TUGAS AKHIR – IF184802

KLASIFIKASI MULTI LABEL GAYA BELAJAR VAK BERDASARKAN PERILAKU PEMBELAJARAN PADA E-LEARNING MENGGUNAKAN METODE SUPERVISED LEARNING

SAFIRA VANILLIA PUTRI
NRP 05111640000001

Dosen Pembimbing I
Dini Adni Navastara, S.Kom., M.Sc.

Dosen Pembimbing II
Dr. Umi Laili Yuhana, S.Kom., M.Sc.

Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

TUGAS AKHIR – IF184802

KLASIFIKASI MULTI LABEL GAYA BELAJAR VAK BERDASARKAN PERILAKU PEMBELAJARAN PADA E-LEARNING MENGGUNAKAN METODE SUPERVISED LEARNING

SAFIRA VANILLIA PUTRI
NRP 05111640000001

Dosen Pembimbing I
Dini Adni Navastara, S.Kom., M.Sc.

Dosen Pembimbing II
Dr. Umi Laili Yuhana, S.Kom., M.Sc.

Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

(Halaman ini sengaja dikosongkan)

UNDERGRADUATE THESIS – IF184802

MULTI LABEL CLASSIFICATION OF VAK LEARNING STYLES BASED ON LEARNING BEHAVIOR IN E-LEARNING USING THE SUPERVISED LEARNING METHOD

SAFIRA VANILLIA PUTRI
NRP 05111640000001

Dosen Pembimbing I
Dini Adni Navastara, S.Kom., M.Sc.

Dosen Pembimbing II
Dr. Umi Laili Yuhana, S.Kom., M.Sc.

Department of Informatics
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2020

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

KLASIFIKASI MULTI LABEL GAYA BELAJAR VAK BERDASARKAN PERILAKU PEMBELAJARAN PADA E-LEARNING MENGGUNAKAN METODE SUPERVISED LEARNING

TUGAS AKHIR

Diajukan untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh:
SAFIRA VANILLIA PUTRI
NRP: 05111640000001

Disetujui oleh Pembimbing Tugas Akhir:

1. Dini Adni Navastara, S.Kom., M.Sc. .....
   (NIP. 19851017 201504 2 001) (Pembimbing 1)

2. Dr. Umi Laili Yuhana, S.Kom., M.Sc. .....
   (NIP. 19790626 200501 2 002) (Pembimbing 2)

SURABAYA
Juni, 2020

(Halaman ini sengaja dikosongkan)

KLASIFIKASI MULTI LABEL GAYA BELAJAR VAK BERDASARKAN PERILAKU PEMBELAJARAN PADA E-LEARNING MENGGUNAKAN METODE SUPERVISED LEARNING

Nama Mahasiswa	**: Safira Vanillia Putri**
NRP	**: 05111640000001**
Departemen	**: Teknik Informatika, FTEIC-ITS**
Dosen Pembimbing 1	**: Dini Adni Navastara, S.Kom., M.Sc.**
Dosen Pembimbing 2	**: Dr. Umi Laili Yuhana, S.Kom., M.Sc.**

ABSTRAK

Penggunaan e-learning pada institusi pendidikan telah banyak dimanfaatkan sebagai media yang memudahkan proses pembelajaran. Di dalam e-learning telah tersedia berbagai macam modul, salah satunya data log. Dengan adanya data log, hal ini memungkinkan untuk melakukan analisis data yang dapat menghasilkan pengetahuan berguna bagi tenaga pendidik. Salah satunya adalah data log pada e-learning dianggap dapat menjadi preferensi dalam menentukan gaya belajar. Namun masalahnya, pada e-learning belum menyediakan fitur yang mengelola data log pelajar untuk mengetahui gaya belajarnya. Salah satu solusi untuk mengatasi masalah tersebut adalah membuat model untuk klasifikasi gaya belajar yang dibangun menggunakan supervised learning.

Tugas akhir ini membahas pembuatan model untuk klasifikasi multi label gaya belajar yang pada umumnya di Indonesia, yaitu gaya belajar Visual, Auditori, dan Kinestetik (VAK) dengan menggunakan metode supervised learning, antara lain k-Nearest Neighbours (k-NN), Support Vector Machine (SVM), dan Decision Trees (DTs). Data log yang digunakan merupakan data log mahasiswa kedokteran pada Learning Management System (LMS) Universitas Nairobi dengan spesifikasi data log standar LMS Universitas Nairobi. Proses pembuatan model yang dilakukan meliputi, praproses data log, pemilihan

fitur, pemisahan dataset, over-sampling, transformasi data, dan klasifikasi gaya belajar.

Pengujian dilakukan dalam empat skenario, yaitu pengujian pemisahan dataset, pengujian parameter pada supervised learning, pengujian pada jenis dataset, dan pengujian over-sampling pada data latih. Hasil uji coba optimal didapatkan pada model DTs dengan pengaturan parameter pengukuran kualitas split menjadi gini dengan akurasi sebesar 95,63% pada data latih yang dilakukan over-sampling.

Kata kunci: *Data log, Decision Trees,* Gaya belajar VAK, *k-Nearest Neighbors*, *Support Vector Machine*

MULTI LABEL CLASSIFICATION OF VAK LEARNING STYLES BASED ON LEARNING BEHAVIOR IN E-LEARNING USING THE SUPERVISED LEARNING METHOD

Student's Name : Safira Vanillia Putri
Student's ID : 05111640000001
Department : Informatics, Faculty of ELECTICS-ITS
First Advisor : Dini Adni Navastara, S.Kom., M.Sc.
Second Advisor : Dr. Umi Laili Yuhana, S.Kom., M.Sc.

ABSTRACT

The use of e-learning in educational institutions has been widely used as a medium that facilitates the learning process. In e-learning various modules are available, one of them is log data. With the log data, it is possible to conduct data analysis that can produce useful knowledge for educators. One of them is the log data on e-learning is considered to be a preference in determining learning styles. But the problem is, e-learning has not provided a feature that manages student log data to find out its learning style. One solution to overcome this problem is to make a model for the classification of learning styles that are built using supervised learning.

In this undergraduate thesis, a model for the classification of multi-label learning styles that are generally in Indonesia, namely Visual, Auditory, and Kinesthetic (VAK) learning styles using supervised learning methods, such as k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), and Decision Trees (DTs). The log data used is the medical student log data at the University of Nairobi's Learning Management System (LMS) with the LMS University Nairobi standard log data specifications. The model making process includes data log preprocessing, feature selection, dataset separation, over-sampling, data transformation, and learning style classification.

The test is carried out in four scenarios, namely the testing of dataset separation, testing parameters on supervised learning, testing on dataset types, and over-sampling testing on training data. Optimal test results are obtained with the DTs model with the criterion parameter setting being gini with an accuracy of 95.63% in the training data conducted over-sampling.

Keywords: ***Decision Trees, k-Nearest Neighbors, Log Data, Support Vector Machine, VAK Learning Style***

KATA PENGANTAR

Puji syukur penulis sampaikan kepada Tuhan yang Maha Esa karena berkat rahmat-Nya penulis dapat melaksanakan Tugas Akhir yang berjudul:

"KLASIFIKASI MULTI LABEL GAYA BELAJAR VAK BERDASARKAN PERILAKU PEMBELAJARAN PADA E-LEARNING MENGGUNAKAN METODE SUPERVISED LEARNING"

Terselesaikannya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan banyak pihak, oleh karena itu melalui lembar ini penulis ingin mengucapkan terima kasih dan penghormatan kepada:

1. Orang tua dan keluarga penulis, yang telah memberikan dukungan doa, moral, dan material kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
2. Ibu Dini Adni Navastara, S.Kom., M.Sc. dan Ibu Dr. Umi Laili Yuhana, S.Kom., M.Sc. selaku pembimbing I dan II yang telah membimbing, memberikan motivasi, dan masukan dalam menyelesaikan Tugas Akhir ini.
3. Ibu Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. selaku Kepala Departemen Teknik Informatika ITS dan seluruh dosen dan karyawan Departemen Teknik Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa kuliah.
4. Seluruh responden yang sudah membantu penulis untuk menentukan *ground truth* pada *data log* yang digunakan pada Tugas Akhir ini.
5. Teman-teman satu penelitian, Rifka, Farida, Fina, April, dan Sari yang telah mengajak penulis untuk memulai pengerjaan Tugas Akhir ini.

6. Muhammad Danial Ciptaul Khariri dan Modista Garsia yang telah menemani dan mendukung penulis selama perkuliahan hingga semester 8.
7. Ghifaroza Rahmadiana yang telah memberikan tips dalam pengerjaan Tugas Akhir ini.
8. Teman-teman semasa SMA, Nariza, Fuad, dan Andry yang telah mendukung dan menghibur selama pengerjaan Tugas Akhir ini.
9. Seluruh mahasiswa Teknik Informatika ITS angkatan 2016 yang telah menjadi teman penulis selama menjalani masa kuliah di Teknik Informatika ITS.
10. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa laporan Tugas Akhir ini masih jauh dari kata sempurna. Oleh karena itu dengan segala kerendahan hati penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan penulis kedepannya. Selain itu, penulis berharap laporan Tugas Akhir ini dapat berguna bagi pembaca secara umum.

Surabaya, Juni 2020

DAFTAR ISI

DAFTAR TABEL

DAFTAR KODE SUMBER

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Sampai saat ini, penggunaan *e-learning* pada institusi pendidikan telah banyak dimanfaatkan sebagai media yang memudahkan proses pembelajaran. *E-learning* sendiri dikenal sebagai sistem pembelajaran berdasarkan pengajaran formal yang menggunakan bantuan sumber daya elektronik. Proses pembelajaran dapat berbasis di dalam maupun di luar ruang kelas [1]. Di dalam *e-learning* telah tersedia berbagai macam modul, seperti manajemen pengguna, kelas, konten, sistem penilaian, bahkan *data log*. Dengan adanya *data log*, hal ini memungkinkan untuk melakukan analisis data yang dapat menghasilkan pengetahuan berguna bagi tenaga pendidik.

Berdasarkan *data log* pelajar, terdapat perbedaan terhadap preferensi belajar dan perilaku kognitif mengenai materi pembelajaran yang mereka gunakan. Gaya belajar didefinisikan sebagai preferensi pembelajaran yang paling disukai [2]. Dari banyaknya model gaya belajar yang ada, yang paling umum dikenal adalah model gaya belajar dengan tiga modalitas, yaitu visual, auditori, dan kinestetik atau yang populer dengan sebutan VAK. Pengkategorian ini tidak berarti bahwa pelajar hanya memiliki salah satu karakteristik gaya belajar tertentu, sehingga dapat dimungkinkan untuk memiliki lebih dari satu jenis gaya belajar [3]. Namun masalahnya, pada *e-learning* belum menyediakan fitur yang mengelola *data log* pelajar untuk mengetahui gaya belajarnya, sehingga tenaga pendidik melakukan pengajaran dengan cara yang merata kepada semua pelajarnya.

Salah satu solusi yang sudah ada untuk menyelesaikan permasalahan tersebut adalah model yang dibangun menggunakan algoritma *k-Nearest Neighbours* (k-NN) berdasarkan *Felder Silverman Learning Style Model* (FSLSM) [2]. FSLSM mengelompokkan gaya belajar pada empat dimensi yaitu, *active* atau *reflective*, *sensing* atau *intuitive*, *sequential* atau *global*, dan

visual atau verbal, sehingga *data log* diekstraksi ke dalam fitur-fitur yang diperlukan dalam klasifikasi FSLSM. Model diuji menggunakan dataset dari *data log* mahasiswa kedokteran pada *Learning Management System* (LMS) Universitas Nairobi, Kenya. Hasilnya menunjukkan bahwa model yang dibuat memberikan hasil yang menjanjikan, namun penelitian masih berlanjut untuk menghasilkan model yang lebih akurat.

Berdasarkan hal-hal di atas, pada Tugas Akhir ini akan diimplementasikan metode *supervised learning* untuk klasifikasi multi label gaya belajar yang pada umumnya di Indonesia, yaitu gaya belajar VAK dimana menggunakan *data log* dari LMS Universitas Nairobi dengan melakukan penyesuaian fitur.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana cara melakukan ekstraksi *data log* menjadi fitur-fitur yang sesuai dengan model gaya belajar?
2. Bagaimana cara melakukan klasifikasi multi label gaya belajar VAK?
3. Bagaimana evaluasi untuk mengukur kelayakan hasil klasifikasi gaya belajar VAK pada sistem yang dibangun?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada Tugas Akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Implementasi Tugas Akhir ini menggunakan bahasa pemrograman Python 3.
2. Dataset pelatihan dan pengujian merupakan *data log* mahasiswa kedokteran pada LMS Universitas Nairobi dengan spesifikasi *data log* standar LMS Universitas Nairobi.
3. Pemberian label (*ground truth*) dilakukan secara *majority vote* berdasarkan hasil kuesioner pada 8 orang.

4. Klasifikasi gaya belajar VAK terdapat lima kelas, yaitu visual (V), auditori (A), kinestetik (K), VK, dan AK.

1.4 Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah membuat model untuk klasifikasi multi label gaya belajar VAK menggunakan metode *supervised learning* dengan mengekstraksi *data log* pada *e-learning.*

1.5 Manfaat

Tugas akhir ini diharapkan menjadi sistem yang mampu menghasilkan model klasifikasi multi label gaya belajar VAK yang dapat membantu tenaga pendidik dalam menyampaikan materi sesuai gaya belajarnya masing-masing kepada pelajar.

1.6 Metodologi

Pembuatan Tugas Akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1.6.1 Penyusunan Proposal Tugas Akhir

Tahapan awal dari Tugas Akhir ini adalah penyusunan Proposal Tugas Akhir yang berisi pendahuluan, deskripsi dan gagasan metode-metode yang dibuat dalam Tugas Akhir. Pendahuluan ini terdiri atas latar belakang diajukannya Tugas Akhir, rumusan masalah dan batasan masalah yang ditetapkan, serta manfaat dari hasil pembuatan Tugas Akhir ini. Selain itu, dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan Tugas Akhir. Terdapat pula subbab jadwal kegiatan yang menjelaskan jadwal pengerjaan Tugas Akhir.

1.6.2 Studi Literatur

Pada tahap ini dilakukan pencarian literatur berupa jurnal yang digunakan sebagai referensi untuk pengerjaan Tugas Akhir ini. Literatur yang dipelajari pada pengerjaan Tugas Akhir ini berasal dari jurnal ilmiah yang diambil dari berbagai sumber di

internet, beserta berbagai literatur *online* tambahan terkait gaya belajar VAK, *knowledge dicovery in databases* (KDD), dan metode *supervised learning*.

1.6.3 Implementasi Perangkat Lunak

Pada tahap ini akan dilaksanakan implementasi metode dan algoritma yang telah direncanakan. Implementasi sistem menggunakan Python 3 sebagai bahasa pemrograman serta *library* pendukung seperti *scikit-learn, pandas, numpy,* dan pendukung lainnya.

1.6.4 Pengujian dan Evaluasi

Tahap pengujian dan evaluasi dilakukan untuk mengetahui hasil dan performa algoritma yang telah diimplementasikan. Dataset yang digunakan dalam proses pengujian berupa sampel *data log* dari LMS Universitas Nairobi yang diambil selama periode 15 minggu perkuliahan. Pada tahap evaluasi akan dilakukan evaluasi pada model *supervised learning* dari sisi akurasi, presisi, *recall*, dan *f1-score*.

1.6.5 Penyusunan Buku

Pada tahap ini dilakukan penyusunan buku yang menjelaskan seluruh konsep, teori dasar dari metode yang digunakan, implementasi, serta hasil yang telah dikerjakan sebagai dokumentasi dari pelaksanaan Tugas Akhir.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan Tugas Akhir adalah sebagai berikut:

Bab I Pendahuluan

Bab ini berisi penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan Tugas Akhir.

Bab II Tinjauan Pustaka

Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Tugas Akhir ini. Secara garis besar, bab ini berisi tentang metode *supervised learning* dan *library* yang digunakan.

Bab III Perancangan Sistem

Bab ini berisi pembahasan mengenai perancangan dari metode *supervised learning* yang digunakan untuk klasifikasi multi label gaya belajar VAK pada *data log*.

Bab IV Implementasi

Bab ini membahas implementasi dari perancangan yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode sumber yang digunakan dalam proses implementasi.

Bab V Uji Coba dan Evaluasi

Bab ini membahas tahapan uji coba, kemudian hasil uji coba dievaluasi terhadap kinerja dari sistem yang dibangun.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses dan tertulis saat pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.

(Halaman ini sengaja dikosongkan)

BAB II
TINJAUAN PUSTAKA

Bab ini membahas mengenai teori-teori dasar yang digunakan dalam Tugas Akhir. Teori-teori tersebut mengenai gaya belajar VAK, *data log*, *data mining*, *supervised learning*, dan beberapa teori lain yang mendukung pembuatan Tugas Akhir. Penjelasan ini bertujuan untuk memberikan gambaran umum dan diharapkan dapat mendukung sistem yang dibangun.

2.1 Gaya Belajar VAK

Gaya belajar telah dikembangkan berbagai pakar di Amerika, antara lain *Environmental Learning Styles, Felder-Silverman Learning Style Model, Grasha-Riechmann Student Learning Styles, The Gregoric-Butler Theory, Kolb's Learning Style Model, Herrmann Brain Dominance Instrument, Levine's Neurodevelopmental Profiles, The Myers-Briggs Type Indicator, Multiple Intelligences Theory, Media or Sensory Channel, R J Riding's Dimensions, Styles of Mental Self-Government, Priscilla L. Vail's Learning Styles.*

Gaya belajar dapat didefinisikan dengan berbagai cara, tergantung pada perspektif seseorang. Keefe (1979) mendefinisikan gaya belajar sebagai gabungan dari karakteristik kognitif, afektif, dan faktor fisiologis yang berfungsi sebagai indikator yang relatif stabil tentang bagaimana pelajar merasakan, berinteraksi dengan, dan merespon lingkungan belajar.

Brown (2000) mendefinisikan gaya belajar sebagai cara seseorang mempersepsikan dan memproses informasi dalam situasi belajar. Brown berpendapat bahwa preferensi gaya belajar merupakan salah satu aspek gaya belajar, dan mengacu pada pilihan satu situasi belajar atau kondisi di atas preferensi yang lain.

Dari beberapa definisi tersebut, dapat dipahami bahwa hasil belajar seseorang dipengaruhi oleh cara mereka menyerap informasi ketika pembelajaran dalam konteks apapun berlangsung, apakah itu belajar di dalam kelas, atau di luar kelas. Dengan kata

lain, secara sadar atau tidak sadar, saat seseorang tersebut sedang menyerap informasi, di situlah pembelajaran secara umum terjadi.

Dari sekian gaya belajar ini, yang paling sederhana adalah gaya belajar visual-auditori-kinestetik atau populer disebut VAK karena lebih mudah diukur dan cepat untuk mendapatkan gambaran umum tentang gaya belajar seseorang, baik itu pelajar maupun tenaga kependidikan [4].

Konsep gaya belajar VAK yang asli, pertama kali dikembangkan oleh psikolog dan spesialis pengajaran anak yang bernama Fernald, Keller, Orton, Gillingham, Stillman dan Montessori, mulai tahun 1920-an. Gaya belajar ini menggunakan tiga penerima sensor utama, yaitu visual, auditori, dan kinestetik untuk menentukan gaya belajar yang dominan [5]. Namun kemungkin lain yang terjadi adalah seseorang dapat menyerap informasi melalui perpaduan visual-auditori, visual-kinestetik, auditori-kinestetik atau perpaduan ketiganya secara merata.

Seseorang bergaya belajar visual, seperti namanya, belajar paling baik dengan menggunakan mata mereka. Mereka lebih suka melihat bagaimana melakukan sesuatu daripada berdiskusi. Gaya belajar visual, berarti seseorang belajar dengan melihat dan mencermati.

Seseorang bergaya belajar auditori memantapkan pemahaman ketika mereka mendengar informasi. Mereka biasanya mengikuti arah dengan baik, berkonsentrasi lebih baik dengan musik atau gerakan di latar belakang, dan mengulangi semuanya kembali untuk memastikan mereka mendapatkan informasi dengan pemahaman baik. Gaya belajar auditori, berarti siswa belajar dengan mendengar dan menyimak secara intensif.

Sedangkan seseorang yang memiliki gaya belajar kinestetik dapat belajar paling baik dengan berinteraksi atau mengalami hal-hal di sekitarnya. Mereka mendapat manfaat dari keterlibatan langsung, daripada mendengarkan ceramah atau membaca dari sebuah buku. Mereka suka melakukan hal-hal dan menggunakan tubuh mereka untuk mengingat fakta. Gaya belajar kinestetik, berarti siswa belajar dengan menyentuh dan melakukan [4].

Berdasarkan pemaparan gaya belajar tersebut, media pembelajaran maupun penugasan pada gaya belajar VAK dapat diberikan seperti pada Tabel 2.1. Pada Tugas Akhir ini, gaya belajar akan diklasifikasi secara multi label.

Tabel 2.1 Media pada Gaya Belajar VAK [6]

Karakteristik Gaya Belajar	**Media Pembelajaran**	**Media Penugasan**	**Referensi**
Visual	Penggunaan klip video, diagram, gambar atau peta	Peta konsep diagram, konstruksi presentasi PowerPoint, membaca	Peta, diagram, gambar, artikel
Auditori	Mendengarkan kuliah, klip audio	Proyek dengan komponen audio, *interview*, seminar, memberikan laporan dan pidato, PowerPoint dengan komponen audio	Klip video atau audio dari koleksi media
Kinestetik	Latihan di kelas, meminta partisipasi dalam simulasi dan demo kelas	Kuis asesmen mandiri, membuat model, presentasi, demo	Kunjungan lapangan, kerja kelompok

2.2 E-learning

E-learning adalah proses pembelajaran yang difasilitasi dan didukung oleh penggunaan teknologi informasi dan komunikasi. Sementara *Learning Management System* (LMS) merupakan sistem perangkat lunak yang digunakan untuk mengelola pembelajaran dan tersedia dalam berbagai bentuk, seperti situs web yang mengumpulkan semua tautan dan direktori video, serta menyediakan fitur kuis untuk membimbing pelajar ke tahap selanjutnya [7]. Sehingga pada dasarnya, semua pembelajaran LMS adalah *e-learning*, tetapi tidak semua *e-learning* dilakukan dalam LMS [8].

Selama pelajar melakukan pengaksesan, sistem menghasilkan volume data yang sangat besar yang dapat dianalisis menggunakan metode *machine learning* untuk menghasilkan pengetahuan yang berguna bagi tenaga pendidik. Data-data ini termasuk statistik akses, detail *login*, dan *log server*. Tenaga pendidik dalam banyak kasus mengandalkan tampilan cepat (*quick view*) dari data pembelajaran dasar untuk memantau kemajuan [2]. Pada tugas akhir ini akan memanfaatkan *data log* pada *e-learning* untuk menjadi salah satu pengetahuan yang berguna bagi tenaga pendidik.

Dalam komputasi, *data log* merupakan data yang merekam peristiwa yang terjadi dalam sistem operasi atau perangkat lunak yang sedang berjalan. Peristiwa yang terekam tersebut dapat memberikan jejak audit yang digunakan untuk memahami aktivitas sistem maupun mendiagnosis masalah [9]. Pada Tugas Akhir akan menggunakan *data log* dari LMS Universitas Nairobi, yang terdiri atas empat atribut antara lain, user_id, module_id, module_name, dan total_time. Sampel *data log* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Sampel *Data Log* dari LMS Universitas Nairobi

user_id	module_id	module_name	total_time
248291	137	Unit 4 Review Questions	00:00:20
250412	3	HCH 100: BEHAVIOURAL SCIENCES - MODULE 4 - SOCIOLOGY	00:00:07
250412	5	MODULE 4: SOCIOLOGY	00:00:02

2.3 Penemuan Gaya Belajar pada E-learning

Menjawab banyak pertanyaan dari kuesioner gaya belajar cukup memakan waktu. Dengan demikian, banyak peneliti merekomendasikan untuk menyimpulkan dan memperbarui gaya belajar para pelajar secara langsung dari interaktivitas mereka dengan LMS [2], [10]. Melalui pemahaman mendalam tentang model VAK dan LMS yang digunakan di Universitas Nairobi, *data log* dapat digunakan untuk deteksi gaya belajar dari pelajar secara otomatis.

Rujukan [11] membuat pemetaan antara gaya belajar VAK dan perilaku pembelajaran *online* pada *e-learning*. Mereka menggunakan fitur umum pada *e-learning* daripada fitur tertentu yang berjumlah 25 fitur. Dengan menyesuaikan beberapa kesamaan perilaku pembelajaran pada *data log* Tugas Akhir ini, didapatkan hasil pemetaan perilaku pembelajaran pada *e-learning* terhadap gaya belajar VAK sebanyak 11 fitur seperti pada Tabel 2.4. Pengurangan jumlah fitur ini dilakukan karena data yang digunakan pada Tugas Akhir merupakan *data log* saja. Sedangkan pada rujukan [11], melibatkan perilaku pembelajaran diluar ketersediaan informasi *data log* seperti pada Tabel 2.3.

Tabel 2.3 Perilaku Pembelajaran *online* yang Tidak Digunakan

Perilaku Pembelajaran *Online*
Jumlah jawaban benar pada *assesment* soal grafis
Jumlah jawaban benar pada *assesment* soal teks
Cara menjawab secara acak (*random*)
Cara menjawab secara berurutan (*sequential*)
Jumlah pertanyaan yang tidak terjawab
Jumlah pertanyaan yang terjawab
Hasil pada *exercise*
Lamanya waktu kunjungan pada konten grafis
Lamanya waktu kunjungan pada konten audio
Lamanya waktu kunjungan pada konten *example*
Lamanya waktu kunjungan pada konten *outline*
Lamanya waktu kunjungan pada konten *review question*
Lamanya waktu kunjungan pada konten *case study*
Lamanya waktu kunjungan pada konten forum

Tabel 2.4 Hasil Pemetaan Perilaku Pembelajaran *online* terhadap Gaya Belajar VAK

Perilaku Pembelajaran *Online*	**Gaya Belajar**		
	V	**A**	**K**
Kunjungan pada konten grafis (*figure, illustration, chart*, video)	√		
Kunjungan pada konten audio (*recording*)		√	
Kunjungan pada *example*		√	
Kunjungan pada *outline*		√	
Kunjungan pada *review question, activity, case study,* forum			√
Kunjungan pada *exercise*			√
Lamanya waktu kunjungan pada *exercise*			√
Kunjungan pada *assesment*			√
Lamanya waktu kunjungan pada *assesment*			√
Kunjungan pada *self-test*			√
Lamanya waktu kunjungan pada *self-test*			√

Dari hasil pemetaan pada Tabel 2.4, dapat diketahui jumlah kunjungan pelajar pada tiap konten ($LO_{VisitedContent}$). Selain itu juga dapat diidentifikasi jumlah total konten dalam kursus ($LO_{TotalContent}$). Faktor-faktor ini merupakan rasio kunjungan untuk tiap konten ($R_{VisitedContent}$) (2.1).

$$R_{VisitedContent} = \frac{\sum LO_{VisitedContent}}{\sum LO_{TotalContent}} \quad (2.1)$$

Demikian pula, dengan menganalisis waktu yang dihabiskan dalam mengunjungi konten (lamanya waktu kunjungan seperti yang dimaksud pada Tabel 2.4), instruktur atau pakar dapat memperkirakan waktu yang diharapkan untuk dihabiskan pada setiap konten ($TES_{Content}$). Dari *data log*, dimungkinkan untuk mengetahui waktu yang dihabiskan untuk setiap konten ($TS_{Content}$). Faktor-faktor ini merupakan rasio waktu yang dihabiskan pada konten ($R_{TimeSpentContent}$) (2.2) [10].

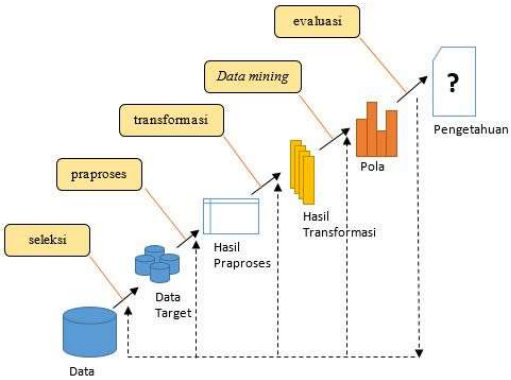
$$R_{TimeSpentContent} = \frac{\sum TS_{Content}}{\sum TES_{Content}} \quad (2.2)$$

Pada Tabel 2.4 dapat dilihat bahwa $R_{TimeSpentContent}$ hanya digunakan pada *assesment*, *exercise*, dan *self-test* dikarenakan memiliki waktu batasan yang jelas untuk digunakan pada $TES_{Content}$. Sedangkan konten lainnya tidak memiliki waktu batasan yang dapat digunakan pada $TES_{Content}$ [12].

2.4 Knowledge Discovery in Databases

Knowledge Discovery in Databases (KDD) merupakan proses pemetaan data tingkat rendah ke dalam bentuk lain yang lebih ringkas atau lebih bermanfaat. KDD mengacu pada keseluruhan proses menemukan pengetahuan yang berguna dari data, dan *data mining* mengacu pada langkah tertentu dalam proses ini [13]. Untuk proses KDD secara keseluruhan, dapat dilihat pada Gambar 2.1. Pada Tugas Akhir ini, akan menggunakan beberapa proses KDD untuk mengubah *data log* menjadi informasi gaya

belajar, seperti praproses, transformasi data, *data mining*, dan evaluasi.



Gambar 2.1 Proses pada KDD

2.4.1 Praproses

Praproses melibatkan penghapusan *outlier* dan data yang hilang atau tidak lengkap, maupun penggabungan data. Umumnya, praproses mengurangi kesalahan dan meningkatkan kualitas data. Sudah diketahui bahwa proses memperbaiki kesalahan dalam data dan menghilangkannya cukup memakan waktu dan melibatkan proses yang membosankan, namun hal ini tidak dapat diabaikan [14]. Pada Tugas Akhir ini, praproses yang dilakukan antara lain, mengecilkan huruf (*case folding*), menghilangkan tanda baca dan angka, serta penghapusan *outlier*. Pada penghapusan *outlier* akan digunakan metode *Median Absolute Deviation* (MADe).

2.4.2 Transformasi Data

Data yang telah dilakukan praproses, akan ditransformasikan menjadi bentuk yang sesuai untuk proses *data mining*. Data dikonsolidasikan sehingga proses *data mining* lebih efisien dan polanya lebih mudah dipahami [15]. Pada Tugas Akhir ini, digunakan salah satu strategi dalam transformasi data, yaitu normalisasi dengan metode min-max.

2.4.3 Data Mining

Data mining merupakan proses untuk mengidentifikasi pola dan pengetahuan dari sejumlah data yang besar. Dalam langkah ini, algoritma diterapkan untuk mengekstrak pola data [15]. Sebelum menerapkan algoritma, penting untuk mengetahui tujuan dari proses KDD yang dilakukan agar metode *data mining* yang dipilih adalah tepat.

Oleh karena itu, pada Tugas Akhir ini akan menggunakan teknik klasifikasi dengan menggunakan *supervised learning*.

2.4.4 Evaluasi

Langkah terakhir adalah evaluasi hasil dari tahap sebelumnya. Hal ini dapat membantu memperbaiki pengetahuan yang diperoleh atau mengubah pengetahuan menjadi bentuk yang jelas bagi pengguna. Pada tahap ini, pola data yang diekstraksi divisualisasikan untuk tinjauan lebih lanjut [15]. Untuk mengukur performa pada teknik *data mining* yang digunakan, Tugas Akhir ini menggunakan metrik penilaian akurasi, presisi, *recall*, dan *f1-score*.

2.5 Outlier

Secara umum, dapat dikatakan bahwa *outlier* adalah dislokasi poin asli dari posisi normal. Ada beberapa penyebab berbeda untuk terjadinya *outlier* antara lain, kesalahan alat ukur, kesalahan perhitungan manusia, perilaku penipuan, penyimpangan alami dalam grafik, atau juga bisa merupakan hasil dari hipotesis yang salah yang dihasilkan oleh peneliti. Adapun kesalahan hasil dalam dataset juga merupakan hasil dari *outlier* [16]. Meskipun *outlier* dianggap sebagai kesalahan atau *noise*, bisa saja *outlier* dapat membawa informasi penting. *Outlier* yang terdeteksi merupakan kandidat untuk data yang menyimpang atau mungkin sebaliknya. Oleh karena itu, sangat penting untuk mengidentifikasi *outlier* sebelum pemodelan dan analisis.

Berdasarkan beberapa metode deteksi *outlier* yang telah dikembangkan, salah satu metode yang dipertimbangkan adalah

Median Absolute Deviation (MADe) [16], [17]. MADe merupakan salah satu metode yang *robust* untuk deteksi *outlier*. Metode ini dikembangkan oleh Ratcliffe (1993), yang mana sebagian besar tidak terpengaruh oleh keberadaan nilai ekstrim dalam data. Pendekatan ini mirip dengan metode standar deviasi, namun dalam metode ini menggunakan median dan *Median Absolute Deviation* (MAD) sebagai ganti dari *mean* dan standar deviasi [17].

Tahap pertama yang dilakukan adalah menghitung median, selanjutnya untuk setiap titik data, hitung jarak antara nilai tersebut dan mediannya seperti pada (2.3). Setelah itu, MAD perlu dikalikan dengan 1,4826 untuk memastikannya mendekati standar deviasi aktual seperti pada (2.4) [18].

$$MAD = median\{|x_i - median(x)|, i = 1,2,..,n\} \tag{2.3}$$

$$MADe = 1.4826 \times MAD \tag{2.4}$$

Dimana:

$median$ = nilai tengah
x_i = nilai data ke-i
i = jumlah data

Median dan MADe digunakan untuk mendeteksi *outlier* dengan menerapkan persamaan pada (2.5) atau (2.6).

$$Metode\ 2MADe = median\ \pm 2MADe \tag{2.5}$$

$$Metode\ 3MADe = median\ \pm 3MADe \tag{2.6}$$

Pada Tugas Akhir ini, digunakan persamaan seperti (2.6) untuk mendeteksi *outlier* pada salah satu atribut *data log* (total_second).

2.6 Data Tidak Seimbang

Data tidak seimbang merupakan suatu keadaan dimana distribusi kelas data tidak seimbang, yang mana jumlah kelas data (*instance*) yang satu lebih sedikit atau lebih banyak dibanding dengan jumlah kelas data lainnya. Kelompok kelas data yang lebih sedikit dikenal sebagai kelompok minoritas (*minority*), sedangkan kelompok kelas data yang lainnya disebut dengan kelompok mayoritas (*majority*).

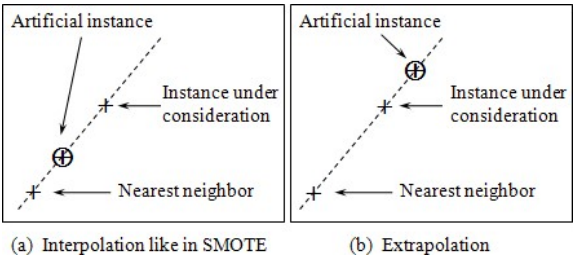
Klasifikasi pada data dengan kelas tidak seimbang merupakan masalah utama pada *machine learning* dan *data mining*, misalnya pada masalah medis, masalah klasifikasi teks, dan sosial media. Jika bekerja pada data tidak seimbang, hampir semua algoritma klasifikasi akan menghasilkan akurasi yang jauh lebih tinggi untuk kelas mayoritas daripada kelas minoritas. Perbedaan ini merupakan suatu indikator performa klasifikasi yang buruk. Pada beberapa kasus, kelas minoritas justru lebih penting untuk diidentifikasi daripada kelas mayoritas [19].

Beberapa metode untuk mengatasi ketidakseimbangan kelas dapat dibagi menjadi tiga kategori. Pertama, dengan teknik tingkat data yang berusaha menyeimbangkan distribusi data dengan metode *over-sampling* dan *under-sampling*. Kedua, pendekatan tingkat algoritma yaitu dengan mengembangkan algoritma baru atau memodifikasi metode yang ada untuk memperhitungkan arti dari kelas minor. Ketiga, yaitu dengan mengombinasikan pendekatan algoritma dan pendekatan level data [20].

Untuk mengatasi ketidakseimbangan kelas, maka pada Tugas Akhir ini menggunakan salah satu variasi dari *Synthetic Minority Over-sampling Technique* (SMOTE), yaitu *Borderline Over-sampling* (BOS) yang menggunakan standar algoritma *Support Vector Machine* (SVM).

BOS merupakan metode *over-sampling* dengan SVM sebagai pengklasifikasi dasar untuk menangani masalah ketidakseimbangan data. Pada BOS, area *borderline* diperkirakan oleh *support vectors* yang diperoleh setelah melakukan pelatihan pada SVM dengan set pelatihan asli. *Instances* baru akan dibuat

secara acak di sepanjang garis yang bergabung dengan masing-masing kelas *support vectors* minoritas dengan sejumlah tetangga terdekat menggunakan teknik interpolasi (Gambar 2.2a) atau ekstrapolasi (Gambar 2.2b) tergantung pada kepadatan *instance* kelas mayoritas di sekitarnya [21].



Gambar 2.2 Menggunakan teknik interpolasi (a) dan ekstrapolasi (b) untuk membuat *instance* buatan [21]

Jika *instance* kelas mayoritas dihitung kurang dari setengah tetangga terdekatnya, *instance* baru akan dibuat dengan ekstrapolasi seperti pada (2.7) untuk memperluas area kelas minoritas ke arah kelas mayoritas. Dalam kasus sebaliknya, karena kepadatan contoh kelas mayoritas, alih-alih memperluas area, area *borderline* saat ini dari kelas minoritas akan dikonsolidasikan dengan cara yang mirip dengan SMOTE seperti pada (2.8).

$$x_{new}^{+} = sv_i^{+} + \rho(sv_i^{+} - nn[i][j]) \tag{2.7}$$

$$x_{new}^{+} = sv_i^{+} + \rho(nn[i][j] - sv_i^{+}) \tag{2.8}$$

Dimana:

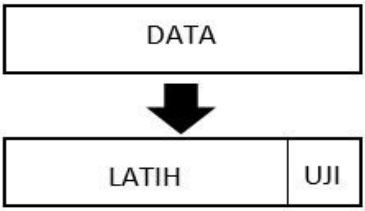
x_{new}^{+} = data sintesis yang akan diciptakan
sv_i^{+} = set *support vectors* positif
ρ = nilai random antara 0 dan 1
nn = k tetangga terdekat positif dari setiap sv positif

2.7 Cross-validation

Cross-validation merupakan teknik yang mana melakukan pelatihan model menggunakan subset dari kumpulan data dan kemudian dievaluasi menggunakan subset komplementer dari kumpulan data [22]. Dari beberapa jenis *cross-validation,* Tugas Akhir ini akan menggunakan metode *hold-out* dan *k-fold.*

2.7.1 Hold-out

Hold-out merupakan jenis *cross-validation* yang paling simpel, yaitu dengan membagi dataset menjadi set latih dan uji seperti pada Gambar 2.3. Set pelatihan merupakan model yang dilatih, sedangkan set pengujian digunakan untuk melihat seberapa baik kinerja model tersebut pada data yang tidak terlihat [23]. Biasanya metode ini melibatkan pemisahan dataset menjadi set uji 20-30% dan sisanya sebagai set pelatihan. Angka-angka ini dapat bervariasi, tetapi persentase yang lebih besar pada data pengujian akan membuat model lebih rentan terhadap kesalahan karena memiliki lebih sedikit pelatihan, sementara persentase yang lebih kecil pada data pengujian dapat memberikan model menjadi bias yang tidak diinginkan terhadap data pelatihan [22].

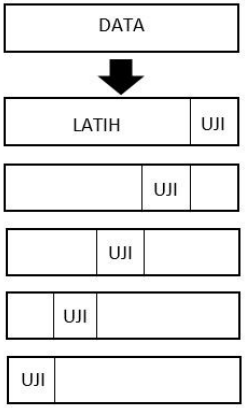


Gambar 2.3 *Hold-out cross-validation*

2.7.2 k-fold Cross-validation

k-fold cross-validation merupakan pemisahan dataset yang dibagi secara acak sebanyak k kelompok. Salah satu kelompok digunakan sebagai set uji dan sisanya digunakan sebagai set pelatihan. Model dilatih pada set pelatihan dan dievaluasi pada set

uji. Kemudian proses ini diulangi sampai masing-masing kelompok digunakan sebagai set uji. Sebagai contoh, pada Gambar 2.4 menggunakan 5-*fold cross-validation*, sehingga dataset akan dibagi menjadi 5 kelompok, dan model akan dilatih dan diuji 5 kali terpisah sehingga setiap kelompok akan mendapat kesempatan untuk menjadi kelompok uji [23].



Gambar 2.4 5-*fold cross-validation*

2.8 Normalisasi

Normalisasi adalah teknik yang sering diterapkan sebagai bagian dari persiapan data untuk *machine learning*. Tujuan normalisasi adalah untuk mengubah nilai-nilai kolom numerik dalam dataset untuk menggunakan skala umum, tanpa mendistorsi perbedaan dalam rentang nilai atau kehilangan informasi. Normalisasi juga diperlukan untuk beberapa algoritma untuk memodelkan data dengan benar [24]. Pada Tugas Akhir ini, salah satu teknik normalisasi yang digunakan adalah min-max.

Normalisasi min-max adalah salah satu cara paling umum untuk menormalkan data. Untuk setiap fitur, nilai minimum dari fitur tersebut ditransformasikan menjadi 0, nilai maksimum ditransformasikan menjadi 1, dan setiap nilai lainnya

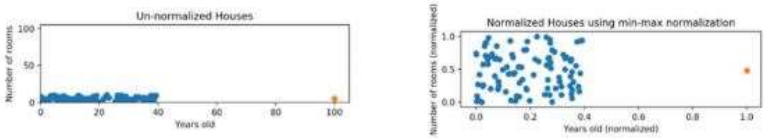
ditransformasikan menjadi desimal antara 0 dan 1 dengan mengikuti (2.6) [25].

$$x_{normalisasi} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.9}$$

Dimana:

x = nilai data sebelum normalisasi
x_{min} = nilai data minimum
x_{max} = nilai data maksimum

Pada Gambar 2.5 dapat dilihat perbandingan data sebelum dan sesudah dinormalisasi. Data yang sudah dinormalisasi menunjukkan bahwa metode min-max tidak mengubah skala data aslinya.



Gambar 2.5 Data sebelum (kiri) dan sesudah normalisasi min-max (kanan) [25].

2.9 Supervised Learning

Supervised learning adalah tipe pembelajaran dimana *label class* telah didefinisikan sebelumnya. Atau dengan kata lain, kita mempunyai variabel *input* dan *output* yang menggunakan satu algoritma atau lebih dengan tujuan untuk memperkirakan fungsi pemetaannya, sehingga ketika mempunyai *input* baru, dapat diprediksi *output* untuk *input* tersebut.

Proses dari sebuah algoritma pembelajaran dari data latih dapat diumpamakan sebagai seorang guru yang mengawasi (*supervising*) proses belajar. Guru tahu jawaban yang benar dan algoritma secara iteratif membuat prediksi pada data latih dan

dikoreksi oleh guru tersebut. Pembelajaran berhenti ketika algoritma mencapai level performansi yang diterima.

Permasalahan pada *supervised learning* dapat dikelompokkan menjadi masalah regresi (*regression problem*) dan masalah klasifikasi (*classification problems*) [26]. Sesuai dengan permasalahan klasifikasi pada Tugas Akhir ini, maka digunakan beberapa algoritma yang dikembangkan pada *supervised learning* antara lain, *k-Nearest Neigbors* (k-NN), *Support Vector Machine* (SVM), dan *Decision Trees* (DTs).

2.9.1 k-Nearest Neighbors

k-Nearest Neighbors (k-NN) merupakan algoritma pembelajaran yang bersifat non-parametrik dan *lazy*. Non-parametrik berarti tidak ada asumsi untuk distribusi data yang mendasarinya. Dengan kata lain, struktur model ditentukan dari dataset. Hal ini sangat membantu dalam praktik dimana sebagian besar dataset di dunia nyata tidak mengikuti asumsi teoritis matematika. Sedangkan *lazy* berarti bahwa ia tidak menggunakan titik data pelatihan untuk melakukan *generalization*. Ini berarti fase pelatihannya berlangsung cukup cepat, namun kurangnya *generalization* berarti bahwa k-NN menyimpan semua data pelatihan, sehingga semua data pelatihan diperlukan selama fase pengujian [27].

Dalam algoritma k-NN, untuk setiap titik data uji akan dilihat titik data pelatihan terdekat sejumlah k dengan diukur berdasarkan salah satu fungsi jarak pada (2.10), atau (2.11), atau (2.12). Kemudian mengambil kelas yang paling sering muncul (*majority vote*), lalu menetapkan kelas tersebut ke data uji [28].

$$Jarak\ Euclidean = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \tag{2.10}$$

$$Jarak\ Manhattan = \sum_{i=1}^{k} |x_i - y_i| \tag{2.11}$$

$$Jarak\ Minkowski = (\sum_{i=1}^{k}(|x_i - y_i|)^p)^{\frac{1}{p}} \quad (2.12)$$

Dimana:

k = jumlah dimensi
x_i, y_i = titik data
p = $l_p - norm$

2.9.2 Support Vector Machine

Support Vector Machine (SVM) dikembangkan oleh Boser, Guyon, Vapnik, dan pertama kali dipresentasikan pada tahun 1992 di *Annual Workshop on Computational Learning Theory*. Konsep kerja SVM yaitu berusaha menemukan *hyperplane* terbaik dalam memisahkan dua buah kelas pada *input space* dengan cara mengukur margin *hyperplane* tersebut dengan mencari titik masksimalnya. Prinsip dasar SVM adalah *linear classifier* dengan memasukkan konsep *kernel trick* pada ruang kerja berdimensi tinggi [29].

Pada SVM linear, misalkan data latih direpresentasikan seperti (2.13) dapat dipisahkan oleh suatu *hyperplane* (2.14). Untuk suatu data latih, dapat diperoleh lebih dari satu *hyperplane* seperti contoh pada Gambar 2.6.

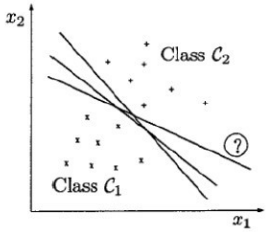
$$(x_1, y_1), \dots, (x_k, y_k), x_i \in R^n, y_i \in \{+1, -1\}, \quad i = 1,2,..l \quad (2.13)$$

$$x_1, y_1 - b = 0 \quad (2.14)$$

Dimana:

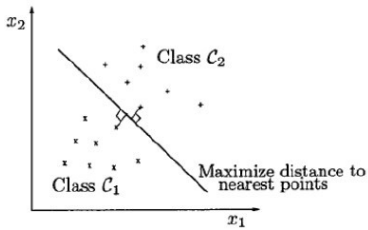
$x_i \in R^n$ = data yang tersedia

$y_i \in \{+1, -1\}$ = label masing-masing
$+1, -1$ = dua buah kelas
l = banyaknya data
b = koefisien



Gambar 2.6. Contoh banyaknya alternatif *hyperplane* untuk suatu masalah klasifikasi [30]

Model SVM bertujuan untuk membangun suatu fungsi yang menghasilkan *hyperplane* yang optimal, yaitu *hyperplane* dengan margin terbesar seperti pada Gambar 2.7 yang dirumuskan pada (2.15) dan (2.16).



Gambar 2.7. Pendefinisian *hyperplane* yang unik berdasarkan konsep optimal *hyperplane* [30]

$$(w.x_i) - b \geq 1, \qquad jika\ y_i = 1, i = 1,2, \dots, l \qquad (2.15)$$

$$(w.x_i) - b \leq -1, \qquad jika\ y_i = -1, i = 1,2, \dots, l \qquad (2.16)$$

Dimana :

w = bobot
x_i = himpunan data latih
y_i = label kelas dari x_i
l = banyaknya data
b = koefisien

Dengan demikian, untuk memperoleh *hyperplane* optimal harus diselesaikan dengan suatu masalah pemrograman kuadratik, sehingga didapatkan aturan separasi berdasarkan *hyperplane* optimal yang diperoleh selanjutnya dapat dihasilkan dari fungsi (2.27).

$$f(x) = sign(\sum_{support\ vectors} y_i a_i^0 (x_i \cdot x) - b_0) \qquad (2.17)$$

Dimana :

x_i = *support vectors*
a_i^0 = koefiesien Lagrange
b_0 = konstanta (*threshold*) yang dapat diperoleh dari (2.18)

$$b_0 = \frac{1}{2}[(w_0 \cdot x^*(1)) + (w_0 \cdot x^*(-1))] \qquad (2.18)$$

Dimana :

$x^*(1)$ = *support vector* yang masuk dalam kelas pertama
$x^*(-1)$ = *support vector* yang masuk dalam kelas kedua

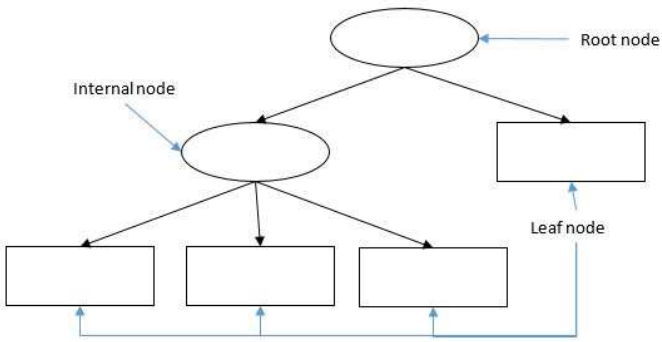
Pada SVM *non linear*, dapat menggunakan fungsi (2.17), kecuali bagian *inner products*, dimana *inner products* $(x_i \cdot x_j)$ digantikan dengan $K(x_i \cdot x_j)$. Terdapat beberapa pilihan fungsi kernel $k(\cdot,\cdot)$, antara lain *polynomial* dan *Radial Basis Function* (RBF) seperti yang dirumuskan masing-masing pada (2.19) dan (2.20) [30].

$$polynomial = (\tau + x_i \cdot x)^d \tag{2.19}$$

$$RBF = \exp(-\|x - x_k\|_2^2 / \sigma^2) \tag{2.20}$$

2.9.3 Decision Trees

Decision Trees (DTs) adalah sebuah struktur yang dapat digunakan untuk membagi kumpulan data yang besar menjadi himpunan-himpunan *record* yang lebih kecil dengan menerapkan serangkaian aturan keputusan. Pada DTs, setiap *leaf node* menandai label kelas. *Node* yang bukan *node* akhir terdiri atas *root* dan *internal node* yang terdiri atas kondisi tes atribut pada sebagian *record* yang mempunyai karakteristik yang berbeda [31]. Berikut adalah struktur DTs seperti yang ditunjukkan pada Gambar 2.8.



Gambar 2.8. Struktur *decision trees* [31]

Terdapat beberapa tipe *tree* yang dimiliki oleh DTs, antara lain *Classification and Regression Tree* (CART), *Iterative Dichotomiser* 3 (ID3), dan C4.5. Setiap tipe, memiliki pengukuran *split* masing-masing. Seperti halnya CART menggunakan *gini index*, sementara ID3 dan C4.5 menggunakan *entropy*. Pada CART, pemilihan atribut sebagai *root node* didasarkan pada nilai *gini split* minimum dari atribut-atribut yang ada dengan menggunakan (2.21). Namun sebelum mendapatkan nilai *gini split,* dilakukan pencarian nilai *gini index* dengan menggunakan (2.22) [32].

$$Gini_{split} = \sum_{i=1}^{k} \frac{n_i}{n} x\ Gini(i) \tag{2.21}$$

$$Gini = 1 - \sum_{i=1}^{k} (p_i)^2 \tag{2.22}$$

Dimana:

k = jumlah partisi atribut
n_i = jumlah kasus pada partisi ke-i
n = jumlah kasus pada *node*
p_i = proporsi pada sampel yang termasuk dalam suatu kelas

Sementara itu, ID3 dan C4.5 memilih atribut sebagai *root node* didasarkan pada nilai *gain* tertinggi dari atribut-atribut yang ada dengan menggunakan (2.23).

$$Gain\ (S, A) = Entropy(S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} x\ Entropy(S_i) \tag{2.23}$$

Dimana:

S = himpunan kasus
A = atribut

n = jumlah partisi atribut A
$|S_i|$ = jumlah kasus pada partisi ke-i
$|S|$ = jumlah kasus dalam S

Sebelum mendapatkan nilai *gain,* dilakukan pencarian nilai *entropy* dengan menggunakan (2.24). *Entropy* digunakan untuk menentukan seberapa informatif sebuah masukan atribut untuk menghasilkan sebuah atribut [33].

$$Entropy(S, A) = -\sum_{i=1}^{n} p_i x \log_2 p_i \qquad (2.24)$$

Dimana:

n = jumlah partisi S
p_i = proporsi dari S_i terhadap S

2.10 Akurasi, Presisi, Recall, dan F1-Score

Evaluasi terhadap kinerja suatu algoritma klasifikasi dilakukan untuk mengetahui seberapa baik algoritma dalam mengklasifikasikan data. *Confusion matrix* merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu algoritma klasifikasi. Pada dasarnya *confusion matrix* mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh algoritma dengan hasil klasifikasi yang seharusnya [34].

Pada pengukuran kinerja menggunakan *confusion matrix*, terdapat empat istilah sebagai representasi hasil proses klasifikasi yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). Nilai TN merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan FP merupakan data negatif namun terdeteksi sebagai data positif. Sementara TP merupakan data positif yang terdeteksi benar dan FN merupakan kebalikan dari TP, yaitu data positif namun terdeteksi sebagai data

negatif. Pada jenis klasifikasi biner yang hanya memiliki dua kelas, *confusion matrix* disajikan seperti pada Tabel 2.5.

Tabel 2.5 *Confusion Matrix*

		Kelas Prediksi	
		Benar	Salah
Kelas Sebenarnya	Benar	*True Positive* (TP)	*False Negative* (FN)
	Salah	*False Positive* (FP)	*True Negative* (TN)

Berdasarkan nilai TN, FP, FN, dan TP dapat diperoleh nilai akurasi, presisi, *recall*, dan *f1-score*. Nilai akurasi pada (2.25) menggambarkan seberapa akurat algoritma dapat mengklasifikasikan data secara benar, dengan kata lain, nilai akurasi merupakan perbandingan antara data yang terklasifikasi benar dengan keseluruhan data. Nilai presisi pada (2.26) menggambarkan jumlah data kategori positif yang diklasifikasikan secara benar dibagi dengan total data yang diklasifikasi positif. Nilai *recall* pada (2.27) menunjukkan berapa persen data kategori positif yang terklasifikasikan dengan benar. Sementara, nilai *f1-score* pada (2.28) menggambarkan rata-rata harmonis antara nilai presisi dan nilai *recall.*

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.25}$$

$$Presisi = \frac{TP}{TP + FP} \tag{2.26}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.27}$$

$$F1 - score = \frac{2\ x\ Presisi\ x\ Recall}{Presisi + Recall} \tag{2.28}$$

Dimana:

1. TP adalah *True Positive*, yaitu jumlah data positif yang terklasifikasi dengan benar.
2. TN adalah *True Negative*, yaitu jumlah data negatif yang terklasifikasi dengan benar.
3. FN adalah *False Negative*, yaitu jumlah data negatif namun terklasifikasi salah.
4. FP adalah *False Positive*, yaitu jumlah data positif namun terklasifikasi salah.

Sementara itu, pada klasifikasi dengan jumlah kelas yang lebih dari dua atau multikelas, cara menghitung akurasi, presisi dan *recall* dapat dilakukan dengan menghitung rata-rata dari nilai akurasi, presisi dan recall pada setiap kelas [35] seperti yang dapat dilihat pada (2.29), (2.30), dan (2.31), sedangkan *f1-score* perhitungannya sama dengan klasifikasi dengan jumlah kelas sebanyak dua.

$$Akurasi = \frac{\sum_{i=1}^{l} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}}{l} \tag{2.29}$$

$$Presisi = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} (TP_i + FP_i)} \tag{2.30}$$

$$Recall = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} (TP_i + FN_i)} \tag{2.31}$$

Dimana:

1. TP_i adalah *True Positive*, yaitu jumlah data positif yang terklasifikasi dengan benar untuk kelas ke-i.
2. TN_i adalah *True Negative*, yaitu jumlah data negatif yang terklasifikasi dengan benar untuk kelas ke-i.
3. FN_i adalah *False Negative*, yaitu jumlah data negatif namun terklasifikasi salah untuk kelas ke-i.
4. FP_i adalah *False Positive*, yaitu jumlah data positif namun terklasifikasi salah untuk kelas ke-i

5. l adalah jumlah kelas.

Karena klasifikasi gaya belajar VAK terdapat lima kelas, maka digunakan perhitungan akurasi, presisi, *recall* dan *f-measure* untuk multikelas seperti pada (2.29), (2.30), (2.31), dan (2.28) yang dimanfaatkan sebagai evaluasi pada saat klasifikasi.

2.11 Python

Python adalah bahasa pemograman yang populer. Python sering dimanfaatkan dalam pengembangan web, perangkat lunak, penelitian, dan *system scripting*. Python dapat digunakan untuk menangani data besar dan melakukan operasi matematika yang kompleks. Python bekerja di berbagai *platform* seperti Windows, DOS, Mac, Amiga, dan lain-lain. Python dirancang untuk mudah dibaca, yaitu memiliki sintaks yang sederhana dan menggunakan bahasa inggris [36].

2.12 Library

Library merupakan merupakan sekumpulan kode yang memiliki fungsi-fungsi tertentu dan dapat dipanggil kedalam program lain. Dengan adanya *library,* dapat mempermudah dalam membangun sebuah program tanpa harus membangun kode dari awal untuk suatu fungsi tertentu. Adapun beberapa *library* yang digunakan dalam melakukan implementasi Tugas Akhir ini. *Library* yang digunakan antara lain, Pandas, NumPy, Regular Expression, Scikit-learn, SciPy, Imbalanced-learn, dan Matplotlib.

2.12.1 Pandas

Pandas adalah *library* bersifat *open source* yang sudah berlisensi *Berkeley Source Distribution* (BSD). *Library* ini memiliki performa tinggi, struktur data yang mudah digunakan, dan alat analisis data untuk bahasa pemrograman Python [37].

Pandas tepat digunakan saat bekerja dengan data tabular, seperti data yang disimpan dalam spreadsheet. Pandas akan membantu dalam mengeksplorasi, membersihkan, dan memproses

data tabular tersebut. Dalam Pandas, data tabular disebut dengan DataFrame [38].

2.12.2 NumPy

NumPy adalah *library Python* yang mendukung pengolahan data pada *array* dan matriks multidimensi yang besar. NumPy menyediakan kumpulan fungsi matematika, seperti aljabar linear, transformasi Fourier, pembuatan angka acak, dan lain-lain. NumPy juga berlisensi BSD dan bersifat *open source,* sehingga banyak dimanfaatkan dalam pengolahan data peneilitian [39].

2.12.3 Regular Expression

Regular expression atau yang bisa disebut juga dengan RE, atau regex, atau pola regex, pada dasarnya adalah bahasa pemrograman yang terspesialisasi tertanam di dalam bahasa pemrograman Python dan tersedia melalui *library* re. *Library* ini dapat dimanfaatkan untuk mengecek kesamaan pada *string*, memodifikasi *string*, atau membaginya dengan berbagai cara [40].

2.12.4 Scikit-learn

Scikit-learn merupakan *open source machine learning library* untuk bahasa pemograman Python yang berlisensi BSD. Scikit-learn menyediakan fitur seperti *classification*, *regression*, *clustering*, termasuk juga didalamnya algoritma SVM, *nearest neighbors*, *decision trees*, dan lain-lain [41].

2.12.5 SciPy

SciPy adalah *library* pada bahasa pemrograman Python yang bersifat *open source* yang digunakan untuk kebutuhan matematika, sains, dan teknik. *Library* ini tergantung pada NumPy, yang menyediakan manipulasi array n-dimensi yang mudah dan cepat seperti rutinitas untuk integrasi dan optimalisasi numerik [42].

2.12.6 Warnings Filter

Pesan *warning* (peringatan) biasanya dikeluarkan dalam situasi untuk mengingatkan pengguna tentang beberapa kondisi dalam suatu program. Namun adanya *library* filter peringatan pada bahasa pemrograman Python, dapat mengontrol apakah peringatan tersebut diabaikan, ditampilkan, atau diubah menjadi *error* (menimbulkan *exception*) [43].

2.12.7 Imbalanced-learn

Imbalanced-learn adalah *library* bersifat *open* source pada bahasa pemrograman Python yang menyediakan berbagai macam metode untuk mengatasi masalah dataset yang menunjukkan ketidakseimbangan antar kelas yang sering ditemui dalam *machine learning* dan pengenalan pola. Teknik *re-sampling* pada *library* ini dibagi dalam empat kategori antara lain, *under-sampling, over-sampling,* kombinasi antara *over-* dan *under-sampling,* serta metode *ensemble learning* [44].

2.12.8 Statistics

Statistics merupakan *library* pada bahasa pemrograman Python yang menyediakan fungsi untuk menghitung statistik matematika dari data numerik. *Library* ini tidak dimaksudkan untuk menjadi pesaing bagi *library* lainnya, seperti NumPy, SciPy, atau paket statistik fitur lengkap yang ditujukan untuk ahli statistik profesional, seperti Minitab, SAS, dan Matlab. Karena pada dasarnya, statistics ditujukan sebagai kalkulator *scientific* [45].

2.12.9 Matplotlib

Matplotlib adalah *library* Python yang mendukung pembuatan grafik dua dimensi dalam berbagai format dan dari berbagai jenis data. Matplotlib bersifat *open source* dan banyak digunakan untuk pengolahan data dalam penelitian. Matplotlib dapat membuat plot, histogram, spektrum daya, diagram batang, diagram kesalahan, plot pencar, dan lain-lain [46].

2.12.10 PySimpleGUI

PySimpleGUI merupakan *library* untuk membuat *graphical user interface* (GUI) dalam bahasa pemrograman Python yang mudah digunakan. Meskipun mudah digunakan, *library* ini memiliki fitur yang dapat dieksplorasi lebih dalam lagi. Beberapa fitur yang tersedia seperti, membuat *button* (*file browse, saveAs*), *slider*, grafik, tabel, dan lain-lain [47].

2.12.11 Subprocess

Library subprocess memungkinkan untuk membuat proses baru yang terhubung ke *input*/*output*/*error* kemudian mendapatkan *return code*. *Library* ini bermaksud untuk menggantikan modul dan fungsi lama, yaitu os.system dan os.spawn. Pendekatan yang disarankan untuk menjalankan library ini adalah menggunakan fungsi run untuk semua kasus penggunaan yang dapat ditangani. Untuk kasus penggunaan yang lebih lanjut, antarmuka Popen yang mendasarinya dapat digunakan secara langsung [48].

BAB III
PERANCANGAN SISTEM

Bab ini menjelaskan tentang perancangan data dan sistem klasifikasi multi label gaya belajar VAK menggunakan *supervised learning*. Bab ini juga akan menjelaskan gambaran umum sistem dalam bentuk diagram alir.

3.1 Perancangan Data

Dalam Tugas Akhir ini, dataset yang akan digunakan sebagai data latih dan data uji berupa *data log* dari LMS Universitas Nairobi yang diambil selama periode 15 minggu perkuliahan. Subjek pada *data log* merupakan mahasiswa kedokteran sebanyak 1.166 orang pada semester pertama. Jumlah keseluruhan *data log* sebanyak 41.638 *records*, yang terdiri atas empat atribut sesuai dengan standar *data log* pada LMS Universitas Nairobi. Sampel *data log* dapat dilihat pada Tabel 2.2, sedangkan keterangan atribut *data log* dapat dilihat pada Tabel 3.1.

Tabel 3.1 Penjelasan Atribut pada *Data Log* dari Universitas Nairobi

Nama Atribut	Keterangan
user_id	Nomor identitas mahasiswa
module_id	Kode pada konten
module_name	Nama konten yang dikunjungi
total_time	Lama waktu yang dihabiskan pada konten

Pada atribut module_name, digunakan konten yang penamaannya bersesuaian dengan pemetaan perilaku pembelajaran yang sudah dijelaskan pada subbab 2.3. Berdasarkan hal tersebut, didapatkan jumlah module_name seperti pada Tabel 3.2 yang selanjutnya digunakan pada Tugas Akhir ini.

Tabel 3.2 Jumlah module_name yang Digunakan

Jenis module_name	Jumlah	Gaya Belajar
Konten grafis (*figure, illustration, chart*, video)	10	Visual
Konten audio (*recording*)	5	Auditori
example	22	Auditori
outline	4	Auditori
review question, activity, case study, forum	20	Kinestetik
exercise	8	Kinestetik
assesment	23	Kinestetik
self-test	12	Kinestetik

3.2 Pemberian Label

Pemberian label (*ground truth*) pada dataset dilakukan dengan membagikan kuesioner yang berisi 30 sampel *data log* pelajar kepada responden untuk menentukan gaya belajarnya. Sebelum mengisi kuesioner, responden diberikan penjelasan mengenai gaya belajar VAK untuk dapat menjawab setiap pertanyaan yang diajukan pada kuesioner. Terdapat 30 sampel *data log* yang dipilih merupakan cakupan dari keseluruhan *data log* yang ada. Dalam Tugas Akhir ini, diambil 8 responden yang terlibat dalam pengisian kuesioner, diantaranya 7 orang dari kalangan mahasiswa dan 1 orang dari pegawai swasta. Selanjutnya, hasil jawaban kuesioner yang telah terkumpul akan dilakukan pengambilan gaya belajar yang sering muncul pada jawaban responden (*majority vote*) untuk menentukan keputusan final pada gaya belajar. Selanjutnya, *ground truth* untuk *data log* yang lainnya akan ditentukan secara manual dengan menyesuaikan hasil dari sampel *data log* yang serupa pada kuesioner. Hasil pelabelan pada dataset dapat dilihat pada Tabel 3.3.

Tabel 3.3 Hasil Pelabelan pada Dataset sebagai *Ground Truth*

Label	Jumlah
V	68
A	75
K	285
VA	**1**
VK	33
AK	19
VAK	**2**
Total	**483**

Berdasarkan Tabel 3.3, total mahasiswa yang semula berjumlah 1.166 orang kemudian berkurang menjadi 483 orang dikarenakan hanya 483 mahasiswa yang *data log* nya dapat dilakukan analisis gaya belajar lebih lanjut. Hal ini dikarenakan, terdapat beberapa nama modul yang tidak dapat terbaca seperti pada Tabel 3.4. Kemudian dari nama modul ada juga yang penamaannya hanya secara umum dan tidak menggambarkan pada fitur gaya belajar yang sudah dipaparkan pada Tabel 2.4. Sampel *data log* tersebut dapat dilihat pada Tabel 3.5.

Tabel 3.4 Sampel *Data Log* pada module_name yang Tidak Dapat Terbaca

module_name
3.5 ╨òm╨╛ ti╨╛ ns
2.4 ╨Ptsp╨╡ r╨▒nt ╨í╨╛ nditi╨╛ ning
S╨ò╨íTI╨PtsN 1: Intr╨╛ du╤üti╨╛ n t╨╛ Psy╤üh╨╛ l╨╛ gy

Tabel 3.5 Sampel *Data Log* pada module_name yang Penamaannya Secara Umum

module_name
UNIT 1: FUNDAMENTALS OF SOCIOLOGY
LECTURE 6: ENDOCRINE GLANDS OF THE HEAD AND NECK
2.4.6.2: Genetic and Environment Factors influencing Intelligence

Selain itu, jumlah label K pada Tabel 3.3 menunjukkan perbedaan yang sangat jauh jika dibandingkan dengan label yang lainnya. Hal ini berhubungan dengan pengaruh jumlah module_name untuk gaya belajar kinestetik pada Tabel 3.2 dimana jumlah yang tersedia lebih banyak jika dibandingkan dengan jumlah module_name pada gaya belajar visual dan auditori. Perbedaan jumlah inilah yang menyebabkan terjadinya ketidakseimbangan label pada dataset.

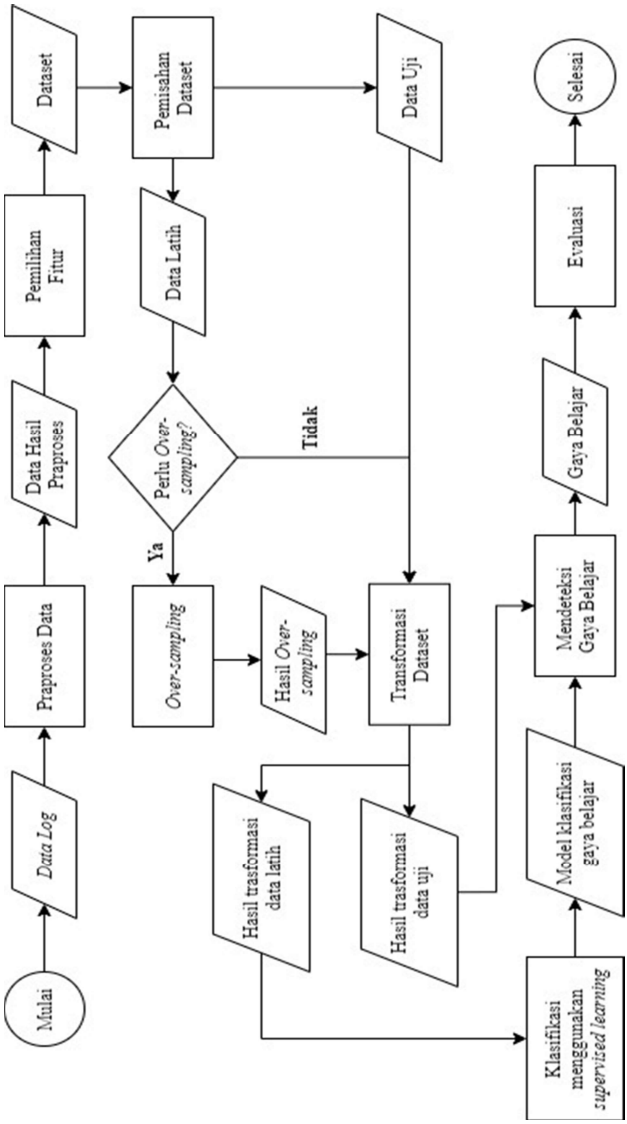
Kemudian, label VA dan VAK pada Tabel 3.3 masing-masing hanya berjumlah 1 dan 2, dimana tidak mencukupi untuk dilakukan pelatihan maupun pengujian. Sehingga, dalam hal ini dilakukan penghapusan *record* pada *data log* yang memiliki gaya belajar VA dan VAK. Total *ground truth* yang semula 483 kemudian berkurang menjadi 480 yang selanjutnya digunakan pada Tugas Akhir ini.

3.3 Desain Umum Sistem

Sistem klasifikasi multi label gaya belajar VAK yang dibangun memiliki enam proses utama yaitu praproses *data log*, pemilihan fitur, pemisahan dataset, *over-sampling*, transformasi data, serta klasifikasi *k-Nearest Neighbors* (k-NN), *Support Vector Machine* (SVM), dan *Decision Trees* (DTs). Diagram alir dari sistem ditunjukan pada Gambar 3.1.

Pada tahap pertama, akan dilakukan praproses pada *data log* dengan melakukan pemilihan data, pengubahan menjadi huruf

kecil (*case folding*), *regular expression* (regex), dan penghapusan *outlier*. Kemudian hasil praproses *data log* akan digunakan pada tahap berikutnya, yaitu pemilihan fitur. Fitur yang digunakan sebanyak 11 fitur seperti yang digambarkan pada pada Tabel 2.4. Dari hasil pemilihan fitur tersebut, dihasilkan sebuah dataset yang akan dilakukan pemisahan dataset. Pemisahan dataset dilakukan dengan menggunakan dua metode yang berbeda, yaitu metode *hold-out* dan *k-fold* untuk membagi data latih dan data uji. Setelah itu, untuk mendapatkan kelas yang seimbang, maka data latih akan dilakukan proses *over-sampling* dengan menggunakan BOS. Selanjutnya, dilakukan transformasi data pada data latih hasil *over-sampling* dan data uji dengan menggunakan salah satu teknik normalisasi, yaitu min-max. Kemudian, hasil transformasi pada data latih dilakukan tahap klasifikasi dengan menggunakan k-NN, SVM, dan DTs. Model dari masing-masing hasil klasifikasi, disimpan dan selanjutnya dilakukan pengujian untuk memprediksi gaya belajar pada data uji. Selanjutnya hasil prediksi gaya belajar tersebut akan dibandingkan dengan *ground truth* dan model dievaluasi dengan menghitung nilai akurasi, *precision*, *recall*, dan *f1-score*.



Gambar 3.1 Diagram alir sistem yang dibangun

3.3.1 Tahap Praproses Data Log

Tahap praproses pada Tugas Akhir ini dimulai dengan melakukan pemilihan data. Pemilihan yang dimaksud adalah menghapus salah satu atribut pada Tabel 3.1 yang tidak digunakan dan mengolah atribut lainnya. Dalam hal ini, atribut module_id akan dihilangkan dalam dataframe. Kemudian dilakukan pengurutan pada atribut user_id dari yang terkecil. Terakhir, mengonversi atribut total_time yang memiliki format hh:mm:ss menjadi total waktu dalam detik yang disimpan pada atribut total_second. Contoh hasil dari pemilihan data dapat dilihat pada Gambar 3.2.

	user_id	module_name	total_second
36923	1	7.5: Clinical features of fractures of the Max...	18.0
20097	1	SCH 306: QUANTITATIVE AND QUALITATIVE ANALYSIS	21.0
20098	1	Autoprotolysis	25.0
20099	1	Conjugate Aids and Bases	8.0
36924	1	5.3.3: Closed Reduction	40.0

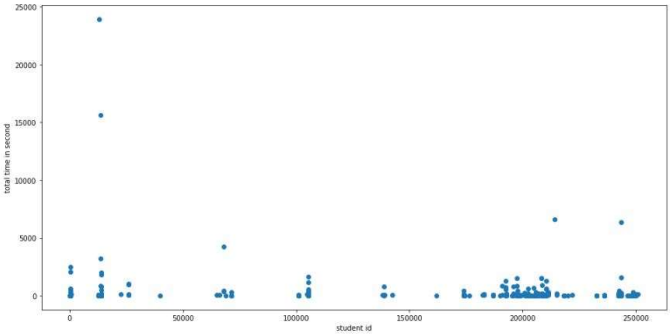
Gambar 3.2 Hasil pemilihan *data log*

Selanjutnya, hasil dari pemilahan *data log* akan dilakukan *case folding* pada atribut module_name agar setiap hurufnya menjadi kecil. Selain itu, diperlukan *library* regex untuk menghilangkan angka serta tanda baca, sehingga didapatkan contoh hasil seperti pada Gambar 3.3.

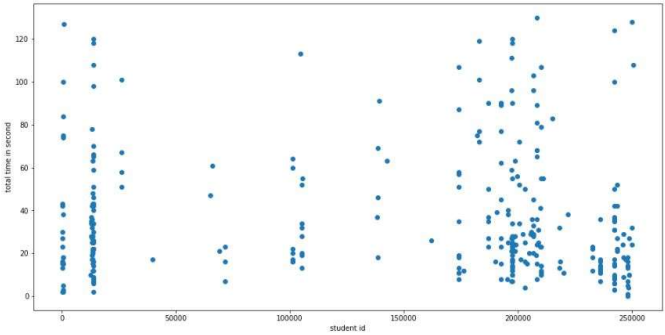
	user_id	module_name	total_second
36923	1	clinical features of fractures of the maxilla	18.0
20097	1	sch quantitative and qualitative analysis	21.0
20098	1	autoprotolysis	25.0
20099	1	conjugate aids and bases	8.0
36924	1	closed reduction	40.0

Gambar 3.3 Hasil *case folding* dan regex

Setelah hasil *case folding* dan regex didapatkan, tahap praproses yang terakhir yaitu, penghapusan *outlier*. *Outlier* yang dimaksud adalah atribut total_second pada *exercise*, *assesment*, dan *self-test*. Deteksi *outlier* menggunakan MADe seperti yang sudah dijelaskan pada subbab 2.5. Sebelum melakukan penghapusan *outlier*, akan dilihat terlebih dahulu secara visualisasi persebaran total_second untuk dibandingkan dengan total_second tanpa *outlier*. Berdasarkan Gambar 3.4 dapat disimpulkan bahwa penghapusan *outlier* membuat persebaran menjadi lebih merata.



(a)



(b)

Gambar 3.4 (a) Atribut total_second sebelum penghapusan *outlier*; (b) Atribut total_second sesudah penghapusan *outlier*

3.3.2 Tahap Pemilihan Fitur

Setelah tahap praproses, tahap selanjutnya adalah pemilihan fitur dengan mencari rasio kunjungan untuk tiap konten dan rasio waktu yang dihabiskan pada konten seperti yang sudah dijelaskan pada subbab 2.3. Dapat dilihat pada Tabel 2.4, nantinya dibutuhkan 11 fitur sesuai dengan perilaku pembelajaran *online* terhadap gaya belajar VAK, diantaranya 9 fitur kunjungan dan 3 fitur lamanya waktu yang dihabiskan.

Untuk mendapatkan 9 fitur kunjungan, dilakukan perhitungan rasio kunjungan untuk tiap konten. Sedangkan untuk mendapatkan 3 fitur lainnya, dilakukan perhitungan rasio waktu yang dihabiskan pada konten. Saat mencari rasio waktu, perkiraan waktu yang diharapkan untuk dihabiskan pada setiap konten ($\mathrm{TES}_{\mathrm{Content}}$) menggunakan rata-rata waktu pada tiap konten. Hasil pemilihan fitur dapat dilihat pada Gambar 3.5 dimana merupakan kondisi ideal dataset yang digunakan untuk menentukan gaya belajar VAK.

tes	exercise	assesment	aktifitas	konten_ilustrasi	konten_audio	outline	example
0.000000	37.5	0.000000	0.0	0.0	0.0	0.0	0.000000
8.333333	0.0	8.695652	10.0	0.0	0.0	0.0	0.000000
0.000000	0.0	0.000000	5.0	0.0	0.0	0.0	0.000000
0.000000	0.0	0.000000	0.0	0.0	0.0	0.0	4.545455

(a)

waktu_tes	waktu_exercise	waktu_asses
0.000000	26.139218	0.000000
133.013436	0.000000	21.245902
0.000000	0.000000	0.000000
0.000000	0.000000	0.000000

(b)

Gambar 3.5 Hasil pemilihan fitur. (a) Rasio kunjungan; (b) Rasio waktu yang dihabiskan.

3.3.3 Tahap Pemisahan Dataset

Dataset akan dibagi menjadi data latih dan data uji. Pemisahan dataset dimaksudkan agar model yang diperoleh memiliki generalisasi yang baik dalam melakukan klasifikasi data. Dalam melakukan pemisahan dataset digunakan dua teknik *cross-validation* yang berbeda, yaitu *hold-out* dan *k-fold* seperti yang sudah dijelaskan pada subbab 2.7. Hal ini dilakukan untuk diuji coba dan dibandingkan dengan hasil evaluasi.

Pada *hold-out*, dataset dipisah menjadi data latih dan uji dengan perbandingan masing-masing sebesar 90% dan 10% dengan 5 kali pengujian. Proses pemisahan teknik ini akan dibantu dengan fungsi train_test_split dari *library* scikit-learn dengan mengatur parameter stratify agar mendapat proporsi jumlah label yang sama. Sedangkan untuk *k-fold*, dataset dipisah menjadi 10 bagian (*fold*), meliputi 9 *fold* sebagai data latih dan 1 *fold* sebagai data uji. Proses pemisahan teknik ini akan dibantu dengan fungsi StratifiedKFold dari *library* scikit-learn.

3.3.4 Tahap Over-sampling

Berdasarkan pada Tabel 3.6, hasil persebaran label pada data latih dapat disimpulkan bahwa tidak seimbang karena label dengan jumlah paling sedikit tidak mencapai 1/10 dari jumlah label yang lain. Oleh karena itu, dilakukan *over-sampling* untuk diuji coba dan dibandingkan dengan hasil evaluasi. Salah satu teknik *over-sampling* yang akan digunakan, yaitu *Borderline Over-sampling* (BOS) seperti yang sudah dijelaskan pada subbab 2.6. Proses *over-sampling* ini akan dibantu dengan fungsi SVMSMOTE dari *library* scikit-learn.

Tabel 3.6 Persebaran Data Latih pada *Fold* Pertama

Label	Jumlah
V	61
A	67
K	256
VK	30
AK	18

3.3.5 Tahap Transformasi Data

Setelah melalui tahap *over-sampling*, data latih hasil *over-sampling* dan data uji perlu dilakukan transformasi terlebih dahulu karena nilai antar fitur memiliki rentang yang berbeda seperti yang dilihat pada Gambar 3.6. Dalam hal ini, metode yang digunakan adalah normalisasi min-max seperti yang sudah dijelaskan pada subbab 2.8. Nantinya, fitur-fitur akan ditransformasikan sehingga berada pada rentang 0 sampai dengan 1.

```
tes                    0
exercise               0
assesment              0
aktifitas              0
konten_ilustrasi       0
konten_audio           0
outline                0
example                0
waktu_tes              0
waktu_exercise         0
waktu_asses            0
```

```
tes                 41.6667
exercise               62.5
assesment           43.4783
aktifitas              140
konten_ilustrasi        50
konten_audio            20
outline                275
example                100
waktu_tes          392.706
waktu_exercise     321.259
waktu_asses        294.787
```

Gambar 3.6 Nilai minimum (kiri) dan maksimum (kanan) pada tiap fitur

3.3.6 Tahap Klasifikasi dan Evaluasi Gaya Belajar

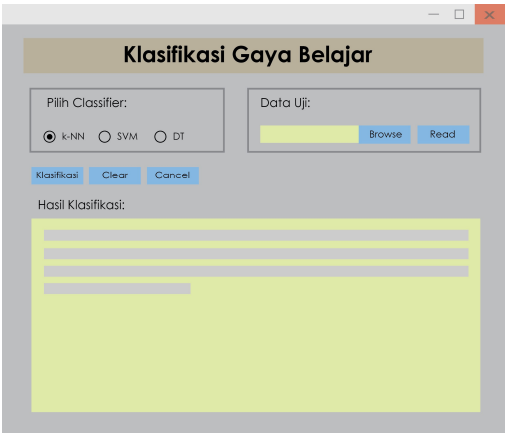
Terdapat beberapa algoritma *supervised learning* yang akan digunakan, yaitu *k-Nearest Neighbors* (k-NN), *Support Vector Machine* (SVM), dan *Decision Trees* (DTs) seperti yang sudah dijelaskan pada subbab 2.9. Masing-masing algoritma tersebut akan diuji coba dan evaluasi dengan mengatur parameternya untuk menemukan model yang memiliki akurasi

terbaik. Pengaturan parameter pada k-NN dilakukan pada n_neighbors, yaitu jumlah k tetangga terdekat mulai dari 1-10. Untuk SVM dilakukan pengaturan parameter pada kernel, yaitu RBF, *linear*, dan *polynomial*. Sedangkan DTs dilakukan pengaturan pada parameter *criterion,* yaitu pengukuran kualitas dari *split*, meliputi *gini* dan *entropy*.

Pada saat pelatihan dan pengujian untuk klasifikasi gaya belajar VAK dilakukan dengan menggunakan dataset dari pemilihan fitur sebanyak 480 *records*. Pemisahan dataset menggunakan dua metode, antara lain *hold-out* dan *k-fold* yang sudah dijelaskan pada subbab 3.3.3. Kemudian hasil dari pengujian akan dievaluasi seperti pada subbab 2.10 dengan menggunakan akurasi, presisi, *recall*, dan *f1-score*.

3.4 Desain Umum User Interface

Pada Tugas Akhir ini *Graphical User Interface* (GUI) dibangun menggunakan *library* bawaan dari bahasa pemrograman Python, yaitu PySimpleGUI sebagai *layouting* dan subprocess untuk menghubungkan antara *input* dan *output* pada program Python yang terpisah. *Mockup* GUI dapat dilihat pada Gambar 3.7.



Gambar 3.7 *Mockup* GUI program klasifikasi gaya belajar VAK

BAB IV
IMPLEMENTASI

Bab ini menjelaskan mengenai implementasi perangkat lunak dari rancangan sistem yang telah dibahas pada Bab 3 yang meliputi kode program. Selain itu, implementasi dari tiap proses, parameter masukan dan keluaran, serta keterangan yang berhubungan dengan program juga akan dijelaskan pada bab ini.

4.1 Lingkungan Implementasi

Dalam mengimplementasikan sistem klasifikasi multi label gaya belajar VAK digunakan beberapa perangkat pendukung sebagai berikut.

4.1.1 Perangkat Keras

Implementasi Tugas Akhir ini menggunakan *notebook* UX430UN. Sistem operasi yang digunakan adalah Windows 10 Home 64-bit. *Notebook* yang digunakan memiliki spesifikasi Intel® Core™ i7-8550U dengan kecepatan 4.0 GHz, *Random Access Memory* (RAM) sebesar 16 GB, dan mempunyai *Video Graphics Adapter* (VGA) yaitu NVIDIA GeForce MX150.

4.1.2 Perangkat Lunak

Perangkat lunak yang digunakan memiliki spesifikasi antara lain menggunakan bahasa pemograman Python 3.6, dilengkapi dengan *library* antara lain Pandas, NumPy, Regular Expression, Scikit-learn, SciPy, Imbalanced-learn, Statisctics, Matplotlib, PySimpleGUI, dan Subprocess.

4.2 Implementasi Praproses Data Log

Pada subbab ini akan dijelaskan mengenai implementasi praproses pada *data log*.

4.2.1 Implementasi Pemilihan Data Log

Langkah pertama dalam pemilihan *data log* adalah mengurutkan *data log* berdasarkan atribut user_id secara

ascending (dari yang terkecil) dan menghilangkan atribut module_id yang diimplementasikan pada Kode Sumber 4.1. Pada baris 1, pengurutan *data log* memanfaatkan fungsi sort_values pada *library* pandas dimana sudah secara *default* mengurutkan dari yang terkecil. Sedangkan pada baris 2, penghapusan memanfaatkan fungsi drop pada *library* pandas dengan mengatur parameter axis=1 dimana berpengaruh pada seluruh kolom pada tiap baris.

```
1. data_log = data_log.sort_values('user_id')
2. data_log = data_log.drop(['module_id'],axis=1)
```

Kode Sumber 4.1 Fungsi pengurutan dan penghapusan atribut module_id

Langkah selanjutnya adalah mengonversi waktu dalam detik. Konversi waktu diimplementasikan pada Kode Sumber 4.2. Atribut total_time pada *data log* memiliki format waktu hh:mm:ss, sehingga pada baris 1-3 dilakukan pengambilan waktu dalam jam, menit, dan detik secara terpisah dengan menggunakan fungsi to_datetime pada *library* pandas yang disimpan pada atribut baru. Selanjutnya, baris 4 menyimpan total waktu dalam detik dengan menjumlahkan nilai waktu dari hasil baris 1-3, dimana satuan dalam jam dikalikan dengan 3600 detik dan satuan menit dikalikan dengan 60 detik.

```
1. data_log['hour'] = pd.to_datetime(data_log['total_t
   ime'],format= '%H:%M:%S', errors='coerce').dt.hour
2. data_log['minute'] = pd.to_datetime(data_log['total
   _time'],format= '%H:%M:%S', errors='coerce').dt.min
   ute
3. data_log['second'] = pd.to_datetime(data_log['total
   _time'],format= '%H:%M:%S', errors='coerce').dt.sec
   ond
4. data_log['total_second'] = (data_log['hour'] * 3600
   ) + (data_log['minute'] * 60) + data_log['second']
```

Kode Sumber 4.2 Fungsi konversi waktu dalam detik

4.2.2 Implementasi Case Folding

Case folding diimplementasikan pada Kode Sumber 4.3 dengan memanfaatkan fungsi lower pada *library* pandas. Dalam Tugas akhir ini, atribut module_name pada *data log* dikonversi menjadi huruf kecil.

```
1. data_log['module_name'] = data_log['module_name'].s
   tr.lower()
```

Kode Sumber 4.3 Fungsi untuk *case folding*

4.2.3 Implementasi Menghilangkan Angka dan Tanda Baca

Menghilangkan angka dan tanda baca diimplementasikan pada Kode Sumber 4.4 dimana menggunakan fungsi replace pada *library* pandas dengan mengatur parameter regex=True.

```
1. data_log['module_name'] = data_log['module_name'].r
   eplace('[\().":-
   ]','', regex=True).str.replace('\d+', '')
```

Kode Sumber 4.4 Fungsi penghapusan angka dan tanda baca

4.2.4 Implementasi Penghapusan Outlier

Langkah pertama pada penghapusan *outlier* adalah pengambilan *data log* kuis, seperti *self test, exercise,* dan *assesment* menjadi *dataframe* baru yang diimplementasikan pada Kode Sumber 4.5 dimana baris 2 melakukan *looping* pada *data log* untuk melakukan pencarian *data log* yang memiliki atribut module_name kuis pada baris 3-5. Baris 3 melakukan *compile* pola regex secara terpisah dengan menggunakan fungsi compile untuk menghemat waktu. Kemudian baris 4 digunakan untuk pencarian *data log* yang bersesuaian dengan pola yang dibuat pada baris 3 dengan menggunakan fungsi findall. Hasil pencarian kemudian disimpan pada *dataframe* baru pada baris 5.

```
1. df_o = pd.DataFrame(columns = ['user_id','module_name','total_time','hour','minute','second','total_second','k_time'])
2. for index, row in data_log.iterrows():
3.   pattern2=re.compile(r'self test|exercise|assessment')
4.   if pattern2.findall(row.module_name):
5.     df_o.loc[index, ['user_id','module_name','total_time','hour','minute','second','total_second','k_time']] = row.user_id,row.module_name,row.total_time,row.hour,row.minute,row.second,row.total_second,0.0
```

Kode Sumber 4.5 Fungsi pengambilan *data log* kuis

Langkah selanjutnya adalah menghitung nilai *Median Absolute Deviation* (MADe) untuk deteksi *outlier* yang diimplementasikan pada Kode Sumber 4.6 dimana pada baris 1-2, median dan *Median Absolute Deviation* (MAD) dihitung masing-masing menggunakan fungsi numpy.median dan stats.median_absolute_deviation. Hasil perhitungan tersebut, digunakan pada baris 3-4 untuk menghitung MADe.

```
1. median_y = np.median(df_o['total_second'])
2. mad = stats.median_absolute_deviation(df_o['total_second'])
3. m1= median_y+(3*mad)
4. m2= median_y-(3*mad)
```

Kode Sumber 4.6 Fungsi perhitungan *Median Absolute Deviation* (MADe)

Selanjutnya akan dilakukan penghapusan *outlier* pada *data log* yang diimplementasikan pada Kode Sumber 4.7 dimana baris 1 menyimpan data *outlier* yang didapatkan dari rentang MADe pada variabel m1 dan m2. Variabel *outlier* digunakan untuk *intersection* pada *data log*, sehingga *outlier* dapat terhapus pada baris 2. Proses *intersection* menggunakan fungsi merge, query, dan drop pada *library* pandas dengan mengatur parameter how='outer'.

```
1. outlier=df_o[((df_o['total_second'] < m2) |(df_o['t
   otal_second'] > m1))]
2. data_log=pd.merge(data_log, outlier, how='outer', i
   ndicator=True).query('_merge == "left_only"').drop(
   '_merge', 1)
```

Kode Sumber 4.7 Fungsi penghapusan *outlier*

4.3 Implementasi Pemilihan Fitur

Langkah pertama dalam pemilihan fitur adalah menghitung rata-rata waktu tiap konten yang diimplementasikan pada 4.8 dimana baris 2-4 melakukan *compile* pola regex untuk digunakan masing-masing pada baris 5, 8, dan 11. Nama konten yang sesuai dengan pola, akan diambil nilai pada atribut total_second lalu menjumlahkan dengan nilai total_second berikutnya serta menyimpan jumlah konten sampai *looping* pada *data log* berakhir. Jumlah dari total_second tiap pola disimpan masing-masing pada variabel summ, summ2, dan summ3, sedangkan jumlah konten disimpan pada masing-masing variabel div, div2, dan div3. Selanjutnya baris 14-16 melakukan pembagian antara jumlah dari total_second dengan jumlah keseluruhan konten untuk mendapatkan rata-rata tiap konten yang masing-masing disimpan pada variabel avg, avg2, dan avg3.

```
1.  for index, row in data_log.iterrows():
2.    pattern1=re.compile(r'self test')
3.    pattern2=re.compile(r'exercise')
4.    pattern8=re.compile(r'assessment')
5.    if pattern1.findall(row.module_name):
6.      summ=summ+ row.total_second
7.      div=div+1
8.    elif pattern2.findall(row.module_name):
9.      summ2=summ2+ row.total_second
10.     div2=div2+1
11.   elif pattern8.findall(row.module_name):
12.     summ3=summ3+ row.total_second
13.     div3=div3+1
14. avg=summ/div
15. avg2=summ2/div2
```

```
16. avg3=summ3/div3
```

Kode Sumber 4.8 Fungsi perhitungan rata-rata waktu tiap konten

Langkah selanjutnya adalah menghitung rasio lamanya waktu yang dihabiskan pada tiap konten untuk menjadi fitur lamanya kunjungan yang diimplementasikan pada Kode Sumber 4.9 dimana baris dimana baris 2-4 melakukan *compile* pola regex untuk digunakan masing-masing pada baris 5, 7, dan 9. Nama konten yang sesuai dengan pola, akan diambil rasio lamanya waktu dengan membagi nilai pada total_second tiap konten dengan rata-rata waktu tiap konten yang sudah dihitung sebelumnya kemudian dikalikan dengan 100 sehingga menghasilkan rasio dalam bentuk persentase. Hasil dari perhitungan rasio akan disimpan pada atribut baru pada *data log* yang bernama k_time sampai *looping* pada *data log* berakhir.

```
1.  for index, row in data_log.iterrows():
2.    pattern1=re.compile(r'self test')
3.    pattern2=re.compile(r'exercise')
4.    pattern8=re.compile(r'assessment')
5.    if pattern1.findall(row.module_name):
6.      data_log.loc[[index],['k_time']] = row.total_se
    cond/avg*100
7.    elif pattern2.findall(row.module_name):
8.      data_log.loc[[index],['k_time']] = row.total_se
    cond/avg2*100
9.    elif pattern8.findall(row.module_name):
10.     data_log.loc[[index],['k_time']] = row.total_se
    cond/avg3*100
```

Kode Sumber 4.9 Fungsi perhitungan rasio lamanya waktu yang dihabiskan

Selanjutnya adalah menghitung jumlah konten dalam kursus yang diimplementasikan pada Kode Sumber 4.10 dimana baris 2-9 melakukan *compile* pola regex untuk digunakan masing-masing pada baris 10, 12, 14, 16, 18, 20, 22, dan 24. Nama konten yang sesuai dengan pola, akan diambil nama kontennya pada

module_name sampai *looping* pada *data log* berakhir untuk dihitung masing-masing jumlah kontennya pada baris 26-31 dengan menggunakan fungsi nunique pada *library* pandas. Jumlah konten masing-masing disimpan pada variabel a-h.

```
1.  for index, row in data_log.iterrows():
2.    pattern1=re.compile(r'self test')
3.    pattern2=re.compile(r'exercise')
4.    pattern3=re.compile(r'review question|activity|case study|forum')
5.    pattern4=re.compile(r'illustration|figure|chart|video')
6.    pattern5=re.compile(r'listen|audio|record')
7.    pattern6=re.compile(r'outline')
8.    pattern7=re.compile(r'example')
9.    pattern8=re.compile(r'assessment')
10.   if pattern1.findall(row.module_name):
11.     df7.loc[index] = [row.module_name]
12.   elif pattern2.findall(row.module_name):
13.     df4.loc[index] = [row.module_name]
14.   elif pattern3.findall(row.module_name):
15.     df3.loc[index] = [row.module_name]
16.   elif pattern4.findall(row.module_name):
17.     df1.loc[index] = [row.module_name]
18.   elif pattern5.findall(row.module_name):
19.     df2.loc[index] = [row.module_name]
20.   elif pattern6.findall(row.module_name):
21.     df5.loc[index] = [row.module_name]
22.   elif pattern7.findall(row.module_name):
23.     df6.loc[index] = [row.module_name]
24.   elif pattern8.findall(row.module_name):
25.     df8.loc[index] = [row.module_name]
26. a=df1['visual'].nunique()
27. b=df2['auditori'].nunique()
28. c=df3['kinestetik'].nunique()
29. d=df4['kinestetik'].nunique()
30. e=df5['auditori'].nunique()
31. f=df6['auditori'].nunique();g=df7['kinestetik'].nunique();h=df8['kinestetik'].nunique()
```

Kode Sumber 4.10 Fungsi perhitungan jumlah konten dalam kursus

Setelah menghitung jumlah konten dalam kursus, langkah terakhir adalah menghitung rasio kunjungan pada tiap konten untuk menjadi fitur kunjungan yang diimplementasikan pada Kode Sumber 4.11 dimana baris 2-9 melakukan *compile* pola regex untuk digunakan masing-masing pada baris 12, 16, 20, 23, 26, 29, 32, 35, 42, 47, 52, 56, 60, 64, 68, dan 71. Secara keseluruhan, baris 10-38 merupakan user_id yang sudah terbaca oleh sistem, sedangkan 39-74 merupakan user_id yang belum terbaca oleh sistem. Sehingga ketika user_id baru saja dikenal, maka akan dibuatkan *record* baru pada *dataframe* yang bernama dataset. Apabila sebaliknya, maka tidak perlu membuat *record* baru sehingga hanya perlu menambahkan nilai rasio yang baru.

Nama konten yang sesuai dengan pola, akan diambil rasio kunjungan tiap konten dengan membagi banyaknya konten yang dikunjungi pada tiap pelajar dengan jumlah konten dalam kursus yang sudah dihitung sebelumnya kemudian dikalikan dengan 100 sehingga menghasilkan rasio dalam bentuk persentase. Selain itu, rasio lamanya waktu yang disimpan pada atribut k_time akan dijumlahkan dengan konten kuis lainnya yang sama, kemudian dibagi dengan banyaknya konten kuis tersebut. Proses perhitungan rasio ini berlangsung sampai *looping* pada *data log* berakhir.

```
1.  for index, row in data_log.iterrows():
2.    pattern1=re.compile(r'self test')
3.    pattern2=re.compile(r'exercise')
4.    pattern3=re.compile(r'review question|activity|ca
    se study|forum')
5.    pattern4=re.compile(r'illustration|figure|chart|v
    ideo')
6.    pattern5=re.compile(r'listen|audio|record')
7.    pattern6=re.compile(r'outline')
8.    pattern7=re.compile(r'example')
9.    pattern8=re.compile(r'assessment')
10.   if init == str(row.user_id) :
11.     tt1=0;tt2=0;tt3=0;tt4=0;tt5=0;tt6=0;tt7=0;tt8=0
    ;
12.     if pattern1.findall(row.module_name):
13.       temp1 +=1
```

```
14.         tt1= row.k_time + time1
15.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['tes','waktu_tes']]=temp1/g* 100, tt1/temp1
16.       elif pattern2.findall(row.module_name):
17.         temp2 +=1
18.         tt2= row.k_time + time2
19.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['exercise','waktu_exercise']]=temp2/d* 100, tt2/
    temp2
20.       elif pattern3.findall(row.module_name):
21.         temp3 +=1
22.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['aktifitas']]=temp3/c* 100
23.       elif pattern4.findall(row.module_name):
24.         temp4 +=1
25.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['konten_ilustrasi']]=temp4/a * 100
26.       elif pattern5.findall(row.module_name):
27.         temp5 +=1
28.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['konten_audio']]=temp5/b * 100
29.       elif pattern6.findall(row.module_name):
30.         temp6 +=1
31.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['outline']]=temp6/e * 100
32.       elif pattern7.findall(row.module_name):
33.         temp7 +=1
34.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['example']]=temp7/f * 100
35.       elif pattern8.findall(row.module_name):
36.         temp8 +=1
37.         tt8= row.k_time + time8
38.         dataset.loc[dataset['id'] == str(row.user_id)
    , ['assesment','waktu_asses']]=temp8/h* 100, tt8/te
    mp8
39.     else :
40.       time1=0;time2=0;time3=0;time4=0;time5=0;time6=0
    ;time7=0;time8=0
41.       temp1=0;temp2=0;temp3=0;temp4=0;temp5=0;temp6=0
    ;temp7=0;temp8=0
42.       if pattern1.findall(row.module_name):
43.         init=str(row.user_id)
44.         temp1=1
```

```
45.         time1= row.k_time
46.         dataset.loc[index] = [str(row.user_id),temp1/
    g * 100,0.0,0.0,0.0,0.0,0.0,0.0,0.0,row.k_time/temp
    1,0.0,0.0]
47.       elif pattern2.findall(row.module_name):
48.         init=str(row.user_id)
49.         temp2=1
50.         time2= row.k_time
51.         dataset.loc[index] = [str(row.user_id),0.0,te
    mp2/d * 100,0.0,0.0,0.0,0.0,0.0,0.0,0.0,row.k_time/
    temp2,0.0]
52.       elif pattern3.findall(row.module_name):
53.         init=str(row.user_id)
54.         temp3=1
55.         dataset.loc[index] = [str(row.user_id),0.0,0.
    0,0.0,temp3/c * 100,0.0,0.0,0.0,0.0,0.0,0.0,0.0]
56.       elif pattern4.findall(row.module_name):
57.         init=str(row.user_id)
58.         temp4=1
59.         dataset.loc[index] = [str(row.user_id),0.0,0.
    0,0.0,0.0,temp4/a * 100,0.0,0.0,0.0,0.0,0.0,0.0]
60.       elif pattern5.findall(row.module_name):
61.         init=str(row.user_id)
62.         temp5=1
63.         dataset.loc[index] = [str(row.user_id),0.0,0.
    0,0.0,0.0,0.0,temp5/b * 100,0.0,0.0,0.0,0.0,0.0]
64.       elif pattern6.findall(row.module_name):
65.         init=str(row.user_id)
66.         temp6=1
67.         dataset.loc[index] = [str(row.user_id),0.0,0.
    0,0.0,0.0,0.0,0.0,temp6/e * 100,0.0,0.0,0.0,0.0]
68.       elif pattern7.findall(row.module_name):
69.         init=str(row.user_id);temp7=1
70.         dataset.loc[index] = [str(row.user_id),0.0,0.
    0,0.0,0.0,0.0,0.0,0.0,temp7/f * 100,0.0,0.0,0.0]
71.       elif pattern8.findall(row.module_name):
72.         init=str(row.user_id)
73.         temp8=1;time8= row.k_time
74.         dataset.loc[index] = [str(row.user_id),0.0,0.
    0,temp8/h * 100,0.0,0.0,0.0,0.0,0.0,0.0,0.0,row.k_t
```

Kode Sumber 4.11 Fungsi perhitungan rasio kunjungan pada tiap konten

4.4 Implementasi Pemisahan Dataset

Pada subbab ini akan dijelaskan mengenai implementasi pemisahan dataset.

4.4.1 Implementasi Hold-out

Pemisahan dataset menggunakan metode *hold-out* diimplementasikan pada Kode Sumber 4.12 dimana dataset dipisah menjadi 90% data latih dan 10% data uji dnegan menggunakan fungsi train_test_split pada library scikit-learn. Terdapat parameter yang digunakan, yaitu random_state yang diinisiasi 1, agar ketika *generate* data baru, nilai data latih dan uji tidak berubah. Selain itu, dilakukan pengaturan pada parameter stratify terhadap nilai y, agar pembagian label pada masing-masing data latih dan uji merata.

```
1. X_train, X_test, y_train, y_test = train_test_split
   (X, y, test_size = 0.10, random_state=1,stratify=y)
```

Kode Sumber 4.12 Fungsi pemisahan dataset menggunakan *hold-out*

4.4.2 Implementasi k-fold

Pemisahan dataset menggunakan metode *k-fold* diimplementasikan pada Kode Sumber 4.13 dimana baris 1 menggunakan fungsi StratifiedKFold pada *library* scikit-learn agar pembagian label pada masing-masing data latih dan uji merata. Terdapat parameter yang digunakan, yaitu n_splits atau yang biasa disebut dengan jumlah *fold* sebanyak 10. Sehingga didapatkan data latih sebanyak 9/10 bagian dan data uji sebanyak 1/10 bagian. Selain itu, untuk mengaktifkan parameter random_state, maka parameter shuffle harus bernilai True. Inisiasi random_state dilakukan agar ketika melakukan *generate* data baru, nilai data latih dan uji tidak berubah. Selanjutnya, dilakukan *looping* pada baris 2 untuk masing-masing data latih dan uji dengan menggunakan fungsi split pada *library* scikit-learn. Hasil pemisahan kemudian disimpan seperti pada baris 3-4.

```
1. skf = StratifiedKFold(n_splits=10,shuffle=True, ran
   dom_state = 1)
2. for train_index, test_index in skf.split(X,y):
3.   X_train_cv,X_test_cv=X.iloc[train_index],X.iloc[t
   est_index]
4.   y_train_cv,y_test_cv=y.iloc[train_index],y.iloc[t
   est_index]
```

Kode Sumber 4.13 Fungsi pemisahan dataset menggunakan *k-fold*

4.5 Implementasi Over-sampling

Over-sampling dilakukan dengan menggunakan metode *Borderline Over-sampling* (BOS) yang diimplementasikan menggunakan fungsi SVMSMOTE pada Kode Sumber 4.14. Terdapat parameter yang digunakan, yaitu random_state yang diinisiasi 1, agar ketika *generate* data baru, nilai data latih dan uji tidak berubah. Kemudian pada baris 2, dilakukan *resample* pada data latih.

```
1. smt = SVMSMOTE(random_state=1)
2. X_train, y_train = smt.fit_sample(X_train, y_train)
```

Kode Sumber 4.14 Fungsi BOS

4.6 Implementasi Transformasi Data

Transformasi data dilakukan dengan menggunakan metode normalisasi minmax yang diimplementasikan pada Kode Sumber 4.15 dimana menggunakan fungsi MinMaxScaler pada *library* scikit-learn. Pada baris 2, data latih digunakan sebagai acuan untuk rentang dari normalisasi, sehingga digunakan fungsi fit_transform. Sedangkan pada data uji menggunakan fungsi transform saja karena menggunakan acuan rentang normalisasi pada data latih.

```
1.  scaler = MinMaxScaler()
2.  X_train = scaler.fit_transform(X_train)
3.  X_test = scaler.transform(X_test)
```

Kode Sumber 4.15 Fungsi transformasi data

4.7 Implementasi Klasifikasi dan Evaluasi Gaya Belajar

Pada subbab ini akan dijelaskan mengenai implementasi klasifikasi gaya belajar.

4.7.1 Implementasi Klasifikasi k-Nearest Neighbors

Langkah pertama pada klasifikasi *k-Nearest Neighbors* (k-NN) adalah melakukan pelatihan pada data latih yang diimplementasikan pada Kode Sumber 4.16 dimana memanfaatkan fungsi KNeighborsClassifier pada *library* scikit-learn dengan pengaturan parameter *default*. Dalam hal ini, jumlah k tetangga terdekat sama dengan 5. Kemudian pada baris 2, pelatihan dilakukan dengan menggunakan fungsi fit pada *library* scikit-learn dimana parameter inputnya adalah data latih dan labelnya. Setelah proses pelatihan, model akan disimpan dalam format pickle dengan memanfaatkan fungsi joblib pada *library* scikit-learn.

```
1.  knn = KNeighborsClassifier()
2.  knn.fit(X_train, y_train)
3.  joblib.dump(knn, 'knn.pkl')
```

Kode Sumber 4.16 Fungsi pelatihan k-NN

Langkah selanjutnya adalah model akan diuji dengan menggunakan data uji yang diimplementasikan pada Kode Sumber 4.17. Pada baris 1, variabel y_pred2 digunakan untuk menyimpan hasil prediksi kelas dari data uji menggunakan fungsi *predict* dari *library* scikit-learn. Pada baris 2 digunakan fungsi *classification_report* dari *library* scikit-learn untuk mengevaluasi hasil prediksi kelas terhadap kelas yang sebenarnya untuk masing-masing kelas sehingga didapatkan nilai akurasi, presisi, *recall*, dan *f1-score.*

```
1. y_pred2 = knn.predict(X_test)
2. print(classification_report(y_test, y_pred2))
```

Kode Sumber 4.17 Fungsi evaluasi akurasi, presisi, *recall*, dan *f1-score* pada k-NN

4.7.2 Implementasi Klasifikasi Support Vector Machine

Langkah pertama pada klasifikasi *Support Vector Machine* (SVM) adalah melakukan pelatihan pada data latih yang diimplementasikan pada Kode Sumber 4.18 dimana memanfaatkan fungsi SVC pada *library* scikit-learn dengan pengaturan parameter *default*. Dalam hal ini, kernel yang digunakan adalah RBF. Kemudian pada baris 2, pelatihan dilakukan dengan menggunakan fungsi fit pada *library* scikit-learn dimana parameter inputnya adalah data latih dan labelnya. Setelah proses pelatihan, model akan disimpan dalam format pickle dengan memanfaatkan fungsi joblib pada *library* scikit-learn.

```
1. svclassifier = SVC()
2. svclassifier.fit(X_train, y_train)
3. joblib.dump(svclassifier, 'svm.pkl')
```

Kode Sumber 4.18 Fungsi pelatihan SVM

Langkah selanjutnya adalah model akan diuji dengan menggunakan data uji yang diimplementasikan pada Kode Sumber 4.19. Pada baris 1, variabel y_pred digunakan untuk menyimpan hasil prediksi kelas dari data uji menggunakan fungsi *predict* dari *library* scikit-learn. Pada baris 2 digunakan fungsi *classification_report* dari *library* scikit-learn untuk mengevaluasi hasil prediksi kelas terhadap kelas yang sebenarnya untuk masing-masing kelas sehingga didapatkan nilai akurasi, presisi, *recall*, dan *f1-score.*

```
1. y_pred = svclassifier.predict(X_test)
2. print(classification_report(y_test, y_pred))
```

Kode Sumber 4.19 Fungsi evaluasi akurasi, presisi, *recall*, dan *f1-score* pada SVM

4.7.3 Implementasi Klasifikasi Decision Trees

Langkah pertama pada klasifikasi *Decision Trees* (DTs) adalah melakukan pelatihan pada data latih yang diimplementasikan pada Kode Sumber 4.20 dimana memanfaatkan fungsi DecisionTreeClassifier pada *library* scikit-learn dengan pengaturan parameter *default*. Dalam hal ini, pengukuran kualitas *split* yang digunakan adalah gini. Kemudian pada baris 2, pelatihan dilakukan dengan menggunakan fungsi fit pada *library* scikit-learn dimana parameter inputnya adalah data latih dan labelnya. Setelah proses pelatihan, model akan disimpan dalam format pickle dengan memanfaatkan fungsi joblib pada *library* scikit-learn.

```
1. dt = DecisionTreeClassifier()
2. dt.fit(X_train,y_train)
3. joblib.dump(dt, 'dt.pkl')
```

Kode Sumber 4.20 Fungsi pelatihan DTs

Langkah selanjutnya adalah model akan diuji dengan menggunakan data uji yang diimplementasikan pada Kode Sumber 4.21. Pada baris 1, variabel y_pred digunakan untuk menyimpan hasil prediksi kelas dari data uji menggunakan fungsi *predict* dari *library* scikit-learn. Pada baris 2 digunakan fungsi *classification_report* dari *library* scikit-learn untuk mengevaluasi hasil prediksi kelas terhadap kelas yang sebenarnya untuk masing-masing kelas sehingga didapatkan nilai akurasi, presisi, *recall*, dan *f1-score.*

```
1. y_pred3 = dt.predict(X_test)
2. print(classification_report(y_test, y_pred3))
```

Kode Sumber 4.21 Fungsi evaluasi akurasi, presisi, *recall*, dan *f1-score* pada DTs

BAB V
UJI COBA DAN EVALUASI

Bab ini akan membahas mengenai hasil uji coba sistem yang telah dirancang dan dibuat. Uji coba dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan.

5.1 Lingkungan Uji Coba

Lingkungan uji coba pada Tugas Akhir ini adalah sebuah *notebook* UX430UN. Sistem operasi yang digunakan adalah Windows 10 Home 64-bit. *Notebook* yang digunakan memiliki spesifikasi Intel® Core™ i7-8550U dengan kecepatan 4.0 GHz, *Random Access Memory* (RAM) sebesar 16 GB, dan mempunyai *Video Graphics Adapter* (VGA) yaitu NVIDIA GeForce MX150. Pada sisi perangkat lunak, uji coba pada Tugas Akhir ini dilakukan dengan menggunakan bahasa pemograman Python 3.6, dilengkapi dengan *library* antara lain Pandas, NumPy, Regular Expression, Scikit-learn, SciPy, Imbalanced-learn, dan Matplotlib.

5.2 Dataset

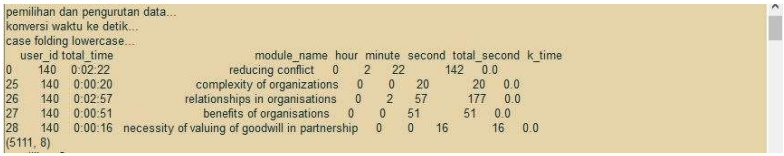
Pada Tugas Akhir ini, dataset yang digunakan pada saat proses pelatihan *k-Nearest Neighbors* (k-NN), *Support Vector Machine* (SVM), dan *Decision Trees* (DTs) menggunakan *data log* dari LMS Universitas Nairobi yang dibagi menjadi data latih dan data uji seperti yang sudah dijelaskan pada subbab 3.1. Pembagian dataset nantinya akan dilakukan uji coba menggunakan metode *hold-out* dan *k-fold* yang selengkapnya dipaparkan pada subbab 5.4.1.

5.3 Hasil Pengujian Seluruh Proses

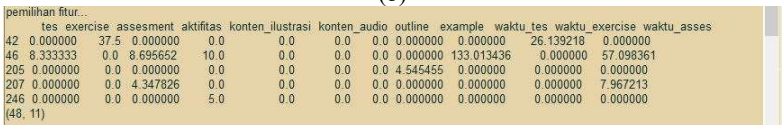
Hasil pengujian seluruh proses dapat dilihat pada Gambar 5.1. Pengujian dimulai dari tahap praproses pada data uji yang meliputi pemilihan *data log*, *case folding,* dan menghilangkan tanda baca. Hasil praproses, akan dilakukan pemilihan fitur untuk selanjutnya diklasifikasi kelasnya menggunakan masing-masing *classifier,* antara lain k-NN, SVM, dan DTs. Hasil dari keseluruhan

proses berupa *string* prediksi dan *ground truth* pada gaya belajar VAK.

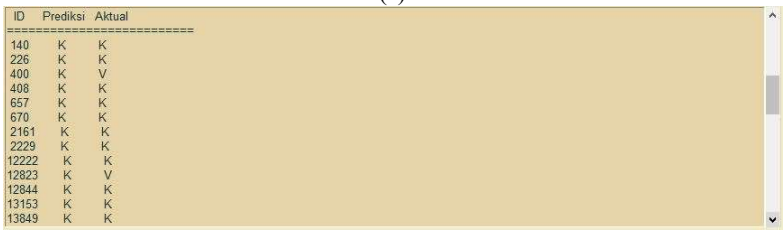
(a)



(b)



(c)



(d)

Gambar 5.1 Hasil pengujian seluruh proses. (a) Pembacaan *input* data uji; (b) Praproses *input*; (c) Pemilihan fitur; (d) Prediksi gaya belajar.

5.4 Skenario Uji Coba

Proses uji coba berguna untuk identifikasi parameter-parameter yang menghasilkan performa model klasifikasi gaya belajar VAK yang paling optimal. Parameter yang tepat akan memberikan hasil yang baik pada saat uji coba. Hasil terbaik dari suatu skenario uji coba akan digunakan untuk skenario uji coba berikutnya. Terdapat empat skenario yang akan diujicobakan yaitu:

1. Uji coba pemisahan dataset
2. Uji coba parameter pada *classifier*
3. Uji coba pada jenis dataset
4. Uji coba *over-sampling*

Parameter awal untuk masing-masing algoritma *classifier* k-NN, SVM, dan DTs merupakan *default* dari bawaan tiap fungsi pada *library* scikit-learn yang dapat berubah pada setiap uji coba yang dilakukan. Pada k-NN memiliki parameter n_neighbors *default* yang bernilai 5, SVM memiliki parameter kernel *default* RBF, dan DTs memiliki parameter *criterion default* gini. Pada setiap skenario uji coba akan ditetapkan nilai parameter yang dapat meningkatkan kinerja *classifier*.

5.4.1 Uji Coba Pemisahan Dataset

Uji coba pemisahan dataset terhadap data latih dan data uji digunakan untuk mengetahui metode pemisahan mana yang menghasilkan performa terbaik dari model klasifikasi gaya belajar VAK menggunakan k-NN, SVM, dan DTs. Metode pemisahan dataset yang akan diujicobakan antara lain *hold-out* dan *k-fold*. Pada metode *hold-out* dilakukan pemisahan data latih sebesar 90% dan data uji 10% dengan melakukan 5 kali pengujian dengan mengatur parameter random_state untuk melakukan pemilihan data latih dan uji secara acak namun tidak memiliki hasil yang berbeda saat diulang kembali. Nilai random_state yang diambil, antara lain 1, 10, 20, 30, dan 40. Sedangkan untuk *k-fold* menggunakan nilai k sebanyak 10, sehingga terdapat 9/10 bagian data latih dan 1/10 bagian data uji. Uji coba pemisahan dataset

menghasilkan perbandingan rata-rata akurasi pada proses pengujian yang dapat dilihat pada Tabel 5.1 dan Tabel 5.2. Didapatkan hasil evaluasi rata-rata akurasi terbaik sebesar 91,32% dimana dataset lengkap dipisah menggunakan 10-*fold*, 90,00% dimana dataset tanpa kuis dipisah menggunakan 10-*fold*, 73,89% dimana dataset tanpa *example*, *outline* dipisah menggunakan 10-*fold*, dan 63,61% dimana dataset lengkap dipisah menggunakan *hold-out*. Dari hasil uji coba maka pemisahan dataset dengan metode 10-*fold* digunakan dalam uji coba selanjutnya.

Tabel 5.1 Perbandingan Rata-Rata Akurasi Menggunakan *Hold-out*

Dataset	**Akurasi *hold-out* 90 persen data latih (%)**			
	k-NN	**SVM**	**DTs**	**Rata-Rata**
Tanpa kuis, *example*, *outline*	63,75	61,67	65,42	**63,61**
Tanpa kuis	88,75	86,67	92,08	89,17
Tanpa *example*, *outline*	72,08	70,42	75,00	72,50
Lengkap	90,00	88,75	95,00	91,25

Tabel 5.2 Perbandingan Rata-Rata Akurasi Menggunakan 10-*fold*

Dataset	**Akurasi 10-*fold* (%)**			
	k-NN	**SVM**	**DTs**	**Rata-Rata**
Tanpa kuis, *example*, *outline*	62,71	60,21	66,88	63,27
Tanpa kuis	89,58	87,29	93,13	**90,00**
Tanpa *example*, *outline*	74,38	71,88	75,42	**73,89**
Lengkap	90,42	88,54	95,00	**91,32**

5.4.2 Uji Coba Parameter pada Classifier

Uji coba parameter pada masing-masing *classifier,* yaitu n_neighbors pada k-NN, kernel pada SVM, dan *criterion* pada DTs digunakan untuk mengetahui parameter mana yang menghasilkan performa terbaik dari model klasifikasi gaya belajar VAK menggunakan k-NN, SVM, dan DTs. n_neighbors yang akan

diujicobakan mulai dari 1 sampai dengan 10. Untuk kernel yang diujicobakan antara lain RBF, *linear*, dan *polynomial*. Sedangkan *criterion* yang diujicobakan antara lain *gini* dan *entropy.* Hasil uji coba parameter pada *classifier* (k-NN, SVM, dan DTs) secara berturut-turut dapat dilihat pada Tabel 5.3 hingga Tabel 5.5.

Pada parameter n_neighbors k-NN didapatkan hasil yang optimal dengan rata-rata akurasi sebesar 79,43% pada n_neighbors=3. Kemudian parameter kernel SVM didapatkan hasil yang optimal dengan rata-rata akurasi sebesar 76,98% pada kernel *Radial Basis Function* (RBF). Terakhir, pada parameter *criterion* DTs didapatkan hasil yang optimal dengan rata-rata akurasi sebesar 82,6075% pada *criterion gini*. Dari hasil uji coba maka parameter n_neighbors=3 pada k-NN, kernel RBF pada SVM, dan *criterion gini* pada DTs digunakan dalam uji coba selanjutnya.

Tabel 5.3 Perbandingan Rata-Rata Akurasi pada Uji Coba Parameter n_neighbors k-NN

n_neighbors	**Akurasi (%)**				
	Lengkap	**Tanpa kuis**	**Tanpa *example*, *outline***	**Tanpa kuis, *example*, *outline***	**Rata-Rata**
1	93,96	91,67	69,38	59,58	78,65
2	91,46	90,00	71,25	56,88	77,40
3	90,42	89,38	74,79	63,13	**79,43**
4	89,58	89,58	73,96	62,92	79,01
5	90,42	89,58	74,38	62,71	79,27
6	89,38	89,79	74,17	61,46	78,70
7	88,96	89,38	72,50	60,42	77,82
8	88,75	89,17	72,50	62,08	78,13
9	88,54	88,33	72,50	62,71	78,02
10	87,92	88,13	72,92	63,54	78,13

Tabel 5.4 Perbandingan Rata-Rata Akurasi pada Uji Coba Parameter kernel SVM

Kernel	**Akurasi (%)**				
	Lengkap	**Tanpa kuis**	**Tanpa *example*, *outline***	**Tanpa kuis, *example*, *outline***	**Rata-Rata**
RBF	88,54	87,29	71,88	60,21	**76,98**
Linear	64,38	64,38	59,38	59,38	61,88
Polynomial	64,17	62,29	59,79	59,17	61,36

Tabel 5.5 Perbandingan Rata-Rata Akurasi pada Uji Coba Parameter criterion DTs

Criterion	**Akurasi (%)**				
	Lengkap	**Tanpa kuis**	**Tanpa *example*, *outline***	**Tanpa kuis, *example*, *outline***	**Rata-Rata**
Gini	95,00	93,13	75,42	66,88	**82,6075**
Entropy	95,21	92,50	76,04	66,67	82,6050

5.4.3 Uji Coba pada Jenis Dataset

Uji coba pada jenis dataset dilakukan untuk mengetahui dataset mana yang menghasilkan performa terbaik dari model klasifikasi gaya belajar VAK menggunakan k-NN, SVM, dan DTs. Selain itu, uji coba ini dilakukan untuk menghindari bias di antara fitur-fitur gaya belajar VAK. Jenis dataset yang diuji cobakan meliputi, dataset gabungan tanpa kuis, *example,* dan *outline,* tanpa kuis, tanpa *example* dan *outline*, tanpa aktifitas, tanpa konten grafis, tanpa konten audio, serta lengkap.

Jenis dataset tanpa kuis, *outline,* dan *example* merupakan dataset yang tidak menggunakan baik fitur kunjungan maupun lamanya waktu yang dihabiskan pada konten kuis, *example*, dan *outline*, sehingga fitur tersisa sebanyak 3 saja. Kemudian pada jenis dataset tanpa kuis merupakan dataset yang tidak menggunakan fitur kunjungan maupun lamanya waktu kunjungan pada konten kuis (*self test, exercise,* dan *assesment*), sehingga fitur menjadi

sebanyak 5 fitur. Selanjutnya, yang dimaksudkan pada jenis dataset tanpa *example* dan *outline* adalah dataset yang tidak menggunakan fitur kunjungan *outline* dan *example*, sehingga fitur menjadi sebanyak 9 fitur. Kemudian, yang dimaksudkan pada jenis dataset tanpa aktifitas merupakan dataset yang tidak menggunakan fitur kunjungan pada konten aktifitas (*review question, activity, case study,* dan forum), sehingga fitur menjadi sebanyak 10 fitur. Lalu, yang dimaksudkan pada jenis dataset tanpa konten grafis, yaitu dataset yang tidak menggunakan fitur kunjungan pada konten grafis, sehingga fitur menjadi sebanyak 10 fitur. Begitu pula pada jenis dataset tanpa konten audio, yaitu dataset yang tidak menggunakan fitur kunjungan pada konten audio, sehingga fitur menjadi sebanyak 10 fitur. Terakhir, jenis dataset lengkap merupakan dataset yang menggunakan semua fitur kunjungan dan lamanya waktu yang dihabiskan, dalam hal ini terdapat sebanyak 11 fitur.

Uji coba pada jenis dataset menghasilkan nilai akurasi, *weighted average precision*, *weighted average recall,* dan *weighted average f1-score* yang dapat dilihat pada Tabel 5.6 hingga Tabel 5.9. Hasilnya, didapatkan dataset lengkap dengan 11 fitur merupakan jenis dataset yang memiliki rata-rata akurasi terbaik sebesar 91,32%. Dari hasil uji coba maka model klasifikasi gaya belajar VAK akan dilatih menggunakan jenis dataset lengkap dalam uji coba selanjutnya.

Tabel 5.6 Perbandingan Rata-Rata Akurasi pada Uji Coba Jenis Dataset

Dataset	**Akurasi (%)**			
	k-NN	**SVM**	**DTs**	**Rata-Rata**
Tanpa kuis, *example*, *outline*	63,13	60,21	67,08	63,47
Tanpa kuis	89,38	87,29	93,13	89,93
Tanpa *example*, *outline*	74,79	71,88	75,42	74,03
Tanpa aktifitas	84,17	85,42	87,50	85,70
Tanpa konten grafis	90,00	87,71	94,38	90,70
Tanpa konten audio	90,00	88,75	95,00	91,25
Lengkap	90,42	88,54	95,00	**91,32**

Tabel 5.7 Perbandingan Rata-Rata *Precision* pada Uji Coba Jenis Dataset

Dataset	**Weighted Average Precision (%)**			
	k-NN	**SVM**	**DTs**	**Rata-Rata**
Tanpa kuis, *example*, *outline*	57,63	43,19	66,06	55,63
Tanpa kuis	88,99	81,46	94,09	88,18
Tanpa *example*, *outline*	67,10	59,86	70,08	65,68
Tanpa aktifitas	83,27	78,45	87,16	82,96
Tanpa konten grafis	88,76	81,68	94,95	88,46
Tanpa konten audio	89,21	82,82	95,82	**89,28**
Lengkap	89,75	82,07	95,70	89,17

Tabel 5.8 Perbandingan Rata-Rata *Recall* pada Uji Coba Jenis Dataset

Dataset	Weighted Average Recall (%)			
	k-NN	**SVM**	**DTs**	**Rata-Rata**
Tanpa kuis, *example*, *outline*	63,13	60,20	67,08	63,47
Tanpa kuis	89,38	87,29	93,13	89,93
Tanpa *example*, *outline*	74,79	71,88	75,42	74,03
Tanpa aktifitas	84,17	85,42	87,50	85,70
Tanpa konten grafis	90,00	87,71	94,38	90,70
Tanpa konten audio	90,00	88,75	95,00	91,25
Lengkap	90,42	88,54	95,00	**91,32**

Tabel 5.9 Perbandingan Rata-Rata *F1-Score* pada Uji Coba Jenis Dataset

Dataset	Weighted Average F1-Score (%)			
	k-NN	**SVM**	**DTs**	**Rata-Rata**
Tanpa kuis, *example*, *outline*	57,13	47,66	63,04	55,94
Tanpa kuis	88,41	83,87	92,55	88,28
Tanpa *example*, *outline*	68,42	62,72	70,45	67,20
Tanpa aktifitas	82,96	81,26	86,61	83,61
Tanpa konten grafis	89,56	83,91	94,23	89,23
Tanpa konten audio	89,22	85,23	95,09	**89,85**
Lengkap	89,58	84,75	95,06	89,80

5.4.4 Uji Coba Over-sampling

Uji coba *over-sampling* dilakukan untuk mengetahui apakah teknik *over-sampling* dapat meningkatkan performa model klasifikasi gaya belajar VAK. Teknik *over-sampling* yang digunakan, yaitu *Borderline Over-sampling* (BOS). BOS diujicobakan pada data latih ketiga *classifier,* yaitu k-NN, SVM,

dan DTs. Hasil uji coba akurasi pada dataset yang menggunakan tanpa *over-sampling* dan *over-sampling* dapat dilihat pada Tabel 5.10.

Didapatkan rata-rata akurasi k-NN terbaik sebesar 90,42% pada penggunaan data latih hasil tanpa *over-sampling* maupun *over-sampling*, rata-rata akurasi SVM terbaik sebesar 89,17% pada penggunaan data latih hasil *over-sampling*, dan rata-rata akurasi DTs terbaik sebesar 95,63% pada penggunaan data latih hasil *over-sampling*.

Tabel 5.10 Perbandingan Akurasi pada Uji Coba *Over-sampling*

Dataset	**Akurasi (%)**		
	k-NN	**SVM**	**DTs**
Tanpa *over-sampling*	**90,42**	88,54	95,00
Over-sampling	**90,42**	**89,17**	**95,63**

5.5 Hasil dan Evaluasi

Pada uji coba pemisahan dataset, diperoleh hasil rata-rata akurasi terbaik pada ketiga *classifier* sebesar 91,32% pada dataset lengkap, 90,00% pada dataset tanpa kuis, dan 73,89% dimana menggunakan 10-*fold*. Teknik 10-*fold* tepat digunakan untuk data multi label yang tidak seimbang karena dapat menghasilkan performa yang tidak bias dan hanya kebetulan saja apabila dilakukan satu kali evaluasi seperti pada teknik *hold-out.* Sedangkan pada metode 10-*fold,* dataset dapat dievaluasi secara keseluruhan dengan merata.

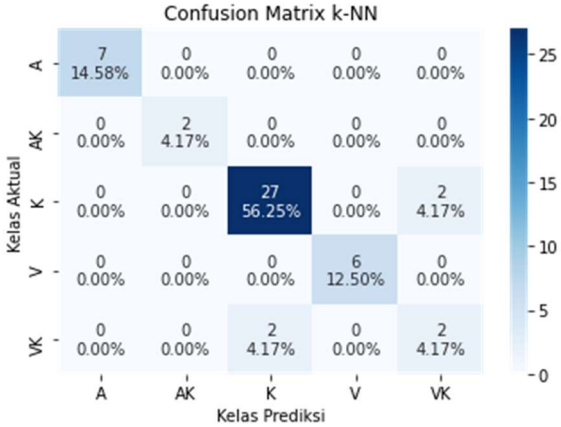
Pada uji coba penggantian parameter pada masing-masing *classifier,* k-NN memperoleh hasil akurasi terbaik sebesar 79,43% pada n_neighbors=3. Nilai tetangga terdekat 3 tepat digunakan untuk k-NN karena menghasilkan performa yang lebih baik dibandingkan dengan nilai tetangga terdekat lainnya. Pada SVM, diperoleh hasil akurasi terbaik sebesar 76,98% pada kernel RBF. Kernel RBF tepat digunakan untuk SVM karena dataset yang digunakan bersifat *non linear* serta menghasilkan performa yang

lebih baik dibandingkan dengan kernel *non linear* lainnya. Kemudian pada DTs, diperoleh hasil akurasi terbaik sebesar 82,6075% pada *criterion gini*. Pengukuran kualitas *split* dengan *gini* tepat digunakan untuk DTs karena menghasilkan performa yang lebih baik dibandingkan dengan *entropy.*

Selanjutnya pada uji coba jenis dataset, didapatkan bahwa dataset lengkap merupakan jenis dataset yang memiliki rata-rata akurasi terbaik sebesar 91,32%. Dataset lengkap tepat digunakan untuk melatih model klasifikasi gaya belajar VAK karena keseluruhan fitur pada dataset, yakni kunjungan dan lamanya waktu yang dihabiskan, dibutuhkan pada pelatihan model dan menghasilkan performa yang lebih baik dibandingkan dengan menghilangkan beberapa fitur.

Pada uji coba *over-sampling* menggunakan BOS pada masing-masing *classifier,* k-NN memperoleh hasil akurasi terbaik sebesar 90,42% pada penggunaan data latih hasil *over-sampling* maupun tanpa *over-sampling*. Pada SVM dan DTs, diperoleh masing-masing hasil akurasi terbaik sebesar sebesar 89,17%, dan 95,63% pada penggunaan data latih hasil *over-sampling*. Sehingga penggunaan *over-sampling* dengan BOS merupakan tepat karena dapat menghasilkan performa model yang lebih baik pada data latih yang tidak seimbang.

Terjadi beberapa misklasifikasi dalam proses klasifikasi gaya belajar VAK. Misklasifikasi pada k-NN terjadi karena terdapat *false positive* dimana beberapa gaya belajar kinestetik diklasifikasi sebagai gabungan gaya belajar visual-kinestetik, dan sebaliknya seperti yang dapat dilihat pada Gambar 5.2 dan Tabel 5.11. Gaya belajar yang dikategorikan dalam *false positive* tersebut memiliki rasio fitur aktifitas 10,0-20,0 dan rasio fitur konten ilustrasi antara 0,0-10,0 sedangkan *true positive* fitur aktifitas juga memiliki rasio 10,0-20,0 dan *true positive* rasio fitur konten ilustrasi antara 0,0-30,0.

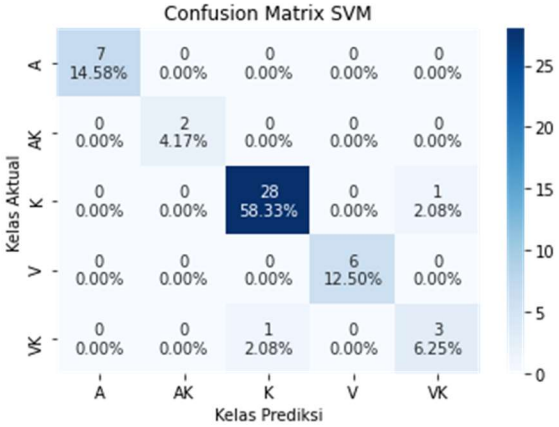


Gambar 5.2 *Confusion matrix* pada metode k-NN

Tabel 5.11 *False Positive* Prediksi Gaya Belajar pada k-NN

user_id	aktifitas	konten_ilustrasi	Kelas Prediksi	Kelas Aktual
226	10,0	0,0	VK	K
13849	20,0	0,0	VK	K
214003	15,0	10,0	K	VK

Kemudian, misklasifikasi pada SVM juga terjadi karena terdapat *false positive* dimana gaya belajar kinestetik diklasifikasi sebagai gabungan gaya belajar visual-kinestetik dan sebaliknya seperti yang dapat dilihat pada Gambar 5.3 dan Tabel 5.12. Gaya belajar yang dikategorikan dalam *false positive* tersebut memiliki rasio fitur konten ilustrasi antara 0,0-10,0 sedangkan *true positive* konten ilustrasi antara 0,0-30,0.

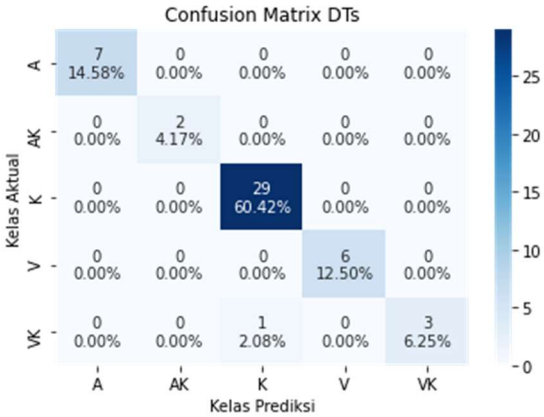


Gambar 5.3 *Confusion matrix* pada metode SVM

Tabel 5.12 *False Positive* Prediksi Gaya Belajar pada SVM

user_id	konten_ilustrasi	Kelas Prediksi	Kelas Aktual
226	0,0	VK	K
214003	10,0	K	VK

Terakhir, misklasifikasi pada DTs terjadi karena terdapat *false positive* dimana gabungan gaya belajar visual-kinestetik diklasifikasi sebagai gaya belajar kinestetik seperti yang dapat dilihat pada Gambar 5.4 Tabel 5.13. Gaya belajar yang dikategorikan dalam *false positive* tersebut memiliki rasio fitur aktifitas 15,0 dan rasio fitur *assesment* 8,70 sedangkan *true positive* memiliki rasio fitur aktifitas antara 10,0-15,0 dan rasio fitur *assesment* juga sebesar 8,70. Semua keadaan misklasifikasi pada SVM, k-NN, dan DTs dapat diatasi dengan menambah data latih pada klasifikasi gaya belajar VAK.

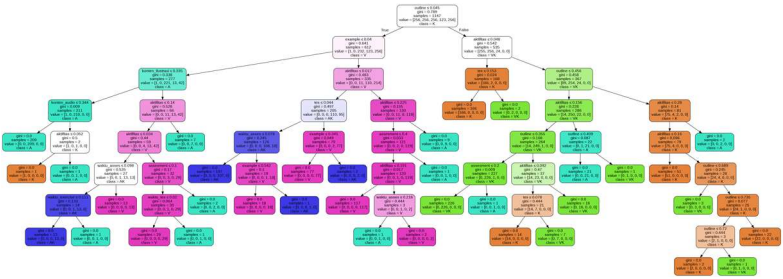


Gambar 5.4 *Confusion matrix* pada metode DTs

Tabel 5.13 *False Positive* Prediksi Gaya Belajar pada DTs

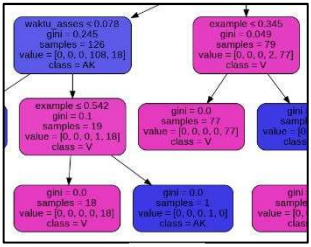
user_id	assesment	aktifitas	Kelas Prediksi	Kelas Aktual
215049	8,70	15,0	K	VK

Secara keseluruhan, model klasifikasi gaya belajar VAK pada DTs memiliki performa terbaik dengan melakukan pelatihan pada data latih hasil *over-sampling*. Berdasarkan hal tersebut, didapatkan visualisasi pohon keputusan pada model yang telah dibangun seperti pada Gambar 5.5. Perbedaan warna pada *leaf node* mengidentifikasikan warna label masing-masing gaya belajar. Seperti halnya, label V berwarna magenta, label A berwarna *tosca*, label K berwarna coklat, label VK berwarna hijau, dan label AK berwarna biru tua.

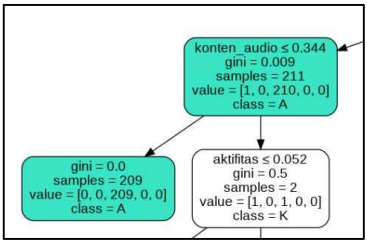


Gambar 5.5 Hasil pohon keputusan pada pembangunan model DTs dengan *over-sampling*

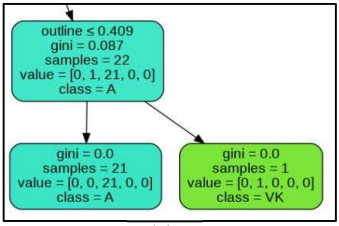
Apabila diamati pada Gambar 5.5, didapatkan relasi fitur gaya belajar pada Gambar 5.6. Pada gaya belajar visual berhubungan kuat dengan fitur *example*, gaya belajar auditori berhubungan kuat dengan fitur *outline* dan konten audio, gaya belajar kinestetik berhubungan kuat dengan fitur aktifitas dan tes, gabungan gaya belajar visual-kinestetik berhubungan kuat dengan fitur aktifitas dan *assesment*, sedangkan gabungan gaya belajar auditori-kinestetik berhubungan kuat dengan fitur waktu *assesment* dan waktu *exercise*.



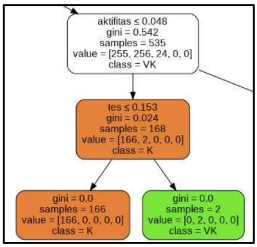
(a)



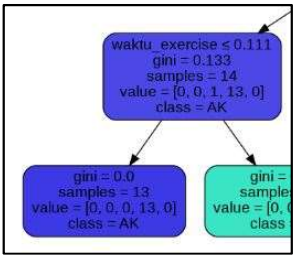
(b)



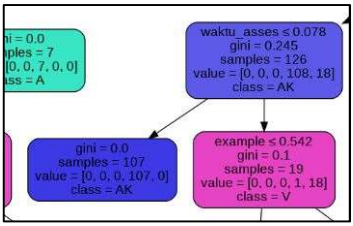
(c)



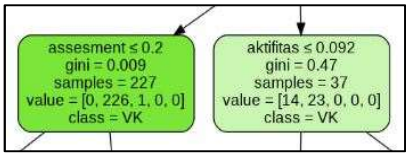
(d)



(e)



(f)



(g)

Gambar 5.6 Relasi fitur pada gaya belajar. (a) Visual; (b) Auditori; (c) Auditori; (d) Kinestetik; (e) Auditori-kinestetik; (f) Auditori-kinestetik; (g) Visual-kinestetik.

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan yang didasari oleh hasil uji coba yang telah dilakukan pada bab sebelumnya. Kesimpulan nantinya sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut di masa yang akan datang.

6.1 Kesimpulan

Dalam pengerjaan Tugas Akhir ini setelah melalui tahap perancangan aplikasi, implementasi metode, serta uji coba, diperoleh kesimpulan sebagai berikut:

1. Cara melakukan ekstraksi *data log* menjadi fitur-fitur gaya belajar, yaitu dengan melakukan praproses (pemilihan data, *case folding*, menghilangkan angka dan tanda baca, dan penghapusan *outlier*). Data hasil praproses, selanjutnya dilakukan perhitungan rasio kunjungan dan lama waktu yang dihabiskan. Berdasarkan hasil uji coba, jenis dataset lengkap memiliki rata-rata akurasi terbaik sebesar 91,32%. Hal ini dikarenakan keseluruhan fitur memang diperlukan dalam pengklasifian multi label gaya belajar VAK.
2. Cara melakukan klasifikasi multi label gaya belajar VAK, yaitu dengan menggunakan metode *supervised learning,* antara lain k-NN, SVM, dan DTs. Berdasarkan uji coba parameter pada k-NN, SVM, dan DTs untuk model klasifikasi gaya belajar VAK didapatkan akurasi terbaik secara berturut-turut sebesar 79,43% menggunakan n_neighbors berjumlah 3, 76,98% menggunakan kernel RBF, dan 82,6075% menggunakan *criterion gini.*
3. Sistem klasifikasi multi label gaya belajar VAK telah berhasil diimplementasikan dengan akurasi terbaik

sebesar 95,63% pada DTs dengan data latih yang dilakukan *over-sampling*.

6.2 Saran

Pada sistem klasifikasi multi label gaya belajar VAK menggunakan *supervised learning,* perlu adanya penambahan skor pada setiap modul untuk digunakan sebagai penentu apakah bersesuaian dengan fitur yang digunakan pada gaya belajar VAK. Kemudian perlu memberi rekomendasi kepada tenaga pendidik untuk menambahkan tipe konten pada setiap modul.

Selain itu, pada dataset dapat dilakukan penambahan sampel *data log* untuk melengkapi label VA dan VAK yang belum terpenuhi agar dapat melengkapi semua gaya belajar VAK. Kemudian dalam penentuan gaya belajar (*ground truth*) pada *data log* dapat mempertimbangkan keterlibatan ahli (*expert*). Jika memungkinkan, penggunaan *data log* pada *e-learning* yang ada di Indonesia juga bisa menjadi pengembangan kedepannya.

DAFTAR PUSTAKA

[1] The Economic Times, "Definition of 'E-learning'," [Online]. Available: https://economictimes.indiatimes.com/definition/e-Learning?from=mdr. [Accessed 12 November 2019].

[2] L. M. ,. R. O. Charles Lwande, "Behaviour Prediction in a Learning Management System," in *IST-Africa 2019*, Afrika, 2019.

[3] D. M. R. S. Raras Setyo Retno, "ANALISIS GAYA BELAJAR SISWA PADA PEMBELAJARAN IPA KELAS V DI SDN 1 NGLURUP KABUPATEN PONOROGO," pp. 336-342, 2019.

[4] P. Wiedarti, Seri Manual GLS Pentingnya Memahami Gaya Belajar, Jakarta: Direktorat Jenderal Pendidikan Dasar dan Menengah Kementerian Pendidikan dan Kebudayaan , 2018.

[5] S. Gholami and M. S. Bagheri, "Relationship between VAK Learning Styles and Problem Solving Styles regarding Gender and Students' Fields of Study," *Journal of Language Teaching and Research,* vol. 4, no. 4, pp. 700-706, 2013.

[6] H. E. H. E. Z. Mawardi Effendi, "Implikasi Gaya Belajar dalam Desain Blended Learning," *Jurnal Teknologi Informasi dan Pendidikan,* vol. 8, no. 1, pp. 72-80, 2015.

[7] E. S. Amir, M. Sumadyo, D. I. Sensuse, Y. G. Sucahyo and H. B. Santoso, "Automatic Detection of Learning Styles in Learning Management System by Using Literature-Based Method and Support Vector Machine," *ICACSIS 2016,* pp. 141-144, 2016.

[8] wisetail, "Is There A Difference Between eLearning and an LMS?," [Online]. Available:

https://www.wisetail.com/lms-questions/elearning-vs-lms/. [Accessed 12 April 2020].

[9] Wikipedia, "Log file," [Online]. Available: https://en.wikipedia.org/wiki/Log_file. [Accessed 16 April 2020].

[10] M. P. P. Liyanage, K. S. L. Gunawardena and M. Hirakawa, "Using Learning Styles to Enhance Learning Management Systems," *International Journal on Advances in ICT for Emerging Regions (ICTer),* vol. 7, no. 6, pp. 1-10, Agustus 2014.

[11] H. Ateia and T. Hamtini, "Designing and Implementing of Dynamic Technique for Detecting Learning Style Using Literature Based Approach," *International Journal of Advanced Science and Technology,* pp. 2-14, 2016.

[12] M. P. P. Liyanage, K. L. Gunawardena1 and M. Hirakawa, "Detecting Learning Styles in Learning Management Systems Using Data Mining," *Madura Prabhani Pitigala Liyanage1,a) K.S. Lasith Gunawardena1,b) Masahito Hirakawa1,c),* vol. 4, no. 4, pp. 740-749, 2016.

[13] M. S. Mohapatra, D. D. Ramesh, M. C. Dash and M. P. C. Behera, "DATA MINING - A DOMAIN SPECIFIC ANALYTICAL TOOL FOR DECISION MAKING," *International Journal of Emerging Trends in Engineering Research (IJETER),* vol. 3, no. 6, pp. 157-167, 2015.

[14] Seek Digital Library, "Data Cleaning in Knowledge Discovery Database (KDD)-Data Mining," [Online]. Available: https://www.techrepublic.com/resource-library/whitepapers/data-cleaning-in-knowledge-discovery-database-kdd-data-mining/. [Accessed 11 April 2020].

[15] Software Testing Help, "Data Mining Process: Models, Process Steps & Challenges Involved," 10 November 2019. [Online]. Available:

https://www.softwaretestinghelp.com/data-mining-process/. [Accessed 11 April 2020].

[16] M. Fatima and J. Kurmi, "Comparative Analysis of Outlier Detection Methods," *International Journal of Engineering Research in Computer Science and Engineering,* vol. 5, no. 4, pp. 252-356, 2018.

[17] E. M. Nkechinyere, I. A. I. and O. Idochi, "Comparison of Different Methods of Outlier Detection in Univariate Time Series Data," *International Journal for Research in Mathematics and Statistics,* vol. 1, no. 1, pp. 55-83, 2015.

[18] A. Saka, "Practical Guide to Outlier Detection Methods," towards data science, 12 September 2019. [Online]. Available: https://towardsdatascience.com/practical-guide-to-outlier-detection-methods-6b9f947a161e. [Accessed 29 Maret 2020].

[19] R. Siringoringo, "KLASIFIKASI DATA TIDAK SEIMBANG MENGGUNAKAN ALGORITMA SMOTE DAN k-NEAREST NEIGHBOR," *Jurnal ISD,* vol. 3, no. 1, pp. 44-48, 2018.

[20] A. Syukron and A. Subekti, "Penerapan Metode Random Over-Under Sampling dan Random Forest untuk Klasifikasi Penilaian Kredit," *JURNAL INFORMATIKA,* vol. 5, no. 2, pp. 175-185, 2018.

[21] H. M. Nguyen, E. W. Cooper and K. Kamei, "Borderline Over-sampling for Imbalanced Data Classification," *Fifth International Workshop on Computational Intelligence & Applications,* pp. 24-30, 2009.

[22] A. Torfi, "Cross-validation," [Online]. Available: https://machine-learning-course.readthedocs.io/en/latest/content/overview/crossvalidation.html. [Accessed 16 April 2020].

[23] E. Allibhai, "Hold-out vs. Cross-validation in Machine Learning," 3 Oktober 2018. [Online]. Available:

https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f. [Accessed 16 April 2020].

[24] Microsoft Azure, "Normalize Data," 5 Juni 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/normalize-data. [Accessed 12 April 2020].

[25] codeacademy, "Normalization," [Online]. Available: https://www.codecademy.com/articles/normalization. [Accessed 11 April 2020].

[26] D. Satria, "ML#2: Jenis-jenis Learning dalam ML," 24 Juni 2018. [Online]. Available: https://medium.com/tulisan-ibe/ml-2-jenis-jenis-learning-dalam-ml-88ba82973067. [Accessed 25 November 2019].

[27] A. Navlani, "KNN Classification using Scikit-learn," 2 Agustus 2018. [Online]. Available: https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn. [Accessed 27 Oktober 2019].

[28] R. Gandhi, "K Nearest Neighbours — Introduction to Machine Learning Algorithms," 14 Juni 2018. [Online]. Available: https://towardsdatascience.com/k-nearest-neighbours-introduction-to-machine-learning-algorithms-18e7ce3d802a. [Accessed 27 Oktober 2019].

[29] ,. R. C. W. D. E. R. Arif Pratama, "Implementasi Algoritme Support Vector Machine (SVM) untuk Prediksi Ketepatan Waktu Kelulusan Mahasiswa," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* vol. 2, pp. 1704-1708, 2018.

[30] E. Suwandi, 2014. [Online]. Available: https://library.binus.ac.id/eColls/eThesisdoc/Bab2/TSA-2014-0085%202.pdf. [Accessed 27 November 2019].

[31] R. A. W. Ari Muzakir, "Model Data Mining sebagai Prediksi Penyakit Hipertensi Kehamilan dengan Teknik Decision Tree," *Scientific Journal of Informatics,* vol. 3, no. 1, pp. 19-26, 2016.

[32] Learn by Marketing, "Decision Tree Flavors: Gini Index and Information Gain," 27 Februari 2016. [Online]. Available: http://www.learnbymarketing.com/481/decision-tree-flavors-gini-info-gain/. [Accessed 10 Mei 2020].

[33] "Analisa Kepuasan Konsumen Menggunakan Klasifikasi Decision Tree Di Restoran Dapur Solo (Cabang Kediri)," *Generation Journal,* vol. 2, no. 1, pp. 9-18, 2018.

[34] E. Prasetyo, Data Mining: Konsep dan Aplikasi menggunakan Matlab, Yogyakarta: Andi Offset, 2012.

[35] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," vol. 45, no. 4, pp. 427-437, 2009.

[36] "About Python," Python, [Online]. Available: https://www.python.org/about/. [Accessed 28 Maret 2020].

[37] Pandas Team, "Pandas Documentation," [Online]. Available: https://pandas.pydata.org/docs/index.html. [Accessed 2020 Maret 28].

[38] Wes McKinney and the Pandas Development Team, "pandas: powerful Python data analysis Release 1.0.3," 18 Maret 2020. [Online]. Available: https://pandas.pydata.org/docs/pandas.pdf. [Accessed 28 Maret 2020].

[39] "NumPy," NumPy, [Online]. Available: http://www.numpy.org/. [Accessed 28 Maret 2020].

[40] A. Kuchling, "Regular Expression HOWTO," [Online]. Available: https://docs.python.org/3/howto/regex.html. [Accessed 28 Maret 2020].

[41] "Scikit-learn," Scikit-learn, [Online]. Available: http://scikit-learn.org/stable/index.html. [Accessed 28 Maret 2020].

[42] SciPy, "scipy 1.4.1," SciPy, [Online]. Available: https://pypi.org/project/scipy/. [Accessed 2020 Maret 28].

[43] Python, "warnings — Warning control," [Online]. Available: https://docs.python.org/3/library/warnings.html. [Accessed 19 April 2020].

[44] G. Lemaıtre, F. Nogueira and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," *Journal of Machine Learning Research,* vol. 18, no. 17, pp. 1-5, 2017.

[45] Python, "statistics — Mathematical statistics functions," [Online]. Available: https://docs.python.org/3/library/statistics.html. [Accessed 3 Mei 2020].

[46] "Matplotlib," Matplotlib, [Online]. Available: https://matplotlib.org/index.html. [Accessed 28 Maret 2020].

[47] PySimpleGUI, "PySimpleGUI," [Online]. Available: https://pysimplegui.readthedocs.io/en/latest/. [Accessed 28 April 2020].

[48] Python, "subprocess — Subprocess management," [Online]. Available: https://docs.python.org/3/library/subprocess.html. [Accessed 28 April 2020].

LAMPIRAN

L.1 Hasil Uji Coba Hold-out pada Dataset Lengkap

random_state	Akurasi (%)		
	k-NN	SVM	DTs
1	89,58	89,58	95,83
10	91,67	87,50	95,83
20	87,50	87,50	93,75
30	89,58	89,58	97,92
40	91,67	89,58	91,67
Rata-Rata	90,00	88,75	95,00

L.2 Hasil Uji Coba Hold-out pada Dataset Tanpa Kuis

random_state	Akurasi (%)		
	k-NN	SVM	DTs
1	89,58	89,58	95,83
10	85,42	83,33	87,50
20	91,67	85,42	91,67
30	89,58	87,50	93,75
40	87,50	87,50	91,67
Rata-Rata	88,75	86,67	92,08

L.3 Hasil Uji Coba Hold-out pada Dataset Tanpa Example dan Outline

random_state	Akurasi (%)		
	k-NN	SVM	DTs
1	72,92	75,00	72,92
10	77,08	72,92	75,00
20	70,83	70,83	70,83
30	68,75	75,00	81,25
40	70,83	58,33	75,00
Rata-Rata	72,08	70,42	75,00

L.4 Hasil Uji Coba Hold-out pada Dataset Tanpa Kuis, Example, dan Outline

random_state	Akurasi (%)		
	k-NN	SVM	DTs
1	64,58	64,58	64,58
10	62,50	60,42	64,58
20	68,75	62,50	62,50
30	58,33	60,42	66,67
40	64,58	60,42	68,75
Rata-Rata	63,75	61,67	65,42

L.5 Hasil Uji Coba 10-fold pada Dataset Lengkap

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
1	93,75	91,67	95,83
2	87,50	89,58	93,75
3	95,83	87,50	93,75
4	95,83	89,58	97,92
5	85,42	87,50	93,75
6	87,50	87,50	93,75
7	85,42	87,50	93,75
8	93,75	93,75	93,75
9	87,50	85,42	95,83
10	91,67	85,42	97,92
Rata-Rata	90,42	88,54	95,00

L.6 Hasil Uji Coba 10-fold pada Dataset Tanpa Kuis

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
1	89,58	87,50	91,67
2	91,67	87,50	93,75
3	89,58	89,58	91,67
4	91,67	89,58	95,83
5	89,58	87,50	93,75

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
6	87,50	85,42	93,75
7	87,50	85,42	93,75
8	89,58	89,58	89,58
9	91,67	85,42	93,75
10	87,50	85,42	93,75
Rata-Rata	89,58	87,29	93,13

L.7 Hasil Uji Coba 10-fold pada Dataset Tanpa Example dan Outline

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
1	77,08	60,42	77,08
2	77,08	77,08	79,17
3	75,00	72,92	72,92
4	75,00	77,08	79,17
5	72,92	77,08	75,00
6	70,83	70,83	72,92
7	75,00	75,00	77,08
8	77,08	75,00	70,83
9	70,83	60,42	77,08
10	72,92	72,92	72,92
Rata-Rata	74,38	71,88	75,42

L.8 Hasil Uji Coba 10-fold pada Dataset Tanpa Kuis, Example, dan Outline

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
1	68,75	60,42	66,67
2	62,50	62,50	64,58
3	68,75	58,33	68,75
4	64,58	60,42	70,83
5	62,50	64,58	72,92
6	64,58	58,33	64,58

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
7	60,42	60,42	64,58
8	47,92	60,42	60,42
9	68,75	60,42	75,00
10	58,33	56,25	60,42
Rata-Rata	62,71	60,21	66,88

L.9 Hasil Uji Coba pada Parameter n_neighbors k-NN pada Dataset Lengkap

Fold ke-	Akurasi (%)				
	K=1	K=2	K=3	K=4	K=5
1	93,75	91,67	91,67	91,67	93,75
2	95,83	93,75	89,58	89,58	87,50
3	91,67	93,75	91,67	91,67	95,83
4	97,92	93,75	93,75	95,83	95,83
5	95,83	91,67	91,67	85,42	85,42
6	89,58	87,50	87,50	87,50	87,50
7	93,75	87,50	85,42	85,42	85,42
8	93,75	93,75	91,67	93,75	93,75
9	91,67	89,58	91,67	87,50	87,50
10	95,83	91,67	89,58	87,50	91,67
Rata-Rata	93,96	91,46	90,42	89,58	90,42

Fold ke-	Akurasi (%)				
	K=6	K=7	K=8	K=9	K=10
1	91,67	91,67	91,67	91,67	91,67
2	89,58	87,50	87,50	89,58	87,50
3	89,58	93,75	89,58	89,58	87,50
4	93,75	91,67	91,67	87,50	89,58
5	85,42	85,42	87,50	87,50	87,50
6	87,50	87,50	87,50	87,50	87,50
7	85,42	85,42	85,42	85,42	85,42
8	93,75	91,67	93,75	91,67	89,58

Fold ke-	Akurasi (%)				
	K=6	K=7	K=8	K=9	K=10
9	87,50	87,50	87,50	87,50	89,58
10	89,58	87,50	85,42	87,50	83,33
Rata-Rata	89,38	88,96	88,75	88,54	87,92

L.10 Hasil Uji Coba pada Parameter n_neighbors k-NN pada Dataset Tanpa Kuis

Fold ke-	Akurasi (%)				
	K=1	K=2	K=3	K=4	K=5
1	83,33	85,42	85,42	87,50	89,58
2	93,75	91,67	91,67	89,58	91,67
3	87,50	89,58	89,58	87,50	89,58
4	97,92	91,67	89,58	93,75	91,67
5	97,92	91,67	91,67	89,58	89,58
6	91,67	89,58	87,50	87,50	87,50
7	91,67	89,58	87,50	87,50	87,50
8	93,75	91,67	93,75	89,58	89,58
9	89,58	91,67	91,67	91,67	91,67
10	89,58	87,50	85,42	91,67	87,50
Rata-Rata	91,67	90,00	89,38	89,58	89,58

Fold ke-	Akurasi (%)				
	K=6	K=7	K=8	K=9	K=10
1	91,67	91,67	89,58	89,58	89,58
2	89,58	89,58	87,50	87,50	87,50
3	89,58	93,75	91,67	91,67	91,67
4	93,75	91,67	91,67	89,58	89,58
5	89,58	87,50	89,58	85,42	85,42
6	87,50	85,42	89,58	89,58	87,50
7	87,50	87,50	87,50	87,50	87,50
8	89,58	89,58	89,58	89,58	89,58
9	87,50	87,50	85,42	85,42	85,42
10	91,67	89,58	89,58	87,50	87,50

Fold ke-	Akurasi (%)				
	K=6	K=7	K=8	K=9	K=10
Rata-Rata	89,79	89,38	89,17	88,33	88,13

L.11 Hasil Uji Coba pada Parameter n_neighbors k-NN pada Dataset Tanpa Example dan Outline

Fold ke-	Akurasi (%)				
	K=1	K=2	K=3	K=4	K=5
1	77,08	77,08	77,08	75,00	77,08
2	75,00	77,08	79,17	77,08	77,08
3	70,83	75,00	75,00	75,00	75,00
4	77,08	62,50	77,08	77,08	75,00
5	75,00	72,92	75,00	72,92	72,92
6	70,83	68,75	70,83	70,83	70,83
7	64,59	72,92	75,00	72,92	75,00
8	72,92	75,00	75,00	77,08	77,08
9	41,67	62,50	70,83	70,83	70,83
10	68,75	68,75	72,92	70,83	72,92
Rata-Rata	69,38	71,25	74,79	73,96	74,38

Fold ke-	Akurasi (%)				
	K=6	K=7	K=8	K=9	K=10
1	77,08	75,00	75,00	75,00	75,00
2	77,08	72,92	72,92	72,92	70,83
3	72,92	77,08	75,00	75,00	75,00
4	77,08	72,92	70,83	68,75	70,83
5	72,92	68,75	72,92	72,92	72,92
6	70,83	72,92	72,92	72,92	72,92
7	75,00	70,83	72,92	72,92	72,92
8	77,08	75,00	75,00	72,92	75,00
9	68,75	68,75	68,75	70,83	72,92
10	72,92	70,83	68,75	70,83	70,83
Rata-Rata	74,17	72,50	72,50	72,50	72,92

L.12 Hasil Uji Coba pada Parameter n_neighbors k-NN pada Dataset Tanpa Kuis, Example, dan Outline

Fold ke-	Akurasi (%)				
	K=1	K=2	K=3	K=4	K=5
1	62,50	66,67	66,67	66,67	68,75
2	58,33	56,25	62,50	62,50	62,50
3	62,50	45,83	60,42	60,42	68,75
4	62,50	60,42	64,58	66,67	64,58
5	66,67	64,58	64,58	64,58	62,50
6	64,58	54,17	64,58	60,42	64,58
7	50,00	56,25	60,42	60,42	60,42
8	52,08	52,08	60,42	60,42	47,92
9	62,50	58,33	68,75	68,75	68,75
10	54,17	54,17	58,33	58,33	58,33
Rata-Rata	59,58	56,88	63,13	62,92	62,71

Fold ke-	Akurasi (%)				
	K=6	K=7	K=8	K=9	K=10
1	66,67	60,42	56,25	66,67	66,67
2	62,50	62,50	62,50	62,50	62,50
3	68,75	70,83	70,83	70,83	70,83
4	62,50	56,25	66,67	64,58	64,58
5	64,58	64,58	64,58	62,50	62,50
6	64,58	64,58	64,58	64,58	64,58
7	50,00	50,00	50,00	50,00	60,42
8	47,92	47,92	58,33	58,33	58,33
9	68,75	68,75	68,75	68,75	66,67
10	58,33	58,33	58,33	58,33	58,33
Rata-Rata	61,46	60,42	62,08	62,71	63,54

L.13 Hasil Uji Coba pada Parameter Kernel SVM pada Dataset Lengkap

Fold ke-	Akurasi (%)		
	RBF	Linear	Polynomial
1	91,67	66,67	68,75
2	89,58	66,67	66,67
3	87,50	64,58	64,58
4	89,58	66,67	66,67
5	87,50	62,50	70,83
6	87,50	62,50	60,42
7	87,50	58,33	58,33
8	93,75	62,50	62,50
9	85,42	66,67	64,58
10	85,42	66,67	58,33
Rata-Rata	88,54	64,38	64,17

L.14 Hasil Uji Coba pada Parameter Kernel SVM pada Dataset Tanpa Kuis

Fold ke-	Akurasi (%)		
	RBF	Linear	Polynomial
1	87,50	66,67	66,67
2	87,50	66,67	66,67
3	89,58	64,58	60,42
4	89,58	66,67	64,58
5	87,50	62,50	64,58
6	85,42	62,50	58,33
7	85,42	60,42	60,42
8	89,58	62,50	58,33
9	85,42	66,67	64,58
10	85,42	64,58	58,33
Rata-Rata	87,29	64,38	62,29

L.15 Hasil Uji Coba pada Parameter Kernel SVM pada Dataset Tanpa Example dan Outline

Fold ke-	Akurasi (%)		
	RBF	Linear	Polynomial
1	60,42	60,42	62,50
2	77,08	60,42	58,33
3	72,92	60,42	60,42
4	77,08	60,42	64,58
5	77,08	60,42	62,50
6	70,83	58,33	58,33
7	75,00	58,33	58,33
8	75,00	58,33	62,50
9	60,42	58,33	56,25
10	72,92	58,33	54,17
Rata-Rata	71,88	59,38	59,79

L.16 Hasil Uji Coba pada Parameter Kernel SVM pada Dataset Tanpa Kuis, Example, dan Outline

Fold ke-	Akurasi (%)		
	RBF	Linear	Polynomial
1	60,42	60,42	58,33
2	62,50	60,42	60,42
3	58,33	60,42	58,33
4	60,42	60,42	62,50
5	64,58	60,42	62,50
6	58,33	58,33	58,33
7	60,42	58,33	58,33
8	60,42	58,33	58,33
9	60,42	58,33	58,33
10	56,25	58,33	56,25
Rata-Rata	60,21	59,38	59,17

L.17 Hasil Uji Coba pada Parameter Criterion DTs pada Dataset Lengkap

Fold ke-	Akurasi (%)	
	Gini	Entropy
1	95,83	95,83
2	93,75	93,75
3	93,75	93,75
4	93,75	100,00
5	97,92	95,83
6	93,75	93,75
7	93,75	93,75
8	93,75	93,75
9	95,83	93,75
10	97,92	97,92
Rata-Rata	95,00	95,21

L.18 Hasil Uji Coba pada Parameter Criterion DTs pada Dataset Tanpa Kuis

Fold ke-	Akurasi (%)	
	Gini	Entropy
1	91,67	91,67
2	93,75	93,75
3	91,67	91,67
4	95,83	95,83
5	93,75	93,75
6	93,75	91,67
7	93,75	93,75
8	89,58	89,58
9	93,75	91,67
10	93,75	91,67
Rata-Rata	93,13	92,50

L.19 Hasil Uji Coba pada Parameter Criterion DTs pada Dataset Tanpa Example dan Outline

Fold ke-	Akurasi (%)	
	Gini	Entropy
1	77,08	77,08
2	79,17	81,25
3	72,92	75,00
4	79,17	77,08
5	72,92	72,92
6	72,92	72,92
7	77,08	79,17
8	70,83	70,83
9	77,08	79,17
10	75,00	75,00
Rata-Rata	75,42	76,04

L.20 Hasil Uji Coba pada Parameter Criterion DTs pada Dataset Tanpa Kuis, Example, dan Outline

Fold ke-	Akurasi (%)	
	Gini	Entropy
1	66,67	64,58
2	64,58	64,58
3	68,75	68,75
4	70,83	68,75
5	72,92	72,92
6	64,58	64,58
7	64,58	64,58
8	60,42	60,42
9	77,08	77,08
10	58,33	60,42
Rata-Rata	66,88	66,67

L.21 Hasil Uji Coba pada Dataset Lengkap Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	91,67	94,37	91,67	92,58
2	89,58	85,03	89,58	87,07
3	91,67	93,06	91,67	92,08
4	93,75	93,19	93,75	93,36
5	91,67	92,69	91,67	91,68
6	87,50	82,58	87,50	84,41
7	85,42	78,13	85,42	81,53
8	91,67	95,02	91,67	92,25
9	91,67	91,67	91,67	91,67
10	89,58	91,81	89,58	89,22
Rata-rata	90,42	89,75	90,42	89,58

L.22 Hasil Uji Coba pada Dataset Lengkap Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	91,67	84,15	91,67	87,71
2	89,58	84,18	89,58	86,33
3	87,50	78,61	87,50	82,77
4	89,58	86,69	89,58	85,96
5	87,50	76,94	87,50	81,78
6	87,50	77,13	87,50	81,85
7	87,50	83,94	87,50	85,23
8	93,75	90,49	93,75	91,71
9	85,42	80,42	85,42	82,59
10	85,42	78,12	85,42	81,58
Rata-rata	88,54	82,07	88,54	84,75

L.23 Hasil Uji Coba pada Dataset Lengkap Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	95,83	97,40	95,83	96,24
2	93,75	96,11	93,75	94,32
3	93,75	94,01	93,75	93,61
4	97,92	97,99	97,92	97,79
5	93,75	93,89	93,75	93,46
6	93,75	94,01	93,75	93,47
7	93,75	93,82	93,75	93,70
8	93,75	94,58	93,75	93,90
9	95,83	96,61	95,83	95,97
10	97,92	98,61	97,92	98,11
Rata-rata	95,00	95,70	95,00	95,06

L.24 Hasil Uji Coba pada Dataset Tanpa Kuis Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	85,42	89,12	85,42	86,41
2	91,67	92,85	91,67	90,94
3	89,58	89,73	89,58	89,21
4	89,59	88,68	89,59	88,19
5	91,67	92,27	91,67	91,22
6	87,50	83,39	87,50	84,65
7	87,50	84,13	87,50	85,35
8	93,75	94,91	93,75	92,68
9	91,67	91,47	91,67	91,25
10	85,42	83,30	85,42	84,25
Rata-Rata	89,38	88,99	89,38	88,41

L.25 Hasil Uji Coba pada Dataset Tanpa Kuis Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	87,50	84,42	87,50	85,69
2	87,50	80,83	87,50	83,38
3	89,58	89,47	89,58	88,87
4	89,58	82,88	89,58	85,52
5	87,50	78,08	87,50	82,15
6	85,42	77,80	85,42	81,02
7	85,42	78,34	85,42	81,71
8	89,58	81,88	89,58	85,16
9	85,42	78,58	85,42	81,74
10	85,42	82,35	85,42	83,41
Rata-Rata	87,29	81,46	87,29	83,87

L.26 Hasil Uji Coba pada Dataset Tanpa Kuis Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	91,67	92,60	91,67	91,56
2	93,75	95,37	93,75	93,16
3	91,67	92,60	91,67	90,89
4	95,83	96,88	95,83	95,44
5	93,75	94,32	93,75	92,86
6	93,75	94,27	93,75	93,52
7	93,75	94,75	93,75	92,92
8	89,58	92,21	89,58	88,87
9	93,75	94,68	93,75	92,97
10	93,75	93,23	93,75	93,33
Rata-Rata	93,13	94,09	93,13	92,55

L.27 Hasil Uji Coba pada Dataset Tanpa Aktifitas Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	83,33	88,30	83,33	84,60
2	83,33	80,89	83,33	81,71
3	87,50	86,55	87,50	85,32
4	83,33	84,33	83,33	83,17
5	89,58	86,78	89,58	87,20
6	81,25	74,10	81,25	77,11
7	81,25	75,92	81,25	78,33
8	87,50	89,17	87,50	87,94
9	83,33	82,31	83,33	82,61
10	81,25	84,38	81,25	81,60
Rata-rata	84,17	83,27	84,17	82,96

L.28 Hasil Uji Coba pada Dataset Tanpa Aktifitas Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	89,58	82,48	89,58	85,77
2	79,17	72,01	79,17	75,03
3	83,33	75,89	83,33	79,20
4	89,58	86,91	89,58	86,03
5	87,50	76,94	87,50	81,78
6	83,33	74,58	83,33	78,19
7	83,33	76,27	83,33	79,62
8	93,75	90,49	91,71	91,71
9	81,25	73,01	76,26	76,26
10	83,33	75,91	83,33	79,05
Rata-rata	85,42	78,45	85,42	81,26

L.29 Hasil Uji Coba pada Dataset Tanpa Aktifitas Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	89,58	89,64	89,58	89,12
2	83,33	82,48	83,33	82,10
3	87,50	86,99	87,50	86,79
4	91,67	88,79	91,67	89,25
5	93,75	92,59	93,75	92,95
6	87,50	89,37	87,50	86,18
7	79,17	78,88	79,17	78,92
8	91,67	92,50	91,67	91,82
9	85,42	83,30	85,42	83,65
10	85,42	87,07	85,42	85,31
Rata-rata	87,50	87,16	87,50	86,61

L.30 Hasil Uji Coba pada Dataset Tanpa Example dan Outline Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	77,08	67,07	77,08	71,16
2	79,17	80,87	79,17	73,14
3	75,00	64,51	75,00	67,50
4	77,08	66,59	77,08	70,53
5	75,00	78,03	75,00	73,84
6	70,83	54,77	70,83	61,44
7	75,00	72,34	75,00	68,00
8	75,00	66,03	75,00	68,79
9	70,83	59,44	70,83	63,66
10	72,92	61,32	72,92	66,11
Rata-Rata	74,79	67,10	74,79	68,42

L.31 Hasil Uji Coba pada Dataset Tanpa Example dan Outline Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	60,42	36,50	60,42	45,51
2	77,08	73,15	77,08	69,14
3	72,92	56,79	72,92	63,50
4	77,08	67,72	77,08	69,16
5	77,08	71,54	77,08	71,30
6	70,83	54,77	70,83	61,44
7	75,00	72,34	75,00	68,00
8	75,00	58,38	75,00	65,15
9	60,42	50,38	60,42	50,23
10	72,92	57,01	72,92	63,80
Rata-Rata	71,88	59,86	71,88	62,72

L.32 Hasil Uji Coba pada Dataset Tanpa Example dan Outline Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	79,17	69,41	79,17	73,31
2	79,17	80,79	79,17	74,35
3	72,92	61,12	72,92	65,82
4	79,17	72,01	79,17	74,21
5	72,92	72,01	72,92	74,21
6	72,92	75,57	72,92	71,21
7	77,08	76,10	77,08	74,02
8	70,83	64,82	70,83	66,22
9	77,08	78,07	77,08	75,34
10	72,92	59,83	72,92	65,25
Rata-Rata	75,42	70,08	75,42	70,45

L.33 Hasil Uji Coba pada Dataset Tanpa Konten Grafis Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	89,58	91,32	89,58	89,93
2	91,67	88,29	91,67	88,75
3	91,67	93,06	91,67	92,08
4	91,67	89,99	91,67	90,42
5	91,67	91,37	91,67	90,99
6	87,50	82,19	87,50	84,31
7	85,42	78,13	85,42	81,53
8	89,58	94,58	89,58	90,64
9	91,67	87,92	91,67	89,73
10	89,58	90,77	89,58	89,56
Rata-rata	90,00	88,76	90,00	89,56

L.34 Hasil Uji Coba pada Dataset Tanpa Konten Grafis Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	91,67	84,15	91,67	87,71
2	87,50	82,17	87,50	84,13
3	89,58	84,34	89,58	86,38
4	89,58	86,69	89,58	85,96
5	83,33	73,90	83,33	77,51
6	87,50	76,79	87,50	81,74
7	85,42	81,94	85,42	83,02
8	95,83	92,50	95,83	93,98
9	81,25	75,64	81,25	77,06
10	85,42	78,62	85,42	81,65
Rata-rata	87,71	81,68	87,71	83,91

L.35 Hasil Uji Coba pada Dataset Tanpa Konten Grafis Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	93,75	94,79	93,75	94,03
2	95,83	97,05	97,05	96,09
3	93,75	94,01	94,01	93,61
4	93,75	94,19	94,19	93,19
5	95,83	96,10	96,10	95,41
6	91,67	91,38	91,38	91,00
7	93,75	93,82	93,75	93,70
8	91,67	93,63	91,67	91,71
9	97,92	97,98	97,92	97,73
10	95,83	96,53	95,83	95,83
Rata-rata	94,38	94,95	94,38	94,23

L.36 Hasil Uji Coba pada Dataset Tanpa Konten Audio Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	91,67	94,37	91,67	92,58
2	89,58	85,03	89,58	87,07
3	89,58	90,90	89,58	90,15
4	93,75	93,19	93,75	93,36
5	91,67	92,69	91,67	91,68
6	87,50	82,58	87,50	84,41
7	85,42	78,13	85,42	81,53
8	91,67	95,02	91,67	92,25
9	91,67	91,67	91,67	91,67
10	87,50	88,54	87,50	87,50
Rata-rata	90,00	89,21	90,00	89,22

L.37 Hasil Uji Coba pada Dataset Tanpa Konten Audio Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	91,67	84,15	91,67	87,71
2	89,58	84,18	89,58	86,33
3	89,58	84,34	89,58	86,38
4	89,58	86,69	89,58	85,96
5	87,50	76,94	87,50	81,78
6	87,50	77,13	87,50	81,84
7	87,50	83,94	87,50	85,23
8	95,83	92,50	95,83	93,98
9	85,42	80,42	85,42	82,59
10	83,33	77,89	83,33	80,47
Rata-rata	88,75	82,82	88,75	85,23

L.38 Hasil Uji Coba pada Dataset Tanpa Konten Audio Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	95,83	97,40	95,83	96,24
2	93,75	96,11	93,75	94,32
3	93,75	94,01	93,75	93,61
4	97,92	97,99	97,92	97,79
5	95,83	96,25	95,83	95,60
6	93,75	94,01	93,75	93,47
7	93,75	93,82	93,75	93,70
8	93,75	94,58	93,75	93,90
9	93,75	95,42	93,75	94,19
10	97,92	98,61	97,92	97,92
Rata-rata	95,00	95,82	95,00	95,09

L.39 Hasil Uji Coba pada Dataset Tanpa Kuis, Example, dan Outline Menggunakan k-NN

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	66,67	67,57	66,67	63,90
2	62,50	58,09	62,50	57,57
3	60,42	59,17	60,42	57,74
4	64,58	61,82	64,58	60,56
5	64,58	55,11	64,58	55,37
6	64,58	53,65	64,58	56,98
7	60,42	49,34	60,42	47,20
8	60,42	56,60	60,42	55,45
9	68,75	60,88	68,75	62,75
10	58,33	54,07	58,33	53,75
Rata-Rata	63,13	57,63	63,13	57,13

L.40 Hasil Uji Coba pada Dataset Tanpa Kuis, Example, dan Outline Menggunakan SVM

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	60,42	36,50	60,42	45,51
2	62,50	51,87	62,50	49,75
3	58,33	35,99	58,33	44,52
4	60,42	36,50	60,42	45,51
5	64,58	50,59	64,58	52,97
6	58,33	34,75	58,33	43,56
7	60,42	49,34	60,42	47,20
8	60,42	56,60	60,42	55,45
9	60,42	45,52	60,42	49,58
10	56,25	34,24	56,25	42,57
Rata-Rata	60,21	43,19	60,20	47,66

L.41 Hasil Uji Coba pada Dataset Tanpa Kuis, Example, dan Outline Menggunakan DTs

Fold ke-	Akurasi (%)	Precision (%)	Recall (%)	F1-Score (%)
1	68,75	68,02	68,75	65,40
2	64,58	72,89	64,58	61,47
3	68,75	57,77	68,75	61,67
4	70,83	69,65	70,83	67,35
5	72,92	79,30	72,92	70,54
6	64,58	61,47	64,58	59,46
7	64,58	64,97	64,58	60,86
8	60,42	56,60	60,42	55,45
9	77,08	71,02	77,08	72,37
10	58,33	58,92	58,33	55,78
Rata-Rata	67,08	66,06	67,08	63,04

L.42 Hasil Uji Coba Over-sampling

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
1	91,67	89,58	95,83
2	85,42	81,25	93,75
3	85,42	93,75	93,75
4	95,83	95,83	97,92
5	93,75	87,50	95,83
6	93,75	89,58	97,92
7	87,50	85,42	93,75
8	87,50	85,42	93,75
9	93,75	89,58	95,83
10	89,58	93,75	97,92
Rata-Rata	90,42	89,17	95,63

L.43 Hasil Uji Coba Tanpa Over-sampling

Fold ke-	Akurasi (%)		
	k-NN	SVM	DTs
1	91,67	91,67	95,83
2	89,58	89,58	93,75
3	91,67	87,50	93,75
4	93,75	89,58	97,92
5	91,67	87,50	93,75
6	87,50	87,50	95,83
7	85,42	87,50	93,75
8	91,67	93,75	93,75
9	91,67	85,42	93,75
10	89,58	85,42	97,92
Rata-Rata	90,42	88,54	95,00

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Safira Vanillia Putri, lahir di Semarang pada tanggal 16 Juli 1998. Penulis menempuh pendidikan mulai dari TK PIKPG (2002-2004), SDN 2 Sidokumpul Gresik (2004-2010), SMP Negeri 1 Gresik (2010-2013), SMA Negeri 1 Gresik (2013-2016), dan saat ini sedang menjalani pendidikan S1 Teknik Informatika di Institut Teknologi Sepuluh Nopember. Penulis aktif dalam organisasi dan kepanitiaan Himpunan Mahasiswa Teknik Computer (HMTC), Badan Eksekutif Mahasiswa Fakultas Teknologi Informasi dan Komunikasi (BEM FTIK), dan Schematics diantaranya menjadi staf Departemen Kesejahteraan Mahasiswa HMTC ITS 2017-2018, staf Departemen Media Informasi BEM FTIK 2017-2018, staf ahli Departemen Kesejahteraan Mahasiswa HMTC ITS 2018-2019, staf Biro Web dan Kesekretariatan Schematics ITS 2017, dan staf ahli Biro Web dan Kesekretariatan Schematics ITS 2018. Komunikasi dengan penulis dapat melalui telepon: +6281934888688 dan *e-mail*: **safiravanillia@gmail.com**.