# Obrada teksta temeljem modela vreća riječi

**Valjavec, Tena**




SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

# ZAVRŠNI RAD

**Tena Valjavec**

Zagreb, 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

# ZAVRŠNI RAD

Mentor: Student:

Izv. Prof. dr. sc. Tomislav Stipančić, dipl. ing. Tena Valjavec

Zagreb, 2023.

Izjavljujem da sam ovaj rad izradila samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se mentoru dr. sc. Tomislavu Stipančiću na pruženoj pomoći, uloženom vremenu, trudu, povjerenju i strpljenju tijekom izrade ovog rada.

Posebno se zahvaljujem svojoj obitelji na bezuvjetnoj podršci i razumijevanju tijekom studiranja te prijateljima koji su mi sve studentske dane učinili ljepšim i lakšim.

Tena Valjavec

SVEUČILIŠTE U ZAGREBU
**FAKULTET STROJARSTVA I BRODOGRADNJE**
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne i diplomske ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

| Sveučilište u Zagrebu<br>Fakultet strojarstva i brodogradnje | |
|---|---|
| Datum | Prilog |
| Klasa: **602 – 04 / 23 – 6 / 1** | |
| Ur.broj: **15 - 1703 - 23 -** | |

# ZAVRŠNI ZADATAK

Student: **Tena Valjavec** JMBAG: **0035219542**

Naslov rada na hrvatskom jeziku: **Obrada teksta temeljem modela vreća riječi**

Naslov rada na engleskom jeziku: **Text processing based on the bag of words model**

Opis zadatka:

U računalnoj obradi podataka tekst je najčešći oblik u kojem se pojavljuju informacije. Jedan od jednostavnijih modela za obradu prirodnog jezika (engl. Natural Language Processing - NLP) naziva se vreća riječi (engl. Bag of Words). Osim kod obrade prirodnog jezika ovaj model se koristi i za pretraživanje te pronalaženje informacija u tekstu (eng. Information Retrieval - IR). U ovom je modelu tekst, kao što je rečenica ili više njih, predstavljen kao mnoštvo povezanih riječi, zanemarujući pritom gramatiku i red riječi, a uvažavajući višestrukost ponavljanja neke od riječi.

U radu je potrebno koristeći model vreće riječi analizirati proizvoljni tekst na automatizirani način, te:

- proučiti i dati pregled najčešće korištenih NLP tehnika,
- pretvoriti sve riječi u tekstu da počinju s malim slovom,
- međusobno odvojiti riječi u tekstu (separacija),
- prebrojati riječi u tekstu,
- sortirati riječi silaznim redom,
- kreirati matricu vektora za analizirane rečenice,
- proučiti Tf-idf funkciju za pronalaženje informacija te je usporediti s modelom vreće riječi.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan: 30. 11. 2022.

Zadatak zadao: Doc. dr. sc. Tomislav Stipančić

Datum predaje rada:
**1. rok:** 20. 2. 2023.
**2. rok (izvanredni):** 10. 7. 2023.
**3. rok:** 18. 9. 2023.

Predviđeni datumi obrane:
**1. rok:** 27. 2. – 3. 3. 2023.
**2. rok (izvanredni):** 14. 7. 2023.
**3. rok:** 25. 9. – 29. 9. 2023.

Predsjednik Povjerenstva: Prof. dr. sc. Branko Bauer

# Sadržaj

# POPIS SLIKA

# SAŽETAK

Tema ovog rada je proučiti jedan od jednostavnijih modela za obradu prirodnog jezika (engl. Natural Language Proccesing - NLP), a to je model „vreća riječi“ (engl. Bag of Words, skraćeno BoW). Model i njegova implementacija razvija se u programskom jeziku Python, a za izvedbu koda koristi se biblioteka NLTK. U prvom dijelu rada objašnjena je teorijska osnova za razumijevanje programa te su dane na uvid neke od najčešće korištenih NLP tehnika, a u drugom dijelu je razvijen kod te je analiziran odabran tekst korištenjem modela „vreća riječi“. Naposljetku je opisana funkcija „Tf-idf“ te je prikazana usporedba navedene funkcije i modela „vreća riječi“.

Ključne riječi: NLP, umjetna inteligencija, Python, vreća riječi

# SUMMARY

Main topic of this assignment is to ponder one of the easiest model for interpretation of Natural Language Processing or NLP for short. Main model is called BoW or „Bag of Words“. Implementation of BoW is done by special programming language named Python and coding part is done by NLTK library. First part of this assignment explains the basic theory to understand the program and there were given some of the most used NLP techniques. Developmenting codes and analysis done by BoW technique was explained in the second part. Finally, explaining and comparing „Tf-idf“ function and BoW model sums up this assignment into one unit.

Key words: NLP, artificial intelligence, Python, Bag of Words

# 1. UVOD

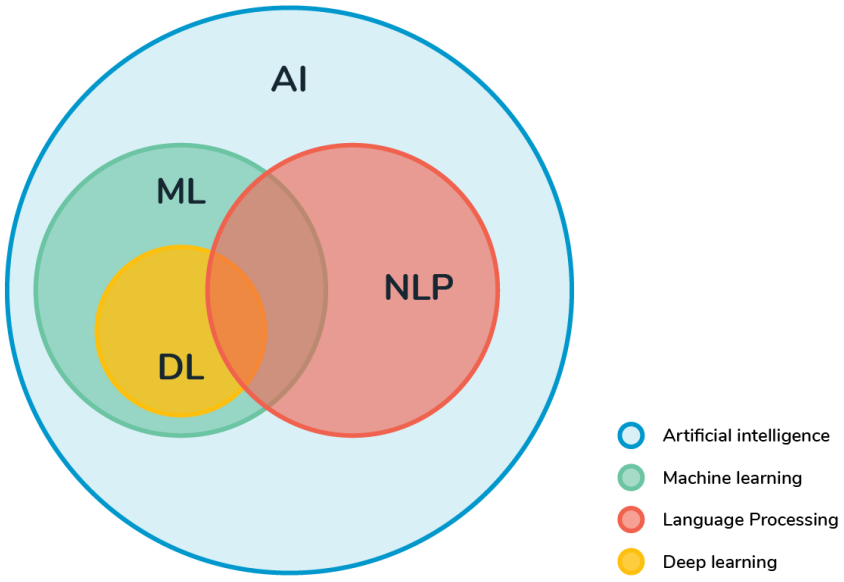
Danas je tekst najčešći oblik u kojem se pojavljuju informacije, bilo u pisanom ili govornom obliku. Ljudi mogu razumjeti tekst, odnosno informacije koje se u njemu javljaju i složiti kontekst odnosno sažetak nekog pročitanog teksta, no kad govorimo o strojevima, stvar je malo drugačija obzirom da oni ne mogu obraditi tekstualne podatke u izvornom obliku već je potrebno raščlaniti tekst u numerički format koji stroj lako čita. To je ujedno i ideja koja stoji iza obrade prirodnog jezika. Postoje razni načini obrade prirodnog jezika, a u ovom će radu ukratko biti pojašnjene najčešće korištene NLP tehnike, tj. tehnike obrade prirodnog jezika te će se detaljnije obraditi jedan od jednostavnijih modela „vreća riječi“ (Bag of Words) na primjeru analize teksta.

# 2. TEORIJSKA OSNOVA RADA

## 2.1. Umjetna inteligencija

Umjetna inteligencija (UI, odnosno akronim AI prema engl. izrazu Artificial Intelligence) je simulacija procesa ljudske inteligencije od strane strojeva, odnosno podrazumijeva takav razvoj računalnih sustava da oni mogu izvršavati zadatke koji zahtijevaju ljudsku inteligenciju. Primjeri takvih zadataka jesu vizualna percepcija, prepoznavanje govora, obrada prirodnog jezika, automatsko prevođenje govore te mnogi drugi, a neki od primjera aplikacija temeljenih na umjetnoj inteligenciji jesu autonomna vozila, navigacijski sustavi, human vs. computer igrice i slično. Način rada umjetne inteligencije je taj da se u sustav unose veće količine podataka koje sustav zatim analizira i koristi obrasce za analiziranje budućih aktivnosti. Upravo taj rad s velikom količinom podataka dajući pritom dosljedne rezultate jedna je od glavnih prednosti umjetne inteligencije.

Možemo razlikovati jaku i slabu umjetnu inteligenciju pri čemu strojevi sa slabom umjetnom inteligencijom mogu reagirati na neke specifične situacije, ali ne mogu misliti sami za sebe dok strojevi sa jakom umjetnom inteligencijom mogu djelovati pa čak i razmišljati kao ljudi. [1]




**Slika 1. Područja umjetne inteligencije [2]**

## 2.2. Obrada prirodnog jezika

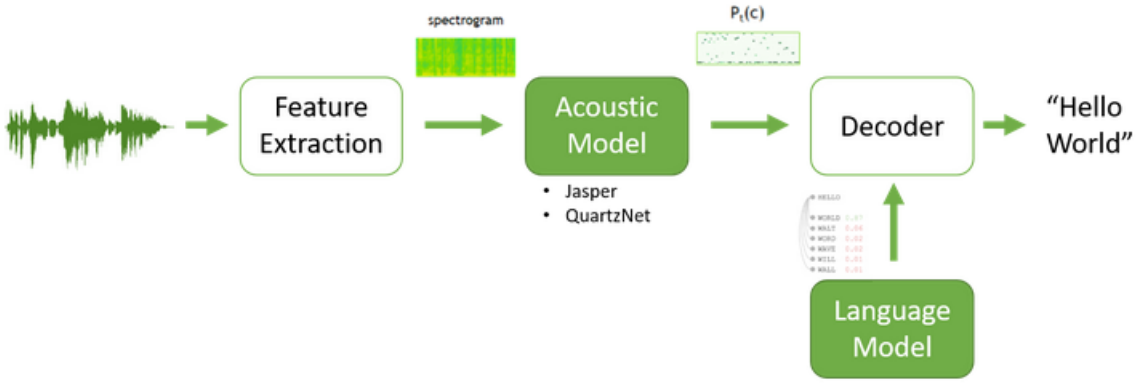
Jedna od grana umjetne inteligencije je obrada prirodnog jezika (NLP). To je dio umjetne inteligencije koji se bavi proučavanjem interakcije između ljudi i računala te načinima kako procesirati i analizirati velike količine podataka iz prirodnog jezika pomoću računala. Također se bavi razvojem sposobnosti računalnih strojeva da razumiju tekst i izgovorene riječi na način kako to rade ljudska bića.

NLP kombinira računalnu lingvistiku, odnosno jezične zakonitosti ljudskog govora s modelima strojnog i dubokog učenja. Zajedno kombinirane, ove tehnike daju mogućnost računalima da obrađuju ljudski jezik u obliku teksta ili govora te da razumiju njegovo puno značenje zajedno s namjerom i emocijom govornika. U primjeni NLP pokreće računalne programe koji prevode tekst s jednog jezika na drugi, sažimaju velike količine teksta ili odgovaraju na glasovne naredbe.

Jedna od stvari koje računalnim sustavima veoma otežavaju razumijevanje ljudskog jezika je njegova dvosmislenost, odnosno korištenje metafora, sarkazma, homofona i homonima, te naglasak. To su neke od nepravilnosti ljudskog jezika koje usporavaju i otežavaju rad algoritma koji određuje namjeru zvuka ili teksta. Neka moguća rješenja tog problema koja omogućuju računalima shvatiti ljudski unos jesu:

- Prepoznavanje govora, tzv. „govor-u-tekst“

  Ovaj zadatak omogućuje pretvaranje glasovnih podataka u tekstualne podatke. Prepoznavanje govora potrebno je za svaku aplikaciju koja prati glasovne naredbe ili odgovara na izgovorena pitanja. Primjer takvih aplikacija danas jesu svima poznati virtualni asistenti Siri (Apple), Alexa (Amazon)… koji uz jezično pravilnu naredbu ispunjavaju želje svojih korisnika.




**Slika 2. Način rada metode "govor-u-tekst" [7]**

- Označavanje dijela govora, tzv. „gramatičko označavanje“
  Ovo je postupak koji se temelji na određivanju dijela govora neke riječi ili pak teksta na temelju njegove upotrebe i konteksta.
- Višeznačnost riječi
  Ovim postupkom se procesom semantičke analize za višeznačne riječi bira ono značenje riječi koje ima najviše smisla u danom kontekstu.
- Koreferentna rezolucija
  Navedeni postupak je zadatak identificiranja odnose li se dvije ili više riječi na isti entitet. Najčešći primjer je određivanje osobe ili predmeta na koje se određena zamjenica odnosi (npr. „ona“ - Marina). [3]
- Generiranje prirodnog jezika
  Ovaj je postupak suprotan onome „govor-u-tekst“, odnosno to je zadatak pretvaranja strukturiranih informacija u ljudski jezik.

NLP danas ima veoma široku primjenu, kao što su otkrivanje neželjene pošte pomoću klasifikacije teksta e-pošte u potrazi za jezikom koji upućuje na spam, a to su loša gramatika, prijeteći jezik, velika uporaba financijskih izraza itd. Nadalje se koristi pri strojnom prevođenju, čega je najbolji primjer Google prevoditelj. Dobro strojno prevođenje osim zamjene riječi na jednom jeziku riječju na drugom jeziku također obuhvaća razumijevanje značenja i tona ulaznog jezika kako bi prijevod bio što vjerodostojniji. Jedna od primjena je također kroz virtualne agente i chatbotove. Prethodno navedeni virtualni agenti Siri i Alexa koriste prepoznavanje govora za prepoznavanje glasovnih naredbi i generiranje prirodnog jezika kako bi odgovarajućom radnjom ili komentarom odgovorili na korisnikov zahtjev. Kako bi NLP mogao obavljati navedene i slične zadatke, postoje određene tehnike koje olakšavaju taj rad. U sljedećem su poglavlju dane na uvid neke od tih najčešće korištenih tehnika. [3][4]


**Slika 3. Primjer virtualnog asistenta Siri (Apple) [8]**

### *2.2.1. Analiza sentimenta*

Ova se tehnika temelji na disekciji podatka (bilo tekstualnih, bilo glasovnih) kako bi se utvrdilo jesu li ti podaci pozitivni, negativni ili neutralni.

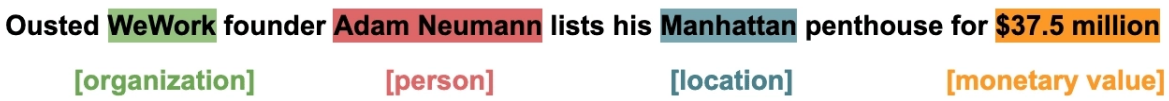

**Slika 4. Primjer analize osjećaja [4]**

Kao što je vidljivo na slici [slika 4], svaka se izjava označava „osjećajem“, a zatim se stvara zbroj osjećaja svih izjava u zadanom skupu podataka. Na taj se način mogu velike količine

podataka u obliku recenzija, povratih informacija kupaca ili reakcija na društvenim mrežama pretvoriti u kvantificirane rezultate. Upravo u tu svrhu se analiza osjećaj najviše i koristi, kako bi se došlo do zaključka jesu li i koliko kupci/korisnici neke usluge bili njome zadovoljni. Ručno pretraživanje se u tom slučaju čini kao prikladno rješenje, međutim kad se radi o velikom broju recenzija i komentara, gotovo je nemoguće pojedinačno analizirati sve podatke i izvršiti analizu. Stoga se do reprezentativne analize zadovoljstva korisnika dolazi automatskim putem, odnosno NLP tehnikom analize osjećaja. [5]

### *2.2.2. Prepoznavanje imenovanog entiteta*

Prepoznavanje imenovanog entiteta (NER, skraćeno od engl. izraza Named Entity Recognition) je tehnika obrade prirodnog jezika koja se temelji na označavanju imenovanih identiteta unutar zadanog teksta te njihovom izdvajanju za daljnju analizu. Na slici [slika 5] prikazan je primjer rada ove tehnike.




**Slika 5. Primjer tehnike prepoznavanja imenovanog entiteta [4]**

Kao što je vidljivo iz primjera na slici, ova metoda označava identitete koji se pojavljuju u zadanom tekstu. To mogu biti organizacije, ljudi, materijalna vrijednost, lokacija, vrijeme itd. Isto tako metoda prati koliko se puta određeni identiteti pojavljuju unutar zadanih podataka. Ta je metoda upravo i dizajnirana kako bi omogućila prikaz „kritičnih“ informacija u nestrukturiranom skupu podataka te dala uvid u njihove odnose.

Postoje tri pristupa NER metodi:

- Sustav temeljen na rječniku

  Ovo je najjednostavniji NER pristup koji koristi velik rječnik, a funkcionira na način da sustav provjerava nalazi li se određeni entitet koji se pojavljuje u tekstu i u vokabularu. Da bi taj pristup mogao učinkovito raditi, potrebna je stalna nadogradnja skupa podataka vokabulara.

- Sustav temeljen na pravilima

  Ovo je pristup koji se temelji na izdvajanju informacija iz zadanog skupa podataka na temelju unaprijed definiranih pravila. Razlikujemo dva skupa pravila, a to su pravila

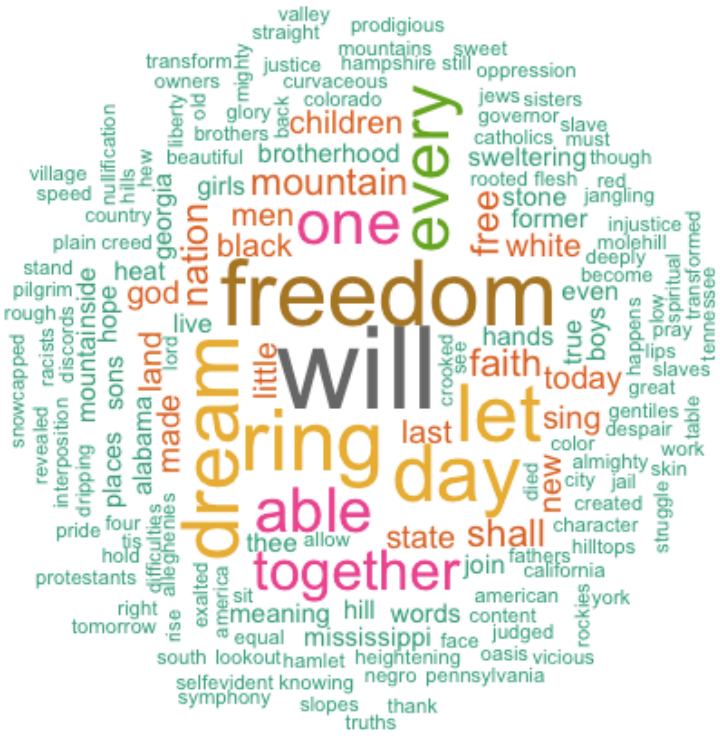
temeljena na uzorku (slijede morfološki obrazac odnosno niz riječi koji je korišten u dokumentu) i pravila temeljena na kontekstu (ovise o značenju i kontekstu riječi u dokumentu).

- Sustav temeljen na strojnom učenju

  U ovom pristupu se za otkrivanje entiteta koristi statističko modeliranje. Ovaj model može prepoznati iste entitete i uz male varijacije pa su time riješeni neki od nedostataka prethodno navedenih pristupa. [6]

### *2.2.3. Oblak riječi (engl. WordCloud)*

Kao što sam naslov i kaže, radi se o oblaku sastavljenom od riječi, kao što prikazuje slika [slika 6].




**Slika 6. WordCloud [10]**

Kako možemo vidjeti na prethodnoj slici, riječi se u oblaku pojavljuju u različitim veličinama fontova. Pojašnjenje toga leži u samoj definiciji te tehnike koja kaže da je Oblak riječi vizualna NLP tehnika koja ulazne tekstualne podatke predstavlja u obliku slika. Računalni sustav zadane podatke generira u sliku načinjenu od korištenih riječi pri čemu veličina svake riječi predstavlja njezinu učestalost i važnost u ulaznom skupu. Budući da ljudski mozak preferira slikovni pristup, ovo je često korištena tehnika. [10]

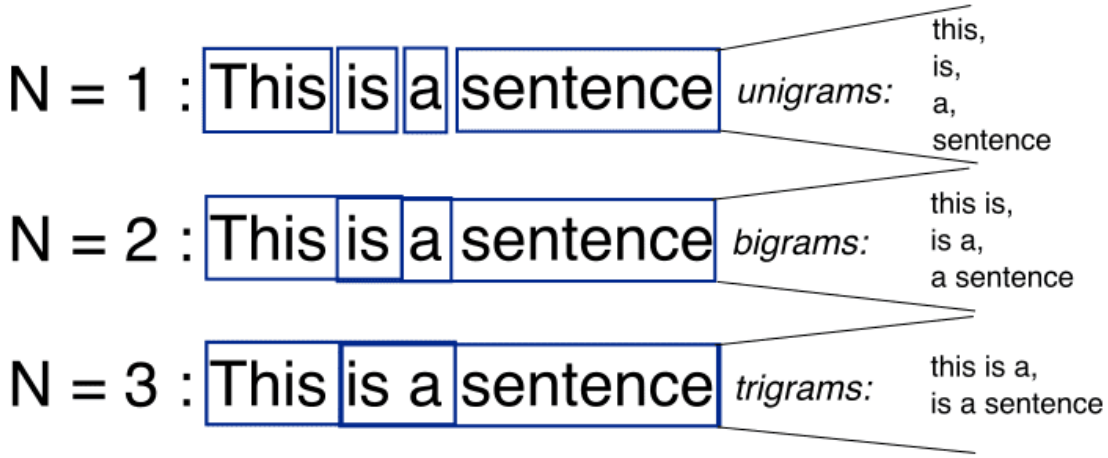
Ova se tehnika najčešće implementira u programskom jeziku Python, a biblioteke potrebne za generiranje oblaka riječi koristeći Python jesu Wordcloud, Matplotlib i Pandas biblioteke. Najčešća primjena ove NLP tehnike je analiza podataka dobivenih sa društvenih mreža, recenzija, informacija i komentara kupaca ili zaposlenika.

### *2.2.4. Sažetak teksta*

Ova se tehnika temelji na sažimanju teksta odnosno rastavljaju zadanog skupa podataka bilo da se radi o medicinskom, tehničkom, znanstvenom ili nekom drugom žargonu. Pritom se tekst rastavlja na najosnovnije pojmove kako bi bio što razumljiviji. Iako su ljudski jezici veoma komplicirani, korištenjem algoritama za povezivanje imenica i glagola brzo možemo dobiti generirani sažetak ulaznih podataka. Dobiveni sažetak daje ključne informacije o ulaznom tekstu te ni na koji način ne mijenja njegovo značenje. Dva su načina sažimanja teksta, a to su ekstraktno sažimanje kod kojeg dobivamo sažetak koji se sastoji isključivo od rečenica iz ulaznih podataka i apstraktno sažimanje koje mijenja riječi u tekstu i po potrebi u sažetak dodaje nove (ne već korištene u zadanom tekstu) riječi. [4]

### *2.2.5. N-grams*

„N-grams“ je tehnika koja izdvaja skup od $N$ broja riječi koje su kontekstno zavisne i više puta se pojavljuju u ulaznom skupu podataka. Provodi se na način da se iz tekstualnog zapisa biraju riječi koje se češće ponavljaju u obliku jedne riječi (engl. *unigrams*), u obliku dvije povezane riječi (engl. *bigrams*) ili u obliku tri povezane riječi (engl. *trigrams*). [11]



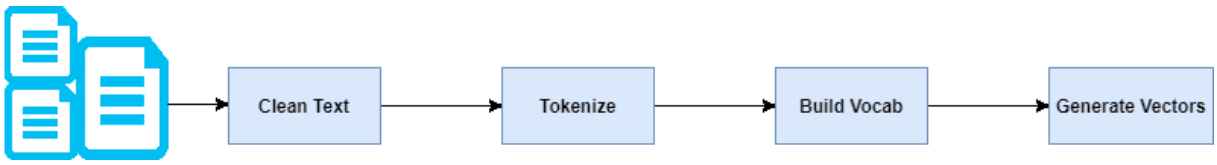

**Slika 7. Primjer rada N-grams tehnike [12]**

Neke od primjena „N-grams“ tehnike u obradi prirodnog jezika jesu provjera gramatike, automatsko prepoznavanje nepravilno napisanih riječi, automatsko popunjavanje rečenica i mnoge druge. Za početak je potrebno utrenirati model s velikom bazom podataka nakon čega će predviđanje riječi nakon neke određene riječi imati veću točnost. U modeliranju skupa riječi korištenjem 2-grams modela, za predviđanje trenutne riječi u obzir se uzima samo prethodna riječ. Sukladno tome pri 3-grams modelu u obzir se uzimaju dvije prethodne riječi, u 4-grams modelu tri prethodne riječi itd. Kako se računalni sustavi uglavnom oslanjaju na bazu podataka i daju veću vjerojatnost češće pojavljivanim izrazima i oni mogu pogriješiti. Da bi se to izbjeglo preporučljivo je koristiti „N-grams“ većeg stupnja, od 3-grams pa nadalje. [12]

## 2.3. Model „vreća riječi“

Kako danas velika većina računalnih sustava odnosno algoritama radi s numeričkim podacima, prijeka je potreba za tehnikama koje će skupove ulaznih podataka tekstualnog oblika pretvoriti u brojeve da bi bili pogodni za analizu. Do danas su razvijene mnoge funkcije i tehnike za pretvorbu teksta u brojčane vrijednosti, a najpoznatije od njih su „vreća riječi“, Tf-idf i word2vec. U ovom će poglavlju detaljnije biti obrađen model „vreća vriječi“.

U tehničkom smislu možemo reći da „vreća riječi“ predstavlja pojednostavljen prikaz tekstualnog dokumenta koji je shvatljiv računalnim sustavima. Ime dolazi od načina na koji sam model radi a to je da zadani tekst prezentira doslovno kao vreću riječi od kojih se tekst sastoji zanemarujući pritom i redoslijed riječi i gramatiku, a uzimajući u obzir višestruko ponavljanje riječi. Dakle bit ove metode je zabilježiti koliko se puta neka riječ pojavljuje u dokumentu neovisno o njenoj lokaciji pri čemu se najveći značaj pridaje riječi koja se ponavlja najviše puta i suprotno. Korake od kojih se sastoji model „vreća riječi“ prikazani su na slici [slika 8].




**Slika 8. Koraci BoW metode [13]**

Prije same implementacije metode potrebno je prikupiti podatke za analizu. To može biti jedna rečenica, dokument, knjiga, čak i cijela biblioteka knjiga. Nakon toga riječi se pretvaraju u

tokene i kreira se rječnik koji se sastoji od svih riječi pojavljivanih u ulaznom dokumentu. Ti se rječnici potom pretvaraju u vektore duljine ekvivalentne rječniku. Pri primjenjivanju metode na veće skupove podataka ti vektori rječnika mogu postati iznimno veliki, a često se sastoje od mnogo nula pa se nazivaju rijetkim vektorima (engl. sparse vectors). Neki od načina da se ti rječnici smanje jesu normalizacija, ignoriranje interpunkcijskih znakova te uklanjanje često ponavljanih riječi nebitnih za kontekst poput nekih veznika, priloga i slično. Sljedeći je korak analiza teksta odnosno provjera pojavljuju li se i koliko puta riječi iz rječnika. Na temelju toga možemo za svaku rečenicu zasebno prikazati matricu vektora riječi iz kreiranog rječnika. Ukoliko se neka riječ iz rječnika ne pojavljuje u analiziranoj rečenici, u vektoru je vrijednost te riječi jednaka 0, a u suprotnom vrijednost je jednaka broju ponavljanja te riječi u rečenici. [13] Ovaj je model najbolje opisati na samom primjeru analize određenog teksta što će biti odrađeno u sljedećem poglavlju.

# 3. IZVEDBA ZADATKA

U ovom je poglavlju razrađen praktičan dio rada tj. prikazana je izvedba zadatka u programskom jeziku Python. U prvom dijelu je obrađena teorijska osnova Pythona i biblioteka potrebnih za izvedbu zadatka, a dalje je prikazana analiza odabranog teksta kao primjer primjene zadanog modela „vreća riječi“.

## 3.1. Programski jezik Python

Python je interpretacijski, objektno orijentiran programski jezik opće namjene, ali i visoke razine, a to mu omogućuje korištenje velikih broja biblioteka. Može se koristiti za pisanje jednostavnih, ali i kreiranje velikih složenih programa. Pogodan je za učenje osnova programiranja. Brz je i otkriva pogreške. Prolazi liniju po liniju koda što olakšava pregled globalnih i lokalnih varijabli te pronalaženje greške, a radi svoje jednostavnosti sintakse i brzine rada prikladan je za razvoj i razna istraživanja. Python je besplatan softver, a podržavaju ga gotovo svi operativni sustavi, što mu omogućuje široku primjenu. U Pythonu je moguće analizirati podatke, automatizirati dijelove procesa te razvijati web stranice i različite softwere. Neke od najčešćih primjena Pythona jesu upravo analitika i manipulacija podataka, strojno i duboko učenje, vizualizacija podataka, razvoj web aplikacija i mnoge druge. [15]

### *3.1.1. Python biblioteke*

Za lakšu implementaciju koda, Python koristi brojne razvijene biblioteke koje smanjuju opsežnost koda i uveliko olakšavaju njegovo korištenje. Te su biblioteke razvijene za rješavanje specifičnih problema i zadatka. Postoje brojne biblioteke za različitu namjenu. Najpoznatije od njih su Pandas, biblioteka za analizu podataka, NumPy, biblioteka za izračunavanje i obradu višedimenzionalnih elemenata polja, Matplotlib, biblioteka za vizualizaciju podataka itd.
Ovaj je zadatak u nastavku riješen korištenjem biblioteka NLTK i Scikit-learn. [15]

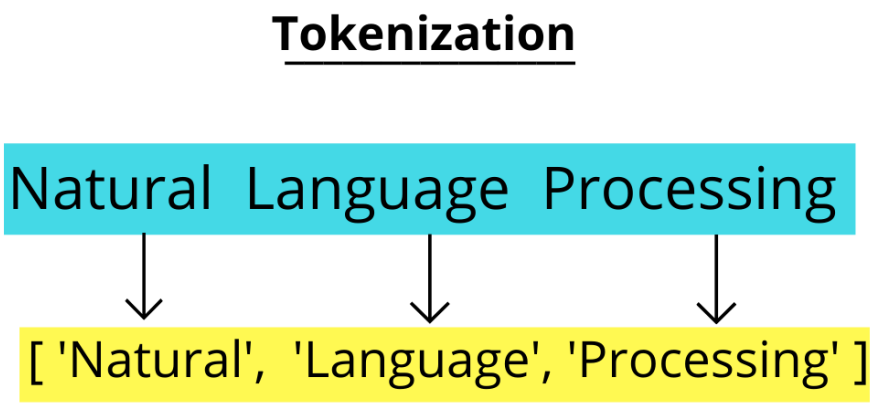
#### *3.1.1.1. NLTK biblioteka*

NLTK (engl. Natural Language Toolkit) je biblioteka za rad ljudskim jezikom. Sastoji se od manjih biblioteka koji omogućuju izradu programa za obradu prirodnog jezika programskim jezikom Python. To su biblioteke koje omogućuju analizu emocija, pretvorbu govora u tekst,

odvajanje riječi i slično. Koristeći NLTK biblioteku mogu se implementirati različiti postupci obrade prirodnog jezika od kojih su neki lematizacija, tokenizacija, prepoznavanje sinonima i slično, a omogućuju analizu velikih skupova ulaznih podataka. [16]

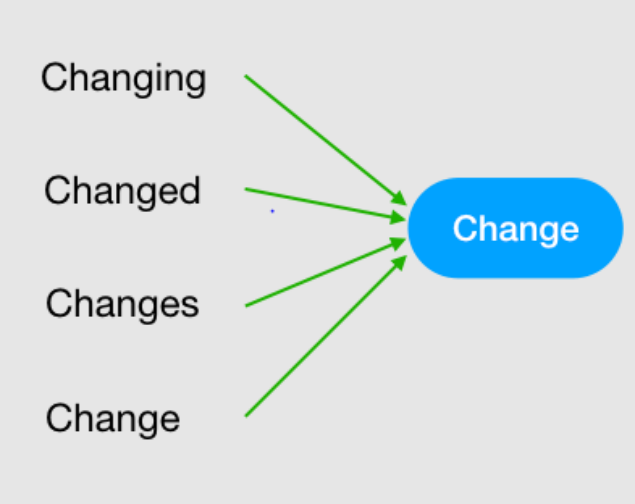
Tokenizacija je postupak kojim se ulazni podaci rastavljaju na pojedinačne tokene (riječi, rečenice ili znakove) od kojih su sastavljeni. [18]

Primjer tokenizacije prikazan je na slici [slika 9].




**Slika 9. Primjer tokenizacije [18]**

Lematizacija je korjenovanje riječi odnosno svođenje riječi na njen korijen. Postupak lematizacije riječi daje značaj koji ima smisla, a za to koristi ugrađen rječnik i morfološku analizu. Često korišten WordNetLemmatizer je opširna leksička baza engleskog jezika kojoj je cilj uspostavljanje semantičkih veza između riječi. [18] Princip funkcioniranja procesa lematizacijen vidljiv je na slici [slika 10].




**Slika 10. Primjer lematizacije [18]**

#### *3.1.1.2. Scikit-learn*

Scikit-learn ili skraćeno sklearn je biblioteka za strojno učenje u Pythonu koja daje širok izbor algoritama za strojno učenje. Neki od tih algoritama su nadzirano učenje (npr. regresija i klasifikacija) i nenadzirano učenje (npr. smanjenje dimenzionalnosti). Tipične značajke ove biblioteke jesu jednostavni, a učinkoviti alati za analizu podataka, široka dostupnost i mogućnost korištenja u različite svrhe. [17]

## 3.2. Analiza teksta pomoću modela „vreća riječi“

U svrhu pojašnjenja načina rada navedenog modela koristi se članak objavljen na portalu *Science daily* na temu prikaza vizualizacije potrošnje energije nekog grada. Članak je objavljen 22. srpnja 2021. godine, a kao izvor navodi se Sveučilište Pittsburgh. U prilogu se nalazi članak u cijelosti, a za analizu teksta koristi se ovaj odlomak [19]:

“The researchers used publicly available Geographic Information System (GIS) data and street-level images to develop their UBEM, then created 20 archetypes of buildings that comprised eight commercial use types. The buildings were sorted into the groups based on categories including use type and construction period.

With street-level images to determine the building material, window-to-wall ratio, and number of floors, and LiDAR data from the U.S. Geological Survey to determine building height, the researchers were able to simulate and map the annual energy use intensity of 209 structures in Pittsburgh. When they validated their findings using other publicly available data, they had just a 7 percent error rate. Though it's currently mostly guided by the researchers looking at the images, the researchers hope that this modeling framework can eventually take advantage of machine learning to more quickly analyze and categorize building images.

The focus on commercial buildings, as well, was an important addition to the field of research.”

Za početak je potrebno implementirati biblioteku NLTK koja svojim alatima omogućuje stvaranje programa za analizu prirodnog jezika te umetnuti željeni tekst za analizu. Taj je postupak prikazan na slici [slika 11].

```
import nltk
tekst="The researchers used publicly available Geographic Information System GIS data and street
-level images to develop their UBEM, then created 20 archetypes of buildings that comprised eight
commercial use types. The buildings were sorted into the groups based on categories including us
e type and construction period. With street-level images to determine the building material, window-
to-wall ratio, and number of floors, and LiDAR data from the U.S. Geological Survey to determine b
uilding height, the researchers were able to simulate and map the annual energy use intensity of 20
9 structures in Pittsburgh. When they validated their findings using other publicly available data, the
y had just a 7 percent error rate. Though it's currently mostly guided by the researchers looking at t
he images, the researchers hope that this modeling framework can eventually take advantage of m
achine learning to more quickly analyze and categorize building images. The focus on commercial
buildings, as well, was an important addition to the field of research."
```

**Slika 11. Unos teksta za analizu**

Prvi korak analize teksta je pretvoriti sve riječi u tekstu da počinju malim slovom. Funkcija koja to radi je *text.lower(),* a prikazana je na slici [slika 12].

```
tekst=tekst.lower()
tekst
"the researchers used publicly available geographic information system gis data and street-level im
ages to develop their ubem, then created 20 archetypes of buildings that comprised eight commerc
ial use types. the buildings were sorted into the groups based on categories including use type and
construction period. with street-level images to determine the building material, window-to-wall ratio
, and number of floors, and lidar data from the u.s. geological survey to determine building height, th
e researchers were able to simulate and map the annual energy use intensity of 209 structures in p
ittsburgh. when they validated their findings using other publicly available data, they had just a 7 per
cent error rate. though it's currently mostly guided by the researchers looking at the images, the res
earchers hope that this modeling framework can eventually take advantage of machine learning to
more quickly analyze and categorize building images. the focus on commercial buildings, as well,
was an important addition to the field of research."
```

**Slika 12. Sve riječi napisane malim slovom**

Idući korak u analizi je odvajanje riječi u tekstu odnosno separacija. To ćemo učiniti funkcijom *word_tokenize* koju je potrebno pozvati iz NLTK biblioteke. Prije toga ćemo očistiti ulazni tekst tako da uklonimo sve točke i zareze iz rečenica što nam omogućuje implementirana biblioteka *re*. Postupak je prikazan na slici [slika 13].

```
import re
remove="[.,]"
matches=re.sub(remove,"",tekst)
print(matches)
the researchers used publicly available geographic information system gis data and street-level im
ages to develop their ubem then created 20 archetypes of buildings that comprised eight commerci
al use types the buildings were sorted into the groups based on categories including use type and c
onstruction period with street-level images to determine the building material window-to-wall ratio a
nd number of floors and lidar data from the us geological survey to determine building height the re
searchers were able to simulate and map the annual energy use intensity of 209 structures in pitts
burgh when they validated their findings using other publicly available data they had just a 7 percent
error rate though it's currently mostly guided by the researchers looking at the images the research
ers hope that this modeling framework can eventually take advantage of machine learning to more
quickly analyze and categorize building images the focus on commercial buildings as well was an i
mportant addition to the field of research
```

**Slika 13. Očišćen tekst**

Iz tako dobivenog, očišćenog teksta možemo razdvojiti riječi, odnosno svaku riječ odvojiti zasebno te pritom dobivamo polje riječi od kojih se sastoji ulazni tekst.

```
from nltk.tokenize import word_tokenize
result=word_tokenize(matches)
print(result)
['the', 'researchers', 'used', 'publicly', 'available', 'geographic', 'information', 'system', 'gis', 'data', 'an
d', 'street-level', 'images', 'to', 'develop', 'their', 'ubem', 'then', 'created', '20', 'archetypes', 'of', 'buildin
gs', 'that', 'comprised', 'eight', 'commercial', 'use', 'types', 'the', 'buildings', 'were', 'sorted', 'into', 'the'
, 'groups', 'based', 'on', 'categories', 'including', 'use', 'type', 'and', 'construction', 'period', 'with', 'stre
et-level', 'images', 'to', 'determine', 'the', 'building', 'material', 'window-to-wall', 'ratio', 'and', 'number',
'of', 'floors', 'and', 'lidar', 'data', 'from', 'the', 'us', 'geological', 'survey', 'to', 'determine', 'building', 'heig
ht', 'the', 'researchers', 'were', 'able', 'to', 'simulate', 'and', 'map', 'the', 'annual', 'energy', 'use', 'inten
sity', 'of', '209', 'structures', 'in', 'pittsburgh', 'when', 'they', 'validated', 'their', 'findings', 'using', 'other',
'publicly', 'available', 'data', 'they', 'had', 'just', 'a', '7', 'percent', 'error', 'rate', 'though', 'it', "'s", 'current
ly', 'mostly', 'guided', 'by', 'the', 'researchers', 'looking', 'at', 'the', 'images', 'the', 'researchers', 'hope'
, 'that', 'this', 'modeling', 'framework', 'can', 'eventually', 'take', 'advantage', 'of', 'machine', 'learning',
'to', 'more', 'quickly', 'analyze', 'and', 'categorize', 'building', 'images', 'the', 'focus', 'on', 'commercial'
, 'buildings', 'as', 'well', 'was', 'an', 'important', 'addition', 'to', 'the', 'field', 'of', 'research']
```

**Slika 14. Separacija teksta**

Iz slike [slika 14] vidimo da nam funkcija word_tokenize daje polje u kojem se pojavljuju sve riječi iz teksta, redoslijedom kao u tekstu, a svaka riječ predstavlja token za sebe. Nakon razdvajanja možemo prebrojati sve riječi u tekstu. Na sljedećoj slici [slika 15] prikazana je *for* petlja koja prolazi kroz polje riječi i broji koliko se puta riječ ponavlja u tom skupu.

```
count=0
for i in range(0,len(result)):
    for j in range(0,len(result)):
        if(result[i]==result[j]):
            count=count+1
    print(result[i], count)
    count=0
```

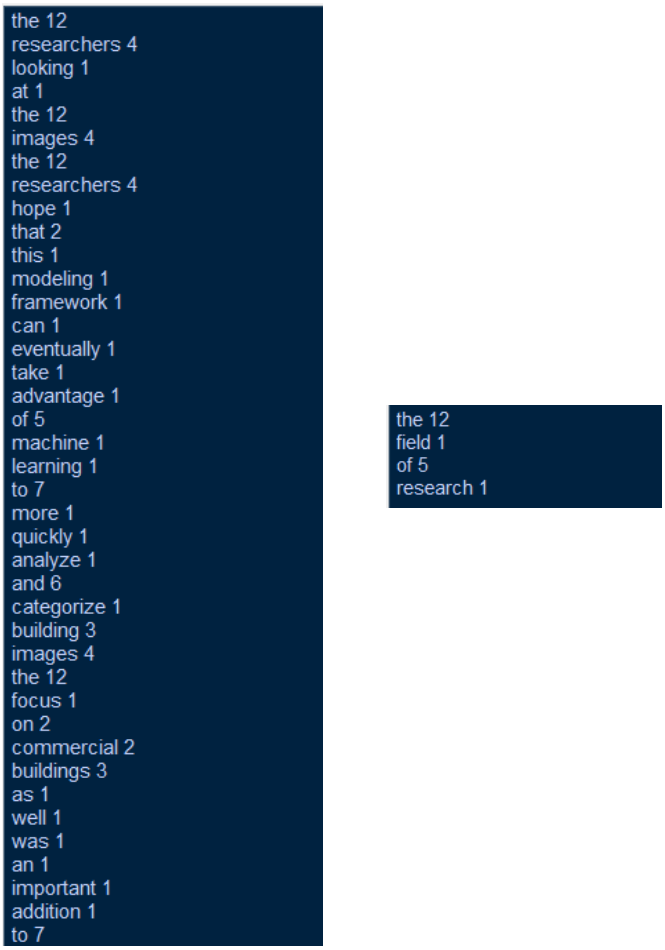
**Slika 15. For petlja za brojenje riječi**

Nakon provođenja programa za prebrojavanje riječi dobivamo ispis riječi redoslijedom kojim se pojavljuju i tekstu i brojem ponavljanja.

```
the 12
researchers 4
used 1
publicly 2
available 2
geographic 1
information 1
system 1
gis 1
data 3
and 6
street-level 2
images 4
to 7
develop 1
their 2
ubem 1
then 1
created 1
20 1
archetypes 1
of 5
buildings 3
that 2
comprised 1
eight 1
commercial 2
use 3
types 1
the 12
buildings 3
were 2
sorted 1
into 1
the 12
groups 1
```

```
based 1
on 2
categories 1
including 1
use 3
type 1
and 6
construction 1
period 1
with 1
street-level 2
images 4
to 7
determine 2
the 12
building 3
material 1
window 1
to 7
wall 1
ratio 1
and 6
number 1
of 5
floors 1
and 6
lidar 1
data 3
from 1
the 12
us 1
geological 1
survey 1
to 7
determine 2
building 3
height 1
the 12
researchers 4
were 2
```

```
able 1
to 7
simulate 1
and 6
map 1
the 12
annual 1
energy 1
use 3
intensity 1
of 5
209 1
structures 1
in 1
pittsburgh 1
when 1
they 2
validated 1
their 2
findings 1
using 1
other 1
publicly 2
available 2
data 3
they 2
had 1
just 1
a 1
7 1
percent 1
error 1
rate 1
though 1
it 1
's 1
currently 1
mostly 1
guided 1
by 1
```



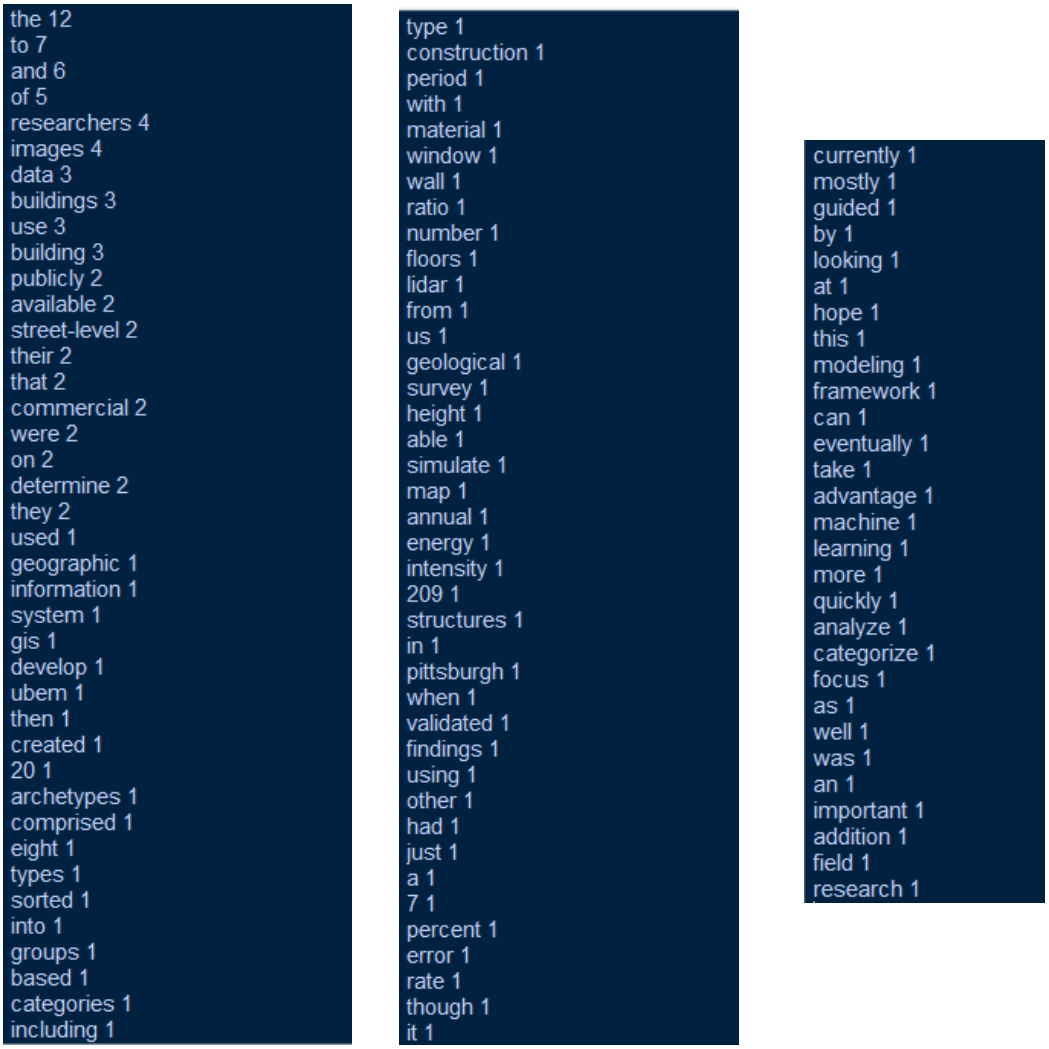

**Slika 16. Prebrojene riječi iz teksta**

Kako možemo vidjeti iz prethodne slike, neke su riječi zastupljenije odnosno pojavljuju se više puta od ostalih. Kako bismo znali redoslijed učestalosti riječi, odnosno koje se riječi pojavljuju najviše, a koje najmanje puta možemo sortirati riječi silaznim redom, tj. načiniti popis riječi od onih najčešćih pa do najrjeđih. Na slici [slika 17] prikazan je kod za sortiranje riječi silaznim redom.

```
from collections import Counter
word_counts=Counter(result)
for result, count in word_counts.most_common():
    print(result,count)
```

**Slika 17. Sortiranje riječi silaznim redom**

Pokretanjem ovih linija koda dobivamo listu riječi poredanih od onih najviše do onih najmanje korištenih, što je prikazano na slici [slika 18].




**Slika 18. Riječi sortirane silaznim redom**

Zadnji korak analize teksta je kreiranje matrice vektora za analizirane rečenice. Da bismo to mogli učiniti, za početak je potrebno kreirati „rječnik“ koji se sastoji od svih riječi koje se barem jednom pojavljuju u ulaznom tekstu. Stvaranje rječnika prikazano je na slici [slika 19]. Potrebno je pozvati *fit* metodu koja tokenizira tekst i stvara vokabular odnosno rječnik jedinstvenih riječi koje se pojavljuju u zadanom tekstu.

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer=CountVectorizer()
vectorizer.fit([tekst])
CountVectorizer()
print(vectorizer.vocabulary_)
{'the': 86, 'researchers': 77, 'used': 97, 'publicly': 72, 'available': 12, 'geographic': 38, 'information': 5
0, 'system': 83, 'gis': 40, 'data': 25, 'and': 7, 'street': 80, 'level': 56, 'images': 46, 'to': 92, 'develop': 27
, 'their': 87, 'ubem': 95, 'then': 88, 'created': 23, '20': 0, 'archetypes': 9, 'of': 66, 'buildings': 15, 'that':
85, 'comprised': 21, 'eight': 28, 'commercial': 20, 'use': 96, 'types': 94, 'were': 103, 'sorted': 79, 'into':
 52, 'groups': 41, 'based': 13, 'on': 67, 'categories': 18, 'including': 49, 'type': 93, 'construction': 22, 'p
eriod': 70, 'with': 106, 'determine': 26, 'building': 14, 'material': 61, 'window': 105, 'wall': 100, 'ratio': 7
5, 'number': 65, 'floors': 34, 'lidar': 57, 'from': 37, 'geological': 39, 'survey': 82, 'height': 44, 'able': 2, 's
imulate': 78, 'map': 60, 'annual': 8, 'energy': 29, 'intensity': 51, '209': 1, 'structures': 81, 'in': 48, 'pittsb
urgh': 71, 'when': 104, 'they': 89, 'validated': 99, 'findings': 33, 'using': 98, 'other': 68, 'had': 43, 'just':
54, 'percent': 69, 'error': 30, 'rate': 74, 'though': 91, 'it': 53, 'currently': 24, 'mostly': 64, 'guided': 42, 'b
y': 16, 'looking': 58, 'at': 11, 'hope': 45, 'this': 90, 'modeling': 62, 'framework': 36, 'can': 17, 'eventually
': 31, 'take': 84, 'advantage': 4, 'machine': 59, 'learning': 55, 'more': 63, 'quickly': 73, 'analyze': 6, 'cat
egorize': 19, 'focus': 35, 'as': 10, 'well': 102, 'was': 101, 'an': 5, 'important': 47, 'addition': 3, 'field': 32,
 'research': 76}
```

**Slika 19. Kreiranje vokabulara za zadani tekst**

Dobiveni rječnik daje popis svih riječi iz analiziranog teksta, a broj koji stoji uz riječ govori na kojem se mjestu u vektoru ta riječ nalazi. Numeriranje elemenata vektora kreće od nule pa se tako npr. riječ „research“ u matrici vektoru nalazi na 77. mjestu. U dobivenom rječniku, svaka riječ je ključ, a vrijednost je broj pojavljivanja te riječi u ulaznom tekstu. Tako će se na 77. mjestu u matrici vektoru nalaziti broj 1 jer se riječ „research“ u cijelom tekstu pojavljuje jednom. Dobivena matrica vektora za analiziran tekst dana je na slici [slika 20].

```
vector=vectorizer.transform([tekst])
print(vector.shape)
(1, 107)
print(vector.toarray())
[[1 1 1 1 1 1 1 6 1 1 1 1 2 1 3 3 1 1 1 1 2 1 1 1
  1 3 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1
  1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 5 2 1 1 1 1
  2 1 1 1 1 4 1 1 2 1 1 1 1 2 12 2 1 2 1 1 7 1 1 1
  3 1 1 1 1 1 1 2 1 1 1]]
```

**Slika 20. Matrica vektor za ulazni tekst**

Iz slike vidimo funkcije potrebne za kreiranje vektora. Vector.transform() zadani tekst transformira u „vreću riječi“. Rezultat toga je vektor u kojem svaki stupac predstavlja riječ iz vokabulara. Vector.shape daje nam broj redova i stupaca matrice vektora. U ovom slučaju to je 1 red i 107 stupaca, što znači da se u zadanom tekstu pojavljuje 107 različitih riječi. Funkcija vector.toarray() stvara polje brojčanih vrijednosti koje daje broj ponavljanja svake riječi iz rječnika.

Osim za analizu cijelog teksta, dobiveni rječnik može se koristiti i za analizu pojedinačnih rečenica iz zadanog teksta. Taj je primjer dan na slici [slika 21].

```
sentence="The buildings were sorted into the groups based on categories including use type and c
onstruction period."

vector=vectorizer.transform([sentence])
print(vector.shape)
(1, 107)
print(vector.toarray())
[[0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0
  0 0 0 0 0 0 0 1 0 0 0 0 0 0 2 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0]]
```

**Slika 21. Matrica vektora za analiziranu rečenicu**

Matrica vektora za zadanu rečenicu također ima 107 stupaca jer se za analizu rečenice koristi cijeli rječnik koji je bio kreiran za analizu teksta. Obzirom da se većina riječi iz vokabulara ne pojavljuje u analiziranoj rečenici, većina vrijednosti vektora je 0 pa dobivamo prethodno spomenuti rijedak vektor (engl. sparse vector).

# 4. Funkcija TF-IDF

TF-IDF je skraćenica za učestalost izraza i inverzna frekvencija dokumenta (engl. term frequency – inverse document frequency), a predstavlja statistiku o važnosti riječi ili skupova riječi u nekom korpusu na temelju učestalosti i broja ponavljanja određene riječi.
Učestalost izraza (TF) je jednostavno broj pojavljivanja određene riječi u cijelom dokumentu. Računa se prema sljedećoj formuli:

$$tf(t,d) = \frac{\text{broj koliko se puta izraz } t \text{ pojavljuje u dokumentu } d}{\text{ukupan broj izraza u dokumentu } d}.$$

Inverzna frekvencija dokumenta (IDF) koristi se kako bi se odredilo koliko informacija pruža neka riječ ili izraz, a računa se prema formuli:

$$idf(t) = \log\left(\frac{\text{ukupan broj dokumenata}}{\text{broj dokumenata koji sadrže izraz } t}\right).$$

Na kraju računamo TF-IDF množenjem učestalosti izraza (TF) i inverzne frekvencije dokumenta (IDF):

$$tfidf(t,d) = tf(t,d) \cdot idf(t).$$

Riječi sa većim TF-IDF koeficijentom smatraju se važnijima za razumijevanje značenja dokumenta od onih s nižim TF-IDF koeficijentom. [20]
Visoka vrijednost TF-IDF dobiva se visokom frekvencijom unutar dokumenta i niskom frekvencijom dokumenta koji sadrže izraz jer se na taj način filtriraju izrazi koji se često ponavljaju unutar dokumenata u korpusu.
Možemo prikazati primjenu funkcije TF-IDF na jednostavnom primjeru:
Razmatramo dvije rečenice, odnosno dva dokumenta:

„Ova mačka je bijela.“
„Ova mačka je crna.“

Kako vidimo jedina razlika ovih rečenica su riječi „bijela“ i „crna“. To su riječi koje bi trebale imati visoku TF-IDF vrijednost za razliku od riječi „ova“ pa provjerimo je li to stvarno tako.

1. $tfidf(crna) = \frac{1}{4} \cdot \log\left(\frac{2}{1}\right) \approx 0{,}17$
2. $tfidf(ova) = \frac{1}{4} \cdot \log\left(\frac{2}{2}\right) \approx 0$

Dobiveni rezultati također potvrđuju pretpostavku da riječ „crna“ ima veću važnost od riječi „ova“.

### 4.1. Usporedba funkcije *TF-IDF* i modela „vreća riječi“

Obje metode, i „vreća riječi“ i TF-IDF su često korištene tehnike u obradi prirodnog jezika i pronalaženju informacija, a razlikuju se u načinu interpretacije ulaznih podataka.

„Vreća riječi“ je vrlo jednostavna tehnika koja funkcionira na način da broji riječi u tekstu, odnosno koliko se puta riječi u tekstu ponavljaju. Kao izlaz daje vektor čije su vrijednosti brojevi koliko se puta riječi iz rječnika kreiranog za analizu proizvoljnog teksta pojavljuju unutar analiziranog dijela teksta ili analiziranog cijelog teksta, pri čemu se zanemaruju i redoslijed riječi i gramatika. S druge strane TF-IDF je funkcija koja u analizi tekstualnih podataka u obzir uzima učestalost riječi u tekstu i na temelju toga pridaje važnost riječima za shvaćanje konteksta ulaznog teksta. [21]

Ovisno o zadatku i pristupu, oba pristupa analizi teksta mogu biti prikladna, a za dobivanje boljih rezultata također se može koristiti kombinacija oba pristupa.

# 5. ZAKLJUČAK

U ovom je radu dan prikaz analize proizvoljnog teksta korištenjem jedne od tehnike obrade prirodnog jezika, a to je model „vreća riječi“. Kroz teorijsku podlogu umjetne inteligencije i obrade jezika na samom početku rada, stvoren je okvir potreban za razumijevanje analiziranja teksta od strane računalnih sustava. Za jednostavnu razradu ovog zadatka pri implementaciji koda korištene su neke od postojećih Pytonovih biblioteka, a to nam omogućuje brži rad te također manje šanse za pogreškom prilikom pisanja koda. Bez uvođenja ovih biblioteka, razrada zadatka bi također bila moguća, ali puno kompliciranija.

Tehnika analize teksta „Bag-of-Words“ pokazala se kao učinkovita, a može se primjenjivati na velikoj količini ulaznih podataka, kao i na samo jednoj rečenici. Osim nje postoje i druge atraktivne metode koje na različite načine također analiziraju tekstualne podatke.

Obzirom da se računalni sustavi i njihove mogućnosti svakodnevno razvijaju i napreduju iz dana u dan, samim time dolazi i do napretka u tehnikama obrade prirodnog jezika. To danas uvelike olakšava posao rudarenja podataka, pogotovo kada je npr. iz velikog broja dobivenih informacija ili recenzija potrebno odrediti zadovoljstvo korisnika i slično.

# LITERATURA

[1] Izv. Prof. dr. sc. Tomislav Stipančić, dipl. ing.: podloge s predavanja (dostupno dana 10.02.2023.)

[2] https://devopedia.org/natural-language-processing (dostupno dana 11.02.2023.)

[3] https://www.ibm.com/topics/natural-language-processing (dostupno dana 11.02.2023.)

[4] https://monkeylearn.com/blog/natural-language-processing-techniques/ (dostupno dana 11.02.2023.)

[5] https://www.netguru.com/blog/sentiment-analysis-nlp (dostupno dana 13.02.2023.)

[6] https://www.shaip.com/solutions/named-entity-recognition-ner/ (dostupno dana 13.02.2023.)

[7] https://developer.nvidia.com/blog/how-to-build-domain-specific-automatic-speech-recognition-models-on-gpus/ (dostupno dana 13.02.2023.)

[8] https://it-mixer.com/siri-najveci-virtuelni-pomocnik-na-svijetu/ (dostupno dana 13.02.2023.)

[9] https://medium.com/@harinisureshla/wordclouds-basics-of-nlp-5b60be226414 (dostupno dana 13.02.2023.)

[10] https://medium.com/@harinisureshla/wordclouds-basics-of-nlp-5b60be226414 (dostupno dana 13.02.2023.)

[11] https://www.analyticsvidhya.com/blog/2021/09/what-are-n-grams-and-how-to-implement-them-in-python/ (dostupno dana 13.02.2023.)

[12] https://www.kdnuggets.com/2022/06/ngram-language-modeling-natural-language-processing.html (dostupno dana 13.02.2023.)

[13] https://www.freecodecamp.org/news/an-introduction-to-bag-of-words-and-how-to-code-it-in-python-for-nlp-282e87a9da04/ (dostupno dana 14.02.2023.)

[14] https://stackabuse.com/python-for-nlp-creating-bag-of-words-model-from-scratch/#bagofwordsmodelinpython (dostupno dana 14.02.2023.)

[15] https://www.megatrend.com/rast-programskog-jezika-python/ (dostupno dana 15.02.2023)

[16] https://www.nltk.org/ (dostupno dana 17.02.2023.)

[17] https://www.tutorialspoint.com/scikit_learn/index.htm# (dostupno dana 17.02.2023.)

[18] https://www.analyticsvidhya.com/blog/2021/07/getting-started-with-nlp-using-nltk-library/ (dostupno dana 17.02.2023.)

[19] https://www.sciencedaily.com/releases/2021/07/210722113106.htm (dostupno dana 20.02.2023.)

[20] https://www.educative.io/answers/what-is-tf-idf (dostupno dana 20.02.2023.)

[21] https://medium.com/@sewwandikaus.13/bow-vs-tf-idf-in-information-retrieval-a325b5e61984 (dostupno dana 21.02.2023.)

# PRILOZI

I. Python kod
II. Članak za analizu

# I. Python kod

```
Python 3.10.10 (tags/v3.10.10:aad5f6a, Feb  7 2023, 17:20:36) [MSC v.1929 64 bit (AMD64)] on win
Type "help", "copyright", "credits" or "license()" for more information.
import nltk
tekst="The researchers used publicly available Geographic Information System (GIS) data and stree

tekst=tekst.lower()

import re
remove="[.,]"
matches=re.sub(remove,"",tekst)

from nltk.tokenize import word_tokenize
result=word_tokenize(matches)

count=0
for i in range(0,len(result)):
    for j in range(0,len(result)):
        if(result[i]==result[j]):
            count=count+1
            count=0


from collections import Counter
word_counts=Counter(result)
for result, count in word_counts.most_common():
    print (result,count)

from sklearn.feature_extraction.text import CountVectorizer
vectorizer=CountVectorizer()
vectorizer.fit([tekst])

vector=vectorizer.transform([tekst])
print(vector.shape)
(1, 107)
print(vector.toarray())
```

II. Članak za analizu teksta

Obzirom da je u zadatku bilo potrebno proučiti i analizirati proizvoljan tekst korištenjem medote „vreća riječi“, za to je korišten odlomak iz članka „Visualizing a city's energy use“ s portala „Science daily“, a članak u cijelosti slijedi u nastavku.

Researchers at the University of Pittsburgh Swanson School of Engineering and the Mascaro Center for Sustainable Innovation used the City of Pittsburgh to create a model built upon the design, materials and purpose of commercial buildings to estimate their energy usage and emissions. While other models may be hindered by a scarcity of data in public records, the researchers' Urban Building Energy Model (UBEM) uses street-level images to categorize and estimate commercial buildings' energy use. Their findings were recently published in the journal *Energy & Buildings*.

"We found that in the existing literature, the scale of commercial buildings was always one of the challenges. It's cumbersome or even impossible to find and process detailed information about hundreds or thousands of buildings in an urban environment," said Rezvan Mohammadiziazi, lead author and graduate student in the Swanson School's Department of Civil and Environmental Engineering. "Researchers need to rely on assumptions based on when buildings were built or what the mechanical and electrical systems look like. Our hope is that by using image processing, we can build a framework that reduces some assumptions."

The researchers used publicly available Geographic Information System (GIS) data and street-level images to develop their UBEM, then created 20 archetypes of buildings that comprised eight commercial use types.The buildings were sorted into the groups based on categories including use type and construction period.

With street-level images to determine the building material, window-to-wall ratio, and number of floors, and LiDAR data from the U.S. Geological Survey to determine building height, the researchers were able to simulate and map the annual energy use intensity of 209 structures in Pittsburgh. When they validated their findings using other publicly available data, they had just a 7 percent error rate. Though it's currently mostly guided by the researchers looking at the images, the researchers hope that this modeling framework can eventually take advantage of machine learning to more quickly analyze and categorize building images.

The focus on commercial buildings, as well, was an important addition to the field of research.

"A lot of good work has already been done in this field, but there are fewer studies focusing on commercial buildings, because data about them is more difficult to capture than residential buildings. They're bigger and have more diverse uses," explained Mohammadiziazi. "We wanted to determine if an urban model for commercial buildings could be accurate based on acceptable errors, and it was."

While one goal was to create a framework that other researchers could replicate and build upon, another was to help the City of Pittsburgh meet its ambitious energy goals. Pittsburgh has joined 22 other U.S. cities as a 2030 District, pledging to reduce energy use, water consumption and transportation emissions by 50 percent by the year 2030. By creating a tool to estimate current usage, the research can aid policy makers in setting energy goals and efficiency regulations. The University of Pittsburgh is a member of the 2030 District, and hasalso pledged to be carbon neutral by 2037 to mark Pitt's 250th anniversary. Pitt Assistant Professor Melissa Bilec has been involved in the Pittsburgh 2030 District since its inception.

"We are fortunate -- and have worked diligently -- to have a strong partnership with the City of Pittsburgh, along with our own University of Pittsburgh's Facilities Management. The Mascaro Center for Sustainable Innovation values and fosters our internal and external partners," said Bilec, who is also Roberta A. Luxbacher Faculty Fellow in the department of civil and

environmental engineering, and deputy director of the Mascaro Center for Sustainable Innovation. "We will not meet or exceed our climate and energy goals without aggressive action and solid planning in the building sector. Models, such as the one we created, are intended to aid in the planning process to meet our goals."