Univerzitet u Nišu

Elektronski fakultet

Katedra za računarstvo

Seminarski rad

Analiza sentimenta u recenzijama filmova

Predmet: Veb majning

Mentor:

doc. dr Miloš Bogdanović

Student:

Miloš Veljanovski 1559

Niš, jun 2024.

Sadržaj

1. Uvod

Obrada prirodnog jezika (*NLP*) je oblast veštačke inteligencije koja se fokusira na interakciju između računara i ljudi putem prirodnog jezika. Cilj NLP-a je omogućiti računarima da razumeju, tumače i generišu ljudski jezik na način koji je smislen i koristan. Primene NLP-a uključuju prevođenje teksta, analizu sentimenta, četbotove i još mnogo toga.

1.1. Pregled analize sentimenta

Analiza sentimenta, poznata i kao rudarenje mišljenja, uključuje upotrebu NLP-a, analize teksta i računarske lingvistike za identifikaciju i ekstrakciju subjektivnih informacija iz izvora materijala. Koristi se za određivanje da li je neki tekst pozitivan, negativan ili neutralan. Analiza sentimenta ima brojne primene, uključujući praćenje društvenih mreža, povratne informacije korisnika i istraživanje tržišta.

U ovom radu vršimo analizu sentimenta na filmskim recenzijama sa *IMDb*-a kako bismo utvrdili da li su recenzije pozitivne ili negativne. Ovo može pomoći korisnicima da donesu informisane odluke o tome koje filmove da gledaju na osnovu agregiranih recenzija, štedeći im vreme i trud. Pored toga, filmski stvaraoci i kritičari mogu koristiti analizu sentimenta kako bi procenili javno mnjenje i poboljšali svoje ponude.

Za analizu sentimenta koristimo nekoliko različitih klasifikacionih modela kako bismo uporedili njihove performanse. Korišćeni klasifikatori uključuju: *K-Nearest Neighbors* (KNN), *Multinomial Naive Bayes*, *Logistic Regression*, *Stochastic Gradient Descent Classifier* (SGD), *Decision Tree Classifier* i *Random Forest Classifier*. Ovi modeli su implementirani pomoću Python biblioteka kao što su *scikit-learn* (sklearn) i koriste se u okruženju *Jupyter Notebook*-a, što omogućava interaktivnu analizu i vizualizaciju podataka. Pored toga, koristimo biblioteke kao što su *BeautifulSoup* za čišćenje teksta, kao i *nltk* za rad sa prirodnim jezikom.

1.2. Pregled Dataset-a

U ovom radu koristimo poznati dataset IMDb filmskih recenzija, koji je javno dostupan i široko korišćen za zadatke analize sentimenta. Dataset sadrži 50,000 filmskih recenzija koje su ručno označene kao pozitivne ili negativne. Ovaj dataset je izuzetno pogodan za analizu sentimenta zbog svoje uravnoteženosti i veličine, što omogućava temeljnu obuku i evaluaciju modela.

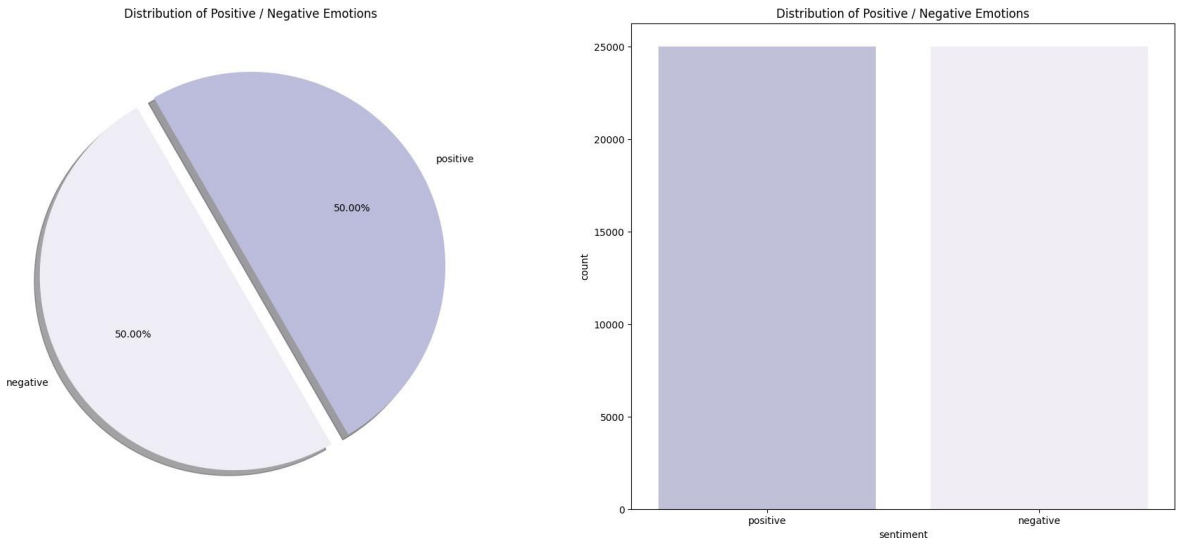
IMDb (Internet Movie Database) je najpoznatija online baza podataka o filmovima, televizijskim emisijama, glumcima i ekipama. Recenzije na IMDb-u pružaju bogat izvor podataka za istraživanje mišljenja i sentimenta gledalaca o različitim filmskim naslovima. Dataset koji koristimo u ovom radu sadrži jednak broj pozitivnih i negativnih recenzija, što olakšava balansiranu obuku modela.

Dataset je preuzet sa sajta Stanford univerziteta, gde je pripremljen od strane istraživača za potrebe naučnih istraživanja u oblasti analize sentimenta. Ovaj dataset je deo šireg istraživačkog projekta koji ima za cilj unapređenje tehnika obrade prirodnog jezika i mašinskog učenja za analizu tekstualnih podataka.

Dataset se sastoji od 50,000 recenzija, gde je svaka recenzija označena kao pozitivna ili negativna. Ovo omogućava ravnomernu raspodelu podataka za obuku i testiranje modela. Sledeća tabela prikazuje osnovne statistike dataset-a:

Atribut	Vrednost
Ukupan broj recenzija	50,000
Broj pozitivnih recenzija	25,000
Broj negativnih recenzija	25,000
Prosečna dužina recenzije (broj reči)	~230
Prosečna dužina recenzije (broj reči)	10
Maksimalna dužina recenzije (broj reči)	2470

Grafički prikaz distribucije pozitivnih i negativnih recenzija prikazan je na sledećem dijagramu:



Slika 1. Grafički prikaz distribucije pozitivnih i negativnih recenzija

Dijagram je generisan korišćenjem *matplotlib* i *seaborn* biblioteka na sledeći način:

```
a, ax = plt.subplots(1, 2, figsize=(22,9))
df['sentiment'].value_counts().plot.pie(explode=[0,0.1], autopct='%1.2f%%',
ax=ax[0], shadow=True, startangle=300, colors=["#bcbddc", "#efedf5"])
ax[0].set_title('Distribution of Positive / Negative Emotions')
ax[0].set_ylabel('')
sns.countplot(x='sentiment', data = df, ax=ax[1], palette=["#bcbddc", "#efedf5"])
ax[1].set_title('Distribution of Positive / Negative Emotions')
plt.show()
```

2. Obrada podataka

2.1. Čišćenje teksta

Čišćenje teksta je ključni korak u analizi prirodnog jezika, jer sirovi tekstualni podaci često sadrže različite šumove i nepotrebne informacije. Ovaj korak uključuje uklanjanje *HTML* oznaka, specijalnih karaktera, brojeva, kao i *stop* reči. Cilj je pretvoriti tekst u format koji je pogodan za analizu.

2.1.1. Čišćenje teksta na primeru jedne recenzije

Kao primer, uzimamo prvu recenziju iz dataset-a i korak po korak je čistimo:

```
example_review = df.review[0]
example_review
```

Originalna recenzija može izgledati ovako:

```
"One of the other reviewers has mentioned that after watching just 1 Oz episode
you'll be hooked. They are right, as this is exactly what happened with me.<br
/><br />The first thing that struck me about Oz was its brutality and unflinching
scenes of violence, which set in right from the word GO. Trust me, this is not a
show for the faint hearted or timid. This show pulls no punches with regards to
drugs, sex or violence. Its is hardcore, in the classic use of the word.<br /><br
/>It is called OZ as that is the nickname given to the Oswald Maximum Security
State Penitentary. It focuses mainly on Emerald City, an experimental section of
the prison where all the cells have glass fronts and face inwards, so privacy is
not high on the agenda. Em City is home to many..Aryans, Muslims, gangstas,
Latinos, Christians, Italians, Irish and more....so scuffles, death stares, dodgy
dealings and shady agreements are never far away.<br /><br />I would say the main
appeal of the show is due to the fact that it goes where other shows wouldn't dare.
Forget pretty pictures painted for mainstream audiences, forget charm, forget
romance...OZ doesn't mess around. The first episode I ever saw struck me as so
nasty it was surreal, I couldn't say I was ready for it, but as I watched more, I
developed a taste for Oz, and got accustomed to the high levels of graphic
violence. Not just violence, but injustice (crooked guards who'll be sold out for a
nickel, inmates who'll kill on order and get away with it, well mannered, middle
class inmates being turned into prison bitches due to their lack of street skills
or prison experience) Watching Oz, you may become comfortable with what is
uncomfortable viewing....thats if you can get in touch with your darker side."
```

- Najpre, vršimo uklanjanje HTML oznaka (tagova) pomoću *BeautifulSoup* biblioteke:

```
example_review = BeautifulSoup(example_review).get_text()
```

Rezultat nakon uklanjanja HTML oznaka:

```
"One of the other reviewers has mentioned that after watching just 1 Oz episode
you'll be hooked. They are right, as this is exactly what happened with me. The
first thing that struck me about Oz was its brutality and unflinching scenes of
violence, which set in right from the word GO. Trust me, this is not a show for the
faint hearted or timid. This show pulls no punches with regards to drugs, sex or
violence. Its is hardcore, in the classic use of the word. It is called OZ as that
is the nickname given to the Oswald Maximum Security State Penitentary. It focuses
mainly on Emerald City, an experimental section of the prison where all the cells
have glass fronts and face inwards, so privacy is not high on the agenda. Em City
is home to many..Aryans, Muslims, gangstas, Latinos, Christians, Italians, Irish
and more....so scuffles, death stares, dodgy dealings and shady agreements are
never far away. I would say the main appeal of the show is due to the fact that it
goes where other shows wouldn't dare. Forget pretty pictures painted for mainstream
audiences, forget charm, forget romance...OZ doesn't mess around. The first episode
I ever saw struck me as so nasty it was surreal, I couldn't say I was ready for it,
but as I watched more, I developed a taste for Oz, and got accustomed to the high
levels of graphic violence. Not just violence, but injustice (crooked guards who'll
be sold out for a nickel, inmates who'll kill on order and get away with it, well
mannered, middle class inmates being turned into prison bitches due to their lack
of street skills or prison experience) Watching Oz, you may become comfortable with
what is uncomfortable viewing....thats if you can get in touch with your darker
side."
```

- Zatim, vršimo uklanjanje specijalnih karaktera i brojeva (svi karakteri koji nisu iz *a-zA-Z* skupa regularnih izraza bivaju zamenjeni praznim karakteriom):

```
example_review = re.sub("[^a-zA-Z]", ' ', example_review)
```

Rezultat nakon uklanjanja specijalnih karaktera i brojeva:

```
"One of the other reviewers has mentioned that after watching just  Oz episode you
ll be hooked  They are right  as this is exactly what happened with me The first
thing that struck me about Oz was its brutality and unflinching scenes of violence
which set in right from the word GO  Trust me  this is not a show for the faint
hearted or timid  This show pulls no punches with regards to drugs  sex or violence
Its is hardcore  in the classic use of the word It is called OZ as that is the
nickname given to the Oswald Maximum Security State Penitentary  It focuses mainly
on Emerald City  an experimental section of the prison where all the cells have
glass fronts and face inwards  so privacy is not high on the agenda  Em City is
home to many  Aryans  Muslims  gangstas  Latinos  Christians  Italians  Irish and
more    so scuffles  death stares  dodgy dealings and shady agreements are never
far away I would say the main appeal of the show is due to the fact that it goes
where other shows wouldn t dare  Forget pretty pictures painted for mainstream
audiences  forget charm  forget romance    OZ doesn t mess around  The first episode
I ever saw struck me as so nasty it was surreal  I couldn t say I was ready for it
```

```
but as I watched more  I developed a taste for Oz  and got accustomed to the high
levels of graphic violence  Not just violence  but injustice  crooked guards who ll
be sold out for a nickel  inmates who ll kill on order and get away with it  well
mannered  middle class inmates being turned into prison bitches due to their lack
of street skills or prison experience  Watching Oz  you may become comfortable with
what is uncomfortable viewing    thats if you can get in touch with your darker
side"
```

- Nakon ovoga, neophodno je izvršiti pretvaranje teksta u mala slova. Ovaj korak je važan jer osigurava doslednost u podacima. Bez ovog koraka, reči koje su identične po značenju ali različite po veličini slova tretirale bi se kao različiti tokeni. Na primer, reči „Film“ i „film“ bile bi tretirane kao dve različite reči. Ovo je posebno bitno jer se time smanjuje broj jedinstvenih reči (tokena) koje model mora da obradi, čime se poboljšava efikasnost i tačnost modela.

```
example_review = example_review.lower()
```

Rezultat nakon pretvaranja u mala slova:

```
"one of the other reviewers has mentioned that after watching just  oz episode you
ll be hooked  they are right  as this is exactly what happened with me the first
thing that struck me about oz was its brutality and unflinching scenes of violence
which set in right from the word go  trust me  this is not a show for the faint
hearted or timid  this show pulls no punches with regards to drugs  sex or violence
its is hardcore  in the classic use of the word it is called oz as that is the
nickname given to the oswald maximum security state penitentary  it focuses mainly
on emerald city  an experimental section of the prison where all the cells have
glass fronts and face inwards  so privacy is not high on the agenda  em city is
home to many  aryans  muslims  gangstas  latinos  christians  italians  irish and
more    so scuffles  death stares  dodgy dealings and shady agreements are never
far away i would say the main appeal of the show is due to the fact that it goes
where other shows wouldn t dare  forget pretty pictures painted for mainstream
audiences  forget charm  forget romance    oz doesn t mess around  the first episode
i ever saw struck me as so nasty it was surreal  i couldn t say i was ready for it
but as i watched more  i developed a taste for oz  and got accustomed to the high
levels of graphic violence  not just violence  but injustice  crooked guards who ll
be sold out for a nickel  inmates who ll kill on order and get away with it  well
mannered  middle class inmates being turned into prison bitches due to their lack
of street skills or prison experience  watching oz  you may become comfortable with
what is uncomfortable viewing    thats if you can get in touch with your darker
side"
```

- Zatim, funkcija *split* razdvaja tekst na pojedinačne reči na osnovu belina (praznih mesta). Ovaj korak je bitan jer omogućava da se svaka reč tretira kao poseban token, što je neophodno za dalju obradu teksta:

```
example_review = example_review.split()
```

Rezultat nakon razdvajanja teksta na reči:

```
['one', 'of', 'the', 'other', 'reviewers', 'has', 'mentioned', 'that', 'after',
'watching', 'just', 'oz', 'episode', 'you', 'll', 'be', 'hooked', 'they', 'are',
'right', 'as', 'this', 'is', 'exactly', 'what', 'happened', 'with', 'me', 'the',
'first', 'thing', 'that', 'struck', 'me', 'about', 'oz', 'was', 'its', 'brutality',
'and', 'unflinching', 'scenes', 'of', 'violence', 'which', 'set', 'in', 'right',
'from', 'the', 'word', 'go', 'trust', 'me', 'this', 'is', 'not', 'a', 'show',
'for', 'the', 'faint', 'hearted', 'or', 'timid', 'this', 'show', 'pulls', 'no',
'punches', 'with', 'regards', 'to', 'drugs', 'sex', 'or', 'violence', 'its', 'is',
'hardcore', 'in', 'the', 'classic', 'use', 'of', 'the', 'word', 'it', 'is',
'called', 'oz', 'as', 'that', 'is', 'the', 'nickname', 'given', 'to', 'the',
'oswald', 'maximum', 'security', 'state', 'penitentary', 'it', 'focuses', 'mainly',
'on', 'emerald', 'city', 'an', 'experimental', 'section', 'of', 'the', 'prison',
'where', 'all', 'the', 'cells', 'have', 'glass', 'fronts', 'and', 'face',
'inwards', 'so', 'privacy', 'is', 'not', 'high', 'on', 'the', 'agenda', 'em',
'city', 'is', 'home', 'to', 'many', 'aryans', 'muslims', 'gangstas', 'latinos',
'christians', 'italians', 'irish', 'and', 'more', 'so', 'scuffles', 'death',
'stares', 'dodgy', 'dealings', 'and', 'shady', 'agreements', 'are', 'never', 'far',
'away', 'i', 'would', 'say', 'the', 'main', 'appeal', 'of', 'the', 'show', 'is',
'due', 'to', 'the', 'fact', 'that', 'it', 'goes', 'where', 'other', 'shows',
'wouldn', 't', 'dare', 'forget', 'pretty', 'pictures', 'painted', 'for',
'mainstream', 'audiences', 'forget', 'charm', 'forget', 'romance', 'oz', 'doesn',
't', 'mess', 'around', 'the', 'first', 'episode', 'i', 'ever', 'saw', 'struck',
'me', 'as', 'so', 'nasty', 'it', 'was', 'surreal', 'i', 'couldn', 't', 'say', 'i',
'was', 'ready', 'for', 'it', 'but', 'as', 'i', 'watched', 'more', 'i', 'developed',
'a', 'taste', 'for', 'oz', 'and', 'got', 'accustomed', 'to', 'the', 'high',
'levels', 'of', 'graphic', 'violence', 'not', 'just', 'violence', 'but',
'injustice', 'crooked', 'guards', 'who', 'll', 'be', 'sold', 'out', 'for', 'a',
'nickel', 'inmates', 'who', 'll', 'kill', 'on', 'order', 'and', 'get', 'away',
'with', 'it', 'well', 'mannered', 'middle', 'class', 'inmates', 'being', 'turned',
'into', 'prison', 'bitches', 'due', 'to', 'their', 'lack', 'of', 'street',
'skills', 'or', 'prison', 'experience', 'watching', 'oz', 'you', 'may', 'become',
'comfortable', 'with', 'what', 'is', 'uncomfortable', 'viewing', 'thats', 'if',
'you', 'can', 'get', 'in', 'touch', 'with', 'your', 'darker', 'side']
```

- Finalno, vrši se uklanjanje stop reči. Stop reči su česte reči u jeziku koje same po sebi ne nose mnogo informacija o sadržaju teksta. Primeri stop reči na engleskom jeziku jesu: *„the“*, *„is“*, *„in“*, *„and“*, *„to“*, itd. Uklanjanje ovih reči pomaže da se smanji šum u podacima i fokusira se na značajnije reči koje doprinose razumevanju konteksta i sentimenta teksta.

```
words = set(stopwords.words("english"))
example_review = [word for word in example_review if word not in words]
```

Rezultat nakon uklanjanja stop reči:

```
['one', 'reviewers', 'mentioned', 'watching', 'oz', 'episode', 'hooked', 'right',
'exactly', 'happened', 'first', 'thing', 'struck', 'oz', 'brutality',
'unflinching', 'scenes', 'violence', 'set', 'right', 'word', 'go', 'trust', 'show',
```

```
'faint', 'hearted', 'timid', 'show', 'pulls', 'punches', 'regards', 'drugs', 'sex',
'violence', 'hardcore', 'classic', 'use', 'word', 'called', 'oz', 'nickname',
'given', 'oswald', 'maximum', 'security', 'state', 'penitentary', 'focuses',
'mainly', 'emerald', 'city', 'experimental', 'section', 'prison', 'cells', 'glass',
'fronts', 'face', 'inwards', 'privacy', 'high', 'agenda', 'em', 'city', 'home',
'many', 'aryans', 'muslims', 'gangstas', 'latinos', 'christians', 'italians',
'irish', 'scuffles', 'death', 'stares', 'dodgy', 'dealings', 'shady', 'agreements',
'never', 'far', 'away', 'would', 'say', 'main', 'appeal', 'show', 'due', 'fact',
'goes', 'shows', 'dare', 'forget', 'pretty', 'pictures', 'painted', 'mainstream',
'audiences', 'forget', 'charm', 'forget', 'romance', 'oz', 'mess', 'around',
'first', 'episode', 'ever', 'saw', 'struck', 'nasty', 'surreal', 'say', 'ready',
'watched', 'developed', 'taste', 'oz', 'got', 'accustomed', 'high', 'levels',
'graphic', 'violence', 'violence', 'injustice', 'crooked', 'guards', 'sold',
'nickel', 'inmates', 'kill', 'order', 'get', 'away', 'well', 'mannered', 'middle',
'class', 'inmates', 'turned', 'prison', 'bitches', 'due', 'lack', 'street',
'skills', 'prison', 'experience', 'watching', 'oz', 'may', 'become', 'comfortable',
'uncomfortable', 'viewing', 'thats', 'get', 'touch', 'darker', 'side']
```

2.1.2. Sumiranje procesa čišćenja teksta

Sve opisane korake za čišćenje teksta - uklanjanje HTML oznaka, uklanjanje specijalnih karaktera, pretvaranje teksta u mala slova, razdvajanje teksta na reči i uklanjanje stop reči - smo implementirali u okviru jedne funkcije nazvane *process*. Ova funkcija se koristi za obradu celokupnog dataset-a, primenjujući sve navedene korake kako bi se tekstualni podaci pripremili za dalju analizu i obuku modela mašinskog učenja. Korišćenjem funkcije process, obezbeđujemo doslednost u obradi svih recenzija i osiguravamo kvalitetne podatke za našu analizu sentimenta.

```python
def process(review):
    # Remove HTML tags
    review = BeautifulSoup(review).get_text()
    # Remove punctuation and numbers
    review = re.sub("[^a-zA-Z]", ' ', review)
    # Converting into lowercase and splitting to eliminate stopwords
    review = review.lower()
    review = review.split()
    # Remove stopwords
    swords = set(stopwords.words("english"))
    review = [w for w in review if w not in swords]
    # Combine splitted paragraphs with space
    return(" ".join(review))
# Apply the process function to the reviews
df['cleaned_review'] = df['review'].apply(process)
```

2.2. Oblaci reči

Oblaci reči (*Word Clouds*) su vizuelna reprezentacija učestalosti reči u tekstu. Ove vizualizacije pomažu da se brzo identifikuju najčešće korišćene reči u pozitivnim i negativnim recenzijama. Veće reči u oblakovima reči predstavljaju reči koje se češće pojavljuju u tekstu, dok manje reči predstavljaju reči koje se ređe pojavljuju.

U našem radu, kreirali smo oblake reči za pozitivne i negativne recenzije kako bismo vizuelno prikazali koje reči dominiraju u različitim tipovima recenzija. Ovi oblaci reči mogu pomoći u razumevanju ključnih tema i emocija koje se javljaju u recenzijama.

2.2.1. Implementacija oblaka reči

- Prvo, odvojili smo pozitivne i negativne recenzije iz našeg dataset-a.

```
positive_reviews = df[df['sentiment'] == 'positive']['cleaned_review']
negative_reviews = df[df['sentiment'] == 'negative']['cleaned_review']
```

- Zatim smo spojili sve pozitivne recenzije u jedan veliki string, kao i sve negativne recenzije:

```
positive_text = " ".join(positive_reviews)
negative_text = " ".join(negative_reviews)
```

- Koristili smo biblioteku *WordCloud* za generisanje oblaka reči za pozitivne i negativne recenzije:

```
positive_wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(positive_text)
negative_wordcloud = WordCloud(width=800, height=400,
background_color='black').generate(negative_text)
```

- Na kraju, prikazali smo oblake reči koristeći *matplotlib*:

```
plt.figure(figsize=(16,8))
plt.subplot(1, 2, 1)
plt.imshow(positive_wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Positive Reviews Word Cloud')

plt.subplot(1, 2, 2)
plt.imshow(negative_wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Negative Reviews Word Cloud')

plt.show()
```

2.2.2. Vizuelizacija podataka

Slika 2. Oblaci reči pozitivnih i negativnih recenzija

Na osnovu oblaka reči, možemo primetiti da pozitivne recenzije često sadrže reči kao što su „*great*“, „*amazing*“, „*best*“, dok negativne recenzije često sadrže reči kao što su „*bad*“, „*worst*“, „*boring*“. Ovi nalazi su u skladu sa očekivanjima i pomažu nam da bolje razumemo jezičke karakteristike koje se povezuju sa različitim sentimentima.

3. Ekstrakcija karakteristika

Ekstrakcija karakteristika je ključan korak u pripremi teksta za mašinsko učenje. U ovom radu, koristimo *Bag of Words* (BoW) metodu za ekstrakciju karakteristika. Ova metoda transformiše tekstualne podatke u numeričke vektore koje mašinski algoritmi mogu da koriste za klasifikaciju. Takođe ćemo objasniti *TF-IDF* (Term Frequency-Inverse Document Frequency) tehniku kao komparativnu metodu, iako je nismo direktno koristili u ovom radu.

3.1. Bag of Words (BoW)

Bag of Words je jednostavna i često korišćena metoda za ekstrakciju karakteristika iz teksta. U ovoj tehnici, tekst se razdvaja na pojedinačne reči (tokene), a zatim se računa frekvencija pojavljivanja svake reči u tekstu. Rezultat je vektorska reprezentacija teksta gde svaki vektor predstavlja broj pojavljivanja reči u tekstu. Bag of Words zanemaruje redosled reči i fokusira se isključivo na njihovu frekvenciju.

3.1.1. Implementacija Bag of Words

U implementaciji Bag of Words, koristili smo *CountVectorizer* iz biblioteke sklearn. Limitirali smo broj karakteristika na 10,000 kako bismo smanjili dimenzionalnost podataka i ubrzali proces treniranja modela.

```
vectorizer = CountVectorizer(max_features=10000)
train_x = vectorizer.fit_transform(train_x)
```

Koristeći ovu tehniku, svaka recenzija u skupu podataka je predstavljena kao vektor koji sadrži broj pojavljivanja reči iz skupa od 10,000 najčešćih reči. Ovo omogućava algoritmima mašinskog učenja da obrađuju tekstualne podatke na strukturisan način. Na primer, ako imamo recenzije koji sadrže reči „film“, „dobar“, „gluma“, „loš“, BoW će pretvoriti ove recenzije u vektore sa brojem pojavljivanja ovih reči. Ova reprezentacija omogućava mašinskim algoritmima da kvantitativno analiziraju tekstualne podatke i identifikuju obrasce.

3.2. TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF je naprednija tehnika koja uzima u obzir ne samo frekvenciju reči u dokumentu već i koliko je ta reč informativna za celu kolekciju dokumenata. TF-IDF smanjuje težinu reči koje su česte u mnogim dokumentima i povećava težinu reči koje su specifične za pojedinačne dokumente.
TF-IDF kombinuje dve mere:

- ***Term Frequency* (TF)**: Broj pojavljivanja određene reči u dokumentu podeljen sa ukupnim brojem reči u tom dokumentu. Ovo daje meru koliko je određena reč važna u okviru jednog dokumenta.

$$TF(t,d) = \frac{Broj\ pojavljanja\ reči\ t\ u\ dokumentu\ d}{Ukupan\ broj\ reči\ u\ dokumentu\ d}$$

- ***Inverse Document Frequency* (IDF)**: Logaritamski odnos ukupnog broja dokumenata prema broju dokumenata koji sadrže određenu reč. Ovo daje meru koliko je određena reč važna u okviru celog korpusa dokumenata. Ako je reč veoma česta u mnogim dokumentima, IDF će biti nizak, što smanjuje njenu težinu.

$$IDF(t) = \log\left(\frac{Ukupan\ broj\ dokumenata}{Broj\ dokumenata\ koji\ sadrže\ reč\ t}\right)$$

Kombinovanjem TF i IDF dobijamo TF-IDF vrednost:

$$TF - IDF(t,d) = TF(t,d) \times IDF(t)$$

Iako nismo direktno koristili TF-IDF u našem radu, važno je razumeti ovu tehniku jer može pružiti poboljšanu reprezentaciju teksta u poređenju sa Bag of Words, posebno u situacijama gde su određene reči veoma česte ali ne informativne za klasifikaciju.

3.3. Priprema podataka za treniranje

Nakon što smo ekstraktovali karakteristike, podelili smo podatke na trening i test setove. Koristili smo *train_test_split* funkciju iz sklearn biblioteke kako bismo podelili podatke na 80% za treniranje i 20% za testiranje. Ovaj korak osigurava da imamo dovoljno podataka za obuku modela, dok i dalje imamo deo podataka na kojima možemo da testiramo performanse modela.

```
independent = train_data
dependent = np.array(df["sentiment"])

train_x, test_x, y_train, y_test = train_test_split(independent, dependent,
stratify=dependent, test_size=0.2, shuffle=True, random_state=25)
```

4. Izbor modela i treniranje

Izbor odgovarajućih modela za klasifikaciju i njihovo treniranje predstavlja ključni korak u analizi sentimenta. U ovom radu, koristili smo nekoliko popularnih klasifikacionih algoritama kako bismo uporedili njihove performanse i identifikovali koji model najbolje odgovara zadatku analize sentimenta. Modeli koji su korišćeni jesu:

- K-Nearest Neighbors
- Multinomial Naive Bayes
- Logistic Regression
- Stochastic Gradient Descent Classifier
- Decision Tree Classifier
- Random Forest Classifier

Svaki od ovih modela ima svoje prednosti i nedostatke, kao i specifične primene u različitim kontekstima analize podataka. U nastavku će biti detaljno objašnjen svaki od modela i proces njihovog treniranja.

4.1. Metrike za procenu tačnosti modela

Evaluacija performansi modela je ključni korak u mašinskom učenju. Korišćenjem odgovarajućih metrika možemo precizno proceniti koliko dobro model funkcioniše na neviđenim podacima. U ovom radu, koriste se sledeće metrike za procenu tačnosti modela: tačnost (*accuracy*), F1 skor (*F1 score*), preciznost (*precision*), odziv (*recall*) i *ROC AUC* skor. Sve navedene metrike koristimo za evaluaciju performansi modela. Na ovaj način možemo dobiti sveobuhvatan uvid u to kako model funkcioniše u različitim aspektima, čime osiguravamo pouzdane rezultate.

4.1.1. Tačnost (Accuracy)

Tačnost je osnovna metrika koja meri procenat ispravno klasifikovanih primera u odnosu na ukupan broj primera. Računa se kao odnos broja tačno predviđenih primera (i pozitivnih i negativnih) i ukupnog broja primera.

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

gde su:

- TP (True Positives) – tačno pozitivne vrednosti,
- TN (True Negatives) – tačno negativne vrednosti,
- FP (False Positives) – lažno pozitivne vrednosti,
- FN (False Negatives) – lažno negativne vrednosti.

4.1.2. Preciznost (Precision)

Preciznost meri koliko od predviđenih pozitivnih primera stvarno pripada pozitivnoj klasi. Visoka preciznost znači da je malo lažno pozitivnih primera.

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

4.1.3. Odziv (Recall)

Odziv meri koliko od stvarno pozitivnih primera je ispravno identifikovano od strane modela. Visok odziv znači da je malo lažno negativnih primera.

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

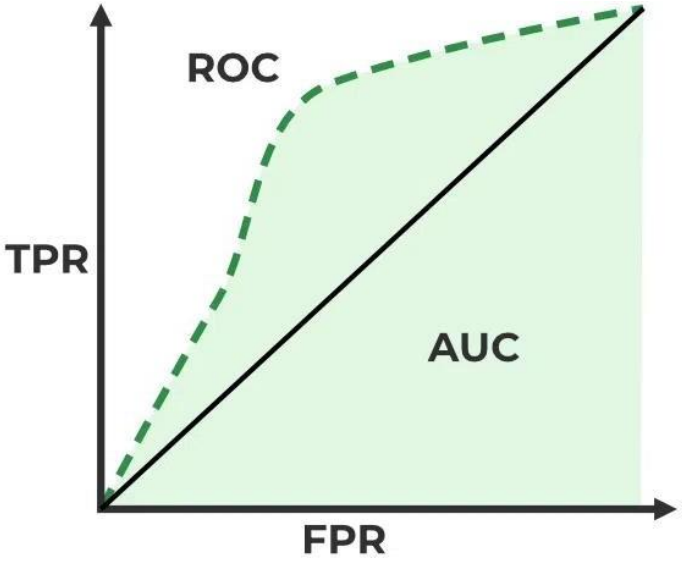
4.1.4. F1 skor (F1 score)

F1 skor je harmonijska sredina između preciznosti i odziva. Koristi se kada je balans između preciznosti i odziva važan. F1 skor uzima u obzir i lažno pozitivne i lažno negativne vrednosti, te je korisna metrika kada imamo neuravnotežene klase.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

4.1.5. ROC AUC skor (ROC AUC score)

ROC AUC (*Receiver Operating Characteristic Area Under the Curve*) skor je metrika koja meri sposobnost modela da razlikuje između klasa. ROC kriva prikazuje odnos između true positive rate (TPR) i false positive rate (FPR) na različitim pragovima klasifikacije. AUC (Area Under the Curve) predstavlja ukupnu površinu ispod ROC krive, pri čemu vrednost bliža 1 označava bolji model.



Slika 3. Metrika procene ROC AUC klasifikacije

4.2. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) je jedan od najstarijih i najjednostavnijih algoritama za klasifikaciju i regresiju u mašinskom učenju. Funkcioniše tako što klasifikuje ulazni podatak na osnovu klasa njegovih najbližih suseda u prostoru karakteristika. KNN je primer lenjog algoritma učenja jer ne gradi eksplicitni model tokom faze treniranja; umesto toga, svo računanje se dešava tokom faze predikcije.

Kako ovaj model funkcioniše korak po korak:

- Definisanje vrednosti K: Prvo se odredi vrednost K, koja predstavlja broj najbližih suseda koji će se koristiti za klasifikaciju.
- Izračunavanje udaljenosti: Za svaku tačku u testnom skupu, KNN algoritam izračunava udaljenost između te tačke i svih tačaka u trening skupu. Najčešće korišćena metrika udaljenosti je *Euklidova* udaljenost, ali se mogu koristiti i druge metrike kao što su *Manhetn* ili *Minkovski*.
- Pronalaženje najbližih K suseda: Nakon što su sve udaljenosti izračunate, algoritam identifikuje K tačaka iz trening skupa koje su najbliže testnoj tački.

- Glasanje za klasu: Svaki od K suseda glasa za svoju klasu, a testna tačka se klasifikuje kao klasa koja ima najviše glasova među K susedima. U slučaju regresije, KNN vraća prosečnu vrednost K najbližih suseda.

KNN model je implementiran korišćenjem KneighborsClassifier-a iz *sklearn.neighbors* biblioteke. Sledeći deo koda prikazuje kako je model dodat u listu modela:

```
models.append(('K-Nearest Neighbors', KNeighborsClassifier()))
```

Nakon što je model dodat, treniranje i evaluacija modela su sprovedeni koristeći sledeći kod:

```
for name, model in models:
    %time model.fit(train_x, train_y)
    test_pred = model.predict(test_result)
    print(name, 'Accuracy Score: ', accuracy_score(y_test, test_pred))
    print(name, 'F1 Score: ', f1_score(y_test, test_pred, average='weighted'))
    print(name, 'Precision Score: ', precision_score(y_test, test_pred,
average='weighted'))
    print(name, 'Recall Score: ', recall_score(y_test, test_pred,
average='weighted'))
    print(name, 'ROC AUC Score: ', roc_auc_score(pd.get_dummies(y_test).values,
pd.get_dummies(test_pred).values, average='weighted'))
    print(confusion_matrix(y_test, test_pred))
    print(classification_report(y_test, test_pred))
    print('-------------------------------------------------------------------------')
```

Nakon treniranja modela, procenjena je njegova tačnost, F1 skor, preciznost, odziv i ROC AUC skor. Rezultati evaluacije modela su sledeći:

```
CPU times: user 41.4 ms, sys: 42 µs, total: 41.4 ms
Wall time: 42.3 ms
K-Nearest Neighbors Accuracy Score:  0.6239
K-Nearest Neighbors F1 Score:  0.622590687840162
K-Nearest Neighbors Precision Score:  0.6256435352354973
K-Nearest Neighbors Recall Score:  0.6239
K-Nearest Neighbors ROC AUC Score:  0.6239
[[3414 1586]
 [2175 2825]]
              precision    recall  f1-score   support

    negative       0.61      0.68      0.64      5000
    positive       0.64      0.56      0.60      5000

    accuracy                           0.62     10000
   macro avg       0.63      0.62      0.62     10000
weighted avg       0.63      0.62      0.62     10000
```

Prednosti ovog modela jesu:

- Jednostavnost: KNN je jednostavan za implementaciju i razumevanje.
- Bez treniranja: Nema faze treniranja, što znači da svi podaci ostaju netaknuti do faze predikcije.
- Fleksibilnost: KNN može da koristi različite metrike udaljenosti i može biti prilagođen različitim vrstama podataka.

Nedostaci ovog modela jesu:

- Velika memorijska potrošnja: KNN zahteva čuvanje celokupnog trening skupa, što može biti memorijski intenzivno za velike skupove podataka.
- Sporost: Izračunavanje udaljenosti za svaku predikciju može biti sporo, posebno za velike skupove podataka.
- Osetljivost na skaliranje podataka: KNN je veoma osetljiv na skaliranje podataka, jer različite metrike udaljenosti mogu dati različite rezultate ako podaci nisu pravilno skalirani.

4.3. Multinomial Naive Bayes (MNB)

Multinomial Naive Bayes (MNB) je klasifikacioni algoritam zasnovan na *Bayes*-ovoj teoremi i posebno je efikasan za probleme klasifikacije teksta. MNB je deo porodice algoritama Naive Bayes, koji se zasnivaju na pretpostavci o uslovnoj nezavisnosti između atributa.

Kako ovaj model funkcioniše:

MNB model pretpostavlja da je verovatnoća pojavljivanja određene reči u dokumentu nezavisna od pojavljivanja bilo koje druge reči. Ovo je poznato kao "naivna" pretpostavka nezavisnosti. Iako je ova pretpostavka pojednostavljena, pokazalo se da je veoma efikasna u praksi, posebno kod klasifikacije tekstualnih podataka.

Bayes-ova teorema se koristi za izračunavanje verovatnoće da dokument pripada određenoj klasi C_k, s obzirom na prisustvo skupa reči $W = \{w_1, w_2, \dots w_n\}$ u dokumentu. Formula Bayes-ove teoreme glasi:

$$P(C_k|W) = \frac{P(W|C_k) \cdot P(C_k)}{P(W)}$$

gde je:

- $P(C_k|W)$ verovatnoća da dokument pripada klasi C_k s obzirom na reči W,
- $P(W|C_k)$ verovatnoća pojavljivanja reči W u dokumentu klase C_k,
- $P(C_k)$ apriorna verovatnoća klase C_k,
- $P(W)$ ukupna verovatnoća pojavljivanja reči W u svim dokumentima.

U kontekstu MNB, verovatnoća $P(W|C_k)$ se računa kao proizvod verovatnoća pojavljivanja pojedinačnih reči u klasi C_k:

$$P(W|C_k) = \prod_{i=1}^{n} P(w_i|C_k)$$

gde je $P(w_i|C_k)$ verovatnoća pojavljivanja reči w_i u dokumentima klase C_k.

MNB model je implementiran korišćenjem MultinomialNB iz *sklearn.naive_bayes biblioteke*. Sledeći deo koda prikazuje kako je model dodat u listu modela:

```
models.append(('Multinomial Naive Bayes', MultinomialNB()))
```

Treniran je i evaluiran koristeći isti postupak kao i prethodni model.

Nakon treniranja modela, procenjena je njegova tačnost, F1 skor, preciznost, odziv i ROC AUC skor. Rezultati evaluacije modela su sledeći:

```
CPU times: user 35.2 s, sys: 53.1 ms, total: 35.2 s
Wall time: 35.5 s
Multinomial Naive Bayes Accuracy Score:  0.8549
 Multinomial Naive Bayes F1 Score:  0.854898419843792
 Multinomial Naive Bayes Precision Score:  0.8549154601174427
 Multinomial Naive Bayes Recall Score:  0.8549
 Multinomial Naive Bayes ROC AUC Score:  0.8549
 [[4291  709]
 [ 742 4258]]
               precision    recall  f1-score   support

 negative            0.85      0.86      0.86      5000
 positive            0.86      0.85      0.85      5000

 accuracy                              0.85     10000
 macro avg           0.85      0.85      0.85     10000
 weighted avg        0.85      0.85      0.85     10000
```

Prednosti ovog modela jesu:

- Jednostavnost i efikasnost: MNB je jednostavan za implementaciju i vrlo efikasan, posebno kod velikih skupova podataka.
- Dobri rezultati za tekstualnu klasifikaciju: Pokazao je odlične performanse kod problema klasifikacije teksta, kao što su filtriranje spam-a i analiza sentimenta.
- Nema potrebe za puno podataka: Za razliku od mnogih drugih algoritama mašinskog učenja, MNB može da radi dobro čak i sa manjim količinama podataka.

Nedostaci ovog modela jesu:

- Pretpostavka o nezavisnosti: Naivna pretpostavka o nezavisnosti između reči nije uvek tačna, što može dovesti do lošijih performansi u određenim situacijama.
- Neefikasan za kontinuirane atribute: MNB nije pogodan za rad sa kontinuiranim atributima, već je dizajniran za rad sa diskretnim podacima.

4.4. Logistička regresija (Logistic Regression)

Logistička regresija je popularan model za binarnu klasifikaciju, posebno zbog svoje jednostavnosti i efikasnosti. Iako se koristi naziv „regresija“, ovaj model je zapravo namenjen klasifikaciji i koristi se za predviđanje verovatnoće da određeni uzorak pripada jednoj od dve klase. Njegova popularnost proizilazi iz jednostavnosti, efikasnosti, i sposobnosti da pruža interpretabilne rezultate.

Kako ovaj model funkcioniše:

Logistička regresija koristi *logit* funkciju za modelovanje verovatnoće da uzorak pripada pozitivnoj klasi (y=1). Formula za logističku regresiju je:

$$P(y = 1|X) = \sigma(z) = \frac{1}{1 + e^{-z}}$$

gde je $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$.

Ova funkcija pretvara linearno kombinovane ulazne varijable (x1, x2, ..., xn) u verovatnoću koja se kreće između 0 i 1. Na osnovu ove verovatnoće, model donosi odluku o klasifikaciji. Ako je verovatnoća veća od određenog praga (najčešće 0.5), uzorak se klasifikuje kao pozitivan (y=1); u suprotnom, kao negativan (y=0).

Treniranje modela podrazumeva optimizaciju parametara β kako bi se minimizirala logistička funkcija gubitka, što je zapravo log-loss ili binarna cross-entropy.

Logistička regresija je implementirana koristeći LogisticRegression iz *sklearn.linear_model* biblioteke. Sledeći deo koda prikazuje kako je model dodat u listu modela:

```
models.append(('Logistic Regression', LogisticRegression()))
```

Treniran je i evaluiran koristeći isti postupak kao i prethodni modeli.

Nakon treniranja modela, procenjena je njegova tačnost, F1 skor, preciznost, odziv i ROC AUC skor. Rezultati evaluacije modela su sledeći:

```
CPU times: user 1min 42s, sys: 10.6 s, total: 1min 53s
Wall time: 1min 8s
Logistic Regression Accuracy Score:  0.8767
Logistic Regression F1 Score:  0.8766926891095372
Logistic Regression Precision Score:  0.8767893593644669
Logistic Regression Recall Score:  0.8767
Logistic Regression ROC AUC Score:  0.8767
[[4345  655]
 [ 578 4422]]
              precision    recall  f1-score   support

    negative       0.88      0.87      0.88      5000
    positive       0.87      0.88      0.88      5000
```

```
    accuracy                           0.88     10000
   macro avg       0.88      0.88      0.88     10000
weighted avg       0.88      0.88      0.88     10000
```

Prednosti ovog modela jesu:

- Jednostavnost: Lako za implementaciju i interpretaciju.
- Efikasnost: Brzo treniranje čak i na velikim datasetima.
- Robustnost: Dobro se ponaša kada su klase linearno separabilne.

Nedostaci ovog modela jesu:

- Linearnost: Nije efikasan za nelinearne odnose između varijabli.
- Osetljivost na multikolinearnost: Kada su prediktori visoko korelisani, performanse modela mogu opasti.

4.5. Klasifikator stohastičkog gradijenta (Stochastic Gradient Descent Classifier)

Stohastički gradijentni spust (SGD) je jedan od najčešće korišćenih optimizacionih algoritama u mašinskom učenju, naročito za obuku linearnih modela, kao što su linearna regresija i SVM. SGD je izuzetno efikasan, posebno kada se radi sa velikim datasetovima, jer ažurira parametre modela iterativno, koristeći pojedinačne primere iz trening skupa umesto celog skupa podataka.

Kako ovaj model funkcioniše:

SGD radi tako što na svakoj iteraciji algoritam nasumično bira jedan uzorak iz trening skupa i ažurira parametre modela na osnovu gradijenta funkcije gubitka u odnosu na taj uzorak. Ovo omogućava brže konvergiranje nego kod klasičnog gradijentnog spusta, posebno na velikim datasetovima.

Formula za ažuriranje parametara je:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla L(\theta_t; x_i, y_i)$$

gde je:

- θ_t trenutni parametar modela,
- η stopa učenja,
- $\nabla L(\theta_t; x_i, y_i)$ gradijent funkcije gubitka za uzorak (x_i, y_i).

Zahvaljujući ovom iterativnom procesu, SGD omogućava modelu da brzo uči i konvergira ka rešenju, iako sa određenim fluktuacijama koje su posledica stohastičkog (nasumičnog) izbora uzoraka.

SGD je implementiran koristeći SGDClassifier iz *sklearn.linear_model* biblioteke. Model je dodat u listu modela koristeći sledeći kod:

```
models.append(('Stochastic Gradient Descent Classifier', SGDClassifier()))
```

Treniran je i evaluiran koristeći isti postupak kao i prethodni modeli.

Nakon treniranja modela, procenjena je njegova tačnost, F1 skor, preciznost, odziv i ROC AUC skor. Rezultati evaluacije modela su sledeći:

```
CPU times: user 52.6 s, sys: 6.26 s, total: 58.9 s
Wall time: 59.7 s
Stochastic Gradient Descent Classifier Accuracy Score:  0.8699
Stochastic Gradient Descent Classifier F1 Score:  0.8698968762239981
Stochastic Gradient Descent Classifier Precision Score:  0.8699355286081675
Stochastic Gradient Descent Classifier Recall Score:  0.8699
Stochastic Gradient Descent Classifier ROC AUC Score:  0.8699
[[4325  675]
 [ 626 4374]]
              precision    recall  f1-score   support

    negative       0.87      0.86      0.87      5000
    positive       0.87      0.87      0.87      5000

    accuracy                           0.87     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.87      0.87      0.87     10000
```

Prednosti ovog modela jesu:

- Efikasnost: Vrlo brzo treniranje, posebno na velikim datasetovima.
- Fleksibilnost: Može se koristiti sa različitim funkcijama gubitka, što ga čini pogodnim za različite vrste problema.

Nedostaci ovog modela jesu:

- Osetljivost na hiperparametre: Performanse mogu varirati u zavisnosti od izbora stope učenja i drugih hiperparametara.
- Nestabilnost: Može biti nestabilan i fluktuirati ako se koristi prevelika stopa učenja.

4.6. Klasifikator stabla odluka (Decision Tree Classifier)

Klasifikator stabla odluka jedan je od najintuitivnijih modela u mašinskom učenju, poznat po svojoj jednostavnosti i lakoći interpretacije. Ovaj model koristi strukturu stabla za donošenje odluka, gde svaki čvor predstavlja atribut ili karakteristiku, svaka grana predstavlja odluku na osnovu tog atributa, a svaki list predstavlja ishod ili klasu kojoj podaci pripadaju.

Kako ovaj model funkcioniše:

Decision Tree funkcioniše tako što deli dataset na manje podskupove koristeći atribute podataka. Ova podela se vrši tako da se maksimalizuje čistoća podskupova, tj. da se što više smanji nečistoća (impurity) unutar čvorova stabla. Proces podele se nastavlja rekurzivno, sve dok svi podaci u podskupu ne pripadaju istoj klasi ili dok dalja podela ne donosi poboljšanja u tačnosti modela. Ovaj proces je poznat kao rekurzivna podela (*recursive partitioning*).

Algoritmi koji se koriste za kreiranje stabala odluka uključuju ID3, C4.5, i CART (Classification and Regression Tree). Ovi algoritmi treniraju stablo minimizovanjem *impurity* funkcije, kao što su entropija ili Gini indeks, koje mere koliko dobro čvor deli podatke.

Decision Tree je implementiran koristeći DecisionTreeClassifier iz *sklearn.tree* biblioteke. Model je dodat u listu modela koristeći sledeći kod:

```
models.append(('Decision Tree Classifier', DecisionTreeClassifier()))
```

Treniran je i evaluiran koristeći isti postupak kao i prethodni modeli.

Nakon treniranja modela, procenjena je njegova tačnost, F1 skor, preciznost, odziv i ROC AUC skor. Rezultati evaluacije modela su sledeći:

```
CPU times: user 5min 10s, sys: 698 ms, total: 5min 10s
Wall time: 5min 12s
Decision Tree Classifier Accuracy Score:  0.7193
Decision Tree Classifier F1 Score:  0.7192777639916857
Decision Tree Classifier Precision Score:  0.7193695050339749
Decision Tree Classifier Recall Score:  0.7193
Decision Tree Classifier ROC AUC Score:  0.7192999999999999
[[3641 1359]
 [1448 3552]]
              precision    recall  f1-score   support

    negative       0.72      0.73      0.72      5000
    positive       0.72      0.71      0.72      5000

    accuracy                           0.72     10000
   macro avg       0.72      0.72      0.72     10000
weighted avg       0.72      0.72      0.72     10000
```

Prednosti ovog modela jesu:

- Jednostavnost i interpretabilnost: Lako je razumeti i vizualizovati model.

- Nema potrebe za skaliranjem podataka: Decision Tree modeli ne zahtevaju normalizaciju ili standardizaciju atributa.
- Rukovanje sa nelinearnošću: Mogu da modeluju kompleksne nelinearne odnose.

Nedostaci ovog modela jesu:

- Overfitting: Decision Trees su skloni overfitting-u, posebno na bučnim podacima.
- Nestabilnost: Mala promena u podacima može rezultirati potpuno drugačijom strukturom stabla.
- Nedostatak generalizacije: Manje su robusni u odnosu na druge modele poput Random Forest-a.

4.7. Klasifikator slučajnih šuma (Random Forest Classifier)

Klasifikator slučajnih šuma je jedan od najpopularnijih i najmoćnijih algoritama za klasifikaciju i regresiju. Ovaj algoritam koristi skup (*ensemble*) metoda, što znači da kombinuje predikcije više osnovnih modela kako bi poboljšao tačnost i robusnost krajnjeg modela. Random Forest je posebno koristan za rukovanje velikim količinama podataka sa mnogim prediktorima.

Kako ovaj model funkcioniše:

Random Forest funkcioniše tako što stvara mnoštvo nezavisnih stabala odluke tokom procesa treniranja. Svako stablo odluke u šumi je trenirano na različitim podskupovima podataka i koristi različite podskupove karakteristika. Proces donošenja odluka se može sumirati u sledeće korake:

1. Bagging (Bootstrap Aggregating): Svako stablo u šumi trenira se na različitim bootstrap-ovanim uzorcima originalnog skupa podataka. Ovaj pristup smanjuje varijansu modela i povećava njegovu sposobnost generalizacije na nepoznate podatke.
2. Random Subspace Method: Na svakoj tački grananja, stablo koristi nasumično izabran podskup karakteristika, što stvara različita stabla sa manjom korelacijom između njih. Ova raznolikost u stablu doprinosi smanjenju overfitting-a i poboljšanju performansi modela.

Na kraju, predikcije svih stabala u šumi se kombinuju (na primer, kroz glasanje u klasifikaciji) kako bi se dobila konačna predikcija. Ovaj pristup ne samo da smanjuje varijansu i overfitting, već rezultira i boljom generalizacijom modela na novim podacima.

Random Forest je implementiran koristeći RandomForestClassifier iz *sklearn.ensemble* biblioteke. Model je dodat u listu modela koristeći sledeći kod:

```
models.append(('Random Forest Classifier', RandomForestClassifier()))
```

Treniran je i evaluiran koristeći isti postupak kao i prethodni modeli.

Nakon treniranja modela, procenjena je njegova tačnost, F1 skor, preciznost, odziv i ROC AUC skor. Rezultati evaluacije modela su sledeći:

```
CPU times: user 3min 55s, sys: 712 ms, total: 3min 56s
Wall time: 3min 57s
Random Forest Classifier Accuracy Score:  0.8468
Random Forest Classifier F1 Score:  0.8467937246709626
Random Forest Classifier Precision Score:  0.8468568290228672
Random Forest Classifier Recall Score:  0.8468
Random Forest Classifier ROC AUC Score:  0.8468
[[4266  734]
 [ 798 4202]]
              precision    recall  f1-score   support

    negative       0.84      0.85      0.85      5000
    positive       0.85      0.84      0.85      5000

    accuracy                           0.85     10000
   macro avg       0.85      0.85      0.85     10000
weighted avg       0.85      0.85      0.85     10000
```

Prednosti ovog modela jesu:

- Visoka tačnost: Random Forest često daje bolju tačnost u poređenju sa pojedinačnim stablima odluke.
- Smanjenje overfitting-a: Kombinovanjem više stabala, model smanjuje rizik od overfitting-a.
- Rukovanje sa nedostajućim vrednostima: Može efikasno rukovati sa nedostajućim vrednostima u podacima.
- Robusnost: Dobro se nosi sa velikim brojem karakteristika i uzoraka.

Nedostaci ovog modela jesu:

- Složenost i vreme treniranja: Random Forest može biti spor za treniranje zbog stvaranja mnogih stabala.
- Manja interpretabilnost: Teže je interpretirati rezultate modela u poređenju sa pojedinačnim stablom odluke.
- Memorijska potrošnja: Može zahtevati značajnu memorijsku potrošnju za skladištenje mnogih stabala.

4.8. Zaključak

Rezultati evaluacije šest različitih modela za analizu sentimenta nad IMDb recenzijama filmova pokazali su jasne razlike u performansama. Na osnovu preciznosti, F1 skora, recall-a, i ROC AUC skora, Logistic Regression i Stochastic Gradient Descent Classifier su se izdvojili kao najefikasniji modeli, sa Logistic Regression-om koji je postigao najvišu tačnost od 87.67%.

Logistička regresija je pokazala izuzetne performanse u ovom specifičnom zadatku analize sentimenta iz nekoliko ključnih razloga. Prvo, priroda teksta kao podatka često rezultira u linearno separabilnim klasama kada se koristi Bag of Words ili TF-IDF kao metod ekstrakcije karakteristika. Logistička regresija, kao linearni model, je idealno prilagođena za rešavanje ovakvih problema, jer može efikasno da izračuna verovatnoću pripadnosti klasi na osnovu linearnih kombinacija karakteristika.

Dodatno, logistička regresija je veoma robusna u prisustvu visokodimenzionalnih podataka, kao što je slučaj sa ovim dataset-om. Sa maksimalnih 10,000 karakteristika koje smo koristili, model je uspeo da zadrži stabilnost i tačnost zahvaljujući inherentnoj otpornosti na overfitting, posebno kada se primenjuju regularizacione tehnike poput L2 regularizacije.

Još jedan bitan faktor je efikasnost algoritma treniranja. Logistička regresija koristi optimizacione tehnike kao što je Newton-Raphson ili gradijentni spust, koje omogućavaju brzo konvergiranje čak i na velikim dataset-ovima. Ovo je posebno značajno u zadacima gde je potrebno obraditi veliku količinu tekstualnih podataka u relativno kratkom vremenskom periodu.

U poređenju sa drugim modelima, poput K-Nearest Neighbors ili Decision Tree Classifier, Logistička regresija je uspela da pruži dosledno visoke rezultate bez značajnih varijacija u performansama. Iako modeli poput Random Forest-a ili Multinomial Naive Bayes-a imaju svoje prednosti, posebno u složenijim ili diskretnim slučajevima, Logistička regresija je pokazala najizbalansiraniji odnos između tačnosti, jednostavnosti implementacije, i efikasnosti.

Zaključno, smatram da je Logistička regresija za ovaj specifičan slučaj pokazala najbolje rezultate zbog svoje sposobnosti da efikasno modeluje linearno separabilne podatke, otpornosti na overfitting, i brzine konvergencije tokom procesa treniranja. Ovi faktori čine je idealnim izborom za analizu sentimenta u ovakvim dataset-ovima, gde je kombinacija tačnosti i efikasnosti ključna za uspešnu klasifikaciju.

5. Literatura

[1] Jurafsky, D., Martin, J.H. (2023). *"Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition".* Pearson.
[2] Liu, B. *(2012). "Sentiment Analysis and Opinion Mining".* Morgan and Claypool Publishers.
[3] Maas, A., Daly, R., Pham, P., Huang, D., Ng, A., & Potts, C. *"Learning Word Vectors for Sentiment Analysis".* [Online], [Accessed: 12-June-2024]
[4] GeeksforGeeks. *"K-Nearest Neighbor(KNN) Algorithm".* [Online], [Accessed: 22-June-2024]
[5] GeeksforGeeks. *"Multinomial Naive Bayes".* [Online], [Accessed: 24-June-2024]
[6] GeeksforGeeks. *"Logistic Regression in Machine Learning".* [Online], [Accessed: 25-June-2024]
[7] GeeksforGeeks. *"Stochastic Gradient Descent Classifier".* [Online], [Accessed: 27-June-2024]
[8] GeeksforGeeks. *"Decision Tree in Machine Learning ".* [Online], [Accessed: 30-June-2024]

6. Listing