Modul NLP: Bag of Words, IDF, TF-IDF, dan Word2Vec

Import Library

```
import numpy as np
import pandas as pd
import os
from luwiji.text_proc import illustration
from nltk.tokenize import word_tokenize
from tqdm.auto import tqdm
from gensim.models import Word2Vec
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer,
TfidfVectorizer
```

Data

```
text = ['Ini adalah pensil',
        'Ini adalah pulpen saya dan ini pulpen dia',
        'Saya beli pensil ini',
        'Saya beli pulpen itu']
text
```

I. Bag of Words (BoW)

Definisi : BoW adalah pendekatan sederhana dalam NLP yang mengubah teks menjadi vektor kata-kata.

Langkah-langkah BoW: Menguraikan langkah-langkah utama dalam membuat representasi BoW dari teks. Ini mencakup tokenisasi, penciptaan vektor kata, dan menghitung frekuensi kata.

Representasi Teks: Bagaimana BoW menggambarkan teks sebagai vektor dengan kata-kata sebagai fitur dan frekuensi kata sebagai nilai-nilai dalam vektor.

Syntax:

```
bow = CountVectorizer()
bow_matrix = bow.fit_transform(text)
bow_matrix
```

Jika ditampilkan akan seperti ini:

	adalah	beli	dan	dia	ini	itu	pensil	pulpen	saya
Ini adalah pensil	1	0	0	0	1	0	1	0	0
Ini adalah pulpen saya dan ini pulpen dia	1	0	1	1	2	0	0	2	1
Saya beli pensil ini	0	1	0	0	1	0	1	0	1
Saya beli pulpen itu	0	1	0	0	0	1	0	1	1

III. Inverse Document Frequency (IDF)

A. Konsep Dasar IDF

IDF adalah metrik untuk menilai pentingnya kata dalam koleksi dokumen.

Document Frequency & Inverse Document Frequency

	ini	adalah	pensil	pulpen	saya	beli	itu
Ini adalah pensil	1	1	1	0	0	0	0
Ini adalah pulpen	1	1	0	1	0	0	0
Saya beli pensil ini	1	0	1	0	1	1	0
Saya beli pulpen itu	0	0	0	1	1	1	1
DF	3/4	2/4	2/4	2/4	2/4	2/4	1/4
IDF	4/3	2	2	2	2	2	4

B. Perhitungan IDF

IDF dihitung sebagai logaritma dari jumlah dokumen dalam koleksi dibagi oleh jumlah dokumen yang mengandung kata tertentu.

$$\text{IDF}(t) = \log \frac{1+n}{1+\text{df}(t)} + 1$$

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

A. Konsep Dasar TF-IDF

TF-IDF adalah metrik yang menggabungkan "Term Frequency" (TF) dan "Inverse Document Frequency" (IDF) untuk menilai pentingnya kata dalam teks.

TF mengukur seberapa sering kata muncul dalam dokumen, sedangkan IDF mengevaluasi seberapa unik dan penting kata itu dalam seluruh koleksi dokumen.

B. Perhitungan TF-IDF

TF-IDF dihitung dengan mengalikan nilai TF dan IDF untuk setiap kata dalam dokumen.

Ini menghasilkan skor yang menyoroti kata-kata yang sering muncul dalam dokumen tertentu dan juga relatif unik dalam konteks koleksi dokumen.

$$tf_{ij} = \frac{f_d(i)}{\max\limits_{j \in d} f_d(j)}$$

$$\text{IDF}(t) = \log \frac{1+n}{1+\text{df}(t)} + 1$$

C. Kelebihan dan Kekurangan TF-IDF

Kelebihan: Membantu mengidentifikasi kata-kata kunci, memberikan bobot pada kata-kata penting, dan digunakan dalam berbagai aplikasi seperti pencarian informasi dan analisis teks.

Kekurangan: Mungkin tidak memperhitungkan konteks teks secara menyeluruh dan bisa rumit dalam perhitungan jika koleksi dokumen besar, urutan tidak ngaruh.

Syntax:

```
tfidf = TfidfVectorizer()
tfidf_matrix = tfidf.fit_transform(text)
print(tfidf_matrix)
```

Jika dicetak maka akan keluar hasil:

```
(0, 6)        0.6136667440107333
(0, 0)        0.6136667440107333
(0, 4)        0.4968161174826459
(1, 3)        0.3741092304629382
(1, 2)        0.3741092304629382
(1, 8)        0.23878907574757022
(1, 7)        0.5899040287253255
(1, 0)        0.2949520143626628
(1, 4)        0.4775781514951043
(2, 1)        0.5495783541874525
(2, 8)        0.4449310425319985
(2, 6)        0.5495783541874525
(2, 4)        0.4449310425319985
(3, 5)        0.6142260844216118
(3, 1)        0.4842629000360712
(3, 8)        0.392052553254539
(3, 7)        0.4842629000360712
```

Menampilkan nama fiturnya:

```
tfidf.get_feature_names_out()
```

Output:

```
array(['adalah', 'beli', 'dan', 'dia', 'ini', 'itu', 'pensil', 'pulpen',
       'saya'], dtype=object)
```

Jika diload dalam dataframe hasilnya akan seperti:

```
pd.DataFrame(tfidf_matrix.toarray(), columns=tfidf.get_feature_names_out(),
index=text)
```

	adalah	beli	dan	dia	ini	itu	pensil	pulpen	saya
Ini adalah pensil	0.613667	0.000000	0.000000	0.000000	0.496816	0.000000	0.613667	0.000000	0.000000
Ini adalah pulpen saya dan ini pulpen dia	0.294952	0.000000	0.374109	0.374109	0.477578	0.000000	0.000000	0.589904	0.238789
Saya beli pensil ini	0.000000	0.549578	0.000000	0.000000	0.444931	0.000000	0.549578	0.000000	0.444931
Saya beli pulpen itu	0.000000	0.484263	0.000000	0.000000	0.000000	0.614226	0.000000	0.484263	0.392053

V. Word2Vec

A. Konsep Dasar Word2Vec

Word2Vec adalah teknik dalam NLP yang digunakan untuk menghasilkan representasi vektor kata-kata.

Ini memungkinkan kata-kata untuk direpresentasikan dalam ruang vektor sehingga kata-kata yang serupa memiliki vektor yang mendekati satu sama lain.

B. Algoritma Skip-gram dan Continuous Bag of Words (CBOW)

Algoritma Skip-gram: Membelajari vektor kata berdasarkan kata target dan kata-kata konteks di sekitarnya.

Algoritma Continuous Bag of Words (CBOW): Memprediksi kata target berdasarkan kata-kata konteks di sekitarnya.

C. Kelebihan dan Kekurangan Word2Vec

Kelebihan: Mampu menghasilkan representasi kata-kata yang kaya, mendeteksi hubungan semantik antara kata, dan digunakan dalam berbagai tugas pemrosesan bahasa alami.

Kekurangan: Memerlukan jumlah data yang besar untuk hasil yang optimal, membutuhkan waktu untuk melatih model pada data besar, dan tergantung pada parameter yang sesuai.

D. Implementasi Word2Vec

Word2Vec dapat diimplementasikan menggunakan berbagai framework, seperti Gensim atau TensorFlow. Peserta akan mempelajari cara melatih model Word2Vec pada teks yang diberikan.

Load dataset:

```
df = pd.read_csv('hasil_saya (1).csv')
df = df.drop(columns='Unnamed: 0')
df.head()
```

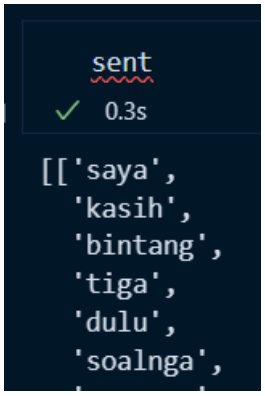
Output:

	userName	content	score
0	Gadri Bandu	Saya kasih bintang tiga dulu soalnga gemenya b...	3
1	070_Putri Azzahra Nurdin	Bagus bgtt. Dari segi story, grafik sampe puzz...	5
2	Novrealita Setiyana	Sejauh ini aku main aku suka banget sama model...	5
3	auah gelap	Semuanya sudah sempurna tapi Sangat disayangka...	3
4	Adi Nugroho	Grafik bagus,story bagus,terutama archon quest...	5

Ubah kedalam format yang diminta Word2Vec

```
sent = [word_tokenize(content.lower()) for content in tqdm(df.content)]
```

Contoh output:



Lakukan training di word2vec

```
# size = neuron
# window = berapa lihat kanan kirinya
# min_count = min berapa kali kata muncul baru dianggap vocab
# workers = jumlah cpu
# iter = epoch
# sg = skipgram (kalau 1 akan pake cbow) default = 0
model = Word2Vec(sent, size=128, window=5, min_count=3, workers=8, iter=1000,
sg=0)
```

Buat direktori untuk menyimpan modelnya

```
os.makedirs("model/w2v/", exist_ok=True)
```

Lakukan save model

```
model.save("model/w2v/review.w2v")
```