# Tweet Analiza

Vještačka Inteligencija

**Vladimir Jovanović 42/19**

Računarske nauke
UCG
Podgorica
2021

# Sadržaj

# 1 Uvod

Svjedoci smo da u modernim vremenima društvene mreže postaju sve popularnije i popularnije. Kao posljedica toga, jako je česta pojava da neki podatak pročitan na internetu utiče na, ili u potpunosti definiše, nečije mišljenje o određenim temama, kao i da ljudi objavljuju mnogo informacija o svojoj svakodnevnosti i nekim aktuelnim informacijama.

Imajući u obzir koliko su informacije bitne, organizacijama je potrebno da imaju uvid u sve što se nalazi u polju njihovog djelovanja. S druge strane, zbog obilja informacija, posjedovanje uvida na sve njih, kao i provjeriti da li su one tačne, je jako zahtjevan zadatak.

Ovaj projekat koristi podatke dobijene sa društvene mreže Twitter, i za cilj ima da napravi što bolji model koji će na osnovu objava na Twitter-u moći da pruži informaciju o tome da li neka objava sadrži upozorenje za neku nepogodu, poput poplava, požara, zemljotresa i slično. Ovakav model bi koristile odgovarajuće organizacije u cilju što bržeg i efikasnijeg odgovora na ovakve situacije.

Izvještaj je podijeljen na dvije manje cjeline: preprocesiranje, u kojoj je opisano kako je dati dataset obrađen, kao i kratak opis tehnika koje se koriste, i ekstrakciju podataka, u kojoj su opisani implementirani pristupi za rad sa podacima. Konačno, u izvještaju je takođe i jedna velika cjelina u kojoj će biti opisani algoritmi mašinskog učenja koji su primijenjeni zajedno sa rezultatima njihove primjene na obrađenim podacima.

# 2 Preprocesiranje

Preprocesiranje je tehnika koja dozvoljava transformaciju sirovih podataka u neki razumljiv format. Na primjer, podaci koji su dati u dataset-u sadrže mnoge specijalne karaktere koji nam nisu od značaja i mogu da otežaju rad algoritama.

U ovoj fazi izvršavamo sledeće akcije:

- lowercasing - sav dostupan tekst prebacujemo u mala slova da bi pojednostavili rad sa njim

- uklanjanje karaktera koji označava novi red

- uklanjanje specijalnih karaktera

- uklanjanje brojeva

- uklanjanje više vezanih bjelina i zamjena sa jednim blank karakterom

- lematizacija - tehnika u kojoj svaku riječ ponaosob prevodimo u svoj osnovni oblik i na taj način na osnovu starih rečenica formulišemo nove

# 3 Ekstrakcija podataka

Nakon što smo preprocesirali podatke, sada možemo da izvučemo korisne informacije i da ih reprezentujemo. U projektu smo koristili dva načina reprezentacije: Bag of words i n-Grams.

## 3.1 Bag of words

Jedan od najveći problem u radu sa običnim tekstom je to što je on neuređen. Takođe, algoritmi mašinskog učenja rade sa brojnim vrijednostima, tako da reprezentacija teksta putem stringova nije nešto što će im olakšati rad.

Kao rješenje na oba ova problema javlja se Bag of words. On funkcioniše tako što se prosto prati koliko puta se koja riječ javila u tekstu, što će kasnije moći da se iskoristi u svrsi određivanja važnosti određenih riječi. U ovom modelu je samo bitno koliko puta se riječ javila u tekstu, a ne gdje konkretno se riječ nalazi.

Sada se vidi značenje preprocesiranja. Konkretno, moguće je da se neka riječ javlja jako rijetko, ali da ona gotovo izvjesno označava da je došlo do nepogode. Međutim, takođe je moguće da se neki specijalni karakter igrom slučaja javlja u rečenicama u kojima je nepogoda opisana, takođe jako rijetko. Tada će algoritmi pretpostaviti da taj specijalni karakter označava da je došlo do nepogode, iako to nije nužno tako.

Takođe, bitna karakteristika Bag of words-a je da razlikuje velika i mala slova u istoj riječi, pa će tako, na primjer, riječi Učenje i učenje biti posmatrane kao različite, iako su iste. Ovo je takođe jedan od problema koje preprocesiranje rješava.

## 3.2 N-Grams

Iako izvršava ono što tražimo, Bag of words ima jednu evidentnu manu - uzima samo jednu riječ. Uzmimo, na primjer, rečenicu ”Bag of words je jako dobar.”. Bag of words pristup bi posebno uzeo riječi ”jako” i ”dobar”, no to narušava kontekst fraze ”jako dobar”. Cilj ovog primjera je bio da se u svim jezicima jako često javljaju riječi koje ponaosob ne doprinose pravom značenju neke rečenice, ali u kombinaciji sa drugim riječima u potpunosti mijenjanju značenje i daju doprinos kontekstu rečenice. Ovo je problem koji rješavaju n-grami.

Ovo je pogotovo evidentno kod negacije. Rečenica ”nema zemljotresa”označava da nema nepogoda, međutim kada Bag of words vidi riječ zemljotres, on može odmah da pretpostavi da je riječ o nepogodi. U skladu sa prirodom problema, ovakva pojava nije u potpunosti loša - bolje je biti spreman nego iznenađen, ali u drugim problemima ovo može da se pokaže kao veliki problem.

N-gram je tehnika u kojoj se umjesto jedne uzima više riječi, preciznije, n riječi. Na osnovu ovoga se pravi Bag of n-grams. U ovoj implementaciji su isprobani 2-gram i 3-gram (bigrami i trigrami).

## 3.3 TF-IDF

TF-IDF označava Term frequency - inverse document frequency. Primjetimo da Bag of words broji koliko puta se koja riječ nalazi u tekstu. Šta znači da se riječ često pojavljuje ? Da li je to znak da je ta riječ ona koja nam pomaže da pronađemo nepogodu ili to znači da je to neka riječ koja se prosto često javlja u jeziku i ne doprinosi značenju teksta ?

U praksi, jako često se dešava da je upravo slučaj da riječ koja se jako često javlja ne doprinosi značenju teksta mnogo. Uzmimo za primjer riječ ”the”u engleskom jeziku. Ona se javlja jako često u rečenicama svih priroda i značenja, i kao takva nam ne doprinosi značenju teksta uopšte.

Jedan pristup je da se učestalost riječi reskalira na osnovu podatka u koliko se dokumenata ta riječ pojavljuje, tako da se riječi koje se jako često pojavljuju u mnogo dokumenata, poput ”the”, ”kazne” smanjivanjem vrijednosti u Bag of words-u. Ovo je pristup koji Tf-Idf koristi.

TF ili term frequency se može računati na više načina. Na primjer, moguće je prosto prebrojati koliko puta se riječ pojavljuje u tekstu. S druge strane, takođe je moguć i napredniji postupak, gdje broj pojavljivanja riječi u tekstu dijelimo sa brojem riječi, sa brojem ponavljanja riječi koja se najčešće pojavljuje u tekstu ili sa zbirom ponavljanja svih riječi. Na osnovu toga, formula za TF glasi:

$$TF(i,j) = \frac{n(i,j)}{\sum n(i,j)} \tag{1}$$

gdje je n(i,j) - broj koji označava koliko puta se n-ta riječ javila u tekstu.

IDF ili inverse document frequency označava koliko često se riječ javlja u različitim dokumentima. Što je bliže 0, to je riječ češća u dokumentima. Računa se tako što se uzima ukupan broj dokumenata, dijeli sa brojem dokumenata koji sadrže riječ i zatim logaritmuje.

$$IDF = 1 + \log(\frac{N}{dN}) \tag{2}$$

gdje je N broj dokumenata, a dN broj dokumenata u kojima se riječ pojavila. Primjećujemo da se vrši sabiranje sa 1 iako to nije u opisu naznačeno. To je da se riječi koje se javljaju u svakom dokumentu ne izostave u potpunosti, već da se uzmu sa malim značenjem. Ovaj proces se zove IDF smoothing.

Konačno TF-IDF vrijednost se računa kao proizvod 1 i 2.

$$TF - IDF = TF * IDF \tag{3}$$

# 4 Analiza sentimenta teksta

U ovom poglavlju vršimo analizu sentimenta uz pomoć sledećih algoritama klasifikacije: Logistička regresija, Support vector machine, K-nearest neighbours, Decision tree, Random forest i Naive Bayes - Gaussian i Mutinomial. Za svaki od algoritama je istaknut opis načina rada, kao i rezultati koje postiže koristeći Bag of words, bigrame i trigrame. Detaljno izvođenje neće biti prikazano.

Prije toga, uvodimo mjere za uspješnost algoritma, precision, recall i f1-score.

## 4.1 Mjere

Uvodimo sledeće oznake: TP - true positive, rezultat gdje je model ispravno svrstao u pozitivnu klasu TN - true negative, rezultat gdje je model ispravno svrstao u negativnu klasu FP - false positive, rezultat gdje je model neispravno svrstao u pozitivnu klasu FN - false negative, rezultat gdje je model neispravno svrstao u negativnu klasu

Kao što je već naznačeno, mjere koje ćemo koristiti su precision, recall i f1-score.

One imaju sledeće formule:

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

Precision označava koliko je od onih koje je model svrstao u pozitivne zapravo pozitivno.

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

Recall označava koliko je od onih koji su pozitivni model svrstao u pozitivne.

$$f1\_score = \frac{2 * (Precision * Recall}{Precision + Recall} \tag{6}$$

## 4.2 Logistička regresija

Logistička regresija je jedan od algoritama nadgledanog mašinskog učenja koji se koristi za binarne klasifikacione probleme. Vrlo je sličan linearnoj regresiji, ali za hipotezu uzima funkciju:

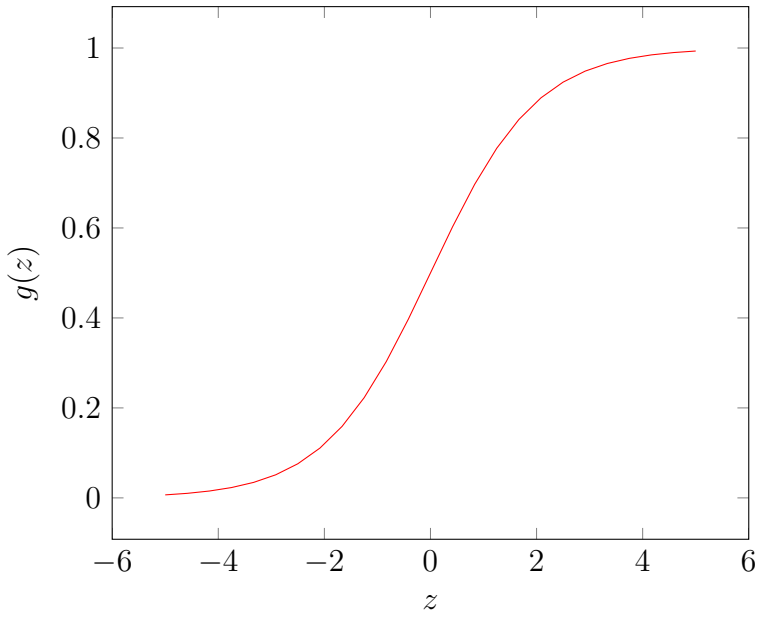
$$h_\theta = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} \tag{7}$$

gdje je

$$g(z) = \frac{1}{1 + e^{-z}} \tag{8}$$

poznata kao logistička ili sigmoid funkcija.

Grafik sigmoid funkcije axis lines




Primjetimo da je sigmoid funkcija ograničena između 0 i 1. Druge funkcije koje su monotone i ograničene na 0 i 1 takođe mogu biti iskorištene.

Ovakva hipoteza predstavlja vjerovatnoću da neki primjer pripada klasi 1. Ukoliko je hipoteza veća od 0.5, on pripada klasi 1, a ukoliko je manja, pripada klasi 0.

Kao i kod linearne regresije, maksimizaciju odgovarajuće funkcije:

$$L(\theta) = \prod_{i=1}^{m} p(y = y^{(i)} | x^{(i)}; \theta) \tag{9}$$

ćemo izvesti koristeći algoritam gradijentnog spusta:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{10}$$

gdje je:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) * x_j^{(i)} \tag{11}$$

Rezultati koje pokazuje logistička regresija prikazujemo u sledećim tabelama.

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 80 | 88 | 84 |
| 1 | 81 | 70 | 75 |

Tabela 1: Logistička regresija sa Bag of words (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 65 | 96 | 78 |
| 1 | 86 | 34 | 48 |

Tabela 2: Logistička regresija sa bigramima (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 60 | 99 | 75 |
| 1 | 95 | 14 | 25 |

Tabela 3: Logistička regresija sa trigramima (%)

Međutim, kako se problem tiče nepogoda koje imaju ogroman uticaj na ljude, jedna od mogućih varijanti je promjena težine klasa. Preciznije, bolje je staviti da je klasa 1 (da se desila nepogoda) važnija od klase 0, to jest da ima veću težinu. Rezultate kada je klasa 1 duplo važnija od klase 0 prikazujemo u sledećim tabelama.

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 84 | 72 | 78 |
| 1 | 69 | 82 | 75 |

Tabela 4: Logistička regresija sa Bag of words i promjenjenim težinama(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 81 | 12 | 20 |
| 1 | 45 | 96 | 62 |

Tabela 5: Logistička regresija sa bigramima i promijenjenim težinama(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 78 | 5 | 9 |
| 1 | 43 | 98 | 60 |

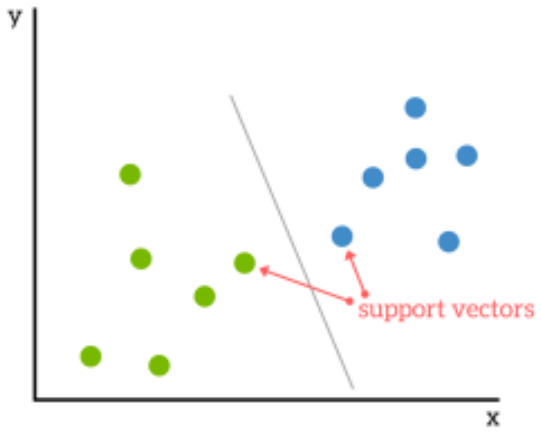
Tabela 6: Logistička regresija sa trigramima i promijenjenim težinama (%)

Shodno očekivanjima, model je više mnogo primjeraka svrstao u pozitivne, čak i kada oni to nisu. To vidimo po sniženoj preciznosti i znatno povećanom recallu za klasu 1.

## 4.3 Support Vector Machine

Support Vector Machine (SVM) je algoritam nadgledanog mašinskog učenja koji može da se koristi i za probleme klasifikacije i za probleme regresije. Zasnovan je na ideji nalaženja hiperravni koja će će najbolje podijeliti podatke na one iz klase 1 i klase 0.

Support vektori su podaci koji su najbliži hiperravni, to jest podaci koji, kada bi bili uklonjeni, u potpunosti mijenjaju poziciju hiperravni.




Slika 1: Support vektori

Odgovor na pitanje šta je hiperravan ćemo dati kroz primjer. Uzimimo 2D prostor. U njemu hiperravan možemo zamisliti kao liniju koja dijeli podatke na dvije klase. Intuitivno, što je tačka dalja od ravni, to smo sigurniji da se ona nalazi u klasi koja joj je dodijeljenja. Samim tim nam je cilj da sve tačke budu što dalje od hiperravni, a da i dalje ostanu na pravoj strani te ravni.

Udaljenost između hiperravni i najbližeg podatka se naziva margina, i cilj je odabrati hiperravan takvu da je margina što veća, jer ćemo onda biti sigurniji da će novi podaci biti svrstani ispravno. Kada podatke nije moguće podijeliti u n-dimenzionom prostoru, onda se prelazi na n+1-dimenzioni prostor, što analogno znači da naša hiperravan takođe postaje n+1-dimenziona. Algoritam funkcioniše tako što povećava dimenzije prostora, sve dok ne dođe do one u kojoj hiperravan pravilno dijeli podatke na klase.

Rezultate modela prikazujemo u sledećim tabelama:

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 79 | 89 | 84 |
| 1 | 83 | 69 | 75 |

Tabela 7: SVM sa Bag of words (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 67 | 93 | 78 |
| 1 | 79 | 35 | 48 |

Tabela 8: SVM sa bigramima (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 60 | 99 | 75 |
| 1 | 90 | 16 | 26 |

Tabela 9: SVM sa trigramima (%)

SVM u sklearn-u ima parametar gamma,za koji, što je on veći, to se model više trudi da prilagodi podatke training setu. Jasno, postavljanje ovog parametra na preveliku vrijednost dovodi do overfittinga. Prikazujemo rezultat postavljanja ovog parametra na 1 (po defaultu je mnogo manji od 1).

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 76 | 91 | 83 |
| 1 | 84 | 64 | 73 |

Tabela 10: SVM sa Bag of words i gamma=1 (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 67 | 95 | 79 |
| 1 | 83 | 36 | 50 |

Tabela 11: SVM sa bigramima i gamma=1 (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 63 | 99 | 77 |
| 1 | 95 | 15 | 26 |

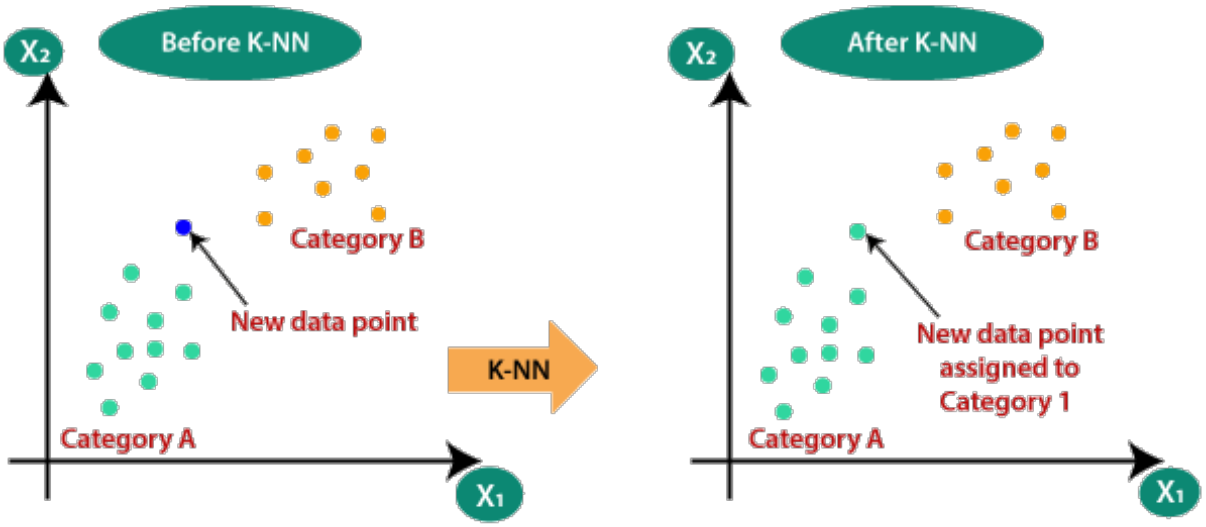
Tabela 12: SVM sa trigramima i gamma=1 (%)

## 4.4 K-Nearest Neighbours

K-Nearest neighbours je nadgledani algoritam mašinskog učenja. On pokušava da predvidi ispravnu klasu za test podatke tako što računa rastojanje između test podataka i svih trening podataka.

Pretpostavimo da želimo da dodamo novu tačku x1 modelu. KNN funkcioniše na sledeći način:

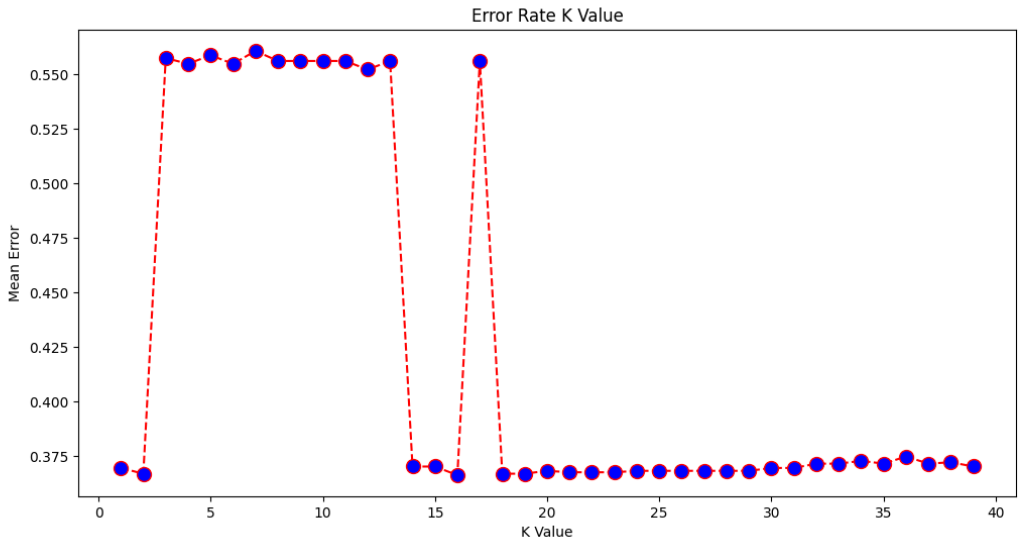
1. Bira se broj K koji će da predstavlja broj najbližih susjeda tačke x1.
2. Za svaki od susjeda, računa se Euklidska distanca.
3. Na osnovu izračunatih distanci, bira se K najbližih susjeda tački x1.
4. Broji se u koje klase spada koliko od K susjeda.
5. Tačka x1 se dodjeljuje klasi koja ima najviše susjeda.




Slika 2: KNN

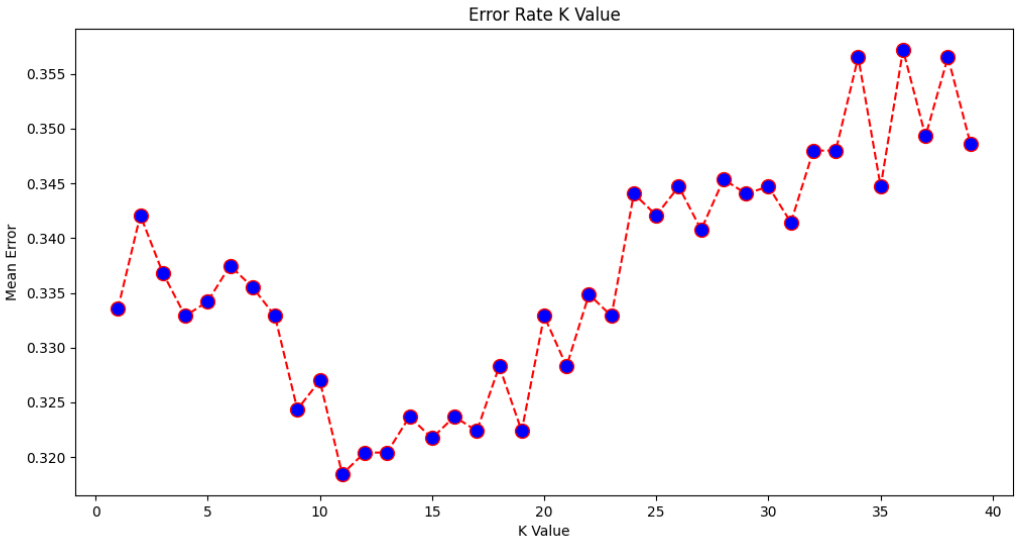
Jasno, K je parametar koji sami unosimo. Kao takav, trebao bi da postoji efikasan način za biranje broja K. Nažalost, ne postoji predefinisana statistička metoda za odabir te vrijednosti. Međutim, postoje neke generalne smjernice.

1. Preporučljivo je, na početku, staviti nasumično odabrano K i vidjeti kako se algoritam ponaša.
2. Odabir premale vrijednosti K povećava rizik od greške.
3. Veće K je preporučljivo u problemima klasifikacije jer će doprinijeti boljem modelu.




Slika 3: Vrijednosti za K na našem modelu za trigrame

Kao što vidimo sa slike iznad, vrijednosti za K između 3 i 13 daju jako loše rezultate, dok vrijednosti 18 i više daju mnogo bolje. Ovaj primjer je rađen sa trigramima. Sličnom dedukcijom dolazimo do zaključka da je u Bag of words pristupu najbolje K=1, a u bigramima K=11.




Slika 4: Vrijednosti za K na našem modelu za bigrame

U našem modelu je uzeto K=1 za BoW, K=11 za bigrame i K=20 za trigrame. Rezultati takvog modela su prikazani u sledećim tabelama.

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 69 | 88 | 77 |
| 1 | 76 | 50 | 60 |

Tabela 13: KNN sa Bag of words (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 64 | 96 | 77 |
| 1 | 88 | 35 | 50 |

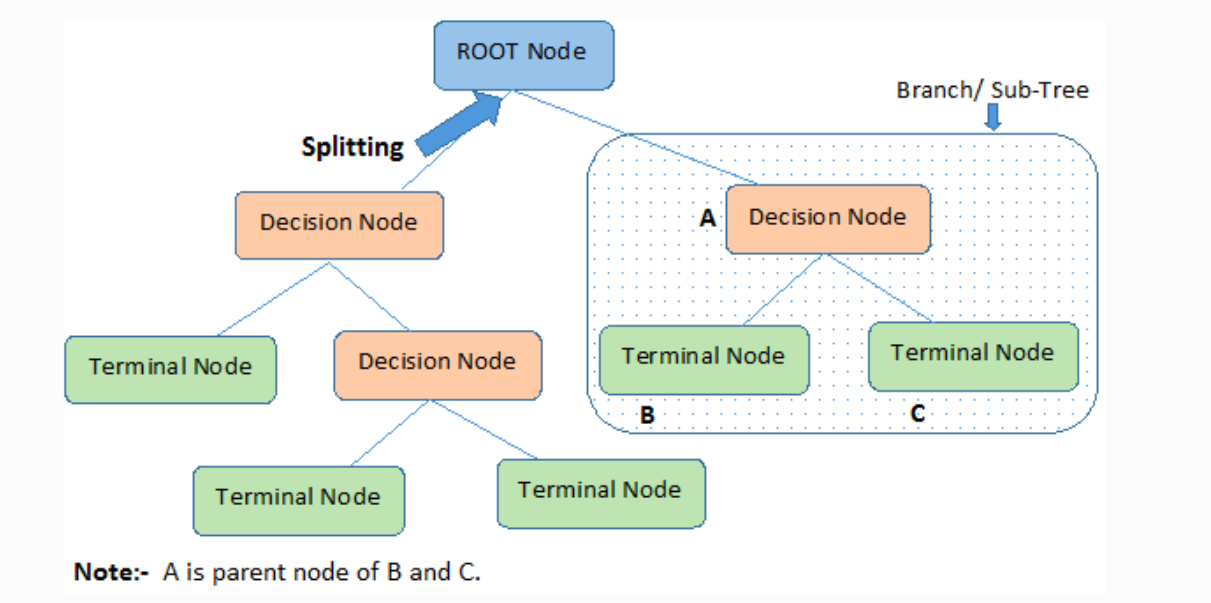
Tabela 14: KNN sa bigramima (%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 60 | 99 | 75 |
| 1 | 93 | 12 | 22 |

Tabela 15: KNN sa trigramima(%)

## 4.5 Decision Tree

Decision tree ili Stablo odlučivanja takođe pripada familiji algoritama nadgledanog mašinskog učenja. Cilj stabla odlučivanja je da napravi model koji se može iskoristiti za klasifikaciju tako što će naučiti jednostavna pravila zaključivanja koja su izvedena iz podataka za treniranje.




Slika 5: Opšti dijagram stabla odlučivanja

Na slici, Root node predstavlja uzorak na kom se primjenjuje model. Decision node je čvor koji se dalje grana na više čvorova, i on služi da se odabere neka odluka sa ciljem dolaska do listova, to jest terminalnih čvorova.

Stabla odlučivanja vrše klasifikaciju tako što sprovedu uzorak od korijena do lista, gdje list predstavlja neku od mogućih klasa.

Koriste se mnogi algoritmi pri odlučivanju da li se, i na koje čvorove neki čvor grana. Stablo odlučivanja dijeli čvor po svim dostupnim promjenljivima i bira podjelu u kojoj su podčvorovi najviše homogeni. Neki od tih algoritama su ID3, CART, CHAID... Oni neće biti dalje elaborirani u ovom izvještaju.

Takođe, postoje određene metode kojima se procjenjuje kvalitet podjele, poput Information gain-a i Gini-a. U implementaciji koristimo Gini metod procjene kvaliteta podjele, tako da ćemo ga ukratko objasniti.

Kao što je već rečeno, Gini indeks se može interpretirati kao funkcija cijene neke podjele. Računa se sledećom formulom:

$$Gini = 1 - \sum_{i=1}^{C} (P_i)^2 \tag{12}$$

gdje je Gini vrijednost Gini indeksa, C broj klasa u modelu i

$$P_i$$

je vjerovatnoća da uzorak pripada klasi i. Ovakav metod procjene podjele favorizuje veće particije i lak je za implementaciju.

Rezultati primjene decision tree algoritma su prikazani u sledećim tabelama:

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 75 | 76 | 76 |
| 1 | 67 | 66 | 67 |

Tabela 16: Decision tree sa Bag of Words(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 66 | 93 | 79 |
| 1 | 88 | 33 | 48 |

Tabela 17: Decision tree sa bigramima(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 61 | 99 | 76 |
| 1 | 96 | 17 | 29 |

Tabela 18: Decision tree sa trigramima(%)

## 4.6 Random Forest

Random forest, poput svih algoritama do sada, je algoritam klasifikacije, i kao što mu naziv implicira, sastoji se od mnogo stabala odlučivanja koja funkcionišu nezavisno. Svako stablo nezavisno od ostalih daje svoj rezultat i uzima se ona klasa koja je odabrana najviše puta.

Ovaj princip funkcionisanja, iako jednostavan, je jako moćan. Razlog tome je što stabla međusobno ”brane” jedna druge od individualnih grešaka, pod uslovom da sva ne griješe na istom mjestu.

Odgovor na pitanje kako random forest sprječava da ponašanje jednog stabla ima preveliku korelaciju sa ponašanjem drugog stabla se odražava kroz dva svojstva: Bagging i Feature randomness.

Stabla odlučivanja su vrlo osjetljiva na promjene na podacima na kojima su trenirana, to jest, male promjene podacima za treniranje mogu da imaju velike posljedice na model stabla odlučivanja. Random forest koristi ovo kao svoju prednost tako što dozvoljava stablima da nasumično uzimaju podatke iz seta za treniranje - proces koji se naziva Bagging.

Kod stabala odlučivanja, kada treba podijeliti čvor, uzimamo u obzir svaki mogući čvor i biramo onaj koji pravi najveću separaciju između čvorova lijevog i desnog podčvora. S druge strane, u random forest-u, svako stablo može da bira iz nasumično odabranog skupa osobina, što dodatno povećava varijacije između stabala i pomaže random forest modelu da vrati što bolji rezultat.

Random forest model nam donosi sledeće rezultate:

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 76 | 86 | 81 |
| 1 | 79 | 66 | 72 |

Tabela 19: Random forest sa Bag of Words(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 65 | 96 | 78 |
| 1 | 85 | 34 | 48 |

Tabela 20: Random forest sa bigramima(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 59 | 100 | 74 |
| 1 | 99 | 13 | 24 |

Tabela 21: Random forest sa trigramima(%)

## 4.7 Naive Bayes

Naive Bayes je familija klasifikatora koji određuju vjerovatnoću klasa na osnovu Bajesove teoreme.

$X = (X_1, X_2, ..., X_k)$ predstavlja K osobina (klasa). Y je promjenljiva koja može da uzme K mogućih vrijednosti. Sa stanovišta vjerovatnoće, $X_i$ i Y su slučajne promjenljive, koje uzimaju vrijednosti $x_i$ i y respektivno.

Da bi napravili klasifikaciju, moramo da iskoristimo X da predvidimo Y. Drugim riječima, za dato $X = (x_1, x_2, ..., x_k)$, treba da nađemo vjerovatnoću da je Y jednako y. Po Bajesovoj teoremi:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{13}$$

Ne možemo da dobijemo $P(Y|X)$ direktno, ali možemo da dobijemo $P(X|Y)$ i $P(Y)$ iz seta za treniranje.

U teoriji, nije teško naći $P(X|Y)$, međutim, što vise broj parametara raste, to postaje sve teže i teže. Kao rješenje na ovaj problem, Naive Bayes pretpostavlja da su sve osobine međusobno nezavisne.

$$P((X_1, X_2, .., X_k)|Y) = P(X_1|Y)P(X_2|Y)...P(X_k|Y) \tag{14}$$

U praksi, nije uvijek tačno da su sve osobine međusobno nezavisne, no uprkos tome, Naive Bayes i dalje daje dosta dobre rezultate i za takve probleme. U implementaciji smo koristili Gaussian i Multinomial Naive Bayes koji će biti dalje objašnjeni u narednim poglavljima.

### 4.7.1 Gaussian Naive Bayes

Tokom rada sa podacima, u GNB je pretpostavljeno da vrijednosti povezane sa svakom klasom imaju normalnu (Gausovu) raspodjelu.

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{15}$$

Pod pretpostavkom da su osobine nezavisne, model može da se trenira prosto računanjem očekivanja i disperzije za svaki podatak.

Rezultati takvog modela su prikazani u sledećim tabelama:

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 71 | 92 | 80 |
| 1 | 84 | 53 | 65 |

Tabela 22: Gaussian Naive Bayes sa Bag of Words(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 64 | 99 | 77 |
| 1 | 94 | 24 | 38 |

Tabela 23: Gaussian Naive Bayes sa bigramima(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 60 | 100 | 75 |
| 1 | 98 | 14 | 25 |

Tabela 24: Gaussian Naive Bayes sa trigramima(%)

### 4.7.2 Multinomial Naive Bayes

Multinomial Naive Bayes koristi multinomnu distribuciju da predstavi vektor osobina, u kom se čuvaju, na primjer, vrijednosti o tome koliko puta koji izraz pojavio, ili njegova relativna frekvencija.

Ako vektor osobina ima n elemenata, i svaki od njih može da uzme k vrijednosti, sa vjerovatnoćom $p_k$, onda je:

$$P(X_1 = x_1 \cap X_2 = x_2 \cap .. \cap X_k = x_k) = \frac{n!}{\prod_i x_i!} \prod_i p_i^{x_i} \tag{16}$$

Rezultati ovog algoritma su sledeći:

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 78 | 88 | 83 |
| 1 | 82 | 68 | 74 |

Tabela 25: Multinomial Naive Bayes sa Bag of Words(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 63 | 98 | 77 |
| 1 | 93 | 28 | 43 |

Tabela 26: Multinomial Naive Bayes sa bigramima(%)

| Klasa | Precision | Recall | f1_score |
|---|---|---|---|
| 0 | 62 | 100 | 76 |
| 1 | 99 | 17 | 29 |

Tabela 27: Multinomial Naive Bayes sa trigramima(%)

# 5 Zaključak

Na osnovu svih priloženih podataka, zaključujemo da se Bag of words model pokazao kao najtačniji. To je zato što je dataset relativno mali, pa je vrlo teško naći sekvence od dvije ili tri riječi koje se ponavljaju često i nose korisnu informaciju. Modeli sa bigramima i trigramima generalno pokazuju veliki recall za klasu 0, to jest, odlično svrstavaju uzorke iz test seta u klasu 0. Izuzetak ovome je slučaj kada se izvrši promjena težina klasa, kao u primjerima 4, 5 i 6.

Takođe, na osnovu podataka se vidi da svi algoritmi daju relativno slična rješenja za određeni model reprezentacije. Ipak, kao najbolji se pokazuju Support Vector Machine i Multinomial Naive Bayes.