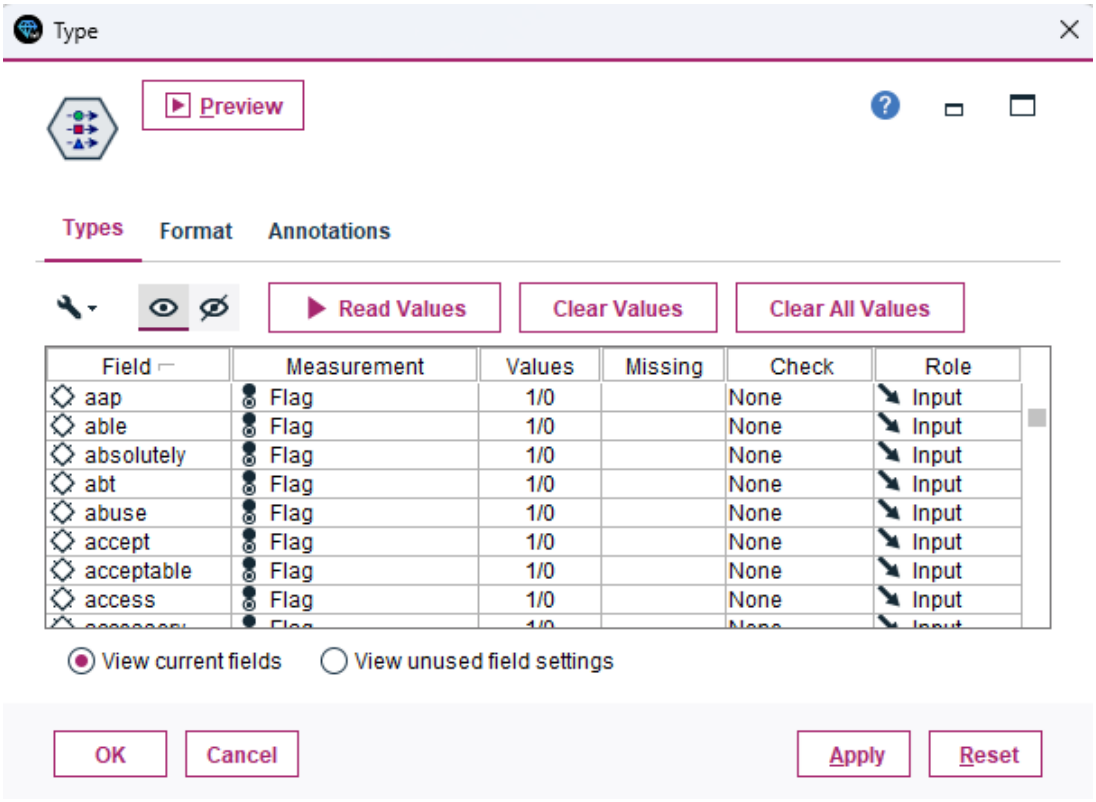
Kako je koriscena Tfidf matrica nad zajednickim skupom, moguce je da se u odvojenim skupovima ne nadju neke reci ni u jednoj instanci, pa je zbog potrebe Apriori algoritma potrebno izbaciti te instance i zato mozemo videti da je broj atributa manji od 3000.

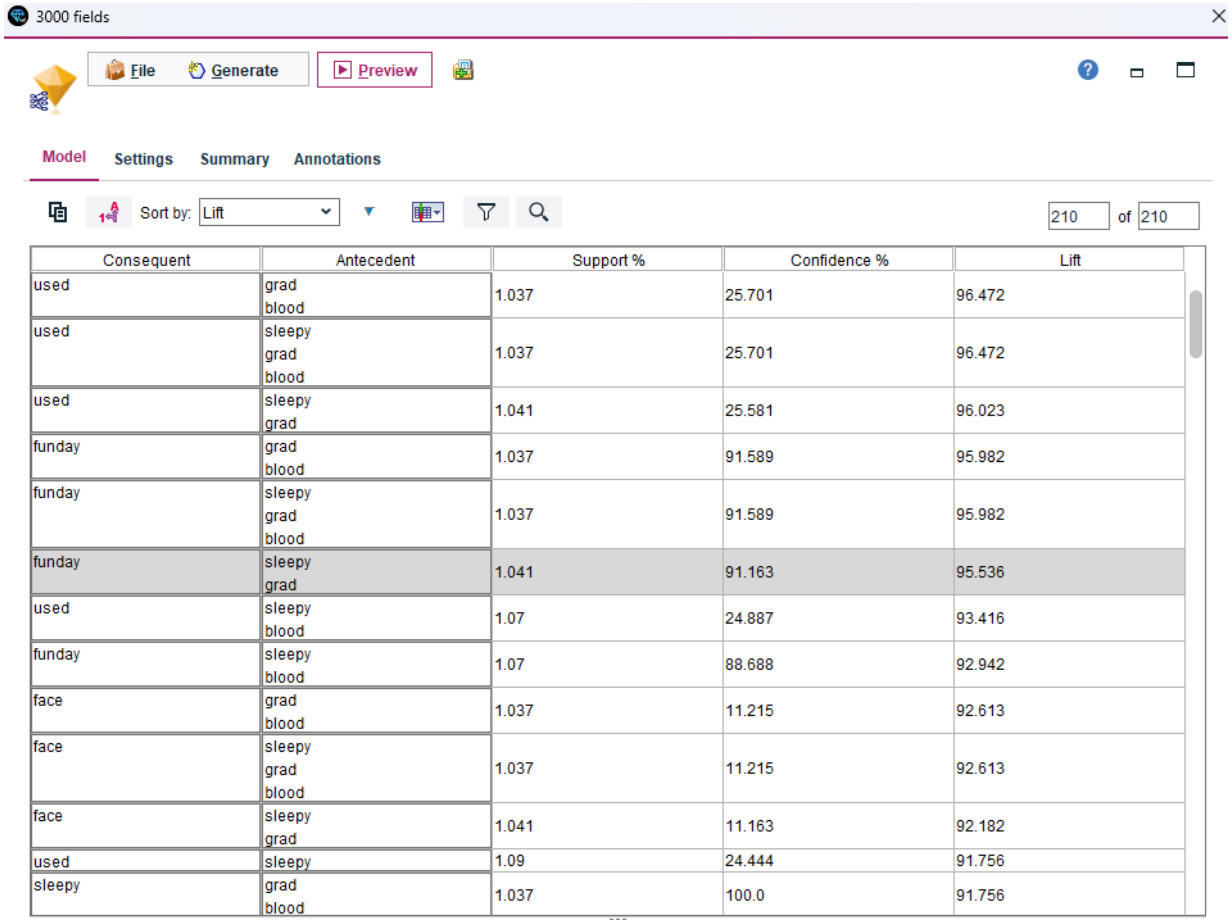


Na slici je dat prikaz obrade, pre svega ucitavanja podataka, odredjivanja tipa podataka gde je izvrsena koverzija iz neprekidnih u binarni tip (0 i 1, sto oznacava pojavljivanje odnosno nepojavljivanje reci).

Konfiguracija unutar Type cvora

Ceo skup podataka

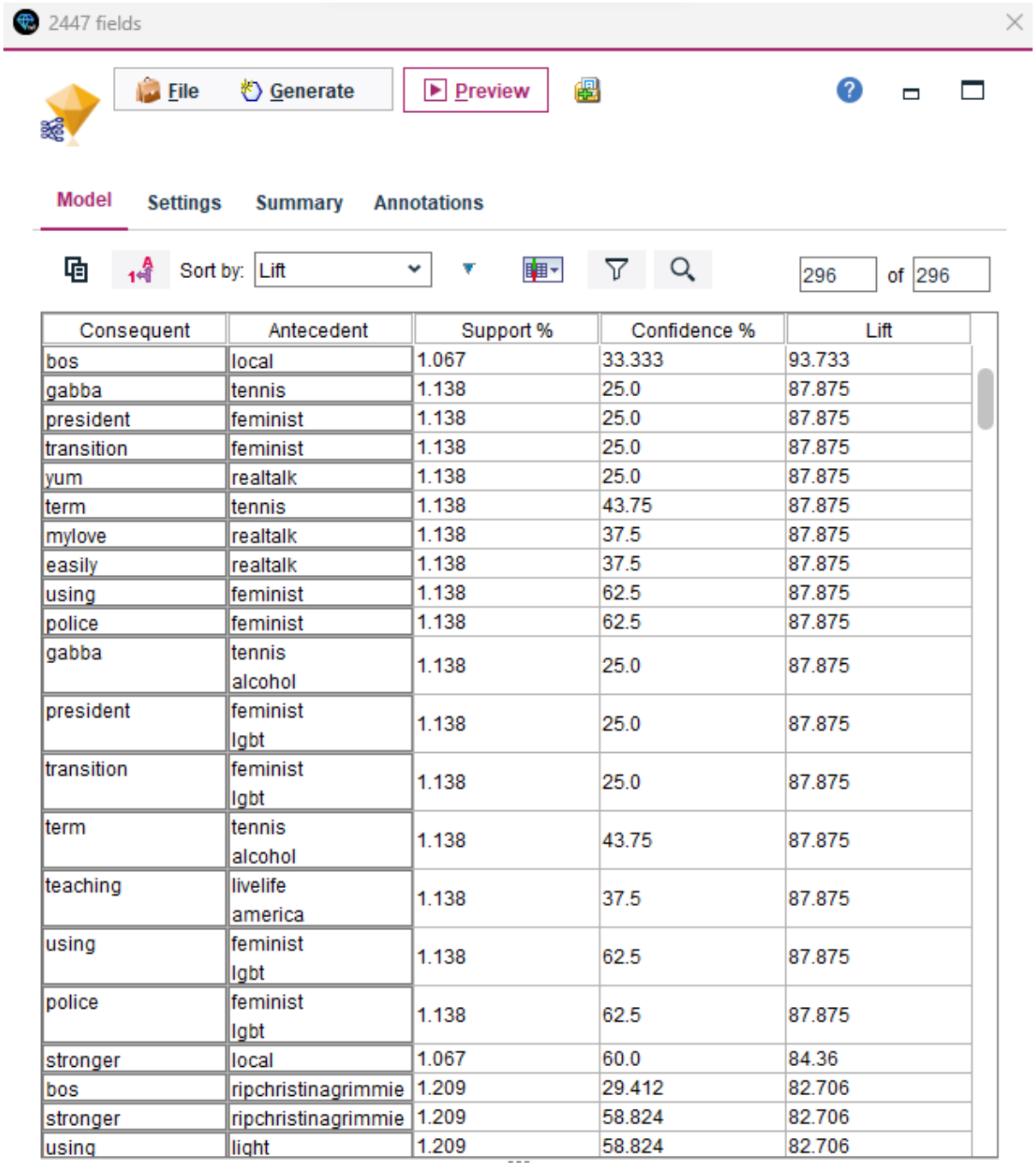




Consequent	Antecedent	Support %	Confidence %	Lift
used	grad blood	1.037	25.701	96.472
used	sleepy grad blood	1.037	25.701	96.472
used	sleepy grad	1.041	25.581	96.023
funday	grad blood	1.037	91.589	95.982
funday	sleepy grad blood	1.037	91.589	95.982
funday	sleepy grad	1.041	91.163	95.536
used	sleepy blood	1.07	24.887	93.416
funday	sleepy blood	1.07	88.688	92.942
face	grad blood	1.037	11.215	92.613
face	sleepy grad blood	1.037	11.215	92.613
face	sleepy grad	1.041	11.163	92.182
used	sleepy	1.09	24.444	91.756
sleepy	grad blood	1.037	100.0	91.756

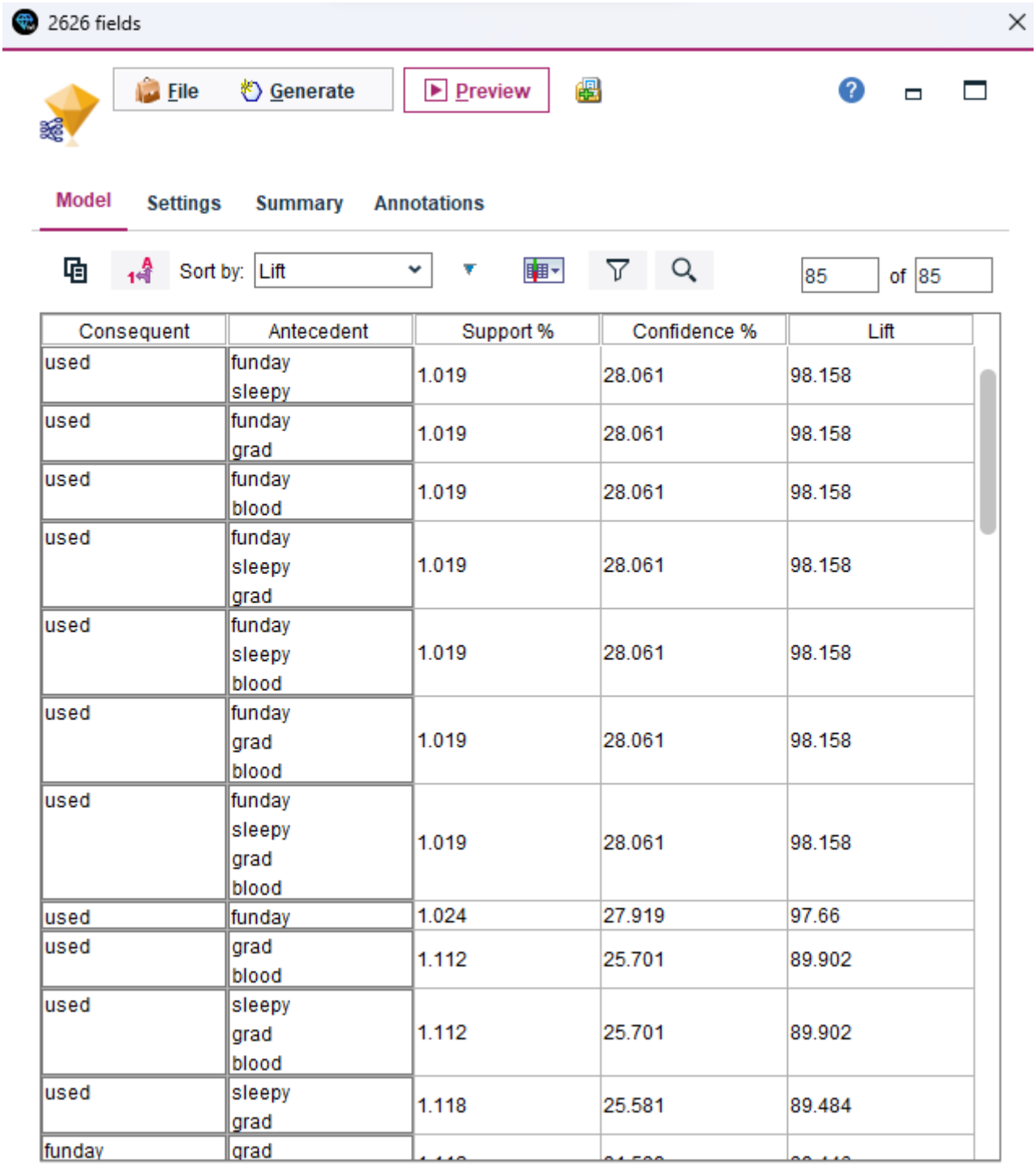
Dat je prikaz nekih pravila, od ukupno 210 koji su nadjeni uz konfiguraciju support 1%, confidence 10% (tj. minimalna podrska i minimalna pouzdanost) . Antecedent predstavlja glavu pravila dok Consequent predstavlja telo pravila. Tako npr. mozemo videti da u slucaju pojavljivanja reci {sleepy, grad} verovatno ce se i rec {funday} naci u tweet-u.

Hatred skup podataka



Consequent	Antecedent	Support %	Confidence %	Lift
bos	local	1.067	33.333	93.733
gabba	tennis	1.138	25.0	87.875
president	feminist	1.138	25.0	87.875
transition	feminist	1.138	25.0	87.875
yum	realtalk	1.138	25.0	87.875
term	tennis	1.138	43.75	87.875
mylove	realtalk	1.138	37.5	87.875
easily	realtalk	1.138	37.5	87.875
using	feminist	1.138	62.5	87.875
police	feminist	1.138	62.5	87.875
gabba	tennis alcohol	1.138	25.0	87.875
president	feminist lgbt	1.138	25.0	87.875
transition	feminist lgbt	1.138	25.0	87.875
term	tennis alcohol	1.138	43.75	87.875
teaching	livelife america	1.138	37.5	87.875
using	feminist lgbt	1.138	62.5	87.875
police	feminist lgbt	1.138	62.5	87.875
stronger	local	1.067	60.0	84.36
bos	ripchristinagrimmie	1.209	29.412	82.706
stronger	ripchristinagrimmie	1.209	58.824	82.706
using	light	1.209	58.824	82.706

Non-hatred skup podataka



Consequent	Antecedent	Support %	Confidence %	Lift
used	funday sleepy	1.019	28.061	98.158
used	funday grad	1.019	28.061	98.158
used	funday blood	1.019	28.061	98.158
used	funday sleepy grad	1.019	28.061	98.158
used	funday sleepy blood	1.019	28.061	98.158
used	funday grad blood	1.019	28.061	98.158
used	funday sleepy grad blood	1.019	28.061	98.158
used	funday	1.024	27.919	97.66
used	grad blood	1.112	25.701	89.902
used	sleepy grad blood	1.112	25.701	89.902
used	sleepy grad	1.118	25.581	89.484
funday	grad			

Zakljucak

Kako smo na pocetku imali jako nebalansirane klase u odnosu 93% - 7% u korist tweet-ova koji ne sadrze govor mrznje, nasi modeli nisu imali veoma velike ocene. U slucaju da se neki od algoritama koristi u pozadini da automatski nudi mogucnost odbacivanja tweet-ova koji bi bili prikazani sirem auditorijumu i potencijalne probleme koje bi mogao da izazove (kao sto su sirenje laznih informacija, sirenje mrznje koje moze da izazove fatalan ishod i sl.), skup podataka bi morao da sadrzi

priblizno podjednake klase i takodje bi morao da bude znatno veci s obzirom na broj korisnika drustvene mreze Twitter i kolicine informacija koje se kreiraju svake sekunde na istoj. Iako odredjeni modeli prikazuju jako veliku preciznost, to cesto ne bude tako dobra metrika za kvalitet nasih modela. U nasem slucaju bi znatno veci problem predstavljali tweet-ovi koji sadrze govor mrznje a koji bi od strane nasih modela bili okarakterisani kao tweet-ovi koji ne sadrze govor mrznje i potencijalno dosegli siroki auditorijum i izazvali potencijalne prethodno pomenute probleme.