

**TUGAS 6: MEMBANGUN FITUR DAN
MELAKUKAN KALSIFIKASI DENGAN METODE
SVM DAN KNN**

disusun untuk memenuhi tugas
Teks dan Web Mining

Oleh:

**SITI NURRAHMASITA
(2108107010015)**



**PROGRAM STUDI INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS SYIAH KUALA
DARUSSALAM, BANDA ACEH
2024**

A. Deskripsi Tugas

Menggunakan sekumpulan halaman web berkategori POSITIF (+) dan halaman web berkategori NEGATIVE (-) yang sesuai dengan kategori yang telah dikumpulkan pada tugas 4 sebelumnya, anda diminta untuk mengumpulkan minimal 8000 halaman web gabungan dari dua kategori tersebut. Jadi lebih kurang 3000 untuk masing-masing kategori (2 kategori) telah digunakan untuk membangun kamus. Lakukan preproses untuk ke 2000 file baru tersebut dan bangkitkan fitur untuk setiap halaman dari 6 bagian dalam halaman web, yaitu bagian title, bagian content (bagian konten ini dibagi menjadi 5 sub bagian). Pelajari bahan kuliah yang membahas tentang pembangkitan fitur untuk membantu penyelesaian tugas ini. Gunakan kamus POSITIF dan NEGATIF yang telah anda bangkitkan pada tugas sebelumnya. Fitur disusun dalam format ARFF dan format SVM Light atau format lain untuk menjalankan SVM (boleh menggunakan scikit-learn).

Gunakan 80% dari 2000 data yang telah dibangkitkan fiturnya untuk training set dan 20% sisanya untuk menjadi testing set. Gunakan metode SVM dan KNN untuk melakukan klasifikasi dan tentukan kinerjanya.

Buat laporan yang menjelaskan hasil yang diperoleh. Format laporan dapat mengikuti format laporan tugas sebelumnya. File Laporan Tugas 6 dalam format PDF dikumpulkan paling telat tanggal 13 Mei 2024 NAMA_NIM.tar.gz.

B. Pendahuluan

Proses pengolahan dan analisis data memiliki peran yang semakin penting dalam mendukung pengambilan keputusan yang tepat dan efisien di berbagai bidang. Dalam rangka menjawab tantangan tersebut, tugas ini bertujuan untuk mengimplementasikan serangkaian langkah dari preprocessing hingga pembuatan fitur dan klasifikasi pada data teks. Melalui langkah-langkah ini, diharapkan dapat diperoleh informasi yang lebih terstruktur dan dapat dipahami dengan baik, sehingga mendukung pengambilan keputusan yang lebih cerdas.

a. Langkah-langkah Pengolahan Data

Langkah awal dalam proses ini adalah melakukan preprocessing terhadap teks yang telah diambil dari sumbernya. Preprocessing dilakukan untuk membersihkan teks dari karakter-karakter yang tidak relevan dan mengubahnya ke dalam format yang lebih sesuai untuk analisis selanjutnya. Setelah itu, dilakukan penghapusan duplikasi kata-kata berdasarkan observasi eliminasi rasio tertentu. Hal ini bertujuan untuk memastikan bahwa data yang digunakan dalam analisis merupakan representasi yang akurat dari informasi yang ada.

b. Pembuatan Fitur dan Klasifikasi

Selanjutnya, dilakukan pembuatan fitur-fitur yang merepresentasikan informasi dari teks. Fitur-fitur ini dihasilkan melalui penghitungan frekuensi kemunculan kata-kata dalam teks yang telah diproses sebelumnya. Fitur-fitur tersebut kemudian digunakan dalam proses klasifikasi menggunakan algoritma SVM (Support Vector Machine) dan KNN (K-Nearest Neighbors). Dengan demikian, diharapkan dapat diperoleh model klasifikasi yang mampu mengidentifikasi dan mengklasifikasikan data dengan akurasi yang tinggi.

C. Penjelasan Umum

a. Remove-One-Gram.py, Remove-Two-Gram.py, Remove-Three-Gram.py:

Ketiga file tersebut bertanggung jawab untuk menghapus duplikasi kata-kata dari kamus-kamus yang telah dihasilkan sebelumnya. Proses penghapusan duplikasi dilakukan berdasarkan observasi eliminasi rasio, di mana kata-kata yang memiliki frekuensi kemunculan yang rendah dibandingkan dengan kata-kata yang serupa akan dihapus. Ini bertujuan untuk memastikan bahwa kamus-kamus yang digunakan dalam analisis hanya mengandung kata-kata yang relevan dan memiliki frekuensi kemunculan yang cukup signifikan.

b. Create-Feature.py:

File ini bertanggung jawab untuk menghasilkan fitur-fitur yang akan digunakan dalam proses klasifikasi data. Fitur-fitur ini dihasilkan dari analisis teks yang telah diproses sebelumnya, seperti judul dan konten artikel. Proses pembuatan fitur melibatkan perhitungan frekuensi kemunculan kata-kata dalam teks, dengan mempertimbangkan bagian-bagian tertentu dari teks, seperti bagian judul, bagian atas, tengah, dan bawah dari konten artikel. Fitur-fitur ini kemudian akan digunakan sebagai masukan untuk algoritma klasifikasi.

c. Classification.py:

File ini berisi implementasi dari dua algoritma klasifikasi, yaitu Support Vector Machine (SVM) dan K-Nearest Neighbors (KNN). Algoritma SVM dan KNN digunakan untuk melakukan klasifikasi terhadap data berdasarkan fitur-fitur yang telah dihasilkan sebelumnya. Tujuan dari proses klasifikasi ini adalah untuk mengklasifikasikan data ke dalam kategori yang tepat berdasarkan informasi yang terkandung dalam teksnya. Akurasi dari masing-masing model klasifikasi juga dihitung untuk mengevaluasi performanya dalam mengklasifikasikan data.

D. Penjelasan Scripts

a. Remove-one-gram.py, remove-two-gram.py, dan remove-three-gram.py

```
import os

# Fungsi untuk membaca kamus dari file dengan format frekuensi,kata1
def read_dictionary(file_path):
    dictionary = {}
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            line = line.strip()
            if line:
                # Ambil bagian frekuensi dan kata
                frequency = line.split()[-1] # Ambil data di bagian akhir
                word = line[:-len(frequency)].strip() # Ambil sisa sebagai kata
                dictionary[word] = float(frequency)
            else:
                print("Error: Empty line found.")
    return dictionary

# Fungsi untuk menghapus duplikasi kata berdasarkan observasi eliminasi rasio
def remove_duplicates(dictionary_A, dictionary_B, threshold):
    unique_A = {}
    unique_B = {}

    for word, freq_A in dictionary_A.items():
        freq_B = dictionary_B.get(word, 0)

        # Hitung nilai rasio
        max_freq = max(freq_A, freq_B)
        min_freq = min(freq_A, freq_B)
        ratio = min_freq / max_freq if max_freq != 0 else 0

        # Hapus kata duplikat berdasarkan threshold
        if ratio < threshold:
            if freq_A >= freq_B:
                unique_A[word] = freq_A
            else:
                unique_B[word] = freq_B

    return unique_A, unique_B

# Fungsi untuk menyimpan kamus ke file
def save_dictionary(dictionary, output_file):
    with open(output_file, 'w', encoding='utf-8') as file:
        # Urutkan kamus berdasarkan frekuensi dari yang terbesar ke terkecil
        sorted_dict = sorted(dictionary.items(), key=lambda x: x[1], reverse=True)
        for word, frequency in sorted_dict:
            if frequency.is_integer():
                file.write(f"{int(frequency)}\t{word}\n")
            else:
                file.write(f"{frequency:.1f}\t{word}\n")

# Path ke kamus berita dan sepakbola dengan format frekuensi,kata1
path_news = 'kamus/one-gram/berita_txts.txt'
path_football = 'kamus/one-gram/sepakbola_txts.txt'

# Baca kamus berita dan sepakbola
dict_news = read_dictionary(path_news)
dict_football = read_dictionary(path_football)

# Threshold untuk observasi eliminasi rasio
threshold_1 = 0.50
threshold_2 = 0.55

# Hapus duplikasi kata berdasarkan threshold
unique_news_1, unique_football_1 = remove_duplicates(dict_news, dict_football, threshold_1)
unique_news_2, unique_football_2 = remove_duplicates(dict_news, dict_football, threshold_2)

# Simpan kamus yang telah dihapus duplikasinya
output_news_1 = 'kamus_distinct/berita/one-gram_unique_50.txt'
output_football_1 = 'kamus_distinct/sepakbola/one-gram_unique_50.txt'
output_news_2 = 'kamus_distinct/berita/one-gram_unique_55.txt'
output_football_2 = 'kamus_distinct/sepakbola/one-gram_unique_55.txt'

save_dictionary(unique_news_1, output_news_1)
save_dictionary(unique_football_1, output_football_1)
save_dictionary(unique_news_2, output_news_2)
save_dictionary(unique_football_2, output_football_2)

print("Penghapusan duplikasi selesai.")
```

Gambar D.a.1. Scripts remove-one-gram.py

```

import os

# Fungsi untuk membaca kamus dari file dengan format frekuensi,kata
def read_dictionary(file_path):
    dictionary = {}
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            line = line.strip()
            if line:
                # Ambil bagian frekuensi dan kata
                frequency = line.split()[-1] # Ambil data di bagian akhir
                word = line[:-len(frequency)].strip() # Ambil sisa sebagai kata
                dictionary[word] = float(frequency)
            else:
                print("Error: Empty line found.")
    return dictionary

# Fungsi untuk menghapus duplikasi kata berdasarkan observasi eliminasi rasio
def remove_duplicates(dictionary_A, dictionary_B, threshold):
    unique_A = {}
    unique_B = {}

    for word, freq_A in dictionary_A.items():
        freq_B = dictionary_B.get(word, 0)

        # Hitung nilai rasio
        max_freq = max(freq_A, freq_B)
        min_freq = min(freq_A, freq_B)
        ratio = min_freq / max_freq if max_freq != 0 else 0

        # Hapus kata duplikat berdasarkan threshold
        if ratio < threshold:
            if freq_A >= freq_B:
                unique_A[word] = freq_A
            else:
                unique_B[word] = freq_B

    return unique_A, unique_B

# Fungsi untuk menyimpan kamus ke file
def save_dictionary(dictionary, output_file):
    with open(output_file, 'w', encoding='utf-8') as file:
        # Urutkan kamus berdasarkan frekuensi dari yang terbesar ke terkecil
        sorted_dict = sorted(dictionary.items(), key=lambda x: x[1], reverse=True)
        for word, frequency in sorted_dict:
            freq_str = str(int(frequency)) if frequency.is_integer() else f"{frequency:.1f}"
            file.write(f"{freq_str}\t{word}\n")

# Path ke kamus berita dan sepakbola
path_news = 'kamus/two-gram/berita_txts.txt'
path_football = 'kamus/two-gram/sepakbola_txts.txt'

# Baca kamus berita dan sepakbola
dict_news = read_dictionary(path_news)
dict_football = read_dictionary(path_football)

# Threshold untuk observasi eliminasi rasio
threshold_1 = 0.50
threshold_2 = 0.55

# Hapus duplikasi kata berdasarkan threshold
unique_news_1, unique_football_1 = remove_duplicates(dict_news, dict_football, threshold_1)
unique_news_2, unique_football_2 = remove_duplicates(dict_news, dict_football, threshold_2)

# Simpan kamus yang telah dihapus duplikasinya
output_news_1 = 'kamus_distinct/berita/two-gram_unique_50.txt'
output_football_1 = 'kamus_distinct/sepakbola/two-gram_unique_50.txt'
output_news_2 = 'kamus_distinct/berita/two-gram_unique_55.txt'
output_football_2 = 'kamus_distinct/sepakbola/two-gram_unique_55.txt'

save_dictionary(unique_news_1, output_news_1)
save_dictionary(unique_football_1, output_football_1)
save_dictionary(unique_news_2, output_news_2)
save_dictionary(unique_football_2, output_football_2)

print("Penghapusan duplikasi selesai.")

```

Gambar D.a.2. Scripts remove-two-gram.py

```

import os

# Fungsi untuk membaca kamus dari file dengan format frekuensi,kata
def read_dictionary(file_path):
    dictionary = {}
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            line = line.strip()
            if line:
                # Ambil bagian frekuensi dan kata
                frequency = line.split()[-1] # Ambil data di bagian akhir
                word = line[:-len(frequency)].strip() # Ambil sisa sebagai kata
                dictionary[word] = float(frequency)
            else:
                print("Error: Empty line found.")
    return dictionary

# Fungsi untuk menghapus duplikasi kata berdasarkan observasi eliminasi rasio
def remove_duplicates(dictionary_A, dictionary_B, threshold):
    unique_A = {}
    unique_B = {}

    for word, freq_A in dictionary_A.items():
        freq_B = dictionary_B.get(word, 0)

        # Hitung nilai rasio
        max_freq = max(freq_A, freq_B)
        min_freq = min(freq_A, freq_B)
        ratio = min_freq / max_freq if max_freq != 0 else 0

        # Hapus kata duplikat berdasarkan threshold
        if ratio < threshold:
            if freq_A >= freq_B:
                unique_A[word] = freq_A
            else:
                unique_B[word] = freq_B

    return unique_A, unique_B

# Fungsi untuk menyimpan kamus ke file
def save_dictionary(dictionary, output_file):
    with open(output_file, 'w', encoding='utf-8') as file:
        # Urutkan kamus berdasarkan frekuensi dari yang terbesar ke terkecil
        sorted_dict = sorted(dictionary.items(), key=lambda x: x[1], reverse=True)
        for word, frequency in sorted_dict:
            freq_str = str(int(frequency)) if frequency.is_integer() else f"{frequency:.1f}"
            file.write(f"{freq_str}\t{word}\n")

# Path ke kamus berita dan sepakbola
path_news = 'kamus/three-gram/berita_txts.txt'
path_football = 'kamus/three-gram/sepakbola_txts.txt'

# Baca kamus berita dan sepakbola
dict_news = read_dictionary(path_news)
dict_football = read_dictionary(path_football)

# Threshold untuk observasi eliminasi rasio
threshold_1 = 0.50
threshold_2 = 0.55

# Hapus duplikasi kata berdasarkan threshold
unique_news_1, unique_football_1 = remove_duplicates(dict_news, dict_football, threshold_1)
unique_news_2, unique_football_2 = remove_duplicates(dict_news, dict_football, threshold_2)

# Simpan kamus yang telah dihapus duplikasinya
output_news_1 = 'kamus_distinct/berita/three-gram_unique_50.txt'
output_football_1 = 'kamus_distinct/sepakbola/three-gram_unique_50.txt'
output_news_2 = 'kamus_distinct/berita/three-gram_unique_55.txt'
output_football_2 = 'kamus_distinct/sepakbola/three-gram_unique_55.txt'

save_dictionary(unique_news_1, output_news_1)
save_dictionary(unique_football_1, output_football_1)
save_dictionary(unique_news_2, output_news_2)
save_dictionary(unique_football_2, output_football_2)

print("Penghapusan duplikasi selesai.")

```

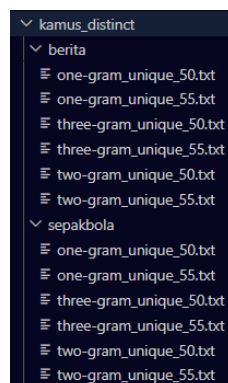
Gambar D.a.3. Scripts remove-three-gram.py

File Remove-One-Gram.py, Remove-Two-Gram.py, dan Remove-Three-Gram.py, memiliki tujuan yang serupa, yaitu menghapus duplikasi kata-kata dari kamus-kamus yang telah dibuat sebelumnya, namun dengan tingkat n-gram berbeda. Berikut adalah tahapan-tahapannya:

1. Pada tahap awal, masing-masing skrip membaca kamus yang berisi daftar kata dan frekuensi kemunculannya dari file teks. Fungsi 'read_dictionary(file_path)' digunakan untuk ini. Fungsi ini membuka file, membaca setiap baris, dan memisahkan kata dari frekuensinya untuk dimasukkan ke dalam sebuah dictionary.
2. Setelah membaca kamus-kamus, dilakukan proses penghapusan duplikasi kata-kata. Hal ini dilakukan dengan membandingkan frekuensi kemunculan kata-kata antara kamus satu dengan kamus lainnya. Fungsi 'remove_duplicates(dictionary_A, dictionary_B, threshold)' melakukan ini dengan membandingkan frekuensi kemunculan kata yang sama di kedua kamus, maka kata tersebut dianggap sebagai duplikasi dan akan dihapus.
3. Proses penghapusan duplikasi didasarkan pada observasi eliminasi rasio. Rasio ini merupakan perbandingan antara frekuensi kemunculan kata yang akan dihapus dengan frekuensi kemunculan kata yang serupa di kamus lain. Jika rasio ini lebih kecil dari threshold tertentu, maka kata tersebut dihapus.
4. Setelah proses penghapusan duplikasi, kamus yang telah dibersihkan disimpan kembali ke file teks. Fungsi 'save_dictionary(dictionary, output_file)' digunakan untuk menyimpan dictionary ke dalam file. Fungsi ini juga mengurutkan kata-kata berdasarkan frekuensinya dari yang terbesar hingga terkecil sebelum menyimpannya. Kamus akan ditampilkan sebagai output dengan format frekuensi dan data.

```
PS C:\Users\USER DK\OneDrive\Documents\sem 6\coba> py remove-one-gram.py
Penghapusan duplikasi selesai.
PS C:\Users\USER DK\OneDrive\Documents\sem 6\coba> py remove-two-gram.py
Penghapusan duplikasi selesai.
PS C:\Users\USER DK\OneDrive\Documents\sem 6\coba> py remove-three-gram.py
Penghapusan duplikasi selesai.
```

Gambar D.a.4. Output sukses membangun kamus one-gram, two-gram, dan three-gram



Gambar D.a.5. Directory yang dihasilkan

Output yang dihasilkan:

1. Remove-one-gram.py

kamus_distinct > berita > one-gram_unique_50.txt	kamus_distinct > berita > one-gram_unique_50.txt
1 7483 ada	41010 1 intimasjid
2 5380 tersebut	41011 1 tautau
3 3940 hari	41012 1 1489
4 3305 atau	41013 1 sgr
5 3297 jalan	41014 1 rung
6 3096 warga	41015 1 umpetan
7 3061 korban	41016 1 kabare
8 3051 orang	41017 1 miftakhurrohman
9 3035 2024	41018 1 ngawal
10 2945 kota	41019 1 190an
11 2560 telah	41020 1 kasda
12 2462 rumah	41021 1 bapas
13 2329 kabupaten	41022 1 desal
14 2318 puasa	41023 1 apbdes
15 2289 terjadi	41024 1 nsyaallah
16 2146 mengatakan	41025 1 keribo
17 2129 wib	41026 1 inalilahi
18 1935 sekitar	41027 1 wainilahi
19 1902 kemudian	41028 1 prayogha
20 1879 allah	41029 1 cirendue
21 1839 bulan	41030 1 tanggerang
22 1812 pukul	41031 1 allahuma
23 1793 dapat	41032 1 firlahu

Gambar D.a.6. Hasil kamus one-gram_unique_50.txt kategori berita

kamus_distinct > berita > one-gram_unique_55.txt	kamus_distinct > berita > one-gram_unique_55.txt
1 7483 ada	42137 1 intimasjid
2 5380 tersebut	42138 1 tautau
3 5074 kita	42139 1 1489
4 3940 hari	42140 1 sgr
5 3859 kata	42141 1 rung
6 3305 atau	42142 1 umpetan
7 3297 jalan	42143 1 kabare
8 3217 oleh	42144 1 miftakhurrohman
9 3096 warga	42145 1 ngawal
10 3061 korban	42146 1 190an
11 3057 kepada	42147 1 kasda
12 3051 orang	42148 1 bapas
13 3035 2024	42149 1 desal
14 2945 kota	42150 1 apbdes
15 2650 bandung	42151 1 nsyaallah
16 2560 telah	42152 1 keribo
17 2462 rumah	42153 1 inalilahi
18 2454 hingga	42154 1 wainilahi
19 2329 kabupaten	42155 1 prayogha
20 2318 puasa	42156 1 cirendue
21 2289 terjadi	42157 1 tanggerang
22 2146 mengatakan	42158 1 allahuma
23 2129 wib	42159 1 firlahu

Gambar D.a.7. Hasil kamus one-gram_unique_55.txt kategori berita

kamus_distinct > sepakbola > one-gram_unique_50.txt	kamus_distinct > sepakbola > one-gram_unique_50.txt
1 6067 pemain	2706 3 manut
2 5020 laga	2707 3 muliakan
3 4325 indonesia	2708 3 keluyuran
4 3819 tim	2709 3 308
5 3769 liga	2710 3 menyontek
6 3474 gol	2711 3 professional
7 3220 pertandingan	2712 3 memories
8 2846 piala	2713 3 ed
9 2815 timnas	2714 3 kun
10 2651 stadion	2715 3 percepat
11 2635 sepakbola	2716 3 bertutur
12 2619 persib	2717 3 pelampiasan
13 2328 pelatih	2718 3 pilihlah
14 1978 bermain	2719 3 bai
15 1945 babak	2720 3 lose
16 1941 musim	2721 3 disembuhkan
17 1926 menit	2722 3 digit
18 1884 dunia	2723 3 sandungan
19 1840 poin	2724 3 fabian
20 1765 kedua	2725 3 mengucur
21 1697 2023	2726 3 mencakmencak
22 1682 pss	2727 3 20182020
23 1664 pertama	2728 3 sekecamatan

Gambar D.a.8. Hasil kamus one-gram_unique_50.txt kategori sepakbola

kamus_distinct > sepakbola > one-gram_unique_55.txt			kamus_distinct > sepakbola > one-gram_unique_55.txt		
1	6067	pemain	3360	2	menubruk
2	5020	laga	3361	2	duke
3	4325	indonesia	3362	2	mumet
4	3819	tim	3363	2	refresh
5	3769	liga	3364	2	berikrar
6	3474	gol	3365	2	ortopedi
7	3220	pertandingan	3366	2	curahan
8	2846	piala	3367	2	padepokan
9	2815	timnas	3368	2	mission
10	2651	stadion	3369	2	riwayatnya
11	2635	sepakbola	3370	2	mengadvokasi
12	2619	persib	3371	2	reaksinya
13	2328	pelatih	3372	2	perbaikanperbaikan
14	1978	bermain	3373	2	meledakkan
15	1945	babak	3374	2	stephen
16	1941	musim	3375	2	upload
17	1926	menit	3376	2	menggelayut
18	1884	dunia	3377	2	rintik
19	1840	poin	3378	2	suhendar
20	1765	kedua	3379	2	scientists
21	1697	2023	3380	2	kondom
22	1682	pss	3381	2	loloskan
23	1664	pertama	3382	2	standup

Gambar D.a.9. Hasil kamus one-gram_unique_55.txt kategori sepakbola

2. Remove-two-gram.py

kamus_distinct > berita > two-gram_unique_50.txt			kamus_distinct > berita > two-gram_unique_50.txt		
1	1164	hari ini	385715	1	saat tubuh
2	900	tidak ada	385716	1	tubuh berhenti
3	839	kota bandung	385717	1	berhenti memproduksi
4	807 00	wib	385718	1	memproduksi cukup
5	745	di jalan	385719	1	cukup sel
6	713	jawa barat	385720	1	sel darah
7	692	buka puasa	385721	1	darah baru
8	654	idul fitri	385722	1	seseorang lebih
9	632	ada yang	385723	1	terhadap infeksi
10	626	lalu lintas	385724	1	infeksi serta
11	617	di lokasi	385725	1	serta pendarahan
12	615	orang yang	385726	1	dari mayo
13	605	maret 2024	385727	1	clinic kondisi
14	591	terjadi di	385728	1	termasuk langka
15	561	yang telah	385729	1	berkembang pada
16	560	april 2024	385730	1	aplastik bisa
17	534	di wilayah	385731	1	tibatiba atau
18	526	sekitar pukul	385732	1	perlahan baik
19	505	yang tidak	385733	1	ringan atau
20	500	dari arah	385734	1	berat sampai
21	498	allah swt	385735	1	akhirnya memburuk
22	471	yang ada	385736	1	tasikmalaya 25
23	464	di dalam	385737	1	jadwalnya umat

Gambar D.a.10. Hasil kamus two-gram_unique_50.txt kategori berita

kamus_distinct > berita > two-gram_unique_55.txt			kamus_distinct > berita > two-gram_unique_55.txt		
1	1164	hari ini	392266	1	saat tubuh
2	900	tidak ada	392267	1	tubuh berhenti
3	839	kota bandung	392268	1	berhenti memproduksi
4	807 00	wib	392269	1	memproduksi cukup
5	745	di jalan	392270	1	cukup sel
6	713	jawa barat	392271	1	sel darah
7	692	buka puasa	392272	1	darah baru
8	654	idul fitri	392273	1	seseorang lebih
9	632	ada yang	392274	1	terhadap infeksi
10	626	lalu lintas	392275	1	infeksi serta
11	617	di lokasi	392276	1	serta pendarahan
12	615	orang yang	392277	1	dari mayo
13	605	maret 2024	392278	1	clinic kondisi
14	591	terjadi di	392279	1	termasuk langka
15	561	yang telah	392280	1	berkembang pada
16	560	april 2024	392281	1	aplastik bisa
17	534	di wilayah	392282	1	tibatiba atau
18	526	sekitar pukul	392283	1	perlahan baik
19	505	yang tidak	392284	1	ringan atau
20	500	dari arah	392285	1	berat sampai
21	498	allah swt	392286	1	akhirnya memburuk
22	471	yang ada	392287	1	tasikmalaya 25
23	464	di dalam	392288	1	jadwalnya umat

Gambar D.a.11. Hasil kamus two-gram_unique_55.txt kategori berita

kamus_distinct > sepakbola > ≡ two-gram_unique_50.txt		kamus_distinct > sepakbola > ≡ two-gram_unique_50.txt	
1 1544 di stadion		7870 3 sejak era	
2 1308 liga 1		7871 3 tidak terkejut	
3 1288 piala dunia		7872 3 penting daripada	
4 1066 timnas indonesia		7873 3 memiliki opsi	
5 988 piala asia		7874 3 dan ceo	
6 803 di laga		7875 3 pernah mengungkapkan	
7 788 di liga		7876 3 akan berperan	
8 765 pss sleman		7877 3 telah menganalisis	
9 722 di babak		7878 3 bisa berfoto	
10 681 musim ini		7879 3 kok ujar	
11 609 1 20232024		7880 3 tak bikin	
12 606 pemain yang		7881 3 kini semua	
13 586 para pemain		7882 3 dirinya mau	
14 571 dengan skor		7883 3 rp 95	
15 544 liga 2		7884 3 ikut di	
16 533 persib bandung		7885 3 disayangkan oleh	
17 532 di piala		7886 3 sebagai pahlawan	
18 528 mencetak gol		7887 3 mereka kami	
19 508 shin taeyong		7888 3 tidak membantu	
20 490 di posisi		7889 3 untuk diganti	
21 485 tuan rumah		7890 3 gaji mereka	
22 466 persis solo		7891 3 aspirasi mereka	
23 452 lolos ke		7892 3 ayah kami	

Gambar D.a.12. Hasil kamus two-gram_unique_50.txt kategori sepakbola

kamus_distinct > sepakbola > ≡ two-gram_unique_55.txt		kamus_distinct > sepakbola > ≡ two-gram_unique_55.txt	
1 1544 di stadion		12798 2 burung dan	
2 1308 liga 1		12799 2 lihat kita	
3 1288 piala dunia		12800 2 sampai dewasa	
4 1066 timnas indonesia		12801 2 kebetulan berada	
5 988 piala asia		12802 2 hadiah itu	
6 803 di laga		12803 2 itu balik	
7 788 di liga		12804 2 prestasi para	
8 765 pss sleman		12805 2 mereka kita	
9 722 di babak		12806 2 meninggalkan motornya	
10 681 musim ini		12807 2 bantul untuk	
11 609 1 20232024		12808 2 oleh pemangku	
12 606 pemain yang		12809 2 ikut main	
13 586 para pemain		12810 2 yang memunculkan	
14 571 dengan skor		12811 2 abad ke10	
15 544 liga 2		12812 2 ke12 di	
16 533 persib bandung		12813 2 wib karena	
17 532 di piala		12814 2 persidangan dan	
18 528 mencetak gol		12815 2 persidangan dengan	
19 508 shin taeyong		12816 2 masa persiapan	
20 490 di posisi		12817 2 2020 dengan	
21 485 tuan rumah		12818 2 proyek apa	
22 466 persis solo		12819 2 ada 44	
23 452 lolos ke		12820 2 serius yang	

Gambar D.a.13. Hasil kamus two-gram_unique_55.txt kategori sepakbola

3. Remove-three-gram.py

kamus_distinct > berita > ≡ three-gram_unique_50.txt		kamus_distinct > berita > ≡ three-gram_unique_50.txt	
1 343 jadwal buka puasa		811851 1 aplastik bisa terjadi	
2 313 yang ada di		811852 1 terjadi secara tibatiba	
3 240 di media sosial		811853 1 secara tibatiba atau	
4 209 arus lalu lintas		811854 1 tibatiba atau bisa	
5 206 selengkapnya di sini		811855 1 bisa juga terjadi	
6 205 yang berada di		811856 1 juga terjadi secara	
7 168 di bulan ramadhan		811857 1 terjadi secara perlahan	
8 165 di kota bandung		811858 1 secara perlahan baik	
9 150 kabupaten bandung barat		811859 1 perlahan baik ringan	
10 147 saat ditemui di		811860 1 baik ringan atau	
11 145 ke rumah sakit		811861 1 ringan atau berat	
12 144 gibran rakabuming raka		811862 1 atau berat sampai	
13 138 di lokasi kejadian		811863 1 berat sampai akhirnya	
14 137 hari raya idul		811864 1 sampai akhirnya memburuk	
15 137 raya idul fitri		811865 1 akhirnya memburuk seiring	
16 137 أكبر الله أكبر		811866 1 memburuk seiring berjalannya	
17 131 saat ini masih		811867 1 waktu imsak garut	
18 129 di jawa barat		811868 1 dan tasikmalaya 25	
19 128 artikel ini ditulis		811869 1 tasikmalaya 25 maret	
20 126 dan wakil presiden		811870 1 2024 ini jadwalnya	
21 126 ini ditulis oleh		811871 1 ini jadwalnya umat	
22 126 kampus merdeka di		811872 1 jadwalnya umat islam	
23 126 merdeka di detikcom		811873 1 tasikmalaya hari senin	

Gambar D.a.14. Hasil kamus three-gram_unique_50.txt kategori berita

kamus_distinct > berita > ≡ three-gram_unique_55.txt	kamus_distinct > berita > ≡ three-gram_unique_55.txt
1 343 jadwal buka puasa	815185 1 aplastik bisa terjadi
2 313 yang ada di	815186 1 terjadi secara tibatiba
3 240 di media sosial	815187 1 secara tibatiba atau
4 209 arus lalu lintas	815188 1 tibatiba atau bisa
5 206 selengkapnya di sini	815189 1 bisa juga terjadi
6 205 yang berada di	815190 1 juga terjadi secara
7 168 di bulan ramadhan	815191 1 terjadi secara perlahan
8 165 di kota bandung	815192 1 secara perlahan baik
9 150 kabupaten bandung barat	815193 1 perlahan baik ringan
10 147 saat ditemui di	815194 1 baik ringan atau
11 145 ke rumah sakit	815195 1 ringan atau berat
12 144 gibran rakabuming raka	815196 1 atau berat sampai
13 138 di lokasi kejadian	815197 1 berat sampai akhirnya
14 137 hari raya idul	815198 1 sampai akhirnya memburuk
15 137 raya idul fitri	815199 1 akhirnya memburuk seiring
16 137 أكبر الله أكبر	815200 1 memburuk seiring berjalannya
17 131 saat ini masih	815201 1 waktu imsak garut
18 129 di jawa barat	815202 1 dan tasikmalaya 25
19 128 artikel ini ditulis	815203 1 tasikmalaya 25 maret
20 126 dan wakil presiden	815204 1 2024 ini jadwalnya
21 126 ini ditulis oleh	815205 1 ini jadwalnya umat
22 126 kampus merdeka di	815206 1 jadwalnya umat islam
23 126 merdeka di detikcom	815207 1 tasikmalaya hari senin

Gambar D.a.15. Hasil kamus three-gram_unique_55.txt kategori berita

kamus_distinct > sepakbola > ≡ three-gram_unique_50.txt	kamus_distinct > sepakbola > ≡ three-gram_unique_50.txt
1 608 liga 1 20232024	2533 3 bukan hanya dari
2 375 piala asia u23	2534 3 nothing to lose
3 375 piala asia 2023	2535 3 pada senin 294
4 269 di piala asia	2536 3 dalam pernyataannya yang
5 260 di stadion manahan	2537 3 saya bisa melihat
6 240 dini hari wib	2538 3 yang sedang terjadi
7 230 stadion manahan solo	2539 3 situasi yang kurang
8 220 asia u23 2024	2540 3 kabag ops polresta
9 186 david da silva	2541 3 ia juga berterima
10 185 timnas indonesia u23	2542 3 merupakan kali kedua
11 151 kualifikasi piala dunia	2543 3 besar dan itu
12 149 di piala dunia	2544 3 sarung tangan dan
13 147 berlangsung di stadion	2545 3 tepatnya di rt
14 131 piala dunia 2026	2546 3 tidak bisa berbuat
15 120 dalam lanjutan liga	2547 3 hasil penelitian yang
16 118 lanjutan liga 1	2548 3 lebih penting daripada
17 115 berada di posisi	2549 3 keras lagi untuk
18 115 si jalak harupat	2550 3 oleh sejumlah pejabat
19 113 bandung lautan api	2551 3 selama 2 hari
20 111 akan digelar di	2552 3 pungkasnya seperti diketahui
21 110 di liga 2	2553 3 tidak pernah melihat
22 106 pukul 1900 wib	2554 3 halhal di luar
23 106 sesi jumpa pers	2555 3 dengan cara seperti

Gambar D.a.16. Hasil kamus three-gram_unique_50.txt kategori sepakbola

kamus_distinct > sepakbola > ≡ three-gram_unique_55.txt	kamus_distinct > sepakbola > ≡ three-gram_unique_55.txt
1 608 liga 1 20232024	5052 2 kami meminta agar
2 375 piala asia u23	5053 2 karena jika tidak
3 375 piala asia 2023	5054 2 jika tidak bisa
4 269 di piala asia	5055 2 saat ini posisi
5 260 di stadion manahan	5056 2 pernah mengungkapkan bahwa
6 240 dini hari wib	5057 2 oleh menteri dalam
7 230 stadion manahan solo	5058 2 dengan misi yang
8 220 asia u23 2024	5059 2 europa untuk pertama
9 186 david da silva	5060 2 bersama jokowi di
10 185 timnas indonesia u23	5061 2 jokowi akan mengunjungi
11 151 kualifikasi piala dunia	5062 2 ini beberapa kali
12 149 di piala dunia	5063 2 berjalan di bawah
13 147 berlangsung di stadion	5064 2 yang kebetulan berada
14 131 piala dunia 2026	5065 2 kebetulan berada di
15 120 dalam lanjutan liga	5066 2 taiwan dan korea
16 118 lanjutan liga 1	5067 2 di hari valentine
17 115 berada di posisi	5068 2 masa lalu dan
18 115 si jalak harupat	5069 2 pernah melihat orang
19 113 bandung lautan api	5070 2 pidana yang dilakukan
20 111 akan digelar di	5071 2 sudah lama di
21 110 di liga 2	5072 2 dia belum bisa
22 106 pukul 1900 wib	5073 2 kami bisa membawa
23 106 sesi jumpa pers	5074 2 sudah tiga bulan

Gambar D.a.17. Hasil kamus three-gram_unique_55.txt kategori sepakbola

b. Create_feature.py

```
import os
import re
import random
from collections import Counter

# Fungsi untuk membersihkan teks
def clean_text(text):
    cleaned_text = re.sub(r'^a-zA-Z\s', '', text)
    cleaned_text = cleaned_text.lower()
    return cleaned_text

# Fungsi untuk membaca daftar stopwords dari file eksternal
def read_stopwords(file_path):
    stopwords = set()
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            word = line.strip()
            if word:
                stopwords.add(word)
    return stopwords

# Fungsi untuk menghitung jumlah kata dalam teks (tidak termasuk stopwords)
def count_words(text, stopwords):
    words = text.split()
    filtered_words = [word for word in words if word not in stopwords]
    return len(filtered_words)

# Fungsi untuk menghitung fitur untuk bagian judul
def calculate_title_feature(title, dictionary, stopwords):
    title_words = clean_text(title).split()
    total_title_words = count_words(title, stopwords)
    word_counter = Counter(title_words)
    feature = sum(word_counter[word] for word in word_counter if word in dictionary)
    return feature / total_title_words if total_title_words > 0 else 0

# Fungsi untuk menghitung fitur untuk bagian konten (bagian atas, tengah, atau bawah)
def calculate_content_feature(content, dictionary, weight, stopwords):
    content_words = clean_text(content).split()
    total_content_words = count_words(content, stopwords)
    word_counter = Counter(content_words)
    feature = sum(word_counter[word] for word in word_counter if word in dictionary)
    return weight * feature / total_content_words if total_content_words > 0 else 0

# Fungsi untuk menghitung semua fitur untuk satu halaman web
def calculate_all_features(title, content_top, content_middle, content_bottom, dictionaries, weights, stopwords):
    features = []
    for category_dict, weight in zip(dictionaries, weights):
        category_features = []
        for dictionary in category_dict:
            title_feature = calculate_title_feature(title, dictionary, stopwords)
            top_feature = calculate_content_feature(content_top, dictionary, weight[0], stopwords)
            middle_feature = calculate_content_feature(content_middle, dictionary, weight[1], stopwords)
            bottom_feature = calculate_content_feature(content_bottom, dictionary, weight[2], stopwords)
            category_features.extend([title_feature, top_feature, middle_feature, bottom_feature])
        features.extend(category_features)
    return features

# Direktori tempat menyimpan file teks
directories = [r"content_splitting/berita", r"content_splitting/sepakbola"]

# Kamus-kamus untuk setiap kategori (1-kata, 2-kata, dan 3-kata)
positive_dictionaries = [
    [
        "kamus_distinct/berita/one-gram_unique_55.txt",
        "kamus_distinct/berita/two-gram_unique_55.txt",
        "kamus_distinct/berita/three-gram_unique_55.txt"
    ],
    [
        "kamus_distinct/sepakbola/one-gram_unique_55.txt",
        "kamus_distinct/sepakbola/two-gram_unique_55.txt",
        "kamus_distinct/sepakbola/three-gram_unique_55.txt"
    ]
]
```

```

# Bobot untuk bagian konten (bagian atas, tengah, dan bawah)
weights = [(0.5, 0.4, 0.3), (0.5, 0.4, 0.3)]

# Mendefinisikan path untuk file stopwords
stopwords_path = "stopword.txt"
# Membaca stopwords dari file eksternal
stopwords = read_stopwords(stopwords_path)

# Menyimpan data fitur dan label
data = {0: [], 1: []} # 0 untuk berita, 1 untuk sepak bola

# Proses setiap folder
for label, directory in enumerate(directories):
    # Mendapatkan daftar file dalam direktori
    txt_files = [file for file in os.listdir(directory) if file.endswith(".dat")]

    # Filter file yang memiliki urutan lebih dari 3000
    txt_files = [file for file in txt_files if int(re.findall(r'\d+', file)[0]) > 3000]

    # Proses setiap file teks
    for txt_file in txt_files:
        file_path = os.path.join(directory, txt_file)
        with open(file_path, "r", encoding="utf-8") as file:
            content = file.read()

            # Pisahkan teks menjadi bagian judul, atas, tengah, dan bawah (masing-masing 30%, 40%, 30%)
            content_words = content.split()
            total_content_words = len(content_words)
            top_end = int(total_content_words * 0.3)
            middle_start = int(total_content_words * 0.3)
            middle_end = int(total_content_words * 0.7)
            title = " ".join(content_words[:top_end])
            content_top = " ".join(content_words[:top_end])
            content_middle = " ".join(content_words[middle_start:middle_end])
            content_bottom = " ".join(content_words[middle_end:])

            # Menghitung fitur untuk setiap halaman web
            features = calculate_all_features(title, content_top, content_middle, content_bottom,
            positive_dictionaries, weights, stopwords)

            # Simpan fitur dan label ke dalam data
            data[label].append(features)

# Pembagian data menjadi training set (80%) dan testing set (20%)
training_data = []
testing_data = []

for label in data:
    # Acak data
    random.shuffle(data[label])
    # Hitung jumlah data untuk training set
    train_size = int(len(data[label]) * 0.8)
    # Pisahkan data
    training_data.extend([(features, label) for features in data[label][:train_size]])
    testing_data.extend([(features, label) for features in data[label][train_size:]])

# Gabungkan data training dan testing ke dalam satu file CSV
combined_data = training_data + testing_data

def save_to_csv(file_path, data):
    with open(file_path, "w", encoding="utf-8") as file:
        for features, label in data:
            file.write(",".join(map(str, features)) + "," + str(label) + "\n")

save_to_csv("combined_features.csv", combined_data)
save_to_csv("training_features.csv", training_data)
save_to_csv("testing_features.csv", testing_data)

print("Features extracted and saved to CSV files.")

```

Gambar D.b.1. Scripts create_feature.py

Berikut adalah tahapan dalam kode `create_feature.py`:

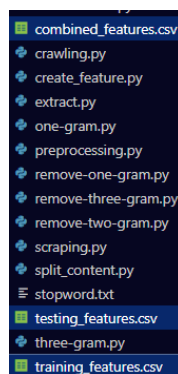
1. Fungsi `clean_text` digunakan untuk membersihkan teks dari karakter khusus dan angka, serta mengubahnya menjadi huruf kecil. Kemudian, fungsi `read_stopwords` membaca daftar stopwords dari file eksternal dan menyimpannya dalam sebuah set.
2. Fungsi `count_words` digunakan untuk menghitung jumlah kata dalam teks setelah menghapus stopwords. Selanjutnya, fungsi `calculate_title_feature` dan `calculate_content_feature` digunakan untuk menghitung fitur untuk bagian judul dan

konten (bagian atas, tengah, atau bawah) dari teks. Ini dilakukan dengan memperhitungkan kamus yang telah dibentuk, bobot bagian konten, dan stopwords.

3. Setiap file teks diproses dengan membaginya menjadi bagian judul, atas, tengah, dan bawah. Fitur-fitur diekstraksi menggunakan fungsi-fungsi yang telah disebutkan sebelumnya dalam looping, dengan memperhitungkan kamus-kamus yang relevan dan stopwords. Hasil ekstraksi fitur disimpan dalam bentuk list.
4. Data untuk masing-masing kategori diacak dan dibagi menjadi 80% untuk training set dan 20% untuk testing set. Setiap data fitur dan label disimpan dalam satu file CSV terpisah, yaitu ``combined_features.csv``, ``training_features.csv``, dan ``testing_features.csv``. Di mana, `combined_features.csv` ialah penggabungan data training dan testing. Proses penyimpanan dilakukan dengan fungsi ``save_to_csv``.
5. Pesan "Features extracted and saved to CSV files." ditampilkan untuk memberi informasi bahwa ekstraksi fitur telah selesai dan data telah disimpan dalam file CSV.

```
PS C:\Users\USER DK\OneDrive\Documents\sem 6\coba> py create_feature.py
Features extracted and saved to CSV files.
```

Gambar D.b.2 Output sukses membangun model hasil perhitungan fitur dari setiap artikel



Gambar D.b.3. Output yang dihasilkan

[illegible]

Gambar D.b.4. Tampilan isi dari file training_features.csv


```

1  testing_features.csv
2  0.06293706293706294, 0.01146851146851147, 0.018947368421052633, 0.017763157894736842, 0.06293706293706294, 0.01146851146851147, 0.018947368421052633, 0.017763157894736842, 0.06293706293706294
3  0.07692307692307693, 0.038461538461538464, 0.03333333333333333, 0.024999999999999998, 0.07692307692307693, 0.038461538461538464, 0.03333333333333333, 0.024999999999999998, 0.07692307692307693
4  0.043478260869565216, 0.021739130434782608, 0.042551914893617, 0.024999999999999998, 0.043478260869565216, 0.021739130434782608, 0.042551914893617, 0.024999999999999998, 0.043478260869565216
5  0.07079646017699115, 0.035398230088495575, 0.03333333333333334, 0.01764705882352941, 0.07079646017699115, 0.035398230088495575, 0.03333333333333334, 0.01764705882352941, 0.07079646017699115
6  0.04597701149425287, 0.022988505747126436, 0.023931623931623933, 0.013888888888888888, 0.04597701149425287, 0.022988505747126436, 0.023931623931623933, 0.013888888888888888, 0.04597701149425287
7  0.058823529411764705, 0.029411764705882353, 0.009756975697569756, 0.04285714285714286, 0.058823529411764705, 0.029411764705882353, 0.009756975697569756, 0.04285714285714286, 0.058823529411764705
8  0.014492753623188406, 0.007246376811594203, 0.0, 0.014285714285714286, 0.014492753623188406, 0.007246376811594203, 0.0, 0.014285714285714286, 0.014492753623188406
9  0.07575757575757576, 0.03787878787878788, 0.0, 0.01935483870967242, 0.07575757575757576, 0.03787878787878788, 0.0, 0.01935483870967242, 0.07575757575757576
10 0.01333333333333334, 0.00666666666666667, 0.012403180775193798, 0.021428571428571425, 0.01333333333333334, 0.00666666666666667, 0.012403180775193798, 0.021428571428571425, 0.01333333333333334
11 0.0196078431372549, 0.00980392156862745, 0.0070921985815602835, 0.003157894736842105, 0.0196078431372549, 0.00980392156862745, 0.0070921985815602835, 0.003157894736842105, 0.0196078431372549
12 0.028985507246376812, 0.014492753623188406, 0.005405405405405406, 0.024193548387096724, 0.028985507246376812, 0.014492753623188406, 0.005405405405405406, 0.024193548387096724, 0.028985507246376812
13 0.008138081380813809, 0.0040650406504065045, 0.0, 0.009756975697569756, 0.008138081380813809, 0.0040650406504065045, 0.0, 0.009756975697569756, 0.008138081380813809
14 0.02272727272727273, 0.011363636363636364, 0.014545454545454545, 0.015789473684210527, 0.02272727272727273, 0.011363636363636364, 0.014545454545454545, 0.015789473684210527, 0.02272727272727273
15 0.08571428571428571, 0.04285714285714286, 0.01, 0.02571428571428571, 0.08571428571428571, 0.04285714285714286, 0.01, 0.02571428571428571, 0.08571428571428571
16 0.0594587155963303, 0.027522935779816515, 0.006349206349206349, 0.01276595744680851, 0.0594587155963303, 0.027522935779816515, 0.006349206349206349, 0.01276595744680851, 0.0594587155963303
17 0.09756975697569756, 0.04878048780487805, 0.02666666666666667, 0.02054794520547945, 0.09756975697569756, 0.04878048780487805, 0.02666666666666667, 0.02054794520547945, 0.09756975697569756
18 0.02048081632653062, 0.01026582278481013, 0.0, 0.02048081632653062, 0.02048081632653062, 0.01026582278481013, 0.0, 0.02048081632653062, 0.02048081632653062
19 0.10256410256410256, 0.05128205128205128, 0.007692307692307693, 0.008333333333333333, 0.10256410256410256, 0.05128205128205128, 0.007692307692307693, 0.008333333333333333, 0.10256410256410256
20 0.07187031707170717, 0.036585365853658534, 0.01214953271028034, 0.000, 0.07187031707170717, 0.036585365853658534, 0.01214953271028034, 0.000, 0.07187031707170717
21 0.06790123456790123, 0.033950617283950615, 0.0150943362264151, 0.02142857142857143, 0.06790123456790123, 0.033950617283950615, 0.0150943362264151, 0.02142857142857143, 0.06790123456790123
22 0.0375998496240601, 0.01796992481203006, 0.015384615384615387, 0.007258064516129032, 0.0375998496240601, 0.01796992481203006, 0.015384615384615387, 0.007258064516129032, 0.0375998496240601
23 0.0163934426295982, 0.00819672131147541, 0.004, 0.0, 0.0163934426295982, 0.00819672131147541, 0.004, 0.0, 0.0163934426295982, 0.00819672131147541
24 0.021255811953488372, 0.011627980976744108, 0.02711864467796613, 0.0069767441806405115, 0.021255811953488372, 0.011627980976744108, 0.02711864467796613, 0.0069767441806405115, 0.021255811953488372

```

Gambar D.b.5. Tampilan isi dari file testing_features.csv

```

1  combined_features.csv
2  0.0784313725490196, 0.0392156862745098, 0.02711864467796613, 0.024489795918367346, 0.0784313725490196, 0.0392156862745098, 0.02711864467796613, 0.024489795918367346, 0.0784313725490196
3  0.04225352112676056, 0.02112676056338028, 0.009195402298050575, 0.0, 0.04225352112676056, 0.02112676056338028, 0.009195402298050575, 0.0, 0.04225352112676056
4  0.02222222222222223, 0.011111111111111112, 0.03636363636363636, 0.01875, 0.02222222222222223, 0.011111111111111112, 0.03636363636363636, 0.01875, 0.02222222222222223
5  0.0, 0.0, 0.010526315789473684, 0.006666666666666666, 0.0, 0.0, 0.010526315789473684, 0.006666666666666666, 0.0, 0.0, 0.010526315789473684, 0.006666666666666666, 0.0, 0.0, 0.010526315789473684
6  0.053763440886215055, 0.026881720430107527, 0.020512820512820516, 0.017857142857142856, 0.053763440886215055, 0.026881720430107527, 0.020512820512820516, 0.017857142857142856, 0.053763440886215055
7  0.03636363636363636, 0.01818181818181818, 0.0077669902912621365, 0.004054054054054054, 0.03636363636363636, 0.01818181818181818, 0.0077669902912621365, 0.004054054054054054, 0.03636363636363636
8  0.030383038303830384, 0.015151515151515152, 0.0, 0.005668377358490566, 0.030383038303830384, 0.015151515151515152, 0.0, 0.005668377358490566, 0.030383038303830384
9  0.09230769230769231, 0.046153846153846156, 0.013483146067415732, 0.008955223880597015, 0.09230769230769231, 0.046153846153846156, 0.013483146067415732, 0.008955223880597015, 0.09230769230769231
10 0.06349206349206349, 0.031746031746031746, 0.016438151643815164, 0.006249206349206349, 0.06349206349206349, 0.031746031746031746, 0.016438151643815164, 0.006249206349206349, 0.06349206349206349
11 0.03706703706703707, 0.01048315168351684, 0.006611570247933885, 0.0, 0.03706703706703707, 0.01048315168351684, 0.006611570247933885, 0.0, 0.03706703706703707
12 0.052924931086350974, 0.026402395543175487, 0.013513513513513514, 0.0033994334277620396, 0.052924931086350974, 0.026402395543175487, 0.013513513513513514, 0.0033994334277620396, 0.052924931086350974
13 0.052631578947368421, 0.02631578947368421, 0.02753036437246964, 0.0173399014778325, 0.052631578947368421, 0.02631578947368421, 0.02753036437246964, 0.0173399014778325, 0.052631578947368421
14 0.07563025210084033, 0.037815126050420166, 0.035164835164835165, 0.04191176470588236, 0.07563025210084033, 0.037815126050420166, 0.035164835164835165, 0.04191176470588236, 0.07563025210084033
15 0.018518518518518517, 0.009259259259259259, 0.027659574468085108, 0.006521739130434782, 0.018518518518518517, 0.009259259259259259, 0.027659574468085108, 0.006521739130434782, 0.018518518518518517
16 0.043478260869565216, 0.021739130434782608, 0.009756975697569756, 0.00821917808219178, 0.043478260869565216, 0.021739130434782608, 0.009756975697569756, 0.00821917808219178, 0.043478260869565216
17 0.0, 0.0, 0.016666666666666666, 0.007692307692307692, 0.0, 0.0, 0.016666666666666666, 0.007692307692307692, 0.0, 0.0, 0.016666666666666666, 0.007692307692307692
18 0.052631578947368421, 0.02631578947368421, 0.02753036437246964, 0.0173399014778325, 0.052631578947368421, 0.02631578947368421, 0.02753036437246964, 0.0173399014778325, 0.052631578947368421
19 0.02989074626056716, 0.01492537313428358, 0.016438151643815164, 0.002916666666666667, 0.02989074626056716, 0.01492537313428358, 0.016438151643815164, 0.002916666666666667, 0.02989074626056716
20 0.08333333333333333, 0.041666666666666664, 0.04, 0.020689655172413793, 0.08333333333333333, 0.041666666666666664, 0.04, 0.020689655172413793, 0.08333333333333333
21 0.029411764705882353, 0.014705882352941176, 0.01038961038961039, 0.016, 0.029411764705882353, 0.014705882352941176, 0.01038961038961039, 0.016, 0.029411764705882353
22 0.0273972602739726, 0.0136986301369863, 0.003448275862086955, 0.0273972602739726, 0.0273972602739726, 0.0136986301369863, 0.003448275862086955, 0.0273972602739726, 0.0273972602739726
23 0.00976744180640512, 0.03488172093021256, 0.014035807719298246, 0.0, 0.00976744180640512, 0.03488172093021256, 0.014035807719298246, 0.0, 0.00976744180640512
24 0.04166666666666666, 0.08033333333333333, 0.004538771640493715, 0.010714785714285714, 0.04166666666666666, 0.08033333333333333, 0.004538771640493715, 0.010714785714285714, 0.04166666666666666

```

Gambar D.b.6. Tampilan isi dari file combined_features.csv

c. Classification.py

```

import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, accuracy_score
from sklearn.preprocessing import StandardScaler

# Fungsi untuk membaca data dari CSV dan memisahkan fitur dan label
def load_data(file_path):
    df = pd.read_csv(file_path, header=None)
    X = df.iloc[:, :-1].values
    y = df.iloc[:, -1].values
    return X, y

# Baca data dari CSV
X_train, y_train = load_data("training_features.csv")
X_test, y_test = load_data("testing_features.csv")

# Normalisasi data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Hyperparameter tuning untuk SVM
svm_params = {
    'C': [0.1, 1, 10, 100],
    'gamma': [1, 0.1, 0.01, 0.001],
    'kernel': ['rbf', 'linear']
}

svm_grid = GridSearchCV(SVC(), svm_params, refit=True, verbose=2, cv=5)
svm_grid.fit(X_train, y_train)
print("Best SVM Parameters:", svm_grid.best_params_)
svm_best = svm_grid.best_estimator_
y_pred_svm = svm_best.predict(X_test)
print("SVM Classification Report:")
print(classification_report(y_test, y_pred_svm))
print("SVM Accuracy:", accuracy_score(y_test, y_pred_svm))

# Hyperparameter tuning untuk KNN
knn_params = {
    'n_neighbors': range(1, 31),
    'weights': ['uniform', 'distance']
}

knn_grid = GridSearchCV(KNeighborsClassifier(), knn_params, refit=True, verbose=2, cv=5)
knn_grid.fit(X_train, y_train)
print("Best KNN Parameters:", knn_grid.best_params_)
knn_best = knn_grid.best_estimator_
y_pred_knn = knn_best.predict(X_test)
print("KNN Classification Report:")
print(classification_report(y_test, y_pred_knn))
print("KNN Accuracy:", accuracy_score(y_test, y_pred_knn))

```

Gambar D.c.1. Scripts create_feature.py

Kode dalam `classification.py` bertujuan untuk melatih dan mengevaluasi model klasifikasi menggunakan dua algoritma, yaitu Support Vector Machine (SVM) dan K-Nearest Neighbors (KNN).

1. Fungsi `load_data` digunakan untuk membaca data dari file CSV. Data dibaca menggunakan `pandas` dan dipisahkan menjadi fitur (X) dan label (y). Fungsi ini membaca data dari file `training_features.csv` dan `testing_features.csv` dan mengembalikan dua array: `X_train`, `y_train` untuk data latih dan `X_test`, `y_test` untuk data uji.
2. Untuk memastikan bahwa semua fitur berada pada skala yang sama, data fitur dinormalisasi menggunakan `StandardScaler` dari `sklearn.preprocessing`. `StandardScaler` menyesuaikan skala data latih (`X_train`) guna memastikan setiap fitur memiliki rata-rata 0 dan standar deviasi 1, yang penting bagi algoritma seperti SVM dan KNN yang sensitif terhadap skala data. Kemudian diterapkan transformasi yang sama ke data uji (`X_test`).
3. Model SVM (`SVC`) dilatih dan dioptimalkan menggunakan `GridSearchCV` untuk menemukan kombinasi parameter terbaik. Parameter yang dicoba meliputi 'C' (regulasi), 'gamma' (koefisien kernel), dan 'kernel' (jenis kernel). Grid search dilakukan dengan 5-fold cross-validation (`cv=5`). Model terbaik dipilih berdasarkan hasil tuning, dan prediksi dilakukan pada data uji menggunakan model terbaik (`svm_best`). Evaluasi performa model dilakukan menggunakan `classification_report` dan `accuracy_score`.
4. Model KNN (`KNeighborsClassifier`) juga dioptimalkan menggunakan `GridSearchCV`. Parameter yang dicoba meliputi 'n_neighbors' (jumlah tetangga terdekat) dan 'weights' (bobot voting). Sama seperti pada SVM, grid search dilakukan dengan 5-fold cross-validation. Model terbaik dipilih berdasarkan hasil tuning, dan prediksi dilakukan pada data uji menggunakan model terbaik (`knn_best`). Evaluasi performa model dilakukan dengan `classification_report` dan `accuracy_score`.

[illegible]

Gambar D.c.2 Tampilan nilai akurasi model SVM

F. Kesimpulan

1. Proses ekstraksi fitur dari teks dilakukan menggunakan skrip `Create-Feature.py`. Fitur-fitur diekstraksi dari judul dan konten artikel dengan memperhitungkan kamus-kamus yang telah dibentuk sebelumnya. Fitur-fitur ini kemudian digunakan sebagai masukan untuk algoritma klasifikasi. Tahapan-tahapan dalam skrip ini meliputi membersihkan teks, menghitung jumlah kata, dan menghitung fitur-fitur berdasarkan kamus-kamus yang telah dibangun sebelumnya.
2. Model klasifikasi menggunakan dua algoritma, yaitu Support Vector Machine (SVM) dan K-Nearest Neighbors (KNN). Proses ini dilakukan dalam skrip `Classification.py`. Tahapan-tahapan dalam skrip ini meliputi pembacaan data, normalisasi data, hyperparameter tuning menggunakan `GridSearchCV`, pelatihan model dengan data latih, prediksi pada data uji, dan evaluasi performa model menggunakan metrik-metrik klasifikasi seperti precision, recall, f1-score, dan akurasi.
3. Untuk memperoleh model terbaik, kedua algoritma klasifikasi, SVM dan KNN, dioptimalkan menggunakan teknik Grid Search Cross Validation dengan 5-fold cross-validation. Proses ini membantu menemukan kombinasi parameter terbaik untuk masing-masing algoritma, seperti 'C', 'gamma', dan 'kernel' untuk SVM, serta 'n_neighbors' dan 'weights' untuk KNN. Hasil dari proses optimasi ini memungkinkan untuk memilih model dengan performa terbaik.
4. Dari hasil evaluasi yang diberikan, terlihat bahwa model SVM (0.7633451957295374 atau 76.33%) memiliki performa yang lebih baik dibandingkan dengan model KNN (0.6512455516014235 atau 65.12%) dalam memprediksi kelas-kelas dalam dataset tersebut. Untuk SVM, parameter terbaik yang ditemukan adalah $C=100$, $\gamma=0.001$, dan $\text{kernel}='rbf'$. Sedangkan untuk KNN, parameter terbaik yang ditemukan adalah $n_neighbors=28$ dan $\text{weights}='distance'$.