Doğal Dil İşleme'de Python ile Metin Normalleştirme Tekniklerinin Kullanımı [9 Yöntem]

Metin normalleştirme, doğal dil işleme (NLP) içindeki önemli bir adımdır. Metin verilerini temizleme ve önişleme yaparak farklı NLP görevleri için tutarlı ve kullanılabilir hale getirir. Bu süreç, büyük/küçük harf normalleştirmesi, noktalama işaretlerinin kaldırılması, durak kelime (stop word) kaldırma, kök çıkarma (stemming) ve lemmatizasyon gibi çeşitli teknikleri içerir. Bu makalede, farklı metin normalleştirme tekniklerini tartışacak ve Python'da örnekler, avantajlar, dezavantajlar ve örnek kodlar sunacağız.

NLP'de metin normalleştirmek için adımlar;

1.Case Normalization (Büyük/Küçük Harf Normalleştirme)

Büyük/küçük harf normalleştirme, metin verisini standart hale getirmek için tüm metni küçük harfe ya da büyük harfe dönüştürmektir. Bu teknik, büyük ve küçük harflerin karışımını içeren metin verileriyle çalışırken faydalıdır.

Example text normalization

Input: “The quick BROWN Fox Jumps OVER the lazy dog.”

Output: “the quick brown fox jumps over the lazy dog.”

Avantajlar

Büyük/küçük harf duyarlılığını ortadan kaldırarak metin verisini tutarlı hale getirir ve işlemesi kolaylaştırır. Verinin boyutunu azaltarak NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Büyük harfler özel isimleri veya vurguyu belirtebilir, bu nedenle bilgi kaybına yol açabilir.

Text normalization code in Python

```
text = "The quick BROWN Fox Jumps OVER the lazy dog."
text = text.lower()
print(text)
```

2. Punctuation Removal(Noktalama İşaretlerinin Kaldırılması)

Noktalama işaretlerinin kaldırılması, metinden özel karakterleri ve noktalama işaretlerini kaldırmaktır. Bu teknik, metin verisi içinde birçok noktalama işareti bulunduğunda faydalıdır, çünkü bu işaretler metni işlemeyi zorlaştırabilir.

Example text normalization

Input: “The quick BROWN Fox Jumps OVER the lazy dog!!!”

Output: “The quick BROWN Fox Jumps OVER the lazy dog”

Avantajlar

Gereksiz karakterleri kaldırarak metni temizler ve işlemesi kolaylaştırır. Verinin boyutunu azaltarak NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Noktalama işaretleri duygu durumunu veya vurguyu belirtebilir, bu nedenle bilgi kaybına yol açabilir. Python'da metin normalleştirme kodu

Text normalization code in Python

```
import string
text = "The quick BROWN Fox Jumps OVER the lazy dog!!!"
text = text.translate(text.maketrans("", "", string.punctuation))
print(text)
```

3. Stop Word Removal (Durak Kelime Kaldırma)

Durak kelime kaldırma, “the” ve “bir” gibi anlamı çok az olan yaygın kelimelerin kaldırılmasıdır. Bu teknik, metin verisi içinde birçok durak kelime bulunduğunda faydalıdır, çünkü bu kelimeler metni işlemeyi zorlaştırabilir.

Example text normalization

Input: “The quick BROWN Fox Jumps OVER the lazy dog.”

Output: “quick BROWN Fox Jumps OVER lazy dog.”

Avantajlar

Gereksiz kelimeleri kaldırarak metni temizler ve işlemesi kolaylaştırır. Verinin boyutunu azaltarak NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Durak kelimeler bağlamı veya duygu durumunu belirtebilir, bu nedenle bilgi kaybına yol açabilir.

Text normalization code in Python

```
from nltk.corpus import stopwords
text = "The quick BROWN Fox Jumps OVER the lazy dog."
stop_words = set(stopwords.words("english"))
words = text.split()
filtered_words = [word for word in words if word not in stop_words]
text = " ".join(filtered_words)
print(text)
```

4. Stemming (Kök Çıkarma)

Kök çıkarma, “koşuyor” kelimesini “koş” olarak kök haline getirerek kelimeleri kök haline getirme işlemidir. Bu yöntem, metin verisi içinde aynı kelimenin birçok farklı versiyonunun bulunduğunda faydalıdır, çünkü bu durum metni işlemeyi zorlaştırabilir.

Example text normalization

Input: “running,runner,ran”

Output: “run,run,run”

Avantajlar

Verinin boyutunu azaltarak NLP algoritmalarının performansını artırır. Kelimenin temel anlamını tanımlamayı kolaylaştırır.

Dezavantajlar

Kök kelime, her zaman doğru kelime olmayabilir, bu nedenle bilgi kaybına yol açabilir. Varolmayan kelimeler üretebilir.

Text normalization code in Python

```
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
text = "running,runner,ran"
words = text.split(",")
stemmed_words = [stemmer.stem(word) for word in words]
text = ",".join(stemmed_words)
print(text)
```

5. Lemmatization

Lemmatizasyon, “koşuyor” kelimesini “koş” olarak kök haline getirirken kelimenin kullanıldığı bağlamı dikkate alarak kelimeleri kök haline getirme işlemidir. Bu teknik, kök çıkarmaya benzer ancak kelimenin bağlamını dikkate alarak daha doğru sonuçlar verir.

Example text normalization

Input: “running,runner,ran”

Output: “run,runner,run”

Avantajlar

Verinin boyutunu azaltarak NLP algoritmalarının performansını artırır. Kelimenin temel anlamını tanımlamayı kolaylaştırır ve bağlamı korur.

Dezavantajlar

Kök çıkarmadan daha fazla hesaplama maliyetine sahip olabilir. Tüm kelimeleri veya formları işleyemeyebilir.

Text normalization code in Python

```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
text = "running,runner,ran"
words = text.split(",")
lemmatized_words = [lemmatizer.lemmatize(word) for word in words]
text = ",".join(lemmatized_words)
print(text)
```

6. Tokenization (Belirteçleme)

Belirteçleme, metni bireysel kelimeler veya ifadeler olarak ayırmak anlamına gelir. Bu teknik, metin verisinin kelime veya ifade düzeyinde analiz edilmesi gerektiğinde faydalıdır, örneğin metin sınıflandırma veya dil çevirisi görevlerinde.

Example text normalization

Input: “The quick BROWN Fox Jumps OVER the lazy dog.”

Output: [“The”, “quick”, “BROWN”, “Fox”, “Jumps”, “OVER”, “the”, “lazy”, “dog.”]

Avantajlar

Metin verisindeki bireysel kelimeleri veya ifadeleri analiz etmeyi sağlar. Kelime veya ifade düzeyinde analiz yapmayı gerektiren NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Metnin anlamı, kelimelerin bağlamına bağlı olarak değişebileceğinden bilgi kaybına neden olabilir. Tüm metin türlerini işleyemeyebilir.

Text normalization code in Python

```
from nltk.tokenize import word_tokenize
text = "The quick BROWN Fox Jumps OVER the lazy dog."
tokens = word_tokenize(text)
print(tokens)
```

7. Replacing synonyms and Abbreviation to their full form to normalize the text in NLP (Eşanlamlıları ve Kısaltmaları Tam Hâllerine Dönüştürme)

Bu teknik, metin verisindeki eşanlamlıları veya kısaltmaları tam hâllerine dönüştürmek gerektiğinde faydalıdır.

Example text normalization

Input: “I’ll be there at 2pm”

Output: “I will be there at 2pm”

Avantajlar

Metin verisini daha okunabilir ve anlaşılır hâle getirir. Kelime veya ifade düzeyinde analiz yapmayı gerektiren NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Metnin anlamı, kelimelerin bağlamına bağlı olarak değişebileceğinden bilgi kaybına neden olabilir. Tüm metin türlerini işleyemeyebilir.

Text normalization code in Python

```
text = "I'll be there at 2pm"
synonyms = {"I'll": "I will", "2pm": "2 pm"}
for key, value in synonyms.items():
   text = text.replace(key, value)
print(text)
```

8. Removing numbers and symbol to normalize the text in NLP (Sayıların ve Sembollerin Kaldırılması)

Bu teknik, metin verilerinde önemli olmayan sayıları ve sembolleri kaldırmak gerektiğinde faydalıdır.

Example text normalization

Input: “I have 2 apples and 1 orange #fruits”

Output: “I have apples and orange fruits”

Avantajlar

Gereksiz sayıları ve sembolleri kaldırarak metni temizler ve işlemesi kolaylaştırır. Verinin boyutunu azaltarak NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Sayılar ve semboller miktarı veya duygu durumunu belirtebilir, bu nedenle bilgi kaybına yol açabilir.

Text normalization code in Python

```
import re
text = "I have 2 apples and 1 orange #fruits"
text = re.sub(r"[\d#]", "", text)
print(text)
```

9. Removing any remaining non-textual elements to normalize the text in NLP (Kalan Metin Dışı Unsurların Kaldırılması)

Bu teknik, metin verilerinde HTML etiketleri, URL'ler ve e-posta adresleri gibi metin dışı unsurları kaldırmak gerektiğinde faydalıdır.

Example text normalization

Input: “Please visit example.com for more information or contact me at info@example.com”

Output: “Please visit for more information or contact me at “

Avantajlar

Gereksiz metin dışı unsurları kaldırarak metni temizler ve işlemesi kolaylaştırır. Verinin boyutunu azaltarak NLP algoritmalarının performansını artırabilir.

Dezavantajlar

Metin dışı unsurlar bağlamı veya duygu durumunu belirtebilir, bu nedenle bilgi kaybına yol açabilir.

Text normalization code in Python

```
import re
text = "Please visit <a href='www.example.com'>example.com</a> for more information or contact me at info@example.com"
text = re.sub(r"(<[^>]+>)|(http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+)", "", text)
print(text)
```

Belirtilmelidir ki, bu adımların belirli bir NLP görevinin gereksinimlerine ve işlenen metin verisinin türüne bağlı olarak uygulanması gerekmektedir.

Metin normalleştirme, tekrarlayan bir süreçtir ve adımlar NLP görevinin gereksinimlerine bağlı olarak birden çok kez tekrarlanabilir.

Keyword normalization techniques in NLP(NLP'de anahtar kelime normalleştirme teknikleri)

Metin verilerindeki anahtar kelimeleri veya ifadeleri standart hale getirip temizlemek için kullanılır, böylece bunlar doğal dil işleme görevlerinde daha kullanışlı hâle gelir.

Metin normalleştirme teknikleri, metin verilerini doğal dil işleme (NLP) görevleri için hazırlamak için temeldir. Her teknik avantajları ve dezavantajları bulunmaktadır ve uygun tekniği belirlemek, belirli bir NLP görevinin gereksinimlerine ve işlenen metin verisinin türüne bağlıdır.

Ayrıca metin normalleştirme işleminin tekrarlayan bir süreç olduğu ve adımların NLP görevinin gereksinimlerine bağlı olarak birden çok kez tekrarlanabileceği unutulmamalıdır.