TUGAS AKHIR – KI141502

KLASIFIKASI KEMURNIAN DAGING SAPI BERBASIS ELECTRONIC NOSE

ALDHIAZ FATHRA DAIVA
NRP 5113100006

Dosen Pembimbing
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dedy Rahman Wijaya, S.T., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KI141502

KLASIFIKASI KEMURNIAN DAGING SAPI BERBASIS ELECTRONIC NOSE

ALDHIAZ FATHRA DAIVA
NRP 5113100006

Dosen Pembimbing I
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.

Dosen Pembimbing II
Dedy Rahman Wijaya, S.T., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

FINAL PROJECT– KI141502

ELECTRONIC NOSE BASED BEEF PURENESS CLASSIFICATION

ALDHIAZ FATHRA DAIVA
NRP 5113100006

Supervisor I
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.

Supervisor II
Dedy Rahman Wijaya, S.T., M.T.

DEPARTMENT OF INFORMATICS
Faculty of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

KLASIFIKASI KEMURNIAN DAGING SAPI BERBASIS ELECTRONIC NOSE

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Manajemen Informasi
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
ALDHIAZ FATHRA DAIVA
NRP : 5113 100 006

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
NIP: 19590803 198601 1001

.....
(Pembimbing 1)

Dedy Rahman Wijaya, S.T., M.T.
NIP: 07840011

.....
(Pembimbing 2)

SURABAYA
JULI 2018

[Halaman ini sengaja dikosongkan]

KLASIFIKASI KEMURNIAN DAGING SAPI BERBASIS ELECTRONIC NOSE

Nama Mahasiswa	: Aldhiaz Fathra Daiva
NRP	: 5113 100 006
Jurusan	: Informatika FTIK-ITS
Dosen Pembimbing 1	: Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dosen Pembimbing 2	: Dedy Rahman Wijaya, S.T., M.T.

Abstrak

Daging merupakan salah satu konsumsi utama manusia. Namun dalam praktik jual beli masih terdapat pencampuran jenis daging untuk mendapatkan keuntungan yang lebih secara curang. Terlebih pencampuran menggunakan daging Babi yang dalam ajaran Islam merupakan makanan yang dilarang. Dengan dilakukannya penelitian ini, kami merancang dan mengusulkan sistem sederhana berbiaya rendah yang dapat m*endeteksi kemurnian daging berdasarkan bau yang dideteksi dengan mengukur konsentrasi gas yang dikeluarkan daging melalui electronic nose yang dirakit sendiri.*

Sistem ini dibangun melalui enam tahap: pembuatan perangkat keras electronic nose menggunakan sensor gas dan Arduino; pengambilan sampel data; praproses data menggunakan discrete wavelet transform untuk mengurangi noise; ekstraksi fitur statistik; seleksi fitur menggunakan chi-squared statistic weighting; dan klasifikasi menggunakan metode kNN, decision tree, naïve bayes, dan SVM.

Hasil penelitian menunjukkan bahwa sistem ini dapat membedakan daging sapi yang dicampur dengan daging babi

dengan perbandingan 0%, 10%, 25%, 50%, 75%, 90%, dan 100% dengan akurasi 97,86% menggunakan algoritma kNN.

Kata kunci: klasifikasi, daging, electronic nose, Arduino, pemrosesan sinyal

ELECTRONIC NOSE BASED BEEF PURENESS CLASSIFICATION

Student Name	: Aldhiaz Fathra Daiva
Student ID	: 5113 100 006
Major	: Informatics, FTIK-ITS
Supervisor 1	: Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Supervisor 2	: Dedy Rahman Wijaya, S.T., M.T.

Abstract

Meat is one of the mainly consumed food for human. However in the real practice, meat adultery by mixing other type of meat is still occurring. Especially mixing using pork meat which in Islamic view is a prohibited food. This practice is done to fraudulently gain more income. In this research, we develop and propose an easy to use, low cost system to classify wether the meat is a pure beef or not.

The system is developed through six main stages: developing of electronic nose hardware using gas sensors and Arduino, gathering ground truth data for the training set and evalution, data preprocessing by utilizing discrete wavelet transform for denoising, statistical features extraction, feature selection using chi-squared weighting, and classification using kNN, decision tree, naïve bayes, and SVM.

The experiment results show that this system can classify pork adulteration in a beef with percentage of 0% 10%, 25%, 50%, 75%, 90%, and 100% with performance of 97.86% of accuracy using kNN algorithm.

Keyword: classification, meat, electronic nose, Arduino, signal processing

KATA PENGANTAR

Puji syukur penulis ucapkan ke hadirat Allah Azza wa Jalla, karena atas segala berkat, rahmat, dan karunia-Nya dapat terselesaikannya penulisan tugas akhir ini yang berjudul:

KLASIFIKASI KEMURNIAN DAGING SAPI BERBASIS ELECTRONIC NOSE

Izinkan penulis berterus terang bahwa penulis sangat bersemangat untuk mengerjakan penelitian ini saat penulis pertama kali mendengar dan kemudian memilih topik ini. Hal ini karena di dalam topik ini selain mengandung bidang yang penulis minati yaitu *machine learning* tetapi juga menggunakan perangkat yang dari dulu ingin penulis pelajari dan implementasikan tetapi belum terlaksana yaitu papan Arduino.

Tugas akhir ini tidak akan berhasil terselesaikan kalau - dukungan dan bantuan dari berbagai pihak. Oleh karena itu, melalui lembar ini, penulis ingin menyampaikan ucapan terima kasih dan penghormatan yang sebesar-besarnya kepada seluruh pihak yang telah banyak sekali memberi perhatian khusus serta membimbing penulis tanpa kenal lelah. Akan tetapi tidak akan kan satu persatu secara eksplisit pihak-pihak tersebut karena sangat banyak individu yang telah memberikan dukungan terhadap penulis.

Pada dasarnya penulis sangat menyadari masih terdapat banyak sekali kekurangan pada pengerjaan tugas akhir ini, baik dalam pelaksanaan penelitian maupun penyusunan buku. Namun penulis berharap pengerjaan buku tugas akhir ini dapat menjadi pelajaran berharga untuk pribadi penulis ke depan dan hasilnya dapat memberi manfaat bagi pihak-pihak yang terlibat dalam pelaksanaan penelitian termasuk juga para pembaca.

Surabaya, Juli 2018

Penulis

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Bab Pendahuluan ini berisi mengenai dasar tugas akhir yang terdiri dari latar belakang masalah, rumusan permasalahan, batasan permasalahan, tujuan, manfaat, metodologi, dan sistematika penulisan.

1.1 Latar Belakang Masalah

Dewasa ini ditemukan praktik pencampuran suatu jenis daging dengan daging jenis lain yang lebih murah untuk meraup keuntungan yang lebih. Salah satu contohnya adalah pencampuran daging sapi dengan daging babi. Kegiatan ini dilakukan tanpa adanya izin dari pihak yang berwenang maupun sepengetahuan dari pihak konsumen. Selain merugikan, pencampuran menggunakan daging babi juga sangat meresahkan masyarakat. Hal ini karena daging babi merupakan makanan yang diharamkan bagi umat Islam.

Penelitian sebelumnya telah dilakukan mengenai klasifikasi kemurnian daging sapi dengan menggunakan bantuan *electronic nose*. Fachri Rosyad et al. telah melakukan penelitian dalam hal ini dengan menggunakan metode *Principal Component Analysis* (PCA). Pada penelitian ini algoritma *machine learning* masih belum dilibatkan.

Pada tugas akhir ini akan diterapkan analisis lebih lanjut dengan melibatkan algoritma *machine learning*, dengan ekspektasi hasil klasifikasi yang lebih optimal. Algoritma machine learning yang digunakan adalah kNN, *decision tree*, *naïve bayes*, dan *support vector machine*. Performa dari kedua algoritma ini akan dibandingkan dan didapat hasil yang paling optimal.

1.2 Rumusan Permasalahan

Rumusan masalah yang terdapat dalam tugas akhir ini antara lain:

1. Bagaimana cara mendapatkan data dari bau daging?
2. Bagaimana cara merakit perangkat yang digunakan untuk mendapatkan data bau daging?
3. Bagaimana cara pembelajaran data untuk klasifikasi kemurnian daging sapi dari pencampuran daging babi berdasarkan data dari *electronic nose*?
4. Bagaimana cara penyajian hasil analisis data dari alat yang akan dibangun?

1.3 Batasan Permasalahan

Dalam tugas akhir ini, bahasan permasalahan dibatasi pada hal-hal sebagai berikut:

1. Data daging dikumpulkan dari daging sapi dan daging babi segar yang belum diolah, dicincang menggunakan *blender* untuk mempermudah pencampuran daging.
2. Alat yang dipakai untuk pengumpulan data adalah *electronic nose*, dirakit sendiri menggunakan sensor gas elektrokimia yang tersambung pada Arduino.
3. Pengumpulan data dilakukan pada lingkungan kamar yang menggunakan pendingin ruangan pada suhu ~26°C dan kelembaban ~65%.
4. Estetika penyajian hasil analisa adalah di luar konteks penelitian ini.
5. Ilmu dasar dari metode klasifikasi yang diuji coba pada penelitian ini berada di luar bidang penelitian, sehingga tidak akan dibahas secara mendalam.

1.4 Tujuan

Tugas akhir ini dibuat dengan tujuan sebagai berikut:

1. Membuat alat *electronic nose* untuk mangumpulkan data gas pada daging.
2. Melakukan klasifikasi berdasarkan persentase kadar daging sapi dari data yang direkam oleh *electronic nose*.

1.5 Manfaat

Dengan disusunnya tugas akhir ini diharapkan dapat memberikan kemudahan dalam mendeteksi kemurnian daging sapi dari daging babi secara praktis dengan biaya alat yang terjangkau.

1.6 Metodologi

Metode penyusunan tugas akhir ini adalah:

a. Penyusunan proposal tugas akhir

Langkah pertama adalah menyusun proposal tugas akhir. Proposal terdiri dari pendahuluan dan penjelasan umum tugas akhir. Pendahuluan terdiri dari latar belakang, rumusan masalah, batasan masalah, tujuan, dan manfaat tugas akhir. Sebagai pendukung dalam penyusunan tugas akhir dibahas pula mengenai tinjauan pustaka dari beberapa penelitian terdahulu sebagai referensi awal dalam tugas akhir. Di dalam proposal dijelaskan juga mengenai metodologi penyusunan tugas akhir yang menjelaskan tentang tahapan penyusunan tugas akhir. Sebagai pelengkap proposal disajikan juga daftar kegiatan dan jadwal pembuatan tugas akhir.

b. Studi literatur

Tahap studi literatur merupakan tahap yang penting dalam penyusunan tugas akhir untuk mempelajari penelitian-penelitain sebelumnya maupun bacaan yang relevan yang bisa dijadikan referensi. Beberapa literatur yang relevan

yaitu mengenai daging, mikrokontroler dan Arduino, pemrograman Arduino, sensor elektrokimia, dan sensor udara. Selain itu diperlukan juga literatur tentang cara pengolahan data, pengukuran konsentrasi gas, analisis sinyal, seleksi fitur, serta algoritma dan metode klasifikasi.

c. Desain perangkat lunak dan analisis

Tahap-tahap desain perangkat lunak yaitu pembuatan *electronic nose*, pengumpulan data, pengujian metode, dan pembuatan GUI (*Graphical User Interface*).

Electronic nose dibuat dengan menggunakan sensor elektrokimia dan sensor udara baik temperatur, tekanan, maupun kelembapan udara, Dengan *electronic nose* ini dilakukan pengumpulan data daging yang akan dianalisis lebih lanjut.

Langkah berikutnya adalah uji coba metode praproses dan metode klasifikasi untuk mengevaluasi performa metode-metode yang digunakan untuk klasifikasi daging. Untuk menampilkan hasil prediksi oleh sistem maupun pemilihan parameter digunakan GUI (*Graphical User Interface*). Pada langkah ini dibuat GUI berbasis *desktop.*

Adapun analisis yang dilakukan adalah pengujian algoritma klasifikasi kNN.

d. Implementasi perangkat lunak

Berikut ini adalah lingkungan pemrograman yang digunakan untuk membuat sistem:

1. Hyper 2.0.0: untuk mengonfigurasi lingkungan pemrograman dan melakukan uji coba program.
2. Sublime Text 3.1.1, *Build* 3176: untuk menulis program.
3. Arduino IDE 1.8.5 (*Windows Store* 1.8.10.0): untuk mengunggah program ke papan Arduino.

4. Rapidminer Studio 8.1.0: untuk melakukan seleksi fitur dan uji coba klasifikasi.
5. Microsoft Visio 2016: untuk membuat diagram.

Bahasa pemrograman yang digunakan pada sistem ini adalah sebagai berikut:

1. Bahasa Pemrograman Arduino: untuk memrogram papan Arduino.
2. Python 3.6.5: untuk membuat model klasifikasi menggunakan *library* scikit-learn 0.19.1 dan membangun antarmuka menggunakan kerangka kerja PyQt5.

e. Pengujian dan evaluasi

Pengujian dan evaluasi dari sistem ini dilakukan dengan beberapa cara yaitu:

1. Pengujian pembacaan sensor
   Hasil pembacaan sensor perlu diuji apakah berhasil mendapatkan data yang diinginkan dan apakah komunikasi antara papan Arduino dengan komputer dapat berjalan dengan lancar.
2. Pengujian kinerja sensor
   Pengujian kinerja sensor baik keakuratan, kepresisian, dan sensitivitas dilakukan menggunakan *dataset* yang dihimpun dari hasil uji coba. Pengujuan dilakukan dengan menggunakan metode evaluasi validasi silang dan perhitungan akurasi, presisi, sensitivitas, dan statistik kappa untuk mengukur performa sistem dalam melakukan klasifikasi daging.

f. Penyusunan Buku Tugas Akhir

Tahap penyusunan buku meliputi dasar teori, penjelasan metode, perancangan beserta hasil implementasi aplikasi sistem yang telah dibuat, serta pembahasan hasilnya. Bab-bab yang terdiri dalam buku ini antara lain:

1. Pendahuluan
2. Dasar Teori
3. Metodologi
4. Analisis dan Perancangan Sistem
5. Implementasi
6. Pengujian dan Evaluasi
7. Kesimpulan dan Saran
8. Daftar Pustaka

1.7 Sistematika Penulisan

Tujuan dari penyusunan buku adalah untuk memberi gambaran menyeluruh mengenai hasil penelitian yang telah dilakukan. Selain itu dengan buku ini diharapkan bisa menjadi bahan bacaan untuk dikembangkan lebih lanjut oleh pembaca yang tertarik. Sistematika penulisannya yaitu sebagai berikut:

Bab I Pendahuluan

Pada bab Pendahuluan disajikan mengenai latar belakang masalah, rumusan permasalahan, batasan permasalahan, tujuan, manfaat, metodologi, serta sistematika penulisan.

Bab II Dasar Teori

Bab Dasar Teori membahas penelitian sejenis yang pernah dilakukan serta beberapa teori yang relevan dengan pokok pembahasan untuk dijadikan sebagai dasar penyusunan tugas akhir ini.

Bab III Metodologi

Bab ini menjelaskan metodologi yang digunakan dalam pengerjaan tugas akhir yaitu perancangan *electronic nose*, pengujian alat, dan pengumpulan data.

Bab IV Analisis dan Perancangan Sistem

Pada bab ini dibahas mengenai perancangan perangkat lunak yang meliputi perancangan data, arsitektur, proses, dan perancangan antarmuka aplikasi.

Bab V Implementasi

Implementasi dari perancangan perangkat lunak serta antarmuka aplikasi dipaparkan pada bab ini.

Bab VI Pengujian dan Evaluasi

Bab ini membahas tentang lingkungan serta skenario pengujian dan evaluasi terhadap aplikasi yang telah dikembangkan.

Bab VII Kesimpulan dan Saran

Pada bab ini disajikan mengenai kesimpulan yang diperoleh dari hasil pengujian yang dilakukan dan saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Daftar Pustaka berisi daftar referensi yang digunakan dalam pembuatan tugas akhir.

Lampiran

Lampiran berisi data atau daftar istilah yang penting pada tugas akhir ini.

[Halaman ini sengaja dikosongkan]

BAB II
DASAR TEORI

Bab ini membahas mengenai penelitian sebelumnya serta teori-teori yang menjadi dasar dalam pengerjaan tugas akhir.

2.1 Penelitian Sebelumnya

Beberapa studi sebelumnya telah memperlihatkan kapabilitas *electronic nose* dalam membedakan beberapa tipe daging. Sensor semikonduktor digunakan untuk pengambilan data dari bau daging. Studi oleh Nurjuliana et al. menggunakan *principal component analysis* (PCA) untuk membedakan antara lemak babi, lemak ayam, lemak sapi, lemak domba, dan sampel campuran dari lemak babi dan lemak ayam. Eksperimen dilakukan pada temperature 50°C to 200°C[1]. Studi lainnya adalah mengenai mendeteksi daging domba cincang yang dicampur dengan daging babi dengan perbandingan 0%, 20%, 40%, 60%, 80%, dan 100%. Pendeteksian dilakukan pada suhu ruangan[2].

Terdapat juga studi lain yang metode klasifikasinya dapat digunakan dalam penelitian ini. Studi ini menggunakan *Information Quality Ratio* (IQR) untuk menganalisis sinyal dari tujuh sensor untuk menemukan *Mother Wavelet Transform* (MWT) yang paling cocok untuk digunakan dalam mengklasifikasi kualitas daging sapi[3].

2.2 Daging Sapi dan Daging Babi

Daging sapi merupakan salah satu makanan yang banyak dikonsumsi manusia. Hal ini karena daging sapi mengandung protein hewani yang bermanfaat bagi kesehatan dan bisa diolah menjadi makanan yang lezat. Di Indonesia pada saat-saat tertentu seperti pada saat hari raya, permintaan daging sapi meningkat. Di sisi lain suplai daging sapi tidak mencukupi banyaknya permintaan sehingga mahal. Dengan kondisi semacam ini, seringkali ada

penjual yang curang dengan melakukan praktik pencampuran daging dengan daging babi yang lebih murah. Kenyataan ini sangat merugikan masyarakat bahkan meresahkan karena mayoritas masyarakat Indonesia adalah muslim yang mengharamkan daging babi.

Daging sapi dan daging babi sesungguhnya bisa dibedakan dari warna, rasa, dan baunya. Namun demikian, pembedaan tersebut sulit dilakukan pada daging yang dicampur. Selain itu pembedaan berdasarkan warna, rasa, dan bau oleh manusia bersifat subjektif sehingga kurang tepat. Oleh karena itu, dibutuhkan instrumen yang lebih baik dalam mendeteksi kemurnian daging sapi.

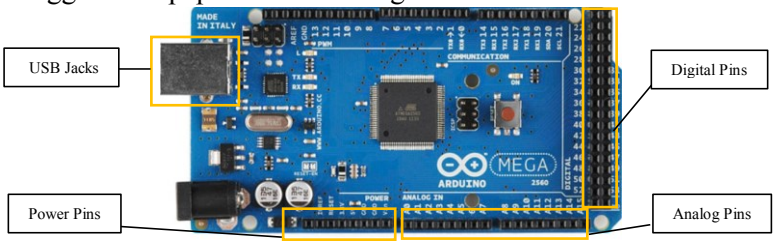
Pendeteksian berdasarkan warna bisa dilakukan sebagai contoh dengan menggunakan klasifikasi citra. Namun warna antara daging yang sejenis sudah berbeda membuat implementasinya menjadi sulit. Pendeteksian berdasarkan rasa bisa dilakukan dengan menggunakan *electronic tongue*. Demikian juga pendeteksian berdasarkan bau dapat dilakukan dengan menggunakan *electronic nose*. Kedua metode ini bisa dilakukan namun pada penelitian ini deteksi kemurnian daging sapi dilakukan dengan menggunakan *electronic nose*. Hal ini karena pada umumnya pembedaan kemurnian daging dilakukan pada daging mentah sehingga lebih lazim melakukan pembedaan kedua daging tersebut berdasarkan bau.

2.3 Papan Arduino

Arduino merupakan papan elektronik dengan komponen utama pengendali mikro dan didukung perangkat lunak yang bersifat open source sehingga mudah digunakan dalam berbagai bidang [4]. Mikrokontroler pada dasarnya merupakan suatu komputer yang berukuran kecil dengan beberapa fungsi-fungsi dasar computer yang cukup untuk mengendalikan beberapa komponen dan perangkat tambahan yang diperlukan

menyesuaikan kebutuhan proyek yang dikerjakan. Perangkat dan komponen tambahan ini dikendalikan dengan menuls program dan mengunggahnya ke mikrokontroler [1].

Pada penelitian ini, papan Arduino yang dilengkapi dengan beberapa sensor digunakan untuk membuat *electronic nose.* Alat ini digunakan untuk mengukur kandungan gas, suhu, dan kelembapan dalam sampel daging dengan menggunakan sensor elektrokimia maupun sensor lingkungan. Penelitian ini menggunakan papan Arduino Mega 2560.



Gambar 2.1 Papan Arduino Mega 2560

Arduino memiliki empat komponen utama yang digunakan dalam perangkaian *electronic nose*. Penjelasan mengenai fungsi komponen-komponen tersebut dapat dilihat pada Tabel 2.1.

Tabel 2.1 Nama dan Fungsi Komponen pada Papan Arduino MEGA 2560

No.	Nama	Fungsi
1.	*USB Jacks*	Penghubung antara Arduino dan komputer pada saat pengiriman data melalui kabel USB. Juga berfungsi sebagai sumber *power*.

2.	*Power Pins*	Sekumpulan pin yang berhubungan dengan arus masuk dan keluar. Berikut adalah pin yang digunakan: • Vin: sebagai pin arus keluar dari Arduino • GND (*Ground*): sebagai pin arus masuk ke Arduino
3.	*Analog Input Pins*	Sekumpulan pin untuk membaca sinyal analog dan mengubah data analog menjadi digital. Sensor yang menggunakan pin ini adalah MQ2, MQ4, MQ6, MQ9, MQ135, MQ136, MQ137, dan MQ138.
4.	*Digital Input Pins*	Sekumpulan pin untuk menangkap sinyal digital. Sensor yang menggunakan pin ini adalah DHT22.

2.4 Sensor Gas Elektrokimia

Informasi mengenai gas yang dikeluarkan dari bau daging didapat dengan menggunakan sensor elektrokimia. Sensor ini mendeteksi gas sesuai selektivitasnya dengan menghasilkan arus kecil yang diakibatkan dari reaksi kimia antara oksigen di dalam sensor dengan gas. Hasilnya bervariasi sesuai proporsi konsentrasi gas. Sensor ini dihubungkan ke Arduino agar data yang ditangkap dari sensor dapat dikirimkan ke komputer.

Penelitian ini menggunakan delapan sensor elektrokimia dan 1 sensor suhu dan kelembapan. Kedelapan sensor elektrokimia tersebut memiliki kepekaan terhadap gas-gas yang sebagaimana dipaparkan pada Tabel 2.2. Kombinasi dari delapan sensor gas

tersebut diperlukan untuk mengidentifikasi pola kandungan gas dari daging yang diuji coba. Sensor suhu dan kelembapan diperlukan untuk meyakinkan bahwa pengukuran gas dilakukan pada suhu dan kelembapan yang tidak bervariasi. Dengan demikian, pengukuran kandungan gas pada daging tidak terganggu oleh perbedaan lingkungan uji coba.

Data bau diambil menggunakan Arduino. Sensor gas elektrokimia bekerja berdasarkan reaksi antara komponen sensor dengan analit dari polutan gas yang dideteksi dalam bentuk ion ataupun gas sehingga sinyal yang dihasilkan koresponden dengan konsentrasi analit [5]. Sensor elektrokimia (MOS Sensor) inilah yang akan digunakan untuk mendeteksi data kandungan gas dalam daging mengguanakan papan Arduino. Penilitian ini menggunakan delapan macam sensor elektrokimia yang berbeda. Selain itu untuk mengukur temperatur dan kelembapan udara selama pengukuran digunakan sebuah sensor lingkungan. MOS sensor yang digunakan pada penelitian ini merupakan sensor analog yang mengeluarkan *output* digital melalui peralatan *analog to digital conversion* (ADC). Sensor ini mempunyai empat pin. Pin-pin tersebut adalah VCC, GND, AO, dan DO. Sesuai fungsinya keempat pin tersebut dihubungkan pada mikrokontroler. Daya pada sensor diberikan dengan menghubungkan pin VCC dan GND pada mikrokontroler. Pin AO dihubungkan untuk membaca data.

Terdapat beberapa sensor gas yang dapat digunakan dalam pembuatan *electronic nose* dengan karakteristik dan fungsi yang berbeda. Nama sensor dan fungsinya ditunjukkan pada Tabel 2.2 berikut. Karakteristik sensor dibahas pada subbab 2.4.1 sampa 2.4.9.

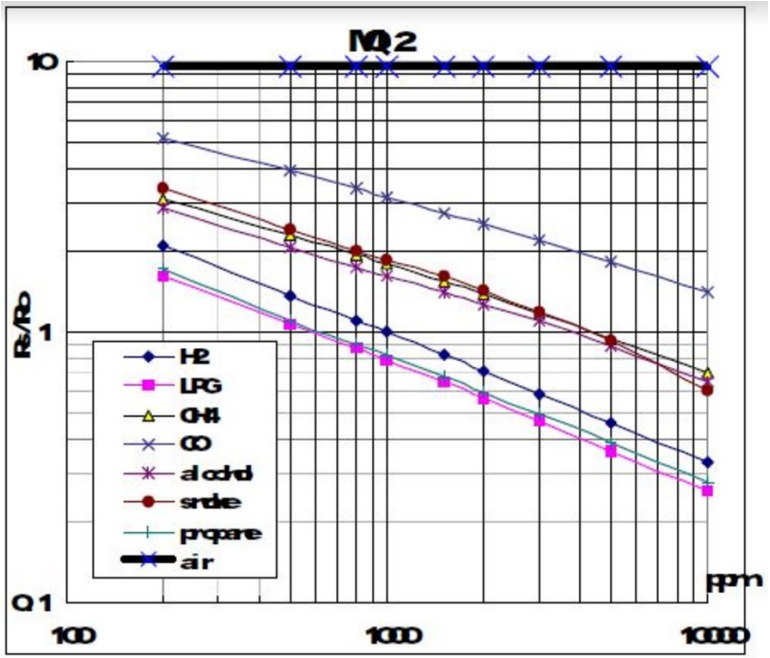
Tabel 2.2 Daftar Sensor Gas dan Fungsinya

Sensor	**Fungsi**

MQ2	LPG, i-butane, propane, methane, alcohol, H_2, smoke
MQ4	Methane (CH_4) Natural gas
MQ6	LPG, iso-butane, propane
MQ9	Methane, Propane and CO
MQ135	Carbon Dioxide (CO_2)
MQ136	Hydrogen Sulfide (H_2S)
MQ137	Amonia (NH_3)
MQ138	Ketone
DHT22	Temperature-Humidity

2.4.1 MQ2

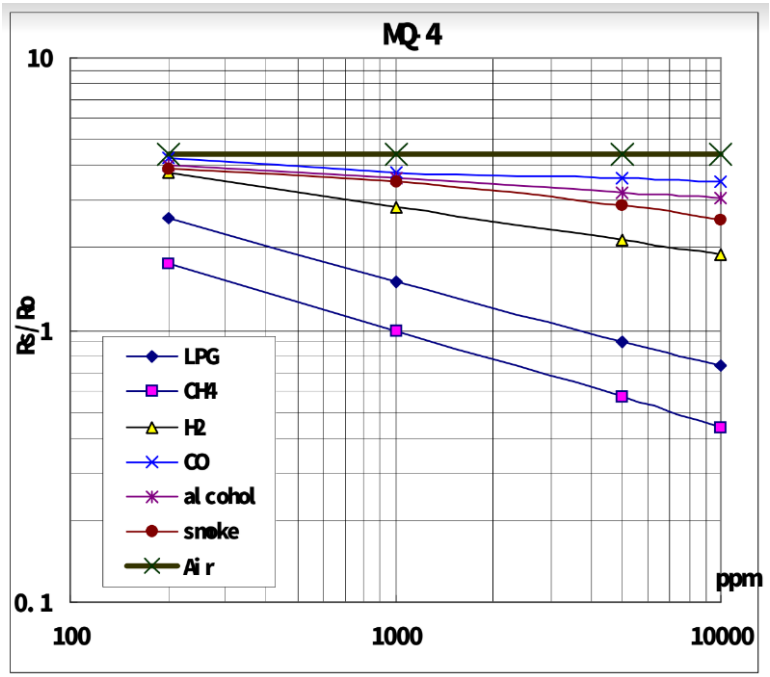
Sensor MQ2 memiliki sensitivitas terhadap H_2, LPG, NH_4, CO, alkohol, asap, dan propana. Karakteristik sensor MQ2 dapat dilihat pada Gambar 2.2 [6].



Gambar 2.2 Datasheet Sensor MQ2

2.4.2 MQ4

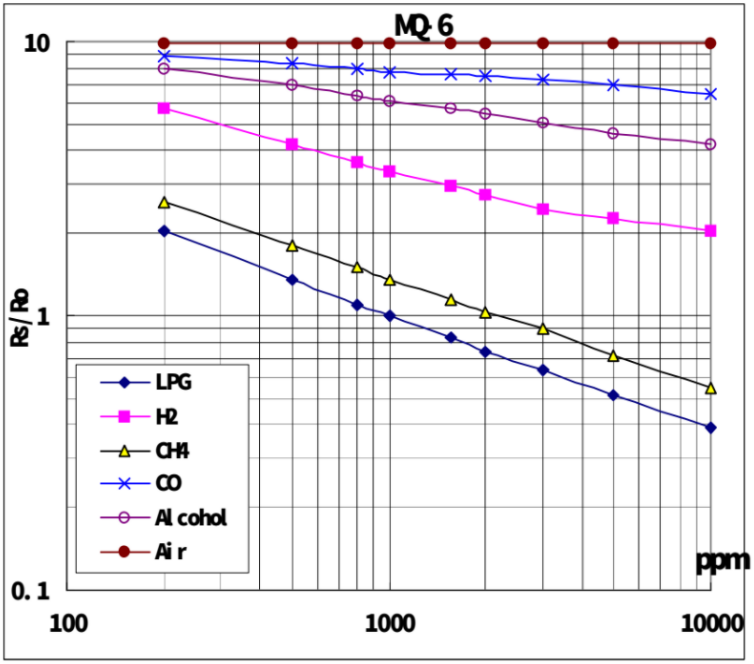
Sensor MQ4 memiliki sensitivitas terhadap LPG, CH_4, H_2, CO, alkohol, dan asap. Karakteristik sensor MQ4 dapat dilihat pada Gambar 2.3[7][8].



Gambar 2.3 Datasheet Sensor MQ4

2.4.3 MQ6

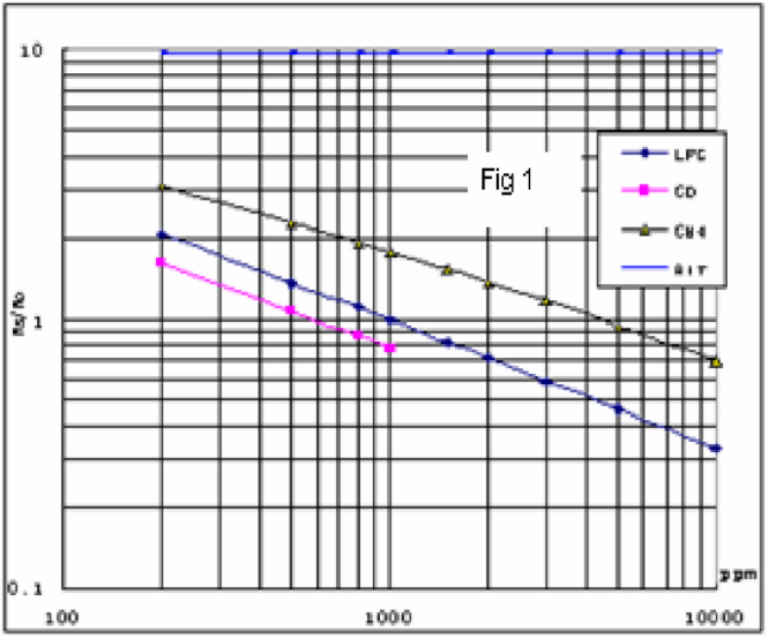
Sensor MQ6 memiliki sensitivitas terhadap LPG, H_2, CH_4, CO, dan alkohol. *Datasheet* sensor MQ6 dapat dilihat pada Gambar 2.4[9].



Gambar 2.4 Datasheet Sensor MQ6

2.4.4 MQ9

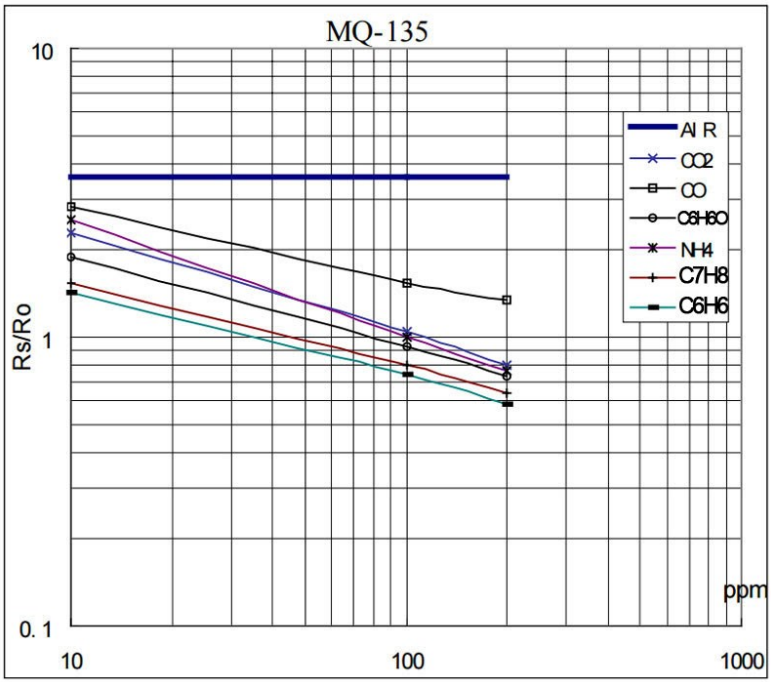
Sensor MQ9 memiliki sensitivitas terhadap LPG, CO, dan CH_4. *Datasheet* sensor MQ9 dapat dilihat pada Gambar 2.5[10].



Gambar 2.5 Datasheet Sensor MQ9

2.4.5 MQ135

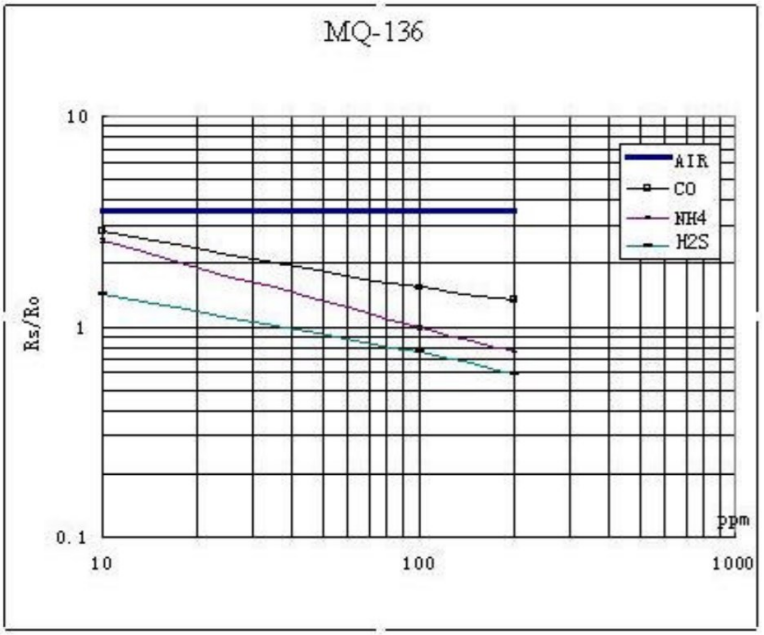
Sensor MQ135 memiliki sensitivitas terhadap CO_2, CO, C_6H_6O, NH_4, C_7H_8, dan C_6H_6. *Datasheet* sensor MQ135 dapat dilihat pada Gambar 2.6[11].



Gambar 2.6 Datasheet Sensor MQ135

2.4.6 MQ136

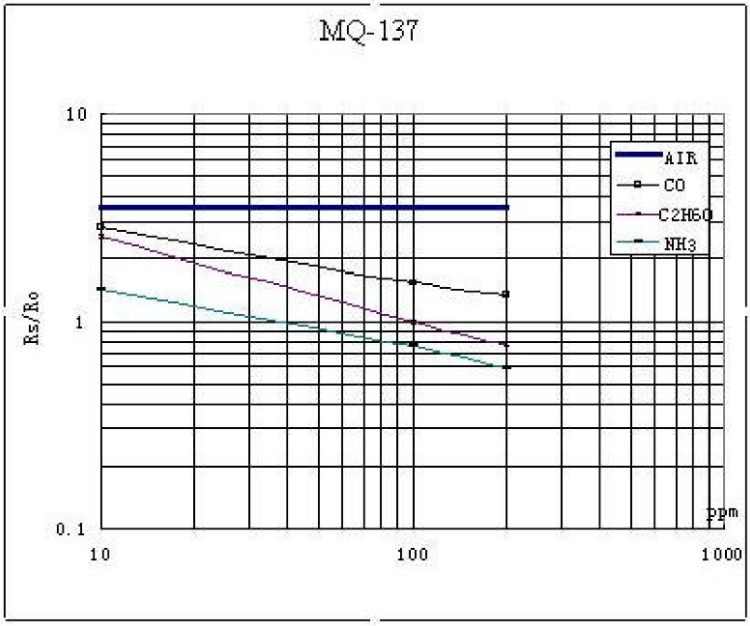
Sensor MQ136 memiliki sensitivitas terhadap CO, NH_4, dan H_2S. *Datasheet* sensor MQ136 dapat dilihat pada Gambar 2.7 [12].



Gambar 2.7 Datasheet Sensor MQ136

2.4.7 MQ137

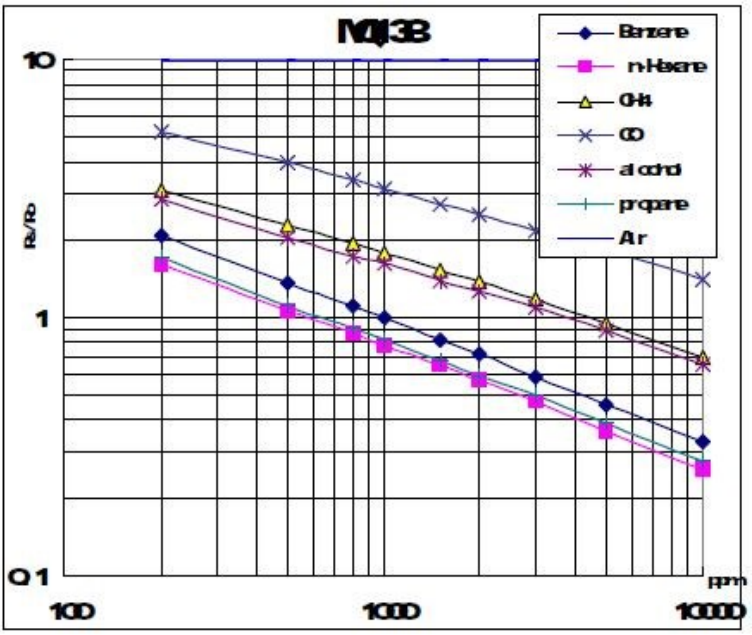
Sensor MQ137 memiliki sensitivitas terhadap CO, C_2H_6O, dan NH_3. *Datasheet* sensor MQ137 dapat dilihat pada Gambar 2.8 [13].



Gambar 2.8 Datasheet Sensor MQ137

2.4.8 MQ138

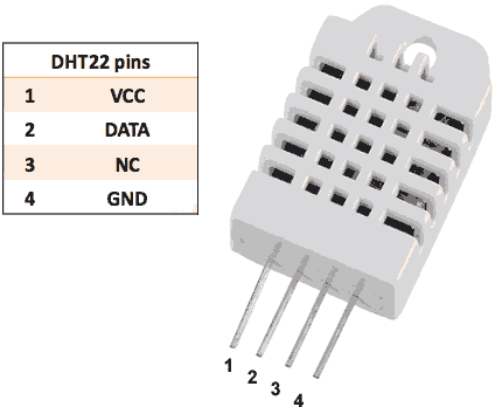
Sensor MQ138 memiliki sensitivitas terhadap CH_4, CO, alkohol, dan propana. *Datasheet* sensor MQ138 dapat dilihat pada Gambar 2.9 [8].



Gambar 2.9 Datasheet Sensor MQ138

2.4.9 DHT22

Sensor DHT22 digunakan untuk mengukur suhu dan kelembapan. Bentuk sensor DHT22 dapat dilihat pada Gambar 2.10.



Gambar 2.10 Sensor DHT22

2.5 Praproses

Praproses data merupakan tahapan awal pada pengolahan data. Tahapan ini penting karena akan meningkatkan kualitas data dan berpengaruh pada proses lain setelahnya sampai hasil akhir klasifikasi[14]. Praproses data yang dilakukan pada penelitian ini adalah menghilangkan *noise* pada sinyal.

Denoising sinyal merupakan tahapan praproses data yang penting karena akan meningkatkan performa pada tahap selanjutnya[15]. Pada langkah ini digunakan metode *discrete wavelet transform* (DWT).

DWT dilakukan pada sinyal respons sensor terhadap perubahan konsentrasi gas dalam daging. Setelah DWT dilakukan akan dihasilkan dua nilai, yaitu *lowpass* dan *highpass*. Nilai *lowpass* adalah respons asli sensor terhadap perubahan konsentrasi suatu gas yang diukur oleh sebuah sensor elektrokimia analog. Nilai *highpass* adalah resistansi sensor terhadap gas lain yang tidak responssive terhadap sensor.

Penelitian ini mengambil koefisisen aproksimasi, yaitu berasal dari nilai *lowpass*. Koefisien yang diambil pada penelitian

ini merupakan koefisien aproksimasi yang berasal dari *lowpass filter*. Metode DWT yang digunakan adalah *daubechies* 6 pada dekomposisi tingkat satu berdasar pada penelitian sebelumnya[16].

Perhitungan DWT memiliki kemiripan dengan perhitungan *Continuous Wavelet Transform* (CWT). Perbedaannya, untuk DWT, $s = 2^j$, $\tau = k^*2^j$, dengan $(j, k) \in Z^2$. Sehingga *mother wavelet* dirumuskan sebagaimana pada persamaan (2.1).

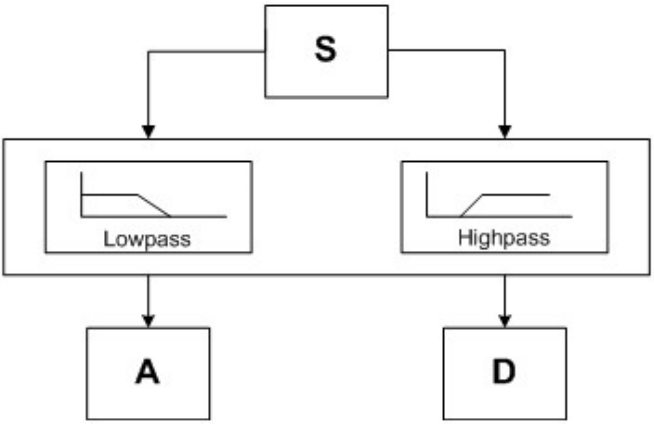
$$\boldsymbol{\psi_{s,\tau}(t)} = \frac{\mathbf{1}}{\sqrt{\mathbf{2^j}}}\boldsymbol{\psi}\left(\frac{\boldsymbol{t-k^*2^j}}{\mathbf{2^j}}\right) \quad (2.1)$$

Time series x(n) didekomposisi menghasilkan dua frekuensi yaitu *lowpass* dan *highpass* menggunakan persamaan (2.2) dan (2.3).

$$\boldsymbol{y_h[m]} = \sum_{\boldsymbol{n}} \boldsymbol{x[n]h_0[2m-n]} \quad (2.2)$$

$$\boldsymbol{y_g[m]} = \sum_{\boldsymbol{n}} \boldsymbol{x[n]g_0[2m-n]} \quad (2.3)$$

Dengan y_h dan y_g masing-masing adalah keluaran dari *highpass* dan *lowpass*. Hasil dekomposisi tingkat satu setelah dilakukan DWT dapat dilihat pada Gambar 2.11.



Gambar 2.11 Dekomposisi Tingkat Satu DWT

2.6 Ekstraksi Fitur

Nilai parameter statistik dari data yang didapat oleh sensor konstan terhadap waktu sehingga merupakan sinyal stasioner[17]. Maka nilai parameter statistik dimanfaatkan pada tahap ekstraksi fitur dengan melakukan ekstraksi fitur statistik. Parameter statistik yang dipakai adalah nilai rata-rata, minimum, maksimum, serta standar deviasi. Persamaan masing-masing fitur statistik dapat dilihat pada persamaan (2.4), (2.5), (2.6), dan (2.7).

Nilai rata-rata dihitung menggunakan persamaan berikut:

$$\boldsymbol{\mu = \frac{1}{n}\, x\, \sum_{t=0}^{n} y(t)} \qquad (2.4)$$

dengan n adalah panjang sinyal, t adalah waktu dan y(t) adalah nilai sinyal terhadap waktu.

Nilai standar deviasi sinyal y dihitung menggunakan rumus berikut:

$$\boldsymbol{\sigma = \sqrt{\frac{1}{N}\, x\, \sum_{t=0}^{N} (y(t) - \mu)^2}} \qquad (2.5)$$

dengan N adalah panjang sinyal, y(t) adalah nilai sinyal terhadap waktu (t), dan μ adalah nilai rata-rata sinyal y.

Nilai minimum S dan maksimum V dari sinyal y dapat dihitung menggunakan persamaan berikut:

$$\boldsymbol{S = min(y(t))} \qquad (2.6)$$

$$\boldsymbol{V = max(y(t))} \qquad (2.7)$$

2.7 Seleksi Fitur

Seleksi fitur merupakan tahapan pada praproses untuk memilih fitur-fitur yang relevan, yaitu yang memiliki kolerasi tinggi, sehingga dapat menghindari redundansi fitur. Hal ini

dicapai dengan menemukan sensor yang optimal dari delapan sensor semikonduktor yang dipakai dan memakai satu atribut optimal dari empat fitur statistik data sensor yang didapat pada tahap sebelumnya [18].

Penelitian ini menggunakan metode *weighting by chi-squared statistic* untuk seleksi fitur. Tingkat relevansi setiap atribut dilakukan perhitungan nilai statistik *chi-square*. Perhitungan ini didefinisikan sebagai persamaan (2.8)

$$\boldsymbol{X^2} = \sum \frac{(\boldsymbol{O}-\boldsymbol{E})^2}{\boldsymbol{E}} \quad (2.8)$$

dengan O adalah frekuensi hasil observasi dan E adalah frekuensi yang diharapkan.

2.8 Metode Klasifikasi

Pada penelitian ini terdapat empat metode klasifikasi yang diuji coba, yaitu kNN, *decision tree*, *naïve bayes*, dan *support vector machine*.

2.8.1 kNN

kNN merupakan metode klasifikasi dengan hasil dari instansi baru yang diklasifikasikan berdasarkan mayoritas banyaknya tetangga yang terdekat dari suatu kelas. Algoritma kNN didasarkan dari banyaknya *voting* terhadap sampel pelatihan untuk mendapatkan kelas prediksi dari data baru.

Penelitian ini menggunakan kNN untuk memprediksi klasifikasi dengan menggunakan jarak Canberra untuk perhitungan similaritas. Jarak Canberra vektor x terhadap vektor y dapat dihitung dengan persamaan (2.9)

$$\boldsymbol{d_{CAD}(x,y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \frac{|\boldsymbol{x_i}-\boldsymbol{y_i}|}{|\boldsymbol{x_i}|+|\boldsymbol{y_i}|} \quad (2.9)$$

dengan n, x, dan y masing-masing adalah panjang vektor atribut, vektor atribut data, dan vektor atribut data pelatihan.

2.8.2 *Decision Tree*

Decision tree merupakan metode klasifikasi yang menyerupai pohon yang bercabang-cabang. Cabang-cabang ini disebut *node*. Pada setiap *node* dilakukan pengujian atau pengambilan keputusan baru, yang mana akan menghasilkan *node-node* baru dan cabang-cabang baru sampai dihasilkan *node* terakhir disebut *leaf node*, yang merepresentasikan label kelas. Metode ini termasuk dalam algoritma *supervised learning*.

2.8.3 *Naïve Bayes*

Naïve bayes merupakan metode klasifikasi yang mengaplikasikan teorema bayes secara naif pada setiap fitur. Teorema bayes merupakan persamaan yang menggambarkan hubungan probabilitas kondisional dari kuantitas statistik. Metode ini termasuk dalam algoritma *supervised learning*.

2.8.4 *Support Vector Machine*

Support vector machine pada dasarnya merupakan algoritma klasifikasi yang didesain untuk kelas biner. Algoritma ini bekerja dengan menemukan *hyperplane* terbaik untuk membedakan antar label yang berbeda. Hal ini dilakukan dengan mengukur jarak *hyperplane* yang terdekat dengan masing-masing label. Pada kasus ini algoritma SVM dilakukan dengan pendekatan *one-versus-one* (1V1)[19].

2.9 Metode Evaluasi

Pada penelitian ini metode evaluasi yang digunakan adalah metode *training-testing* dan pengukuran untuk mengukur kinerja algoritma klasifikasi yang digunakan.

2.9.1 Metode Validasi Silang

Validasi silang adalah metode untuk mengevaluasi algoritma klasifikasi. Pertama data dibagi menjadi data pelatihan dan pengujian. Selanjutnya dilakukan beberapa kali rotasi sehingga semua data pernah menjadi data pelatihan dan data pengujian[20].

Penelitian ini menggunakan metode validasi silang sebanyak k kali lipat. Pada metode ini data pelatihan dibagi menjadi beberapa subset sebanyak k buah, dengan k adalah jumlah kelipatan data. Masing-masing subset akan dijadikan data pengujian dari hasil klasifikasi yang yang dihasilkan dari k-1 subset lainnya. Sebagai contoh untuk k=10 maka akan ada 10 kali pengujian, dengan masing-masing bagian akan menjadi data pengujian sebanyak sekali dan menjadi data pelatihan sebanyak k-1 kali. Nilai eror dari k pengujian tersebut kemudian dihitung rata-ratanya.

2.9.2 Confusion Matrix

Confusion matrix adalah tabel yang merepresentasikan secara visual jumlah aktual dan prediksi dari masing-masing kelas. Tabel ini memperlihatkan prediksi mana yang benar dan prediksi mana yang salah dari aktualnya. Dengan melihat tabel ini maka dapat diketahui dengan jelas seberapa sering suatu kelas mendapatkan label yang salah. Hal ini berguna untuk menguji performa algoritma klasifikasi.

2.9.3 Akurasi

Nilai akurasi didefinisikan sebagai persentase prediksi yang sesuai dengan aktualnya[21]. Nilai akurasi dapat dihitung menggunakan persamaan (2.10).

$$\boldsymbol{Akurasi} = \frac{\sum TP + \sum TN}{\sum Jumlah\ sampel} \qquad (2.10)$$

2.9.4 Presisi

Presisi adalah persentase banyaknya atribut terpilih yang relevan. Definisi terpilih dalam konteks ini adalah sesuatu yang dianggap benar oleh mesin terlepas dari aktualnya, yaitu *true positive*. Definisi relevan dalam konteks ini adalah sesuatu yang secara aktual adalah benar terlepas dari tebakan mesin, yaitu *true*

positive dan *false positive*. Perhitungan secara rumus dapat dilihat pada persamaan (2.11).

$$\boldsymbol{Presisi} = \frac{\sum TP}{\sum TP + \sum FP} \qquad (2.11)$$

2.9.5 Sensitivitas

Sensitivitas atau *recall* adalah persentase banyaknya atribut relevan yang terpilih. Perhitungan sensitivitas didefinisikan pada persamaan (2.12).

$$\boldsymbol{Sensitivitas} = \frac{\sum TP}{\sum TP + \sum FN} \qquad (2.12)$$

2.9.6 F_1 Score

F_1 score adalah nilai rata-rata dari presisi dan sensitivitas. *F_1 score* mencapai nilai tertinggi di 1, yaitu pada nilai presisi dan sensitivitas yang sempurna, dan mencapai nilai terendah di 0. Perhitungan sensitivitas didefinisikan pada persamaan (2.13).

$$\boldsymbol{F1\ Score} = 2 \bullet \frac{\boldsymbol{Presisi \bullet Sensitivitas}}{\boldsymbol{Presisi + Sensitivitas}} \qquad (2.133)$$

[Halaman ini sengaja dikosongkan]

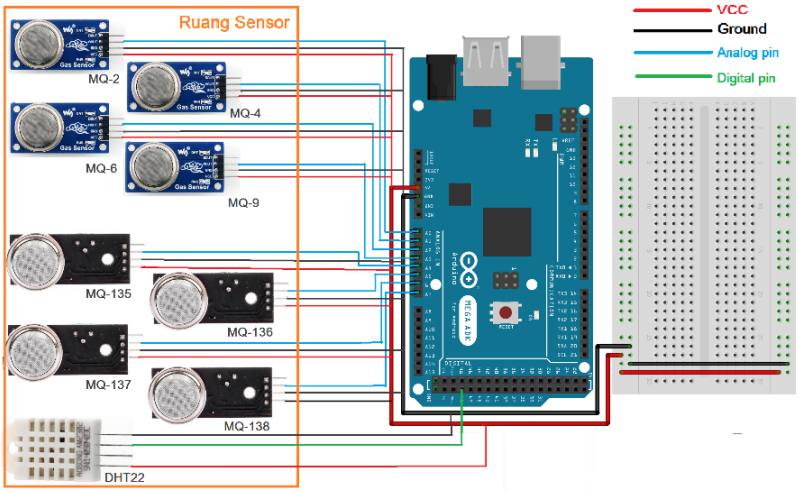
BAB III
METODOLOGI

Pada bab ini dijelaskan metodologi dari penelitian yang dilakukan. Terdapat beberapa tahapan yang harus dilakukan agar tujuan tugas akhir ini tercapai. Tahapan pertama adalah perancangan *electronic nose* yang digunakan untuk pengambilan data bau daging. Tahapan selanjutnya adalah pengumpulan data *ground truth* daging. Tahapan terakhir adalah pengujian performa *electronic nose* dalam mendeteksi bau daging.

3.1 Perangkaian *Electronic Nose*

Electronic nose adalah sebuah instrumen yang mereplikasi indra penciuman manusia untuk mendeteksi bau[22]. Data mengenai bau didapatkan melalui sensor kimia yang terdapat dalam *electronic nose*. Sensor-sensor yang terinstalasi pada *electronic nose* dalam penelitian ini adalah sensor-sensor yang memiliki selektivitas terhadap hidrogen (H_2), metana (CH_4), alkohol, karbon monoksida (CO), karbon dioksida (CO_2), hidrogen sulfat (H_2S), dan keton; digunakan untuk membedakan beberapa tingkat kemurnian daging sapi berdasarkan persentase pencampuran dengan daging babi.

Skematik *electronic nose* dapat dilihat pada Gambar 3.1. Peralatan yang diperlukan dalam merakit *electronic nose* ditunjukkan pada Tabel 3.1.



Gambar 3.1 Skematik Electronic Nose

Tabel 3.1 Daftar Komponen untuk Merakit Electronic Nose

No.	Nama Alat	Jumlah	Fungsi
1	ARDUINO MEGA 2560	1x	Mikrokontroler untuk mengambil data dari sensor
2	Kabel *Jumper Male to Female*	18x	Menghubungkan sensor dengan mikrokontroler melalui *breadboard*
3	Kabel *Jumper Male to Male*	4x	Menghubungkan *power pins* pada mikrokontroler dengan *breadboard*

4	*Breadboard* 400 *Tie Point Project Board*	1x	Media penghubung antar kabel
5	Sensor MQ2	1x	Deteksi kandungan H_2, CH_4, alkohol
6	Sensor MQ4	1x	Deteksi kandungan H_2, CH_4, alkohol
7	Sensor MQ6	1x	Deteksi kandungan H_2, CH_4, alkohol
8	Sensor MQ9	1x	Deteksi kandungan CH4, CO
9	Sensor MQ135	1x	Deteksi kandungan CO_2
10	Sensor MQ136	1x	Deteksi kandungan H_2S
11	Sensor MQ137	1x	Deteksi kandungan NH_3
12	Sensor MQ138	1x	Deteksi kandungan keton
13	Sensor DHT22	1x	Pengukuran suhu dan kelembapan
14	Kabel USB	1x	Menghubungkan komputer dan mikrokontroler
15	Ruang Sensor	1x	Penampung gas daging sekaligus tempat sensor

Pin pada sensor perlu dihubungkan dengan Arduino agar berfungsi. Pada masing-masing sensor terdapat pin arus masuk, pin arus keluar, dan pin data. Pin arus masuk dan pin arus keluar masing-masing dihubungkan dengan pin Vin dan pin GND pada Arduino untuk mendapat tenaga. Sedangkan pin data dihubungkan dengan pin analog atau digital pada Arduino tergantung pada jenis sensor untuk pengiriman data. Kabel *Jumper* adalah media yang digunakan untuk menghubungkan antara pin pada sensor dengan pin pada Arduino. Selain itu *breadboard* juga diperlukan untuk mempermudah perangkaian koneksi terutama untuk arus masuk dan keluar dari Arduino dan sensor. Pada Tabel 3.2 dapat dilihat sambungan antara pin pada *breadboard ke mikrokontroler*. Pada Tabel 3.3 dapat dilihat sambungan pin antara mikrokontroler dan sensor gas.

Tabel 3.2 Sambungan Pin pada Breadboard ke Mikrokontroler

BreadBoard	**Mikrokontroler**
+	5V
-	GND

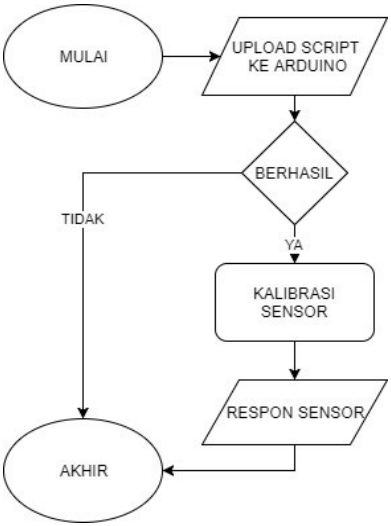
Tabel 3.3 Sambungan Pin pada Sensor ke Mikrokontroler

Sensor Gas	**Mikrokontorler**
VCC	Vin
GND	GND
AO MQ2	A0

AO MQ4	A1
AO MQ6	A2
AO MQ9	A3
AO MQ135	A4
AO MQ136	A5
AO MQ137	A6
AO MQ138	A7
Pin pada Sensor Suhu-Kelembapan	**Pin pada Mikrokontroler**
VCC	Vin
GND	GND
DO	DHT22

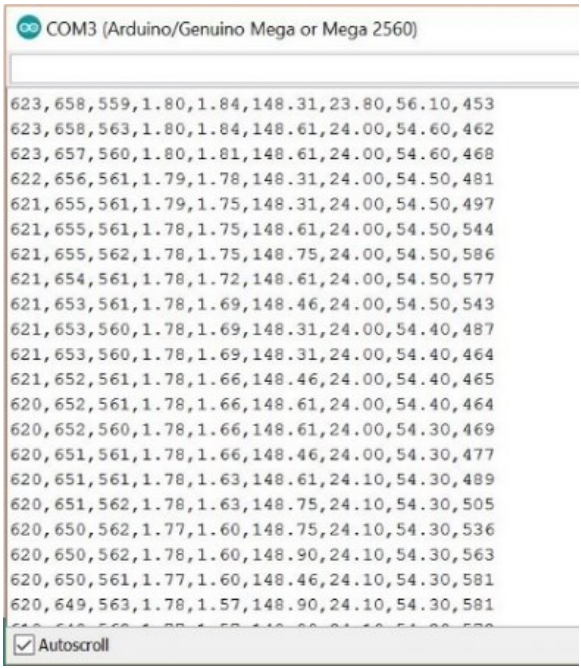
3.2 Pengujian Performa *Electronic Nose*

Pengujian performa *electronic nose* dilakukan untuk memastikan bereaksi atau tidaknya susunan sensor pada *electronic noes* terhadap perubahan konsentrasi gas, suhu, dan kelembapan pada sampel daging. Pada Gambar 3.2 dapat dilihat *flowchart* uji performa.



Gambar 3.2 Flowchart Uji Performa Electronic Nose

Arduino diprogram dengan menggunakan Bahasa Arduino dengan *Baud Rate* 9600. Arduino diprogram sebatas untuk mencetak data yang ditangkap setiap sensor pada *serial monitor* Arduino IDE. Pada akhirnya pencatatan akan diotomasikan menggunakan program Python dalam pengaturan waktu dan pembentukan *file*.



Gambar 3.3 Keluaran Serial Monitor pada Arduino IDE

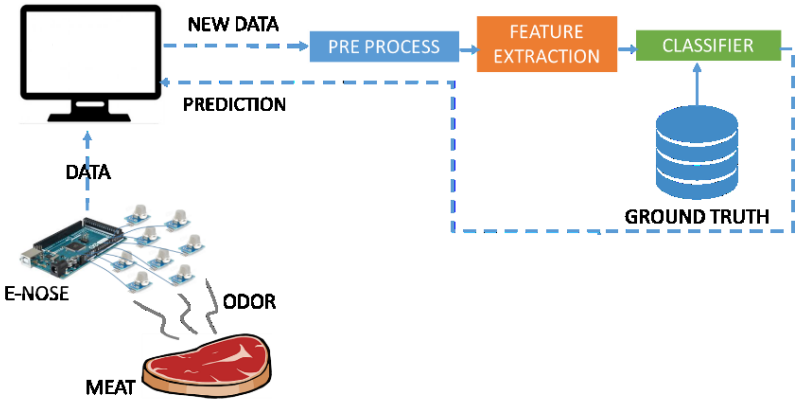
Setelah dilihat bahwa setiap sensor memberikan respons secara stabil, dapat dinyatakan *electronic nose* siap dipakai.

3.3 Pengumpulan Data

Pengumpulan data dilakukan untuk mendapatkan data pelatihan. Data ini juga akan digunakan sebagai data pengujian menggunakan validasi silang. Pada Gambar 3.4 dapat dilihat tata cara pengambilan data daging. Daging yang digunakan sebagai *ground truth* adalah daging sapi segar dan daging babi segar dalam keadaan mentah dalam keadaan tercincang halus menggunakan *blender*. Data yang terkumpul sejumlah 420 data *ground truth*

dibagi menjadi tujuh kelas dengan peperincian masing-masing 60 data sebagai berikut:

1. S0% : 100% daging babi (daging babi murni),
2. S10% : Campuran 10% daging sapi 90% daging babi,
3. S25% : Campuran 25% daging sapi 75% daging babi,
4. S50% : Campuran 50% daging sapi 50% daging babi,
5. S75% : Campuran 75% daging sapi 25% daging babi,
6. S90% : Campuran 90% daging sapi 10% daging babi,
7. S100% : 100% daging sapi (daging sapi murni).



Gambar 3.4 Flow Pengambilan Data Daging

Terdapat tujuh langkah dalam pengambilan data sampel daging:

1. Menyiapkan sampel daging dan perangkat *electronic nose*.
2. Daging dimasukkan ke ruang sampel.
3. Program pencatatan dinyalakan.
4. Menunggu dua menit.
5. *File* csv pencatatan bau daging terbentuk.
6. Sampel daging dikeluarkan.
7. Udara pada ruang sampel disterilkan menggunakan kipas angin.

BAB IV
ANALISIS DAN PERANCANGAN SISTEM

Bab ini terdiri dari dua subbab yaitu analisis sistem yang menjelaskan ide dasar pembuatan aplikasi dan perancangan sistem yang menjelaskan dasar-dasar pembuatan sistem. Subbab analisis meliputi masalah yang melatarbelakangi, analisis perbedaan respons sensor terhadap daging, analisis kebutuhan, dan deskripsi umum sistem. Pada subbab perancangan sistem dibahas mengenai perancangan proses serta kasus penggunaan.

4.1 Analisis

Subbab ini membahas mengenai analisis permasalahan, analisis perbedaan respons sensor terhadap daging, analisis kebutuhan, serta deskripsi umum sistem baik masukan, proses, maupun keluaran serta diagram kasus.

4.1.1 Analisis Permasalahan

Daging merupakan salah satu konsumsi utama manusia. Adanya praktik pencampuran daging dalam jual beli merupakan masalah yang perlu diatasi. Terutama pencampuran daging sapi dengan daging babi yang dalam Islam merupakan makanan haram. Oleh karena itu diperlukan alat yang bisa mendeteksi kemurnian daging sapi secara praktis dan instan. Penelitian ini dilakukan untuk merancang sistem yang mudah digunakan dan rendah biaya yang dapat mendeteksi kemurnian daging sapi dari daging babi.

4.1.2 Analisis Perbedaan Respons Sensor terhadap Daging

Dari setiap sensor yang digunakan akan dihasilkan beberapa atribut dari hasil proses ekstraksi fitur, lalu hanya diambil yang paling relevan. Persamaan (2.13) digunakan untuk mengukur nilai

statistik *chi-square*. Peringkat atribut yang didapatkan dari perhitungan statistik *chi-square* dapat dilihat pada Tabel 4.1.

Tabel 4.1 Nilai Relevansi Atribut Berdasarkan Perhitungan Chi-Square

Attributes	***Chi-Squared Statistic Weight Value***
Mean MQ135	746,137
Mean MQ136	662,983
Mean MQ4	654,513
Mean MQ9	594,602
Range MQ137	564,374
Mean MQ6	538,663
Mean MQ2	483,643
Mean MQ138	443,722

Vektor atribut *ground truth* sebelum dilakukan seleksi fitur berjumlah 56 atribut yaitu tujuh fitur statistik dikali delapan respons sensor gas. Proses seleksi fitur kemudian melakukan perhitungan terhadap masing-masing atribut tersebut dan didapatkan delapan atribut paling relevan.

4.1.3 Analisis Kebutuhan

Analisis kebutuhan meliputi cakupan kebutuhan fungsional. Berikut dapat dilihat pada Tabel 4.2 daftar kebutuhan fungsional dari sistem yang dibangun.

Tabel 4.2 Daftar Kebutuhan Fungsional Sistem

Kode Kebutuhan	**Kebutuhan Fungsional**	**Deskripsi**

F-01	Sistem memungkinkan pengguna melihat cara menggunakan program	Pengguna dapat melihat petunjuk berupa gambar mengenai cara menggunakan program
F-02	Sistem memungkinkan pengguna memuat file dataset yang akan digunakan	Pengguna dapat memuat dan menggunakan file dataset sendiri dalam bentuk .csv
F-03	Sistem memungkinkan pengguna melakukan uji coba klasifikasi dalam berbagai metode serta mengatur parameternya	Pengguna dapat melakukan uji coba klasifikasi berdasarkan metode dan parameternya serta melihat laporan hasilnya

4.1.4 Deskripsi Umum Sistem

Sistem ini mengimplementasikan aplikasi berbasis *desktop*. Subbab ini menjelaskan deskripsi umum sistem mulai dari masukan, proses, dan keluaran.

4.1.4.1 Masukan

Masukan pada sistem ini berupa *file* data set. *File* data set yang digunakan harus memiliki ekstensi (.csv), menggunakan delimiter (,), tanpa *header* yang berupa penamaan masing-masing kolom, serta peletakkan kelas harus berada pada paling ujung kanan baris (contoh: att1,att2,att3,class).

4.1.4.2 Proses

Terdapat dua sisi proses pada sistem ini, yaitu proses pada sisi *machine learning* dan proses pada sisi aplikasi. Pada sisi *machine learning*, digunakan *library* scikit-learn pada Python untuk menangani proses dalam hal praproses, ekstraksi fitur, dan klasifikasi data. Pada sisi *aplikasi desktop*, digunakan library PyQt pada Python untuk menangani proses dalam hal menerima masukan berupa dataset dan menampilkan hasil klasifikasi berdasarkan parameter yang dapat diatur.

4.1.4.3 Keluaran

Sistem mengeluarkan keluaran adalah performa dari uji coba klasifikasi yang dilakukan, berupa *confusion matrix* beserta nilai presisi, sensitivitas, *f1 score*, dan akurasi.

4.2 Perancangan Sistem

Tahap ini membahas rancangan alur proses yang digunakan sebagai acuan dalam pembangunan sistem.

4.2.1 Perancangan Proses

Tahap ini membahas rancangan alur proses yang digunakan sebagai acuan dalam pembangunan sistem. Alur proses dimulai dari masukan, proses, dan keluaran.

4.2.1.1 Masukan

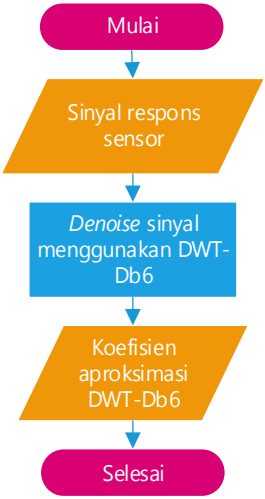
Pencatatan data yang ditangkap *electronic nose* dilakukan menggunakan program Python untuk mengotomasi pengaturan waktu dan pencatatan nama *file*. Data yang didapat dari pencatatan ini adalah berupa kumpulan angka-angka dari berbagai sensor yang digunakan yang direkam seiring waktu. Bila setiap perubahan angka ini ditarik garis lurus akan terbentuk grafik satu dimensi yang dalam ilmu *machine learning* disebut data sinyal. Diagram alur dari proses ini dapat dilihat pada Gambar 4.1.

Gambar 4.1 Diagram Alur Masukan

4.2.1.2 Pra Proses

Data yang telah didapatkan dari tahap *input* masih memiliki *noise*. *Noise* merupakan noda-noda ketidakstabilan data yang menyebabkan bentuk sinyal yang tidak mulus. Data-data ini tidak relevan dan mengganggu performa *machine learning*.

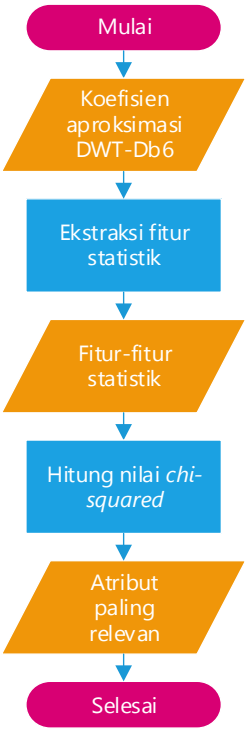
Untuk menghilangkan *noise* ini dilakukan transformasi *wavelet* diskrit. Metode yang digunakan pada penelitian ini adalah *mother wavelet transform daubechies 6*. Dari proses ini didapatkan koefisien aproksimasi, yaitu sinyal mulus dengan frekuensi yang lebih rendah. Pada Gambar 4.2 dapat dilihat diagram alur dari tahap ini.



Gambar 4.2 Diagram Alur Praproses

4.2.1.3 Ekstraksi Fitur

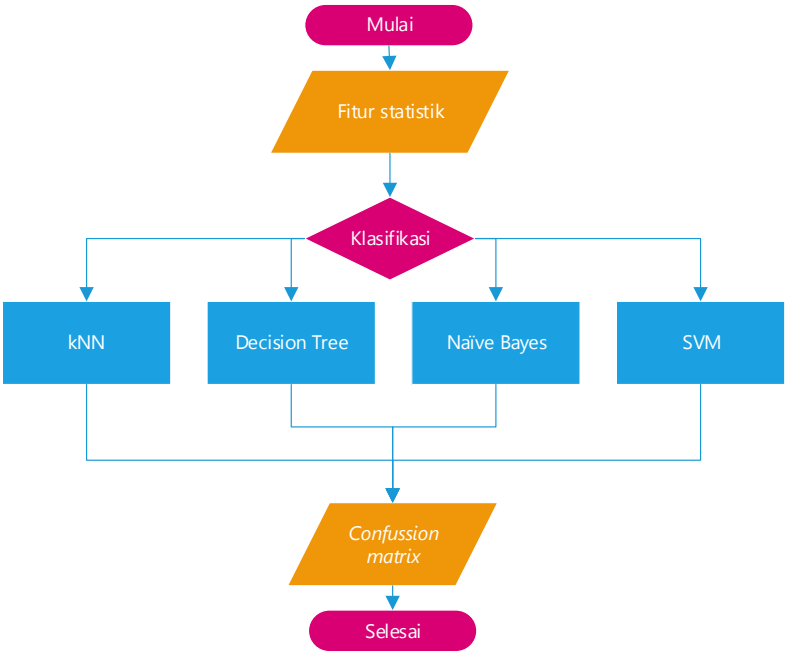
Ekstraksi fitur dilakukan terhadap koefisien aproksimasi yang diperoleh dari proses denoising menggunakan metode DWT *daubechies 6*. Ektraksi fitur menghasilkan fitur-fitur statistik yang kemudian diseleksi berdasarkan nilai *chi-squared* menghasilkan atribut paling relevan. Tahap ekstraksi fiture ini bisa dilihat pada diagram alur proses sebagaimana ditunjukkan pada Gambar 4.3**.**



Gambar 4.3 Diagram Alur Ekstraksi Fitur

4.2.1.4 Klasifikasi

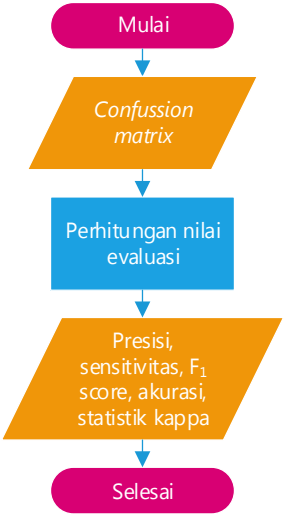
Atribut-atribut yang telah didapat dari proses ekstraksi fitur selanjutnya digunakan untuk klasifikasi. Terdapat empat metode klasifikasi yang diuji coba pada kasus ini, yaitu kNN, *decision tree*, *naïve bayes*, dan *support vector classification*. Setelah menerapkan algoritma klasifikasi maka akan didapatkan hasil berupa tabel *confusion matrix*. Diagram alur dapat dilihat pada Gambar 4.4.



Gambar 4.4 Diagram Alur Klasifikasi

4.2.1.5 Evaluasi

Tahap ini merupakan tahap terakhir, yang digunakan untuk mengevaluasi performa metode klasifikasi. Dari hasil klasifikasi yang didapatkan pada tahap sebelumnya berupa *confusion matrix* dilakukan beberapa perhitungan. Perhitungan tersebut antara lain adalah presisi, sensitivitas, F_1 *score*, dan akurasi. Pada Gambar 4.5 dapat dilihat diagram alur evaluasi**.**



Gambar 4.5 Diagram Alur Evaluasi

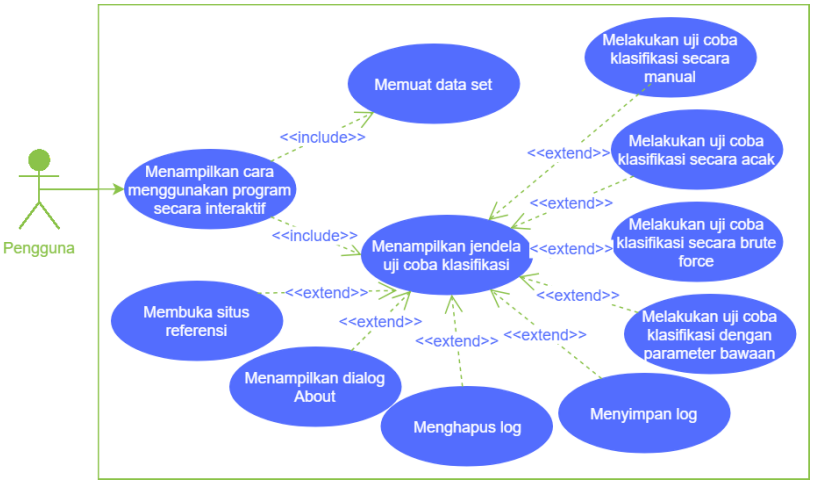
4.2.2 Diagram Kasus

Tabel kasus penggunaan disusun berdasarkan analisis kebutuhan fungsional yang telah dipaparkan pada subbab 4.1.3. Penjelasan lebih lanjut mengenai kasus penggunaan dapat dilihat pada Tabel 4.3 serta diagram kasus penggunaan dapat dilihat pada Gambar 4.6.

Tabel 4.3 Daftar Kasus Penggunaan

Kode Kasus Penggunaan	Nama	Aktor
UC-1	Menampilkan cara menggunakan program secara interaktif	Pengguna
UC-2	Memuat dataset	Pengguna

UC-3	Menampilkan jendela uji coba klasifikasi	Pengguna
UC-4	Melakukan uji coba klasifikasi secara manual	Pengguna
UC-5	Melakukan uji coba klasifikasi dengan parameter acak	Pengguna
UC-6	Melakukan uji coba klasifikasi secara *brute force*	Pengguna
UC-7	Melakukan uji coba klasifikasi dengan parameter bawaan	Pengguna
UC-8	Menghapus *log*	Pengguna
UC-9	Menyimpan *log*	Pengguna
UC-10	Menampilkan dialog tentang program	Pengguna
UC-11	Menampilkan situs mengenai penjelasan parameter	Pengguna



Gambar 4.6 Diagram Kasus Penggunaan

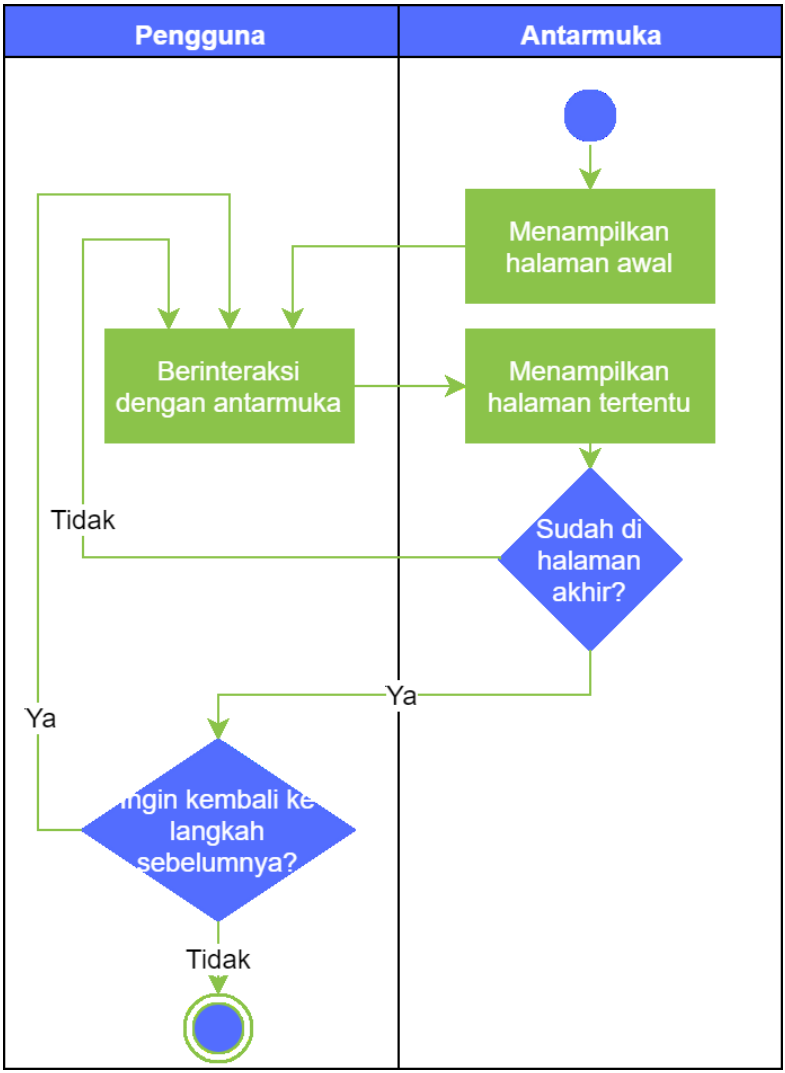
4.2.2.1 Menampilkan Cara Menggunakan Program Secara Interaktif (UC-1)

Pada kasus penggunaan ini, pengguna dapat menjalankan program sambil dipandu secara interaktif. Spesifikasi kasus penggunaan dapat dilihat Pada Tabel 4.4 dan diagram aktivitas pada Gambar 4.7.

Tabel 4.4 Spesifikasi Kasus Penggunaan UC-1

Kode	**UC-1**
Nama	Menampilkan cara menggunakan program secara interaktif
Aktor	Pengguna
Deskripsi	Pengguna menjalankan program sambil dipandu secara interaktif
Kondisi Awal	Sistem menampilkan halaman awal
Kondisi Akhir	Sistem menampilkan halaman terakhir

Alur Kejadian Normal
1. Sistem menampilkan halaman judul **2. Pengguna menekan halaman pada bagian mana saja** **3. Sistem menampilkan halaman tertentu** **4. Pengguna berinteraksi dengan antarmuka mengikuti alur program** **5. Sistem menampilkan halaman terakhir**
Alur Kejadian Alternatif
5.A. Sistem kembali ke halaman sebelumnya



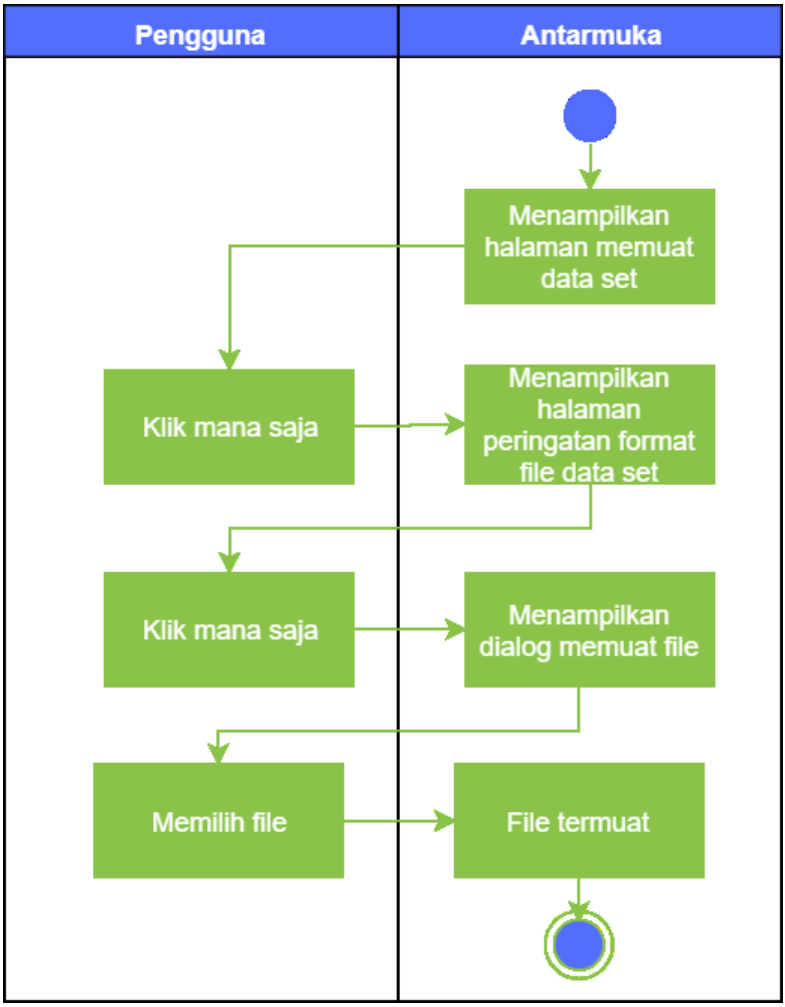
Gambar 4.7 Diagram Aktivitas UC-1

4.2.2.2 Memuat Dataset (UC-2)

Pada kasus penggunaan ini, pengguna dapat Memuat Dataset yag dipilih. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.5 dan diagram aktivitas pada Gambar 4.8.

Tabel 4.5 Spesifikasi Kasus Penggunaan UC-2

Kode	**UC-2**
Nama	Memuat dataset
Aktor	Pengguna
Deskripsi	Pengguna dapat memuat file dataset berbentuk (.csv) yang akan dipakai untuk uji coba klasifikasi
Kondisi Awal	Sistem tidak memiliki data dataset. Sistem tidak menampilkan opsi metode klasifikasi untuk dapat lanjut langkah selanjutnya.
Kondisi Akhir	Sistem memiliki data dataset. Sistem menampilkan opsi metode klasifikasi.
Alur Kejadian Normal	
1. **Sistem menampilkan dialog memuat *data set*** 2. **Pengguna memilih *file*** 3. **Sistem menutup dialog memuat *data set***	
Alur Kejadian Alternatif	
-	



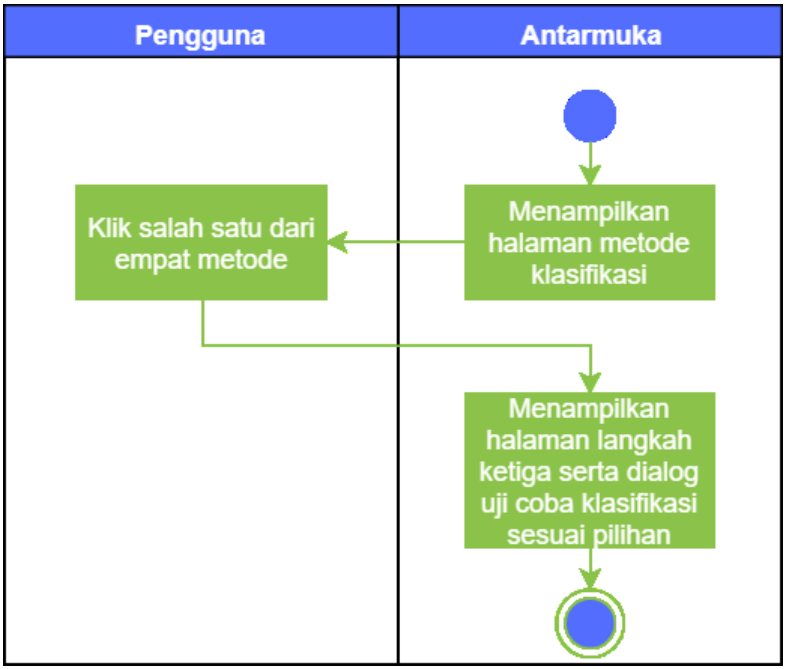
Gambar 4.8 Diagram Aktivitas UC-2

4.2.2.3 Menampilkan Jendela Uji Coba Klasifikasi (UC-3)

Pada kasus penggunaan ini, pengguna dapat memilih metode klasifikasi yang akan diuji coba dan menampilkan dialog uji coba klasifikasi sesuai metode yang dipilih. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.6 dan diagram aktivitas pada Gambar 4.9.

Tabel 4.6 Spesifikasi Kasus Penggunaan UC-3

Kode	**UC-3**
Nama	Menampilkan jendela uji coba klasifikasi
Aktor	Pengguna
Deskripsi	Pengguna dapat memilih metode klasifikasi yang akan diuji coba dan menampilkan dialog uji coba klasifikasi sesuai metode yang dipilih
Kondisi Awal	Sistem berada pada keadaan menampilkan halaman memilih metode klasifikasi
Kondisi Akhir	Sistem menampilkan dialog uji coba klasifikasi
Alur Kejadian Normal	
1. **Sistem berada pada keadaan menampilkan halaman memilih metode klasifikasi** 2. **Pengguna menekan tombol metode klasifikasi yang dipilih** 3. **Sistem menampilkan halaman langkah ketiga** serta **dialog uji coba klasifikasi**	
Alur Kejadian Alternatif	
-	



Gambar 4.9 Diagram Aktivitas UC-3

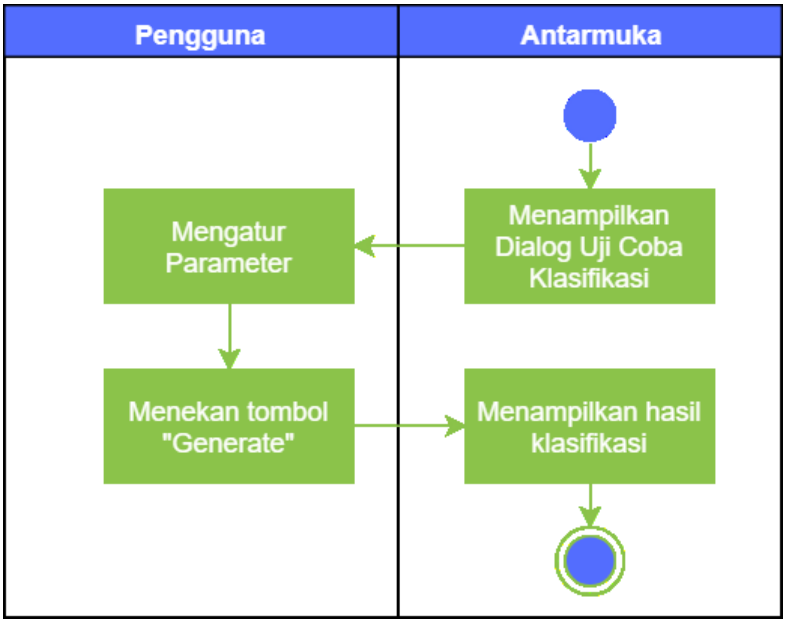
4.2.2.4 Melakukan Uji Coba Klasifikasi Secara Manual (UC-4)

Pada kasus penggunaan ini, pengguna dapat melakukan uji coba klasifikasi dengan mengatur parameter-parameter yang tersedia secara manual. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.7 dan diagram aktivitas dapat dilihat pada Gambar 4.10.

Tabel 4.7 Spesifikasi Kasus Penggunaan UC-4

Kode	**UC-4**
Nama	Melakukan uji coba klasifikasi secara manual
Aktor	Pengguna

Deskripsi	Pengguna melakukan uji coba klasifikasi dengan mengatur parameter-parameter yang tersedia secara manual
Kondisi Awal	Sistem berada pada dialog uji coba klasifikasi
Kondisi Akhir	Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks
Alur Kejadian Normal	
1. **Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi** 2. **Pengguna mengatur parameter** 3. **Pengguna menekan tombol "Generate"** 4. **Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks**	
Alur Kejadian Alternatif	
-.	



Gambar 4.10 Diagram Aktivitas UC-4

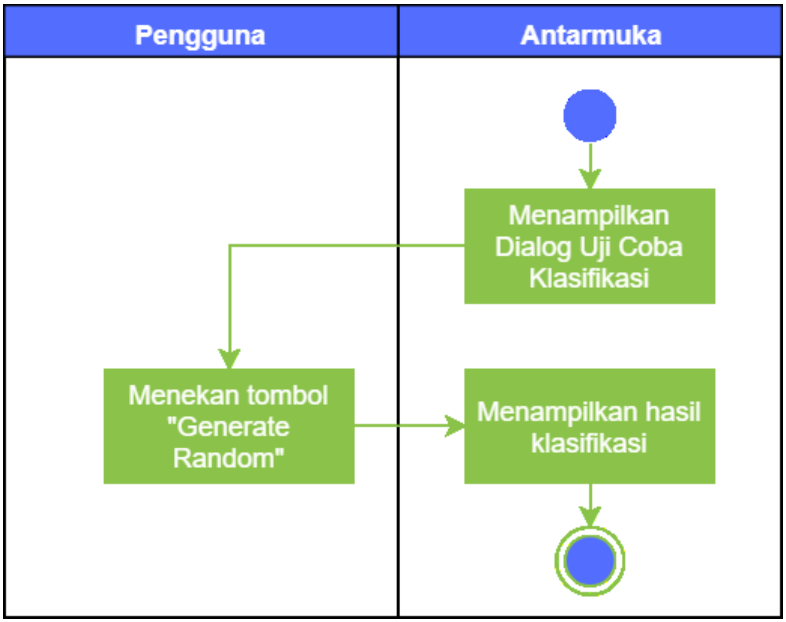
4.2.2.5 Melakukan Uji Coba Klasifikasi Dengan Parameter Acak (UC-5)

Pada kasus penggunaan ini, pengguna melakukan uji coba klasifikasi dengan parameter acak. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.8 dan diagram aktivitas dapat dilihat pada Gambar 4.11.

Tabel 4.8 Spesifikasi Kasus Penggunaan UC-5

Kode	**UC-5**
Nama	Melakukan uji coba klasifikasi dengan parameter acak
Aktor	Pengguna

Deskripsi	Pengguna melakukan uji coba klasifikasi dengan parameter acak
Kondisi Awal	Sistem berada pada dialog uji coba klasifikasi
Kondisi Akhir	Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks
Alur Kejadian Normal	
5. **Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi** 6. **Pengguna menekan tombol "Generate Brute Force"** 7. **Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks secara *brute force* satu per satu sampai semua kemungkinan parameter telah terpakai**	
Alur Kejadian Alternatif	
-	



Gambar 4.11 Diagram Aktivitas UC-5

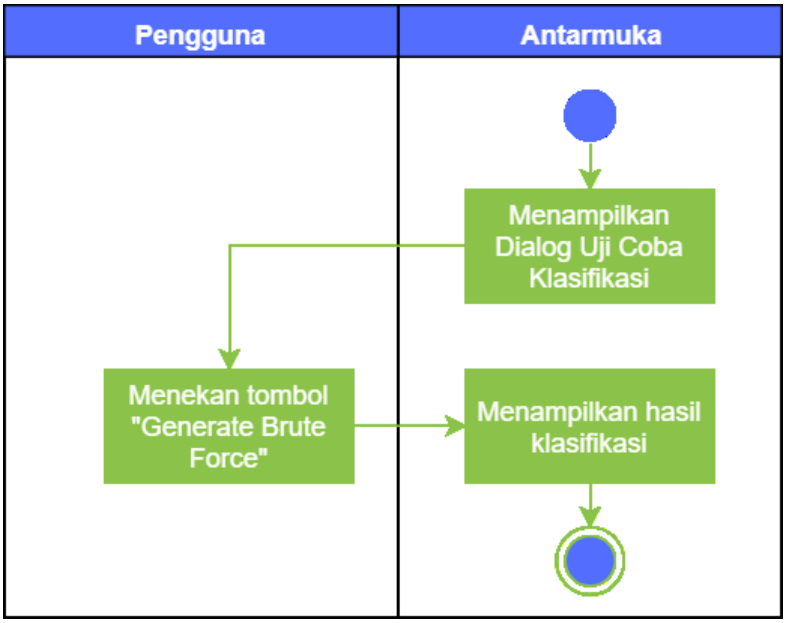
4.2.2.6 Melakukan Uji Coba Klasifikasi Secara *Brute Force* (UC-4)

Pada kasus penggunaan ini, pengguna dapat menghapus dari system data daging yang dipilih. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.9 dan diagram aktivitas dapat dilihat pada Gambar 4.12.

Tabel 4.9 Spesifikasi Kasus Penggunaan UC-6

Kode	**UC-6**
Nama	Melakukan uji coba klasifikasi secara *brute force*
Aktor	Pengguna

Deskripsi	Pengguna dapat melakukan uji coba klasifikasi
Kondisi Awal	Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi
Kondisi Akhir	Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks
Alur Kejadian Normal	
1. **Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi** 2. **Pengguna menekan tombol "Generate Brute Force"** 3. **Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks secara *brute force* satu per satu sampai semua kemungkinan parameter telah terpakai**	
Alur Kejadian Alternatif	
-	



Gambar 4.12 Diagram Aktivitas UC-6

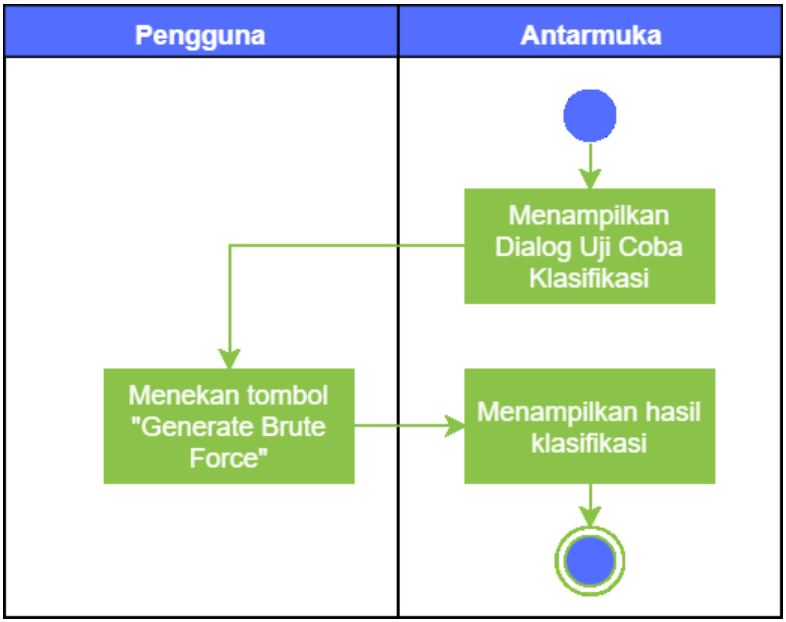
4.2.2.7 Melakukan Uji Coba Klasifikasi Dengan Parameter Bawaan (UC-7)

Pada kasus penggunaan ini, pengguna melakukan uji coba klasifikasi dengan parameter bawaan. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.10 dan diagram aktivitas dapat dilihat pada Gambar 4.12.

Tabel 4.10 Spesifikasi Kasus Penggunaan UC-7

Kode	**UC-7**

Nama	Melakukan uji coba klasifikasi dengan parameter bawaan
Aktor	Pengguna
Deskripsi	Pengguna dapat melakukan uji coba klasifikasi dengan parameter bawaan
Kondisi Awal	Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi
Kondisi Akhir	Sistem menampilkan hasil uji coba klasifikasi dengan parameter bawaan dalam bentuk teks
Alur Kejadian Normal	
1. **Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi** 2. **Pengguna menekan tombol "Generate Default"** 3. **Sistem menampilkan hasil uji coba klasifikasi dalam bentuk teks dengan parameter bawaan**	
Alur Kejadian Alternatif	
Tidak	



Gambar 4.13 Diagram Aktivitas UC-7

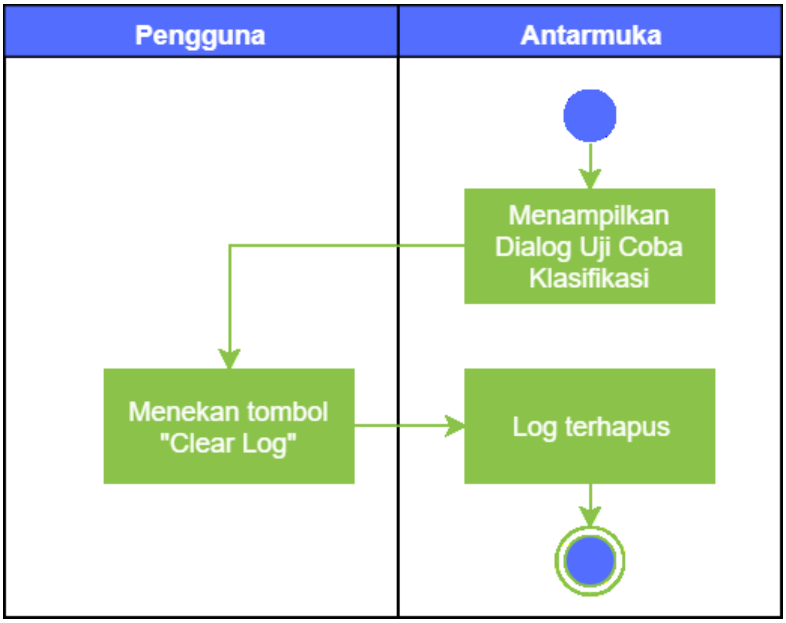
4.2.2.8 Menghapus Catatan Riwayat Uji Coba Klasifikasi (UC-8)

Pada kasus penggunaan ini, pengguna dapat menghapus catatan riwayat uji coba klasifikasi. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.11 dan diagram aktivitas dapat dilihat pada Gambar 4.14.

Tabel 4.11 Spesifikasi Kasus Penggunaan UC-8

Kode	**UC-8**
Nama	Menghapus catatan riwayat uji coba klasifikasi
Aktor	Pengguna

Deskripsi	Pengguna dapat menghapus riwayat hasil uji coba klasifikasi yang telah ditampilkan pada dialog uji coba klasifikasi
Kondisi Awal	Sistem berada pada kondisi menampilkan dialog uji coba klasifikasi dengan catatan riwayat yang belum terhapus jika ada
Kondisi Akhir	Teks pada catatan riwayat terhapus
Alur Kejadian Normal	
1. **Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi dengan catatan riwayat yang belum terhapus jika ada** 2. **Pengguna menekan tombol "Clear Log"** 3. **Sistem menghapus catatan riwayat uji coba klasifikasi**	
Alur Kejadian Alternatif	
-	



Gambar 4.14 Diagram Aktivitas UC-8

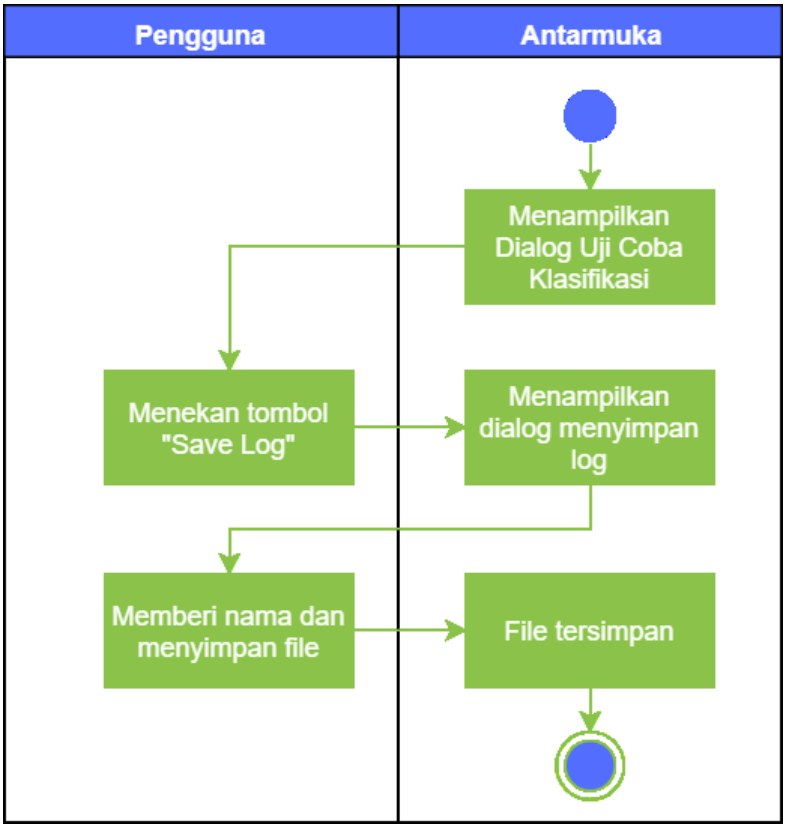
4.2.2.9 Menyimpan Catatan Riwayat Uji Coba Klasifikasi (UC-9)

Pada kasus penggunaan ini, pengguna dapat menyimpan catatan riwayat dari hasil uji coba klasifikasi. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.12 dan diagram aktivitas dapat dilihat pada Gambar 4.15.

Tabel 4.12 Spesifikasi Kasus Penggunaan UC-9

Kode	**UC-9**
Nama	Menyimpan catatan riwayat uji coba klasifikasi
Aktor	Pengguna
Deskripsi	Pengguna dapat meyimpan catatan riwayat uji coba klasifikasi dalam bentuk (.log)

Kondisi Awal	Sistem berada pada kondisi menampilkan dialog uji coba klasifikasi
Kondisi Akhir	Sistem menyimpan catatan riwayat uji coba klasifikasi dalam bentuk (.log)
Alur Kejadian Normal	
1. **Sistem berada pada keadaan menampilkan dialog uji coba klasifikasi** 2. **Pengguna menekan tombol "Save Log"** 3. **Sistem menampilkan dialog menyimpan *log*** 4. **Pengguna memberi nama dan menyimpan *file*** 5. ***File* (.log) tersimpan** 6. **Sistem menutup dialog menyimpan *log***	
Alur Kejadian Alternatif	
-	



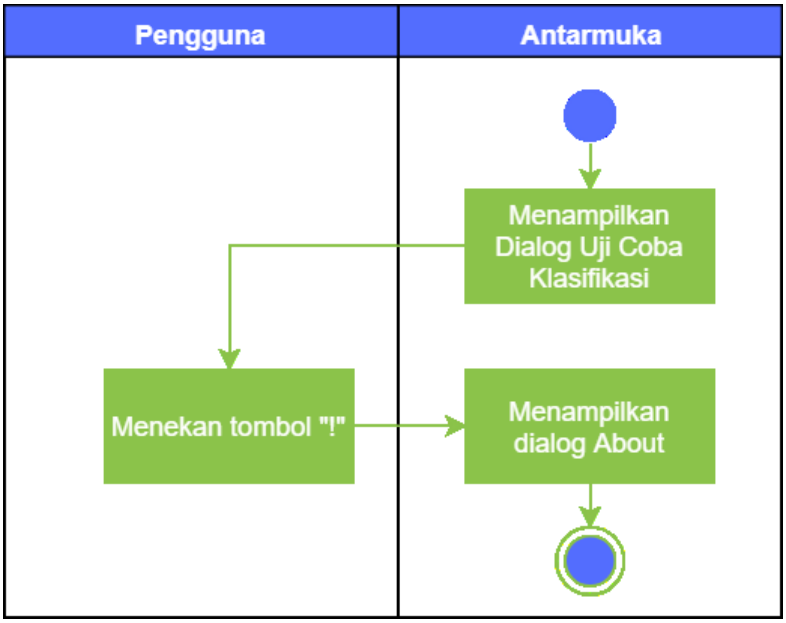
Gambar 4.15 Diagram Aktivitas UC-9

4.2.2.10 Menampilkan Dialog Tentang Program (UC-10)

Pada kasus penggunaan ini, pengguna dapat melihat dialog tentang program. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.13 dan diagram aktivitas dapat dilihat pada Gambar 4.15.

Tabel 4.13 Spesifikasi Kasus Penggunaan UC-10

Kode	**UC-10**
Nama	Menampilkan dialog tentang program
Aktor	Pengguna
Deskripsi	Pengguna dapat menampilkan dialog tentang program
Kondisi Awal	Sistem
Kondisi Akhir	Data daging terhapus dari basis data
Alur Kejadian Normal	
1. **Sistem berada pada dialog uji coba klasifikasi** 2. **Pengguna menekan tombol "About"** 3. **Sistem menampilkan dialog tentang program**	
Alur Kejadian Alternatif	
-	



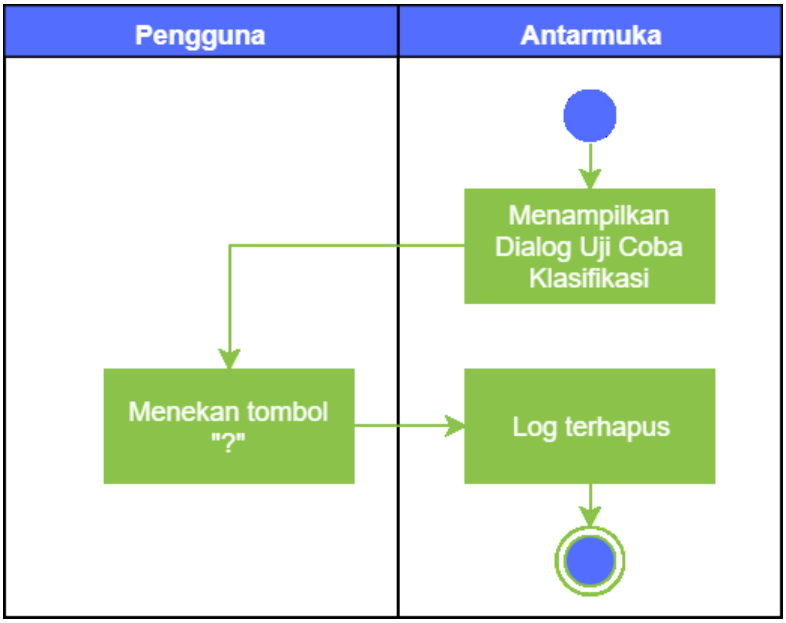
Gambar 4.16 Diagram Aktivitas UC-10

4.2.2.11 Menampilkan Situs Mengenai Penjelasan Parameter (UC-11)

Pada kasus penggunaan ini, pengguna dapat membuka situs yang berisi penjelasan mengenai metode klasifikasi yang diuji coba beserta rincian parameternya. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 4.14 dan diagram aktivitas dapat dilihat pada Gambar 4.17.

Tabel 4.14 Spesifikasi Kasus Penggunaan UC-11

Kode	**UC-11**
Nama	Menampilkan situs mengenai penjelasan parameter
Aktor	Pengguna
Deskripsi	Pengguna dapat membuka situs yang berisi penjelasan mengenai metode klasifikasi yang digunakan beserta rincian parameternya
Kondisi Awal	Sistem berada pada dialog uji coba klasifikasi
Kondisi Akhir	Sistem menampilkan situs mengenai penjelasan metode klasifikasi beserta rincian parameternya
Alur Kejadian Normal	
1. **Sistem berada pada dialog uji coba klasifikasi** 2. **Pengguna menekan tombol "Help"** 3. **Sistem menampilkan situs yang berisi penjelasan mengenai metode klasifikasi yang digunakann beserta rincian parameternya**	
Alur Kejadian Alternatif	
-	



Gambar 4.17 Diagram Aktivitas UC-11

[Halaman ini sengaja dikosongkan]

BAB V
IMPLEMENTASI

Bab ini merupakan implementasi dari bahasan pada bab sebelumnya yaitu Bab IV Analisis dan Perancangan Sistem.

5.1 Lingkungan Implementasi

Implementasi dilakukan pada lingkungan yang sesuai dengan rancangan, yaitu menggunakan perangkat-perangkat pendukung berupa perangkat keras dan perangkat lunak. Perangkat pendukung yang digunakan diuraikan pada subbab berikut.

5.1.1 Lingkungan Implementasi Perangkat Keras

Dalam implementasi, perangkat keras yang digunakan adalah:

Jenis	: Laptop
Tipe	: Lenovo ideapad 320
Prosesor	: AMD A9-9420 Radeon R5 @ 3,0 GHz
Memori	: 3,11 GB RAM

5.1.2 Lingkungan Implementasi Perangkat Lunak

Dalam implementasi, digunakan perangkat lunak sebagai berikut:

1. Hyper 2.0.0: untuk menggonfigurasi lingkungan pemrograman dan melakukan uji coba program.
2. Sublime Text 3.1.1, *Build* 3176: untuk menulis program.
3. Arduino IDE 1.8.5 (*Windows Store* 1.8.10.0): untuk mengunggah program ke papan Arduino.
4. Rapidminer Studio 8.1.0: untuk melakukan perhitungan seleksi fitur dan uji coba metode klasifikasi.
5. Microsoft Visio 2016: untuk membuat diagram.

Sedangkan bahasa pemrograman yang dipakai untuk pembuatan sistem ini adalah:

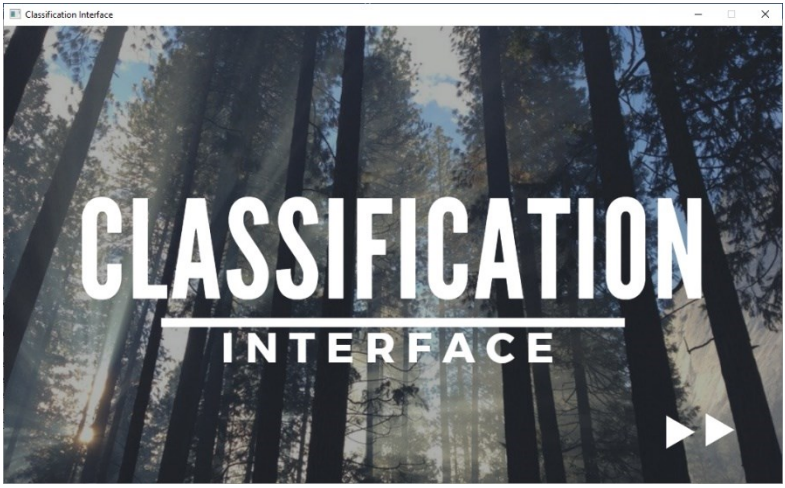
1. Bahasa Pemrograman Arduino: untuk memrogram papan Arduino.
2. Python 3.6.5: untuk membuat model klasifikasi menggunakan *library* scikit-learn 0.19.1 dan membangun antarmuka menggunakan kerangka kerja PyQt5.

5.2 Implementasi Tampilan Antarmuka

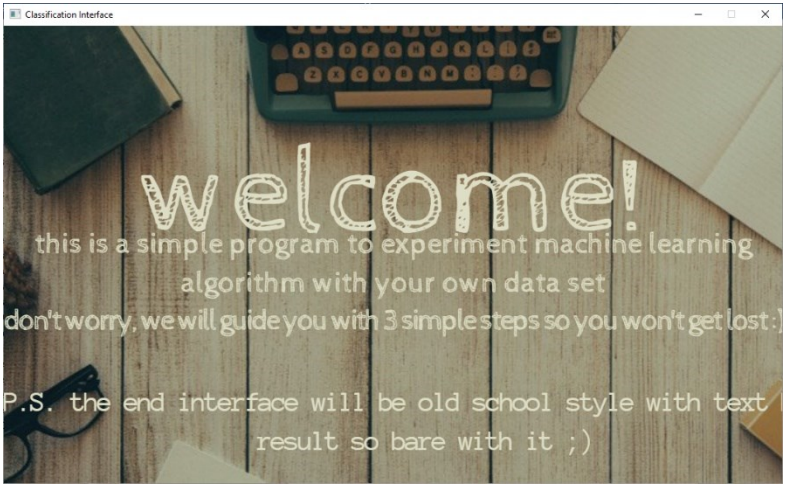
Subbab ini memaparkan tampilan antarmuka yang diimplementasikan pada sistem

5.2.1 Implementasi Tampilan Jendela Utama

Jendela utama tersusun dari beberapa halaman yang masing-masing memuat gambar interaktif yang memberi petunjuk cara menggunakan program seiring pengguna menjalankan program. Implementasi dari tampilan jendela utama dapat dilihat pada Gambar 5.1, Gambar 5.2, Gambar 5.3, Gambar 5.4, Gambar 5.5, Gambar 5.6, Gambar 5.7, dan Gambar 5.8.



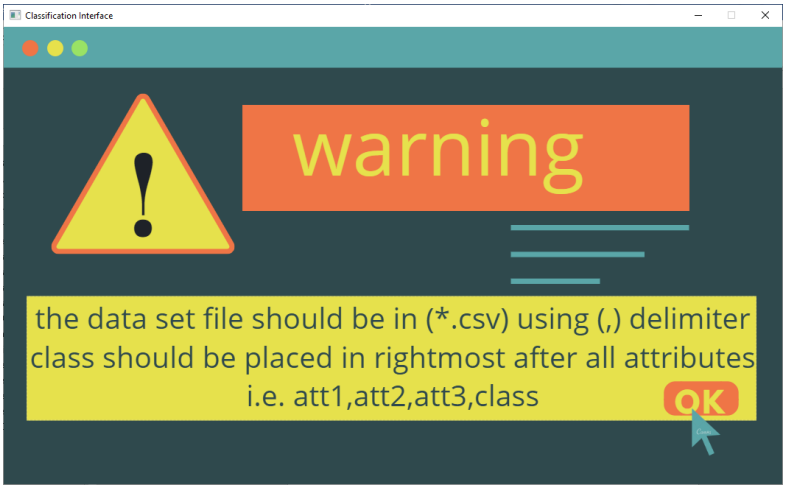
Gambar 5.1 Halaman Deskripsi program



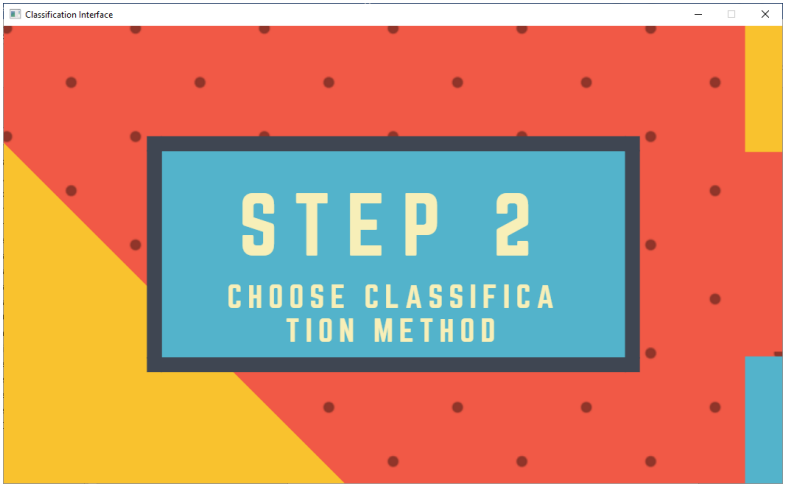
Gambar 5.2 Halaman Deskripsi program

Gambar 5.3 Halaman Langkah Pertama

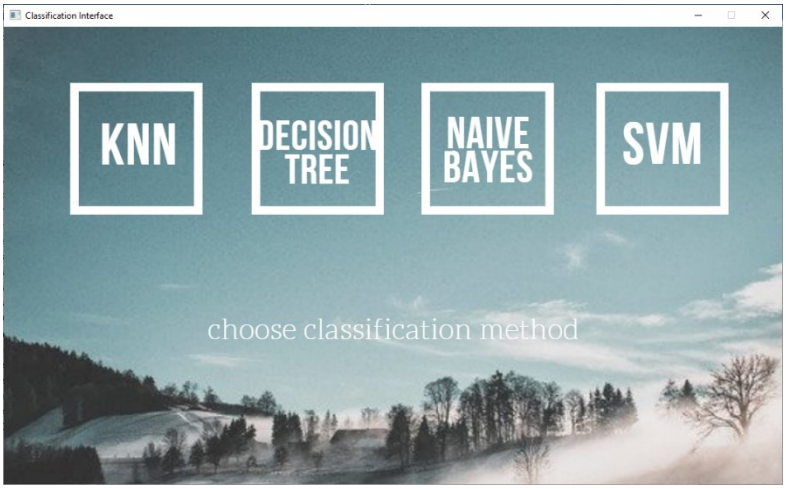
Gambar 5.4 Halaman Memilih File Data Set



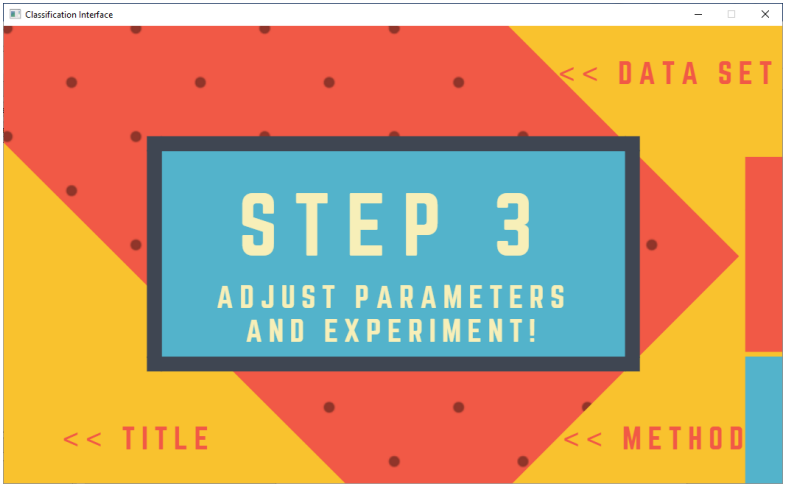
Gambar 5.5 Halaman Peringatan Format File Data Set



Gambar 5.6 Halaman Langkah Kedua



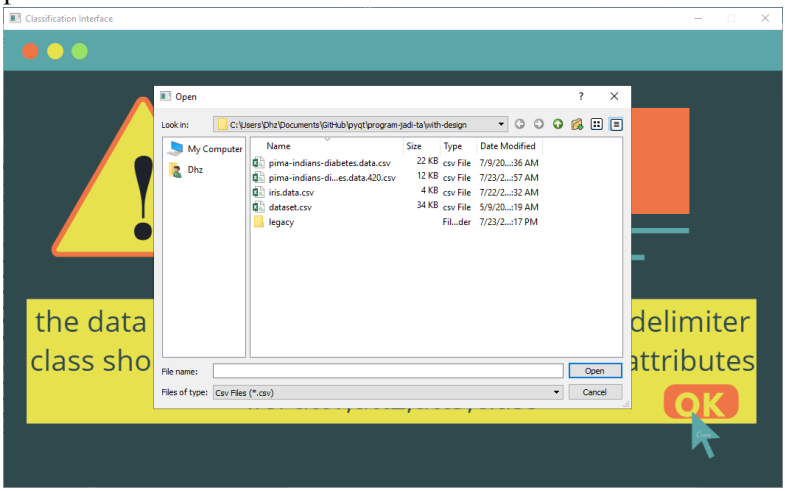
Gambar 5.7 Halaman Memilih Metode Klasifikasi



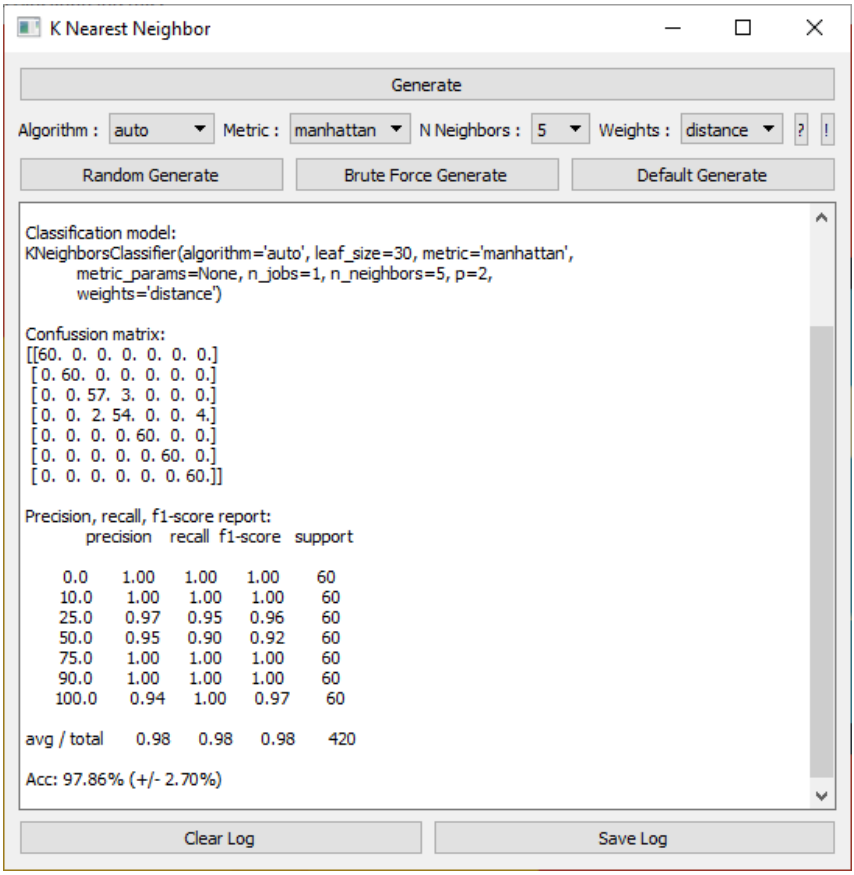
Gambar 5.8 Halaman Langkah Ketiga

5.2.2 Implementasi Tampilan Dialog

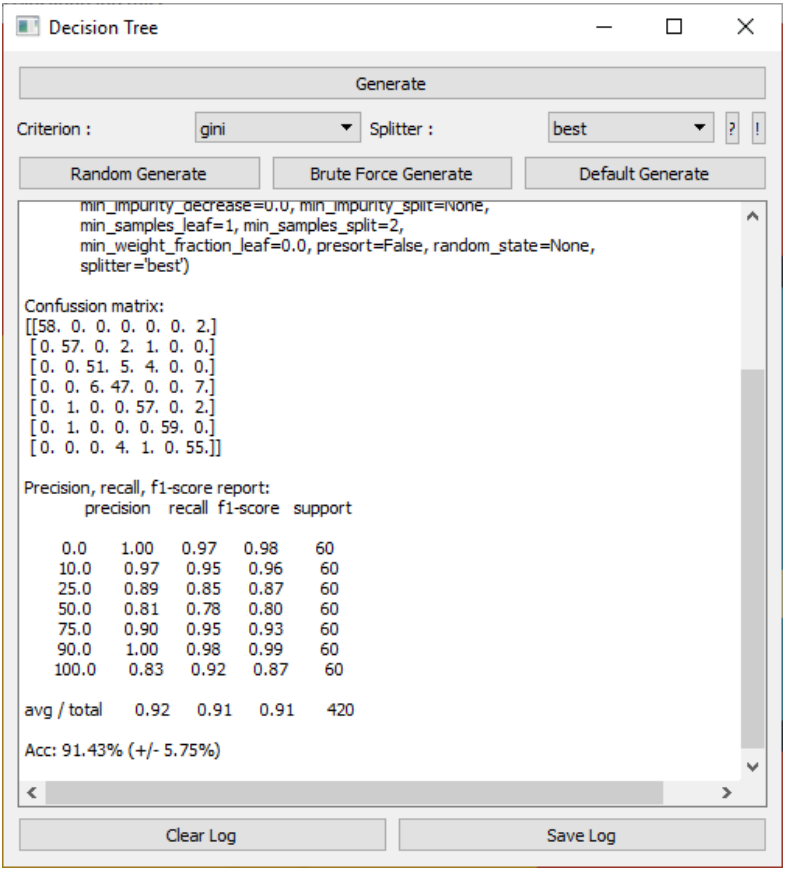
Subbab ini memaparkan dialog-dialog yang dapat dimunculkan pada sistem ini. Dialog untuk memuat *file data set* dapat dilihat pada Gambar 5.9. Dialog untuk melakukan uji coba klasifikasi dapat dilihat pada Gambar 5.10, Gambar 5.11, Gambar 5.12, dan Gambar 5.13. Dialog untuk menyimpan *log* dapat dilihat pada Gambar 5.14.



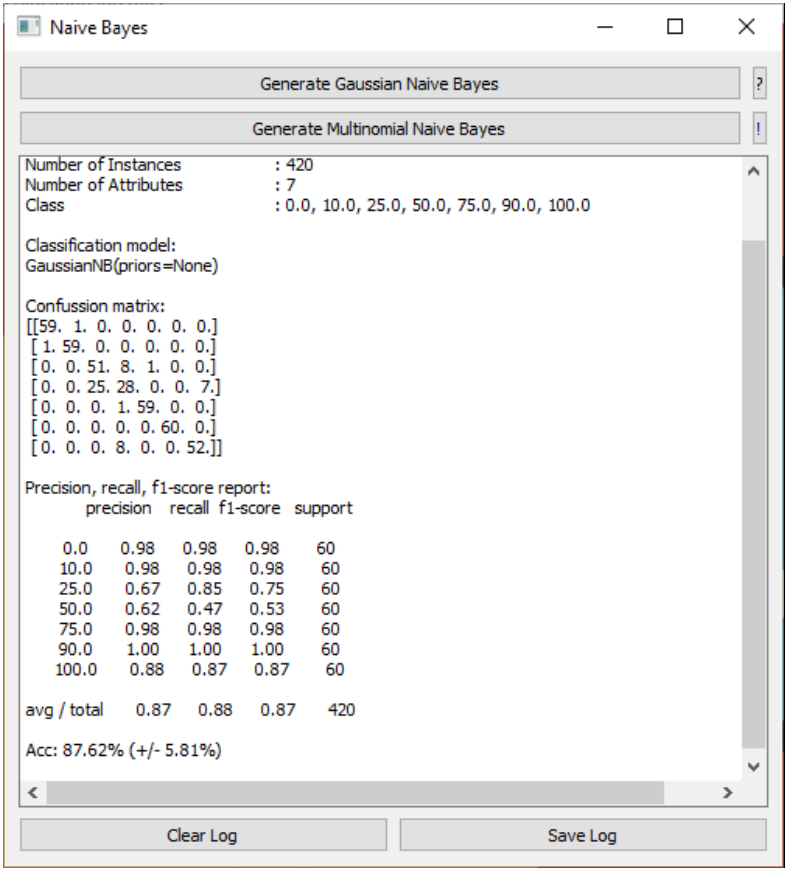
Gambar 5.9 Dialog Memuat File Data Set



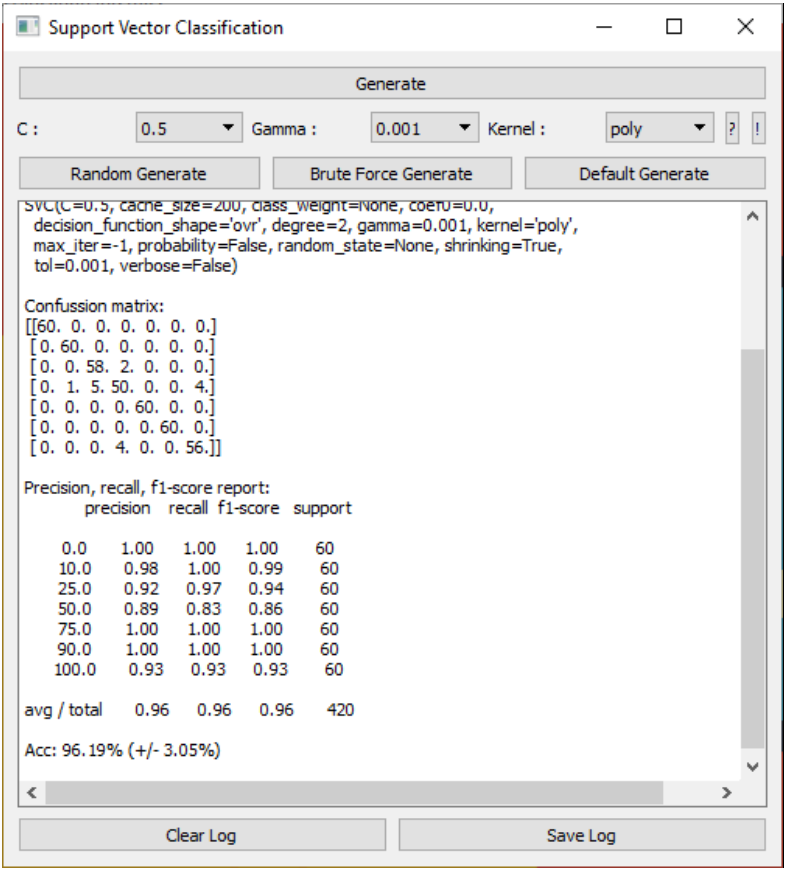
Gambar 5.10 Dialog Uji Coba Klasifikasi Metode kNN



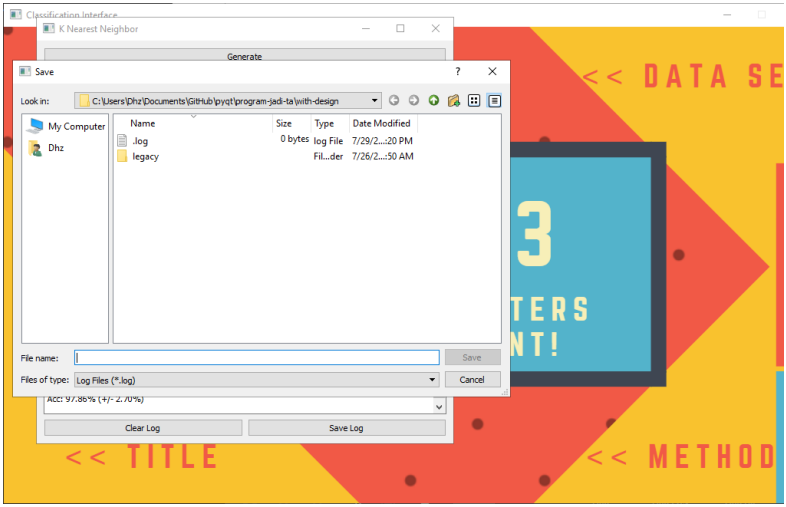
Gambar 5.11 Dialog Uji Coba Klasifikasi Metode Decision Tree



Gambar 5.12 Dialog Uji Coba Klasifikasi Metode Naïve Bayes



Gambar 5.13 Dialog Uji Coba Klasifikasi Metode Support Vector Classification



Gambar 5.14 Dialog Menyimpan Log

5.3 Implementasi Perangkat Lunak

Pada subbab ini dipaparkan implementasi sistem yang dibangun meliputi antarmuka dan logika di dalamnya.

5.3.1 Implementasi Jendela Utama

Halaman ini berupa tampilan pada jendela utama. Implementasi untuk halaman ini dapat dilihat pada Kode Sumber 5.1.

```
1.  class Classification(object):
2.      def __init__(self, model, dataset):
3.          # Fix random seed for reproducibility
4.          self.seed = 1909
5.          np.random.seed(self.seed)
6.
7.          # Configure classification model
8.          self.model = model
9.
10.         # Configure dataset
11.         self.dataset = dataset
```

```
12.
13.         self.pddataset = pd.read_csv(dataset, header
    =None)
14.
15.         # npdataset = np.loadtxt(dataset, delimiter=
    ",")
16.         npdataset = self.pddataset.values
17.         # Detect how many column in .csv
18.         # last column is class not attribute, so "-
    1" is to exclude that
19.         self.data = npdataset[:, 0:npdataset.shape[1
    ] - 1]
20.         self.target = npdataset[:, npdataset.shape[1
    ] - 1]
21.
22.         # Detect how many class
23.         self.n_class = len(set(self.target))
24.
25.     def Classify(self):
26.         print("")
27.         print("Data set details:")
28.         print("Filename\t\t: {}".format(os.path.base
    name(self.dataset)))
29.         print("Number of Instances\t: {}".format(sel
    f.data.shape[0]))
30.         print("Number of Attributes\t: {}".format(se
    lf.data.shape[1] - 1))
31.         print("Class\t\t: {}".format(
32.             ", ".join(str(e) for e in sorted(set(sel
    f.target)))
33.         ))
34.         # print(self.pddataset.groupby('class').size
    ())
35.
36.         print("")
37.         print("Classification model:")
38.         print(self.model)
39.
40.         expected, predicted, conf_matrix, cvscores =
     self.DataTrainTest()
41.
42.         print("")
```

```
43.         print("Confussion matrix:")
44.         print(conf_matrix)
45.
46.         print("")
47.         print("Precision, recall, f1-
    score report:")
48.         print(classification_report(expected, predic
    ted))
49.         print("Acc: %.2f%% (+/- %.2f%%)" %
50.               (np.mean(cvscores), np.std(cvscores)))


51.
52.     def DataTrainTest(self):
53.         n_expected = np.zeros((0))
54.         n_predicted = np.zeros((0))
55.         n_conf_matrix = np.zeros((self.n_class, self
    .n_class), dtype=np.float32)
56.         cvscores = []
57.
58.         # Define 10-
    fold cross validation test harness
59.         kfold = StratifiedKFold(
60.             n_splits=10, shuffle=True, random_state=
    self.seed)
61.
62.         for train, test in kfold.split(self.data, se
    lf.target):
63.             self.model.fit(self.data[train], self.ta
    rget[train])
64.
65.             # Make predictions
66.             expected = self.target[test]
67.             predicted = self.model.predict(self.data
    [test])
68.
69.             # Summarize the fit of the model
70.             n_expected = np.concatenate((n_expected,
     expected))
71.             n_predicted = np.concatenate((n_predicte
    d, predicted))
```

```
72.             conf_matrix = confusion_matrix(expected,
    predicted)
73.             n_conf_matrix += conf_matrix
74.
75.             # Evaluate the model
76.             self.scores = self.model.score(self.data
   [test], self.target[test])
77.             cvscores.append(self.scores * 100)
78.
79.         return n_expected, n_predicted, n_conf_matri
   x, cvscores
```

Kode Sumber 5.1 Implementasi Tampilan Jendela Utama

5.3.2 Implementasi Memuat *Data Set* (UC-2)

Dialog ini digunakan untuk memuat *data set*. Implementasinya dapat dilihat pada Kode Sumber 5.2 dan Kode Sumber 5.3.

```
1.  def BrowseFile(self):
2.      # Create textbox
3.      self.textbox = QLineEdit(self)
4.      self.textbox.move(20, 120)
5.      self.textbox.resize(370, 20)
6.      self.textbox.textChanged.connect(self.textChange
   dHandle)
7.      self.textbox.hide()
```

Kode Sumber 5.2 Implementasi Text Box Memuat Data Set

```
1.  def open_csv_dialog(self):
2.      options = QFileDialog.Options()
3.      options |= QFileDialog.DontUseNativeDialog
4.      filename, _ = QFileDialog.getOpenFileName(
5.          self, "Open", "", "Csv Files (*.csv)", opti
   ons=options)
6.      if filename:
7.          return filename
```

Kode Sumber 5.3 Implementasi Dialog Memuat Data Set

5.3.3 Implementasi Melakukan Uji Coba Klasifikasi

Berikut ini merupakan implementasi uji coba klasifikasi beserta dialog-dialog untuk masing-masing metode. Implementasi model klasifikasi beserta dialog-dialog untuk setiap metode dapat dilihat pada Kode Sumber 5.4, Kode Sumber 5.5, Kode Sumber 5.6, Kode Sumber 5.7, dan Kode Sumber 5.8.

```
1.  class Classification(object):
2.      def __init__(self, model, dataset):
3.          # Fix random seed for reproducibility
4.          self.seed = 1909
5.          np.random.seed(self.seed)
6.
7.          # Configure classification model
8.          self.model = model
9.
10.         # Configure dataset
11.         self.dataset = dataset
12.
13.         self.pddataset = pd.read_csv(dataset, heade
    r=None)
14.
15.         npdataset = self.pddataset.values
16.         # Detect how many column in .csv
17.         # last column is class not attribute, so "-
     1" is to exclude that
18.         self.data = npdataset[:, 0:npdataset.shape[
    1] - 1]
19.         self.target = npdataset[:, npdataset.shape[
    1] - 1]
20.
21.         # Detect how many class
22.         self.n_class = len(set(self.target))
23.
24.     def Classify(self):
25.         print("")
26.         print("Data set details:")
```

```
27.         print("Filename\t\t: {}".format(os.path.bas
    ename(self.dataset)))
28.         print("Number of Instances\t: {}".format(se
    lf.data.shape[0]))
29.         print("Number of Attributes\t: {}".format(s
    elf.data.shape[1] - 1))
30.         print("Class\t\t: {}".format(
31.             ", ".join(str(e) for e in sorted(set(se
    lf.target)))
32.         ))
33.
34.         print("")
35.         print("Classification model:")
36.         print(self.model)
37.
38.         expected, predicted, conf_matrix, cvscores
    = self.DataTrainTest()
39.
40.         print("")
41.         print("Confussion matrix:")
42.         print(conf_matrix)
43.
44.         print("")
45.         print("Precision, recall, f1-
    score report:")
46.         print(classification_report(expected, predi
    cted))
47.         print("Acc: %.2f%% (+/- %.2f%%)" %
48.               (np.mean(cvscores), np.std(cvscores))
    )
49.
50.     def DataTrainTest(self):
51.         n_expected = np.zeros((0))
52.         n_predicted = np.zeros((0))
53.         n_conf_matrix = np.zeros((self.n_class, sel
    f.n_class), dtype=np.float32)
54.         cvscores = []
55.
56.         # Define 10-
    fold cross validation test harness
57.         kfold = StratifiedKFold(
```

```
            n_splits=10, shuffle=True, random_state
=self.seed)

        for train, test in kfold.split(self.data, s
elf.target):
            self.model.fit(self.data[train], self.t
arget[train])

            # Make predictions
            expected = self.target[test]
            predicted = self.model.predict(self.dat
a[test])

            # Summarize the fit of the model
            n_expected = np.concatenate((n_expected
, expected))
            n_predicted = np.concatenate((n_predict
ed, predicted))
            conf_matrix = confusion_matrix(expected
, predicted)
            n_conf_matrix += conf_matrix

            # Evaluate the model
            self.scores = self.model.score(self.dat
a[test], self.target[test])
            cvscores.append(self.scores * 100)

        return n_expected, n_predicted, n_conf_matr
ix, cvscores
```

Kode Sumber 5.4 Model Klasifikasi

```
class GenerateKNN(QMainWindow):
    def __init__(self, dataset, parent=None):
        super(GenerateKNN, self).__init__(parent)
        self.setGeometry(50, 50, 500, 425)
        self.setWindowTitle("K Nearest Neighbor")

        self.home(dataset)

```

```
9.          sys.stdout = Stream(newText=self.onUpdateTe
    xt)
10.
11.     def onUpdateText(self, text):
12.         cursor = self.process.textCursor()
13.         cursor.movePosition(QTextCursor.End)
14.         cursor.insertText(text)
15.         self.process.setTextCursor(cursor)
16.         self.process.ensureCursorVisible()
17.
18.     def __del__(self):
19.         sys.stdout = sys.__stdout__
20.
21.     def home(self, dataset):
22.         w = QWidget()
23.         self.setCentralWidget(w)
24.         lay = QVBoxLayout(w)
25.         btn = QPushButton("Generate")
26.
27.         hbox = QHBoxLayout()
28.
29.         lbl_algorithm = QLabel(w)
30.         lbl_algorithm.setText("Algorithm :")
31.         cbx_algorithm = QComboBox(w)
32.         cbx_algorithm.addItems(['auto', 'ball_tree'
    , 'kd_tree', 'brute'])
33.
34.         lbl_metric = QLabel(w)
35.         lbl_metric.setText("Metric :")
36.         cbx_metric = QComboBox(w)
37.         cbx_metric.addItems(
38.             ['euclidean', 'manhattan', 'chebyshev',
    'minkowski'])
39.         cbx_metric.setCurrentIndex(1)
40.
41.         lbl_n_neighbors = QLabel(w)
42.         lbl_n_neighbors.setText("N Neighbors :")
43.         cbx_n_neighbors = QComboBox(w)
44.         cbx_n_neighbors.addItems(["%s" % i for i in
    range(2, 17)])
45.         cbx_n_neighbors.setCurrentIndex(3)
46.
```

```
47.         lbl_weights = QLabel(w)
48.         lbl_weights.setText("Weights :")
49.         cbx_weights = QComboBox(w)
50.         cbx_weights.addItems(['uniform', 'distance'
    ])
51.         cbx_weights.setCurrentIndex(1)
52.
53.         btn.clicked.connect(lambda: self.TextFSM(mo
    del=KNeighborsClassifier(
54.             algorithm=cbx_algorithm.currentText(),

55.             metric=cbx_metric.currentText(),
56.             n_neighbors=int(cbx_n_neighbors.current
    Text()),
57.             weights=cbx_weights.currentText()
58.         ), dataset=dataset))
59.
60.         self.process = QTextEdit()
61.         self.process.moveCursor(QTextCursor.Start)

62.         self.process.ensureCursorVisible()
63.         self.process.setLineWrapColumnOrWidth(500)

64.         self.process.setLineWrapMode(QTextEdit.Fixe
    dPixelWidth)
65.
66.         lay.addWidget(btn)
67.         lay.addLayout(hbox)
68.
69.         hbox.addWidget(lbl_algorithm)
70.         hbox.addWidget(cbx_algorithm)
71.         hbox.addWidget(lbl_metric)
72.         hbox.addWidget(cbx_metric)
73.         hbox.addWidget(lbl_n_neighbors)
74.         hbox.addWidget(cbx_n_neighbors)
75.         hbox.addWidget(lbl_weights)
76.         hbox.addWidget(cbx_weights)
77.
78.         lay.addWidget(self.process)
79.
80.         self.show()
```

```
81.
82.     def TextFSM(self, model, dataset):
83.         Classifier = Classification(
84.             model=model, dataset=np.loadtxt(dataset
    , delimiter=","))
85.         Classifier.Classify()
```

Kode Sumber 5.5 Dialog Metode kNN

```
1.  class GenerateDecisionTree(QMainWindow):
2.      def __init__(self, dataset, parent=None):
3.          super(GenerateDecisionTree, self).__init__(
    parent)
4.          self.setGeometry(50, 50, 500, 425)
5.          self.setWindowTitle("Decision Tree")
6.          self.home(dataset)
7.
8.          sys.stdout = Stream(newText=self.onUpdateTe
    xt)
9.
10.     def onUpdateText(self, text):
11.         cursor = self.process.textCursor()
12.         cursor.movePosition(QTextCursor.End)
13.         cursor.insertText(text)
14.         self.process.setTextCursor(cursor)
15.         self.process.ensureCursorVisible()
16.
17.     def __del__(self):
18.         sys.stdout = sys.__stdout__
19.
20.     def home(self, dataset):
21.         w = QWidget()
22.         self.setCentralWidget(w)
23.         lay = QVBoxLayout(w)
24.         btn = QPushButton("Generate")
25.
26.         hbox = QHBoxLayout()
27.
28.         lbl_criterion = QLabel(w)
29.         lbl_criterion.setText("Criterion :")
30.         cbx_criterion = QComboBox(w)
```

```
31.         cbx_criterion.addItems(['gini', 'entropy'])
32.
33.         lbl_splitter = QLabel(w)
34.         lbl_splitter.setText("Splitter :")
35.         cbx_splitter = QComboBox(w)
36.         cbx_splitter.addItems(['best', 'random'])
37.
38.         btn.clicked.connect(lambda: self.TextFSM(mo
    del=DecisionTreeClassifier(
39.             criterion=cbx_criterion.currentText(),
40.             splitter=cbx_splitter.currentText()
41.         ), dataset=dataset))
42.
43.         self.process = QTextEdit()
44.         self.process.moveCursor(QTextCursor.Start)
45.         self.process.ensureCursorVisible()
46.         self.process.setLineWrapColumnOrWidth(500)
47.         self.process.setLineWrapMode(QTextEdit.Fixe
    dPixelWidth)
48.
49.         lay.addWidget(btn)
50.         lay.addLayout(hbox)
51.
52.         hbox.addWidget(lbl_criterion)
53.         hbox.addWidget(cbx_criterion)
54.         hbox.addWidget(lbl_splitter)
55.         hbox.addWidget(cbx_splitter)
56.
57.         lay.addWidget(self.process)
58.
59.         self.show()
60.
61.     def TextFSM(self, model, dataset):
62.         Classifier = Classification(
63.             model=model, dataset=np.loadtxt(dataset
    , delimiter=","))
64.         Classifier.Classify()
```

Kode Sumber 5.6 Dialog Metode Decision Tree

```
1.  class GenerateNaiveBayes(QMainWindow):
2.      def __init__(self, dataset, parent=None):
3.          super(GenerateNaiveBayes, self).__init__(pa
    rent)
4.          self.setGeometry(50, 50, 500, 425)
5.          self.setWindowTitle("Naive Bayes")
6.          self.home(dataset)
7.
8.          sys.stdout = Stream(newText=self.onUpdateTe
    xt)
9.
10.     def onUpdateText(self, text):
11.         cursor = self.process.textCursor()
12.         cursor.movePosition(QTextCursor.End)
13.         cursor.insertText(text)
14.         self.process.setTextCursor(cursor)
15.         self.process.ensureCursorVisible()
16.
17.     def __del__(self):
18.         sys.stdout = sys.__stdout__
19.
20.     def home(self, dataset):
21.         w = QWidget()
22.         self.setCentralWidget(w)
23.         lay = QVBoxLayout(w)
24.         btn = QPushButton("Generate Gaussian Naive
    Bayes")
25.         btn2 = QPushButton("Generate Multinomial Na
    ive Bayes")
26.         model = GaussianNB()
27.         model2 = MultinomialNB()
28.
29.         btn.clicked.connect(lambda: self.TextFSM(mo
    del, dataset))
30.         btn2.clicked.connect(lambda: self.TextFSM(m
    odel2, dataset))
31.
32.         self.process = QTextEdit()
```

```
33.         self.process.moveCursor(QTextCursor.Start)
34.         self.process.ensureCursorVisible()
35.         self.process.setLineWrapColumnOrWidth(500)
36.         self.process.setLineWrapMode(QTextEdit.Fixe
    dPixelWidth)
37.
38.         lay.addWidget(btn)
39.         lay.addWidget(btn2)
40.         lay.addWidget(self.process)
41.
42.         self.show()
43.
44.     def TextFSM(self, model, dataset):
45.         Classifier = Classification(
46.             model=model, dataset=np.loadtxt(dataset
    , delimiter=","))
47.         Classifier.Classify()
```

Kode Sumber 5.7 Dialog Metode Naïve Bayes

```
1.  class GenerateSVC(QMainWindow):
2.      def __init__(self, dataset, parent=None):
3.          super(GenerateSVC, self).__init__(parent)
4.          self.setGeometry(50, 50, 500, 425)
5.          self.setWindowTitle("Support Vector Classif
    ication")
6.          self.home(dataset)
7.
8.          sys.stdout = Stream(newText=self.onUpdateTe
    xt)
9.
10.     def onUpdateText(self, text):
11.         cursor = self.process.textCursor()
12.         cursor.movePosition(QTextCursor.End)
13.         cursor.insertText(text)
14.         self.process.setTextCursor(cursor)
15.         self.process.ensureCursorVisible()
16.
17.     def __del__(self):
```

```
18.         sys.stdout = sys.__stdout__
19.
20.     def home(self, dataset):
21.         w = QWidget()
22.         self.setCentralWidget(w)
23.         lay = QVBoxLayout(w)
24.         btn = QPushButton("Generate")
25.
26.         hbox = QHBoxLayout()
27.
28.         lbl_C = QLabel(w)
29.         lbl_C.setText("C :")
30.         cbx_C = QComboBox(w)
31.         cbx_C.addItems(['0.25', '0.5', '0.75', '1.0
    '])
32.         cbx_C.setCurrentIndex(3)
33.
34.         lbl_gamma = QLabel(w)
35.         lbl_gamma.setText("Gamma :")
36.         cbx_gamma = QComboBox(w)
37.         cbx_gamma.addItems(['auto', '0.001'])
38.         cbx_gamma.setCurrentIndex(1)
39.         _dict = {'auto': 'auto', '0.001': 0.001}
40.
41.         lbl_kernel = QLabel(w)
42.         lbl_kernel.setText("Kernel :")
43.         cbx_kernel = QComboBox(w)
44.         cbx_kernel.addItems(['linear', 'poly', 'rbf
    ', 'sigmoid'])
45.         cbx_kernel.setCurrentIndex(1)
46.
47.         btn.clicked.connect(lambda: self.TextFSM(mo
    del=SVC(
48.             C=float(cbx_C.currentText()),
49.             degree=2,
50.             kernel=cbx_kernel.currentText(),
51.             gamma=_dict[cbx_gamma.currentText()]
52.         ), dataset=dataset))
53.
54.         self.process = QTextEdit()
55.         self.process.moveCursor(QTextCursor.Start)
```

```
56.         self.process.ensureCursorVisible()
57.         self.process.setLineWrapColumnOrWidth(500)

58.         self.process.setLineWrapMode(QTextEdit.Fixe
    dPixelWidth)
59.
60.         lay.addWidget(btn)
61.         lay.addLayout(hbox)
62.
63.         hbox.addWidget(lbl_C)
64.         hbox.addWidget(cbx_C)
65.         hbox.addWidget(lbl_gamma)
66.         hbox.addWidget(cbx_gamma)
67.         hbox.addWidget(lbl_kernel)
68.         hbox.addWidget(cbx_kernel)
69.
70.         lay.addWidget(self.process)
71.
72.         self.show()
73.
74.     def TextFSM(self, model, dataset):
75.         Classifier = Classification(
76.             model=model, dataset=np.loadtxt(dataset
    , delimiter=","))
77.         Classifier.Classify()
```

Kode Sumber 5.8 Dialog Metode SVM

5.3.4 Implementasi Menampilkan Dialog Tentang Program (UC-4)

Pada dialog ini ditampilkan tentang program. Implementasi dapat dilihat pada Kode Sumber 5.9.

```
1.  def about(self):
2.      QMessageBox.question(
3.          self, "About",
4.          "<font size=5> Meat Classification </font><
    br>" +
5.          "<font size=3> User Interface </font><br><b
    r>" +
```

```
6.          "<font size=3> Created by Aldhiaz Fathra Da
    iva </font><br>" +
7.          "<font size=3>" +
8.          "Copyright © 2018 Institut Teknologi Sepulu
    h Nopember </font>",
9.          QMessageBox.Ok
10.     )
```

Kode Sumber 5.9 Implementasi Tentang Program

[Halaman ini sengaja dikosongkan]

BAB VI
PENGUJIAN DAN EVALUASI

Pada bab ini dibahas mengenai pengujian dan evaluasi sistem yang telah dibuat.

6.1 Lingkungan Pengujian

Lingkungan pengujian sistem yang digunakan pada pengerjaan tugas akhir ini adalah sebagai berikut:

Jenis	: Laptop
Tipe	: Lenovo ideapad 320
Prosesor	: AMD A9-9420 Radeon R5 @ 3,0 GHz
Memori	: 3,11 GB RAM

6.2 Skenario Pengujian

Skenario pengujian yang dilakukan adalah uji coba klasifikasi daging dengan menggunakan metode kNN, *decision tree, naïve bayes*, dan *Support Vector Machine*

Uji coba dilakukan menggunakan Python 3.6.5 dengan *library* scikit-learn 0.19.1. Uji coba yang dimaksud adalah evaluasi akurasi dari masing-masing modul. Sampel dibagi menjadi tujuh kelas:

1. S0% : 100% daging babi (daging babi murni),
2. S10% : Campuran 10% daging sapi 90% daging babi,
3. S25% : Campuran 25% daging sapi 75% daging babi,
4. S50% : Campuran 50% daging sapi 50% daging babi,
5. S75% : Campuran 75% daging sapi 25% daging babi,
6. S90% : Campuran 90% daging sapi 10% daging babi,
7. S100%: 100% daging sapi (daging sapi murni).

Selanjutnya divisualisasikan tabel *confusion matrix* dan dilakukan proses penghitungan nilai presisi, sensitivitas, F_1 *score*, dan akurasi.

6.2.1 Skenario Pengujian 1

Pada skenario pengujian ini dilakukan uji coba klasifikasi daging dengan menggunakan metode kNN.

Tabel 6.1 Confussion Matrix Metode kNN

		Prediksi						
	Kelas	S0%	S10%	S25%	S50%	S75%	S90%	S100%
Aktual	S0%	60	0	0	0	0	0	0
	S10%	0	60	0	0	0	0	0
	S25%	0	0	57	3	0	0	0
	S50%	0	0	2	54	0	0	4
	S75%	0	0	0	0	60	0	0
	S90%	0	0	0	0	0	60	0
	S100%	0	0	0	0	0	0	60

Tabel 6.2 Nilai Presisi, Sensitivitas, dan F_1 Score Metode kNN

Kelas	Presisi	Sensitivitas	*F_1 Score*	Sampel
S0%	1,00	1,00	1,00	60
S10%	1,00	1,00	1,00	60
S25%	0,97	0,95	0,96	60
S50%	0,95	0,90	0,92	60
S75%	1,00	1,00	1,00	60
S90%	1,00	1,00	1,00	60
S100%	0,94	1,00	0,97	60
Avg/Total	0,98	0,98	0,98	420

Hasil uji coba pada skenario pengujian 1 didapatkan hasil akurasi 97.86% dengan standar deviasi 2.70%. Hasil *confussion matrix* dapat dilihat pada Tabel 6.1.

6.2.2 Skenario Pengujian 2

Pada skenario pengujian ini dilakukan uji coba klasifikasi daging dengan menggunakan metode *decision tree*.

Tabel 6.3 Confussion Matrix Metode Decision Tree

		Prediksi						
	Kelas	S0%	S10%	S25%	S50%	S75%	S90%	S100%
Aktual	S0%	58	0	0	0	0	0	2
	S10%	0	57	0	0	0	0	0
	S25%	0	0	51	0	0	0	0
	S50%	0	0	6	47	0	0	7
	S75%	0	1	0	0	57	0	2
	S90%	0	1	0	0	0	59	0
	S100%	0	0	0	4	1	0	55

Tabel 6.4 Nilai Presisi, Sensitivitas, dan F_1 Score Metode Decision Tree

Kelas	Presisi	Sensitivitas	*F_1 Score*	Sampel
S0%	1,00	0,97	0,98	60
S10%	0,97	0,95	0,96	60
S25%	0,89	0,85	0,87	60
S50%	0,81	0,78	0,80	60
S75%	0,90	0,95	0,93	60
S90%	1,00	0,98	0,99	60
S100%	0,83	0,92	0,87	60
Avg/Total	0,92	0,91	0,91	420

Hasil uji coba pada skenario pengujian 2 didapatkan hasil akurasi 91.43% dengan standar deviasi 5.75%. Hasil *confussion matrix* dapat dilihat pada Tabel 6.3.

6.2.3 Skenario Pengujian 3

Pada skenario pengujian ini dilakukan uji coba klasifikasi daging dengan menggunakan metode *naïve bayes*.

Tabel 6.5 Confussion Matrix Metode Naïve Bayes

		Prediksi						
	Kelas	S0%	S10%	S25%	S50%	S75%	S90%	S100%
Aktual	S0%	59	1	0	0	0	0	0
	S10%	1	59	0	0	0	0	0
	S25%	0	0	51	8	1	0	0
	S50%	0	0	25	28	0	0	7
	S75%	0	0	0	1	59	0	0
	S90%	0	0	0	0	0	60	0
	S100%	0	0	0	8	0	0	52

Tabel 6.6 Nilai Presisi, Sensitivitas, dan F_1 Score Metode Naïve Bayes

Kelas	Presisi	Sensitivitas	*F_1 Score*	Sampel
S0%	0,98	0,98	0,98	60
S10%	0,98	0,98	0,98	60
S25%	0,67	0,85	0,75	60
S50%	0,62	0,47	0,53	60
S75%	0,98	0,98	0,98	60
S90%	1,00	1,00	1,00	60
S100%	0,88	0,87	0,87	60
Avg/Total	0,87	0,88	0,87	420

Hasil uji coba pada skenario pengujian 3 didapatkan hasil akurasi 87.62% dengan standar deviasi 5.81%. Hasil *confussion matrix* dapat dilihat pada Tabel 6.5.

6.2.4 Skenario Pengujian 4

Pada skenario pengujian ini dilakukan uji coba klasifikasi daging dengan menggunakan metode *Support Vector Machine*.

Tabel 6.7 Confussion Matrix Metode Support Vector Machine

		Prediksi						
	Kelas	S0%	S10%	S25%	S50%	S75%	S90%	S100%
Aktual	S0%	60	0	0	0	0	0	0
	S10%	0	60	0	0	0	0	0
	S25%	0	0	56	4	0	0	0
	S50%	0	1	5	50	0	0	4
	S75%	0	0	0	0	60	0	0
	S90%	0	0	0	0	0	60	0
	S100%	0	0	0	4	0	0	56

Tabel 6.8 Nilai Presisi, Sensitivitas, dan F_1 Score Metode Support Vector Machine

Kelas	Presisi	Sensitivitas	*F_1 Score*	Sampel
S0%	1,00	1,00	1,00	60
S10%	0,98	1,00	0,99	60
S25%	0,92	0,97	0,94	60
S50%	0,89	0,83	0,86	60
S75%	1,00	1,00	1,00	60
S90%	1,00	1,00	1,00	60
S100%	0,93	0,93	0,93	60
Avg/Total	0,96	0,96	0,96	420

Hasil uji coba pada skenario pengujian 4 didapatkan hasil akurasi 96.19% dengan standar deviasi 3.05%. Hasil *confussion matrix* serta perhitungan dapat dilihat pada Tabel 6.7.

BAB VII
KESIMPULAN DAN SARAN

Berdasarkan penyusunan tugas akhir yang dilakukan mulai dari penyusunan proposal, pembuatan alat *electronic nose*, pembuatan perangkat lunak, pengambilan data, serta penyusunan tugas akhir, maka pada subbab ini diberikan kesimpulan dan saran yang dibahas pada bab ini.

7.1 Kesimpulan

Berdasarkan hasil pengamatan selama proses perancangan, pembuatan *electronic nose*, pembuatan perangkat lunak, dan pengujian yang dilakukan, maka dapat diambil kesimpulan sebagai berikut:

1. Data bau daging berhasil terekam menggunakan susunan sensor gas elektrokimia.
2. Perangkat yang digunakan untuk merekam data bau daging, *electronic nose*, berhasil dirakit menggunakan dua komponen utama, yaitu susunan sensor gas elektrokimia dan papan Arduino.
3. Data bau daging yang berbentuk analog berhasil diolah untuk machine learning dengan: direkam menggunakan *electronic nose* berdasarkan konsentrasi gasnya; dikonversikan menjadi bentuk digital berupa data sinyal; didekomposisi menggunakan *discrete wavelet transform* untuk menghilangkan noise yang tidak relevan; dilakukan ekstraksi fitur statistik; diambil fitur statistik yang paling relevan dengan menggunakan chi-squared weighting; dilakukan klasifikasi menggunakan metode kNN, *decision tree*, *naïve bayes*, dan *support vector machine*.

4. Sistem ini menyajikan hasil analisa menggunakan aplikasi interaktif berbasis desktop dengan format penyajian berbasis teks.

7.2 Saran

Berikut adalah saran-saran yang dapat dilakukan untuk pengembangan di masa yang akan datang, didasarkan pada hasil perancangan, pembuatan *electronic nose*, penyusunan aplikasi, pengukuran dan pengolahan data, serta pengujian yang telah dilakukan.

1. Untuk memperkuat kepintaran *electronic nose* dalam menglasifikasi daging, ruang sampel dapat diperluas dengan menambah banyaknya data sampel yang digunakan.
2. Hasil analisa dapat disajikan dengan lebih indah jika aplikasi dibangun menggunakan teknologi web.

DAFTAR PUSTAKA

[1] M. Nurjuliana, Y. B. Che Man, and D. Mat Hashim, "Analysis of lard's aroma by an electronic nose for rapid Halal authentication," *JAOCS, J. Am. Oil Chem. Soc.*, vol. 88, no. 1, pp. 75–82, 2011.

[2] X. Tian, J. Wang, and S. Cui, "Analysis of pork adulteration in minced mutton using electronic nose of metal oxide sensors," *J. Food Eng.*, vol. 119, no. 4, pp. 744–749, 2013.

[3] D. R. Wijaya, R. Sarno, and E. Zulaika, "Information Quality Ratio as a novel metric for mother wavelet selection," *Chemom. Intell. Lab. Syst.*, vol. 160, no. July 2016, pp. 59–71, 2017.

[4] "arduino.cc." [Online]. Tersedia: https://www.arduino.cc/. [Diakses: 06-Jun-2017].

[5] H. Gunawan, *Prinsip-prinsip Elektronik edisi ke-2*, 2nd ed. Jakarta: Erlangga, 2010.

[6] R. Tem, U. Tem, and S. Tem, "Technical MQ2 Gas Sensor," vol. 1, p. 2, 2015.

[7] L. Henan Hanwei Electronics Co., "Technical Datasheet MQ4 Gas Sensor," pp. 2–4, 2015.

[8] R. Tem, "MQ138 Gas Sensor," vol. 1, pp. 3–4, 2011.

[9] MQ6 datasheet, "MQ6 datasheet," *Structure*, pp. 1–2, 2011.

[10] Hanwei Electronics, "MQ9 combustible gas sensor datasheet," vol. 1, pp. 1–14, 2015.

[11] R. Tem, "Technical MQ135 Gas Sensor," vol. 1, pp. 3–4, 2015.

[12] T. Data, "MQ136 Semiconductor Sensor for Sulfur Dioxide," pp. 2–4, 2015.

[13] B. Environment, "MQ137 GAS SENSOR Rs / Ro," vol. 1, pp. 3–4, 2015.

[14] M. Rouse, "data preprocessing." [Online]. Tersedia:

http://searchsqlserver.techtarget.com/definition/data-preprocessing. [Diakses: 02-Jun-2017].

[15] H. Kim *et al.*, "Electronic-nose for detecting environmental pollutants: Signal processing and analog front-end design," *Analog Integr. Circuits Signal Process.*, vol. 70, no. 1, pp. 15–32, 2012.

[16] X. Guo *et al.*, "A novel feature extraction approach using window function capturing and QPSO-SVM for enhancing electronic nose performance," *Sensors (Switzerland)*, vol. 15, no. 7, pp. 15198–15217, 2015.

[17] A. DLI, "Stationary Signals," 2009. [Online]. Tersedia: http://www.azimadli.com/vibman/stationarysignals.htm. [Diakses: 26-Jun-2017].

[18] D. R. Wijaya, R. Sarno, and E. Zulaika, "Sensor array optimization for mobile electronic nose: Wavelet transform and filter based feature selection approach," *Int. Rev. Comput. Softw.*, vol. 11, no. 8, pp. 659–671, 2016.

[19] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network," *Neurocomputing*, pp. 41–50, 1990.

[20] A. Fauzan, "Penerapan Cross Validation Sebagai Metode Pengujian Sistem," 2015. [Online]. Tersedia: http://www.charisfauzan.net/2015/02/penerapan-cross-validation-sebagai.html. [Diakses: 09-Jun-2017]

[21] U. of R. Department of Computer Sciences, "Confusion Matrix." [Online]. Tersedia: http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/%0Aconfusion_matrix.html%0A. [Diakses: 03-Jul-2017].

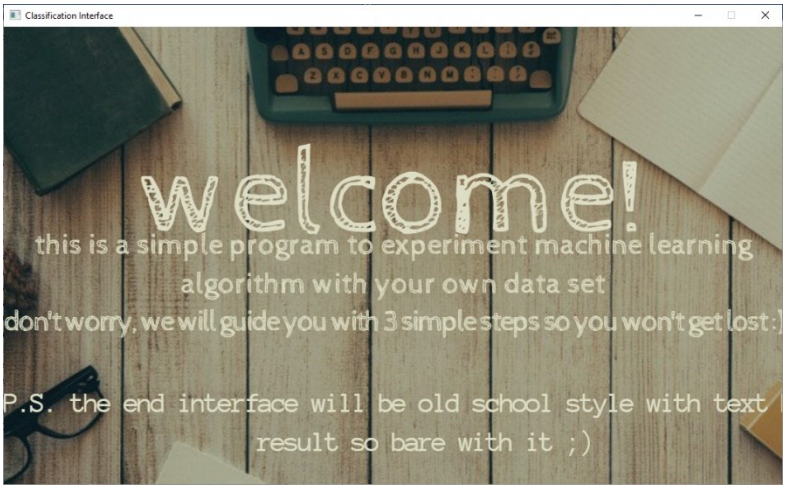
[22] H. K. Patel, R. H. Austin, J. Barber, and H. K. Patel, *The Electronic Nose: Artificial Olfaction Technology*. 2014.

LAMPIRAN

Panduan Penggunaan Program

Program yang dibangun pada tugas akhir ini berfungsi untuk melakukan uji coba klasifikasi menggunakan *data set* apapun dengan jumlah atribut dan instansi bebas selama memiliki ekstensi (.csv) dan mengikuti *template*, yaitu menggunakan delimiter (,) dan peletakkan kelas berada pada paling akhir/kanan baris (contoh: atr1, atr2, atr3, kelas).

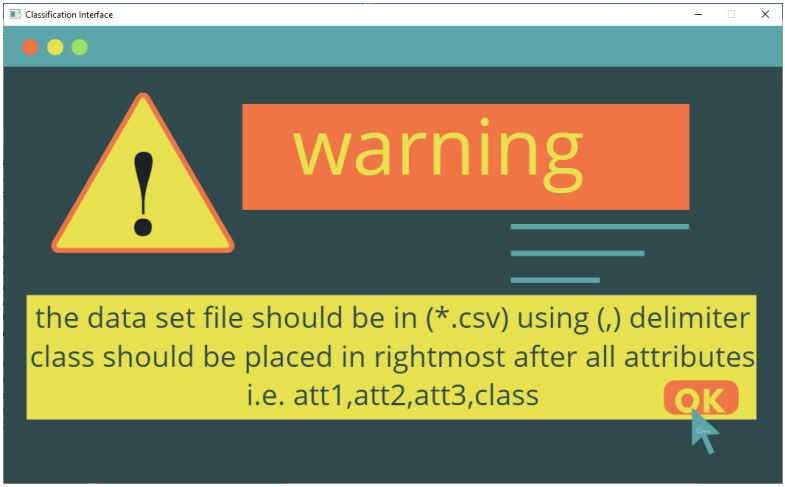
Gambar di atas merupakan tampilan awal dari program. Klik di mana saja untuk melanjutkan.



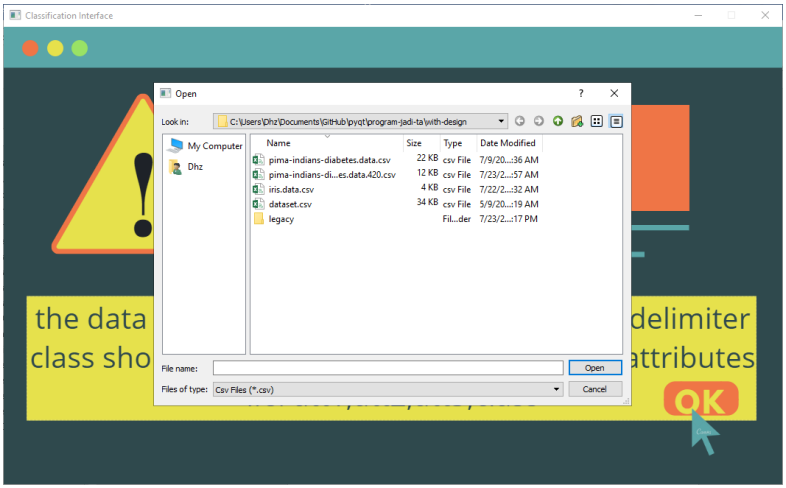
Pada tampilan berikutnya diterangkan deskripsi singkat mengenai program. Klik di mana saja untuk melanjutkan.

Gambar di atas adalah halaman awal dari langkah pertama, yaitu memuat *data set*. Klik di mana saja untuk melanjutkan.

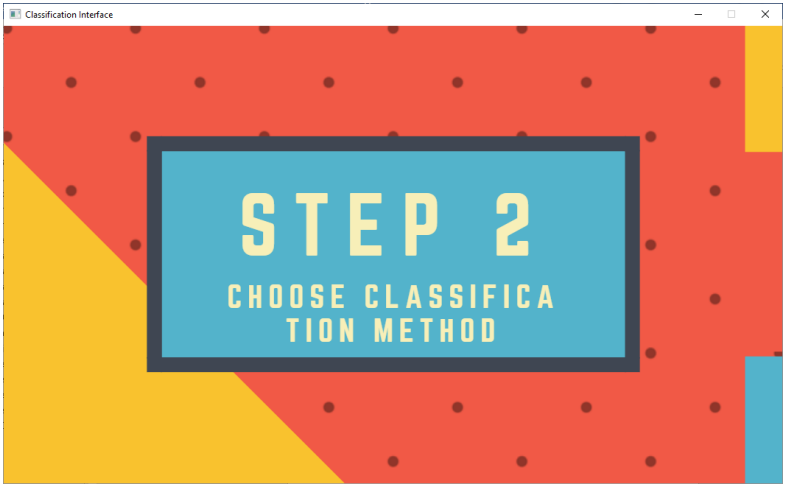
Klik di mana saja untuk memuat *file data set* yang akan diproses.



Ups! Sebelum melanjutkan, diingatkan terlebih dahulu bahwa *file* yang akan dimuat harus memiliki format seperti yang dipaparkan pada program. Klik di mana saja untuk melanjutkan.

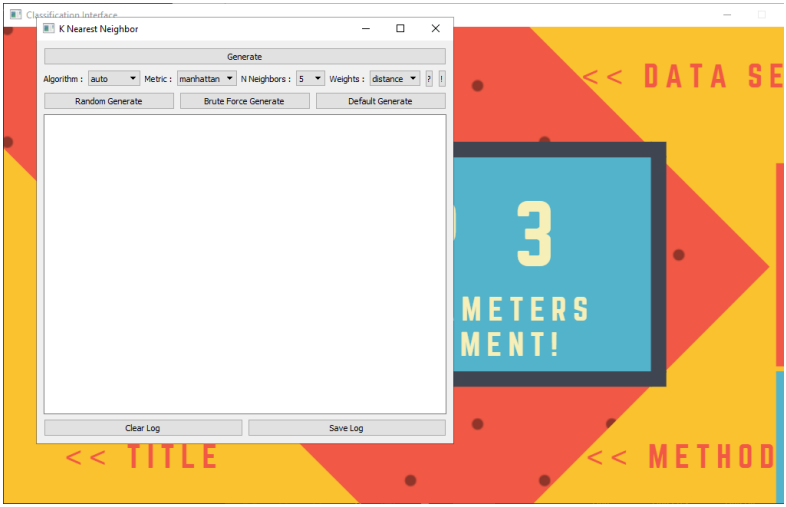


Akhirnya muncul dialog untuk membuka file. Pilih *file data set* yang akan digunakan untuk uji coba klasifikasi.



Gambar di atas adalah halaman awal dari langkah pertama, yaitu memilih metode klasifikasi. Klik di mana saja untuk melanjutkan.

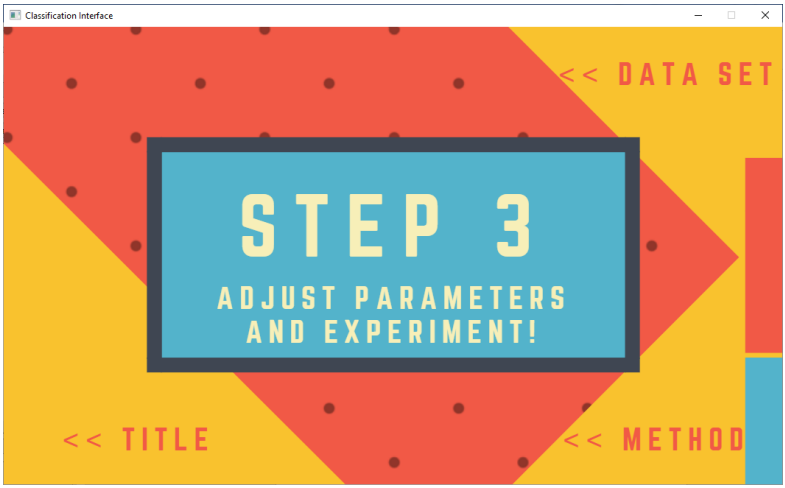
Langkah ini sangat sederhana, yaitu hanya perlu memilih salah satu dari empat metode klasifikasi yang tersedia. Klik pada salah satu dari empat kotak metode klasifikasi untuk melanjutkan.



Gambar di atas adalah dialog uji coba klasifikasi. Di sini digunakan metode klasifikasi kNN sebagai contoh. Sebagaimana gambar di atas, pada metode ini terdapat beberapa *menu dropdown* di bagian atas. Menu tersebut adalah parameter-parameter yang bisa diatur.

Pengguna dapat melakukan beberapa hal dalam dialog ini, yaitu:

1. Melakukan uji coba klasifikasi secara:
   a. Manual, dengan mengatur parameter pada *menu-menu dropdown* yang tersedia dan menekan tombol “Generate”.
   b. Acak, dengan menekan tombol “Random Generate”.
   c. *Brute force*, dengan menekan tombol “”.
   d. *Default*, dengan menekan tombol “”.
2. Menghapus *log* dengan menekan tombol “Clear Log”.
3. Menyimpan *log* dengan menekan tombol “Save Log.
4. Melihat penjelasan mengenai parameter yang digunakan dengan menekan tombol “?”.
5. Melihat tentang program dengan menekan tombol “?”



Di balik dialog uji coba klasifikasi, jendela utama akan menampilkan halaman terakhir, yaitu halaman langkah ketiga. Di sini anda dapat bernavigasi ke: halaman paling depan dengan menekan TITLE, halaman memilih *data set* dengan menekan DATA SET, halaman memilih metode klasifikasi dengan menekan METHOD.

Selesai!

Terima kasih karena telah membaca panduan program ini!

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Aldhiaz Fathra Daiva, lahir pada tanggal 24 Juli 1996 di Serang, Banten. Penulis menempuh pendidikan mulai dari SD Al-Azhar Syifa Budi YPWKS Cilegon (2002 – 2007), SMP Negeri 1 Cilegon (2007 – 2010) dan SMA Negeri 1 Kota Serang (2010-2013). Selama masa kuliah, penulis aktif dalam organisasi Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) dan Keluarga Muslim Informatika (KMI).

Dalam menyelesaikan pendidikan S1, penulis mengambil bidang minat Manajemen Informasi (MI). Penulis dapat dihubungi melalui email: dhiazfathra@gmail.com.