Modul 2 Praktikum Natural Language Processing
Preprocessing Data Teks mengunakan NLTK

Deskripsi Singkat

Modul ini membahas tentang preprocessing data teks menggunakan Natural Language Toolkit (NLTK). Dalam modul ini, peserta akan mempelajari cara menggunakan regex untuk parsing teks, teknik-teknik penanganan string, dan berbagai metode untuk mengeksplorasi serta memproses data teks sebelum digunakan dalam analisis lebih lanjut.

Tujuan

1. Memahami konsep dasar preprocessing data teks.
2. Menggunakan regex untuk parsing teks.
3. Menerapkan teknik-teknik penanganan string.
4. Mengeksplorasi dan memproses data teks menggunakan NLTK.
5. Menyiapkan data teks untuk analisis lebih lanjut.

Materi 1 - Parsing teks menggunakan regex

RegEx (Regular Expressions) yaitu sekumpulan karakter yang membentuk pola pencarian. Regex sangat berguna dalam berbagai pemrosesan teks, seperti validasi input, pencarian dan penggantian teks, serta ekstraksi informasi dari teks.

Regex	Deskripsi
`[ab]`	Mencocokkan karakter a dan b
`^ab`	Mencocokkan karakter dari a sampai d
`[a-z]`	Mencocokkan karakter dari a sampai z
`[A-Z]`	Mencocokkan karakter dari A sampai Z
`[0-9]`	Mencocokkan karakter dari 0 sampai 9
`\s`	Mencocokkan spasi
`\S`	Mencocokkan karakter bukan spasi
`\d`	Mencocokkan digit
`\D`	Mencocokkan karakter bukan digit
`\w`	Mencocokkan kata
`\W`	Mencocokkan selain kata
`+`	Mencocokkan satu atau lebih karakter
`*`	Mencocokkan nol atau lebih karakter
`?`	Mencocokkan nol atau satu karakter

Untuk simbol-simbol lainnya, silahkan kunjungi
https://www.w3schools.com/python/python_regex.asp.

Mengambil email dari sebuah text

```
import re

doc = "Informasi lebih lanjut hubungi: xyz@abc.com, pqr@mno.com"

addresses = re.findall(r'[\w\.-]+@[\w\.-]+', doc)
for address in addresses:
    print(address)

# Output:
xyz@abc.com
pqr@mno.com
```

Berikut adalah penjelasan dari regex yang digunakan:

- `[\w\.-]+` : Mencocokkan satu atau lebih karakter alfanumerik, titik, atau strip.
- `@` : Mencocokkan karakter @.
- `[\w\.-]+` : Mencocokkan satu atau lebih karakter alfanumerik, titik, atau strip.

Merubah email dari sebuah text

```
import re

doc = "Informasi lebih lanjut hubungi: xyz@abc.com"

email_baru = re.sub(r'([\w\.-]+)@([\w\.-]+)', r'pqr@mno.com', doc)
print(email_baru)

# Output
Informasi lebih lanjut hubungi: pqr@mno.com
```

Function `re.sub()` digunakan untuk mengganti teks yang cocok dengan pola tertentu dengan teks lain.

Melakukan tokenisasi teks menggunakan regex

```
import re

doc = "Saya sedang belajar NLP."

token = re.split(r'\s+', doc)

# Output:
['Saya', 'sedang', 'belajar', 'NLP.']
```

Berikut adalah penjelasan dari regex yang digunakan:

- `\s+` : Mencocokkan satu atau lebih spasi.

Latihan I Preprocessing Data Teks

Pada latihan ini akan menggunakan dataset dari sebuah project yang bermana GUTENBERG EBOOK. Dataset ini berisi teks dari buku-buku yang sudah di public domain.

Pada latihan ini akan menggunakan dataset berikut:
https://www.gutenberg.org/files/2638/2638-0.txt

Jalankan kode berikut untuk mendownload dataset tersebut:

```
import re
import requests

def get_gutenberg_book():
    url = 'https://www.gutenberg.org/files/2638/2638-0.txt'

    raw = requests.get(url).text

    start = re.search(r"\*\*\* START OF THE PROJECT GUTENBERG EBOOK
        THE IDIOT \*\*\*", raw).end()
    stop = re.search(r"II", raw).start()

    text = raw[start:stop]

    return text

book = get_gutenberg_book()
print(book)
```

Selanjutnya adalah mengambil semua kata yang ada pada teks tersebut dan merubahnya menjadi huruf kecil.

```
clean_book = re.sub('[^A-Za-z0-9.]+' , ' ', sentence).lower()
print(clean_book)
```

Selanjutnya adalah mencoba untuk menghitung jumlah kata 'the' yang ada pada teks tersebut.

```
count_the = len(re.findall(r'the', clean_book))
print(count_the)
```

Materi 2 - Handling String pada Python

Teknik Preprocessing Data Teks yang kedua adalah Handling String menggunakan Python. Python memiliki berbagai fungsi bawaan yang dapat digunakan untuk melakukan manipulasi string.

Menggabungkan String

```
s1 = "Hello"
s2 = "World"
s3 = s1 + s2

# Output
HelloWorld
```

Mencari sebuah substring dalam sebuah string

```
s = "I am learning NLP"
print(s.find("learn"))

# Output
5
```

Mengganti sebuah substring dalam sebuah string

```
s = "I am learning NLP"
print(s.replace("NLP", "Natural Language Processing"))

# Output
I am learning Natural Language Processing
```

Slice String

```
String_v1 = "I am exploring NLP"
print(String_v1[5:14]) # Memotong string dari index 5 - 14
# Output
exploring

print(String_v1[:5]) # Memotong string dari index 0 - 5
# Output
I am

print(String_v1[5:]) # Memotong string dari index 5 sampai akhir
# Output
exploring NLP

print(String_v1[-3:]) # Memotong string dari index 3 dari akhir
# Output
NLP

print(String_v1[:-3]) # Memotong string dari index 0-3 dari akhir
# Output
I am exploring
```

Untuk lebih lengkapnya, silahkan kunjungi
https://www.programiz.com/python-programming/methods/string.

Materi 3 - Ekplorasi dan Preprocessing Data Teks

Pada meteri ini akan membahas tentang cara melakukan ekplorasi dan preprocessing data teks menggunakan NLTK dan Pandas. Sebelum mengikuti materi ini, pastikan sudah menginstall NLTK dan Pandas.

Menginstall NLTK dan Pandas

```
pip install nltk
pip install pandas
```

Membaca Data Teks dan Menjadikan DataFrame

```
import pandas as pd

text=[
    'This is introduction to NLP',
    'It is likely to be useful, to people ',
    'Machine learning is the new electrcity',
    'R is good langauage',
    'I like this book',
    'I want more books like this']

df = pd.DataFrame({'tweet': text})
print(df)
```

DataFrame adalah struktur data dua dimensi yang terdiri dari baris dan kolom yang dapat digunakan untuk menyimpan data dalam bentuk tabel. Perintah **pd.DataFrame()** digunakan untuk membuat DataFrame dari data yang diberikan. Silahkan jalankan kode diatas dan perhatikan outputnya.

Menhapus semua tanda baca yang ada pada teks

Dalam preprocessing data teks, seringkali kita perlu menghapus tanda baca yang ada pada teks. Penghapusan ini agar tanda baca tidak mempengaruhi analisis yang akan dilakukan.

```
# Contoh
s = "I. like. This book!"
s1 = re.sub(r'[^\w\s]','',s)
print(s1) # 'I like This book'

# atau bisa dengan menggunakan perulangan
import string

s = "I. like. This book!"

for c in string.punctuation:
    s = s.replace(c, "")

print(s) # 'I like This book'
```

Selanjutnya ubahlah hilangkan semua tanda baca pada teks yang ada pada DataFrame yang sudah dibuat sebelumnya.

```
df['tweet'] = df['tweet'].str.replace('r[^\w\s]','')
print(df['tweet'])
```

Menghapus Stopwords

Stopwords adalah kata-kata yang sering muncul dalam teks dan tidak memiliki makna yang signifikan seperti 'the', 'is', 'and' dan lain-lain atau dalam bahasa Indonesia seperti 'adalah', 'aku', 'apa' dan lain-lain. Sebelum itu silahkan install library NLTK dengan menjalankan perintah berikut untuk mengunduh stopwords yang ada pada NLTK.

```
import nltk
nltk.download('stopwords')
```

```
# Sebelum menghapus stopword, ubahlah teks menjadi huruf kecil

df['tweet'] = df['tweet'].apply(lambda x: " ".join(x.lower() for x
    in x.split()))

from nltk.corpus import stopwords

stopword = stopwords.words('english')
df['tweet'] = df['tweet'].apply(lambda x: " ".join(x for x in x.
    split() if x not in stop))
print(df['tweet'])
```

Standardisasi teks

Standardisasi teks adalah proses mengubah teks menjadi format yang seragam. Misalnya merubah kata-kata yang disingkat menjadi kata lengkap atau kata yang benar.

```
# Buatlah sebuah dictionary yang berisi kata-kata yang akan diubah
lookup_dict = {
    'nlp': 'natural language processing',
    "u": "you",
    'r': 'are',
    'd': 'the',
    'ur': 'your'}

def text_std(input_text):
    words = input_text.split()
    new_words = []
    for word in words:
        word = re.sub(r'[^\w\s]','',word)
        if word.lower() in lookup_dict:
            word = lookup_dict[word.lower()]
            new_words.append(word)
            new_text = " ".join(new_words)
    return new_text

```

```
print(text_std("I like nlp"))

# Output
natural language processing
```

Spelling Correction (Perbaikan Ejaan)

Spelling correction adalah proses untuk memperbaiki kata-kata yang salah pada pengejaannya. Salah satu library yang dapat digunakan untuk spelling correction adalah **TextBlob** ataupun **autocorrect**. Untuk menggunakan TextBlob dan autocorrect, silahkan install library tersebut dengan menjalankan perintah berikut:

```
pip install textblob
pip install autocorrect
```

```
# Menggunakan TextBlob
from textblob import TextBlob

text = "Machine learning is the new electrcity"
new_text = TextBlob(text).correct()

print(new_text) # Output: Machine learning is the new electricity

# Menggunakan autocorrect
from autocorrect import Speller
spell = Speller(lang='en')

text = "Machine learning is the new electrcity"
new_text = spell(text)

print(new_text) # Output: Machine learning is the new electricity
```

Pada contoh diatas, kita menggunakan TextBlob dan autocorrect untuk melakukan spelling correction pada teks yang diberikan. TextBlob akan mengubah kata 'electrcity' menjadi 'electricity' karena kata tersebut salah dalam pengejaannya. Selanjutnya lakukan spelling correction pada teks yang ada pada DataFrame yang sudah dibuat sebelumnya.

```
df['tweet'].apply(lambda x: str(TextBlob(x).correct()))
```

Tokenisasi

Tokenisasi adalah proses memecah teks menjadi bagian-bagian yang lebih kecil yang disebut dengan token. Tokenisasi dapat dilakukan dengan menggunakan TextBlob , NLTK atau menggunakan fungsi split() pada Python.

```
# Menggunakan TextBlob
from textblob import TextBlob

text = "Machine learning is the new electrcity"
new_text = TextBlob(text).words

print(new_text) # Output: ['Machine', 'learning', 'is', 'the', 'new
    ', 'electricity']
```

```
# Menggunakan NLTK
from nltk.tokenize import word_tokenize

text = "Machine learning is the new electrcity"
new_text = word_tokenize(text)

print(new_text) # Output: ['Machine', 'learning', 'is', 'the', 'new
    ', 'electricity']
```

```
# Menggunakan split()
text = "Machine learning is the new electrcity"

new_text = text.split()
print(new_text) # Output: ['Machine', 'learning', 'is', 'the', 'new
    ', 'electricity']
```

Stemming

Stemming adalah proses menghilangkan imbuhan pada kata untuk mendapatkan kata dasar. NLTK memiliki library yang dapat digunakan untuk melakukan stemming yaitu PorterStemmer.

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()

text = "There are many fishes in pound"
new_text = " ".join([stemmer.stem(word) for word in text.split()])
print(new_text) # Output: there are mani fish in pound
```

Untuk melakukan stemming pada bahasa Indonesia, kita dapat menggunakan library Sastrawi. Penggunaan PySastraawi nanti akan dibahas pada pertemuan selanjutnya.

Lemmatization

Lemmatization adalah proses mengubah kata-kata yang ada pada teks menjadi kata dasar. Lemmatization lebih kompleks dibandingkan dengan stemming karena lemmatization memperhitungkan konteks kata dalam kalimat. Contoh lemmatization adalah kata 'leaves' akan diubah menjadi 'leaf' karena kata 'leaf' adalah kata dasar dari kata 'leaves'.

```
import nltk
nltk.download('wordnet')
```

```
# Lemmatization menggunakan TextBlob
from textblob import Word

text = "leaves and leaf"
new_text = " ".join([Word(word).lemmatize() for word in text.split
    ()])
print(new_text) # Output: leaf and leaf
```

```python
# Lemmatization menggunakan NLTK
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

text = "leaves and leaf"
new_text = " ".join([lemmatizer.lemmatize(word) for word in text.
    split()])
print(new_text) # Output: leaf and leaf
```

Tugas

1. Kerjakan seluruh materi yang sudah dijelaskan pada modul diatas.