Implementacija i evaluacija tehnika klasifikacije teksta zasnovanih na Bajesovoj teoremi

Andrija Đurišić

Oktobar 2014

Rezime

Količina dokumenata u digitalnom obliku se drastičnom brzinom povećava a sa tim i potreba za boljom organizacijom tih podataka. Ovaj rad predstavlja osnovu i "dobro mesto za početak" rešavanja problema organizacije digitalnih podataka. U radu su opisane odabrane i jednostavne ali važne metode klasifikacije koje su u praksi pokazale izvanredne rezultate a akcenat je na naivnoj Bajesovoj metodi kojom je rešen problem klasifikacije novinskih članaka.

Sadržaj

Poglavlje 1

Uvod

Klasifikacija ili *kategorizacija* teksta predstavlja zadatak svrstavanja teksta u jednu ili više predefinisanih kategorija. Na primer, novinski članci su najčešće organizovani po rubirkama kojima pripadaju, naučni radovi su klasifikovani po oblastima i podoblastima koje obrađuju, medicinski kartoni pacijenata su često indeksirani iz više aspekata – istorijat bolesti, tipovi hirurških poduhvata, brojevi osiguranja itd.

U navedenim primerima svaka instanca (novinski članak, naučni rad, medicinski karton itd.) se može predstaviti nekim izabranim skupom njenih atributa. Svakoj instanci se može dodeliti oznaka klase – *ciljna vrednost*, kojoj instanca pripada. Problem klasifikacije se sastoji u određivanju ovih vrednosti na osnovu atributa instance. Formalnije, problem klasifikacije se može posmatrati kao aproksimacija funkcije koja svakoj instanci dodeljuje oznaku klase kojoj ta instanca pripada [2].

Često je korisno razlikovati probleme klasifikacije teksta po broju elemenata skupa ciljnih vrednosti. Ako skup ciljnih vrednosti sadrži tačno dve klase problem klasifikacije teksta je *binarni*. Primer binarne klasifikacije teksta je filtriranje elektronske pošte u dve klase – "željena" i "neželjena" pošta (eng. spam). Još neki primeri binarne klasifikacija teksta su: klasifikacija po polu autora, pri čemu se na osnovu zadate instance određuje da li je autor muškog ili ženskog pola, zatim klasifikacija filmskih kritika na pozitivne i negativne kritike itd. Slično, problem klasifikacije teksta čiji skup ciljnih vrednosti sadrži više od dva elementa kažemo da je *višeklasni* (eng. multi-class). Primer višeklasnog problema klasifikacije teksta je problem prepoznavanja jezika na kom je tekst napisan. U mnogo slučajeva tekst može spadati u više klasa u isto vreme. Na primer, naučni rad može u isto vreme biti svrstan u oblast pretraga informacija (eng. information retirval), mašinsko učenje (eng. machine learning) i u još neku njihovu podoblast. Ovaj tip klasifikacije teksta naziva se *višeznačna* (eng. multi-label) klasifikacija [3].

Jedna od prvih primena klasifikacije teksta bila je vezana za određivanje autora određenog dokumenta (eng. authorship attribution). Godine 1787. objavljena je serija od 85 anonimnih eseja, "The Federalist papers" [4], čija je uloga bila podsticanje države New York na ratifikaciju ustava Sjedninjenih Američkih Država. Kasnije se saznalo da su tri autora napisala svih 85 pisama ali se za 12 nije znalo tačno ko je napisao koje. 1963. Mosteller i Wallace su uspeli da identifikuju ko je napisao određeno pismo služeći se Bajesovim metodama. Upravo ove metode su podstakle stvaranje metode koja je tema ovog rada.

Poglavlje 2

Klasifikacija teksta

2.1 Matematička formulacija problema

Problem klasifikacije teksta možemo definisati na sledeći način [1]:

Problem. Neka je x instanca (tekst) i $V = \{v_1, v_2, ..., v_k\}$ diskretan skup klasa. Zadatak je što bolje opisati instancu x atributima $a_1, a_2, ..., a_n$ a zatim pronaći jednu ili više vrednosti skupa V kojoj instanca x pripada.

Formalnije, klasifikacija teksta je problem aproksimacije funkcije $f(x)$ gde je svaka instanca teksta x predstavljena kao skup atributa $a_1, a_2, ..., a_n$, pri čemu $f(x)$ uzima vrednosti iz diskretnog skupa ciljnih vrednosti V. Zadatak je predvideti vrednost funkcije f nove instance x [2].

Jedna od oblasti mašinskog učenja koja se između ostalog bavi i automatizacijom klasifikacije teksta zove se *nadgledano mašinsko učenje* (eng. supervised machine learning). Ideja je da se na osnovu velikog broja unapred klasifikovanih instanci – trening skupa podataka napravi model koji će novoj instanci dodeliti odgovarajuću klasu. Problem u tom slučaju izgleda ovako:

Problem. Neka je x instanca, $V = \{v_1, v_2, ..., v_k\}$ diskretan skup klasa i T skup ručno klasifikovanih instanci – trening skup. Zadatak je opisati instacu x atributima $a_1, a_2, ..., a_n$ i izgraditi model koji aproksimira vrednost funkcije f na osnovu trening skupa.

Postoji veliki broj metoda nadgledanog mašinskog učenja kojima se ovaj problem rešava. Neke od njih su metode zasnovane na instancama (eng. instance based classification), učenje stabla odlučivanja (eng. decision tree induction), neuralne mreže (eng. neural networks), metoda potpornih vektora (eng. support vector machines) i metode Bajesovske klasifikacije zasnovane na verovatnoći (eng. Bayesian classification) [2].

Kada trening skup nije unapred određen problem klasifikacije teksta rešavamo metodama *nenadgledanog mašinskog učenja* (eng. unsupervised learning). Primer nenadgledanog učenja je takozvano *klasterovanje* – uočavanje grupa na neki način sličnih objekata kada nemamo prethodno znanje o tome koliko grupa postoji ili koje su njihove karakteristike [2]. Pored klasterovanja najpoznatije metode nenadgledanog učenja su *skriveni Markovljevi modeli*, *EM algoritam*, *algoritmi analize komponenata* itd.

Mnogi istraživači oblasti mašinskog učenja zaključili su da neoznačeni podaci korišćeni u konjukciji sa malom količinom označenih podataka mogu da proizvedu značajna poboljšanja u procesu učenja. Oblast koja se bavi ovim problemima naziva se *polunadgledano mašinsko učenje* (eng. semi-supervised learning).

2.2 Mere kvaliteta i tehnike evaluacije klasifikacije

U ovom poglavlju ćemo objasniti kako procenjujemo uspešnost klasifikatora i formalno definisati mere evaluacije – *tačnost*, *preciznost*, *odziv* i *f-meru*. Ovi koncepti se mogu primenti ne samo na probleme klasifikacije teksta već i na puno drugih problema.

Za evaluaciju klasifikatora se mora koristiti nezavisan skup podataka (podaci koji nisu korišćeni za učenje). Kada se klasifikator dobro ponaša na skupu trening podataka a loše na skupu podataka za test radi se o *problemu previše prilagođenog modela* (eng. overfitting problem). Za svaki trening skup može se naći model koji ga savršeno opisuje. Međutim, prilagođavajući se trening podacima do krajnosti, gubi se svaka moć generalizacije [2].

Proces evaluacije se sastoji u poređenju unapred poznate klase sa onom koju je predložio klasifikator. Time se dobijaju ispravno i neispravno klasifikovani podaci [5].

Početna tačka za razumevanje evaluacije klasifikacije je *matrica konfuzije* (eng. cofusion matrix). Ovaj pojam ćemo objasniti kroz primer binarne klasifikacije – da li je imejl spem ili ne? Za svaku instancu koju želimo da klasifikujemo razlikujemo četiri stanja i mogući ishodi klasifikacije su sledeći:

1. *TP* – Vrednost "stvarno pozitivno" (eng. true positive) predstavlja broj dokumenata koji su zaista spemovi a koje je klasifikator prepoznao kao spemove.

2. *FP* – Vrednost "lažno pozitivno" (eng. false positive) predstavlja broj dokumenata koji nisu spemovi a koje je klasifikator svrstao u grupu spemova.

3. *TN* – Vrednost "stvarno negativno" (eng. true negative) predstavlja broj dokumenata koji nisu spemovi i klasifikator ih je svrstao u grupu ne-spemova.

4. *FN* – Vrednost "lažno negativno" (eng. false negative) predstavlja broj dokumenata koji su spemovi a koji su svrstani u grupu ne-spemova.

Matrica konfuzije prikazana je tabelom 2.1.

	spam	ne-spam
spam	TP	FN
ne-spam	FP	TN

Tabela 2.1: Matrica konfuzije za klasifikator imejlova u kategorije spem i ne-spem, pri čemu vrste matrice predstavljaju stvarne vrednosti dokumenta dok kolone predstavljaju vrednosti koje je dao klasifikator.

Tačnost (eng. accuracy) klasifikatora je mera koja nam daje procenat uspešno klasifikovanih dokumenata. Tačnost definišemo na sledeći način:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

Za mnoge primere klasifikacije tačnost je korisna mera, međutim postoje slučajevi kada nam tačnost ne može dati informacije koje tražimo. To su uglavnom scenariji u kojima je jedna klasa značajno manja od druge. Tada je moguće dobiti visoku tačnost svrstavnjem svih instanci u veću grupu.

Preciznost (eng. precission) je mera koja nam daje informaciju o udelu stvarno pozitivnih instanci. Od instanci koje su selektovane kao pozitivne koji procenat je zaista pozitivan?

$$Precission = \frac{TP}{TP + FP} \tag{2.2}$$

Odziv (eng. recall) je mera suprotna preciznosti. Od instanci koje su stvarno pozitivne koji procenat je selektovan kao pozitivan?

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

Kombinovanjem ove dve mere dobijena je *f1* mera:

$$f1 = \frac{2PrecissionRecall}{Precission + Recall} \tag{2.4}$$

U slučajevima kada problem sadrži više od jedne klase koriste se mere koje predstavljaju usrednjavanje mera po svim klasama. To se može uraditi na dva načina [5]:

1. *Makro-prosek* – svakoj klasi se pridaje isti značaj. Naime, najpre se izračunava vrednost mere za svaku od klasa pojedinačno nakon čega se vrši usrednjavanje tih vrednosti po broju klasa.

2. *Mikro-prosek* – favorizuju se klase koje sadrže veći broj dokumenata. Najpre se izračunavaju vrednosti TP, FN , FP i TN za svaku klasu pojedinačno. Zatim se izračunaju vrednosti TP', TN', FP' i FN' i to TP' kao suma svih TP, TN' kao suma svih TN itd. Na kraju se vrednosti mera izračunavaju za dobijene sumirane vrednosti TP', TN', FP' i FN'.

Takođe, često korišćena tehinka za evaluaciju klasifikatora je *n-unakrsnih validacija* (eng. n-cross validation). Naime, najpre se skup podataka podeli na n delova, a zatim se za svaki deo preostalih $n-1$ delova iskoristi kao skup za učenje a sam taj izabrani deo kao skup za testiranje. Rezultat je prosek svih dobijenih vrednosti.

Poglavlje 3

Najznačajniji algoritmi nadgledane klasifikacije

3.1 Stabla odlučivanja

Učenje stabala odlučivanja predstavlja metod aproksimacije diskretne funkcije u kojem je kreirana funkcija predstavljena n-arnim stablom. Ova stabla se mogu predstaviti i kao skup if-then pravila kojima je razumevanje stabla odlučivanja pojednostavljeno, pa je upravo to jedan od glavnih uzroka popularnosti ove metode [1].

3.1.1 Reprezentacija stabla odlučivanja

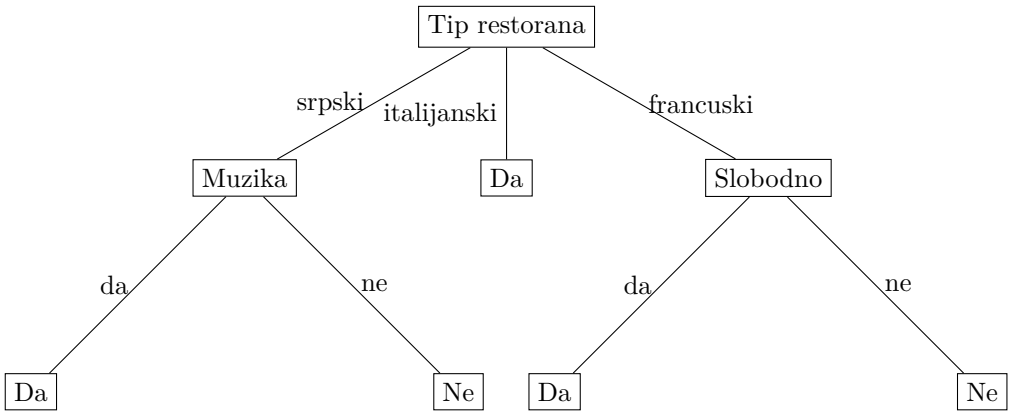
Stablo odlučivanja klasifikuje instancu polazeći od korena stabla do listova koji predstavljaju rezultat klasifikacije. Svaki čvor sadrži atribut instance a svaka njegova grana odgovara vrednosti tog atributa [1].

Najjednostavniji način za razumevanje stabla odlučivanja je kroz primer.

Problem. Nalazite se na ljubavnom sastanku i pokušavate da odlučite u koji ćete restoran ići, pri čemu su vam sledeće stvari od značaja:

- U vašoj blizini nalaze se italijanski i francuski restoran i restoran domaće kuhinje.
- U restoranu Francuske kuhinje je jako lepo ali je i jako teško pronaći mesto.
- U restoranu domaće kuhinje je dobar provod ali samo ukoliko ima žive muzike.
- Italijanski restoran je uvek dobar izbor.

Zadatak je odrediti da li ući u restoran ili ne na osnovu ulaznih podataka.



Slika 3.1: Stablo odlučivanja koje će vas lišiti muke.

Slika 3.1 ilustruje "naučeno" stablo odlučivanja koje rešava problem binarne klasifikacije da li ući u restoran ili ne. Na primer, instanca

$$Tip\ restorana = francuski,\ Ima\ slobodnih\ mesta = Ne,\ Muzika = Da$$

će biti klasifikovana niz kranje desnu granu i rezultat će biti NE.

Uopšteno, stabla odlučivanja predstavljaju disjunkciju konjukcije vrednosti atributa date instance. Na primer, stablo odluke prikazano na slici 3.1 se može prikazati sledećom formulom:

$$\begin{gathered} Tip\ restorana = srpski \wedge Muzika = Da \\ \vee \\ Tip\ restorana = italijanski \\ \vee \\ Tip\ restorana = francuski \wedge Ima\ slobodnih\ mesta = Da \end{gathered}$$

3.1.2 Osnovni algoritam za učenje stabala odlučivanja

ID3 algoritam je osnovni algoritam za konstrukciju stabla odlučivanja i konstruiše stablo od vrha ka dnu (eng. top-down) [1]. Proces započinje pitanjem – "koji atribut treba izabrati kao koren stabla"? Kako bi dao odgovor na ovo pitanje ID3 algoritam procenjuje svaki atribut određenim statističkim metodama, prolazeći kroz trening skup, kako bi utvrdio koliko je atribut važan za proces klasifikacije. Kada je najpogodniji atribut selektovan on se postavlja kao koren stabla. Zatim, za svaku vrednost atributa se kreiraju čvorovi sledbenici a trening skup se sortira prema tim vrednostima. Ceo proces se ponavlja koristeći sortirane delove trening skupa po vrednostima atributa iz kog su kreirani čvorvi sledbenici.

Algoritam: ID3

Neka je R prazan čvor
Beskonačna petlja:
- Biramo najpogodniji atribut A
- Čvoru R dodeljujemo atribut A
- Za svaku vrednost atributa A kreiramo čvorove sledbenike
- Za svaku vrednost atributa A izdvojamo podskupove trening skupa sortirane prema toj vrednosti
- Ako je trening skup savršeno podeljen po kategorijama izlazimo iz ciklusa, inače iteriramo kroz novokreirane čvorove sledbenike, dodeljujemo ih redom promenljivoj R i ponavljamo proces za svaki od njih

Tabela 3.1: ID3 algoritam

3.1.3 Primer klasifikacije teksta ID3 algoritmom

Sada ćemo pokazati rad ID3 algoritma, korak po korak, na problemu klasifikacije teksta – "da li je tekst o mašinskom učenju?". Naravno, za ovakav eksperiment potreban nam je trening skup tekstova čija je tema mašinsko učenje i skup tekstova čija tema nije mašinsko učenje.

Posmatrajmo sledeći tekst:

"***Mašinsko učenje*** *je oblast* ***veštačke inteligencije*** *koja se može definisati na različite načine. Jedna definicija bi mogla biti da je to* ***disciplina*** *koja se bavi izgradnjom prilagodljivih* ***računarskih sisitema*** *koji su sposobni da poboljšavaju svoje performanse koristeći informacije iz* ***iskustva****.* ***Mašinsko učenje*** *bi se moglo definisati i kao* ***disciplina*** *koja se bavi proučavanjem* ***generalizacije*** *i konstrukcijom i analizom* ***algoritama*** *koji generalizuju.*"

Slika 3.2: Primer tekstualne instance čija je tema mašinsko učenje.

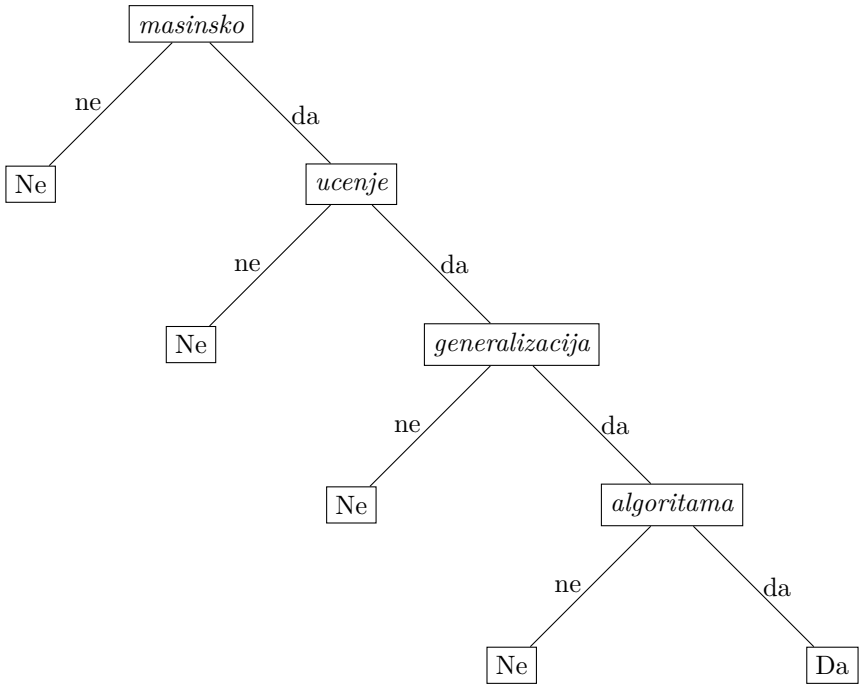
Najpre, potrebno je opisati instance trening skupa pogodnim atributima. U mnogim algoritmima mašinskog učenja tekstualni dokument je kompresovan u formu *vreća reči* (eng. bag of words). U poglavlju 3.3.2 je ovaj pojam detaljno objašnjen ali za sada je dovoljno reći da vreća reči predstavlja skup reči koje opisuju zadati tekst, pri čemu se gubi informacija o rasporedu reči u tekstu. Na primer, članak 3.2 možemo opisati skupom podebljanih reči u tekstu.

Sada, možemo primeniti ID3 algoritam pri čemu je kriterijum konstrukcije stabla odgovor na pitanje – "da li data reč pripada vreći reči koja opisuje dati tekst?".

U prvom koraku biramo "najpogodniji" atribut što je intuitivno reč "mašinsko". Zatim, iteriramo kroz trening skup dokumenata i sortiramo ih prema kriterijumu da li sadrže reč "mašinsko" ili ne. Ukoliko članak ne sadrži datu reč, radi lakšeg razumevanja samog algoritama, pretpostavićemo da tada članak

i nije o mašinskom učenju. Algoritam nastavljamo birajući sledeći atribut po važnosti.

Nakon nekoliko koraka dobijeno je sledeće stablo odlučivanja – slika 3.3.



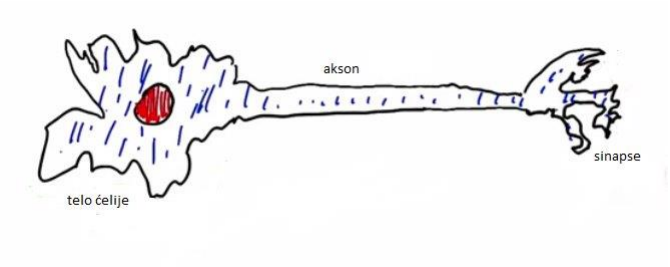
Slika 3.3: Pojednostavljeno stablo odlučivanja za problem da li je tema zadatog teksta mašinsko učenje.

3.2 Veštačke neuronske mreže

Veštačke neuronske mreže (eng. artificial neural network, ANN) predstavljaju moćan alat za rešavanje problema klasifikacije. Ovaj metod je pre svega postigao izvanredne rezultate na mnogim praktičnim problemima kao što su prepoznavanje rukom pisanih karaktera [6], prepoznavanje govora [7] i prepoznavanje lica [8]. Trenutno, jako popularna oblast istrživanja je primena veštačkih neuronskih mreža u medicini. Naime, istraživanja su uglavnom bazirana na modelovanju delova ljudskog tela i prepoznavanju raznih bolesti na osnovu slika skenera (na primer, kardiogram, slike dobijene ultravukom itd.). Trenutno postoje alati za dijagnostiku hepatitisa, detektovanje bolesti srca a još neki primeri primene veštačkih neuronskih mreža su analiza tekstura, prepoznavanje trodimenzionalnih objekata itd.

3.2.1 Biološke neuronske mreže

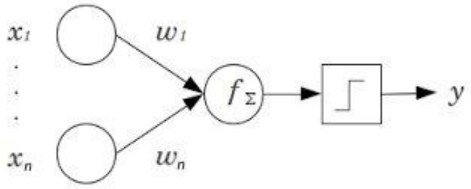
Ljudski mozak se sastoji od oko 10^{11} neurona, slika 3.4, od kojih je svaki neuron povezan sa oko 10^4 drugih [1]. Glavni deo neurona predstavlja telo neurona. Neuroni međusobno komuniciraju pomoću impulsa koji se prenosi iz jednog u drugi neuron. Aksoni su delovi neurona koji su zaduženi za prenos impulsa. Jedan neuron je povezan sa aksonom drugih kroz dendrit koji predstavlja mali produžetak tela neurona. Mesto spoja dentrita i aksona naziva se sinapsa. Neurolozi su otkrili da ljudski mozak uči tako što menja jačinu sinaptičke veze između neurona zbog ponavljajućeg stimulisanja jednim istim impulsom [5].



Slika 3.4: Neuron.

3.2.2 Perceptron

Veštačke neuronske mreže su nastale kao pokušaj da se simulira pravi biološki neuronski sistem, pa je predstavljen model koji se naziva *perceptron*, slika 3.5.



Slika 3.5: Perceptron.

Na ovoj šemi, instanca je opisana vektorom vrednosti $x = [x_1...x_n]$ a vektor $w = [w_1...w_n]$ zovemo *težinskim vektorom*. Perceptron kao ulaz prihvata vektor realnih vrednosti x. Kada aktiviramo perceptron svaka ulazna vrednost je pomnožena odogovarajućim težinskim faktorom i sumirana. Formalnije, prilikom aktivacije perceptrona računa se linearna kombinacija ulaznih podataka. Ova linearna kombinacija naziva se *funkcija aktivacije*. Izlaz funkcije aktivacije je vrednost koja je prosleđena *funkciji praga*. Ova funkcija vraća 1 ako je dobijena vrednost veća od zadatog praga θ, a -1 inače. Formalnije, perceptron model se može predstaviti formulom

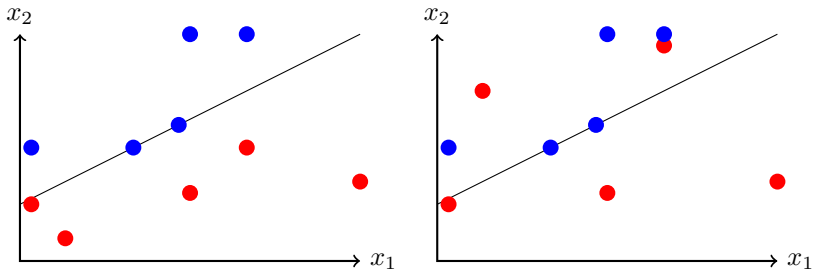
$$y(x_1, ..., x_n, \theta) = \text{sgn}\,(\sum_{k=1}^{n} w_k x_k - \theta) \tag{3.1}$$

pri čemu je x_k k-ta ulazna vrednost, w_k težinski faktor x_k, θ prag, a funkcija praga sgn se definiše na sledeći način:

$$\text{sgn}\,(x) = \begin{cases} 1, & \text{ako je, } x > 0 \\ -1, & \text{inače} \end{cases}$$

Dakle, kada je perceptron aktiviran njegov odgovor ukazuje da li instanca x pripada prvoj klasi (1) ili drugoj klasi (-1).

Takođe, perceptron predstavlja hiperravan u n-dimenzionom prostoru i vraća 1 za instance koje se nalaze sa jedne a -1 za instance koje se nalaze sa druge strane ravni – slika 3.6a. Naravno, postoje slučajevi kada je nemoguće skup instanci razdvojiti sa hiperravni – slika 3.6b. Oni skupovi za koje je to moguće nazivaju se *linearno separabilni* skupovi.



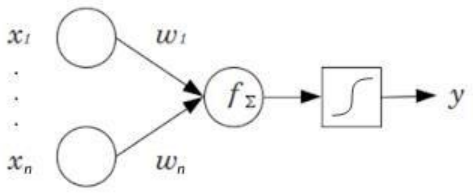
(a) Linearno separabilan skup tačaka. (b) Skup tačaka koji nije linearno separabilan.

Slika 3.6: Prostor koji predstavlja ulazne vrednosti dva perceptrona. Na slici *(a)* je prikazan skup tačaka koje je perceptron ispravno klasifikovao, dok na slici *(b)* skup nije linearno separabilan pa stoga ne može biti klasifikovan pravom linijom.

Kada skupovi nisu linearno separabilni, potreban nam je model drugačiji od perceptrona u smislu da funkcija praga mora da zadovoljava određene uslove (npr. da bude neprekidna ili diferencijabilna). Jedno rešenje bio bi *sigmoid model* - model vrlo sličan perceptronu ali baziran na drugoj funkciji praga - *sigmoid funkciji*.

$$\sigma(y) = \frac{1}{1 + e^{-y}} \tag{3.2}$$

Sigmoid model prikazan je na slici 3.7.



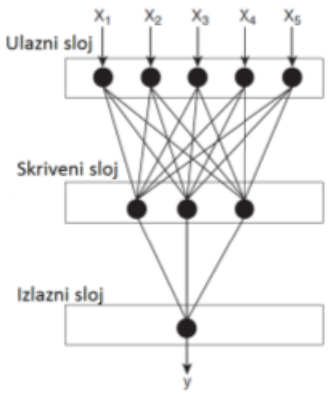
Slika 3.7: Sigmoid.

Često se koriste i druge funkcije praga kao što su log-sigmoid, hiperbolični tangens itd.

Veštačke neuronske mreže su organizovane po *slojevima* a slojevi su sastavljeni od *čvorova* koji se nazivaju *neuroni*. Razlikujemo tri vrste slojeva:

- Ulazni sloj – u kome su ulazni podaci predstavljeni modelu veštačke neuronske mreže.
- Izlazni sloj – u kome je prezentovan odgovor modela.
- Skriveni sloj – posao skrivenih slojeva je da transformiše vrednosti ulaznog sloja u nešto što bi izlazni sloj mogao da iskoristi.

Veštačka neuronska mreža može imati 0 ili više skrivenih slojeva. One koje imaju više skrivenih slojeva nazivaju se *višeslojne* – slika 3.8.



Slika 3.8: Višeslojna veštačka neuronska mreža.

Takođe, razlikuju se mreže koje "idu napred" (eng. feed-forward), gde su čvorovi jednog sloja povezani samo sa čvorovima sledećeg sloja i rekurentne mreže (eng. recurrent) gde čvorovi mogu biti povezani i sa čvorovima istog sloja i sa čvorovima prethodnih slojeva [5].

3.2.3 Osnovni algoritam za učenje perceptrona

Učenje perceptrona predstavlja osnovu za učenje veštačkih neuronskih mreža. Učenje perceptrona podrazumeva "fino podešavanje" težinskih faktora $w_1 ... w_n$ [5]. Na primer, perceptronom se mogu predstaviti mnoge logičke funkcije (jasno je da su ulazne vrednosti za opis logičkih funkcija iz skupa $\{0, 1\}$), a perceptron koji implementira boolean funkciju AND izgleda ovako:

$$y(x_1, x_2, \theta) = \text{sgn}\,(0.5x_1 + 0.5x_2 - 0.8) \tag{3.3}$$

Jedan od osnovnih algoritama za određivanje težinskih faktora opisan je u tabeli 3.2

Algoritam: Učenje perceptrona perceptron pravilom
Nasumično biramo početne težinske faktore $w_i, i = 1...n$ Beskonačna petlja: • Za svaku instancu iz trening skupa aktiviramo perceptron • Ukoliko je svaka instaca trening skupa ispravno klasifikovana izlazimo iz ciklusa • Inače, modifikujemo težinske faktore perceptron pravilom koje glasi: $w_i = w_i + \Delta w_i$, gde je $\Delta w_i = \eta(t - o)x_i$ t - ciljna vrednost trening instance o - ciljna vrednost koju je generisao perceptron η - pozitivna konstanta koja se naziva *faktor učenja*

Tabela 3.2: Učenje perceptrona *perceptron trening pravilom*

3.2.4 Primer klasifikacije teksta perceptron pravilom

U ovom delu pokušaćemo da opišemo rad prethodnog algoritma na primeru rešavanja problema binarne klasifikacije teksta – da li je novinski članak o mašinskom učenju ili ne. I kao i ranije posmatrajmo primer 3.2.

Najpre je potrebno što bolje opisati zadate instance atributima i odrediti ulazne vrednosti perceptrona. Kao i ranije, instance teksta opisaćemo služeći se modelom vreća reči. Trening skup je sada predstavljen u obliku vektora i izgleda ovako:

	mašinsko	učenje	generalizacija	veštačka	košarka
$\vec{x_1}$	2	2	0	1	0
$\vec{x_2}$	2	2	4	1	0
$\vec{x_3}$	0	0	1	0	5

Tabela 3.3: Tabela koja simulira trening skup klasifikatora koji je određen atributima i brojem pojavljivanja atributa u datoj instanci. U ovom primeru, instance $\vec{x_1}$ i $\vec{x_2}$ pretstavljaju tekstove o mašinskom učenju, dok instanca $\vec{x_3}$ tekst koji nije o mašinskom učenju.

Dakle, instance teksta opisane su atributima (rečima) i vrednostima koje predstavljaju broj pojavljivanja date reči u datoj klasi. Tako da, u našem primeru, reč "mašinsko" se u podacima trening skupa koji su o mašinskom učenju pojavljuje ukupno 4 puta (instance $\vec{x_1}$ i $\vec{x_2}$) dok se u skupu koji nije o mašinskom učenju pojavljuje 0 puta (instanca $\vec{x_3}$).

U procesu učenja, cilj je odrediti težinske faktore koji ispravno klasifikuju sve instance trening skupa. Algoritam započinjemo nasumičnim zadavanjem težinskih vrednosti npr. $w = [0, 0, ..., 0]$ i započinjemo prolazak kroz trening skup. U sledećem koraku "aktiviramo" perceptron za datu instancu i ukoliko je instanca pogrešno klasifikovana primenjujemo perceptron pravilo.

Algoritam: Učenje perceptrona perceptron pravilom

Ulaz: $T = \{(\vec{x_1}, y_1), (\vec{x_2}, y_2)...(\vec{x_n}, y_n)\}$

$\vec{w_0} = 0, k = 0$
for $i = 1$ to n
    if $y_i(\vec{w_k}\vec{x_i}) <= 0$
        $\vec{w_{k+1}} = \vec{w_k} + y_i\vec{x_i}$
        $k = k + 1$
    endif
endfor

Izlaz: $\vec{w_k}$

Tabela 3.4: Učenje perceptrona perceptron trening pravilom "korak po korak".

3.3 Naivni Bajesov metod

Naivni Bajesov metod predstavlja jednu od najvažnijih i najuspešnijih metoda klasifikacije teksta. Sistemi za klasifikaciju koje su implemetirali Lewis [14], Lang [12] i Joachims [13] upravo potvrđuju ovo tvrđenje, a performanse koje ovaj metod postiže često se porede sa preformansama koje postižu naprednije metode, kao što su metoda potpornih vektora (eng. support vector machines), stabla odlučivanja i metode bazirane na neuralnim mrežama [1].

Osnovu ove metode predstavlja Bajesova teorema koja je nazvana po Reverendu (Thomas) Bayes-u (1702 – 1761). Nakon njegove smrti, njegov prijatelj Richard Price, editovao je i prezentovao njegov rad 1763. zbog čega možemo sa sigurnošću tvrditi da je naivni Bajesov klasifikator poznat još od druge polovine 18tog veka.

Modeli koji su bazirani na ovoj metodi su u literaturi poznati pod raznim imenima kao što su, *jednostavni Bajes* (eng. simple Bayes) i *nezavisni Bajes* (eng. independence Bayes) i tako referencijraju na korišćenje Bajesove teoreme u klasifikatoru.

Ova metoda se i danas koristi i rešava probleme kao što su filtriranje elektronske pošte, utvrđivanje autorstva, utvrđivanje pola osobe na osnovu visine, težine i veličine stopala itd. Takođe, u skorije vreme metod je korišćen za procenjivanje rizika od raka nakon radioterapije.

Još jedna velika prednost ove metode je to što je dovoljna mala količina trening podataka kako bi se procenili parametri klasifikatora.

U ovom poglavlju predstavićemo naivni Bajesov klasifikator.

3.3.1 Definicija naivnog Bajesovog klasifikatora

Intuicija algoritma je prilično jednostavna i bazirana je na Bajesovoj teoremi i veoma jednostavnoj reprezentaciji teksta – reprezentacija "vreća reči" (eng. bag of words) a kao rezultat novoj instanci se dodeljuje klasa sa najvećom verovatnoćom [1].

Bajesova teorema u svom najjednostavnijem obliku glasi:
Teorema. Neka su A i B slučajni događaji i $P(B|A)$ i $P(A|B)$ uslovne verovatnoće. Tada važi,

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}, P(A) > 0, P(B) > 0. \tag{3.4}$$

Dokaz. Ako pođemo od definicije uslovne verovatnoće imamo $P(B|A) = \frac{P(AB)}{P(A)}$ i $P(A|B) = \frac{P(AB)}{P(B)}$. Poslednju jednakost možemo zapisati i kao $P(AB) = P(A|B)P(B)$. Kada uvrstimo poslednju jednakost u početnu dobijamo Bajesovu formulu.

Bajesov pristup za rešavanje problema klasifikacije teksta se svodi na dodelu ciljne vrednosti $v_j \in V$, za koju je verovatnoća $P(v_j|x)$ najveća, instanci x koja je opisana atributima $a_1, a_2, ..., a_n$. Označimo tu vrednost sa v_{MAP},

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j|a_1, a_2, ..., a_n) \tag{3.5}$$

pri čemu $argmax$ (eng. argument of the maximum) definišemo na sledeći način:

$$\underset{x}{\operatorname{argmax}} f(x) = \{x|\forall y : f(y) \leq f(x)\} \tag{3.6}$$

Primenom Bajesove teoreme na formulu 3.5 dobijamo:

$$\begin{aligned} v_{MAP} &= \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1, a_2, ..., a_n|v_j)P(v_j)}{P(a_1, a_2, ..., a_n)} \\ &= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2, ..., a_n|v_j)P(v_j) \end{aligned} \tag{3.7}$$

Pokušajmo sada da procenimo vrednosti $P(v_j)$ i $P(a_1, a_2, ..., a_n|v_j)$ na osnovu podataka za trening. Vrednost $P(v_j)$ možemo lako izračunati jednostavnim prebrojavanjem broja instanci u trening skupu čija je ciljna vrednost v_j. Vrednost izraza $P(a_1, a_2, ..., a_n|v_j)$ nije tako jednostavno izračunati. Problem je u tome što je broj izraza $P(a_1, a_2, ..., a_n|v_j)$ jednak proizvodu broja svih mogućih instanci opisanih nekim skupom atributa i broja svih ciljnih vrednosti. Stoga bi trening skup morao da sadrži jako veliki skup trening podataka.

Naivni Bajesov klasifikator je zasnovan na pretpostavci da su vrednosti atributa za datu ciljnu vrednost međusobno nezavisni. Formalnije, pretpostavka je da je za datu ciljnu vrednost v_j instance x, verovatnoća konjukcije atributa $a_1, a_2, ..., a_n$ jednaka proizvodu verovatnoća individualnih atributa:

$$P(a_1, a_2, ..., a_n|v_j) = \prod_i P(a_i|v_j)$$

Ako uvrstimo gore dobijenu jednakost u formulu 3.7 dobijamo formulu koja predstavlja naivni Bajesov klasifikator:

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(vj) \prod_i P(a_i|v_j) \tag{3.8}$$

gde v_{NB} predstavlja ciljnu vrednost iz V koju naivni Bajesov klasifikator vraća kao rezultat klasifikacije. Primetimo da je broj različitih izraza $P(a_i|v_j)$, koje računamo iz trening skupa, jednak proizvodu broja različitih atributa i broja ciljnih kategorija, što je znatno manje nego broj svih $P(a_1, a_2, ..., a_n|v_j)$.

Dakle, naivni Bajesov klasifikator se sastoji iz dva dela, gde se u prvom delu sprovodi izračunavanje vrednosti $P(v_j)$ i $P(a_i|v_j)$ na osnovu skupa podataka za trening, dok je drugi deo primena formule 3.8 za datu instancu x opisanu atributima $a_1, a_2, ..., a_n$.

U narednim poglavljima bavićemo se analizom i procenama uspešnosti ove metode, a sledi primer naivnog Bajesovog klasifikatora.

3.3.2 Primer klasifikacije teksta naivnim Bajesovim algoritmom

Ilustrujmo naivni Bajesov algoritam na primeru u kom ćemo odrediti da li nam je zadati novinski članak intersantan ili ne, na osnovu trening skupa dokumenata koje smo unapred označili kao interesantne ili neinteresantane [1].

Dakle, nama je dat trening skup i nepoznata funkcija koja uzima vrednosti iz skupa V, pri čemu skup V sadrži dve vrednosti – *da* za interesantne i *ne* za neinteresantne novinske članke. Zadatak je "naučiti" kako da iz trening skupa predvidimo vrednost novog članka.

Dva glavna problema na koja nailazimo kada želimo da primenimo naivni Bajesov algoritam su prvo, kako predstaviti tekst kao skup atributa i drugi, kako oceniti verovatnoće koje su potrebne za primenu algoritma.

Posmatrajmo tekst 3.2 o mašinskom učenju koji nam je poznat iz primera klasifikacije teksta stablima odlučivanja.

Prvi problem možemo jednostavno rešiti tako što ćemo zadati tekst prikazati kao skup svih reči u tekstu ili samo kao skup odabranih reči (podebljane reči). Bilo da tekst prikazujemo kao skup svih ili odabranih reči naša reprezentacija će izgubiti informaciju o redosledu reči u članku i sve što nam ostaje je skup reči i broj pojavljivanja svake reči. Ovakva reprezentacija teksta zove se "vreća reči" (eng. bag of words). Dakle, za svaku poziciju reči definišemo atribut koji kao vrednost uzima reč na toj poziciji. Gore pomenuti paragraf može biti opisan sa ukupno 61 atributa koji odgovaraju pozicijama reči u paragrafu. Dakle, vrednost prvog atributa je reč "mašinsko", drugog "učenje", trećeg "je" itd.

Sada, sa ovakvom reprezentacijom teksta možemo primeniti algoritam. Radi boljeg razumevanja pretpostavimo da se u našem trening skupu nalazi 700 dokumenata koje smo označili kao neinteresantne i 300 koje smo označili kao interesantne, i pretpostavimo da je gore pomenuti paragraf nova instanca koju treba klasifikovati. Primenimo formulu 3.8 na zadati dokument:

$$\begin{aligned} v_{NB} &= \underset{v_j \in \{da,ne\}}{\operatorname{argmax}} P(vj) \prod_{i=1}^{61} P(a_i|v_j) \\ &= \underset{v_j \in \{da,ne\}}{\operatorname{argmax}} P(vj) P(a_1 = masinsko|v_j) P(a_2 = ucenje|v_j) \\ &\quad ...P(a_{61} = generalizuju|v_j) \end{aligned}$$

Bajesova pretpostavka o nezavisnosti $P(a_1, a_2, ..., a_{61}) = \prod_{i=1}^{61} P(a_i|v_j)$ u ovom slučaju ukazuje da je verovatnoća reči koja se pojavljuje na određenoj poziciji potpuno nezavisna od reči koje se pojavljuju na drugim pozicijama što je očigledno pogrešno. Na primer, verovatnoća da se reč "učenje" nađe na određenoj poziciji je znatno veća ako se ispred nje nalazi reč "mašinsko". Srećom, ovaj metod u praksi pokazuje izvanredne rezultate u raznim problemima klasifikacije uprkos očiglednoj netačnosti ove pretpostavke.

Kako bismo odredili vrednost v_{NB} potrebno je izračunati verovatnoće $P(v_j)$ i $P(a_i = w_k|v_j)$ pri čemu w_k predstavlja k-tu reč u rečniku srpskih reči. Prvu

verovatnoću $P(v_j)$ je lako izračunati. Na osnovu trening skupa znamo da je 700 članka obeleženo kao neinteresantni, a 300 kao interesantni pa je $P(ne) = 0.7$, a $P(da) = 0.3$. Ocena druge, uslovne verovatnoće (na primer $P(a_1 = masinsko|da)$) će biti dosta ozbiljniji zadatak, jer moramo procenti ovu vrednost za sve kombinacije - pozicija, reč, vrednost. U našem slučaju broj pozicija u tekstu je 61, broj reči u srpskom jeziku je nešto više od 180.000, a skup vrednosti sadrži 2 elementa. Dakle, ukupno $2 * 61 * 180.000 \approx 22$ miliona vrednosti koje treba izračunati iz trening skupa.

Ovaj problem rešićemo novom, ovog puta razumnijom pretpostavkom kojom ćemo značajno umanjiti broj verovatnoća koje je potrebno proceniti. Naime, pretpostavimo da je verovatnoća da se u tekstu pojavi reč w_k iz rečnika na nekoj poziciji, jednaka za sve pozicije. Formalnije, pretpostavićemo da je $P(a_i = w_k|v_j) = P(a_m = w_k|v_j)$ za sve i, j, k, m. Sada, procenjujemo skup verovatnoća $P(a_1 = w_k|v_j) = P(a_2 = w_k|v_j)...P(a_{61} = w_k|v_j)$ kao jednu vrednost $P(w_k|v_j)$, koju ćemo koristiti bez obzira na poziciju reči. Dakle, sada je potrebno proceniti 2 * 180.000 različitih verovatnoća što je i dalje veliki broj, ali izvodljiv.

Kako bismo kompletirali algoritam, moramo odabrati metod kojim ćemo konačno procenti verovatnoće $P(w_k|v_j)$. Neka je, na primer, n broj trening dokumenata koji su označeni kao interesantni, a n_c broj pojavljivanja reči "mašinsko" u ovih n dokumenata. Naš zadatak je da ocenimo verovatnoću $P(masinsko|da)$ i kao prirodno rešenje se nameće ocena $P(masinsko|da) = n_c/n$. Dok ovakav metod obezbeđuje dobru procenu u većini slučajeva, postoje slučajevi kada ova ocena nije najbolje rešenje. Razmotrimo slučaj kada je n_c veoma mali broj. Zamislimo da je $P(masinsko|da) = 0.08$ i broj dokumenta označenih kao interesantni $n = 5$. Tada je vrednost $n_c = 0$. Dakle, ovakvim izborom ocene teško je oceniti verovatnoću malo verovatnih događaja.

Kako bismo izbegli ovaj problem možemo u našu ocenu uključiti skup od m "virtuelnih" uzoraka sa verovatnoćom p pa je sada,

$$P(w_k|v_j) = \frac{n_c + mp}{n + m} \tag{3.9}$$

pri čemu se za m često uzima ukupan broj događaja sa verovatnoćama za koje pretpostavljamo da su uniformno raspoređeni tj. $p = \frac{1}{m}$, iz čega proizilazi

$$P(w_k|v_j) = \frac{n_k + 1}{n + |Recnik|} \tag{3.10}$$

gde je n broj pozicija reči u svim trening dokumentima čija je vrednost v_j, n_k broj pojavljivanja reči w_k u tih n pozicija i $|Recnik|$ broj različitih reči sakupljenih iz trening skupa.

Dakle, konačni algoritam koristi naivni Bajesov klasifikator zajedno sa pretpostavkom da je verovatnoća pojavljivanja reči nezavisna od pozicije u tekstu. Konačni algoritam je dat u pseudo kodu, tabele 3.5 i 3.6. Primetimo da je algoritam prilično jednostavan. Za vreme procesa učenja procedura *Naivno Bajesovo učenje* prolazi kroz sve trening dokumente i iz njih izvlači informacije

relevantne za primenu procedure *Naivna Bajesova klasifikacija* – reči, sa brojevima pojavljivanja u svakoj od klasa. Procedura *Naivna Bajesova klasifikacija* koristi ove vrednosti kako bi izračunala vrednost v_{NB}. Bitno zapažanje je da je bilo koja nova reč koja se pojavljuje u novoj instanci, a ne pojavljuje u trening skupu jednostavno ignorisana.

Algoritam: Naivno Bajesovo učenje

Ulaz: $Primeri$ je skup instanci za trening zajedno sa njihovim ciljnim vrednostima, a V je skup ciljnih vrednosti.

1. Sakupiti sve reči, znake interpunkcije i druge tokene koji se pojavljuju u skupu $Primeri$

- $Recnik$ = skup svih različitih reči i drugih tokena koji se pojavljuju u bilo kom dokumentu iz skupa $Primeri$

2. Izračunati potrebne verovatnoće - $P(v_j)$ i $P(w_k|v_j)$

Za svaku klasu v_j iz skupa V
- dok_j = podskup skupa $Primeri$ u kojem se nalaze svi dokumenti označeni klasom v_j
- $P(v_j) = \frac{|dok_j|}{Primeri|}$
- $tekst_j$ = dokument kreiran konkatenacijom svih dokumenata iz skupa dok_j
- n = broj različitih reči u dokumentu $tekst_j$
- Za svaku reč w_k iz $Recnika$
  - n_k = broj pojavljivanja reči w_k u dokumentu $tekst_j$
  - $P(w_k|v_j) = \frac{n_k+1}{n+|Recnik|}$

Izlaz: Ova funkcija izračunava verovatnoće potrebne za primenu naivnog Bajesovog algoritma.

Tabela 3.5: Učenje modela naivnom Bajesovom metodom.

Algoritam: Naivna Bajesova klasifikacija

Ulaz: *Dokument* predstavlja tekst koji želimo da klasifikujemo

$pozicije$ = sve pozicije u *Dokument*
a_i predstavlja reč na i-toj poziciji
Vrati vrednost v_{NB} koja se izračunava po formuli:

$v_{NB} = \text{argmax}_{v_j \in V} P(vj) \prod_i P(a_i|v_j)$

Izlaz: ciljna vrednost skupa V kojoj dokument najverovatnije pripada

Tabela 3.6: Klasifikacija Naivnom Bajesovom metodom.

Poglavlje 4

Implementacija

U ovom poglavlju opisaćemo sistem za klasifikaciju novinskih članaka "Newsy" koja implementira naivni Bajesov algoritam.

4.1 Newsy arihtektura

Počećemo sa idejom Newsy klasifikatora. Naime, konačni cilj je da Newsy postane aplikacija koja bi imala mogućnost klasifikacije novinskih članaka raznim metodama (trenutno postoji implementacija samo naivne Bajesove metode) i time omogući poređenje uspešnosti različitih klasifikatora koji rešavaju ovaj problem.

Kako bi se omogućila modularnost, aplikacija je podeljena na nekoliko komponenata od kojih su najvažnije – Tokenizer, CorpusHandler i NaiveBayesClassifier, i u narednom tekstu ćemo se pozabaviti svakom od njih.

4.1.1 Tokenizer

Tokenizer je komponenta čija je uloga parsiranje i tokenizacija zadatog novinskog članka, a kao ulaz Tokenizer prima novinski članak u specifičnom formatu koji sadrži informacije potrebne za klasifikaciju. Na slici 4.1 je prikazan primer novinskog članka[1] koji zadovoljava gore pomenuti format.

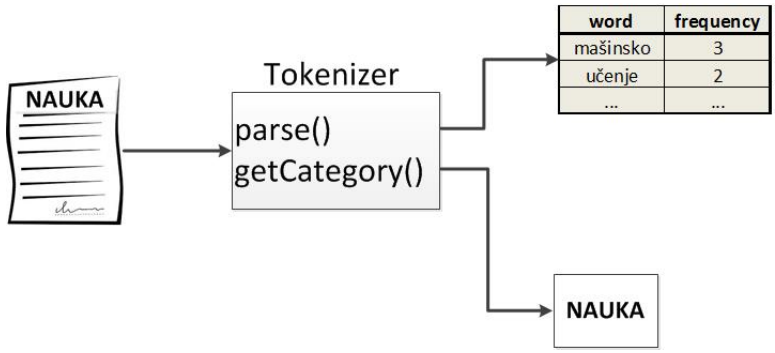
Kao rezultat parsiranja Tokenizer vraća strukturu koja sadrži sve reči koje se pojavljuju u tekstu sa brojem pojavljivanja svake od njih, a koja odgovara ranije pomenutoj reprezentaciji teksta "vreća reči". Takodje, Tokenizer ima mogućnost da na osnovu zaglavlja zadatog novinskog članka izvuče informaciju kojoj kategoriji novinski članak pripada (polje Rubrika) – slika 4.2.

[1] http://www.itnetwork.rs/Sta-je-masinsko-ucenje-Zahvaljujuci-GPU-ovima-vec-koristimo-pomenutu-tehnologiju-article-13140.htm

> Izvor: itnetwork.rs
> Datum: 26.04.2014.
> Rubrika: nauka
> Naslov: Šta je mašinsko učenje?
> Tekst:

Adobe, Baidu, Netflix, Yandex. Neka od najvećih imena u društvenim medijima i klaud računarstvu koriste NVIDIA CUDA bazirane GPU akceleratore kako bismo omogućili na prvi pogled magičnu pretragu, inteligentnu analizu fotografija i personalizovane preporuke filmova, sve bazirano na tehnologiji pod nazivom mašinsko učenje (eng. machine learning). Mašinsko učenje je baš ono kako i zvuči, treniranje računara kako da sami sebe nauče korišćenjem velike količine podataka. Na primer učenje indentifikacije lisice kroz analizu fotografija pasa, mačaka, ptica i drugih životinja, uključujući i lisica. Baš na način kako i ljudska bića uče.

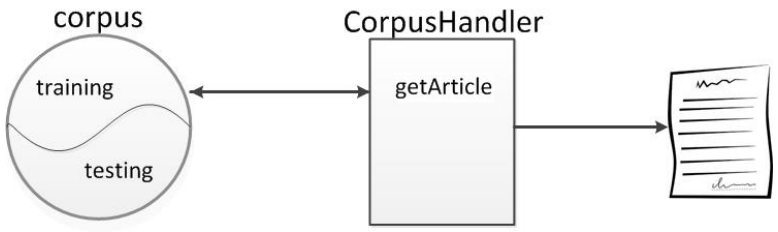
Slika 4.1: Format dokumenta koji *Tokenzer* prihvata.



Slika 4.2: Ilustracija rada objekta *Tokenzer*.

4.1.2 CorpusHandler

CorpusHandler je zadužen za učitavanje novinskih članka iz korpusa i računanje statistika vezanih za korpus (npr. broj članaka po rubrici). Njegova dužnost je da svojim korisnicima dostavi članak iz korpusa. Može biti inicijalizovan za svrhe učenja i testiranja, pri čemu su u prvom slučaju članci koje dostavlja iz dela korpusa namenjenog za učenje a u drugom iz dela namenjenog za testiranje – slika 4.3.

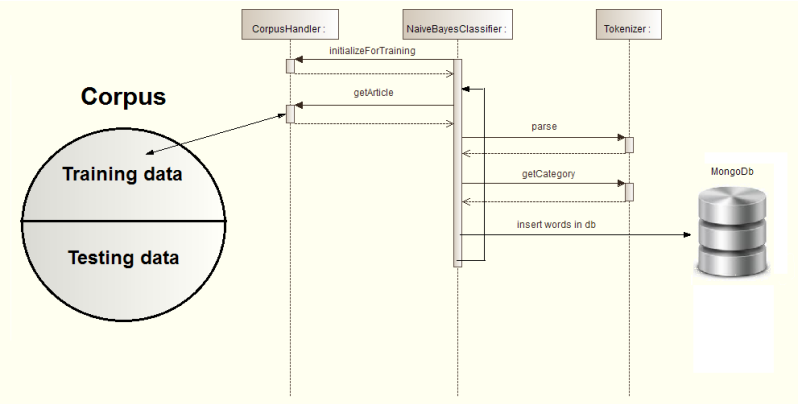


Slika 4.3: Funkcija *CorpusHandler-a.*

4.1.3 NaiveBayesClassifier

NaiveBayesClassifier predstavlja implementaciju samog algoritma i kao i ranije razmatraćemo dva scenarija – učenje klasifikatora i klasifikaciju nove instance.

Proces učenja započinje inicijalizacijom CorpusHandler-a pri čemu se za svaki novi novinski članak koji pristigne šalje Tokenizeru na parsiranje i prikupljanje ostalih informacija potrebnih za klasifikaciju (skup reči sa njihovim frekvencijama i kategorija kojoj novinski članak pripada). Nakon završene tokenizacije informacije se smeštaju u bazu podataka. Proces učenja opisan je diagramom na slici 4.4:



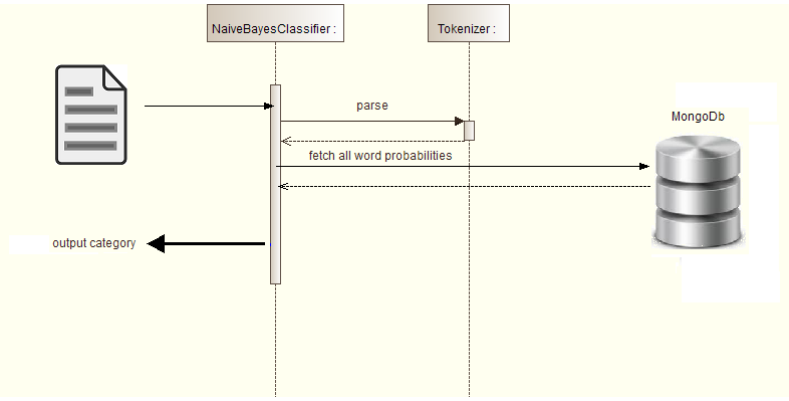
Slika 4.4: Učenje klasifikatora.

Najednostavnija verzija programa čuva u kolekciji sve reči sa brojem pojavljivanja u svakoj kategoriji što u nekim slučajevima nije baš dobro rešenje. Na primer, reč ”a” će imati jako visoku frekvenciju u svim kategorijama i zbog toga nam pojavljivanje te reči u novoj instanci neće dati neke posebno važne informacije. U poglavlju ”Korekcija grešaka sistema i poboljšanja” ćemo prikazati razne tehnike kojima ćemo pokušati da poboljšamo uspešnost klasifikatora. Na slici 4.5 je prikazana kolekcija reči sa brojevima pojavljivanja po katergorijama.

	_id	ekonomija	info_politika	sport
1	našao	4	9	15
2	neizbrojanih		1	
3	nemačkog	1	4	3
4	nemci	1	2	9
5	neshvatljiva		1	
6	nesreća	1	2	
7	nezaborav		1	
8	nezapamćenom		1	
9	obelodanjivanju		1	
10	obrazuju		1	
11	oseća	1	7	10
12	osećaju	1	11	2
13	osnovao		6	
14	ostatak	6	4	4
15	otkrivanju		1	
16	očuvanju		4	
17	pedesetogodišnjice		1	
18	pobrine		1	
19	pobune		6	
20	podjednako	1	7	2
21	pokušaju	2	12	5

Slika 4.5: Prikaz kolekcije reči sa frekvencijama pojavljivanja po kategorijama.

Sada, kada je naš klasifikator uspešno prikupio neophodne informacije za izračunavanje verovatnoća koje predstavljaju rezultat algoritma učenja opisanog u tabeli 3.5 možemo primenti algoritam klasifikacije (tabela 3.6) na novinski članak nepoznate kategorije – slika 4.6.



Slika 4.6: Klasifikacija nove instance.

4.2 Korišćeni alati

Najveći deo aplikacije napisan je u programskom jeziku *C++* a kao platforma za razvoj aplikacije izabran je *Ubuntu Linux* operativni sistem. Za razvoj grafičkog interfejsa korišćene su *QT* [16] bibilioteke. Parsiranje novinskih članaka rađeno je u *Lex*-u a rečnik sakupljen iz testiranih korpusa sačuvan je u *Mongo Db* [15] bazi podatka. B92 korpus je preuzet sa web-a skriptama napisanim u programskom jeziku *Perl*.

Poglavlje 5

Eksperimenti

U ovom poglavlju ćemo najpre predstaviti dostupne korpuse na kojima ćemo vršti testiranja a zatim i prikazti inicijalne rezultate koje je Newsy postigao na svakom od njih.

5.1 Korpusi klasifikovanih dokumenata

U ovom radu su korišćena dva korpusa na srpskom jeziku (Ebart i b92), a u planu je testiranje aplikacije na korpusima novinskih članaka na drugim jezicima.

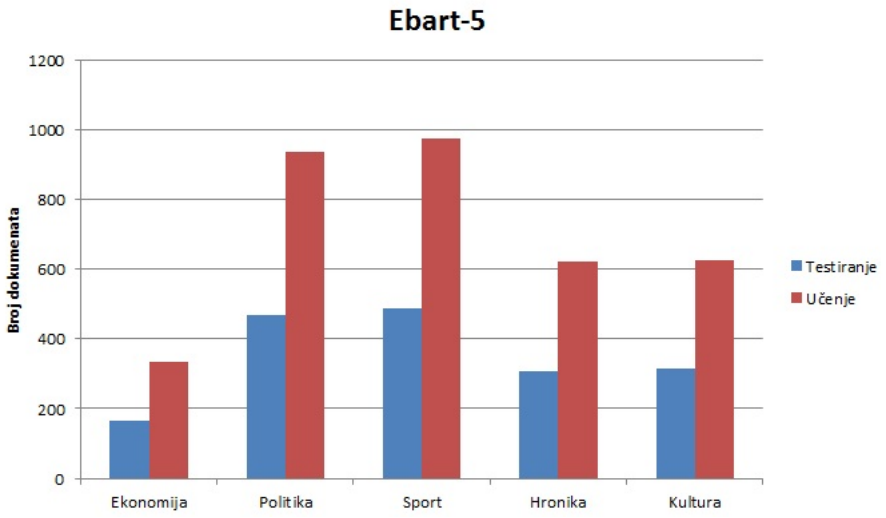
5.1.1 Ebart

Ebart[1] je novinski arhiv koji postoji od 2003. godine i do danas je u njemu uskladišteno više od 2.000.000 tekstova iz štampanih medija koji imaju nacionalnu pokrivenost, kao i odabranih lokalnih medija. Novinski (aktuelni) arhiv Medijskog arhiva Ebart predstavlja najveći korpus modernog srpskog jezika u digitalnom obliku.

Ebart je specijalizovan i za izradu analiza medija. Od osnovnih, kvantitativnih analiza, preko kvalitativnih analiza koje porede medijski tretman zadatih ključnih reči, pa sve do složenih analiza sadržaja prema metodologiji BBC-a.

U ovom radu korišćen je podskup ovog korpusa, Ebart-5, koji obuhvata pet različitih klasa *sport*, *ekonomija*, *politika*, *hronika* i *kultura*. Korpus sadrži 5235 novinskih članaka i podeljen je na podatke za učenje i podatke za testiranje u odnosu 2 : 1. Raspodela po klasama prikazana je na slici 5.1.

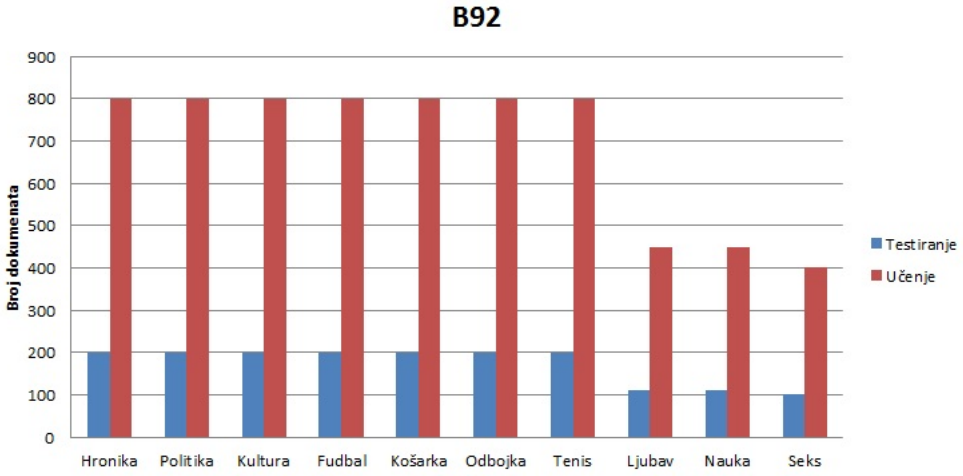
[1] www.arhiv.rs



Slika 5.1: Respodela dokumenata po kategorijama na Ebart-5 korpusu.

5.1.2 B92

Ovaj korpus se sastoji od kompletnih veb-izdanja vesti sakupljenih sa internet stranice televizije "B92" u periodu od 2012. godine do danas. Korpus sadrži oko 10000 članaka i 10 različitih klasa/rubrika – *politika*, *hronika*, *kultura*, *košarka*, *fudbal*, *tenis*, *odbojka*, *seks*, *ljubav* i *nauka*. Ovaj korpus je podeljen na podatke za učenje i podatke za testiranje u odnosu 4 : 1. Slika 5.2 prikazuje raspodelu novinskih članaka po kategorijama.



Slika 5.2: Respodela dokumenata po kategorijama na b92 korpusu.

5.2 Rezultati eksperimenata

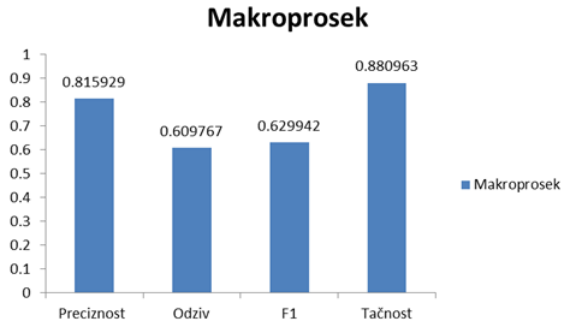
5.2.1 Rezultati na korpusu Ebart5

	ekonomija	hronika	politika	kultura	sport
ekonomija	29	10	114	6	8
hronika	1	132	164	5	7
kultura	0	11	443	5	8
politika	1	3	84	198	27
sport	0	5	58	2	423

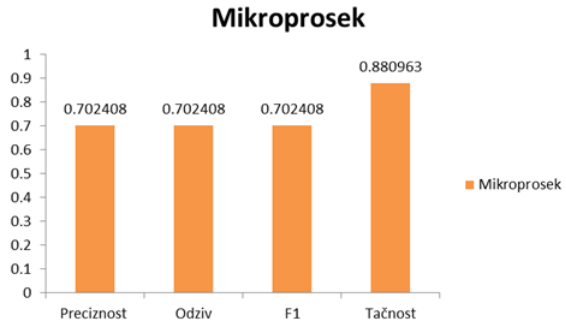
Tabela 5.1: Matrica konfuzije na korpusu Ebart5.

	preciznost	odziv	f1	tačnost
ekonomija	0.935484	0.173653	0.292929	0.919725
hronika	0.819876	0.427184	0.561702	0.881881
politika	0.513326	0.948608	0.666165	0.745413
kultura	0.916667	0.632588	0.748582	0.923739
sport	0.894292	0.866803	0.880333	0.93406

Tabela 5.2: Vrednosti mera uspešnost klasifikatora po kategorijama na korpusu Ebart5.



(a) Makroprosek na korpusu Ebart5.



(b) Mikroprosek na korpusu Ebart5.

Slika 5.3: Mikro i makro mere uspešnosti klasifikatora na korpusu Ebart5.

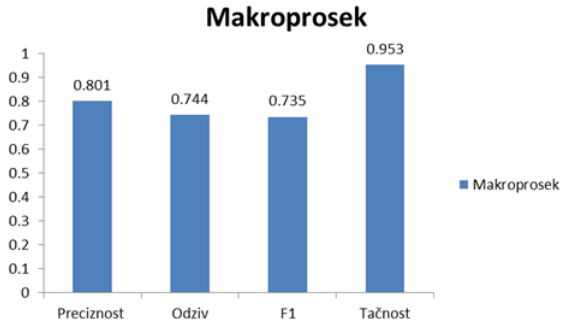
5.2.2 Rezultati na korpusu b92

	hronika	politika	kultura	fudbal	košarka	odbojka	tenis	ljubav	nauka	seks
hronika	142	35	5	0	1	0	0	0	17	0
politika	2	176	5	0	0	0	0	1	16	0
kultura	1	9	160	0	0	0	1	0	29	0
fudbal	2	9	10	142	20	9	0	0	8	0
košarka	1	5	5	3	174	4	0	0	8	0
odbojka	0	5	4	0	8	179	0	0	4	0
tenis	1	7	11	1	10	12	143	1	14	0
ljubav	0	1	2	0	0	0	0	83	26	0
nauka	0	2	6	0	0	0	0	2	100	2
seks	0	2	4	0	0	0	0	49	22	23

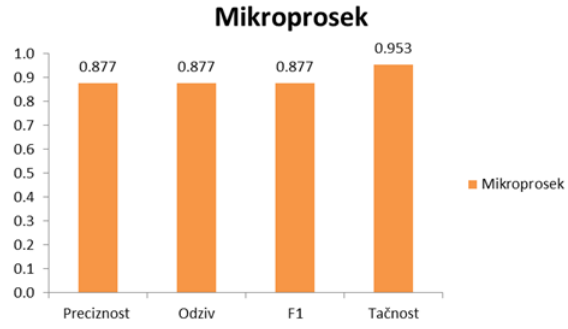
Tabela 5.3: Matrica konfuzije na korpusu b92.

	preciznost	odziv	f1	tačnost
hronika	0.95302	0.71	0.813754	0.962297
politika	0.701195	0.88	0.780488	0.942575
kultura	0.754717	0.8	0.776699	0.946636
fudbal	0.972603	0.71	0.820809	0.964037
košarka	0.816901	0.87	0.842615	0.962297
odbojka	0.877451	0.895	0.886139	0.973318
tenis	0.993056	0.715	0.831395	0.966357
ljubav	0.610294	0.741071	0.669355	0.952436
nauka	0.409836	0.892857	0.561798	0.909513
seks	0.92	0.23	0.368	0.954176

Tabela 5.4: Vrednosti mera uspešnost klasifikatora po kategorijama na korpusu b92.



(a) Makroprosek na korpusu b92.



(b) Mikroprosek na korpusu b92.

Slika 5.4: Mikro i makro mere uspešnosti klasifikatora na korpusu b92.

Poglavlje 6

Korekcija grešaka sistema i poboljšanja

Greške u sistemu za klasifikaciju teksta se najčešće manifestuju favorizovanjem jedne klase u odnosu na drugu. U ovom delu prikazaćemo problem koji sistem za klasifikaciju teksta zasnovan na naivnoj Bajesovoj metodi navodi na grešku prilikom klasifikacije i predložiti nekoliko metoda kojima se uticaji greške mogu značajno ublažiti. Takođe, videćemo kako boljim i pametnijim izborom atributa možemo uticati na performanse klasifikatora.

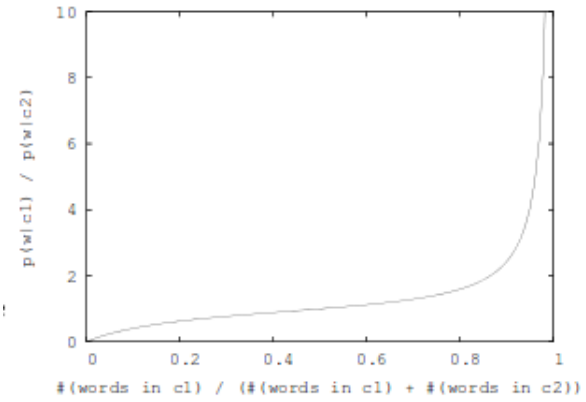
6.1 Problem nebalansiranih veličina klasa trening skupa

U ovoj sekciji pokazaćemo da više trening podataka u jednoj klasi u odnosu na drugu može dovesti do dobijanja težinskih faktora koji imaju loš uticaj na klasifikaciju. Ovo može prouzrokovati neželjeno ponašanje klasifikatora kojim se favorizuje klasa sa više trening podataka. Ovaj problem naziva se *problem nebalansiranih veličina klasa trening skupa* (eng. unbalanced class sizes problem) [10].

Naime, razmotrimo problem binarne klasifkacije u kome je w reč koja je irelevantna za klasifikaciju dokumenta i v_1 i v_2 klase. Ovo znači da verovatnoće $P(w|v_1)$ i $P(w|v_2)$ treba da budu jako bliske tj. $\frac{P(w|v_1)}{P(w|v_2)} = 1$ tako da reč ne bi imala zančajnijeg uticaja na klasifikaciju.

Pretpostavimo da se reč w pojavljuje sa jednakom relativnom frekvencijom od 0.1 u tekstovima obe kategorije. Takođe, pretpostavimo da je broj reči u rečniku odabranih reči 20000 a da ceo korpus sadrži 100000 reči. Slika 6.1 prikazuje odnos verovatnoća $P(w|v_1)$ i $P(w|v_2)$ u zavisnosti od odnosa veličina dve kategorije. Vrednost ima željenu vrednost 1 kada obe klase sadrže isti broj reči. Međutim, situacija se drastično menja ako se veličine klase promene. Na primer, kada bi klasa v_2 sadržala jako mali broj reči u odnosu na klasu v_1

prisustvo irelevantne reči w u test dokumentu bi značajno povećalo verovatnoću da dokument pripada klasi v_1 [11].



Slika 6.1: Odnos verovatnoća kategorija u zavisnosti od odnosa broja reči u klasama trening podataka.

Ukoliko obratimo pažnju na rezultate Newsy klasifikatora na skupu Ebart5 5.1 uočićemo da je klasa *politika* favorizovana u odnosu na klasu *ekonomija* jer su kategorije same po sebi bliske a klasa *politika* ima 3 puta više trening dokumenata. I zaista, nevažna reč kao što je *primer* se u trening dokumentima klase *ekonomija* javlja 25 puta dok se u klasi *politika* javlja 40 puta. Takođe, reč *ekonomija* se u klasi *ekonomija* javlja 3 puta dok se ista reč u klasi *politika* javlja 4 puta itd. Sve ovo uticaće na favorizovanje klase *politika*.

6.2 Eliminacija stop reči

Jedan od načina da se izvrši eliminacija irelevantnih atributa iz skupa jeste eliminacija stop reči. Stop reči su one reči koje ne nose informaciju ni o jednoj klasi. To su najčešće veznici, predlozi i prilozi. Karakteriše ih visoka frekvencija pojavljivanja. Primer stop reči za srpski jezik su: i, na, u, već itd. Stop reči se filtriraju na osnovu pripremljene liste reči ili na osnovu učestalosti pojavljivanja [5]. Slede rezultati testiranja Newsy klasifkatora koji stop reči ne uzima u obzir na prethodno pomenutim korpusima.

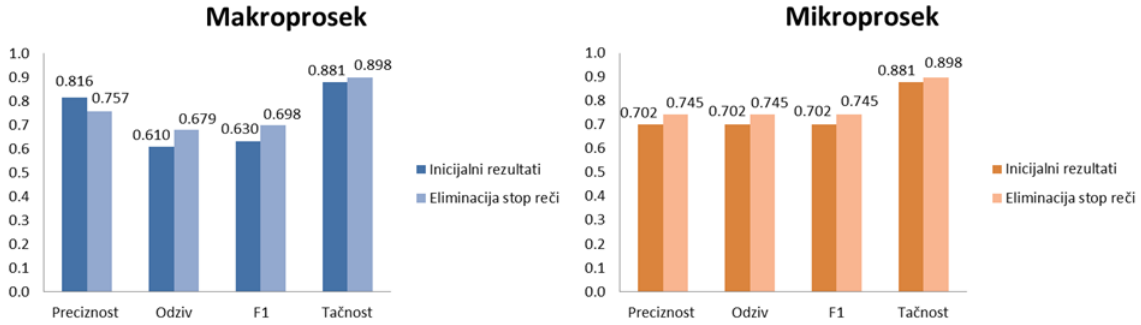
6.2.1 Rezultati na korpusu Ebart5

	ekonomija	hronika	politika	kultura	sport
ekonomija	60	7	74	7	19
hronika	12	173	108	6	10
politika	5	20	429	6	7
kultura	8	11	51	218	25
sport	10	15	38	5	420

Tabela 6.1: Matrica konfuzije na korpusu Ebart5.

	preciznost	odziv	f1	tačnost
ekonomija	0.631579	0.359281	0.458015	0.918578
hronika	0.765487	0.559871	0.646729	0.891628
politika	0.612857	0.91863	0.735219	0.822821
kultura	0.900826	0.696486	0.785586	0.931766
sport	0.873181	0.860656	0.866873	0.926032

Tabela 6.2: Vrednosti mera uspešnost klasifikatora po kategorijama na korpusu Ebart5.



(a) Makroprosek na korpusu Ebart5.

(b) Mikroprosek na korpusu Ebart5.

Slika 6.2: Mikro i makro mere uspešnosti klasifikatora na korpusu Ebart5.

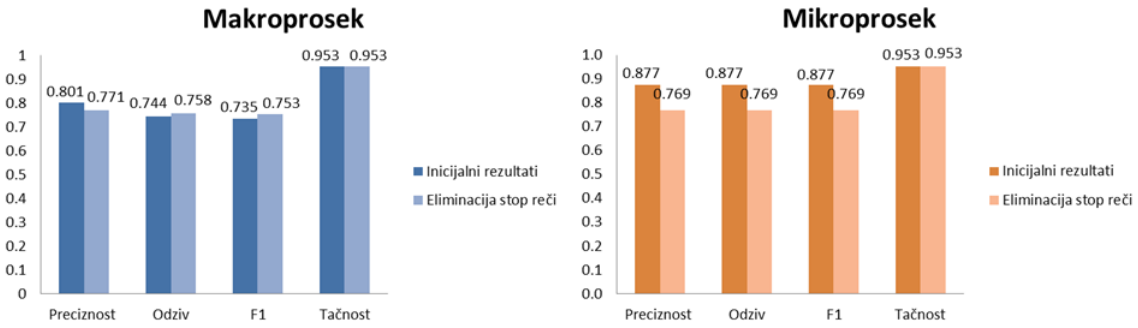
6.2.2 Rezultati na korpusu b92

	hronika	politika	kultura	fudbal	košarka	odbojka	tenis	ljubav	nauka	seks
hronika	154	24	4	0	1	4	1	4	7	1
politika	3	182	3	0	2	1	0	3	3	3
kultura	2	13	136	3	5	7	3	6	22	3
fudbal	1	8	3	122	33	25	4	0	3	1
košarka	2	4	2	5	176	6	1	1	3	0
odbojka	0	2	3	4	7	180	0	1	1	2
tenis	2	9	8	3	9	15	149	0	4	1
ljubav	1	1	2	1	0	1	0	95	6	5
nauka	1	10	7	3	1	2	0	3	83	2
seks	0	6	4	0	1	1	0	30	8	50

Tabela 6.3: Matrica konfuzije na korpusu b92.

	preciznost	odziv	f1	tačnost
hronika	0.927711	0.77	0.84153	0.966357
politika	0.702703	0.91	0.793028	0.944896
kultura	0.790698	0.68	0.731183	0.941995
fudbal	0.865248	0.61	0.715542	0.943735
košarka	0.748936	0.88	0.809195	0.951856
odbojka	0.743802	0.9	0.81448	0.952436
tenis	0.943038	0.745	0.832402	0.965197
ljubav	0.664336	0.848214	0.745098	0.962297
nauka	0.592857	0.741071	0.65873	0.950116
seks	0.735294	0.5	0.595238	0.960557

Tabela 6.4: Vrednosti mera uspešnost klasifikatora po kategorijama na korpusu b92.



(a) Makroprosek na korpusu b92.

(b) Mikroprosek na korpusu b92.

Slika 6.3: Mikro i makro mere uspešnosti klasifikatora na korpusu b92.

6.3 TF-DIF metoda

TF-IDF, skraćenica od eng. term frequency – inverse document frequency, je vrednost koja reflektuje koliko je reč važna u dokumentu u datom korpusu. Razne varijacije ove metode se koriste u sistemima za pretragu (eng. search engines) kao primarni alat za rangiranje važnosti dokumenta na osnovu pretrage koju je korisnik zadao. Takođe, tf-idf metoda se često koristi za odabir stop reči. Naime, reči čiji je tf-idf faktor manji od zadate konstante se mogu smatrati za stop reči [9].

Upoznajmo se sa ovom metodom kroz primer. Pretpostavimo da nam je dat skup tekstualnih dokumenata i želimo da zaključimo koji dokument najviše odgovara upitu "šta je mašinsko učenje". Jednostavan način da započnemo ovu pretragu je da eliminišemo sve dokumente koji ne sadrže sve četiri reči – "šta", "je", "mašinsko" i "učenje". Kako bismo unapredili našu pretragu možemo i prebrojavati koliko puta se data reč pojavila u svakom dokumentu. Broj pojavljivanja date reči u dokumentu se naziva *frekvencija reči* (eng. term frequency).

Frekvencija dokumenata (eng. document frequency) je vrednost koja predstavlja broj dokumenata iz kolekcije u kojima se pojavljuje određena reč. Intuitivno je jasno da reči "šta" i "je" nisu relevantne za razlikovanje važnih i nevažnih dokumenata za ovu pretragu koliko reči "mašinsko" i "učenje", i to je upravo zbog svoje visoke frekvencije dokumenata. Stoga, uvodimo meru *inverzna frekvencija dokumenata* (eng. inverse document frequency) koja predstavlja inverznu funkciju funkcije frekvencija dokumenata. Reči "šta" i "je" imaće nisku dok će reči "mašinsko" i "učenje" imati visoku inverznu frekvenciju dokumenta.

Kombinovanje mera frekvencija reči i inverzna frekvencija dokumenata dobijena je tf-idf mera. Dakle, tf-idf mera tvrdi da su najbolji atributi oni koji su frekventni u individualnim dokumentima ali se retko pojavljuju u ostalom delu kolekcije [5].

Sada ćemo predstaviti rezultate koje je Newsy postigao na test korpusima primenom eliminacije stop reči i tf-idf metode.

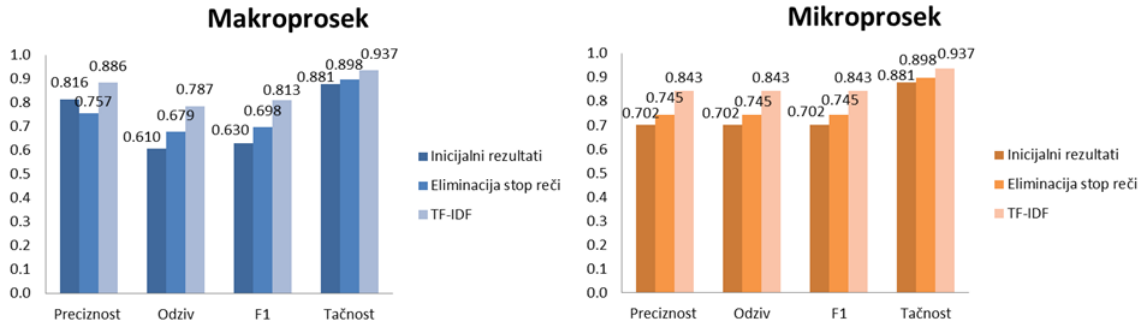
6.3.1 Rezultati na korpusu Ebart5

	ekonomija	hronika	politika	kultura	sport
ekonomija	83	6	75	3	0
hronika	2	193	105	6	3
politika	1	19	437	6	4
kultura	0	0	21	287	5
sport	0	2	13	3	470

Tabela 6.5: Matrica konfuzije na korpusu Ebart5.

	preciznost	odziv	f1	tačnost
ekonomija	0.965116	0.497006	0.656126	0.950115
hronika	0.877273	0.624595	0.729679	0.918005
politika	0.671275	0.93576	0.781753	0.860092
kultura	0.940984	0.916933	0.928803	0.974771
sport	0.975104	0.963115	0.969072	0.982798

Tabela 6.6: Vrednosti mera uspešnost klasifikatora po kategorijama na korpusu Ebart5.



(a) Makroprosek na korpusu Ebart5.

(b) Mikroprosek na korpusu Ebart5.

Slika 6.4: Mikro i makro mere uspešnosti klasifikatora na korpusu Ebart5.

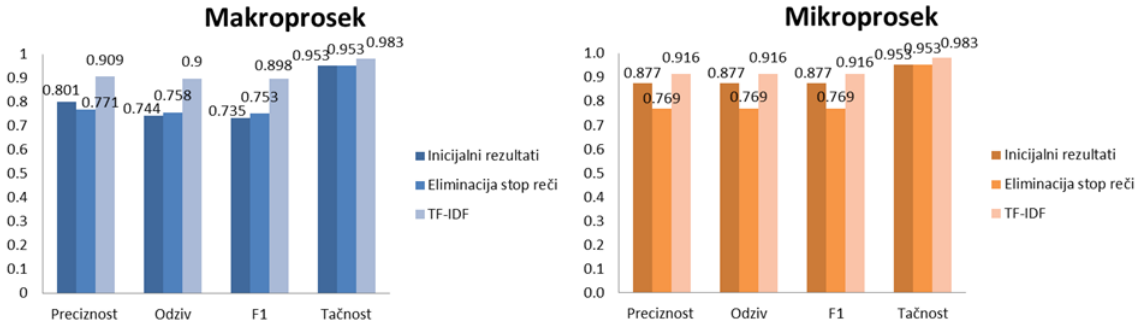
6.3.2 Rezultati na korpusu b92

	hronika	politika	kultura	fudbal	košarka	odbojka	tenis	ljubav	nauka	seks
hronika	183	15	1	0	0	0	0	0	1	0
politika	2	189	4	0	0	0	0	0	5	0
kultura	0	0	194	0	0	0	0	0	6	0
fudbal	0	1	5	186	3	1	0	0	4	0
košarka	1	2	3	1	187	0	0	0	6	0
odbojka	0	3	4	1	3	188	1	0	0	0
tenis	1	3	4	1	0	0	186	0	5	0
ljubav	0	0	4	0	0	0	0	98	6	4
nauka	0	0	2	0	0	0	0	0	110	0
seks	0	0	3	0	0	0	0	31	8	58

Tabela 6.7: Matrica konfuzije na korpusu b92.

	preciznost	odziv	f1	tačnost
hronika	0.97861	0.915	0.945736	0.987819
politika	0.887324	0.945	0.915254	0.979698
kultura	0.866071	0.97	0.915094	0.979118
fudbal	0.984127	0.93	0.956298	0.990139
košarka	0.968912	0.935	0.951654	0.988979
odbojka	0.994709	0.94	0.966581	0.992459
tenis	0.994652	0.93	0.96124	0.991299
ljubav	0.75969	0.875	0.813278	0.973898
nauka	0.728477	0.982143	0.836502	0.975058
seks	0.935484	0.58	0.716049	0.973318

Tabela 6.8: Vrednosti mera uspešnost klasifikatora po kategorijama na korpusu b92.



(a) Makroprosek na korpusu b92.

(b) Mikroprosek na korpusu b92.

Slika 6.5: Mikro i makro mere uspešnosti klasifikatora na korpusu b92.

Poglavlje 7

Zaključak

Ovaj rad opisuje jedan sistem za klasifikaciju teksta zasnovan na naivnoj Bajesovoj metodi i predstavlja osnovne tehinke koje su potrebne za izgradnju jednog sistema za klasifikaciju. Upoznali smo se i sa veoma važnim metodama nadgledanog mašinskog učenja, stablima odlučivanja i neuronskim mrežama, i predstavili ID3 algoritam i perceptron. Takođe, videli smo kako su određene tehnike kao što su eliminacija stop reči i tf-idf metoda uticale na uspešnost klasifikatora.

Newsy je zamišljen kao alat koji bi sadržao implementaciju više različitih metoda i kojim bi mogli jasno da uporedimo uspešnosti metoda koje rešavaju ovaj problem tako da su u planu implementacija *metode podržavajućih vektora* (eng. support vector machines), *metoda k najbližih suseda* (eng. k-nearest neighbours) itd. Takođe, postoji mnoštvo drugih tehnika za poboljšanje uspešnosti klasifikatora kao što su svođenje na koren reči (jedan od najčešće korišćenih algoritama za određivanje korena reči je Porterov algoritam), informativnost atributa (eng. information gain) itd.

Iz eksperimenata opisanih u radu se jasno vidi da ovaj metod zaista postiže odlične rezultate i da sa velikom uspešnoću rešava problem klasifikacije novinskih članaka. Sve ovo ukazuje da *naivni Bajesov algoritam nije ni malo naivan.*

Literatura

[1] Tom Mitchell. *Machine learning*. McGraw-Hill, March 1997.

[2] Predrag Janičić, Mladen Nikolić. *Veštačka inteligencija* - skripta, Matematički fakultet u Beogradu, Jun 2010.

[3] Yiming Yang, Thorsten Joachims. *Text categorization*, Scholarpedia, 2008. `http://www.scholarpedia.org/article/Text_categorization`

[4] `http://en.wikipedia.org/wiki/Federalist_Papers`

[5] Jelena B. Graovac. *Prilog metodama klasifikacije teksta: Matematički modeli i primene*, Jun 2014.

[6] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L.D. *Backpropagation applied to handwritten zip code recognition. Neural Computation*, 1989.

[7] Lang, K. J., Waibel, A. H., Hinton, G. E. *A time-delay neural network architecture for isolated word recognition*, 1990.

[8] Cottrell, G. W. *Extracting features from faces using compression networks: Face, identity, emotion and gender recognition using holons*, 1990.

[9] *TF-IDF method*, Wikipedia, 2014. `http://en.wikipedia.org/wiki/Tf-idf`

[10] Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, David R. Karger. *Tackling the Poor Assumptions of Naive Bayes Text Classifiers*, 1990.

[11] Eibe Frank, Remco R. Bouckaert. *Naive Bayes for Text Classification with Unbalanced Classes*, 1990.

[12] Lang, K. *Newsweeder: Learning to filter netnews*. In Prieditis and Russell (Eds.), Proceedings of the 12th International Conference on Machine Learning (pp. 331-339). San Francisco: Morgan Kaufmann Publishers, 1995.

[13] Joachims, T. *A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization*, (Computer Science Technical Report CMU-CS-96-118). Carnegie Mellon University, 1996.

[14] Lewis, D. *Representation and learning in information retrieval*, (Ph.D. thesis), (COINS Technical Report 91-93). Dept. of Computer and Information Science, University of Massachusetts, 1991.

[15] `http://docs.mongodb.org/manual`

[16] `http://qt-project.org`