PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

1. Akuisisi dan Representasi Data Image

a. Sebelum melakukan akuisisi dan representasi data image, dapat menginstall lalu mengimport library OpenCV CV2 dan Numpy terlebih dahulu.

```
# Import package
import cv2
import numpy as np
```

Untuk membuka citra bernama ‘kitty.png’, maka dapat menggunakan library OpenCV CV2, sebagai berikut:

```
# Membaca citra
img = cv2.imread("C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining II/Week 3/kitty.png")
cv2.imshow("Citra Kitty", img)
cv2.waitKey(0)
```

Berikut adalah hasilnya:

b. Untuk mengetahui ukuran dan matriks dari citra tersebut gunakan kode dibawah ini.

```
# Mengetahui ukuran dan matriks dari citra
print("Ukuran Citra Warna: ", img.shape)
print("Matriks dari Citra Warna pada baris 0 dan kolom 0: ", img[0,0])
```

Hasilnya adalah sebagai berikut:

```
Ukuran Citra Warna:  (373, 293, 3)
Matriks dari Citra Warna pada baris 0 dan kolom 0:  [47 43 32]
```

Citra tersebut berukuran (373, 293, 3) yang artinya citra terbagi menjadi 373 baris dan 293 kolom, dimana pada tiap baris dan kolom (pixel) terdapat 3 kanal (channel), dengan urutan Blue, Green, Red, dimana masing-masing kanal memiliki nilai tersendiri. Sedangkan untuk Hasil Nilai Matriks dari Citra Warna adalah [47 43 32], dimana pada baris ke-0 dan kolom ke-0, nilai kanal Blue adalah 47, kanal Green adalah 43, dan Kanal Red adalah 32.

c. Citra berwarna terdiri dari 3 kanal Red, Green, Blue. Lalu memisahkan suatu citra berwarna menjadi komponen Red, Green, dan Blue. Method dalam OpenCV yang

BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE : SATU (1)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 14 SEPTEMBER 2023

dapat digunakan untuk melakukan pemisahan channel adalah `split()`. Untuk memisahkan ketiga channel, tuliskan kode berikut:

```
# Memisahkan ketiga channel
(blue,green,red)=cv2.split(img)
```

Dengan menggunakan kode diatas, maka telah berhasil memisahkan ketiga channel. Kemudian menampilkan channel Blue dengan menuliskan kode berikut:

```
# Menampilkan channel biru
cv2.imshow("Komponen Biru", blue)
```

Setelah dipisahkan, komponen Blue akan terlihat seperti gambar dibawah. Karena setiap channel direpresentasikan dalam 1 channel saja, maka tidak terlihat warna Blue dan seperti citra grayscale karena hanya memiliki 1 channel.

Agar channel Blue bisa menampilkan warna sesuai channel-nya maka harus direpresentasikan dalam 3 channel menggunakan method `merge()`. Kita harus membuat matriks berisi nilai ‘0’ yang memiliki ukuran sama dengan ukuran gambar kita (gambar ‘kitty.png’ memiliki ukuran 373 x 293), kemudian matriks tersebut digunakan untuk mengisi channel Red dan Green.

```
# Membuat matrix berisi 0 yang berukuran sesuai image asli
zeroMatrix = np.zeros(img.shape[:2], img.dtype)
m = zeroMatrix
blue_img = cv2.merge([blue, m,m])
combine_img = np.hstack((img, blue_img))

# Menampilkan citra gabungan dalam satu jendela
cv2.imshow("Semua Citra", combine_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output dari kode diatas adalah sebagai berikut:

PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

- **Modifikasi kode bagian (a) agar dapat menampilkan citra dalam grayscale.**
  Untuk menampilkan citra gambar 'kitty.png' dalam grayscale, maka dapat menggunakan kode di bawah ini.

```
# Convert the image to grayscale
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

  Kemudian, untuk membuka citra dapat menggunakan kode berikut.

```
# Display the grayscale image
cv2.imshow('Grayscale Image', gray_img)

# Wait for a key press and then close the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```

  Berikut hasilnya:

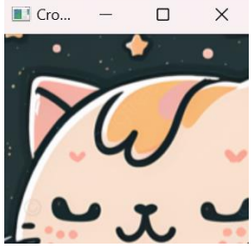
- **Modifikasi kode bagian (a) agar bisa melakukan crop pada citra (terserah dibagian mana saja).**
  Untuk melakukan crop pada citra dapat menggunakan kode berikut.

```
# Define the coordinates of the region you want to crop
# Format: (y_start:y_end, x_start:x_end)
x_start, x_end = 0, 200  # Koordinat horizontal
y_start, y_end = 0, 170  # Koordinat vertikal

# Crop the image
cropped_img = img[y_start:y_end, x_start:x_end]
```

  Kemudian, untuk menampilkan hasil crop, dapat menggunakan kode di bawah ini.

```
# Display the cropped image
cv2.imshow('Cropped Image', cropped_img)

# Wait for a key press and then close the window
cv2.waitKey(0)
```

  Berikut adalah hasil dari kode di atas:



PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

- **Modifikasi kode bagian (b) agar dapat menampilkan ukuran citra grayscale dari 'kitty.png' dan nilai matriks dari citra grayscale pada baris ke-0 dan kolom ke-0. Apakah hasilnya berbeda dengan bagian (b), jelaskan alasannya.**

  Untuk mengetahui ukuran dan matriks dari citra grayscale dari 'kitty.png' dapat menggunakan kode dibawah ini.

```
# Mengetahui ukuran dan matriks dari citra grayscale
print("Ukuran Citra Warna Grayscale: ", gray_img.shape)
print("Matriks dari Citra Warna Grayscale pada baris 0 dan kolom 0: ", gray_img[0,0])
```

  Hasilnya adalah sebagai berikut:

```
Ukuran Citra Warna Grayscale:  (373, 293)
Matriks dari Citra Warna Grayscale pada baris 0 dan kolom 0:  40
```

  Citra tersebut berukuran (373, 293) yang artinya citra terbagi menjadi 373 baris dan 293 kolom. Sedangkan untuk Hasil Nilai Matriks dari Citra Grayscale pada baris ke-0 dan kolom ke-0 adalah 40.

  Hasil Ukuran dan Nilai Matriks pada Citra Warna dan Citra Grayscale berbeda, dimana:

  - Ukuran Citra:
    1) Citra Warna: Ukuran citra warna adalah (373, 293, 3). Angka-angka ini mengindikasikan bahwa citra ini memiliki 3 dimensi, yaitu lebar (373 piksel), tinggi (293 piksel), dan 3 saluran warna (RGB).
    2) Citra Warna Grayscale: Ukuran citra grayscale adalah (373, 293). Ini menunjukkan bahwa citra grayscale hanya memiliki dua dimensi, yaitu lebar (373 piksel) dan tinggi (293 piksel), karena citra grayscale hanya memiliki satu saluran warna.
  - Matriks Nilai Piksel:
    1) Citra Warna: Matriks dari citra warna pada baris 0 dan kolom 0 adalah [47 43 32]. Ini berarti pada koordinat (0, 0) dari citra warna, terdapat tiga nilai warna, masing-masing untuk saluran Red (47), Green (43), dan Blue (32). Oleh karena itu, pada titik ini memiliki informasi tentang warna.
    2) Citra Warna Grayscale: Matriks dari citra grayscale pada baris 0 dan kolom 0 adalah 40. Ini berarti pada koordinat (0, 0) dari citra grayscale, terdapat satu nilai tunggal, yaitu tingkat kecerahan atau intensitas cahaya, yang dalam hal ini adalah 40. Citra grayscale hanya menyimpan informasi tentang tingkat kecerahan dan tidak memiliki informasi warna.

- **Modifikasi kode bagian (c) untuk menampilkan channel Green dan Red.**

  Sebelum menampilkan channel Green dan Red, dilakukan pemisahan ketiga channel seperti yang sudah dilakukan sebelumnya dengan menuliskan kode berikut:

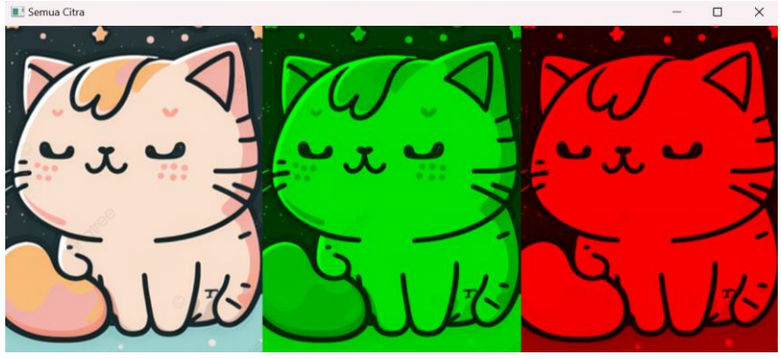
```
# Memisahkan ketiga channel
(blue,green,red)=cv2.split(img)
```

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

Dengan menggunakan kode diatas, maka telah berhasil memisahkan ketiga channel. Kemudian menampilkan channel Green dan Red dengan menuliskan kode berikut:

```
# Menampilkan channel Green dan Red
zeroMatrix = np.zeros(img.shape[:2], img.dtype)
m = zeroMatrix
red = cv2.merge([m, m,red])
green = cv2.merge([m, green,m])

combined_img = np.hstack((img, green, red))

# Menampilkan citra gabungan dalam satu jendela
cv2.imshow("Semua Citra", combined_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Berikut output dari kode di atas.



- **Simpanlah file menjadi format selain format awal (PNG jadi JPG, JPG jadi PNG) dengan menggunakan method `imwrite` pada OpenCV, apakah terdapat perbedaan nilai array pada file citra asli dan file dengan format baru tsb? Jelaskan alasannya.**

Untuk menyimpan file dengan format lain, dapat menggunakan method `imwrite()` pada OpenCV seperti di bawah ini.

```
# Menyimpan citra ke dalam format JPG
cv2.imwrite("kitty.jpg", img)

# Menutup semua jendela
cv2.destroyAllWindows()
```

Output dari kode di atas sebagai berikut.

File ‘kitty.png’ telah tersimpan difolder yang sama dengan format yang telah berubah menjadi ‘kitty.jpg’.

Untuk menampilkan nilai array pada file citra asli dan file dengan format baru, dapat menggunakan kode di bawah ini.

```
# Menampilkan array citra asli dan citra format terbaru
import cv2
img = cv2.imread("C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining II/Week 3/kitty.png")
img2 = cv2.imread("C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining II/Week 3/kitty.jpg")
img
img2
```

BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE

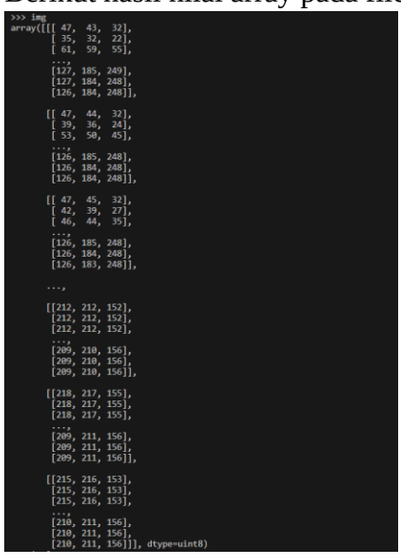
PRAKTIKUM KE : SATU (1)

NAMA : FARIZA SHIELDA AKZATRIA

NIM : 162112133026

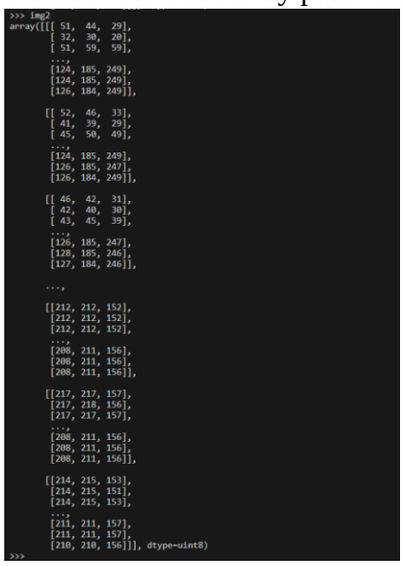
TGL PRAKTIKUM : 14 SEPTEMBER 2023

Berikut hasil nilai array pada file citra asli (format png).



```
>>> img
array([[[ 47,  43,  32],
        [ 35,  32,  22],
        [ 61,  59,  55],
        ...,
        [127, 185, 249],
        [127, 184, 248],
        [126, 184, 248]],

       [[ 47,  44,  32],
        [ 39,  36,  24],
        [ 53,  50,  45],
        ...,
        [126, 185, 248],
        [126, 184, 248],
        [126, 184, 248]],

       [[ 47,  45,  32],
        [ 42,  39,  27],
        [ 46,  44,  35],
        ...,
        [126, 185, 248],
        [126, 184, 248],
        [126, 183, 248]],

       ...,

       [[212, 212, 152],
        [212, 212, 152],
        [212, 212, 152],
        ...,
        [209, 210, 156],
        [209, 210, 156],
        [209, 210, 156]],

       [[218, 217, 155],
        [218, 217, 155],
        [218, 217, 155],
        ...,
        [209, 211, 156],
        [209, 211, 156],
        [209, 211, 156]],

       [[215, 216, 153],
        [215, 216, 153],
        [215, 216, 153],
        ...,
        [210, 211, 156],
        [210, 211, 156],
        [210, 211, 156]]], dtype=uint8)
```

Berikut hasil nilai array pada file citra format terbaru (format jpg).



```
>>> img2
array([[[ 51,  44,  29],
        [ 32,  30,  20],
        [ 51,  59,  59],
        ...,
        [124, 185, 249],
        [124, 185, 249],
        [126, 184, 249]],

       [[ 52,  46,  33],
        [ 41,  39,  29],
        [ 45,  50,  49],
        ...,
        [124, 185, 249],
        [126, 185, 247],
        [126, 184, 249]],

       [[ 46,  42,  31],
        [ 42,  40,  30],
        [ 43,  45,  39],
        ...,
        [126, 185, 247],
        [128, 185, 246],
        [127, 184, 246]],

       ...,

       [[212, 212, 152],
        [212, 212, 152],
        [212, 212, 152],
        ...,
        [208, 211, 156],
        [208, 211, 156],
        [208, 211, 156]],

       [[217, 217, 157],
        [217, 218, 156],
        [217, 217, 157],
        ...,
        [208, 211, 156],
        [208, 211, 156],
        [208, 211, 156]],

       [[214, 215, 153],
        [214, 215, 151],
        [214, 215, 153],
        ...,
        [211, 211, 157],
        [211, 211, 157],
        [210, 210, 156]]], dtype=uint8)
>>>
```

Perbedaan antara kedua array ini dapat dilihat dari nilai-nilai piksel yang berbeda di dalamnya. Di bawah ini adalah perbedaan yang terlihat:

- Warna Piksel:
  1) img: Piksel dalam array img memiliki saluran warna RGB yang bervariasi dalam rentang nilai tertentu. Misalnya, dalam citra img, Anda dapat melihat nilai-nilai seperti [47, 43, 32], [35, 32, 22], yang mewakili saluran merah, hijau, dan biru dari setiap piksel.
  2) img2: Piksel dalam array img2 juga memiliki saluran warna RGB, tetapi nilainya berbeda dari img. Contohnya, [51, 44, 29], [32, 30, 20].

BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE : SATU (1)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 14 SEPTEMBER 2023

- Kecemerlangan Piksel:
  Baik img maupun img2 memiliki citra dengan tingkat kecerahan yang berbeda di beberapa piksel. Misalnya, dalam img, dapat terlihat nilai seperti 127, 185, 249, yang mewakili kecerahan dalam saluran RGB, sedangkan dalam img2, memiliki nilai seperti 124, 185, 249, yang juga mewakili tingkat kecerahan dalam saluran RGB.
- Bentuk Array:
  Kedua array ini memiliki bentuk (shape) yang sama, yaitu (373, 293, 3), yang mengindikasikan bahwa keduanya adalah gambar berwarna dengan lebar 373 piksel, tinggi 293 piksel, dan tiga saluran warna RGB. Perbedaan utama di sini adalah warna piksel dan nilai tingkat kecerahan yang berbeda antara kedua gambar. Selain itu, bentuk array mereka tetap sama, menunjukkan bahwa keduanya masih merupakan citra berwarna dengan tiga saluran warna (RGB).

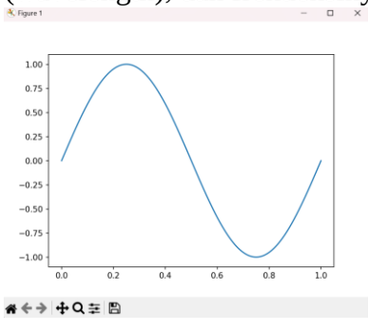
2. Akuisisi dan Representasi Data Audio

a. Sebelum membuat sinyal audio berupa gelombang, install dan import beberapa library yang dibutuhkan. Kemudian harus menentukan dahulu sampling rate, frekuensi, dan panjang menggunakan kode di bawah ini.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile
import librosa
import matplotlib.pyplot as plt
```

```
# Menentukan sampling rate
sr=44100
# Menentukan frequency dalam satu detik
freq=1
# Menentukan time (sumbu x)
length=1

# Membuat fungsi linear dari nol s/d time dengan titik berjumlah sr
# 1/sr --> stepsize, karena kita mau ada sr titik per detik
t=np.arange(0, length, 1.0/sr)
# Membuat wave dengan sin function wave = A*sin(2*pi*freq*t)
signal=np.sin(np.pi*2*freq*t)

# Range satu cycle dalam satu detik, range 1 s/d -1, starting point 0
plt.plot(t, signal)
plt.show()
```

Hasilnya kode diatas adalah gambar gelombang dibawah ini. Gelombang yang dihasilkan memiliki amplitudo antara 1 dan -1, terdapat 1 gelombang dalam 1 detik (wavelength), dan frekuensinya adalah 1.



BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE : SATU (1)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 14 SEPTEMBER 2023

Lalu simpan sinyal gelombang ke dalam format wav dengan nama ‘file.wav’ menggunakan kode dibawah ini.

```
# Menyimpan sinyal gelombang ke dalam format wav
wavfile.write("file.wav", sr, signal)
```

Dalam folder kita akan tersimpan file.wav seperti ini:

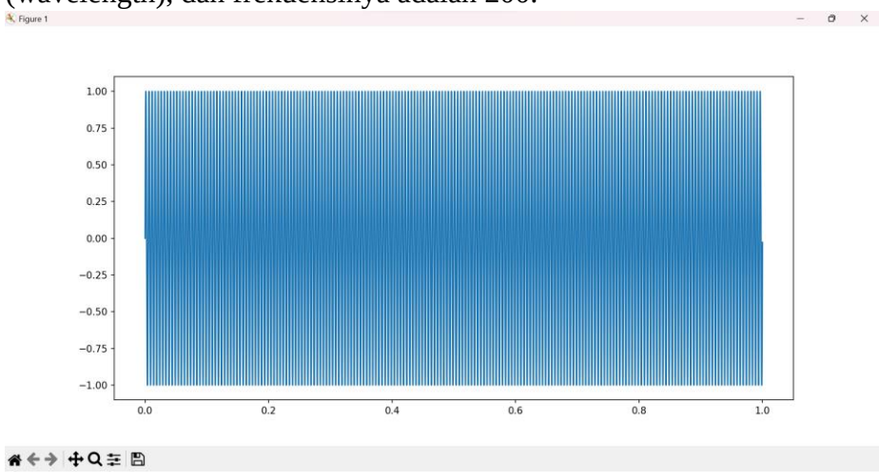
file.wav

Ketika mendengar file tersebut tidak ada suara yang muncul karena dalam program diatur frekuensinya adalah 1, sementara telinga manusia hanya bisa mendengar bunyi yang memiliki frekuensi 20 hingga 20.000 Hz.

b. Supaya terdengar manusia, harus mengganti frekuensinya. Misalkan, mengubah frekuensinya menjadi 200.

```
# Mengubah frequensi agar terdengar manusia
freq = 200
signal2 = np.sin(np.pi*2*freq*t)
plt.plot(t, signal2)
plt.show()
```

Hasilnya kode diatas adalah gambar gelombang dibawah ini. Gelombang yang dihasilkan memiliki amplitudo antara 1 dan -1, terdapat 200 gelombang dalam 1 detik (wavelength), dan frekuensinya adalah 200.



Lalu simpan sinyal gelombang ke dalam format wav dengan nama ‘file1.wav’ menggunakan kode dibawah ini.

```
# Menyimpan sinyal gelombang ke dalam format wav
wavfile.write("file1.wav", sr, signal2)
```

Dalam folder kita akan tersimpan file1.wav seperti ini:

file1.wav

Ketika mendengar file tersebut sudah ada suara yang muncul karena dalam program diatur frekuensinya adalah 200, karena telinga manusia bisa mendengar bunyi yang memiliki frekuensi 20 hingga 20.000 Hz.

PRAKTIKUM DATA MINING II

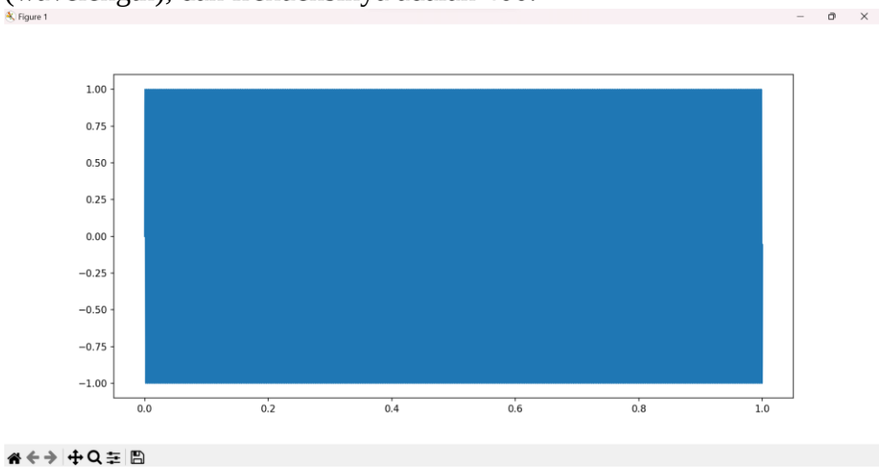
BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

- **Modifikasi kode bagian (a) agar membuat gelombang suara dengan frekuensi 400. Berapa panjang gelombangnya sekarang? Apakah file suara yang dihasilkan frekuensi 400 berbeda dengan file suara yang dihasilkan kode bagian (b)? jelaskan alasannya.**

  Menentukan sampling rate, frekuensi, dan panjang menggunakan kode di bawah ini.

```
# Modifikasi kode (a) agar membuat gelombang suara dengan frekuensi 400
# Menentukan sampling rate
sr = 44100
# Menentukan frequency
freq = 400
# Menentukan length/panjang
length = 1
# Membuat fungsi linear dari 0 s/d length dengan titik berjumlah sr
t = np.arange(0, length, 1.0/sr)
# Membuat wave dengan sin function wave = A*sin(2*pi*freq*t)
signal3 = np.sin(np.pi*2*freq*t)
plt.plot(t, signal3)
plt.show()
```

  Hasilnya kode diatas adalah gambar gelombang dibawah ini. Gelombang yang dihasilkan memiliki amplitudo antara 1 dan -1, terdapat 1 gelombang dalam 1 detik (wavelength), dan frekuensinya adalah 400.



  File suara yang dihasilkan dengan frekuensi 400 berbeda dengan hasil kode bagian (b) yang memiliki frekuensi 200. Perbedaan dalam frekuensi menghasilkan perbedaan dalam tinggi rendahnya nada. Frekuensi 400 akan menghasilkan nada yang lebih tinggi dibandingkan dengan frekuensi 200. Dengan kata lain, meskipun panjang gelombang (waktu) tetap sama, perbedaan dalam frekuensi menghasilkan perbedaan dalam tinggi nada. Frekuensi adalah ukuran berapa kali gelombang lengkap berulang dalam satu detik (Hz). Ketika frekuensi meningkat (misalnya dari 200 Hz menjadi 400 Hz), gelombang lengkap berulang dua kali lebih cepat dalam satu detik, sehingga nada yang dihasilkan terdengar lebih tinggi.

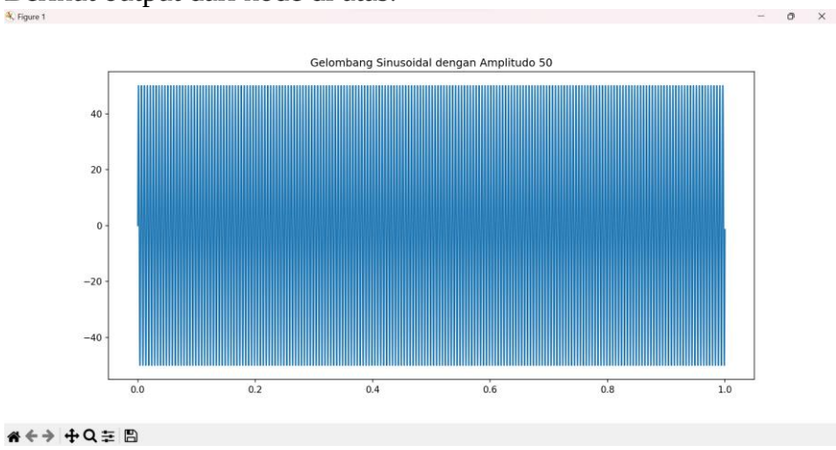
BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE : SATU (1)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 14 SEPTEMBER 2023

- **Modifikasi kode bagian (b) dengan mengganti nilai amplitudo menjadi 50. Pada output gelombang, berapakah panjang dan tinggi gelombang sekarang? Apakah terdapat perbedaan bunyi dengan hasil suara dari kode bagian (b)? jelaskan alasannya.**

  Membuat gelombang suara sinusoidal dengan frekuensi yang sama (400 Hz) tetapi dengan nilai amplitudo yang berbeda (50) dapat menggunakan kode berikut.

```
# Modifikasi kode (b) dengan mengganti nilai amplitudo menjadi 50
# Menentukan sampling rate
sr = 44100
# Menentukan frequency dalam 1 detik
freq = 200
# Menentukan time (sumbu x)
length = 1

# Membuat fungsi linear dari 0 s/d time dengan titik berjumlah sr
# 1/sr --> stepsize, karena kita mau ada sr titik per detik
t = np.arange(0, length, 1.0/sr)

# Membuat wave dengan sin function wave = A*sin(2*pi*freq*t)
amplitude = 50
signal4 = amplitude * np.sin(np.pi*2*freq*t)

plt.plot(t, signal4)
plt.title('Gelombang Sinusoidal dengan Amplitudo 50')
plt.show()
```

  Berikut output dari kode di atas:



  Panjang gelombang tetap tidak berubah karena tidak mengubah nilai variabel length masih tetap 1 detik. Tinggi gelombang (amplitudo) sekarang adalah 50. Ini berarti tinggi gelombang yang dihasilkan sekarang lebih besar daripada yang dihasilkan dalam kode sebelumnya (dengan amplitudo 1). Perubahan amplitudo akan memengaruhi tinggi rendahnya volume suara. Dalam kode ini, tinggi gelombang yang lebih besar (50) akan menghasilkan suara yang lebih keras atau lebih nyaring dibandingkan dengan hasil suara dari kode sebelumnya (amplitudo 1). Ini berarti ada perbedaan bunyi yang signifikan antara hasil suara dari kode ini dengan hasil suara dari kode (b) karena perubahan dalam amplitudo. Tinggi gelombang yang lebih besar menghasilkan suara yang lebih keras atau lebih nyaring.

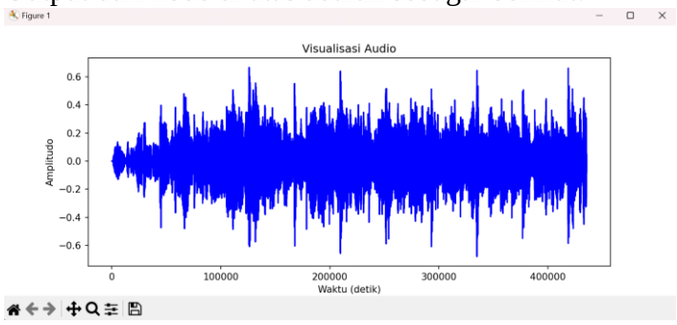
PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

- **Rekam suara baru dari device anda, lakukan visualisasi suara tersebut**
  Setelah melakukan perekaman suara, maka untuk memunculkan visualisasinya dapat menggunakan kode berikut.

```
# Import audio
audio_file = "C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining II/Week 3/Test.wav"
audio_data, sr = librosa.load(audio_file)

# Visualisasi audio
plt.figure(figsize=(10, 4))
plt.plot(audio_data, color='b')
plt.xlabel('Waktu (detik)')
plt.ylabel('Amplitudo')
plt.title('Visualisasi Audio')
plt.show()
```

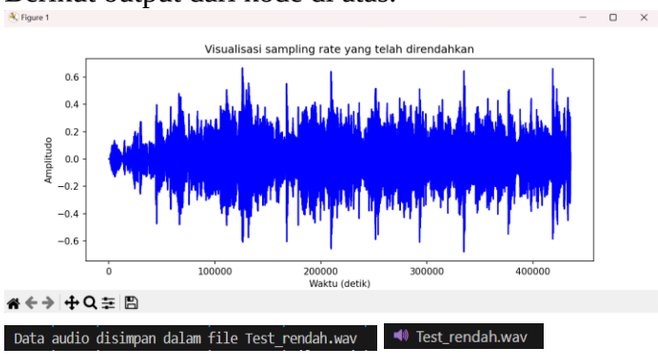
  Output dari kode di atas adalah sebagai berikut:



- **Coba untuk ubah sampling rate suara yang sama menjadi jauh lebih rendah, dengarkan, lalu apa efeknya? Jelaskan alasannya**
  Mengubah sampling rate suara menjadi jauh lebih rendah dapat menggunakan kode berikut.

```
# Mengubah sampling rate suara yang sama menjadi jauh lebih rendah
# Import audio dengan sampling rate asli
audio_file = "C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining II/Week 3/Test.wav"

# Membaca data audio dari file dengan sampling rate 2300 Hz
data, sr = librosa.load(audio_file, sr=2300)

# Visualisasi
plt.figure(figsize=(10, 4))
plt.plot(audio_data, color='b')
plt.xlabel("Waktu (detik)")
plt.ylabel("Amplitudo")
plt.title("Visualisasi sampling rate yang telah direndahkan")
plt.show()

# Menyimpan data audio ke dalam file .wav dengan sampling rate 2300 Hz
output_file = "Test_rendah.wav"
wavfile.write(output_file, sr, data)
print(f"Data audio disimpan dalam file {output_file}")
```

  Berikut output dari kode di atas.



BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE : SATU (1)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 14 SEPTEMBER 2023

Setelah mendengarkan file audio terbaru dengan sampling ratenya yang telah direndahkan, suara menjadi terdengar lebih kasar, dan kurang jelas. Terlebih lagi, hal ini dapat mengakibatkan distorsi audio, di mana komponen frekuensi tinggi dalam audio mungkin terdengar sebagai frekuensi yang lebih rendah, menciptakan efek suara yang tidak diinginkan. Hasilnya, suara tersebut dapat terdengar seperti suara yang dihasilkan oleh komputer atau robot, yang sangat berbeda dari karakter suara aslinya. Mengubah sampling rate suara menjadi jauh lebih rendah akan mengakibatkan pengurangan kualitas audio secara signifikan. Hal ini disebabkan oleh fakta bahwa dengan mengurangi sampling rate, dapat menghilangkan banyak informasi yang relevan dalam audio.

3. Akuisisi dan Representasi Data Text

a. ASCII

Fungsi ‘ord’ pada python digunakan untuk mengubah karakter menjadi ASCII code, sedangkan fungsi ‘char’ digunakan untuk mengubah ASCII code menjadi karakter.

```
char = 'A'
print(ord(char))
```

Berikut output dari kode di atas.

```
65
```

```
ascii = 65
print(chr(ascii))
```

Berikut output dari kode di atas.

```
A
```

b. One-Hot Encoding

Kode berikut digunakan untuk melakukan one-hot encoding pada suatu kalimat.

```
from numpy import array
from numpy import argmax
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder

docs = 'I ate an apple'

#memisah kalimat menjadi kolom
split_docs = docs.split(' ')
data = [doc.split(' ') for doc in split_docs]
values = array(data).ravel()

#integer code
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(values)
print(integer_encoded)

#binary encode
onehot_encoder=OneHotEncoder(sparse=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoded = onehot_encoder.fit_transform(integer_encoded)
print(onehot_encoded)
```

Berikut index dari tiap kata dalam kalimat:

```
[0 3 1 2]
```

Yang memiliki arti:

I	ate	an	apple
0	3	1	2

BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE

PRAKTIKUM KE : SATU (1)

NAMA : FARIZA SHIELDA AKZATRIA

NIM : 162112133026

TGL PRAKTIKUM : 14 SEPTEMBER 2023

Berikut adalah hasil one-hot encoding untuk kalimat “I ate an apple”.

```
[[1. 0. 0. 0.]
 [0. 0. 0. 1.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]]
```

Sehingga [[1 0 0 0] [0 0 0 1] [0 1 0 0] [0 0 1 0]] merupakan one-hot encoding dari masing-masing kata “I ate an apple”.

c. CountVectorizer

Kode berikut digunakan untuk menghitung CountVectorizer pada suatu korpus.

```
# CountVectorizer
from sklearn.feature_extraction.text import CountVectorizer

text = ["everybody love nlp", "nlp is so cool",
"nlp is all about helping machines process language",
"this tutorial is on basic nlp technique"]

vectorizer = CountVectorizer()

# tokenisasi dan membuat vocab
vectorizer.fit(text)
print(vectorizer.vocabulary_)

# encode dokumen
vector = vectorizer.transform(text)

# hasil encode vektor
print(vector.shape)
print(vector.toarray())
```

Daftar vocabulary atau kata dari dokumen adalah sebagai berikut:

```
{'everybody': 4, 'love': 8, 'nlp': 10, 'is': 6, 'so': 13, 'cool': 3, 'all': 1, 'about': 0, 'helping': 5, 'machines': 9, 'process': 12, 'language': 7, 'this': 15, 'tutorial': 16, 'on': 11, 'basic': 2, 'technique': 14}
```

Ukuran vektor adalah **`(4,16)`**, karena dalam korpus terdapat **`4`** dokumen, dimana terdapat **`16`** kata unik dalam korpus tersebut.

Sehingga, CountVectorizer output pada korpus tersebut adalah sebagai berikut:

```
[[0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0]
 [0 0 0 1 0 0 1 0 0 0 1 0 0 1 0 0]
 [1 1 0 0 0 1 1 1 0 1 1 0 1 0 0 0]
 [0 0 1 0 0 0 1 0 0 0 1 1 0 0 1 1]]
```

d. TF-IDF

Kode berikut digunakan untuk menghitung TF-IDF pada suatu korpus.

```
#TF-IDF
from sklearn.feature_extraction.text import TfidfVectorizer
text1 = ['i love nlp', "nlp is so cool",
"nlp is all about helping machines process language",
"this tutorial is on basic nlp technique"]

tf = TfidfVectorizer()
txt_fitted = tf.fit(text1)
txt_transformed = txt_fitted.transform(text1)

idf = tf.idf_
print(dict(zip(txt_fitted.get_feature_names_out(), idf)))
```

Berikut adalah hasilnya:

```
{'about': 1.916290731874155, 'all': 1.916290731874155, 'basic': 1.916290731874155, 'cool': 1.916290731874155, 'helping': 1.916290731874155, 'is': 1.2231435513142097, 'language': 1.916290731874155, 'love': 1.916290731874155, 'machines': 1.916290731874155, 'nlp': 1.0, 'on': 1.916290731874155, 'process': 1.916290731874155, 'so': 1.916290731874155, 'technique': 1.916290731874155, 'this': 1.916290731874155, 'tutorial': 1.916290731874155}
```

Kata ‘nlp’ memiliki nilai paling kecil karena kata tersebut merupakan kata yang paling sering muncul dalam korpus, terdapat kata ‘nlp’ pada tiap kalimat. Diikuti dengan stopword ‘is’ dengan nilai `1.2231435513142097`, karena kata ‘is’ muncul di tiga kalimat dalam korpus.

PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

- **Modifikasi kode bagian (a) agar bisa menampilkan ASCII code untuk kata ‘datA mining’**

  Untuk memodifikasi bagian (a) agar dapat menampilkan ASCII code pada kata ‘datA mining’ dapat menggunakan kode berikut:

```
text = 'datA mining'
for char in text:
    ascii_code = ord(char)
    print(f"Karakter '{char}' memiliki ASCII code: {ascii_code}")
```

  Output dari kode di atas adalah:

```
Karakter 'd' memiliki ASCII code: 100
Karakter 'a' memiliki ASCII code: 97
Karakter 't' memiliki ASCII code: 116
Karakter 'A' memiliki ASCII code: 65
Karakter ' ' memiliki ASCII code: 32
Karakter 'm' memiliki ASCII code: 109
Karakter 'i' memiliki ASCII code: 105
Karakter 'n' memiliki ASCII code: 110
Karakter 'i' memiliki ASCII code: 105
Karakter 'n' memiliki ASCII code: 110
Karakter 'g' memiliki ASCII code: 103
```

- **Tambahkan kode bagian (b) agar bisa menampilkan kembali kata pertama yang di lakukan one-hot encoding**

  Untuk menambahkan kode bagian (b) agar bisa menampilkan kembali kata pertama yang di lakukan one-hot encoding dapat menggunakan kode di bawah ini.

```
# Tambahkan kode (b) agar bisa menampilkan kembali kata pertama yang dilakukan One-Hot Encoding

from sklearn.preprocessing import LabelEncoder, OneHotEncoder
import numpy as np

docs = "I ate an apple"

# Memisahkan kalimat menjadi token
split_docs = docs.split(" ")
data = [doc.split(" ") for doc in split_docs]
values = np.array(data).ravel()

# Integer Encode
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(values)

# Binary Encode
onehot_encoder = OneHotEncoder(sparse=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoded = onehot_encoder.fit_transform(integer_encoded)

# Menampilkan kata pertama yang dilakukan one-hot encoding
first_word_encoded = onehot_encoded[0].reshape(1, -1)
# Mengembalikan representasi integer dari encoding one-hot
first_word_integer_encoded = np.argmax(first_word_encoded)
# Mengembalikan kata pertama dalam bentuk string sesuai dengan encoding one-hot
first_word_original = label_encoder.inverse_transform([first_word_integer_encoded])

print(f"Kata pertama dalam one-hot encoding: {first_word_encoded}")
print(f"Kata pertama dalam bentuk integer: {first_word_integer_encoded}")
print(f"Kata pertama dalam bentuk string: {first_word_original[0]}")
```

  Berikut output dari kode di atas.

```
Kata pertama dalam one-hot encoding: [[1. 0. 0. 0.]]
Kata pertama dalam bentuk integer: 0
Kata pertama dalam bentuk string: I
```

PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

- **Download file https://www.gutenberg.org/files/16328/16328-0.txt, kemudian lakukan CountVectorizer dan TF-IDF pada korpus tersebut. Jelaskan hasil yang didapatkan.**

```
# Impor modul CountVectorizer dan TfidfVectorizer dari scikit-learn
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

# Membaca teks dari file "gutenberg.txt"
with open("gutenberg.txt", "r", encoding="utf-8") as file:
    corpus = file.read()

# Inisialisasi CountVectorizer dengan pengaturan untuk mengabaikan kata-kata stop (stop_words="english")
count_vectorizer = CountVectorizer(stop_words="english")

# Melakukan transformasi menggunakan CountVectorizer pada teks yang dibaca
count_vector = count_vectorizer.fit_transform([corpus])

# Mendapatkan daftar fitur (kata-kata) dari hasil CountVectorizer
count_feature_names = count_vectorizer.get_feature_names_out()

# Inisialisasi TF-IDF Vectorizer dengan pengaturan untuk mengabaikan kata-kata stop (stop_words="english")
tfidf_vectorizer = TfidfVectorizer(stop_words="english")

# Melakukan transformasi menggunakan TF-IDF Vectorizer pada teks yang dibaca
tfidf_vector = tfidf_vectorizer.fit_transform([corpus])

# Mendapatkan daftar fitur (kata-kata) dari hasil TF-IDF Vectorizer
tfidf_feature_names = tfidf_vectorizer.get_feature_names_out()

# Hasil CountVectorizer
print("Hasil CountVectorizer:")
print(count_feature_names)

# Hasil TF-IDF Vectorizer
print("\nHasil TF-IDF Vectorizer:")
print(tfidf_feature_names)
```

Berikut output yang dihasilkan.

```
Hasil CountVectorizer:
['000' '10' '100' ... 'þéow' 'þíne' 'þú']

Hasil TF-IDF Vectorizer:
['000' '10' '100' ... 'þéow' 'þíne' 'þú']
```

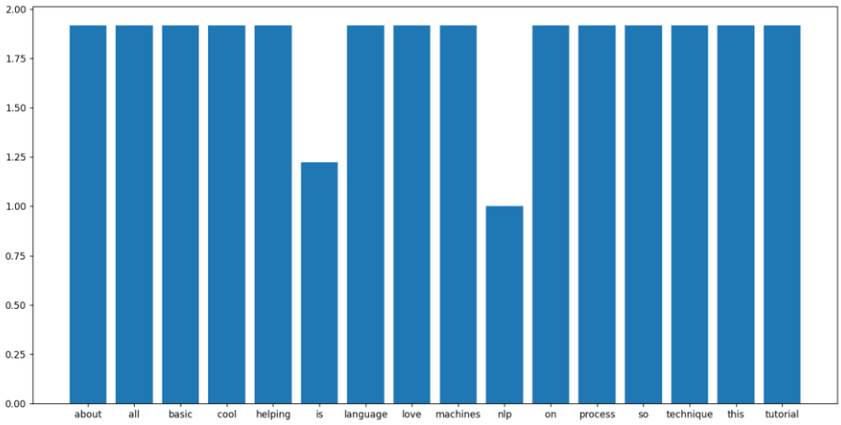
Output yang dapat terlihat adalah daftar kata atau fitur yang telah diekstrak dari korpus teks setelah menerapkan CountVectorizer dan TF-IDF Vectorizer. Di sini, setiap kata atau fitur dalam teks diperlakukan sebagai kolom dalam matriks yang dihasilkan oleh CountVectorizer dan TF-IDF Vectorizer.

- Hasil CountVectorizer: Daftar ini berisi kata-kata unik yang ditemukan dalam korpus setelah menerapkan CountVectorizer. CountVectorizer menghitung berapa kali setiap kata muncul dalam dokumen dan kemudian membuat matriks di mana setiap baris mewakili dokumen, dan setiap kolom mewakili kata. Nilai dalam sel matriks adalah jumlah kemunculan kata dalam dokumen. Jadi, daftar ini berisi semua kata unik yang ditemukan dalam dokumen, termasuk kata-kata umum seperti "the," "and," dan lainnya.
- Hasil TF-IDF Vectorizer: Daftar ini adalah daftar kata unik yang ditemukan dalam korpus setelah menerapkan TF-IDF Vectorizer. TF-IDF Vectorizer mengukur pentingnya setiap kata dalam dokumen dengan mempertimbangkan seberapa sering kata tersebut muncul dalam dokumen dan seberapa umum kata tersebut dalam seluruh korpus. Daftar ini mencakup kata-kata yang memiliki nilai TF-IDF yang lebih tinggi, yang berarti kata-kata ini dianggap lebih penting atau lebih unik dalam konteks dokumen tersebut.

BAB : DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE : SATU (1)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 14 SEPTEMBER 2023

Jadi, perbedaan antara keduanya adalah dalam cara penghitungan pentingnya kata-kata dalam dokumen. Hasil dari CountVectorizer mencakup semua kata dalam dokumen, sementara hasil dari TF-IDF Vectorizer mencakup kata-kata yang memiliki tingkat penting yang lebih tinggi dalam konteks dokumen tersebut. Dalam banyak kasus, hasil dari TF-IDF Vectorizer lebih berguna untuk analisis teks karena mereka memberikan penekanan pada kata-kata yang lebih informatif.

- **Modifikasi kode bagian (d) agar bisa menampilkan grafik dari tiap kata. contohnya seperti gambar di bawah.**



Untuk memodifikasi kode bagian (d) agar bisa menampilkan grafik dari tiap kata dapat menggunakan kode di bawah ini:

```
# Import library yang diperlukan
from sklearn.feature_extraction.text import TfidfVectorizer
import matplotlib.pyplot as plt

# Menampilkan judul
print("\nTF-IDF")

# Membuat daftar teks
text1 = ['i love nlp', "nlp is so cool",
"nlp is all about helping machines process language",
"this tutorial is on basic nlp technique"]

# Membuat objek TfidfVectorizer
tf = TfidfVectorizer()

# Menyesuaikan (fit) teks dengan objek TfidfVectorizer
txt_fitted = tf.fit(text1)

# Mengubah teks menjadi representasi TF-IDF
txt_transformed = txt_fitted.transform(text1)

# Menghitung nilai IDF untuk setiap kata
idf = dict(zip(txt_fitted.get_feature_names_out(), txt_fitted.idf_))

# Menampilkan grafik
plt.figure(figsize=(10, 6))
plt.bar(idf.keys(), idf.values())
plt.xlabel('Kata')
plt.ylabel('Nilai TF-IDF')
plt.title('TF-IDF untuk Setiap Kata')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

PRAKTIKUM DATA MINING II

BAB	: DATA ACQUISITION, REPRESENTATION, AND STORAGE
PRAKTIKUM KE	: SATU (1)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 14 SEPTEMBER 2023

Berikut output yang dihasilkan: