UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

ZAKLJUČNA NALOGA

SAMODEJNO POVZEMANJE SLOVENSKIH BESEDIL S STROJNIM UČENJEM

JAKOB ŠKRLJ

ZAKLJUČNA NALOGA

2020

ŠKRLJ

UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Zaključna naloga

Samodejno povzemanje slovenskih besedil s strojni učenjem

(Automatic summarization of slovenian texts with machine learning)

Ime in priimek: Jakob Škrlj
Študijski program: Računalništvo in informatika
Mentor: izr. prof. dr. Jernej Vičič
Somentor: asist. Aleksandar Tošić

Koper, september 2020

Ključna dokumentacijska informacija

Ime in PRIIMEK: Jakob ŠKRLJ

Naslov zaključne naloge: Samodejno povzemanje slovenskih besedil s strojni učenjem

Kraj: Koper

Leto: 2020

Število listov: 42 Število slik: 31

Število referenc: 25

Mentor: izr. prof. dr. Jernej Vičič

Somentor: asist. Aleksandar Tošić

Ključne besede: Nevronska mreža, umetna inteligenca, strojno učenje

Izvleček:
Zaključna projekta naloga predstavlja implementacijo samodejnega povzemanja slovenskih besedil s strojnim učenjem. Bralcu je najprej predstavljea osnovna ideja povzemanja besedila s tradicionalnimi metodami statistike in strojnega učenja. Opisane so različne metode in pojasnilo zakaj so določene izbrane ali ne. Veliko metod ima novejše implementacije oziroma interpretacije. Bralcu je predstavljen celoten sistem in vsaka komponenta le tega, ter na koncu je predstavljeno izvajane in primeri delovanja.

Key document information

Name and SURNAME: Jakob ŠKRLJ

Title of the final project paper: Automatic summarization of slovenian texts with machine learning

Place: Koper

Year: 2020

Number of pages: 42 Number of figures: 31

Number of references: 25

Mentor: Assoc. Prof. Jernej Vičič, PhD

Co-Mentor: Assist. Aleksandar Tošić

Keywords: Neural network, artificial intelligence, machine learning

Abstract:
The final project presents the implementation of automatic summarization of slovenian texts with machine learning. The reader is first presented with the basic idea of traditional text summarization with methods from machine learning and statistics. Diffrent methods are then introduced and explained why they are included. Many mmethods have updated implementation or rather interpretation. The reader is then introduced to the implementation of the whole system and his components.

Zahvala

Zahvalil bi se mentorju izr. prof. dr. Jerneju Vičiču in somentorju asist. Aleksandru Tošiću za vso podporo, strokovno pomoč in usmeritve tako pri zaključnem delu, kot v času izobraževanja. Prav tako bi se zahvalil družini in prijateljem za vso podporo, ki so mi jo izkazali na izobraževalni poti.
Hvala!

KAZALO VSEBINE

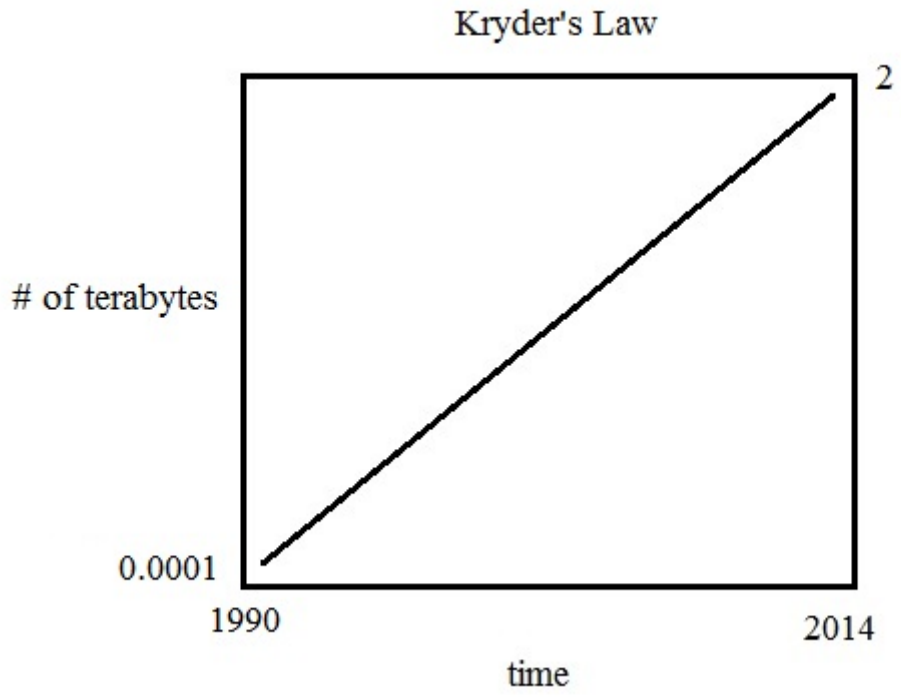
KAZALO SLIK IN GRAFIKONOV

SEZNAM KRATIC

NB	Naivini Bayes
NM	Nevronska mreža(ang. neural network)
GSM	Generalna statitstična metoda
TFIDF	(ang. term frequency–inverse document frequency) je statistična procedura, ki določa pomembnost besede
FL	Fuzzy logika
SSKJ	slovar slovenskega knjižnjega jezika, v algoritmu je uporabljen kot lexikon(shramba besed)
ASCII	ang. American Standard Code for Information Interchange
HTML	ang. Hypertext Markup Language
BNM	Biološka nevronska mreža
UNM	Umetna nevronska mreža
IF-THEN	Pravila, ki so uporabljena v procesu Fuzzy logike
CBOW	(ang. Continious Bag Of Words)
NLTK	(ang. Natural Language Toolkit) python knjižnica

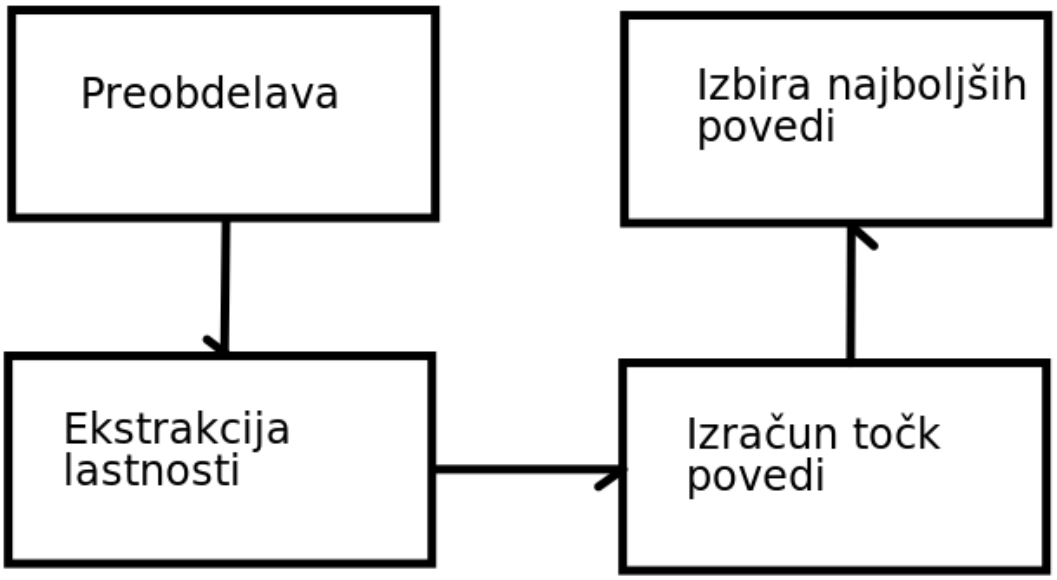
1 UVOD

Povzetek je definiran kot krajše besedilo sestavljeno iz enega ali več original besedil, ki ohranja najbolj relevante informacije in ni daljše od polovice originala.Raziskava samodejnega povzemanja se začelja že leta 1950 v IBM-ovih laboratorijih [7], vendar se je izum interneta pohitril razvoj, zaradi potrebe in rasti procesorske moči. Po nastanku interneta se je kolčina podatkov drastično povečala ([21] Kryder-jev zakon - rast shranjenih informacij je eksoponetna (slika 1)), zato je krčenje besedil na manjše, vendar berljive in informativne povzetke, prektična potreba.



Slika 1: Kryder-jev zakon, rast informacij je eksponentna, število bytov na trdem disku se podvoji vsakih 13 mesecev (skoraj vsako leto), v 1990 je bilo na trdem disku spravljenih povprečno 0.0001 Terabyta podatkov v 2014 pa 2 Terabyta, potem v letu 2020 je to povprečno 128 Terabytov podatkov. (Vir: [24])

Povzemanje besedila lahko razvrstimo v dve metodi, ekstraktivno (ang. extractive) in abstraktivno (ang. abstractive). Abstraktivno povzemanje besedila zahteva, poleg pridobivanja najpomebnejših informacij besedila, tudi generiranje novega stisnjenega besedila z enakim pomenom. Ekstraktivno povzemanje besedila izreže iz originala le najbolj informativno relevantne povedi brez, da bi jih spreminjali, tako da je besedilo še vedno berljivo.
Obstajajo dve vrsti povzemanja, induktivna (ang. inducative) in informativna (ang. informative). Induktivna predstavi glavno idejo celotnega besedila in je približno 5% besedila. Informativna nam da večje dojemanje besedila in predstavlja približno 20% osnovnega besedila. [18]
Cilj zaključne naloge je z uporabo različnih tehnik strojnega učenja, doseči dobro ekstraktivno samodejno povzemanje enega besedila.



Slika 2: Ponazoritev splošnega sistema za povzemanje besedil.

2 PREOBDELAVA

2.1 Preobdelava

Predobdelava podatkov vključuje, pridobitev podatkov (ang. scrapping), ko gre za zbiranje podatkov namenjenih za trening našega modela,tokenizacijo in razčlenjevanje (ang. parsing) dobljenega besedila. Razčlenjevanje odstrani posebne znake, ki so nastali med procesom pridobivanja podatkov, kot so simboli, ki niso ASCII (ang. American Standard Code for Information Interchange, je standart komunikacije, ki omogoča da z 8 biti (255 kombinacij) predstavimo 127 različnih znakov, kar je zelo uporabno v elektronski komunikaciji) ali pa del kakšne druge sintakse npr. HTML (ang. Hypertext Markup Language, je standardni jezik za oblikovanje dokumentov, ki bodo prikazani na internetu, omogoča razlikovanje (v naslove in podnaslove itd.) in pozicioniranje (tabele, odstavki, liste itd.) besedila) značke. Po razčljenjevanju se začne tokenizacija, ta razdeli odstavke v individualne povedi. Po tokenizaciji podatkov odstranimo stop besede (ang. stop words), to so besede, ki se v besedilu največkrat pojavijo, npr. 'v', 'in', 'na' itd. to so večinoma funkcijske besede, ki ne prinašajo pomena v besedilo ampak so večinoma uporabljene za formo (seveda delno tudi spreminjajo pomen).

2.2 Ekstrakcija lastnosti

Ekstrakcija lastnosti je proces, kjer vsako poved pretvorimo v n-dimenzionalni vektor z vnaprej definiranimi lastnostmi za posamezne dimenzije. V temu delu se bomo omejili na 8 lastnosti, te so: podobnost s naslovom, pozicija povedi, teža izrazov (relevanca terminov), dolžina povedi, tematičnost, pravilni samostalniki, podobnost med povedmi in numerični podatki. Vektor povedi:

$$pv_i = (v1, v2, v3, v4, v5, v6, v7, v8)$$

kjer je i pozicija povedi v besedilu.

2.2.1 Podobnost z naslovom

Podobnost s naslovom je velikost ujemanja besed v povedi in naslovu. Izračuna se s kosinusno podobnostjo. kosinusno podobnost dobimo tako, da spremenimo dve besede v n-dimenzionalna vektorja in zračunamo kot med njima, če je kot 0, ni podobnosti, če je kot 90 je popolna podobnost, da dobimo kosinusno podobnost povedi, naredimo to za vse besede v povedi in seštejemo.

$$cosine(p_i, p_j) = \frac{\sum_{t=1}^{N} w_{it} w_{jt}}{\sqrt{\sum_{t=1}^{n} w_{it}^2} \sqrt{\sum_{t=1}^{n} w_{jt}^2}}$$

w_{jt} in w_{it} so frekvence besed v povedi p_j in p_i s vreče besed.

$$v1 = cosine(poved, naslov)$$

2.2.2 Pozicija povedi

Pozicije povedi so običajno pomembne, saj imamo ljudje predvidljiv način pisanja, npr. na začetku napišemo najbolj pomembne stvari ali pa jih na koncu povzamemo, oziroma napišemo zaključek. Pozicija povedi je zelo preprosta, avtor sam določi pomembnost povedi. Prvih pet povedi je običajno izbira, prva poved ima vrednost 1 = 5/5, druga 4/5, itd. (naslednje imajo pa obliko (6-pozicija)/5, 6 ker prva poved ima vrednost 1 in ne 0 (0 je pozicija povedi, ki predstavlja naslov, to preskočimo), povedi, ki niso vključene izbrane s to lastnostjo imajo obliko 0/5 tj. p_1=5/5,...p_5=1/5,p_5=0/5,p_5=0/5,). Lahko tudi, izbereno samo prvo in zadnjo poved (p_1=1,p_n=1), ali pa kombinacijo prvih nekaj povedi in zadnjih nekaj.

$$v2 = \frac{kolikoMestStejemo - pozicija}{kolikoMestStejemo}$$

2.2.3 Teža izrazov

Težo izrazov (ang. term weight), izračunamo s pomočjo metode TFIDF (ang. Term Frequency–Inverse Document Frequency), kratica stoji za TF*IDF, tf_{ij}-ferkvence termina(ang. term frequency) nam da mero pomembnosti besede(termina) i v dokumentu j, idf-obratna frekvenca dokumenta(ang. inverse document frequency) nam pove splošno pomembnost termina. Primer, visoka frkvenca besede politika v dokumentu pomeni, da je za specifičen dokument pomembna, visoka obratna frekvenca te besede v vseh dokumentih pa pomeni da ni zelo pomembna na sploh.
Za zbirko/korpus k besed in množico dokumentov D, je frkvenca trminov

$$tf_{ij} = \frac{n_{ij}}{Max(\sum_{k=1}^{T} n_{kj})}$$

kjer za poved i iz dokumenta j, je n_{ij} število besed v dokumentu.

$$idf_i = log\frac{|D|}{\sum(d_j : d_j \in D)}$$

kjer je $|D|$ število dokumentov v glavni zbirki, deljeno s številom dokumentov, ki vsebujejo izraz. $TFIDF = tf_i * idf_i$

$$v3 = \frac{\sum_{i=1}^{k} w_i(poved)}{Max(\sum_{i=1}^{k} w_i(poved_i^N))}$$

kjer je k število izrazov v povedi.

2.2.4 Dolžina povedi

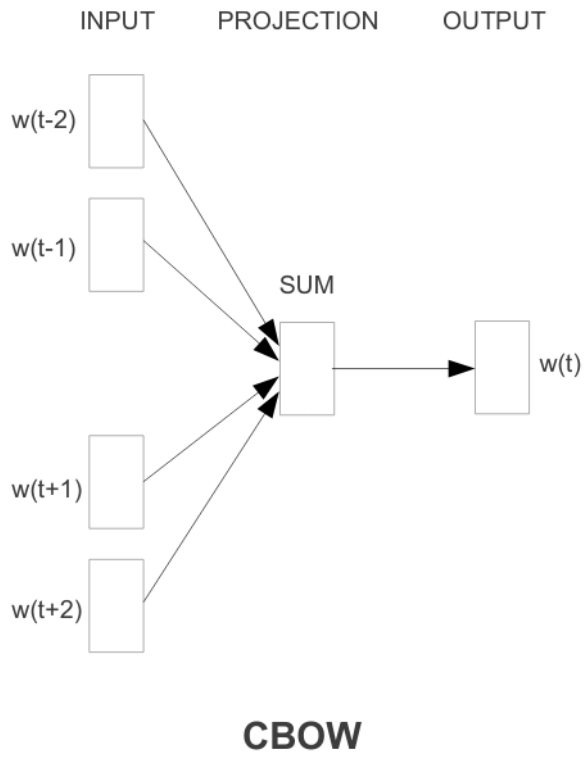
Krajše povedi običajno vsebujejo manj vsebine ali pa so neuporabni dodatki, ki ne sodijo v povzem, npr. avtorjev subjektivni vnos "nekaj. To pa res ne gre tako! nekaj". Glede na domeno problema bi lahko tudi prioritizirali krajše povedi, saj nočemo da je povzetek predolg, vendar takšen postopek zahteva več računanja na drugih mestih.

$$v4 = \frac{len(poved)}{len(najdaljsaPoved)}$$

kjer je len() funkcija, ki vrne dolžino povedi, denominator bi lahko napisali tudi kot Max(len(poved)), ki vrne najdaljšo možno dolžino povedi.

2.2.5 Tematičnost

Za iskanje tematičnosti besed uporabimo CBOW (ang. Continious bag of words) [11]. CBOW je 'plitka' nevronska mreža, ki se iz konteksta nauči iskat ciljno besedo (beseda na sredini konteksta), kontekst sta običajno dve besedi levo in desno od iskane besede. Trening te 'plitke' mreže je trajal 48 ur (2 dni), čeprav je plitka(samo en sloj nevronov) je bila količina podatkov, ki jih je morala obdelati enormna. Vse skupaj v naboru podatkov za treniranje te mreže je 55000 povedi, če vsaka vsebuje povprečno 10 nestop besed je to 10-4 (besede na rabu ne moremo v kontekstu predelat) je to 6*55000 pregledov, skozi 50 epoh (kolikokrat bomo podatke ponovno spustili skozi mrežo), kar je precej dela.



Slika 3: Model CBOW mreže, je zelo plitka mreža z enim samim skritim nevronom. (Vir: [11])

Ko dobimo iskano besedo iz konteksta, jo primerjamo z orginalom s uporabo kosinusne podobnosti, ki nam vrne razdaljo med besedama, potem samo seštejemo te razdalje in jih delimo s dolžino celotne povedi.

$$v5 = \frac{steviloTevmatskihBesed}{dolzinaPovedi}$$

kjer je SteviloTevmatskihBesed vsota vrednosto, ki jih CBOW proizvede.

2.2.6 Pravilni samostalniki

Pravilni samostalniki so imena oseb, krajov, dogodkov itd. Poved, ki vsebuje pravilni samostalnike je običajno zelo pomembna. Da ugotovimo, če je beseda pravilni samostalnik, jo poskusimo poiskat, s Hashmap (to je podatkovna struktura, ki omogoča hitro poizvedbo rezultatovm ki jih iščemo) obliko SSKJ (Slovar Slovenskega Knjižnega Jezika) in če jo ne najdemo je najvrjetneje pravilni samostalnik.

$$v6 = \frac{steviloPravilnihSamostalnikov}{len(povedi)}$$

2.2.7 Podobnost med povedmi

Podobnost med povedmi merimo s cosine() funkcijo. Ideja je da podobne povedi združimo (ang. clustering), saj podobne povedi govorijo o podbni temi besedila, ki je najvrjetneje poglavita.

$$v7 = \frac{podobnost}{Max(podobnost)}$$

kjer je $podobnost = \sum_{i=0}^{d} \sum_{i=0}^{d} cosine(poved_n, poved_i)$ d kot dolžina posameznega članka (število povedi, ne seštevek znakov), in Max(podobnost) največja možna podobnost v celotnem članku med dvema povedima.

2.2.8 Numerični podatki

Numerični podatki predstavljajo datume, vsote denarja, količine itd. Običajno so povedi s takšnimi podatki zelo pomembne, npr. "Študent mora potrdilo odati do 20.aprila".

$$v8 = \frac{številoNumPodatkov}{len(povedi)}$$

3 METODE SAMODEJNEGA POVZEMANJA

3.1 Uvod

Vse metode povzemanja poskušajo na podlagi vektorja določiti točke oziroma rank povedi, potem so povedi enostavno sortirane in glede na tip povzemanja(induktivno (5%) ali informativno (20%)), vzamemo najbolše. Za izbiro najboljših povedi je veliko metod iz statistike, linearne algebre (npr. linearna regresija), teorije grafov, strojnega učenja (npr. skriti Markov model (ang. Hidden Markov model (HMM))) itd., tukaj bodo pojasnjene le uporabljene metode, generalna statistična metoda, naivni Bayes, fuzzy logika in nevronska mreža.

3.2 Metode

3.2.1 Generalna statistična metoda

GSM [15] je samo vsota vseh vektorjev v povedi. Je zelo preprosta metoda in ne vzame drugih povedi v obzir ko vrne rezultat. Je najbolj intuitivna metoda, saj poskušamo oceniti vrednost povedi, ampak včasih manj vredne lastnosti, pokrijejo boljše lastnosti, kar pomeni, da izbrana beseda, je zelo dobra, lahko pa obstaja beseda z veliko pravilnimi samostalniki, ki govori o pomembnih ljudeh ali lokacijah in je posledično zanemarjena.

$$gsm_i = v1 + v2 + v3 + v4 + v5 + v6 + v7 + v8 + v8$$

3.2.2 Naivni Bayes

Iz nekaj besedil sestavimo modela, ki ga potem lahko uporabimo za nadaljno sklepanje. Za izbiro povedi niso potrebne vse vrednosti vektorja, lahko sami izberemo, najboljše lastnosti in na temu baziramo model. Ko imamo model preračunamo vse druge vrednosti.

$$nb_i = P(p \in PV|v1, v2...vk) = \frac{P(v1, v2...vk|p \in PV)P(p \in PV)}{P(v1, v2...vk)}$$

Za vsako poved p preračunamo verjetnost, da bo bila vključena v povzem PV, glede na k lastnosti na katerih odločamo [10]

$$\text{Končna verjetnost} = \frac{\text{Prejšna verjetnost} * \text{verjetnost}}{\text{Dokazi}}$$

Slika 4: Bayes-ov postopek (zgoraj napisan v matematični obliki) lahko opišemo tudi na ta način, kjer je končna verjetnost(ang. posterior probabilty), produkt prejšne verjetnosti (ang. priori) in same verjetnosti dogodka (ang. likelihood), vse skupaj deljeno z dokazi (ang. evidence).

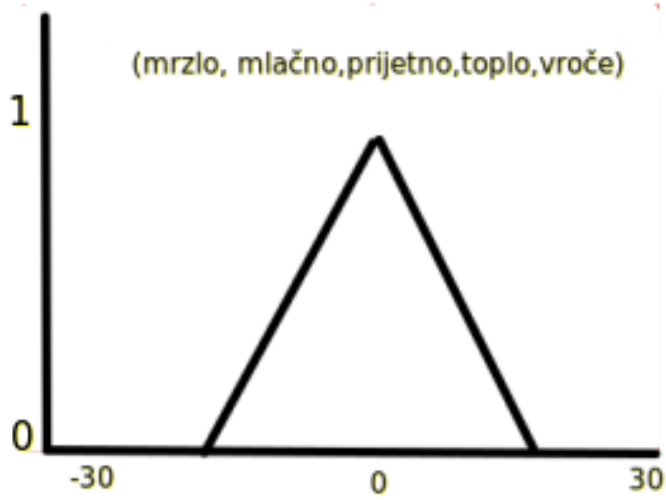
Sama implementacija bayesa je v podpoglavju 4.2.2.

3.2.3 Fuzzy logika

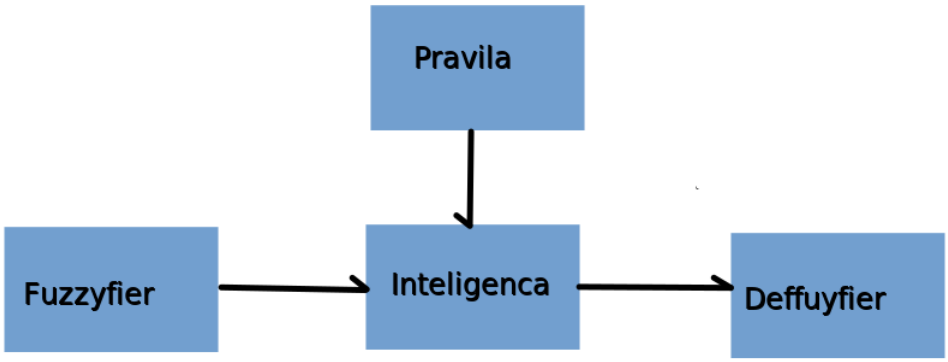
Fuzzy Logika(FL) s pomočjo funkcije pripadnosti [2] (ang. membership function) pretvori dvoumne vrednosti v uniformno obliko (linguistično običajno), npr. temperatura od -5 do 25, je samo 30 različnih števil, FL pa pretvori te vrednosti v mrzlo, mlačno, prijetno, toplo in vroče. Deluje tako da postavi vrednosti na funkcijo pripadnosti, pridobi linguistične vrednosti vektorja in z IF-THEN določi vrednost povedi. [14] Primer funkcije pripadnosti, triangularna funkcija:

$$memFun(x, a, b, c) = max(min(\frac{x-a}{b-a}, \frac{c-x}{c-b}), 0)$$

kjer so a in c, točke kjer se trikotnik(slika 5) dotika x osi in b predstavlja višino trikotnika.



Slika 5: Triangularna funkcija pripadnosti.



Slika 6: Oblika Fuzzy inferenčnega stroja, model prirejen po citatut [8].

Prvi korak(Fuzzyfier), s pomočjo funkcije pripadnosti(slika 5), spremeni vrednosti v svoje se pirpadajoče razrede (npr. iz 0.2687 postane 0.2 iz 0.38 postane 0.4), ta oblika je zelo dobra za učenje NM. Drugi korak (Inteligenca) s pomočjo IF-THEN pravil iz uniformnih vrednosti oziroma razredov določi vrednost povedi oziroma skupen razred, ki bo pozneje ciljana vrednost NM. Tretnji korak nam dodeli vrednosti 0 in 1 za povedi, ki so izključene oziroma vključene v povzetku, tako da lahko tudi z FL generiramo povzetek, za testiranje, ali če tako želimo. IF-THEN pravila v kontekstu tega dela pomenijo le odločitev, katera poved naj je "dvignjena" nad ostale, to storimo s preprostimi if stavki po avtorjevem okusu, tukaj smo se odločili da so lastnosti teža terminov(podpoglavje 2.2.3), tematičnost(podpoglavje 2.2.5), pravilni samostalniki (podpoglavje 2.2.6) in Bayes (podpoglavje 3.2.2) najbolj pomembne, zato jih dodelimo več točk v odvisnosti, koliko teh lastnosti poved vsebuje, ta proces je prikazan na sliki 7

```
importantFeatures = (sentence_vector[2]+sentence_vector[3]+
            sentence_vector[4]+sentence_vector[9])

lessImportantFeatures =(sentence_vector[0]+sentence_vector[1]
            +sentence_vector[5]+sentence_vector[6]+
            sentence_vector[7]+sentence_vector[8])

# Important features get promoted
if(importantFeatures > 18):
    endScore = importantFeatures+100+lessImportantFeatures
elif(importantFeatures > 9):
    endScore = importantFeatures+30+lessImportantFeatures
elif(importantFeatures > 3):
    endScore = importantFeatures+10+lessImportantFeatures
else:
    endScore = importantFeatures+lessImportantFeatures
```

Slika 7: Diskriminacija do manj zaželjenih povedi(ang. less imporant features) in pozitivna diskriminacija do bolj pomembnih povedi (ang. important features), lastnosti od 0 do 7 so vse lastnosti omenjene v poglavju 2.1, 8 predstavlja GSM (podpoglavje 3.2.1) in 9 predstavlja Bayes (podpoglavje 3.2.2), vse skupaj 10 lastnosti.

Da dobimo končne točke posamezne povedi, seštejemo vse njene spremenjene vrednosti skupaj in jih sortiramo od največje do najmanjše. Na koncu ko imamo točke vsake povedi oziroma njen rank ji dodelimo razred, po mejah določenih na sliki 16(ni nikoli enakomerna razdelitev, kar je razloženo v podpoglavju 4.2)

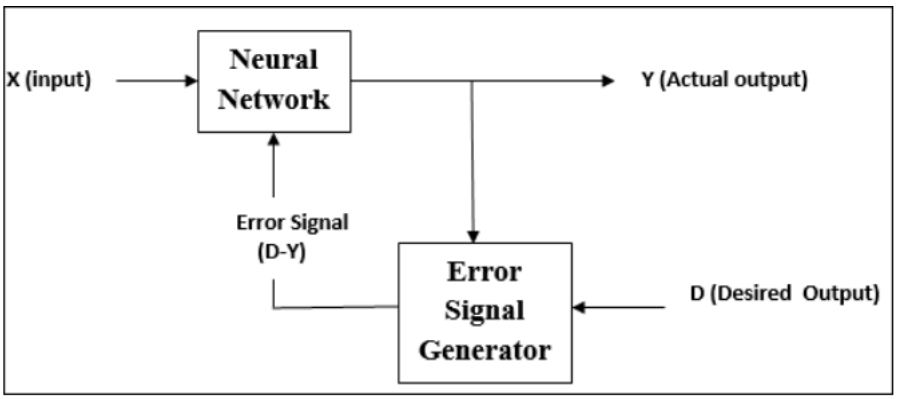
3.2.4 Nevronska mreža

NM je modelirana po človeških možganih oziroma njihovemu delovanj [12] bolj natančno po nevronih in njihovih povezavah v možganih (slika 8), lahko bi celo rekli, da smo odkrili NM skozi študij človeških možganov. NM je sposobna se 'učiti' in prilagoditi svoje delovanje glede na podatke in rezultate, ki jih iščemo. Sestavljena je iz treh komponent vhodni nevroni, skriti nevroni in izhodni nevroni oziroma v najbolj enostavnem primeru samo iz vhodnih in izhodnih, taki mreži rečemo enoslojna podajoča mreža(ang. single layer feedforward network). Vhodni nevroni in izhodni nevroni se v procesu učenja ne spremenijo in so uporabniku vidni. Skriti nevroni so jedro umetne inteligence, običajno so postavljeni v več slojev. Vsebujejo notranjo stanje in aktivacijsko funkcijo, ko nevron dobi vhodne podatke jih združi z notranjim stanjem in jih pošlje čez aktivacijsko funkcijo, tako proizvede izhodne podatke, ki jih pošlje naslednjemu sloju. Vsak nevron je povezan z vsemi nevroni predhodnega in naslednjega sloja, vsaka povezava ima svojo utež, ki predstavlja pomembnost nevrona, ki ga povezuje. Uteži prilagaja funkcija povratnega širjenja(ang. backpropagation func-

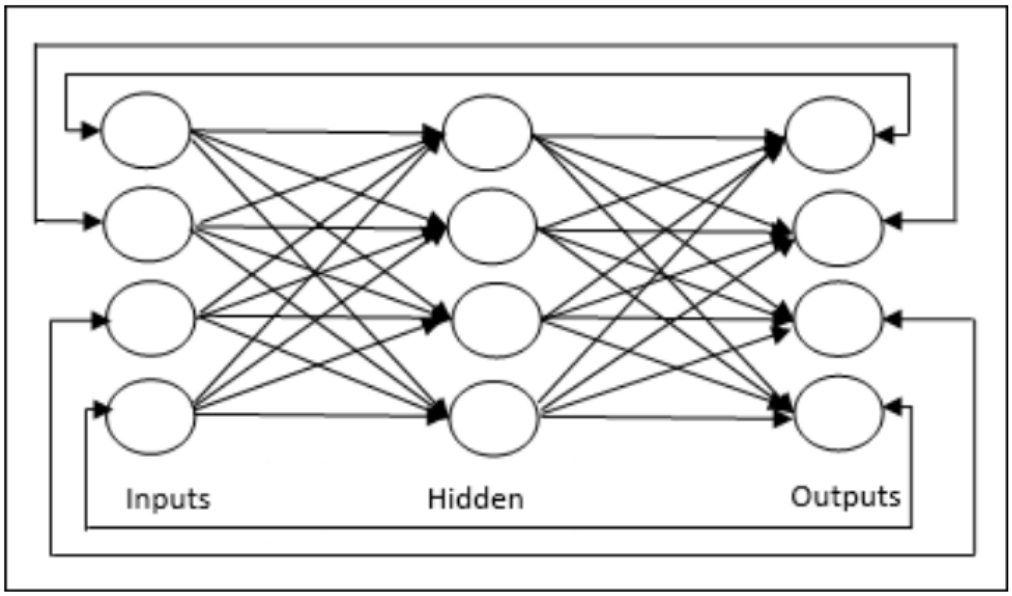
tion). NM je zelo dobra za iskanje korelacij v zaporedju podatkov, kot so 'nevidna' pravila v besedilu iz povedi v naslednjo poved. Obstaja več vrst nevronskih mrež, za različne vrste podatkov in z različnimi pravili učenja, s podajočimi (ang. feed foward) ali ponavljajočimi (ang. recurrent) nevroni. Učenje NM lahko poteka na več načinov samodejno, nadzirano(ang. supervised) ali hibridno (samodejno in nadzirano hkrati, to je precej nov pristop), učenje lahko poteka tudi v obliki tekmnovanja, več NM se primerja ena proti drugi. [20]

Biological Neural Network BNN	Artificial Neural Network ANN
Soma	Node
Dendrites	Input
Synapse	Weights or Interconnections
Axon	Output

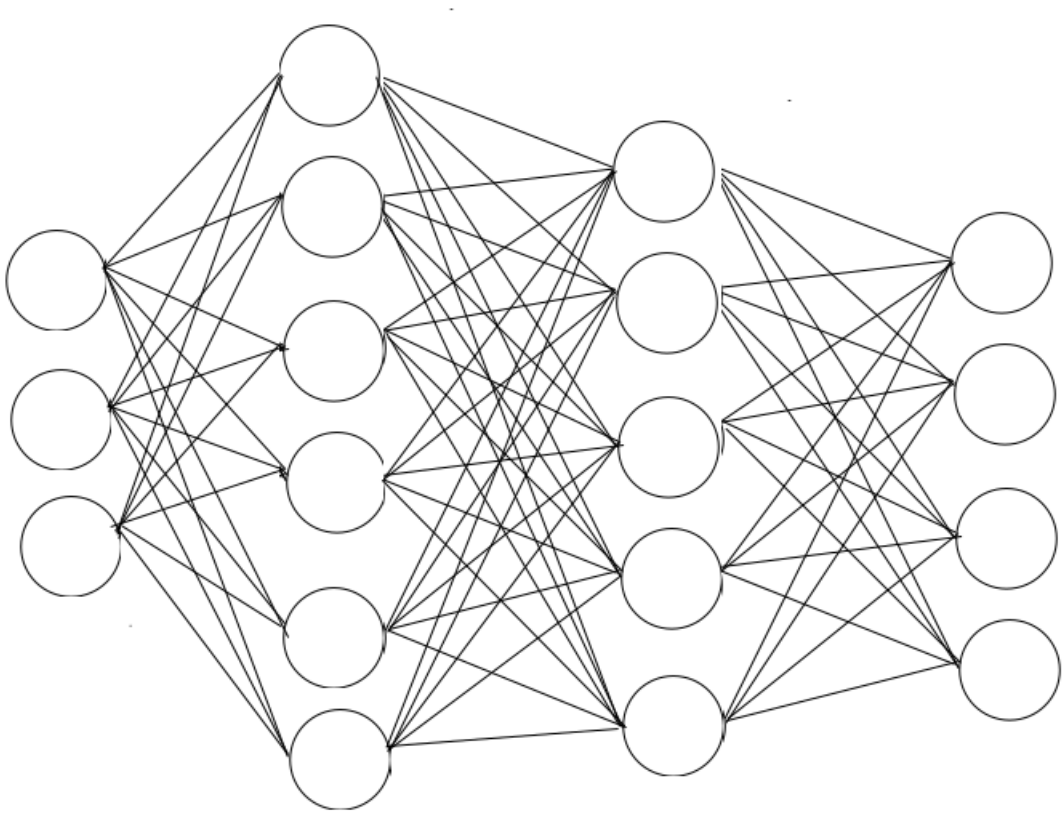
Slika 8: Primerjava med Biološko nevronsko mrežo[BNM] (ang. Biological Neural Network[BNN] oziroma možgani) in umetno nevronsko mrežo (ang. Artificial Neural Network[ANN]). (Vir: [20])



Slika 9: Učenje NM, kjer agent(običajno človek) dodaja stimulacijo. (Vir: [20])



Slika 10: NM, ki je popolnoma zaprta, izhodni nevroni vračajo rezultat nazaj v vhodne, precej zanimiv primer ponavljejoče mreže. (Vir: [20])



Slika 11: NM, ki je uporabljena v implementaciji metodologije, v večji obliki. Oblika je piramida, ki naj bi bila najboljša za učenje.

4 IMPLEMENTACIJA

4.1 Preobdelava

Vsa implementacija je storjena v programskem jeziku python!

4.1.1 Pridobivanje podatkov in preobdelava

Za pridobivanje podatkov poskrbi program, ki odpre spletno stran 24ur.com/arhiv in se članek po članek sprehodi po spletni strani, zbira in arhvira podatke oziroma jih pošlje naprej v obdelavo. Prva implementacija spletnega pobiralca (ang. scrapper) je bila napisana s python knjižnico Requests, vendar ta implementacija ni več delovala, ko so spremenili spletno stran zaradi epidemije Koronavirusa (vrjetno so pričakovali več prometa, kot običajno), zato je bila druga implementacija storjena z bolj počasno metodo, Selenium, je vmesnik, ki simulira brskalnik in fizične po njem, je zelo počasen in omejen z uporabnikovo internetno povezavo in strukturo same strani. Pobiralec (ang. scrapper) mora zbrati zadosti podatkov, zato da lahko poslednje funkcionalnosti delujejo, kot so TDIDF in CBOW metoda, saj obe proceduri zahtevajo veliko količino podatkov, da proizvedejo marginalno dobre podatke. Preobdelava podatkov se začne s tokenizacijo (ang. tokenizatioin), ki iz odstavkov in paragrafov naredi ločene povedi, medtem pa razčlenjevanec (ang. parser) počisti besedilo vseh nazaželjenih simbolov, ki jih je pobiralec proizvedel, kot so HTML značke in simboli drugačnega kodiranaja npr. 0x08 za konec vrstice (ang. endline). Naslednji korak preobdelave je odstranjevanje stop besed, to je storjeno z iteracijo vseh besed skozi tabelo stop besed. Lista slovenskih stop besed, ki je bila pridobljena na internetu vsebuje 450 stop besed [3].

več tem naj tako pri ni sem kar bi ali od iz jih to o be do ne po kot z tudi s bo so ki se pa za da na in v če zelo ker za ko kar pri

Slika 12: Primer stop besed.

Spletni Pobiralec (ang. scrapper) - Requests (in BeatifulSoup) in Selenium

Spletni pobiralec je program, ki odpre ali prenese spletno stran in iz nje izvleče podatke, ki jih iščemo, lahko so to slike, besedilo ali bolj abstrakne strukture kot so tabele iz različne druge HTML značke (in strukture, ki jih te predstavljajo). V python programerski paradigmi so znane dve implementaciji oziroma orodja s katerimi to dejavnost počnemo, to so knjižnici Requests in Selenium.

Requests deluje tako da prenesemo celotno strukturo strani, primer ilustriran s python kodo:

```
indexPageURL = 'https://www.24ur.com/arhiv/novice?stran=1'
mainPageClien = requests.get(indexPageURL, stream=True)
while(mainPageClien.status_code != 200):
   mainPageClien = requests.get(indexPageURL)
```

V temu primeru prenesemo spletno stran
`https://www.24ur.com/arhiv/novice?stran=1`
ta pride do nas v obliki HTML čistega besedila (ang. plain text), s katerega moramo sami razbrati informacije ki jih iščemo. Primer HTML besedila/strukture (HTML značke so znotraj <> 'oklepajev'):

```
<div class="card__thumb">
  <div class="card__label␣label␣label--201␣label—card">
  Slovenija </div>
  <div class="media␣card__img">
    <picture class="media__object">
      <source media="(min-width:␣1200px)"
       srcset="https://images.24ur.com/
␣␣␣␣␣␣␣␣media/images/300x180/
␣␣␣␣␣␣␣␣Apr2020/399deb6c35_62413999.jpg?v=091c">
    </picture>
  </div>
</div>
```

Zato, da lahko navigiramo čez takšne strukture uporabimo HTML razčlenjevalnik (ang. parser), to je komponenta (program), ki podatke razdeli na manjše elemente za enostaven prevod v drug jezik (bolj človeku berljiv). Python seveda ponuja zelo dobro knjižnico za razčlenjevanje HTML, ki se imenuje BeautifulSoup, ta nam omogoča da iz dokumenta pridobimo kakršno koli strukturo (paragraf, naslov, slika itd.) in njene atribute (url, ime, lokacijo itd.).

Selenium je veliko bolj primitivna metoda spletnega pobiranja (ang. scrapping), za razliko od Requests in je, zato redkeje uporabljena, vključno v tem delu, saj je avtor nerad implementiral to zelo časovno zahtevno metodo, ki je za zbiranje 150 strani člankov, to je 150*20(20 člankov na eno stran) = 3000 člankov, zahtevalo 20 ur delovanja.
Primer python selenium kode:

```
driver = webdriver.Firefox()
driver.get("https://www.24ur.com/arhiv/novice?stran=1")
time.sleep(1)
elements = driver.find_elements_by_xpath("/html/\
body/onl-root/div[1]/div[3]/\
div[2]/onl-archive/div/div/\
div/main/div/div[2]/a")
elements[0].click()
```

Prvi python stavek 'driver = webdriver.Firefox()' ustvari simulacijo brskalnika. v temu primeru je to firefox, to simulacijo lahko uporabnik vidi, vendar mu ni potrebno početi nič saj se vse samo izvaja. Drugi python stavek 'driver.get("https://www.24ur.com/arhiv/novice?stran=1")' odpre v simuliranem brskalniku podan url, v temu primeru je to to www.24ur.com/... in omogoči simulaciji navigiranje ali interakcijo s stranjo. Tretji python stavek 'time.sleep(1)' predstavlja ukaz 'počakaj eno sekundo' in je tukaj, ker je internet ali brskalnik včasih počasen in mora simulacija počakati nekaj časa preden začne interakcijo, saj če prične interakcijo preden se karkoli naloži, nam javi napako (zaradi takšnega čakanja je celoten proces zelo zelo počasen). Četrti python stavek 'elements = driver.find_elements_by_xpath("/html/ itd.' poišče določeno html značko (tukaj je to storjeno s xpath-om, to je kot naslov oziroma lokacija HMTL značke znotra HTML hiearhije), v temu primeru sidro (ang. anchor, v HTML-ju je uporabljena da drži url do povezave). Zadnji python stavek 'elements[0].click()' simulira klik, kot da bi človek kliknil, vendar je to vse samodejno. Zaradi čakanja (da zagotovimo delovanje) je ta metoda zelo zelo počasna, v temu primeru je bila samo ena sekunda in že to je ogrmno če se mora 3000-krat zgoditi, ampak v sami implementaciji je več takšnih čakanj, kar pomeni, da se zelo hitro nabere veliko sekund čakanja.

Python pomožne knjižnice(NLTK in Pickle)

NLTK (ang. Natural Language Toolkit) je python knjižnica, ki ponuja orodja za obdelavo naravnih jezikov. Seveda je najbolje implementirana angleščina, vendar podpira veliko jezikov, preostali jeziki specifično evropski so precej dobro podprti. Čeprav knjižnica ponuja veliko orodij bomo uporabili le 'tokenizer', to orodje nam pomaga ločiti povedi iz odstavkov oziroma celotnega besedila, rezultat je lista povedi, glede na to da imamo opravka s slovenščino, ki je zelo težek jezik, moramo včasih malo 'pomagati' temu postopku. Inicializacija tokenizer-ja:

```
tokenizer = nltk.data.load('tokenizers/punkt/slovene.pickle')
```

Druga python podporna knjižnica(ki jo tudi NLTK v zgornjem primeru koristi) je Pickle. Pickle nam omogoča da shranimo celotne podatkovne strukture, npr. če imamo listo in jo želimo shraniti na disk, brez knjižnice Pickle moramo to storiti tako, da vsak element ločeno zapišemo in ravno tako, da ga ločeno preberemo ob drugi potrebi. Ampak s knjžnico Pickle lahko shranimo listo v celoti in jo na drugi strani enako preberemo.
Pickle branje:

```
File_object = open(r"dataset_raw/\
articles_everything_pickled","rb")
data_raw = pickle.load(File_object)
File_object.close()
```

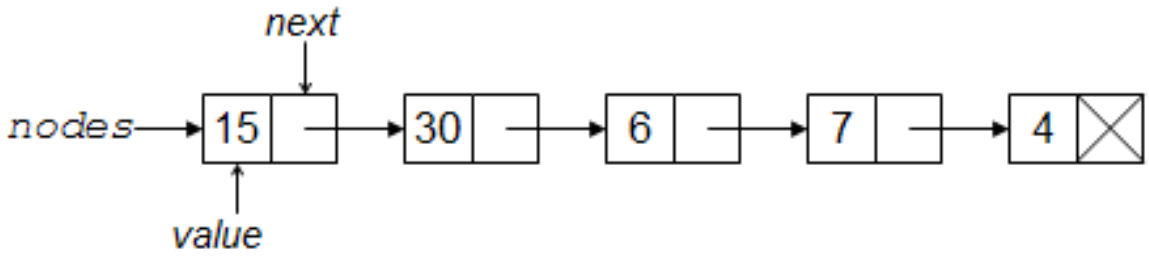
kjer "rb" pomeni 'read binary', dokumenti so shranjeni v bitnem zapisu in ne v čistem besedilu(ang. plain text).

4.1.2 Ekstrakcija lastnosti

Ekstrakcija lastnosti poteka po prej omenjenih postopkih iz poglavja 2.1. Tematičnost besed in teža besed sta si zelo podobna postopka, razlika je da tematičnost je bolj 'lokalna' in meri razliko znotraj enega besedila, medtem ko je teža 'globalna' in meri razliko med TFIDF vrednostmi vseh besedil. Procedura iskanja pravilnih samostalnikov zahteva pregled SSKJ. SSKJ je strukturiran, kot hashmap, to omogoč poizvedbo iskanega podatka s časovno zahtevnostjo O(1), to pomeni da podatkov preden jih poiščemo ni potrebno sortirati ali kakšne druge strukturne manipulacije, vse, kar je potrebno za delovanje je, da pretvorimo SSKJ iz besedila (ang. plaintext) v pythonsko strukturo slovar (ang. dicionary), ki ima v temu primeru obliko npr. SSKJ['gora']=1. Da izvemo če je beseda v SSKJ ali ne, preprosto v pythonske načinu vprašamo if(not SSKJ['Jože']), kar je v temu primeru res, saj pravilni samostalnik 'Jože' definitivno ni del SSKJ.

Python - lista in slovar (ang. dictionary ozitoma hashmap)

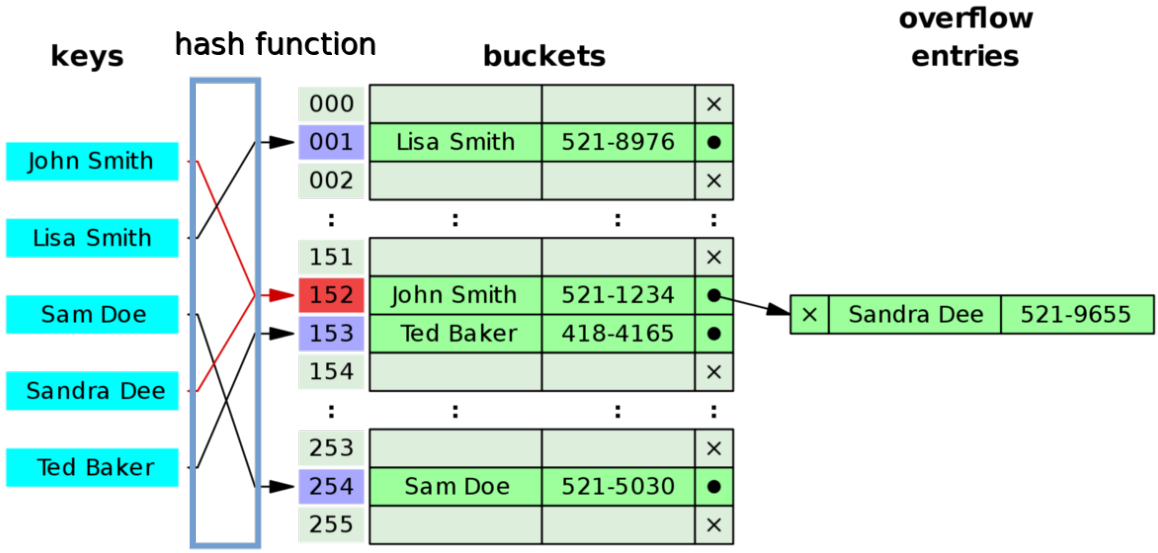
Lista in slovar so dve najbolj uporabljeni programski strukturi v temu delu, zato ju moramo predstaviti in razložiti, da lahko dojamemo delovanje celotnega sistema.
Listi, kot pythonski strukturi, drugače rečemo tudi povezana lista (ang. linked list), iz slike 14, je razvidno zakaj. Celice so sestavljene iz para, informacije in povezave (lokacije naslednje celice), vsaka celica je povezana s svojima sosedoma in nič več kot to. Takšna struktura je najbolj preprosta in ne povzroča nobenih zapletov, vendar je precej počasna saj, se lahko pomikamo le naprej in nazaj, to je zelo počasno če hočemo dobiti neko informacijo, ki je na sredini ali na koncu. Rečemo, da ima metoda O(n) zahtevnost, kar pomeni, da za n elementov v tabeli bomo morali povprečno narediti n operacij (preskokov med celicamoi) za določeno operacijo (vstavljanje, brisanje ali poizvedba elementa).



Slika 13: Slika povezane liste, kot ime pravi, je ta struktura povezava več celic(ki je sestavljena iz informacije in povezavo na drugo celico) v vrsto, ki se razteguje od prve celice do n-te. (Vir: [22])

Druga pythonksa struktura, ki o moramo opisati je slovar oziroma, kot podatkovna struktura znan pod imenom hashmap. Hashmap je zelo hitra metoda (O(1) povprečna zahtevnost operacij brisanja,vstavljanja in poizvedbe, v primerjavi z listo, ki ima O(n)), vendar implementacija ni tako preprosta in zahteva veliko prostora. Hashmap deluje tako, da za vsak podatek, ki ga vstavimo generira 'naslov' (ang. adress) s hash funkcijo. Hash funkcija nam vrne skoraj naključno vrednost, ko nekaj vstavimo notri, ampak cilj ni da so vrednosti naključne vendar drugačne, ker je funkcija (f(x) = y) bo enak x vedno proizvedel enak y. To nam poda intuitivno idejo o ti strukturi, če imamo funkcijo, ki za neko vrednost proizvede vedno enako vrednost ali 'naslov', lahko na ta način povežemo te vrednosti s naslovom, prek te funkcije. Funkcija je lahko nekaj preprostega npr. $f(x) = (x * 5/8)^2 \ mod \ 5$, vendar se pojavi prvi problem, pri veliki količini x-ov se bodo naslovi, ki jih funkcija proizvaja začeli ponavljati. Ta problem je rešen tako, da kjer se naslovi prekrivajo iz njih sestavimo povezano listo, se pravi, ko pogledamo takšen naslov nazaj ne dobimo le specifične vrednosti ampak listo vrednosti. Če je naša hash

funkcija zadosti dobra je malo naslovov, ki se prekrivajo in nam to zagotavlja hitro delovanje (operacije poizvedbo, vstavljanje in brisanje). Slika opisane strukture:



Slika 14: Slika hashmap strukture, kjer lahko vidimo celotno delovanje, seveda moramo besede(v temu primeru imena) spremeniti v numerično obliko in jih poslati čez hash funkcijo, ki potem proizvede naslov do predala, kjer je vrednost spravljena, in če je konflikt naslovov (dva ali več kažejo na isti predal) postane celoten predal povezana lista. Na sliki, ključi(ang. keys) so v tem primeru imena, ki jih v numerični obliki pošljemo skozi hash funkcijo (ang. hash function, kvadrat med povezavami) in postanejo naslovi predala oziroma v temu primeru vedra, v primeru konflikta (npr. John Smith in Sandra Dee na sliki) postane vedro povezana lista, kjer prvi naslov, kaže na drugega. (Vir: [25])

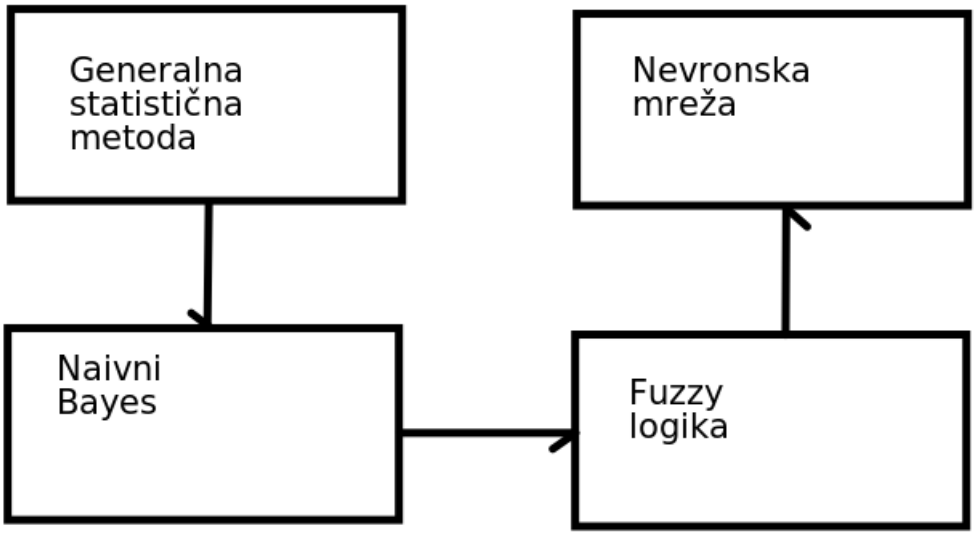
4.2 Implementacija metod

4.2.1 Metodologija

Za ta strojni povzemalnik bomo uporabili skupek metod in vektorju povedi dodali še vrednosti, ki jih vrenjo metode, ta vektor bo na koncu uporabljen za učenje preproste nevronske mreže. Seveda pa vsaka metoda po cevovodu sistema generira svoj povzetek, tako da se lahko na koncu primerjajo. Vektor po obdelavi:

$$kv_i = (v1, v2, v3, v4, v5, v6, v7, v8, gsm, nb, fl)$$

kjer je kv_i končni vektor i-te povedi



Slika 15: Cevovod sistema, po katerem se premika vektor povedi, pri vsakem postopku(na sliki ponazorjeno kot kvadrat) mu je dodana nova vrednost, ki jih na koncu predela nevronska mreža.

GSM, NB in FL predstavljajo vsaka svojo pristranskost (ang. bias) (količina povedi, ki naj bi bile v pozetku), v učenju modela na sliki 21 so vrednosti sledeče: GSM - 40%, NB - 60% in FL - 40%. NB je prekomirno zastopan, ker hočemo motivirati NM, da se nauči dobre kompresije, ki jo proizvede NB na sliki 31. FL na koncu proizvede tudi razrede(0,1,2,3), s katerimi učimo NM, in jih 'pomeša'(slika 16),to stori s funkcijo naključnosti (ang. random function), ki proizvede 'meje' (slika 16) med razredi, da ni vedno uniformna distribucija, s tem preprečimo prekomirno nagnjenje v prid določenim rezultatom (ang. overfitting). Razred 0 se včasih pojavi ničkrat, ker pristranskost (ang. bias) dveh metod vedno proizvede 60% ničel in nočemo razreda 0 prekomerno predstavljat. Razredi so uporabljeni direktno v treningu modela NM v obliki: (kv_i, FL_i)

```
if(sentence_index<int(how_many_sentences*random.uniform(1.1,1.7))):
    fuzzy_score[sentence_position] = 3
elif(sentence_index<int(how_many_sentences*random.uniform(2.,2.3))):
    fuzzy_score[sentence_position] = 2
elif(sentence_index<int(how_many_sentences*random.uniform(3.2,4.1))):
    fuzzy_score[sentence_position] = 1
elif(sentence_index<int(how_many_sentences*random.uniform(4.,4.1))):
    fuzzy_score[sentence_position] = 0
else:
    fuzzy_score[sentence_position] = 0
```

Slika 16: Klasifikacija razredov, vrednosti znotraj funkcije naključnosti so fiksne, pridobljene s raznim testiranjem.

4.2.2 Implementacija naivnega Bayesa

V tem podpoglavju po prikazana in razložena programska implementacija metode naivenga Bayesa. Programska implementacija je sestavljena iz več pomožnih metod, ki skupaj proizvedejo željen rezultat. Koda povzeta in prilagojena po [1].
Prva pomožna metoda je ločevanje po razredih:

```
def separate_by_class(dataset):
  separated = dict()
  for i in range(len(dataset)):
        vector = dataset[i]
        class_value = vector[-1]
        if (class_value not in separated):
          separated[class_value] = list()
        separated[class_value].append(vector)
  return separated
```

Prvo moramo ločiti vrednosti po razredih (0 za izključene in 1 za vključene v povzetku), vrednosti razredov pridobimo iz metode, ki je pred Bayesom v cevovodu (GSM, slika 15). Vrednosti ločimo tako, da uporabimo python slovar(hashmap (podpoglavje 4.1.2)), kjer so GSM razredi ključi in preostal del vektorja je vsebina predala na katerega kaže ključ. vector = dataset[i] je i-ti povedni vektor v besedilu, class_value = vector[-1] (0 ali 1) je zadnja vrednost i-tega vektorja(GSM klasifikacija), python stavek 'if(class_value not in seperate)' je vprašanje ali ključ že obstaja znotraj hashmapa, python stavek 'separated[class_value] = list()' ustvari nov predal s ključom class_value (0 ali 1) če ta že ne obstaja, python stavek 'separated[class_value].append(vector)' doda predalu povedni vektor.

Primer uporabe funkcije in rezulatata:

```
# Test summarizing by class
dataset = [[3.393533211,2.331273381,0],
        [3.110073483,1.781539638,0],
        [1.343808831,3.368360954,0],
        [3.582294042,4.67917911,0],
        [2.280362439,2.866990263,0],
        [7.423436942,4.696522875,1],
        [5.745051997,3.533989803,1],
        [9.172168622,2.511101045,1],
        [7.792783481,3.424088941,1],
        [7.939820817,0.791637231,1]]

separated = separate_by_class(dataset)
print(separated)
```

V primeru nam print(separated) izpiše slovar s dvema predalom (s naslovom 0 in 1), ki vsebujeta listo vektorjev, s vrednostmi in na koncu še razred(zadnji stolpec) kateremu pripada objekt, ki ga vektor predstavlja (v našemu primeru bi vektor zastopal eno poved in njen razred bi bil klasifikator, ki ga je GSM metoda proizvedla(0 ali 1)):

```
{0: [[3.393533211, 2.331273381, 0],
[3.110073483, 1.781539638, 0],
[1.343808831, 3.368360954, 0],
[3.582294042, 4.67917911, 0],
[2.280362439, 2.866990263, 0]],

1: [[7.423436942, 4.696522875, 1],
[5.745051997, 3.533989803, 1],
[9.172168622, 2.511101045, 1],
[7.792783481, 3.424088941, 1],
[7.939820817, 0.791637231, 1]]}
```

Druga pomožna funkcija izračuna povprečje:

```
# Calculate the mean of a list of numbers
def mean(numbers):
  return sum(numbers)/float(len(numbers))
```

Povprečje se računa s formulo:

$$povprečje = \frac{vsota\_Števil\_v\_stolpcu}{dolžina\_stolpca}$$

kjer so npr. vrednosti stolpca (ang. column) [5,6,2,4] njihova vsota 5+6+2+4=17 in dolžina stolpca je 4, povprečje je v tem primeru 17/4=4.25. V sami implementaciji parameter funkcije mean(numbers) predstavlja stolpec.
Tretja pomožna funkcija izračuna standardno deviacijo:

```
# Calculate the standard deviation of a list of numbers
def stdev(numbers):
  avg = mean(numbers)
    if(float(len(numbers)-1) != 0):
      variance = sum([(x-avg)**2 for x in numbers])/
      float(len(numbers)-1)
      return sqrt(variance)
    else:
  return 0
```

Standardna deviacija (ang. standard deviation) je merilo količine variacije ali razpršenosti niza vrednosti. Nizka standardna deviacija pomeni, da so vrednosti blizu povprečne vrednosti (imenovane tudi pričakovana vrednost (ang. expected value)), medtem ko visoka standardna deviacija pomeni, da so vrednosti razporejene po širšem območju. Standardno deviacijo lahko izračunamo s formulo:

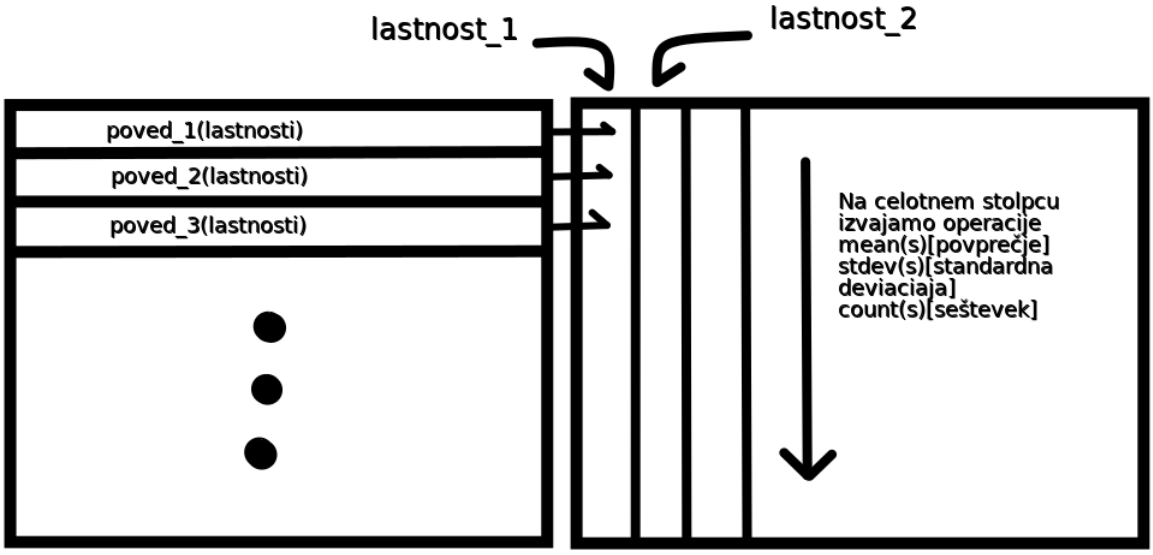
$$SD = \sqrt{\frac{\sum_i^N stolpec_i - povprečje(stolpec)^2}{N-1}}$$

kjer je N število stolpcev, $stolpec_i$ predstavlja i-to vrstico stolpca v temu primeru eno številko, $povprečje(stolpec)^2$ predstavlja povprečje celotnega stolpca na kvadrat.

Četrta pomožna funkcija izračuna standardno deviacijo, povprečje in seštevek stolpca(ang. column):

```
# Calculate the mean, stdev and count for each column
def summarize_dataset(dataset):
    summaries = [(mean(column), stdev(column),
    len(column)) for column in zip(*dataset)]
    del(summaries[-1])
    return summaries
```

Funkcija uporabi prej definirane pomožne funkcije in jih združi, zip(*dataset) spremeni stolpce v vrstice(ang. row) in obratno, se pravi, če je bila prej lista vektorjev(lista povedi in vsaka poved je vektor, ki je predstavljen kot lista lastnosti) je zdaj tudi lista vektorjev vendar vektorji znotraj liste niso več vektorji povedi $p_i = (l_1, l_2...l_8)$ ampak $s_i = (pl_{11}, pl_{12}...)$, kjer je p_i i-ti povedni vektor, $l_1, l_2...$ lastnosti povednega vektorja, s_i stolpec(ang. column) nove liste, pl_{ij} i-ta lastnost j-te povedi.



Slika 17: Grafična reprezentacija zgoraj opisane funkcije,summaries = [(mean(column), stdev(column), len(column)) for column in zip(*dataset)] postane nova lista vektorjev (ki predstavljajo stolpce (ang. column)) s samo tremi lastnostmi standardno deviacijo, povprečjem in številom vrstic(ang. rows)(v temu primeru število povedi).

Primer delovanja teh štirih pomožnih funkcij skupaj:

```
# Test summarizing a dataset
dataset = #st1        #st2     #st3
    [[3.393533211,2.331273381,0],
        [3.110073483,1.781539638,0],
        [1.343808831,3.368360954,0],
        [3.582294042,4.67917911,0],
        [2.280362439,2.866990263,0],
        [7.423436942,4.696522875,1],
        [5.745051997,3.533989803,1],
        [9.172168622,2.511101045,1],
        [7.792783481,3.424088941,1],
        [7.939820817,0.791637231,1]]
summary = summarize_dataset(dataset)
print(summary)
```

Po uporabi funkcije summarize_dataset(dataset) nam print(summary) izpiše rezultat [(5.178333386499999, 2.7665845055177263, 10), (2.9984683241, 1.218556343617447, 10)], to je lista rezultatov za oba stolpca (st1 in st2 v primeru, st3 pobrišemo na koncu funkcije s stavko del(summaries[-1])), vsak rezultat vsebuje povprečje, standardno deviacijo(za izbrani razred) in število vseh vrstic, liste podatkov, ki smo jo vstavili v funkcijo.
Peta pomožna funkcija spet združi prej omenjen metode:

```
#Split dataset by class then calculate statistics for each row
def summarize_by_class(dataset):
  separated = separate_by_class(dataset)
  summaries = dict()
  for class_value, rows in separated.items():
        summaries[class_value] = summarize_dataset(rows)
  return summaries
```

V tej metodi naredimo enak postopek kot pri četrti vendar naredimo to za posamezen razred oziroma listo vektorjev, ki padejo v ta razred, primer:

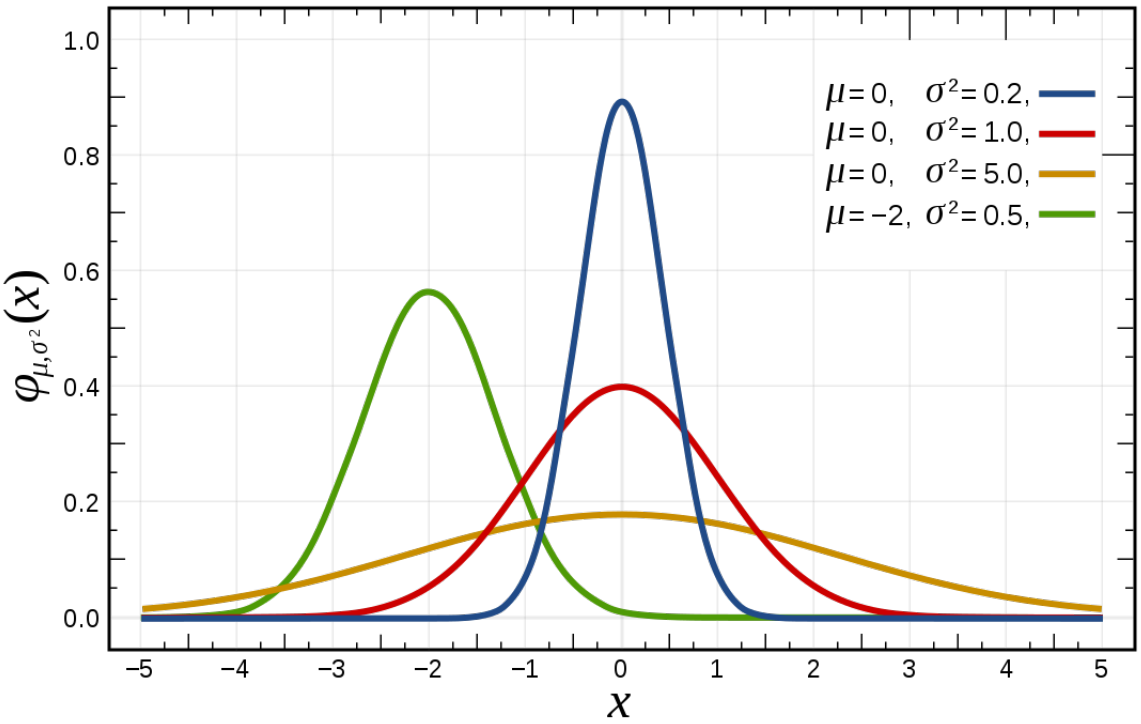
```
# Test summarizing by class
dataset = [[3.393533211,2.331273381,0],
        [3.110073483,1.781539638,0],
        [1.343808831,3.368360954,0],
        [3.582294042,4.67917911,0],
        [2.280362439,2.866990263,0],
        [7.423436942,4.696522875,1],
        [5.745051997,3.533989803,1],
        [9.172168622,2.511101045,1],
        [7.792783481,3.424088941,1],
        [7.939820817,0.791637231,1]]

summary = summarize_by_class(dataset)
for label in summary:
  print(label)
  for row in summary[label]:
        print(row)
```

Metoda summarize_by_class(dataset) nam vrne:

```
0
(2.7420144012, 0.9265683289298018, 5)
(3.0054686692, 1.1073295894898725, 5)
1
(7.6146523718, 1.2344321550313704, 5)
(2.9914679790000003, 1.4541931384601618, 5)
```

To so vrednosti povprečje, standardna deviacija in število vrstic (glede na razred, v temu primeru je to 5 vrstic za razred 0 in 5 za razred 1) za prva dva stolpca (ang. column), saj tretjega pobrišemo (ni smiselno računati standardno deviacijo ali povprečje samih 0 ali 1).
Šesta pomožna funkcija izračuna Gaussovo funkcijo distribucije:

```
#Calculate Gaussian probability distribution function for x
def calculate_probability(x, mean, stdev):
  if(stdev != 0):
        exponent = exp(-((x-mean)**2 / (2 * stdev**2 )))
        return (1 / (sqrt(2 * pi) * stdev)) * exponent
  return 0
```



Slika 18: Normalizirane Gaussove krivulje s povprečno vrednostjo (μ) in varianco (σ). Eden način s katerim lahko izračunamo povprečno vrednost stolpca (ang.column) je da predpostavimo, da je Bellova ali Gaussova distribucija. Za izračun Gaussove distribucije potrebuje le povprečno vrednost in standardno deviacijo, z malo matematike lahko ocenimo verjetnost izbrane vrednosti. (Vir: [23])

Da lahko izračunamo Gaussovo distribucijo potrebujemo le dve meri oziroma parametra, povprečje in standardno deviacijo. Gaussova verjetnostna funkcija(ang. Gaussian probabilty density function) se izračuna s formulo:

$$f(x) = \frac{\frac{1}{\sqrt{2*\pi*\sigma}} * exp(-(x - mean)^2)}{(2 * \sigma^2)}$$

kjer je sigma(σ) standardna deviacija od stolpca (ang. column) x, mean predstavlja povprečje od stolpca (ang. column) x, PI(π) je matematična konstanta (3.14...) in funkcija exp() predstavlja exponentno funkcijo, ki jo napišemo kot $exp(x) = e^x$, kjer je e Eulerjevo število (2.71...). To je vse kar potrebujemo da izračunamo naivni Bayes. Naslednja funkcija ne bo več pomožna temveč, zaključek in bo združila vse te metode v eno. Bayes, ki ga v temu sklopu računamo je malo drugačen kot na sliki 4, saj ima obliko P(razred | data) = P(X | razred) * P(razred), kot vidimo manjka denominator (spodnji del enačbe), ki bi v tem primeru bil P(data). Denominator je bil odstranjen, ker je takšna implmentacija programsko lažja in bolj smiselna, saj nas bolj zanima klasifikacija razreda (maksimizacija gotovosti), kot računanje verjetnosti (našo maksimalno gotovost pretehtati (ang. weighting) s dokazi).

Vstavljene spremenljivke so računane ločeno, to pomeni da vsaka spremenljivka predstavlja svoj dogodek, ki ni odvisen od prejšnega, zato imenujemo ta način 'naivini' Bayes, primer, da stolpec X(X1 in X2 so vrstice (ang. rows) stolpca (ang.columns) X) pridpada razredu 0:
P(razred=0 | X1,X2) = P(X1 | razred=0) * P(X2 | razred=0) * P(razred=0)
Primer uporabe opisane programske funkcija:

```
# Test Gaussian PDF # Probabilty density function
print(calculate_probability(1.0, 1.0, 1.0))
print(calculate_probability(2.0, 1.0, 1.0))
print(calculate_probability(0.0, 1.0, 1.0))
```

Rezultati so:
0.3989422804014327
0.24197072451914337
0.24197072451914337

Zadnja funkcija, ki združi vse prejšne metode v eno:

```
# Calculate the probabilities of predicting each class
def calculate_class_probabilities(summaries, row):
  total_rows = sum([summaries[label][0][2] for
  label in summaries])
  probabilities = dict()
  for class_value, class_summaries in summaries.items():
        probabilities[class_value] =
        summaries[class_value][0][2]/float(total_rows)
        for i in range(len(class_summaries)):
          mean, stdev, _ = class_summaries[i]
          probabilities[class_value] *=
          calculate_probability(row[i], mean, stdev)
  return probabilities
```

Ta metoda nam vrne slovar (stavek probabilities = dict() deklarira prazen slovar), ki ima razrede za predale in notri pa so vrjetnosti, da vektor pripada razredu, prva vrstica total_rows = sum(...) prešteje koliko vrstic (v našem primeru koliko povedi je v nabor podatkov), stavek probabilities[class_value] =summaries[class_value][0][2] /float(total_rows) nam določi kolikšno količino podatkov predstavlja ta nabor povedi (v %), stavek mean, stdev, _ = class_summaries[i] nam vrne povprečje in standardno deviacijo, zdaj ko imamo ta dva parametra jih seveda vstavimo v prej opisano pomožno

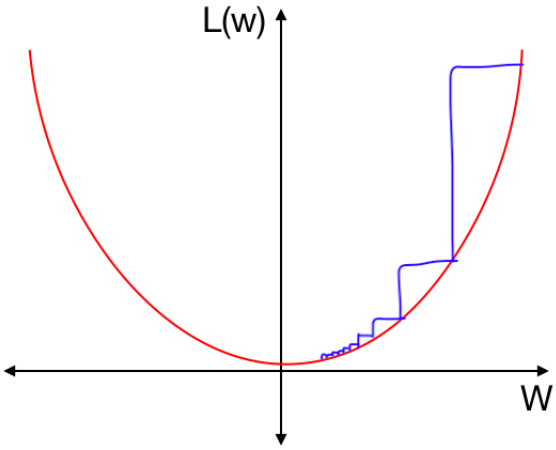
funkcijo (calculate_probability): probabilities[class_value]*=calculate_probability(row[i], mean, stdev), kjer notacija *= pomeni enako kot notacija v matematiki: $\prod_{n=0}^{\infty} a_n = a_0 * a_1 * ...$, kar pomeni, da večkrat množimo vrednost, ki je v predalu slovarja probabilities[class_value], primer:

```
# Test Gaussian PDF # Probabilty density function
# Test calculating class probabilities
dataset = [[3.393533211,2.331273381,0],
        [3.110073483,1.781539638,0],
        [1.343808831,3.368360954,0],
        [3.582294042,4.67917911,0],
        [2.280362439,2.866990263,0],
        [7.423436942,4.696522875,1],
        [5.745051997,3.533989803,1],
        [9.172168622,2.511101045,1],
        [7.792783481,3.424088941,1],
        [7.939820817,0.791637231,1]]
summaries = summarize_by_class(dataset)
probabilities=
calculate_class_probabilities(summaries, dataset[0])
print(probabilities)
```

Stavek print(probabilities) nam izpiše, 0: 0.05032, 1: 0.00011, kar je slovar s dvema naslovoma 0 in 1, ki kažejo vsaka na svojo verjetnost(prepričanje oziroma gotovost) rezultata, v temu primeru Bayes sklepa, na podlagi videnih primerov(preostanek liste), da dataset[0](prva vrstica(ang. row) liste) pripada razredu 0.

4.2.3 Implementacija nevronske mreže

Vrsta NM je preprosti feedfoward sekvenčni model, optimizacijska metoda je stohastični gradientski spust(ang. stohastic gradient descent), algoritem adam [13].



Slika 19: Zelo preprost prikaz delovanja stohastičnega gradientska spusta, metodi rečemo spust, saj se spuščamo dol po 'dolini' funkcije, do dna (lokalni minimum), ki nas zanima.Ta proces izvajamo s matematičnimi odvodi, saj ti nam pokažejo trenutno naklon, kjer na funkciji se nahajamo in potem se premaknemo levo ali desno, v odvisnosti kje je dno. V praksi so funkcije na katerih uporabimo to metodo polinomske in imajo več 'hribov' in 'dolin', zato včasih najdemo najboljše dno (lokalni minumum) včasih pa ne. (Vir: [19])

Nevronske celice vsebujejo sigmoid aktivacijsko funkcijo. NM je sestavljen iz petih gostih plasti (ang. dense layers), to so plasti, kjer so nevroni povezani v razmerju vsak z vsakim naslednje plasti. Pragovna funkcija mreže je redka kategorična navzkrižna entropija (ang. sparse categorical crossentropy), ta funkcija je dobra, ko hočemo klasificirati več različnih razredov, oziroma klasifikacija ni binarna (0,1), kot v temu delu. Oblika NM je piramida, lahko bi bila karkoli, izbrana je bila prva iteracija, ki je proizvedla dobre rezultate, to je 11 vhodnih nevronov (8 za lastnosti vsake povedi, 3 za dodatek GSM, NB in FL), 14*18*22*26*30 skritih nevronov in 4 izhodnih nevronov(za vrednosti 0,1,2,3]).

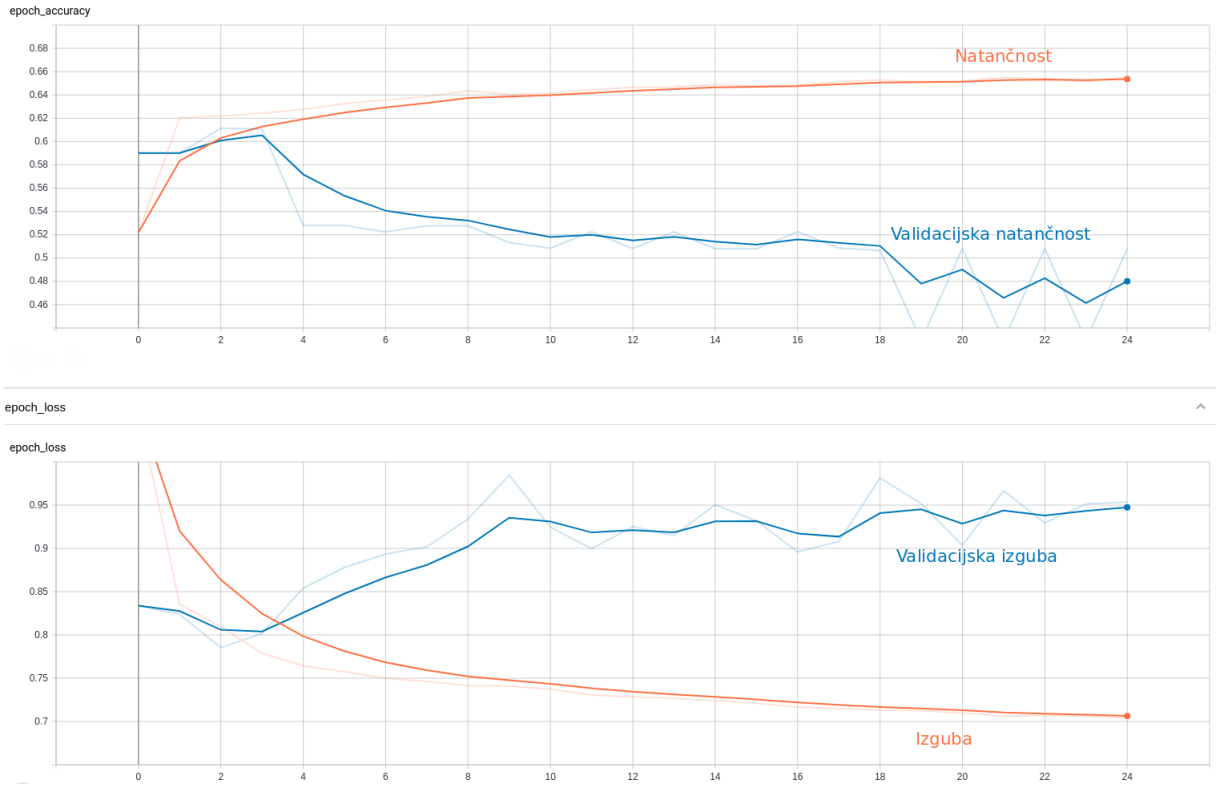
```
model = tf.keras.models.Sequential([
        tf.keras.layers.Dense(30, input_shape=(11,),
        activation='sigmoid'),
        tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(26, activation='sigmoid'),
        tf.keras.layers.Dense(22, activation='sigmoid'),
        tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(18, activation='sigmoid'),
        tf.keras.layers.Dense(14, activation='sigmoid'),
        tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(4, activation='softmax')
])
model.compile(optimizer='adam',
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy'])
```

Namesto dveh (0,1) izhodnih nevronov so štiri, ker hočemo naučiti NM klasificirat med dobrimi in slabimi povedi, če bi bila samo dva izhoda, bi lahko rezultat vseboval premalo povedi za povzetek (manj kot 20%) in po takšni klasifikaciji, bi moral biti tudi podporni algoritem, ki bi naredil ponovno izbiro izmed izbranih povedi. petindvajset epoh (ang. epochs) je optimalna dolžina treninga, je razvidno iz slike 21. podatki so razdeljeni na 80% za trening in 20% za validacijo, to je 1733 člankov (46575 vektorjev povedi) za trening in 434 (8430 vektorjev povedi) za testiranje. Validacijska zguba/natančnost predstavlja rezultat, ko NM primerja rezultate, ki niso znotraj seta za trenening temveč za validacijo(so neznani). Natančnost (ang. accuracy) NM je približno 50% - 60%, klasificira vsaj 10-25% (razred 3) dobrih povedi, 15%-30% prav dobrih (razred 2),15%-30% povprečnih (razred 1) in 15% do 60% (zaradi 40% pristranskosti (ang. bias) FL in GSM) slabih (razred 0), kar pomeni da izloči slabe. Učenje NM (slika 21) ni odvisno od natančnosti NM oziroma validacijske natančnosti, pravzaprav to, da se pomika v drugo smer je dober znak, saj hočemo, da se mreža nauči klasifikacije povedi iz vseh metod in ne samo FL, ki proizvede te razrede klasifikacije, definitivno hočemo, da se nauči največ od NB ampak nočemo pa, da ne samo kopira eno od metod, zato so pristranskosti (ang. bias) vseh treh metod zelo pomembna. Učenje skratka poteka glede na graf (slika 31) in (slika 29), kjer lahko vidimo kje približno stoji NM, v relaciji z vsemi metodami, in potem prilagodimo 'meje'(slika 16) med razredi in pristranskosti (ang. bias) metod.

```
7 0 2 1
16 1 6 2
23 3 10 2
2 2 2 1
4 3 4 0
1 1 0 1
21 2 10 1
2 0 0 1
4 3 5 0
39 1 16 2
54 3 20 3
8 0 2 2
18 1 6 3
12 2 4 2
```

Slika 20: Primer slabe distribucije, s slike je razvidno da je veliko več povedi klasificiranih (0,1,2,3 so razredi katere stolpci predstavljajo), kot razred 0(prvi stolpec)-slabe in razred 2 (tretji stolpec)-prav dobre, to je enako kot če bi klasificirali med samo tema dvema razredoma, v primerjavi z distribucijo na slikah 23 in 22.

Če naše nastavitve niso pravilne je lahko distribucija, ki jo NM proizvede, slaba, kot na sliki 20, kjer sta razreda 0 in 2 (prvi in drugi stolpec) prekomerno zastopana in popolnoma zasenčijo druga dva razreda 1 in 3 (drugi in četrti stolpec), kar pomeni, da je takšna oblika klasifikacije neuporabna saj bi enak rezultat dosegli če bi klasificirali samo med dvema, recimo 0 in 1, kar bi pomenilo dobro in slabo, mi pa hočemo najboljše povedi na vrhu. Še ena pomembna obzervacija je, da pri krajših besedilih te nastavitve (ki niso bile najslabše pravzapra, v fazi testiranje je bilo huje), včasih delujejo, vendar povzetki kratkih besedil niso tako pogosti v praksi in nas bolj zanimajo dolga besedila.



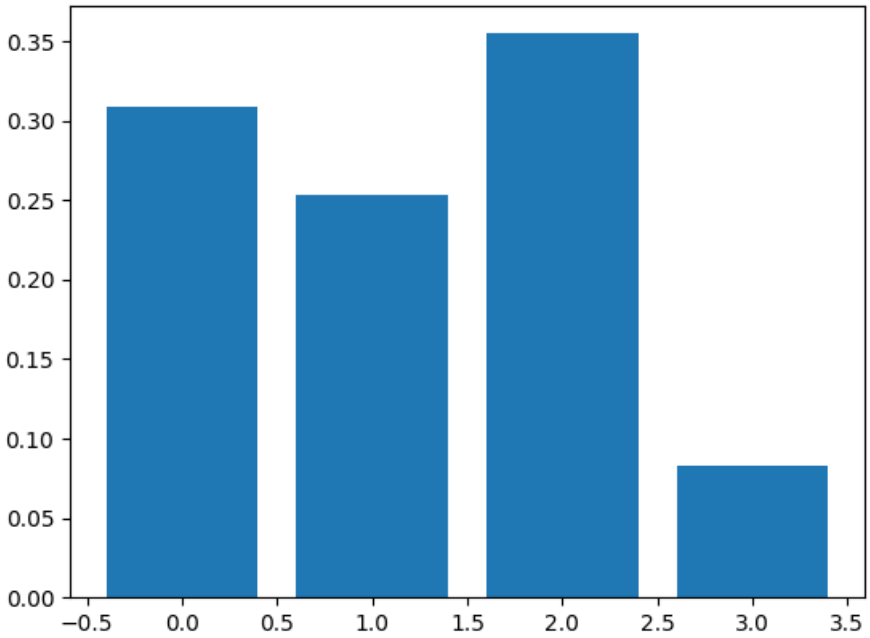
Slika 21: Graf učenja NM, izgleda narobe, saj bi validacijska zguba in validacijska natančnost morala v obratno smer spuščat/naraščat.

Drugi indikator za učenje mreže in prilagajanje parametrov je distribucija razredov (slika 22), ki jo naučena mreža proizvede.

```
4 4 4 2
2 8 6 1
1 2 2 1
16 12 17 4
2 6 3 3
17 11 17 3
22 14 22 3
4 2 2 2
9 6 8 2
13 9 14 2
0 4 2 1
3 3 3 2
1 0 0 2
11 9 11 3
1 0 0 2
2 4 5 1
22 11 22 3
31 15 31 3
4 3 3 2
9 6 10 3
8 4 6 2
2 10 7 1
1 3 1 2
2 2 2 2
5 2 6 2
6 8 10 1
5 4 4 2
19 13 19 3
6 5 5 3
10 8 9 3
```

Slika 22: Distrbucija razredov(0,1,2,3) pomembnosti povedi, 3 je najbolj pomembna poved in 0 najmanj.

Najboljša distribucija bi bila seveda 1/4,1/4,1/4,1/4 vendar je nekaj takšnega nemogoče, je pa približno 30%, 26%, 36% in 8%(slika 23), kar je dovolj dobro za proizvajanje povzetkov dolžine 20% do 35%.



Slika 23: Distrbucija razredov (0,1,2,3 v 0.0, 1.0, 2.0, 3.0). Ta distribucija je normalizirana iz vseh člankov, posamezen članek ima lahko drugačno distribucijo, če bi želeli proizvajat induktivne povzetke(5% besedila, nam ta distribucija to dovoljuje.

5 TESTIRANJE IN EVALUACIJA

5.1 Testiranje

Besedilo, ki ga vrne program ni v orginalni obliki, ampak v preobdelani obliki, ta ne vsebuje "", ki ponazarjajo premi govor npr. 'Janez je rekel:"super"', postane 'Janez je rekel super', kar je še vedno berljivo. Testiranje je storjeno na prvih 20 člankih, trening data baze, iz teh člankov je bil izbran najdaljši, ker najboljše prikazuje razlike med metodami. Povzetek predstavlja 30% povedi, vendar je količina samega besedila malo več kot to.

['Ameriški beli privilegij pognojen z rasizmom.', 'Še en dan v Združenih državah še en neoborožen temnopolti moški ki je umrl zaradi neupravičenega neznosnega in nezaslišanega nasilja policije.', 'Isti dan je v newyorškem Centralnem parku mlada belka poklicala policijo da njeno življenje ogroža afriški Američan.', 'Ker ji je rekel naj psa priveže na povodec.', 'ZDA so še vedno prežete z rasizmom.', 'Smrt Georga Floyda v Minneapolisu je toliko bolj v nebo vpijoča ker je le zadnji primer v dolgi vrsti policijskega nasilja dokumentiranega s telefoni mimoidočih.', 'Zadnje desetletje smo priča številnim smrtim predvsem mladih temnopoltih moških pri stiku s policijo ki so sprožile ostre odzive javnosti in protestno gibanje Temnopolta življenja so pomembna ter obljube in zagotovila o reformah policijskega dela.', 'Šest let po smrti Erica Garnerja ki se je v New Yorku zadušil v grobem prijemu policista znova gledamo prizore belega policista ki z brezbrižnim izrazom na obrazu kleči na vratu temnopoltega moškega ta pa hrope da ne more dihati.', 'In potem ob naraščajoči jezi očividcev izgubi zavest.', 'Po razvidu spletne strani Mapping police violence je policija v ZDA lani ubila 1 099 ljudi od tega je bilo 24 odstotkov temnopoltih.', 'Dvakrat več kot je delež temnopoltih v ZDA.', 'Župan Minneapolisa Jacob Frey je bil jasen Biti temnopolt v ZDA ne bi smela biti smrtna obsodba.', 'Vse štiri policiste ki so sodelovali v aretaciji so po besedah župana že odpustili v torek so morali pred sedežem policije v mestu s solzilcem razganjati besne protestnike.', 'Ogorčeni smo da skoraj šest let po tem ko je Eric Garner hropel Ne morem dihati policisti še vedno ne zmorejo prisluhniti klicem na pomoč je dejala Kristina Roth iz ameriške podružnice organizacije Amnesty International.', 'Predmestje Minneapolisa je bilo pred štirimi leti prizorišče brutalnega umora temnopoltega voznika Philanda Castila policist je vanj iz bližine izstrelil pet krogel a bil kljub temu oproščen.', 'Obsojen je bil le temnopolti policist ki je leto pozneje prav tako v Minneapolisu ustrelil belko ker je prišla do avtomobila in ga presenetila.', 'Če ob strani pustimo dejstvo da ima policija v Minneapolisu očitne težave pri izvajanju svojih osnovnih nalog je jasen rasni podton.', 'Dejanja kažejo na rasistično do tujcev sovražno in predsodkov polno razpoloženje družbe do temnopoltih je opozorila Leslie Remond iz organizacije za pravice temnopoltih NAACP.', 'Cooper proti Cooperju', 'Njene besede je potrdilo dogajanje v New Yorku kjer je mlada belka poklicala policijo ker jo je razjezila zahteva temnopoltega moškega naj psa priveže na povodec.', 'Ramble je gozdnat del Centralnega parka ki je namenjen opazovanju ptic zato lahko lastniki štirinožce tam sprehajajo samo na povodcu.', 'Ko je temnopolti Christian Cooper na to opozoril belo Amy Cooper nista v sorodu je to popadla jeza poklicala je policijo in histerično vpila da Afriški Američan ogroža njeno življenje in naj takoj pošljejo na pomoč patruljo.', 'Christian je dogajanje posnel na telefon in ko je posnetek sprožil vihar na družbenih omrežjih so težave zasule Amy.', 'Toda ponavadi je obratno.', 'Bela oseba pokliče policijo proti temnopoltemu moškemu.', 'Policija prispe in se postavi na stran belega tožnika zavrne pa plat temnopoltega.', 'Tega aretirajo in proti njemu vložijo obtožnico.', 'Sodišče določi čezmerno varščino.', 'Njegova družina si ne more privoščiti da bi kupila njegovo svobodo.', 'Pošljejo ga v ječo kjer sedi dneve mesece včasih celo leta.', 'Na koncu je njegov primer razrešen ali zavrnejo obtožnico ali pa sklene dogovor s tožilci in prizna krivdo za manjše dejanje.', 'V tem času lahko izgubi delo svoj dom otroke je razmere opisala Eliza Orlins javna branilka v New Yorku.', 'Po njenih besedah je že imela primere ko so tožilci histerični klic policije predložili kot dokaz da je bila klicateljica ogrožena saj zakaj bi vendar lagala?', ' .', 'V sistem vgrajena rasna hierarhija', 'V času pandemije koronavirusa ki še posebej ogroža ameriške zapore je pošiljanje v pripor zaradi lažnih obtožb iz parka lahko celo smrtna obsodba opozarja Orlinsova.', 'Kot kaže vrsta primerov zadnjih let ko so telefoni s kamerami postali priča policijskega delovanja je lahko za temnopolte usodno že soočenje s policijo.', 'Če živijo v katerih od južnjaških zveznih držav pa so smrtno nevarni že tisti beli sodržavljani ki v vsakem temnopoltem vidijo potencialnega kriminalca.', 'Tek Ahmauda Arberyja v Georgiji se je končal s streli belca ki je menil da gre za vlomilca in je sklenil zakon vzeti v svoje roke.', 'V državi naphani z lahko dostopnim orožjem so streljanja vsakodnevna odzivi policije pa niti ne presenetljivi saj je lahko vsakdo oborožen.', 'Toda po besedah publicistke Ellen Mcgirt beli Američani moč države že dolgo uporabljajo kot taktiko za vzpostavljanje rasne hierarhije.', 'Aprila 2018 se je moral vrh verige kavarn Starbucks opravičiti ker so v Filadelfiji v eni od njih poklicali policijo proti dvema temnopoltima gostoma.', 'Njun zločin je bil da nista želela naročiti dokler se jima ne pridruži znanec ki sta ga čakala.', 'Takšno vedenje je vsesplošno razširjeno.', 'Beli študent je v študentskem domu univerze Yale kar dvakrat poklical policijo ker so ga zmotili temnopolti.', 'Izkazalo se je da tudi oni študirajo na prestižni univerzi.', 'Belci so klicali policijo ker je 8 letna deklica na ulici prodajala vodo.', 'Ali proti 12 letniku ki je kosil travo v soseščini.', 'V Kaliforniji so s pomočjo policijskega helikopterja pripri temnopolto četverico ki je zapuščala Airbnb ker so beli sosedje mislili da ropajo.', 'Ena od četverice je bila Donisha Prendergast vnukinja slovitega glasbenika Boba Marleyja.']

Slika 24: Original besedilo, ločeno po povedih, prva poved je naslov besedila.

Title: Ameriški beli privilegij pognojen z rasizmom.

Še en dan v Združenih državah še en neoborožen temnopolti moški ki je umrl zaradi neupravičenega neznosnega in nezaslišanega nasilja policije. Isti dan je v newyorškem Centralnem parku mlada belka poklicala policijo da njeno življenje ogroža afriški Američan. Ker ji je rekel naj psa priveže na povodec. ZDA so še vedno prežete z rasizmom. Smrt Georga Floyda v Minneapolisu je toliko bolj v nebo vpijoča ker je le zadnji primer v dolgi vrsti policijskega nasilja dokumentiranega s telefoni mimoidočih. Zadnje desetletje smo priča številnim smrtim predvsem mladih temnopoltih moških pri stiku s policijo ki so sprožile ostre odzive javnosti in protestno gibanje Temnopolta življenja so pomembna ter obljube in zagotovila o reformah policijskega dela. Šest let po smrti Erica Garnerja ki se je v New Yorku zadušil v grobem prijemu policista znova gledamo prizore belega policista ki z brezbrižnim izrazom na obrazu kleči na vratu temnopoltega moškega ta pa hrope da ne more dihati. Po razvidu spletne strani Mapping police violence je policija v ZDA lani ubila 1 099 ljudi od tega je bilo 24 odstotkov temnopoltih. Dejanja kažejo na rasistično do tujcev sovražno in predsodkov polno razpoloženje družbe do temnopoltih je opozorila Leslie Remond iz organizacije za pravice temnopoltih NAACP. Njene besede je potrdilo dogajanje v New Yorku kjer je mlada belka poklicala policijo ker jo je razjezila zahteva temnopoltega moškega naj psa priveže na povodec. Ko je temnopolti Christian Cooper na to opozoril belo Amy Cooper nista v sorodu je to popadla jeza poklicala je policijo in histerično vpila da Afriški Američan ogroža njeno življenje in naj takoj pošljejo na pomoč patruljo. Kot kaže vrsta primerov zadnjih let ko so telefoni s kamerami postali priča policijskega delovanja je lahko za temnopolte usodno že soočenje s policijo. Toda po besedah publicistke Ellen Mcgirt beli Američani moč države že dolgo uporabljajo kot taktiko za vzpostavljanje rasne hierarhije. Beli študent je v študentskem domu univerze Yale kar dvakrat poklical policijo ker so ga zmotili temnopolti. V Kaliforniji so s pomočjo policijskega helikopterja priprli temnopolto četverico ki je zapuščala Airbnb ker so beli sosedje mislili da ropajo.

Slika 25: Povzetek, baziran na generalni statistični metodi.

Title: Ameriški beli privilegij pognojen z rasizmom.

Smrt Georga Floyda v Minneapolisu je toliko bolj v nebo vpijoča ker je le zadnji primer v dolgi vrsti policijskega nasilja dokumentiranega s telefoni mimoidočih. Zadnje desetletje smo priča številnim smrtim predvsem mladih temnopoltih moških pri stiku s policijo ki so sprožile ostre odzive javnosti in protestno gibanje Temnopolta življenja so pomembna ter obljube in zagotovila o reformah policijskega dela. Šest let po smrti Erica Garnerja ki se je v New Yorku zadušil v grobem prijemu policista znova gledamo prizore belega policista ki z brezbrižnim izrazom na obrazu kleči na vratu temnopoltega moškega ta pa hrope da ne more dihati. Župan Minneapolisa Jacob Frey je bil jasen Biti temnopolt v ZDA ne bi smela biti smrtna obsodba. Obsojen je bil le temnopolti policist ki je leto pozneje prav tako v Minneapolisu ustrelil belko ker je prišla do avtomobila in ga presenetila. Če ob strani pustimo dejstvo da ima policija v Minneapolisu očitne težave pri izvajanju svojih osnovnih nalog je jasen rasni podton. Dejanja kažejo na rasistično do tujcev sovražno in predsodkov polno razpoloženje družbe do temnopoltih je opozorila Leslie Remond iz organizacije za pravice temnopoltih NAACP. Policija prispe in se postavi na stran belega tožnika zavrne pa plat temnopoltega. V tem času lahko izgubi delo svoj dom otroke je razmere opisala Eliza Orlins javna branilka v New Yorku. V času pandemije koronavirusa ki še posebej ogroža ameriške zapore je pošiljanje v pripor zaradi lažnih obtožb iz parka lahko celo smrtna obsodba opozarja Orlinsova. Kot kaže vrsta primerov zadnjih let ko so telefoni s kamerami postali priča policijskega delovanja je lahko za temnopolte usodno že soočenje s policijo. Toda po besedah publicistke Ellen Mcgirt beli Američani moč države že dolgo uporabljajo kot taktiko za vzpostavljanje rasne hierarhije. Aprila 2018 se je moral vrh verige kavarn Starbucks opravičiti ker so v Filadelfiji v eni od njih poklicali policijo proti dvema temnopoltima gostoma. Beli študent je v študentskem domu univerze Yale kar dvakrat poklical policijo ker so ga zmotili temnopolti. V Kaliforniji so s pomočjo policijskega helikopterja priprli temnopolto četverico ki je zapuščala Airbnb ker so beli sosedje mislili da ropajo.

Slika 26: Povzetek, generiran z Naivnim Bayesom.

Title: Ameriški beli privilegij pognojen z rasizmom.

Isti dan je v newyorškem Centralnem parku mlada belka poklicala policijo da njeno življenje ogroža afriški Američan. Smrt Georga Floyda v Minneapolisu je toliko bolj v nebo vpijoča ker je le zadnji primer v dolgi vrsti policijskega nasilja dokumentiranega s telefoni mimoidočih. Zadnje desetletje smo priča številnim smrtim predvsem mladih temnopoltih moških pri stiku s policijo ki so sprožile ostre odzive javnosti in protestno gibanje Temnopolta življenja so pomembna ter obljube in zagotovila o reformah policijskega dela. Šest let po smrti Erica Garnerja ki se je v New Yorku zadušil v grobem prijemu policista znova gledamo prizore belega policista ki z brezbrižnim izrazom na obrazu kleči na vratu temnopoltega moškega ta pa hrope da ne more dihati. Župan Minneapolisa Jacob Frey je bil jasen Biti temnopolt v ZDA ne bi smela biti smrtna obsodba. Če ob strani pustimo dejstvo da ima policija v Minneapolisu očitne težave pri izvajanju svojih osnovnih nalog je jasen rasni podton. Dejanja kažejo na rasistično do tujcev sovražno in predsodkov polno razpoloženje družbe do temnopoltih je opozorila Leslie Remond iz organizacije za pravice temnopoltih NAACP. Njene besede je potrdilo dogajanje v New Yorku kjer je mlada belka poklicala policijo ker jo je razjezila zahteva temnopoltega moškega naj psa priveže na povodec. Ko je temnopolti Christian Cooper na to opozoril belo Amy Cooper nista v sorodu je to popadla jeza poklicala je policijo in histerično vpila da Afriški Američan ogroža njeno življenje in naj takoj pošljejo na pomoč patruljo. V času pandemije koronavirusa ki še posebej ogroža ameriške zapore je pošiljanje v pripor zaradi lažnih obtožb iz parka lahko celo smrtna obsodba opozarja Orlinsova. Kot kaže vrsta primerov zadnjih let ko so telefoni s kamerami postali priča policijskega delovanja je lahko za temnopolte usodno že soočenje s policijo. Toda po besedah publicistke Ellen Mcgirt beli Američani moč države že dolgo uporabljajo kot taktiko za vzpostavljanje rasne hierarhije. Aprila 2018 se je moral vrh verige kavarn Starbucks opravičiti ker so v Filadelfiji v eni od njih poklicali policijo proti dvema temnopoltima gostoma. Beli študent je v študentskem domu univerze Yale kar dvakrat poklical policijo ker so ga zmotili temnopolti. V Kaliforniji so s pomočjo policijskega helikopterja priprli temnopolto četverico ki je zapuščala Airbnb ker so beli sosedje mislili da ropajo.

Slika 27: Povzetek, generiran z Fuzzy logiko.

Title: Ameriški beli privilegij pognojen z rasizmom.

Ogorčeni smo da skoraj šest let po tem ko je Eric Garner hropel Ne morem dihati policisti še vedno ne zmorejo prisluhniti klicem na pomoč je dejala Kristina Roth iz ameriške podružnice organizacije Amnesty International. Obsojen je bil le temnopolti policist ki je leto pozneje prav tako v Minneapolisu ustrelil belko ker je prišla do avtomobila in ga presenetila. Če ob strani pustimo dejstvo da ima policija v Minneapolisu očitne težave pri izvajanju svojih osnovnih nalog je jasen rasni podton. Dejanja kažejo na rasistično do tujcev sovražno in predsodkov polno razpoloženje družbe do temnopoltih je opozorila Leslie Remond iz organizacije za pravice temnopoltih NAACP. Njene besede je potrdilo dogajanje v New Yorku kjer je mlada belka poklicala policijo ker jo je razjezila zahteva temnopoltega moškega naj psa priveže na povodec. Ko je temnopolti Christian Cooper na to opozoril belo Amy Cooper nista v sorodu je to popadla jeza poklicala je policijo in histerično vpila da Afriški Američan ogroža njeno življenje in naj takoj pošljejo na pomoč patruljo. V tem času lahko izgubi delo svoj dom otroke je razmere opisala Eliza Orlins javna branilka v New Yorku. . V času pandemije koronavirusa ki še posebej ogroža ameriške zapore je pošiljanje v pripor zaradi lažnih obtožb iz parka lahko celo smrtna obsodba opozarja Orlinsova. Kot kaže vrsta primerov zadnjih let ko so telefoni s kamerami postali priča policijskega delovanja je lahko za temnopolte usodno že soočenje s policijo. V državi naphani z lahko dostopnim orožjem so streljanja vsakodnevna odzivi policije pa niti ne presenetljivi saj je lahko vsakdo oborožen. Toda po besedah publicistke Ellen Mcgirt beli Američani moč države že dolgo uporabljajo kot taktiko za vzpostavljanje rasne hierarhije. Aprila 2018 se je moral vrh verige kavarn Starbucks opravičiti ker so v Filadelfiji v eni od njih poklicali policijo proti dvema temnopoltima gostoma. Beli študent je v študentskem domu univerze Yale kar dvakrat poklical policijo ker so ga zmotili temnopolti. V Kaliforniji so s pomočjo policijskega helikopterja priprli temnopolto četverico ki je zapuščala Airbnb ker so beli sosedje mislili da ropajo.

Slika 28: Povzetek, generiran z nevronsko mrežo.

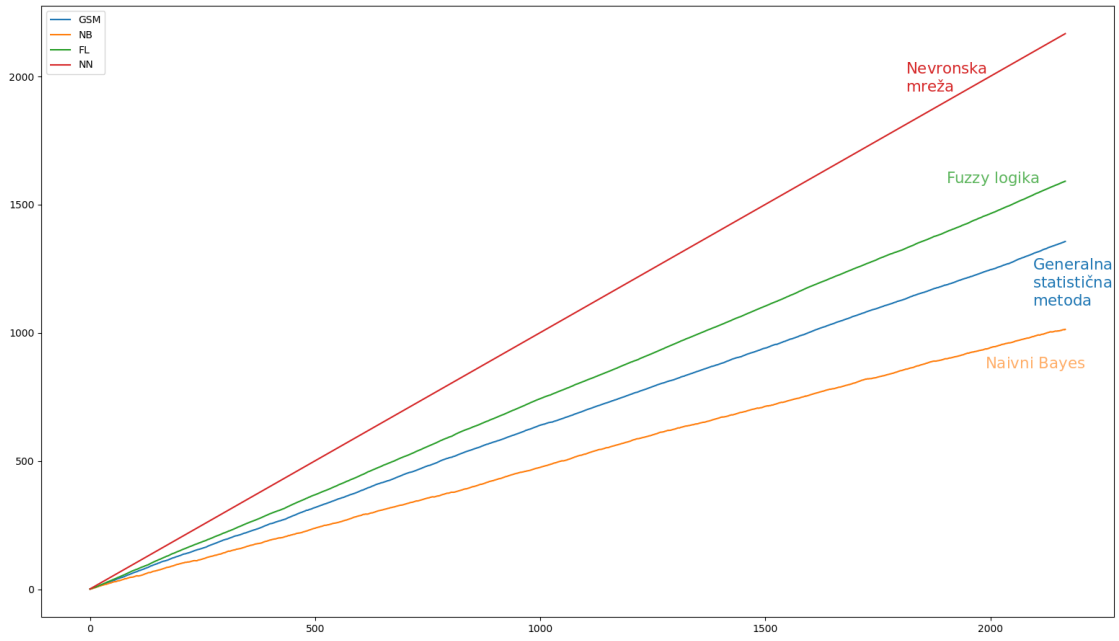
5.2 Evalvacija

Evalvacija meri, efektivnost povzemanlika, koliko kompresije se je zgodilo pri povzemanju in kako dober je bil povzetek. Da dobimo evaluacijo potrebujemo zlati standart, oziroma najboljši možen povzetek, običajno je to človek. Vendar v tem delu, je vse v duhu samodejnosti, kar pomeni da bo tudi evaluacija samodejno storjena. Tako izberemo NM za zlati standart, kot vrhunec vse omenjenih metod strojnega učenja, pa tudi avtorjeve odločitve po ocenitvi povzetkov, ki jih sistem generira. Prvo merimo natančnost, to je mera koliko se metoda razlikuje od zlatega standarta:

$$Natancnost(n) = \frac{pravilnePovedi - napcnePovedi}{VsePovedi}$$

$$a_{n+1} = a_n + Natancost(n) = a_1 + Natancost(1) + a_2 + Natancost(2) ... a_k + Natancost(k)$$

je rekurzivna vrsta, ki nariše graf natančnosti, za k člankov.



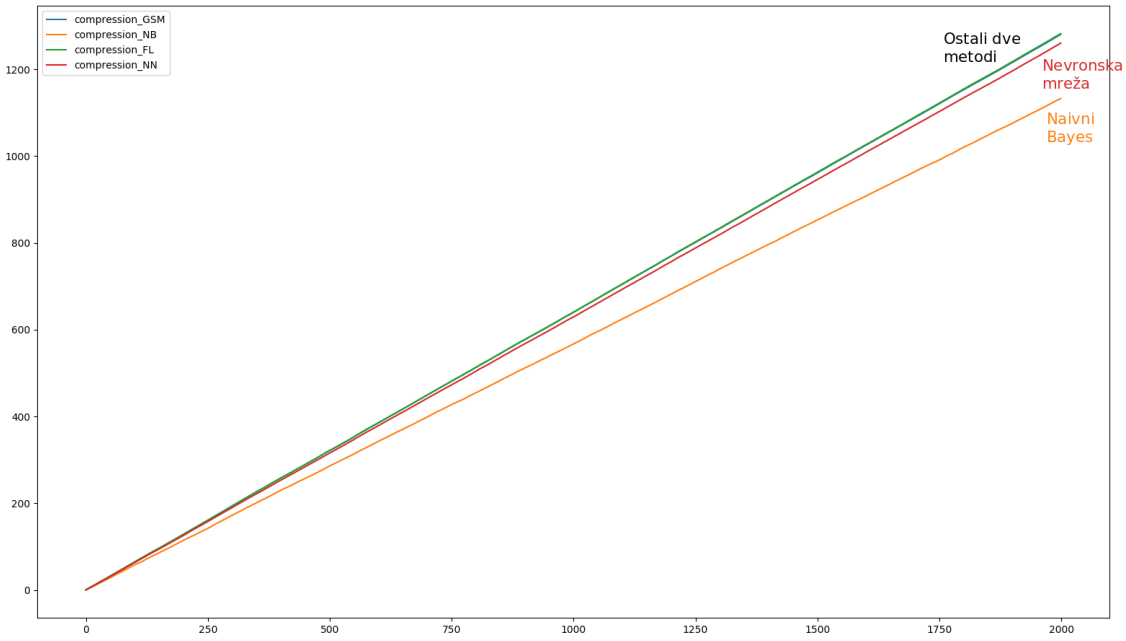
Slika 29: Graf nam pokaže razlike v metodah, kot vidimo NM ne kopira nobene od metod in inkorporira pristrankost (ang. bias) vsake. FL je najbližje, vendar ne prebilzu, kar je tudi zaželjen rezultat.

Drugo merimo kompresijo, to je koliko je povzetek krajši od originala, čeprav nas tukaj to ne zanima, zanima nas le katera metoda je najboljša, mera kompresije se izračuna z enačbo:

$$Kompresija(n) = \frac{VsotaDolzinIzbranihPovedi}{VsotaDolzinVsehPovedi}$$

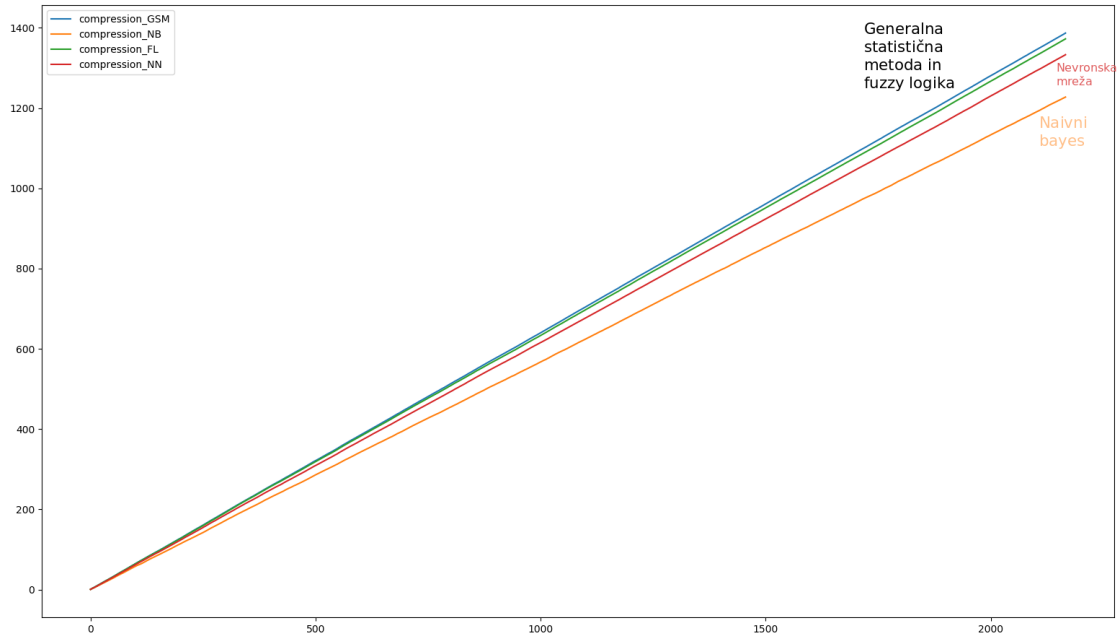
$$a_{n+1} = a_n + Kompresija(n) = a_1 + Kompresija(1) + ... + a_k + Kompresija(k)$$

je rekurzivna vrsta, ki nariše graf kompresije, za k člankov.



Slika 30: Graf kompresije iz zgodnjega obdobja testiranja, s grafa je zelo razvidno, da NB proizvaja krajše povzetke.

Zakaj proizvaja NB krajše povzetke nevemo, vemo pa, da kvaliteta povzetkov pada (slika 29), saj se odaljuje od FL, ki proizvaja precej dobre povzetke, še zlasti pa NM, ki je vrh vseh treh metod.



Slika 31: Grafa kompresije iz zadnje iteracije NM, kot vidimo se je naučila boljše kompresije, kot na sliki 30.

6 ZAKLJUČEK

Delo predstavlja pregled in implementacijo metod strojnega učenja, statistike in linearne algebre v en skupni sistem, ki iz besedila proizvede povzetek. V današnjem času, ko je veliko informacij na voljo in je velika potreba po procesiranju le teh so takšni sistemi nuja. Rezulatati, ki jih je ta preprosti samodejni povzemalnik dosegel so bili dobri in bi ob bolj elaborativnem pristopu proizvedel še boljše rezultate, še je prostora za izboljšavo, še posebno pri besedilih, kot so knjige vseh oblik, kjer besedilo ni uniformno strukturirano, kot v člankih novic. Implementacija bi lahko vključevala še več metod preden se začne trening NM(npr. HMM) in sama nevronska mreža bil lahko bila ponavljoča(ang. reccurent neural network(RNN)), vendar se izkaže, da že navadne metode strojnega učenja so zadostne, na koncu je vse odvisno od domene problema, ki ga rešujemo. Obdelava slovenskega jezika z metoda procesiranja naravngea jezika(ang. Natural language processing(NLP)), se je izkazala, kot precej velika ovira, saj zahteva znananje slovenskega jezikoslovja in zato popularne metode in tehnologije(predvsem pravila o priponah) s angleščine ne delujejo. Z dobrim znanjem slovenskega jezika oziroma slovnice(pravila slovenskega jezika) bi lahko še boljši vendar manj splošen povzemalnik dosegli, še posebaj v smislu predvidevanja(ang. inference) saj jeziki niso naključni ampak predvidljivi, tak povzemalnik bi seveda bil sposoben predelati le besedila v pravilni slovenščini, kar pomeni da starejši dialekti popolnoma odpadejo.

7 LITERATURA IN VIRI

[1] Jason Brownlee - Machine learning mastery, *Naive Bayes Classifier From Scratch in Python*, `https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/`. (Datum ogleda: 06. 08. 2020.)

[2] Sravani Chintaluri; R. Pallavi Reddy; Kalyani Nara, Text Summarization Based on Genetic Fuzzy Techniques. *ISSN 2319-8885 Vol.04,Issue.29, p. 5709-5714*, sprejeto v objavo.2015

[3] Gene Diaz - Github, stopword-iso, *Slovenian stopwords*, `https://github.com/stopwords-iso/stopwords-sl/blob/master/stopwords-sl.txt`. (Datum ogleda: 06. 08. 2020.)

[4] Sherif Elfayoumy in Jenny Thoppil, A Survey of Unstructured Text Summarization Techniques. *(IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 5, No. 4*, sprejeto v objavo.2014

[5] PadmaPriya G. in K. Duraiswamy, An approach for text summarization using deep learnig algorithm. *Journal of Computer Science 10 (1), p. 1-9*, sprejeto v objavo. 2014

[6] Vishal Gupta in Gurpreet Singh Lehal, A Survey of Text Summarization Extractive Techniques. *JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE, VOL. 2, NO. 3*, sprejeto v objavo. 2010

[7] Peter Hans Luhn, The automatic creation of literature exctracts. *IBM journal*, sprejeto v objavo. 1958

[8] A. Kumar in A. Sharma, Systematic literature review of fuzzy logic based text summarization. *Iranian Journal of Fuzzy Systems Volume 16, Number 5, pp. 45-59*, sprejeto v objavo. 2019

[9] Khosrow Kaikhah, Text Summarization Using Neural Networks. *Second IEEE international conference on inteligent systems*, sprejeto v objavo. 2004

[10] Julian KupiecJan Pedersen;Francine Chen, *A trainable text summarizer*, Xerox Palo Alto research center.

[11] Tomas Mikolov; Quoc V Le; Ilya Sutskever, *Exploiting Similarities among Languages for Machine Translation*, arXiv:1309.4168.

[12] Donald Olding Hebb, *The Organization of Behavior*, New York: Wiley. ISBN 978-1-135-63190-1.

[13] Diederik P. Kingma in Jimmy Ba, Adam: A Method for Stochastic Optimization. *3rd International Conference for Learning Representations, San Diego*, sprejeto v objavo. 2015

[14] Rucha S. Dixit in Prof Dr.S.S.Apte, Improvement of Text Summarization using Fuzzy Logic Based Method . *IOSR Journal of Computer Engineering(IOSRJCE) volume 5, issuse 6, p. 5-10*, sprejeto v objavo.2012

[15] Ladda Suanmali; Naomie Salim; Mohammed Salem Binwahlan, Fuzzy Logic Based Method for Improving Text Summarization. *(IJCSIS) International Journal of Computer Science and Information Security, Vol. 2, No. 1*, sprejeto v objavo. 2009

[16] Rajesh. S. Prasadi; Uplavikar Nitish; Wakhare Sanket, *Feature Based Text Summarization*, Department Of Computer Engineering, University of Pune Pune.

[17] S.A.Babara in Pallavi D.Patilb, Improving Performance of Text Summarization. *International Conference on Information and Communication Technologies (ICICT 2014), Procedia Computer Science 46, p. 354—363*, sprejeto v objavo. 2015

[18] Oguzhan Tas in Farzad Kiyani, A survey of automatic text summarization. *Pressacademia, Procedia, 2nd World Conference on Technology, Innovation and Entrepreneurship, p. 204-213*, sprejeto v objavo. 2017

[19] Chi-Feng Wang, *Towards Science - A Newbie's Guide to Stochastic Gradient Descent With Restarts*, `https://miro.medium.com/max/1142/1*_6nUFOhjPv3ftr8CrrWANQ.png`. (Datum ogleda: 06. 08. 2020.)

[20] *Turtorials point*, `Artificial Neural Network - Building Blocks`. (Datum ogleda: `https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_building_blocks.htm`. 06. 08.)2020

[21] *Wikipedia*, `Information age`. (Datum ogleda: `https://en.wikipedia.org/wiki/Information_Age`. 06. 08.)2020

[22] *Wikibooks - Data Structures (fundimental tools)*, `Linked list`. (Datum ogleda: `https://upload.wikimedia.org/wikibooks/en/d/d6/DataStructuresLinkedListofN.png`. 06. 08.)2020

[23] *Wikipedia*, `Gaussian function`. (Datum ogleda: `https://en.wikipedia.org/wiki/Gaussian_function`. 06. 08.)2020

[24] *Computer Science - University of California, Los Angeles, Kryder's law*, `http://web.cs.ucla.edu/classes/spring14/cs111/scribe/2a/kryder.jpg`. (Datum ogleda: 06. 08. 2020.)

[25] *Codeburst - Objects and Hash Tables in Javascript*, `https://cdn-images-1.medium.com/max/1200/1*wHiuDrmhLsyJbgGWCEBBcQ.png`. (Datum ogleda: 06. 08. 2020.)