IMPLEMENTASI ALGORITMA *SUPPORT VECTOR MACHINE* PADA MODEL KLASIFIKASI SEKUENS DNA MANUSIA
Studi Kasus: Penderita Diabetes Mellitus

SKRIPSI

OLEH:
M. Wildan Nuril Akmal
NIM. 210601110096

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

IMPLEMENTASI ALGORITMA *SUPPORT VECTOR MACHINE* PADA MODEL KLASIFIKASI SEKUENS DNA MANUSIA
Studi Kasus: Penderita Diabetes Mellitus

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
M. Wildan Nuril Akmal
NIM. 210601110096

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

IMPLEMENTASI ALGORITMA *SUPPORT VECTOR MACHINE* PADA MODEL KLASIFIKASI SEKUENS DNA MANUSIA
Studi Kasus: Penderita Diabetes Mellitus

SKRIPSI

Oleh
M. Wildan Nuril Akmal
NIM. 210601110096

Telah Disetujui Untuk Diuji

Malang, 21 November 2024

Dosen Pembimbing I	Dosen Pembimbing II
Ari Kusumastuti, M.Si., M. Pd.	Mohammad Nafie Jauhari, M.Si.
NIP. 19770521 200501 2 004	NIPPPK. 19870218 202321 1 018

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005

IMPLEMENTASI ALGORITMA *SUPPORT VECTOR MACHINE* PADA MODEL KLASIFIKASI SEKUENS DNA MANUSIA
Studi Kasus: Penderita Diabetes Mellitus

SKRIPSI

Oleh
M. Wildan Nuril Akmal
NIM. 210601110096

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)
Tanggal 20 Desember 2024

Ketua Penguji	:	Dr. Usman Pagalay, M.Si.	.....
Anggota Penguji 1	:	Mohammad Jamhuri, M.Si.	.....
Anggota Penguji 2	:	Ari Kusumastuti, M.Pd., M.Si.	.....
Anggota Penguji 3	:	Mohammad Nafie Jauhari, M.Si.	.....

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	:	M. Wildan Nuril Akmal
NIM	:	210601110096
Program Studi	:	Matematika
Fakultas	:	Sains dan Teknologi
Judul Skripsi	:	Implementasi Algoritma *Support Vector Machine* Pada Model Klasifikasi Sekuens DNA Manusia Studi Kasus: Penderita Diabetes Mellitus

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan dan pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila di kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 20 Desember 2024

Yang membuat pernyataan,

M. Wildan Nuril Akmal

NIM. 210601110096

MOTO

"*Jangan pernah merendahkan orang lain, karena setiap manusia memiliki nilai dan martabat yang sama di hadapan Allah*"
(*Gus Baha*)

"*ini akan berlalu*"
(*Dr. Fahruddin Faiz*)

"*Jangan mati sebelum kau mati*"
(*Bleach*)

PERSEMBAHAN

Dengan penuh rasa syukur kepada Allah SWT, skripsi ini saya persembahkan kepada:

Pertama Bapak dan ibuk tercinta Afif Luthfi dan Mutimmatul Aliyah yang telah menjadi sumber inspirasi, kekuatan, dan doa tiada henti. Terima kasih atas cinta, pengorbanan, dan dukungan tanpa batas yang telah kalian berikan selama ini. Segala usaha dan perjuangan ini adalah untuk membalas sebagian kecil dari kasih sayang yang tidak ternilai dari kalian.

Kedua Adik Muhammad Adly Habib Akmal Semoga karya ini dapat menjadi pengingat bahwa mimpi dapat diraih dengan usaha dan doa. Jangan pernah berhenti bermimpi, karena setiap langkah kecil yang kau ambil akan membawamu lebih dekat ke cita-cita. Teruslah berjuang dan yakinlah bahwa masa depan yang indah menantimu.

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa, atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Implementasi Algoritma *Support Vector Machine* Pada Model Klasifikasi Sekuens DNA Manusia Studi Kasus: Penderita Diabetes Mellitus". Penulisan skripsi ini dimaksudkan untuk memenuhi salah satu syarat guna memperoleh gelar Sarjana di Fakultas Sains dan Teknologi Jurusan Matematika Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Penyusunan skripsi ini tentu tidak lepas dari dukungan, bimbingan, dan bantuan berbagai pihak. Oleh karena itu, pada kesempatan ini, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Penulis ingin mengucapkan rasa terima kasih yang mendalam kepada Bapak dan Ibu tercinta. Afif Luthfi S. Ag dan Mutimmatul Aliyah S. Ag, terima kasih atas segala doa, dukungan, dan kasih sayang yang tiada henti, yang selalu menjadi kekuatan dan motivasi terbesar saya dalam menuntut ilmu dan menghadapi berbagai tantangan.
2. Penulis ingin menyampaikan rasa terima kasih yang mendalam kepada adik tercinta. Muhammad Adly Habib Akmal, terima kasih atas semangat yang selalu terpancar dalam setiap langkahmu untuk menggapai cita-cita.
3. Prof. Dr. H. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim.
4. Prof. Dr. Hj. Sri Harini, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim.
5. Dr. Elly Susanti, S.Pd., M.Sc., selaku ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim.
6. Ari Kusumastuti, M.Si., M. Pd., selaku dosen pembimbing I yang telah memberikan berbagai pengetahuan, pengalaman, arahan, nasihat, serta motivasi kepada penulis.
7. Mohammad Nafie Jauhari, M.Si., selaku dosen pembimbing II yang telah memberikan ilmu, nasihat, bimbingan, pengalaman, serta motivasi kepada penulis.
8. Dr. Usman Pagalay, M.Si., selaku ketua penguji dalam ujian skripsi yang

telah memberikan saran yang bermanfaat bagi penulis.

9. Mohammad Jamhuri, M.Si., selaku dosen penguji 1 dalam ujian skripsi yang telah memberikan saran yang bermanfaat bagi penulis.
10. Seluruh dosen Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim.
11. Di balik setiap langkah yang penulis ambil, ada satu "*nada sempurna"* dalam melodi hidup penulis, membuat semuanya terasa lengkap. Dukunganmu yang tak pernah surut dan keyakinanmu kepada Penulis telah menjadi alasan bertahan dan berjuang.
12. Seluruh teman-teman yang senantiasa memberikan bantuan, dukungan dan semangat kepada penulis dalam berbagai kondisi.

Akhir kata, penulis mengucapkan terima kasih kepada semua pihak yang tidak dapat disebutkan satu per satu, namun telah memberikan dukungan dan bantuan baik secara langsung maupun tidak langsung.

Malang, 20 Desember 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR SIMBOL

$\boldsymbol{x}_i$	: *Dataset*
y_i	: Label kelas
$\boldsymbol{w}$	: Vektor pembobot
b	: Bias
α_i	: Nilai bobot pada data
t_i	: Variabel *slack*
$K(\boldsymbol{x}_i.\boldsymbol{x}_j)$	: Kernel *trick*
$\emptyset$	: Parameter fungsi phi
h	: Parameter fungsi h
κ	: Parameter fungsi κ
δ	: Parameter fungsi $delta$
σ	: Parameter fungsi $sigma$
C	: Parameter fungsi $cost$
$sign$	: Fungsi tanda atau fungsi *signum*

DAFTAR LAMPIRAN

ABSTRAK

Akmal, M. Wildan Nuril. 2024**. Implementasi Algoritma *Support Vector Machine* Pada Model Klasifikasi Sekuens DNA Manusia Studi Kasus: Penderita Diabetes Mellitus**. Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Ari Kusumastuti, M.Si., M. Pd. (II) Mohammad Nafie Jauhari, M.Si.

Kata Kunci: SVM, DNA, Diabetes Mellitus, Klasifikasi, Kernel RBF.

Diabetes Mellitus adalah penyakit kronis yang kompleks dan memberikan dampak signifikan pada kesehatan masyarakat. Penelitian ini bertujuan untuk menerapkan algoritma Support Vector Machine (SVM) dalam klasifikasi sekuens DNA manusia untuk mengidentifikasi hubungan antara variasi genetik dengan diabetes mellitus. Tahapan preprocessing data meliputi pembersihan data untuk menghilangkan komponen yang tidak relevan, ekstraksi fitur melalui representasi k-mers, tokenisasi untuk mengkonversi sekuens DNA menjadi format numerik, serta oversampling untuk mengatasi ketidakseimbangan kelas pada dataset. Penelitian ini menggunakan SVM dengan kernel non-linear Radial Basis Function (RBF), di mana model yang dikembangkan berhasil mencapai akurasi maksimum sebesar 94% dalam membedakan tipe diabetes, yaitu Diabetes Mellitus Tipe 1, Tipe 2, dan non-diabetes. Studi ini memberikan kontribusi penting dalam penerapan pembelajaran mesin untuk analisis data genomik, serta mendukung inovasi dalam deteksi dini dan pengobatan diabetes yang terpersonalisasi.

ABSTRACT

Akmal, M. Wildan Nuril. 2024. **Implementation of Support Vector Machine Algorithm on Human DNA Sequence Classification Model Case Study: Diabetes Mellitus Patients.** Thesis. Department of Mathematics, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisors: (I) Ari Kusumastuti, M.Si., M.Pd. (II) Mohammad Nafie Jauhari, M.Si.

Keywords: SVM, DNA, Diabetes Mellitus, Classification, RBF Kernel.

Diabetes mellitus is a complex chronic disease that significantly impacts public health. This study aims to apply the Support Vector Machine (SVM) algorithm to classify human DNA sequences and identify the relationship between genetic variations and diabetes mellitus. The data preprocessing stages include data cleaning to remove irrelevant components, feature extraction using k-mer representation, tokenization to convert DNA sequences into numerical format, and oversampling to address class imbalance in the dataset. The study employed SVM with a non-linear Radial Basis Function (RBF) kernel, achieving a maximum accuracy of 94% in differentiating between diabetes types: Type 1, Type 2, and non-diabetes. This research contributes significantly to the application of machine learning in genomic data analysis and supports innovation in the early detection and personalized treatment of diabetes.

مستخلص البحث

أكمل، م. ويلدان نوريل٢٠٢٤ تنفيذ خوارزمية آلة ناقل الدعم على نموذج تصنيف تسلسل الحمض النووي . البشري دراسة حالة نموذجية: مرضى السكري. أطروحة. قسم الرياضيات، كلية العلوم :والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية في مالانج. المشرف(١) آري .كوسوماستوتي، ماجستير في العلوم السياسية، ماجستير في الطب(٢) محمد نافع جوهري .ماجستير في العلوم السياسية

الكلمات المفتاحية: آلة دعم المتجهات المساندة، الحمض النووي، داء السكري، التصنيف، الدالة الأساسية .الشعاعية النواة

داء السكري هو مرض مزمن معقد له تأثير كبير على الصحة العامة. تهدف هذه الدراسة إلى تطبيق خوارزمية آلة دعم المتجهات (SVM) في تصنيف تسلسلات الحمض النووي البشري لتحديد العلاقة بين التباين الجيني وداء السكري. وتتضمن مراحل المعالجة المسبقة للبيانات تنظيف البيانات لإزالة المكونات غير ذات الصلة، واستخراج السمات من خلال تمثيل k-mers، والترميز لتحويل تسلسل الحمض النووي إلى صيغة رقمية، وأخذ عينات زائدة لمعالجة عدم توازن الفئات في مجموعة البيانات. استخدمت هذه الدراسة نموذج SVM مع نواة الدالة القاعدية الشعاعية غير الخطية (RBF)، حيث حقق النموذج المطور دقة قصوى بلغت ٩٤٪ في التفريق بين أنواع مرض السكري، وهي النوع الأول والنوع الثاني وغير السكري. تقدم هذه الدراسة مساهمة مهمة في تطبيق التعلم الآلي لتحليل البيانات الجينومية، وتدعم الابتكار في الكشف المبكر والعلاج الشخصي لمرض السكري.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan *machine learning* telah mengalami kemajuan yang signifikan dan menjadi salah satu teknologi terdepan di era digital saat ini. *Machine learning* sebagai cabang dari kecerdasan buatan yang memiliki berberapa fungsi dalam impelementasi algoritma, seperti sistem komputer yang dapat mengenali pola dan membuat keputusan atau prediksi tanpa memerlukan pemrograman eksplisit dari data yang tersedia (El Naqa & Murphy, 2015). Kemajuan *machine learning* telah membawa dampak signifikan di berbagai bidang, termasuk sektor kesehatan. Salah satunya yaitu untuk menganalisis data medis dalam skala besar dan mengidentifikasi pola-pola yang rumit dalam genom manusia (Saptadi dkk., 2024). *Machine learning* memiliki kemampuan untuk menangani dan menganalisis data dalam jumlah besar dengan cepat dan akurat. Selain itu, menjadikannya alat yang sangat berharga untuk penelitian dan aplikasi praktis, terutama dalam mengatasi tantangan-tantangan kompleks yang ada dalam ilmu pengetahuan modern. Oleh karena itu, analisa yang lebih mendalam mengenai genom manusia dapat dilakukan untuk mengidentifikasi beberapa penyakit salah satunya penyakit diabetes mellitus.

Diabetes mellitus merupakan salah satu penyakit kronis yang prevalensinya terus meningkat di seluruh dunia (Sari & Purnama, 2019). Diabetes mellitus terbagi menjadi beberapa kategori, yaitu diabetes mellitus tipe 1 dan tipe 2 sebagai bentuk yang paling umum (Sari & Purnama, 2019). Diabetes tipe 1 biasanya terjadi pada anak-anak dan remaja yang disebabkan oleh kerusakan autoimun pada sel-sel beta

pankreas yang memproduksi insulin. Sementara itu, Diabetes mellitus tipe 2 lebih sering dijumpai pada orang dewasa yang sering dikaitkan dengan faktor risiko genetik dan gaya hidup, termasuk obesitas dan aktivitas fisik yang rendah. Penyakit ini ditandai oleh hiperglikemia kronis, yaitu peningkatan kadar glukosa dalam darah yang persisten. Hal tersebut disebabkan oleh gangguan pada sekresi insulin, aksi insulin, atau keduanya (Nasution dkk., 2021). Insulin adalah hormon yang berperan penting dalam mengatur gula darah dalam tubuh dan produksinya dikendalikan oleh gen spesifik di DNA yang mengkodekan sekuens DNA (Lathifah, 2017). Maka dari dapat mengidentifikasi sekuens DNA untuk menghasilkan diagnosa penyakit diabetes yang lebih akurat.

Sekuens DNA mengacu pada urutan linear dari nukleotida dalam DNA yang menentukan kode genetik suatu organisme (Agus, 2018). Setiap nukleotida terdiri dari salah satu dari empat basa nitrogen: adenin (A), sitosin (C), guanin (G), dan timin (T) dan menjadi dasar dari semua informasi genetik dalam organisme hidup (Akkaya & Kalkan, 2021). Dalam konteks genom manusia, sekuens DNA merupakan instruksi biologis yang mengatur perkembangan, fungsi, dan reproduksi sel-sel dalam tubuh. Setiap organisme memiliki sekuens DNA yang unik, dan pada manusia, variasi dalam urutan ini dapat mempengaruhi berbagai sifat biologis, termasuk kerentanan terhadap penyakit (Widyanto dkk., 2021). Secara keseluruhan, sekuens DNA manusia terdiri dari sekitar 3 miliar pasangan basa yang tersusun dalam 23 pasang kromosom (Muhammad dkk., 2021). Analisis terhadap sekuens DNA memungkinkan identifikasi mutasi atau polimorfisme nukleotida tunggal (SNPs) yang dapat berperan penting dalam predisposisi seseorang terhadap kondisi kesehatan tertentu, seperti diabetes mellitus. Analisa tersebut dapat dilakukan

dengan berbagai metode seiring berkembangnya teknologi, salah satunya yaitu menggunakan model klasifikasi *Support Vector Machine* (SVM).

Klasifikasi adalah teknik analisis data yang berfokus pada pengelompokan entitas ke dalam kategori yang telah ditentukan berdasarkan fitur-fitur spesifik (Putra dkk., 2023). Dalam genomik, klasifikasi digunakan untuk mengelompokkan data genetik, seperti sekuens DNA, ke dalam kategori yang relevan. Misalnya, individu dengan risiko tinggi atau rendah terhadap suatu penyakit berdasarkan pola-pola yang teridentifikasi dalam data tersebut. Teknik tersebut melibatkan algoritma statistik atau metode pembelajaran mesin untuk membangun model prediktif yang dapat menganalisis data genetik dan memberikan keputusan atau prediksi yang akurat. Terdapat beberapa algoritma yang dapat digunakan dalam pengklasifikasikan, salah satunya *Support Vector Machine* (SVM).

Support Vector Machine (SVM) merupakan salah satu algoritma pembelajaran mesin yang efektif untuk pengklasifikasian, terutama dalam data genomik. Prinsip dasar SVM adalah menemukan hyperplane optimal yang memisahkan data ke dalam kelas yang berbeda dengan margin maksimum (M. R. Adrian dkk., 2021). Margin ini mengacu pada jarak antara hyperplane dan titik data terdekat dari setiap kelas, yang dikenal sebagai support vectors. Dengan memaksimalkan margin, SVM berupaya menghasilkan model klasifikasi yang lebih robust dan mampu melakukan generalisasi dengan baik.

Dalam konteks analisis genetik, SVM digunakan untuk mengklasifikasikan sekuens DNA berdasarkan fitur genetik yang relevan, seperti pola ekspresi gen atau mutasi. Analisis genetik ini melibatkan suatu proses dan interpretasi sekuens DNA untuk mengidentifikasi variasi genetik yang berhubungan dengan kondisi kesehatan

tertentu. Kemampuan SVM dalam menangani data dengan dimensi tinggi dan kompleksitas yang besar membuatnya sangat berguna untuk mengidentifikasi pola genetik yang berhubungan dengan penyakit seperti diabetes mellitus (Ghaddar & Naoum-Sawaya, 2018).

Penelitian yang menggunakan algoritma SVM telah dilakukan untuk mengidentifikasi berbagai penyakit. Salah satunya penelitian oleh Al Kindhi dkk. (2018) yang memperoleh hasil bahwa metode SVM mampu menganalisa adanya mutasi HCV pada isolated DNA dengan menghasilkan akurasi sebesar 99,8%. Selain itu, juga terdapat penelitian oleh Hamed dkk. (2023) untuk klasifikasi sekuens DNA yang menggabungkan pembelajaran mesin dan algoritma pencocokan pola, bertujuan meningkatkan akurasi dan efisiensi klasifikasi. Model diuji menggunakan berbagai algoritma, di mana SVM mencapai akurasi dan skor F1 tertinggi, yaitu 96,3% dan 97%. Studi ini menyoroti pengaruh panjang pola terhadap akurasi dan kompleksitas waktu eksekusi, dengan SVM menunjukkan performa terbaik di berbagai skenario. Oleh karena itu, algoritma SVM dapat memiliki potensi besar dalam mendukung analisis dan pengambilan keputusan, termasuk dalam upaya identifikasi diagnosa penyakit diabetes mellitus melalui klasifikasi sekuens DNA.

Meskipun algoritma SVM telah banyak diterapkan dalam berbagai penelitian klasifikasi sekuens DNA, minimnya studi yang secara khusus mengaplikasikan SVM pada dataset penderita diabetes mellitus menjadi celah yang perlu dieksplorasi lebih lanjut. Selain itu, belum banyak penelitian yang melakukan perbandingan menyeluruh antara metode *k-mers* dan berbagai model kernel SVM. Padahal, perbandingan ini penting agar penelitian di masa mendatang dapat

langsung memilih kombinasi *k-mers* dan model kernel yang paling tepat untuk kasus serupa. Dengan demikian, studi lebih lanjut diperlukan untuk mengevaluasi performa SVM pada dataset DNA penderita diabetes, serta mengidentifikasi kombinasi k-mers dan kernel optimal untuk meningkatkan akurasi dan efisiensi proses klasifikasi.

Dalam ajaran Islam, menjaga kesehatan dianggap sebagai kewajiban penting. Al-Qur'an dan Hadis menekankan pentingnya tindakan preventif dan perawatan untuk melindungi tubuh sebagai bentuk tanggung jawab terhadap amanah dari Allah, sebagaimana ajaran Al-Qur'an dan Hadis. Dalam Al-Qur'an, terdapat ayat yang menekankan pentingnya menjaga diri dari bahaya dan menjaga kesehatan, seperti yang dinyatakan dalam QS. Al-Baqarah ayat 195 (Kemenag, 2024a).

وَاَنْفِقُوْا فِيْ سَبِيْلِ اللّٰهِ وَلَا تُلْقُوْا بِاَيْدِيْكُمْ اِلَى التَّهْلُكَةِۛ وَاَحْسِنُوْاۛ اِنَّ اللّٰهَ يُحِبُّ الْمُحْسِنِيْنَ

"Dan janganlah kamu menjatuhkan dirimu sendiri ke dalam kebinasaan."

Ayat ini menekankan pentingnya upaya pencegahan dan perawatan untuk melindungi tubuh dari penyakit. Umat manusia dianjurkan untuk melakukan tindakan pencegahan sebelum penyakit muncul. Hal tersebut meliputi menjaga kebersihan, menerapkan pola makan sehat, dan menjalani gaya hidup yang seimbang. Pencegahan ini selaras dengan prinsip dalam dunia medis yang menyatakan bahwa mencegah lebih baik daripada mengobati. Dalam perspektif agama, upaya pencegahan penyakit dilihat sebagai langkah proaktif untuk menjaga karunia tubuh yang diberikan oleh Tuhan. Selain itu, dalam Hadis Nabi Muhammad SAW, terdapat sabda yang mengarahkan umat untuk mencari pengobatan dan melakukan usaha preventif, sebagaimana dikatakan, *"Sesungguhnya Allah tidak menurunkan penyakit kecuali Dia juga menurunkan penawarnya"* (HR. Bukhari

5678). Konsep ini mendukung upaya penelitian dalam bidang kesehatan, termasuk penggunaan teknologi modern seperti klasifikasi genetik untuk mendiagnosis dan mencegah penyakit. Dengan memanfaatkan teknologi terkini, seperti algoritma klasifikasi genetik, umat Islam dapat berupaya untuk melindungi diri mereka dari penyakit dan meningkatkan kualitas hidup mereka, sesuai dengan prinsip-prinsip ajaran agama yang mendorong upaya maksimal dalam menjaga kesehatan.

Urgensi dalam identifikasi sekuens DNA dalam penyakit diabetes mellitus dapat menjadi salah satu upaya penanganan penyakit diabetes mellitus. Dengan demikian, penelitian tentang "Implementasi Algoritma *Support Vector Machine* Pada Klasifikasi Sekuens DNA Manusia (Studi Kasus: Penderita Diabetes Mellitus)" diharapkan dapat memberikan kontribusi yang signifikan dalam upaya penanganan penyakit diabetes mellitus di Indonesia.

1.2 Rumusan Masalah

Berdasarkan latar belakang, dapat disimpulkan beberapa rumusan masalah antara lain:

1. Bagaimana metode dan tahapan yang digunakan dalam persiapan dan pengolahan data sekuens DNA pada penderita diabetes mellitus?
2. Bagaimana hasil implementasi dari algoritma *Support Vector Machine* (SVM) dalam klasifikasi sekuens DNA pada penderita diabetes mellitus?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah, dapat disimpulkan tujuan penelitian yaitu:

1. Untuk mengidentifikasi metode dan tahapan yang digunakan dalam persiapan

dan pengolahan data sekuens DNA pada penderita diabetes mellitus.

2. Untuk menganalisis hasil implementasi dari algoritma *Support Vector Machine* (SVM) dalam klasifikasi sekuens DNA pada penderita diabetes mellitus.

1.4 Manfaat Penelitian

Manfaat penelitian pada penelitian ini antara lain:

1. Mengetahui metode dan tahapan yang digunakan dalam persiapan dan pengolahan data sekuens DNA manusia.
2. Mengetahui hasil implementasi dari algoritma SVM dalam klasifikasi sekuens DNA pada penderita diabetes mellitus.

1.5 Batasan Masalah

Batasan masalah pada penelitian ini antara lain:

1. Data yang diunduh yakni data sekuens DNA manusia yang berhubungan dengan diabetes melitus tipe 1, tipe 2, dan data non-diabetes melitus yang didapatkan dari *website* ncbi.nlm.nih.gov.
2. Nilai k-mers yang digunakan dalam proses membagi sekuens DNA menjadi substring adalah 3-mers, 4-mers, dan 5-mers.
3. Perancangan model klasifikasi menggunakan algoritma *machine learning Support Vector Machine* (SVM) non-linier.
4. Proses evaluasi model klasifikasi akan menggunakan *confusion matrix* dengan metrik akurasi, presisi, *recall* dan *F1-Score.*

1.6 Definisi Istilah

Dalam penelitian ini, beberapa istilah yang digunakan antara lain:

Machine Learning	:	Cabang ilmu komputer yang fokus pada pengembangan sistem yang dapat belajar dan meningkatkan performanya berdasarkan data.
Support Vector	:	Titik data yang terletak paling dekat dengan *hyperplane* atau bidang pemisah antara kelas-kelas data.
Hyperplane	:	Sebuah bidang multidimensi yang memisahkan data berdasarkan kelas atau kategori.
Margin	:	Jarak terdekat antara *hyperplane* dan titik data dari masing-masing kelas.
Kernel	:	Fungsi yang digunakan untuk mengubah data ke ruang dimensi yang lebih tinggi agar dapat dipisahkan dengan lebih baik oleh model.
Data *Training*	:	Kumpulan data yang digunakan untuk melatih model pembelajaran mesin dalam proses klasifikasi.
Data *Testing*	:	Kumpulan data yang digunakan untuk menguji performa model yang telah dilatih sebelumnya.

K-Mers	:	Kombinasi dari pasangan nukleotida yang membentuk sekuens DNA manusia.
Tokenizer	:	Proses yang digunakan untuk memecah teks atau sekuens data menjadi elemen-elemen kecil seperti kata, frasa, atau token lain yang lebih mudah diproses oleh model.
Oversampling	:	Teknik untuk menyeimbangkan *dataset* yang tidak seimbang dengan meningkatkan jumlah data dari kelas minoritas agar performa model lebih baik.
Encode	:	Proses mengubah data yang lebih mudah diproses oleh mesin, misalnya mengonversi data kategorikal menjadi bentuk numerik melalui metode seperti Label *encoding*.

BAB II

KAJIAN TEORI

2.1 Teori Pendukung

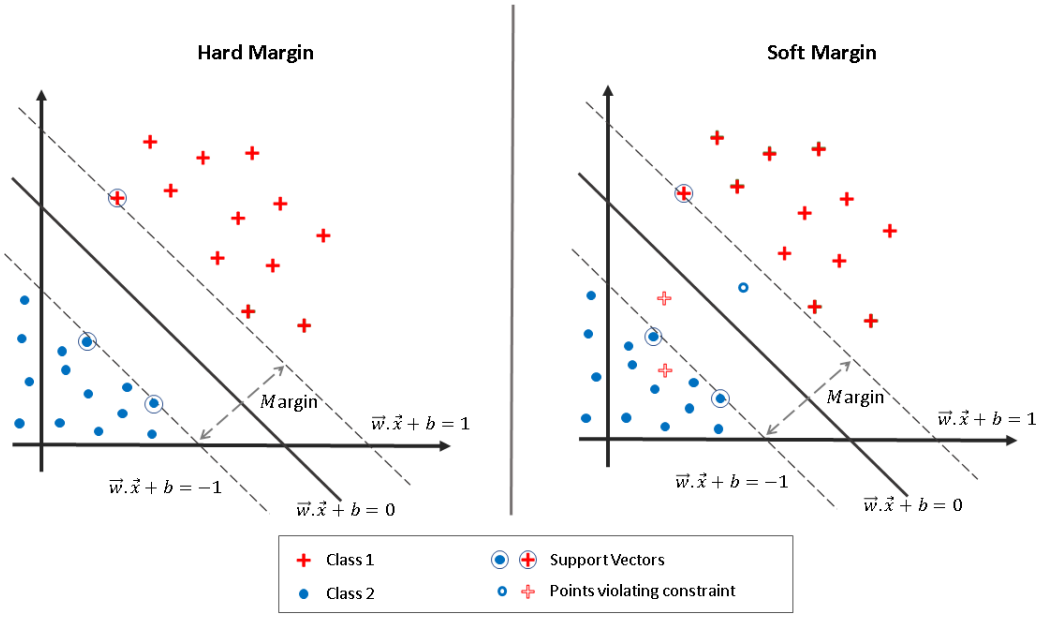
2.1.1 *Support Vector Machine* (SVM)

SVM merupakan metode klasifikasi berbasis machine learning yang digunakan dalam berbagai bidang, termasuk kesehatan (Cortes dkk., 1995). Dalam prosesnya, SVM menciptakan sebuah hyperplane optimal atau garis pemisah yang memaksimalkan margin antara kelas-kelas data yang berbeda. Tujuannya yaitu untuk menciptakan sebuah garis pemisah yang memiliki jarak maksimum dari titik-titik data dari kedua kelas. SVM memiliki beberapa kelebihan, yaitu kemampuannya untuk menangani data yang memiliki dimensi tinggi, seperti data kesehatan yang seringkali kompleks dan multidimensional (Zhang dkk., 2016). Selain itu, SVM juga memiliki toleransi yang baik terhadap data yang tidak terdistribusi secara linear, memungkinkan pengklasifikasiannya dapat bekerja dengan baik bahkan pada data yang kompleks. SVM juga sangat efektif dalam kasus jumlah dimensi yang lebih banyak daripada jumlah sampelnya dan lebih efisien karena menggunakan subset titik pelatihan (M. R. Adrian dkk., 2021).

2.1.2.1 SVM Linier

SVM linier adalah jenis algoritma SVM yang digunakan ketika data dapat dipisahkan secara linier (Han dkk., 2012). Hal tersebut dikarenakan terdapat *hyperplane* (bidang pemisah) yang dapat membagi dua kelas secara

jelas dalam ruang dimensi data. Gambar 2.1 menunjukkan, hyperplane memisahkan titik data secara linear menjadi dua komponen. *Class* 1 di definsikan sebagai kelas positif dinotatiskan dengan 1 dan *Class* 2 didefinisikan sebagai kelas negatif dinotasikan dengan -1.



Gambar 2.1 *Hyperplane* Dengan *Hard Margin* Dan *Soft Margin* (Nimisha., 2023)

Sebuah *hyperplane* yang memisahkan dua kelas dapat ditulis sebagai berikut:

$$f(\boldsymbol{x}_i) = \boldsymbol{w}^T \boldsymbol{x}_i + b = 0 \tag{2.1}$$

Persamaan (2.1) merupakan bentuk umum dari persamaan garis yang digunakan ketika bekerja dalam dimensi lebih tinggi (lebih dari 2 dimensi).

di mana:

$\boldsymbol{w}$: Vektor bobot (Vektor Kolom)

$\boldsymbol{x}_i$: Titik data dalam ruang fitur (Vector Kolom)

b : Bias (Konstanta)

Pada persamaan (2.1) $\boldsymbol{w}^T\boldsymbol{x}_i$ merupakan perkalian antara vektor baris di kali vektor kolom dan $\boldsymbol{w}$ adalah nilai vektor bobot yang didefinisikan $\boldsymbol{w} = \begin{bmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \vdots \\ \boldsymbol{w}_m \end{bmatrix}$ kemudian setelah di transpose menjadi $\boldsymbol{w}^T = [\boldsymbol{w}_1.\boldsymbol{w}_2.\ldots.\boldsymbol{w}_m]$ dan $\boldsymbol{x}_i = \begin{bmatrix} \boldsymbol{x}_{i1} \\ \boldsymbol{x}_{i2} \\ \vdots \\ \boldsymbol{x}_{im} \end{bmatrix}$.

Dengan m menunjukkan banyaknya fitur setiap *dataset* dan $i = 1.2.\ldots.n$ menunjukkan jumlah *dataset*. Kemudian untuk hasil perkalian $\boldsymbol{w}$ transpose dengan $\boldsymbol{x}_i$ menjadi :

$$f(\boldsymbol{x}_i) = \boldsymbol{w}^T\boldsymbol{x}_i + b = [\boldsymbol{w}_1.\boldsymbol{w}_2.\ldots.\boldsymbol{w}_m]\begin{bmatrix} \boldsymbol{x}_{i1} \\ \boldsymbol{x}_{i2} \\ \vdots \\ \boldsymbol{x}_{im} \end{bmatrix} + \mathrm{b} = \boldsymbol{w}_1\boldsymbol{x}_{i1} + \boldsymbol{w}_2\boldsymbol{x}_{i2} + \cdots + \boldsymbol{w}_m\boldsymbol{x}_{im} + b$$

Label kelas dinotasikan sebagai $y_i \in \{+1.-1\}$ (Zaki & JR. Wagner Meira, 2020). Bentuk tersebut merupakan fungsi *hyperplane* yang berperan dalam memisahkan kedua data.

Syarat yang dimiliki oleh *hyperplane* yaitu:

1. Memiliki tingkat kesalahan yang paling rendah dalam mengklasifikasikan suatu data.
2. Memaksimalkan jarak terdekat antara setiap kelas (margin). Margin maksimum merupakan pilihan yang paling optimal karena meminimalkan kemungkinan kesalahan klasifikasi.

$$\boldsymbol{w}^T\boldsymbol{x}_i + b > 0 \qquad (2.2)$$

$$\boldsymbol{w}^T\boldsymbol{x}_i + b < 0 \qquad (2.3)$$

Persamaan (2.2) dan (2.3) menggambarkan jarak antara dua kelompok data. Pada persamaan (2.2) berlaku untuk titik data yang berada di kelas positif

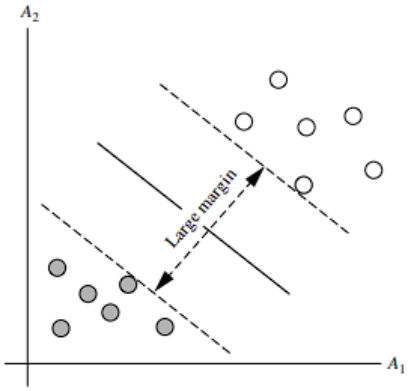
($y_i = +1$). Untuk titik-titik ini, dipastikan bahwa berada di sisi yang benar, yaitu di atas *hyperplane* ($f(\boldsymbol{x}_i) > 0$). Pada persamaan (2.3) berlaku untuk titik yang berada di kelas negatif ($y_i = -1$). Titik ini dibuat di bawah *hyperplane* ($f(\boldsymbol{x}_i) < 0$) (Zaki & JR. Wagner Meira, 2020). Representasi optimal hyperplane dapat ditulisakan sebagai berikut:

$$y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1 \tag{2.4}$$

di mana:

y_i : Nilai kelas (Konstanta)

Margin adalah jarak antara *hyperplane* dengan titik data terdekat dari kedua kelas (Han dkk., 2012). SVM mencoba menemukan *hyperplane* yang membuat margin ini sebesar mungkin. Ini penting karena semakin besar margin, semakin besar kemungkinan bisa memisahkan data baru dengan lebih baik. Pada persamaan (2.2) dan (2.3) menunjukan bahwa setiap titik data akan berada di sisi yang benar dari margin yang disebut *support vector*.



Gambar 2.2 *Support Vector* Yang Paling Dekat Dengan Margin

Pada gambar 2.2 *support vector* ditunjukan dengan titik-titik yang paling dekat dengan garis margin. Memaksimalkan margin dalam SVM berarti mencari vektor $\boldsymbol{w}$ berdimensi terkecil dalam sebuah masalah optimasi konveks

untuk menemukan *hyperplane* optimal. Hal ini dapat diformulasikan sebagai masalah pemrograman kuadrat (*Quadratic Programming*) menggunakan persamaan (2.5) (Zaki & JR. Wagner Meira, 2020).

$$min_{w.b}\left\{\frac{1}{2}||\boldsymbol{w}||^2\right\} \tag{2.5}$$

Dengan syarat :

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1$$

Solusi untuk formula optimasi SVM dengan batasan dalam bentuk pertidaksamaan dapat diperoleh melalui penerapan metode *Lagrange Multipliers.* Dengan menggunakan metode ini, persamaan (2.6) dapat dihasilkan sebagai hasil optimasi.

$$\min L = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{n}\alpha_i[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1] \tag{2.6}$$

Solusi dari optimasi tersebut diperoleh dengan meminimalkan nilai $\left|\left|\boldsymbol{w}\right|\right| = \sqrt{\boldsymbol{w}_1^2 + \boldsymbol{w}_2^2 + \ldots + \boldsymbol{w}_m^2}$ atau $||\boldsymbol{w}||^2 = \boldsymbol{w}_1^2 + \boldsymbol{w}_2^2 + \ldots + \boldsymbol{w}_m^2$, untuk memaksimalkan margin, sambil memastikan kendala $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1$ dipenuhi. Pendekatan ini melibatkan penggunaan pengali Lagrange α_i, di mana solusi dicapai dengan memaksimalkan fungsi dual Lagrangian terhadap α_i dan meminimalkannya terhadap $\boldsymbol{w}$ dan b. Ini dilakukan dengan menghitung turunan pertama dari L terhadap variabel $\boldsymbol{w}$ dan b, lalu menyetarakannya dengan nol. Turunan terhadap variabel $\boldsymbol{w}$ dan b menghasilkan persamaan (2.7) dan (2.8) (Zaki & JR. Wagner Meira, 2020).

$$\frac{\partial}{\partial \boldsymbol{w}}L = \boldsymbol{w} - \sum_{i=1}^{n} y_i\alpha_i\boldsymbol{x}_i \;\; or \; \boldsymbol{w} = \sum_{i=1}^{n} y_i\alpha_i\boldsymbol{x}_i \tag{2.7}$$

$$\frac{\partial}{\partial b} L = \sum_{i=1}^{n} y_i \alpha_i = 0 \tag{2.8}$$

2.1.2.2 SVM Non Linier

SVM non-linier digunakan untuk menangani data yang tidak bisa dipisahkan secara linier. Ketika suatu dataset tidak dapat dipisahkan dengan menggunakan garis lurus, data tersebut disebut sebagai data non-linier, dan metode klasifikasi yang digunakan dikenal sebagai klasifikasi SVM non-linier.

Pada data non-linier, *hyperplane* tidak dapat ditentukan secara efektif menggunakan persamaan (2.5). Sehingga diperlukan margin yang dapat memisahkan data non linier yang disebut *soft* margin. Pada *soft* margin sendiri di butuhkan variabel *slack* yang berperan dalam memperbesar margin, bisa dihitung dengan menggunakan persamaan berikut (Zaki & JR. Wagner Meira, 2020):

$$y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1 - t_i \tag{2.9}$$

di mana:

t_i : Variabel *Slack* (Konstanta)

Misalkan diketahui $\boldsymbol{w} = \begin{bmatrix} 1 \\ 1 \\ 0{,}5 \end{bmatrix}$. $\boldsymbol{x}_1 = \begin{bmatrix} \boldsymbol{x}_{11} \\ \boldsymbol{x}_{12} \\ \boldsymbol{x}_{13} \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$. $\boldsymbol{x}_2 = \begin{bmatrix} \boldsymbol{x}_{21} \\ \boldsymbol{x}_{22} \\ \boldsymbol{x}_{23} \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 1 \end{bmatrix}$ dan bias = 0.5 (Konstanta)

Untuk kelas positif ($y_1 = 1$)

$$1\ [1\ 1\ 0{,}5] \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} + 0{,}5 \geq 1 - t_1$$

$$4 \geq 1 - t_1$$

$$t_1 \geq 1 - 4$$

$$t_1 \geq -3$$

Untuk kelas negatif ($y_2 = -1$)

$$-1\ [1\ 1\ 0{,}5] \begin{bmatrix} 1 \\ 3 \\ 1 \end{bmatrix} + 0{,}5 \geq 1 - t_2$$

$$-2{,}15 + 0{,}5 \geq 1 - t_2$$

$$t_2 \geq 1 + 1{,}65$$

$$t_2 \geq 2{,}65$$

kemudian setelah mendapatkan t_i selanjutnya dilakukan perhitungan soft margin dapat dirumuskan dengan persamaan berikut (Zaki & JR. Wagner Meira, 2020):

$$\min_{\mathbf{w}.\mathrm{b}.\mathrm{t_i}} \left\{ \frac{1}{2} ||\boldsymbol{w}||^2 + C \sum_{i=1}^{n} (t_i) \right\} \tag{2.10}$$

di mana:

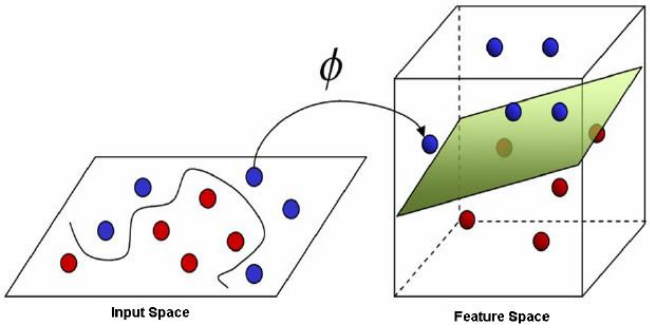
C : Parameter regulasi (Konstanta)

Parameter C berfungsi untuk menyeimbangkan antara kesalahan klasifikasi t_i dan margin. Untuk menentukan nilai yang optimal, berdasarkan penelitian (Huang dkk., 2007), menunjukan bahwa rentang nilai C untuk SVM adalah 10^{-2}dan 10^{4}. saat C dinaikkan nilainya maka lebar margin akan meningkat sehingga meminimalkan kesalahan klasifikasi (Anang & Mike, 2022).

SVM non-linier bekerja dengan cara mentransformasikan data dari ruang dua dimensi ke ruang dengan dimensi lebih tinggi yang disebut ruang kernel seperti pada gambar 2.3.Kernel digunakan untuk lebih mengoptimalkan hasil klasifikasi. Ruang kernel akan menjadikan data terpisah secara linier (Awad & Khanna, 2015). Fungsi kernel memiliki persamaan sebagai berikut:

$$K(\boldsymbol{x}_i.\boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j) \tag{2.11}$$

Dimana $i.j = 1.2.\ldots.n$ yang menunjukkan jumlah *dataset*.



Gambar 2.3 *Input Space To Feature Space*

Misalkan $\boldsymbol{x}_i = [1,2]$ dan $\boldsymbol{x}_j = [2,3]$ kemudian $[\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_1^2, \boldsymbol{x}_1\boldsymbol{x}_2, \boldsymbol{x}_2^2] = \phi(\boldsymbol{x})$. Untuk $\boldsymbol{x}_i, \phi(\boldsymbol{x}_i) = [1,2,1,2,4]$, kemudian untuk $\boldsymbol{x}_j, \phi(\boldsymbol{x}_j) = [2,3,4,6,9]$. Hitung $\phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j) = (1)(2) + (2)(3) + (1)(4) + (2)(6) + (4)(9) = 60$

Beberapa fungsi kernel yang sering diterapkan dalam metode Support Vector Machine (SVM) mencakup (Han dkk., 2012):

1. Kernel Polinomial

$$K(\boldsymbol{x}_i.\boldsymbol{x}_j) = (\boldsymbol{x}_i \cdot \boldsymbol{x}_j + 1)^h \tag{2.12}$$

2. Kernel fungsi Gaussian Radial Basis

$$K(\boldsymbol{x}_i.\boldsymbol{x}_j) = e^{-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma^2}} \tag{2.13}$$

Norm Kuadrat dari $||\boldsymbol{x}_i - \boldsymbol{x}_j||^2$ didefinisikan : $||\boldsymbol{x}_i - \boldsymbol{x}_j||^2 = \sum_{i,j=1}^{n}(\boldsymbol{x}_i - \boldsymbol{x}_j)^2 = ||\boldsymbol{x}_i||^2 - 2\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j + ||\boldsymbol{x}_j||^2$

di mana:

$||\boldsymbol{x}_i||^2 dan\ ||\boldsymbol{x}_j||^2$: Norma kuadrat dari $\boldsymbol{x}_i\, dan\, \boldsymbol{x}_j$

$\boldsymbol{x}_i^{\top}\boldsymbol{x}_j$: Produk dalam antara $\boldsymbol{x}_i\ dan\ \boldsymbol{x}_j$

3. Kernel Sigmoid

$$K(\boldsymbol{x}_i.\boldsymbol{x}_j) = \tanh\left(\kappa \boldsymbol{x}_i \cdot \boldsymbol{x}_j - \delta\right) \tag{2.14}$$

Misalkan $\boldsymbol{x}_1 = \begin{bmatrix}1\\2\end{bmatrix}.\ \boldsymbol{x}_2 = \begin{bmatrix}5\\3\end{bmatrix}.\ \sigma = 1$ (Konstanta) akan digunakan untuk mencari kernel RBF

$$K(\boldsymbol{x}_1.\boldsymbol{x}_1) = e^{-\frac{\left||\boldsymbol{x}_1-\boldsymbol{x}_1|\right|^2}{2\sigma^2}}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_1) = e^{-\frac{0}{2\sigma^2}}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_1) = 1$$

$$K(\boldsymbol{x}_2.\boldsymbol{x}_2) = 1$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = e^{-\frac{\left||\boldsymbol{x}_1-\boldsymbol{x}_2|\right|^2}{2\sigma^2}}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = e^{-\frac{(\mathbf{1}-\mathbf{5})^2+(\mathbf{2}-\mathbf{3})^2}{2\times 1^2}}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = e^{-\frac{17}{2}}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = e^{-8{,}5} \rightarrow \approx 0{,}000203$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = K(\boldsymbol{x}_2.\boldsymbol{x}_1) \approx 0{,}000203$$

Jadi, matriks kernel menjadi:

$$\boldsymbol{K} = \begin{bmatrix}1 & 0{,}000203\\ 0{,}000203 & 1\end{bmatrix}$$

Persamaan umum untuk kasus non linier *support vector* dapat ditulis sebagai berikut (Zaki & JR. Wagner Meira, 2020):

$$\max_{\alpha} L_{dual} = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i.j=1}^{n}\alpha_i\alpha_j y_i y_j\, K(\boldsymbol{x}_i.\boldsymbol{x}_j) \tag{2.15}$$

$$\max_{\alpha} L_{dual} = \alpha_1 + \alpha_2 + \cdots + \alpha_n$$

$$-\frac{1}{2}\big(\alpha_1\alpha_1 y_1 y_1 K(\boldsymbol{x}_1.\boldsymbol{x}_1) + \alpha_1\alpha_2 y_1 y_2 K(\boldsymbol{x}_1.\boldsymbol{x}_2) + \cdots$$

$$+ \alpha_1\alpha_n y_1 y_n K(\boldsymbol{x}_1.\boldsymbol{x}_n) + \alpha_2\alpha_1 y_2 y_1 K(\boldsymbol{x}_2.\boldsymbol{x}_1)$$

$$+ \alpha_2\alpha_2 y_2 y_2 K(\boldsymbol{x}_2.\boldsymbol{x}_2) + \cdots + \alpha_2\alpha_n y_2 y_n K(\boldsymbol{x}_2.\boldsymbol{x}_n) + \cdots$$

$$+ \alpha_n\alpha_1 y_n y_1 K(\boldsymbol{x}_n.\boldsymbol{x}_1) + \alpha_n\alpha_2 y_n y_2 K(\boldsymbol{x}_n.\boldsymbol{x}_2)$$

$$+ \alpha_n\alpha_3 y_n y_3 K(\boldsymbol{x}_n.\boldsymbol{x}_3) + \alpha_n\alpha_4 y_n y_4 K(\boldsymbol{x}_n.\boldsymbol{x}_4)$$

$$+ \alpha_n\alpha_5 y_n y_5 K(\boldsymbol{x}_n.\boldsymbol{x}_5) + \cdots + \alpha_n\alpha_n y_n y_n K(\boldsymbol{x}_n.\boldsymbol{x}_n)\big)$$

Dengan kendala:

$\sum_{i=1}^{n} \alpha_i y_i = 0$ dan $0 \leq \alpha_i \geq C$, di mana $K(\boldsymbol{x}_i.\boldsymbol{x}_j)$ adalah kernel RBF antara dua titik data, y_i adalah label, C adalah parameter pengontrol jarak antar margin, dan n adalah jumlah data (di sini $n = 2$).

Misalkan didapatkan kernel RBF $\boldsymbol{K} = \begin{bmatrix} 1 & 0{,}000203 \\ 0{,}000203 & 1 \end{bmatrix}$, label $y_1 = 1$ dan $y_2 = -1$

$$\max_{\alpha} L_{dual} = \alpha_1 + \alpha_2 - \frac{1}{2}[\alpha_1^2 K(\boldsymbol{x}_1.\boldsymbol{x}_1) + \alpha_2^2 K(\boldsymbol{x}_2.\boldsymbol{x}_2) + 2\alpha_1\alpha_2 y_1 y_2 K(\boldsymbol{x}_1.\boldsymbol{x}_2)]$$

Subtitusikan y_1 dan y_2

$$\max_{\alpha} L_{dual} = \alpha_1 + \alpha_2 - \frac{1}{2}\big[1\alpha_1^2 + 1\alpha_2^2 + 0{,}000203 \times -2\alpha_1\alpha_2\big]$$

$$\max_{\alpha} L_{dual} = \alpha_1 + \alpha_2 - \frac{1}{2}\big[\alpha_1^2 + \alpha_2^2 - 0{,}000406\ \alpha_1\alpha_2\big]$$

Dengan batas

$$\alpha_1 y_1 + \alpha_2 y_2 = 0$$

$$\alpha_1 - \alpha_2 = 0$$

$$\alpha_1 = \alpha_2$$

Optimasi fungsi L dengan mengsubsitusikan $\alpha_1 = \alpha_2$

$$\max_{\alpha} L_{dual} = 2\alpha_1 - \frac{1}{2}\left[2\alpha_1^2 - 0{,}000406\alpha_1^2\right]$$

$$\max_{\alpha} L_{dual} = 2\alpha_1 - \alpha_1^2[1 - 0{,}000203]$$

$$\max_{\alpha} L_{dual} = 2\alpha_1 - 0{,}999797\alpha_1^2$$

$$\frac{\partial L}{\partial \alpha_1} = 2 - 2 \times 0{,}999797\alpha_1$$

$$2 = 1{,}999594\alpha_1$$

$$\alpha_1 = \frac{2}{1{,}999594}$$

$$\alpha_1 \approx 1{,}0002$$

Jadi nilai $\alpha_1 = \alpha_2 = 1{,}0002$

Namun, untuk mendapatkan garis pemisah yang optimal tidak mudah karena nilai vektor pembobot $\boldsymbol{w}$ belum diketahui. Untuk menentukan nilai $\boldsymbol{w}$ dapat menggunakan persamaan (2.17) (Zaki & JR. Wagner Meira, 2020).

$$\boldsymbol{w} = \sum_{i=1}^{n} y_i \alpha_i \phi(\boldsymbol{x}_i) \tag{2.17}$$

Vektor bobot hanya digunakan untuk kernel linier saja karena kernel linier tidak mengubah ruang fitur (tetap berada di ruang dimensi asli), model SVM akan menghasilkan *hyperplane* linier yang sama seperti pada SVM linier. Dalam hal ini, $\boldsymbol{w}$ tetap digunakan untuk menggambarkan *hyperplane* pemisah.

Sedangkan untuk mencari bias dapat menggunakan persamaan (2.18) (Zaki & JR. Wagner Meira, 2020).

$$b_i = y_i - \sum_{i=1}^{n} y_i \alpha_i K(\boldsymbol{x}_i . \boldsymbol{x}_j) \tag{2.18}$$

Dari perhitungan yang telah dilakukan di atas, bias dihitung dengan data 1 dan data 2.

Untuk data 1

$$b_1 = y_1 - y_1\alpha_1 K(\boldsymbol{x}_1.\boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_1.\boldsymbol{x}_2)$$

$$b_1 = 1 - (1{,}0002 \times 1 \times 1 + 1{,}0002 \times -1 \times 0.000203)$$

$$b_1 = 0.0000030406$$

Untuk data 2

$$b_2 = y_2 - y_1\alpha_1 K(\boldsymbol{x}_2.\boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_2.\boldsymbol{x}_2)$$

$$b_2 = -1 - (1{,}0002 \times 1 \times 0.000203 + 1{,}0002 \times -1 \times 1)$$

$$b_2 = 0.0000030406$$

Jadi nilai bias nya adalah 0.0000030406

Pada buku (Zaki & JR. Wagner Meira, 2020) sudah didapatkan persamaan umum untuk SVM non linier. Persamaannya sebagai berikut:

$$f(\boldsymbol{x}) = \sum_{i.j=1}^{n} y_i\alpha_i K(\boldsymbol{x}_i.\boldsymbol{x}_j) + b \tag{2.19}$$

$$g(\boldsymbol{x}) = sign(f(\boldsymbol{x})) \tag{2.20}$$

dimana:

$\boldsymbol{x}_i.\boldsymbol{x}_j$: Support vectors (titik-titik data penting yang sudah ditemukan)

α_i dan b : Angka yang sudah dihitung sebelumnya oleh SVM

n : Banyaknya *support vector*

Jika hasil dari persamaan ini lebih besar dari 0, maka titik tersebut termasuk dalam kelas 1. Jika hasilnya kurang dari 0, maka titik tersebut masuk kelas 2. Misalkan semua varibel sudah di peroleh, kemudian akan dilakukan perhitungan fungsi keputusan

$$f(\boldsymbol{x}_1) = y_1\alpha_1 K(\boldsymbol{x}_1.\boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_1.\boldsymbol{x}_2) + b$$

$$f(\boldsymbol{x}_1) = 1 \times 1{,}0002 \times 1 + 1{,}0002 \times -1 \times 0{,}000203 + 0.0000030406$$

$$f(\boldsymbol{x}_1) = \ 1{,}0002 - 0{,}0002030406 + 0{,}0000030406 = 1$$

$$g(\boldsymbol{x}_1) = sign(1)$$

$$g(\boldsymbol{x}_1) = 1$$

$$f(\boldsymbol{x}_2) = y_1\alpha_1 K(\boldsymbol{x}_2.\boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_2.\boldsymbol{x}_2) + b$$

$$f(\boldsymbol{x}_2) = 1 \ \times 1{,}0002 \times 0{,}000203 + 1{,}0002 \times -1 \times 1 + 0.0000030406$$

$$f(\boldsymbol{x}_2) = \ 0{,}0002030406 - 1{,}0002 + 0{,}0000030406 = -0{,}9999939188$$

$$g(\boldsymbol{x}_2) = sign(-0{,}9999939188)$$

$$g(\boldsymbol{x}_2) = -1$$

$f(\boldsymbol{x}_1) = 1$, jadi $\boldsymbol{x}_1$ diklasifikasikan sebagai **kelas +1**.

$f(\boldsymbol{x}_2) = -0{,}9999939188$, jadi $\boldsymbol{x}_2$ diklasifikasikan sebagai **kelas -1**.

Fungsi keputusan ini bekerja dengan baik, karena data latih diklasifikasikan sesuai dengan label aslinya.

2.1.2 *Confusion Matrix*

Confusion Matrix adalah alat yang digunakan untuk mengevaluasi kinerja model klasifikasi dalam mengidentifikasi kelas data dengan akurat (Maulana Al-Fakih dkk., 2024). Kinerja model klasifikasi dinilai berdasarkan empat komponen utama dalam *confusion matrix*, yaitu:

Tabel 2.1 Komponen Utama *Confusion Matrix*

		Predicted data	
		True	False
Actual Data	True	*True Positive* (TP)	*False Positive* (FP)
	False	*False Negative* (FN)	*True Negative* (TN)

Dimana tiap komponen dari *confusion matrix* diatas adalah (Anang & Mike, 2022):

1. *True Positive* (TP), yaitu data dari kelas positif yang diklasifikasikan dengan benar sebagai kelas positif.
2. *False Positive* (FP), yaitu data dari kelas negatif yang diklasifikasikan dengan salah sebagai kelas positif.
3. *True Negative* (TN), yaitu data dari kelas negatif yang diklasifikasikan dengan benar sebagai kelas negatif.
4. *False Negative* (FN), yaitu data dari kelas positif yang diklasifikasikan dengan salah sebagai kelas negatif.

Melalui penggunaan komponen utama dari confusion matrix yang disebutkan, performa dan efektivitas model klasifikasi dapat dinilai melalui beberapa metrik. Presisi mengukur seberapa efektif model dalam memprediksi data kelas positif. *Recall* menunjukkan seberapa baik model dalam mengidentifikasi kelas positif secara benar. Akurasi dan *F1-Score* adalah metrik yang menggambarkan keseimbangan antara presisi dan *recall* dari model klasifikasi. Rumus untuk menghitung keempat metrik dapat dilihat pada tabel 2.2.

Tabel 2.2 Rumus Performa *Confusion Matrix*

Presisi	$\frac{TP}{TP+FP}$x100%
Akurasi	$\frac{TP+TN}{TP+TN+FP+FN}$x100%
Recall	$\frac{TP}{TP+FN}$x100%
F1-Sore	$2x\frac{Presisi \text{ x } Recall}{Presisi + Recall}$x100%

2.1.3 *K-fold Cross Validation*

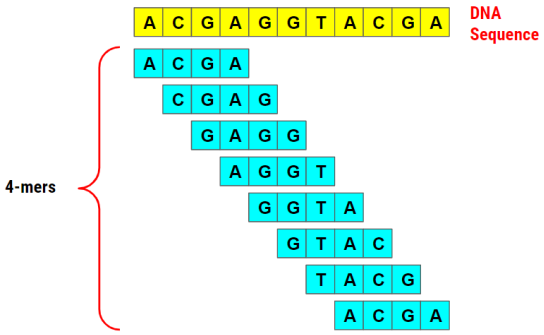
K-fold cross validation digunakan untuk mengatasi keterbatasan ketersediaan dataset yang terpisah untuk proses pelatihan, pemilihan model, dan evaluasi model. Pada prosedur KCV, dataset asli D_n dibagi menjadi k subset dengan nilai k yang sudah ditentukan sebelumnya. Biasanya, $(k-2)$ subset digunakan untuk pelatihan model (*Training - TR*), satu subset untuk pemilihan model (*Model Selection - MS*), dan satu subset untuk evaluasi kesalahan model (*Error Estimation - EE*). Mengingat jumlah lipatan k yang telah ditentukan. Nilai k dipilih secara apriori, seperti 5, 10, atau 20 (Anguita dkk., 2022). Nilai k disini diperlakukan seperti hiperparameter yang bisa bernilai apapun dalam rentang $k \in \{3. \ldots . n\}$, Pendekatan ini memungkinkan penyeimbangan antara jumlah pola yang digunakan membangun model dan persentase data untuk mengevaluasi performa model klasifikasi. Hal ini penting penggunaan kernel fungsi RBF, dimana dilakukan tuning parameter C untuk menghasilkan model yang optimal.

2.1.4 *Synthetic Minority Oversampling* (SMOTE)

Synthetic Minority Oversampling Technique atau SMOTE bekerja dengan menghasilkan data tambahan secara sintetis untuk mengatasi masalah ketidakseimbangan distribusi data (Sutoyo dkk., 2020). Teknik ini sering diterapkan dalam klasifikasi, termasuk dalam bidang bioinformatika, untuk menangani ketidakseimbangan jumlah sampel antara kelas. Dengan meningkatkan jumlah data pada kelas yang kurang representatif, teknik ini membantu memperbaiki kualitas model klasifikasi dan membuatnya lebih akurat dalam mengidentifikasi kelas minoritas. Ini penting dalam aplikasi bioinformatika di mana data seringkali tidak seimbang, seperti dalam analisis sekuens gen.

2.1.5 *K-Mers*

Metode *K-Mers* yang diterapkan untuk ekstraksi fitur bertujuan untuk membagi data menjadi satu atau lebih kelompok yang memiliki kesamaan, sehingga memudahkan analisis dan perhitungan akurasi model (Nolly dkk., 2023). Dengan menggunakan *K-Mers*, fitur-fitur penting dalam data dapat diidentifikasi dan dikelompokkan berdasarkan kesamaan urutan nukleotida atau pola tertentu. Hal ini menyederhanakan proses evaluasi dan membantu dalam memperoleh hasil akurasi yang lebih jelas dan terukur, terutama dalam analisis sekuens genetik. Ekstraksi fitur *K-Mers* dihasilkan dengan melakukan kombinasi dari 4 basa nukleotida, yaitu Adenin (A), Cytosine (C), Guanine (G) dan Thymine (T). Jumlah kombinasi fitur yang terbentuk adalah 4k dengan nilai k ≥ 1 (Choiriyati dkk., 2020).



Gambar 2.4 Contoh 4-Mers

Pada gambar 2.4, *k-mers* dengan panjang *mers* = 4 divisualisasikan dengan menampilkan subsekuens dari urutan DNA yang memiliki panjang empat karakter. Visualisasi ini membantu untuk memahami distribusi dan frekuensi dari setiap kombinasi *k-mers* yang muncul dalam dataset DNA.

2.1.6 *Natural Language Processing* (NLP)

Perkembangan teknologi saat ini NLP dikenal sebagai proses pengolahan teks yang memungkinkan mesin untuk memahami bahasa manusia. Karena mesin secara alami tidak dapat memahami bahasa manusia, NLP diperlukan untuk membantu menerjemahkan dan memproses bahasa alami agar dapat dipahami dan digunakan oleh sistem komputer (Munasatya & Novianto, 2020). Dalam pembentukan sebuah NLP memerlukan beberapa atau salah satu pipeline berikut: (1) Word tokenization, (2) Stopwords removal, (3) Stemming, (4) Build bag of words, (5) Split datasets, (6) Vectorizer.

2.1.7 *Machine Learning*

Machine learning merupakan cabang dari kecerdasan buatan yang memiliki proses komputasi menggunakan input data untuk mencapai tugas tanpa diprogram secara eksplisit sehingga menghasilkan hasil tertentu (El Naqa & Murphy, 2015). Algoritma *machine learning* dapat menganalisis data, mengidentifikasi pola, dan membuat keputusan atau prediksi berdasarkan pengalaman sebelumnya. Dalam hal ini, membuat machine learning menjadi alat yang memudahkan untuk meyelesaikan masalah di berbagai bidang seperti bidang genetika.

Penerapan algoritma *machine learning* dalam bidang genetika dapat digunakan untuk menganalisis data genomik yang kompleks. Selain itu, algoritma ini juga mampu mengidentifikasi pola dan melakukan klasifikasi dalam sekuens DNA. Salah satu metode yang sering digunakan adalah *Support Vector Machine* (SVM), yang memiliki kemampuan untuk menangani data berdimensi tinggi.

Metode ini efektif dalam memisahkan data ke dalam beberapa kategori berdasarkan fitur genetik yang relevan.

2.1.8 Sekuens DNA

DNA (*Deoxyribonucleic Acid*) atau asam deoksiribonukleat adalah molekul yang menyimpan informasi genetik pada semua makhluk hidup, termasuk manusia (Akbar dkk., 2023). DNA terdiri dari dua untai yang membentuk struktur heliks ganda, di mana setiap untai tersusun atas nukleotida. Setiap nukleotida memiliki tiga bagian utama: gula deoksiribosa, gugus fosfat, dan salah satu dari empat basa nitrogen, yaitu adenin (A), guanin (G), sitosin (C), dan timin (T). Urutan dari keempat basa nitrogen inilah yang menentukan informasi genetik yang ada di dalam DNA.

Urutan DNA berfungsi sebagai kode yang menyimpan instruksi untuk sintesis protein melalui proses transkripsi dan translasi, yang pada akhirnya berperan dalam menentukan fungsi biologis dari sel-sel dalam tubuh (Indah dkk., 2007). Variasi atau perubahan pada urutan DNA, yang dikenal sebagai mutasi genetik, dapat mengganggu fungsi normal gen dan menyebabkan berbagai penyakit, seperti diabetes mellitus, kanker, dan sejumlah gangguan genetik lain.

2.1.9 Diabetes Mellitus

Diabetes Mellitus merupakan suatu penyakit menahun atau kronis yang ditandai oleh hiperglikemia, yaitu kadar glukosa darah melebihi nilai normal (Rumpun dkk., 2022). Hiperglikemia kronis pada diabetes melitus (DM) terkait dengan kerusakan, disfungsi, dan kegagalan organ serta jaringan, termasuk retina,

ginjal, saraf, jantung, dan pembuluh darah. Federasi Diabetes Internasional memperkirakan prevalensi global diabetes melitus mencapai 366 juta pada tahun 2011, dan angka ini diperkirakan akan meningkat menjadi 552 juta pada tahun 2030 (Whiting dkk., 2011).

Menurut Organisasi Kesehatan Dunia (WHO), diabetes melitus dibagi menjadi empat kategori utama, yaitu diabetes melitus tipe 1 (T1DM), diabetes melitus tipe 2 (T2DM), diabetes gestasional, dan diabetes monogenik (El-Attar dkk., 2022). Diabetes mellitus tipe 1 terjadi akibat penghancuran autoimun pada sel beta pankreas, yang mengakibatkan tidak adanya produksi insulin. Diabetes mellitus tipe 2, yang paling umum, disebabkan oleh produksi insulin yang tidak mencukupi atau berkurangnya sensitivitas reseptor insulin, sehingga menghambat penyerapan glukosa ke dalam sel. Diabetes gestasional muncul selama masa kehamilan, dengan prevalensi antara 5-15% pada wanita di seluruh dunia. Sementara itu, diabetes monogenik, yang sering salah didiagnosis sebagai diabetes tipe 1 atau 2, disebabkan oleh mutasi pada satu gen atau beberapa gen terkait.

2.2 Kajian Integrasi Topik dengan Al-Quran/Hadits

Al-Qur'an tidak membahas persoalan diabetes mellitus secara spesifik, namun banyak perintah di dalam Al-Quran menganjurkan untuk menjaga kesehatan dan mencegah kebinasaan. Salah satunya dalam Surat Al-Baqarah ayat 195 berbunyi (Kemenag, 2024a):

وَاَنْفِقُوْا فِيْ سَبِيْلِ اللّٰهِ وَلَا تُلْقُوْا بِاَيْدِيْكُمْ اِلَى التَّهْلُكَةِ ۛ وَاَحْسِنُوْا ۛ اِنَّ اللّٰهَ يُحِبُّ الْمُحْسِنِيْنَ

"Dan infakkanlah (hartamu) di jalan Allah, dan janganlah kamu menjatuhkan dirimu sendiri ke dalam kebinasaan, dan berbuat baiklah. Sesungguhnya Allah

menyukai orang-orang yang berbuat baik."(QS. Al-Baqarah: 195)

Ayat ini dapat dihubungkan dengan penelitian ini dalam konteks menjaga kesehatan dan mencegah kebinasaan. Dalam ajaran agama, sering kali ada anjuran untuk memelihara kesehatan tubuh serta mencegah kerusakan, baik secara fisik maupun mental. Ayat tersebut juga menasihati manusia agar tidak merusak diri sendiri, menjaga kebersihan, mengatur asupan makanan, serta menjaga keseimbangan hidup. Merawat kesehatan dianggap sebagai bentuk tanggung jawab pribadi dan bagian dari pelaksanaan ibadah. Diabetes Mellitus adalah penyakit yang bisa menyebabkan komplikasi serius jika tidak ditangani dengan baik. Upaya untuk mencegah penyakit melalui teknologi, seperti penelitian dalam klasifikasi sekuens DNA, merupakan salah satu bentuk ikhtiar manusia untuk tidak jatuh ke dalam kebinasaan. Penelitian ini bertujuan untuk memberikan kontribusi dalam mendeteksi dan mengklasifikasikan DNA terkait Diabetes Mellitus dengan harapan dapat membantu dalam pencegahan dan penanganan lebih dini. Hal ini selaras dengan ajakan dalam ayat tersebut untuk "tidak menjatuhkan diri ke dalam kebinasaan" dengan melakukan upaya-upaya yang bermanfaat, termasuk melalui ilmu pengetahuan dan teknologi.

Ayat dari Surah An-Nur (24:45) yang berbunyi (Kemenag, 2024b):

وَاللّٰهُ خَلَقَ كُلَّ دَاۤبَّةٍ مِّنْ مَّاۤءٍۚ فَمِنْهُمْ مَّنْ يَّمْشِيْ عَلٰى بَطْنِهٖۚ وَمِنْهُمْ مَّنْ يَّمْشِيْ عَلٰى رِجْلَيْنِۚ وَمِنْهُمْ مَّنْ يَّمْشِيْ عَلٰٓى اَرْبَعٍۗ يَخْلُقُ اللّٰهُ مَا يَشَاۤءُۗ اِنَّ اللّٰهَ عَلٰى كُلِّ شَيْءٍ قَدِيْرٌ

"Dan Allah menciptakan setiap makhluk hidup dari air; maka di antara mereka ada yang berjalan di atas perutnya, di antara mereka ada yang berjalan dengan dua kaki, dan di antara mereka ada yang berjalan dengan empat kaki. Allah menciptakan apa yang Dia kehendaki. Sesungguhnya Allah Maha Kuasa atas segala sesuatu."

Allah menyebutkan adanya keragaman ciptaan-Nya dalam hal bentuk, fungsi, dan cara bergerak. Berdasarkan cara bergerak, hewan dapat dibedakan menjadi beberapa macam, yaitu ada yang merayap, berjalan dengan dua kaki, dan berjalan dengan empat kaki. Secara ilmiah, perbedaan cara bergerak ini mencerminkan perbedaan dalam struktur genetik dan fisiologis makhluk hidup. Misalnya, ayam berjalan dengan dua kaki karena adanya evolusi kerangka, yang diatur oleh informasi genetik (DNA). Hewan lain memiliki ciri fisik yang sesuai dengan cara mereka bergerak, dan semua ini diatur oleh DNA mereka. Dalam pandangan Al-Qur'an, keanekaragaman ini adalah tanda-tanda kebesaran Allah yang menunjukkan kekuasaan-Nya dalam menciptakan berbagai makhluk dengan cara hidup dan bergerak yang berbeda.

Hadits Nabi Muhammad SAW yang berbunyi: *"Sesungguhnya Allah tidak menurunkan penyakit kecuali Dia juga menurunkan penawarnya" (HR. Bukhari* 5678*)*. Hadits tersebut mengajarkan bahwa setiap penyakit yang ada di muka bumi ini memiliki solusi atau penawarnya. Penelitian ini, yang berfokus pada klasifikasi sekuens DNA manusia untuk mendeteksi Diabetes Mellitus, sejalan dengan pesan hadits ini. Dalam penelitian ini, algoritma Support Vector Machine (SVM) digunakan untuk membantu memahami pola genetik yang mungkin terkait dengan Diabetes Mellitus. Dengan pemahaman yang lebih baik tentang penyakit ini dari sudut genetika, penelitian ini diharapkan bisa menjadi salah satu jalan menuju penemuan atau pengembangan penanganan yang lebih baik, baik dalam hal deteksi dini maupun intervensi medis yang lebih efektif.

Secara keseluruhan, integrasi ini menunjukkan bahwa usaha dalam melakukan penelitian ilmiah, khususnya yang berhubungan dengan kesehatan

seperti penanganan Diabetes Mellitus, adalah bagian dari upaya untuk menemukan solusi atas penyakit yang telah Allah ciptakan. Penelitian ini menjadi wujud nyata dalam menjalankan perintah Allah untuk menjaga diri dan juga sebagai upaya mencari penawar, sesuai dengan hadits di atas.

2.3 Kajian Topik Dengan Teori Pendukung

Berdasarkan teori pendukung, penelitian ini dapat disusun dalam beberapa tahapan. Tahapan yang dilakukan dalam penelitian ini dimulai dengan pengumpulan data sekuens DNA dari website NCBI. Setelah data terkumpul, dilakukan proses *preprocessing* data yang terdiri dari tiga langkah utama. Pertama, dilakukan pembersihan data untuk mendeteksi dan memperbaiki atau menghapus data yang salah, rusak, duplikat, atau memiliki format yang tidak sesuai, dengan tujuan untuk memastikan keakuratan, konsistensi, dan kegunaan data. Kedua, tahap transformasi data dilakukan dengan mengubah data dari format aslinya menjadi format yang lebih sesuai, seperti mengubah atribut kategorikal menjadi numerik. Ketiga, untuk menangani *dataset* yang tidak seimbang, digunakan teknik oversampling, yaitu SMOTE (*Synthetic Minority Over-sampling Technique*), untuk meningkatkan jumlah sampel dari kelas minoritas.

Setelah proses *preprocessing* selesai, data dibagi menjadi dua bagian, yaitu data pelatihan dan data pengujian, dengan rasio pembagian yang dipilih berdasarkan hasil uji coba yang memberikan tingkat akurasi tinggi. Pada tahap selanjutnya, dilakukan proses klasifikasi menggunakan metode *Support Vector Machine* (SVM) non linier. Dalam proses ini, dipilih fungsi kernel yang tepat untuk membangun model SVM, dan kernel RBF akan diujicobakan pada penelitian ini.

Parameter yang sesuai juga ditentukan sebelum model SVM dibuat dengan fungsi kernel dan parameter yang telah ditentukan tersebut. Terakhir, dilakukan evaluasi kinerja model dilakukan dengan menggunakan KCV dan *confusion matrix* untuk menganalisis hasil klasifikasi model terhadap penderita Diabetes Mellitus.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian yang digunakan dalam studi ini adalah penelitian kuantitatif. Penelitian kuantitatif bertujuan untuk mengumpulkan dan menganalisis data numerik dengan menggunakan metode statistik dan matematis untuk mengukur, membandingkan, dan menjelaskan fenomena. Dalam konteks penelitian ini, pendekatan kuantitatif akan memungkinkan peneliti untuk melakukan analisis yang mendalam terhadap data sekuens DNA dan hasil klasifikasi model SVM, serta untuk menentukan hubungan antara variabel-variabel yang diteliti. Dengan menggunakan teknik kuantitatif, penelitian ini dapat memberikan hasil yang lebih objektif dan dapat digeneralisasikan, serta memberikan dasar yang kuat untuk menarik kesimpulan yang valid dan melakukan prediksi berdasarkan data yang diperoleh.

3.2 Data dan Sumber Data

Data yang digunakan pada penelitian ini adalah data sekuens DNA manusia penderita diabetes mellitus tipe 1 dan 2, serta sekuens DNA manusia sehat, dimana terdapat beberapa protein yang diambil yaitu *autoantigen* pada diabetes tipe 1, Slit-3 pada diabetes tipe 2 dan non diabetes memuat protein selain protein yang dimiliki oleh diabetes tipe 1 dan 2. Data tersebut diperoleh dari *website* NCBI di bagian *databases* nucleotida yang merupakan *website* penyedia data penelitian yang dirancang khusus untuk mendukung penelitian terkait dengan bioinformatika dan

gen makhluk hidup. Data bisa di unduh lewat QR Code berikut

Gambar 3.1 *QR Code Download Dataset*

3.3 Teknik Analisis Data

Adapun tahapan yang dilakukan dalam penelitian adalah sebagai berikut:

1. Melakukan pengambilan data di NCBI dengan cara berikut:
   a. Masuk ke situs NCBI melalui https://www.ncbi.nlm.nih.gov.
   b. Di halaman utama, akan melihat kotak pencarian. Disitu dapat memasukkan kata kunci yang sesuai dengan topik penelitian, seperti : “diabetes type 1”, “diabetes type 2”.
   c. Setelah memasukkan kata kunci, pilih database yang relevan dari menu dropdown di sebelah kotak pencarian, meliputi : Gene, Protein, Nucleotide.
   d. Filter hasil sesuai organisme, misal : *homo sapiens.*
   e. Unduh data dengan format FASTA.
2. Melakukan *Preprocessing* Data.

   Proses *preprocessing* data melibatkan beberapa tahap:

   a. Pembersihan Data: proses untuk mendeteksi dan memperbaiki (atau menghapus) data untuk memastikan data akurat, konsisten, dan dapat

digunakan. Terdapat beberapa proses cleaning data sekuens DNA yang digunakan, sebagai berikut:

1) *Whitespace*: proses *cleaning* data untuk menghapus spasi diantara sekuens DNA
2) Penghapusan nucleotida yang tidak diketahui, seperti adanya ‘N’(yaitu selain A, C, G atau T) pada sekuens DNA, sehingga seluruh sekuens DNA yang mengandung ‘N’ akan dihapus.
3) Penghapusan salah satu sekuens DNA yang duplikat, sehingga hanya satu sekuens DNA yang digunakan

b. Transformasi Data: Tahap ini mengubah data dari format aslinya menjadi format lain yang lebih sesuai untuk keperluan tertentu, sesuai yang ada di sub bab 2.1.5. Misalnya, atribut dengan tipe data kategorikal akan diubah menjadi numerik menggunakan tokenizer dengan cara setiap rangkaian tiga nukleotida berturut-turut (4-mers) diubah menjadi representasi numerik. Setiap kombinasi 4-mers diberi kode angka yang unik, misalnya, kombinasi seperti 'AAAC', 'AACA', dan 'ACGG' masing-masing direpresentasikan dengan angka tertentu.
c. Oversampling Data: Teknik ini digunakan untuk meningkatkan jumlah sampel dari kelas minoritas dalam dataset yang tidak seimbang menjadi seimbang, mendekati atau sama dengan jumlah sampel kelas mayoritas. Metode oversampling yang digunakan adalah SMOTE (Synthetic Minority Over-sampling Technique).

3. Membagi Data untuk Pelatihan dan Pengujian.

Dataset akan dibagi menjadi dua bagian, yaitu data pelatihan (*training*) dan

data pengujian (*testing*). Pada data pelatihan terdiri dari X_train yang merupakan data fitur untuk pelatihan model dan Y_train merupakan label yang sesuai dengan data fitur dalam X_train. Sedangkan data pengujian terdiri dari X_test yang merupakan data fitur untuk pengujian performa model setelah dilatih dan Y_test merupakan label yang sesuai dengan data fitur dalam X_test. Pemilihan rasio berdasarkan hasil uji coba dengan akurasi tinggi.

4. Proses Klasifikasi Menggunakan SVM non-linier.

   Langkah-langkah dalam proses klasifikasi menggunakan metode SVM adalah sebagai berikut:

   a. Memilih fungsi kernel yang tepat untuk model SVM, pada penelitian ini kernel RBF sebagai kernel utama akan diujicobakan dengan perbandingan kernel lain seperti sigmoid dan polinomial. Rumus dari kernel tersebut dapat dilihat di persamaan (2.12), (2.13), dan (2.14).
   b. Menentukan parameter C hingga optimal yang akan digunakan untuk membangun model SVM.
   c. Membuat model SVM dengan fungsi kernel dan parameter yang telah ditentukan menggunakan persamaan (2.11).
      1) Mencari kernel RBF menggunakan persamaan (2.13)

         Sehingga didapatkan hasil dari subtitusi $\boldsymbol{x}_i$ dan $\boldsymbol{x}_j$ sebagai berikut:

         $\boldsymbol{K} = \{K(\boldsymbol{x}_i . \boldsymbol{x}_j)\}$, dimana $i.j = 1.2. \ldots . n$ yang merupakan jumlah data.

      2) Mencari nilai α_i menggunakan persamaan (2.15) sehingga diperoleh α_i dari persamaan *langrange*.

3) Mencari bias menggunakan persamaan (2.18).

4) Memasukan semua variabel yang telah diketahui ke dalam fungsi umum svm non linier pada persamaan (2.19) dan (2.20).

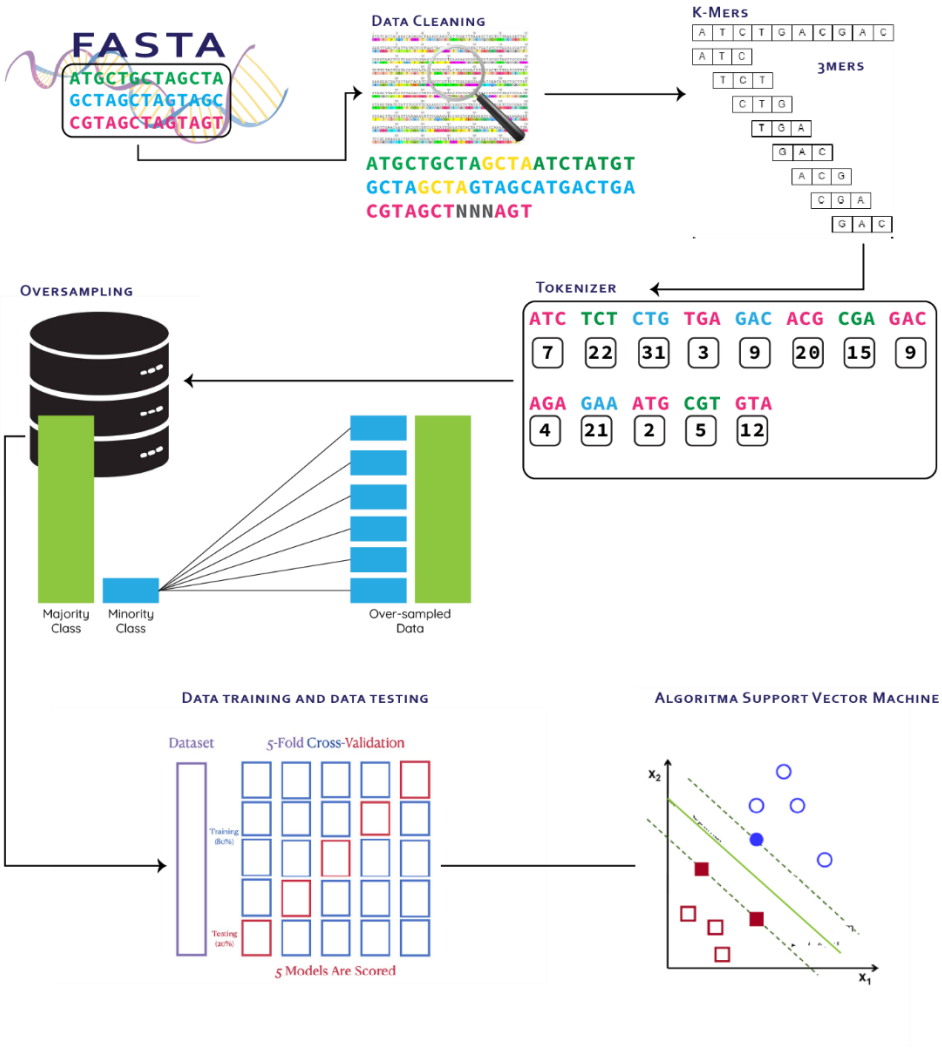
5. Melakukan proses tahapan evaluasi sebagai berikut:

   a. Melakukan *confusion matrix* pada model klasifikasi sekuens DNA pada penderita Diabetes Mellitus.

   b. Melakukan evaluasi model dengan *cross validation KCV.*

   c. Menganalisis hasil klasifikasi.

3.4 Diagram Alur Penelitian



Gambar 3.2 Diagram Alir Penelitian

BAB IV

HASIL DAN PEMBAHASAN

4.1 *Preprocessing* Data

4.1.1 Data *Cleaning*

Proses pembersihan dataset DNA manusia untuk analisis diabetes mellitus mencakup beberapa langkah penting. Pertama, data DNA dari individu dengan diabetes mellitus tipe 1, tipe 2, dan yang tidak memiliki diabetes akan digabungkan menjadi satu dataset untuk mempermudah analisis. Kedua, dilakukan penghapusan data DNA yang memiliki whitespace berlebihan dan filtering panjang sekuens, sehingga hanya data dengan panjang antara 1.000 dan 10.000 nukleotida yang akan disertakan. Ketiga, identifikasi data sekuens yang hilang dilakukan, dan data yang memiliki nilai hilang akan dihapus. Keempat, data sekuens DNA yang terdeteksi sebagai duplikat juga akan dihapus untuk menghindari bias dalam analisis. Berdasarkan penelitian Akkaya & Kalkan (2021), data sekuens DNA manusia terdiri dari empat nukleotida, yaitu A, C, G, dan T. Untuk memastikan kualitas dataset DNA manusia yang akan digunakan dalam pembuatan model, data juga akan dibersihkan dari sekuens berkualitas buruk, tidak lengkap, atau tidak diketahui. Berikut ini adalah ilustrasi proses pembersihan data pada sekuens DNA manusia.

Tabel 4.1 Proses Cleaning Data Sekuens DNA

Proses Cleaning Data	Sebelum	Sesudah
Whitespace	AAC GT A GTCGG ...	AACGTAGTCGG...
Sekuens 1000 – 10.000 nukleotida	(ada)	(dihapus)

Sekuens Buruk dan Hilang	AACNNGTAANA	(dihapus)
Duplikat	AACGTAGTCGG... AACGTAGTCGG...	AACGTAGTCGG...

```
#Cek duplikat DNA
print(f'there are {df.duplicated().sum()} duplicated data from {len(df)} data')

df_after_dup = df.drop_duplicates()

df_after_dup = df_after_dup.reset_index(drop=True)
print(f'now there are {df_after_dup.duplicated().sum()} duplicated data from
{len(df_after_dup)} data')

# check whitespaces
import re

w_idx = []
for i in range(len(df)):
  if re.match(r"\s", df['sequence'][i]):
    w_idx.append(i)

w_idx

for i in range(len(df)):
    df['sequence'][i] = df['sequence'][i].strip()

# filter based on the sequence length between 1000 - 10.000
df_DNA = df_DNA[(df_DNA['length'] >= 1000) & (df_DNA['length'] <=
10000)]

df_DNA = df_DNA.sort_values(by=['length'])
df_DNA = df_DNA.reset_index(drop=True)

# remove non-DNA sequences
true_DNA = df_DNA.copy()
for i in range(len(sequences)):
    if is_DNA(sequences[i]) == False:
        true_DNA = true_DNA.drop(index=i)

true_DNA.shape
```

Gambar 4.1 *Code Cleaning* Data

Tabel 4.2 Frekuensi Data Sekuens DNA Manusia

Sebelum *Cleaning Data*		
DMT1	DMT2	NON-DM
213	1296	1000
Setelah *Cleaning Data*		
DMT1	DMT2	NON-DM
145	443	989

```
from typing_extensions import Concatenate
datas = []

# read csv files
for i in range(len(files)):
    data = pd.read_csv(files[i])
    datas.append(data)

# concatenate csvs files into one
df = pd.concat(datas, ignore_index=True)
```

Gambar 4.2 *Code* Frekuensi Data Setelah *Cleaning*

4.1.2 *K-mers*

K-mers encoding akan membagi struktur DNA menjadi potongan-potongan substring dengan panjang k-nukleotida untuk mengidentifikasi pola dan karakteristik dalam sekuens DNA. Pola atau karakteristik ini nantinya akan digunakan sebagai fitur dari data sekuens DNA. Dalam penelitian ini, nilai k yang digunakan adalah 3, 4, dan 5, sehingga data DNA akan diekstraksi menjadi pasangan 3, 4, dan 5-nukleotida. Proses k-mers encoding dilakukan dengan bantuan fungsi k-mers encoding yang telah dibuat menggunakan Python. Berikut ini adalah tampilan data DNA yang telah diekstraksi polanya menggunakan *k-mers encoding*.

```
def kmers(seq, size):
    kmers = [seq[x:x+size].lower() for x in range(len(seq)-size+1)]
    return kmers

data['3mers'] = data.apply(lambda x: kmers(x['sequence'], size=3), axis=1)
data['4mers'] = data.apply(lambda x: kmers(x['sequence'], size=4), axis=1)
data['5mers'] = data.apply(lambda x: kmers(x['sequence'], size=5), axis=1)
data['6mers'] = data.apply(lambda x: kmers(x['sequence'], size=6), axis=1)
```

Gambar 4.3 *Code K*-mers

Tabel 4.3 Hasil Dari K-mers

Sekuens DNA	K-Mers
AAACGCTAGC	AAA, AAC, ACG, CGC, GCT, CTA, TAG, AGC
ACGATCGAAT	ACGA,CGAT, GATC, ATCG, TCGA, CGAA, GAAT, AATT
ACCAATGTAC	ACCCA, CCAAT, CAATG, AATGT,ATGTA, TGTAC, GTACG

4.1.3 *Tokenizer*

Tokenizer dalam analisis sekuens DNA manusia adalah alat yang berfungsi untuk memecah rantai panjang DNA menjadi unit-unit yang lebih kecil, seperti k-mers atau substring, yang lebih mudah dianalisis. DNA terdiri dari empat nukleotida dasar—Adenin (A), Sitosin (C), Guanin (G), dan Timin (T) dan tokenizer membantu membagi urutan nukleotida ini menjadi pola yang berulang atau unit-unit terstruktur dengan panjang tertentu. Tokenisasi pada data DNA memungkinkan penangkapan pola spesifik, karakteristik, atau motif genetik tertentu yang bisa dikaitkan dengan kondisi atau sifat biologis, seperti predisposisi genetik terhadap penyakit tertentu. Dalam proses pembelajaran mesin, token yang dihasilkan ini berperan sebagai fitur penting yang dapat diolah untuk mengidentifikasi karakteristik atau klasifikasi dalam studi genom.

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_3)

X_tokenized = tokenizer.texts_to_sequences(X_3)

max_dna = max([len(s.split()) for s in X_3])
print('max DNA length {}'.format(max_dna))

X_padded = pad_sequences(X_tokenized, maxlen=max_dna, padding='post')
print({X_padded.shape[0]})
```

Gambar 4.4 *Code Tokenizer*

Tabel 4.4 Hasil Dari *Tokenizer*

Sebelum *Tokenizer*							
AAGC	AGCA	GCAG	CAGA	AGAT	GATC	ATCT	TCGA
Sesudah *Tokenizer*							
1	2	3	4	5	6	7	8

4.1.4 *Oversamping*

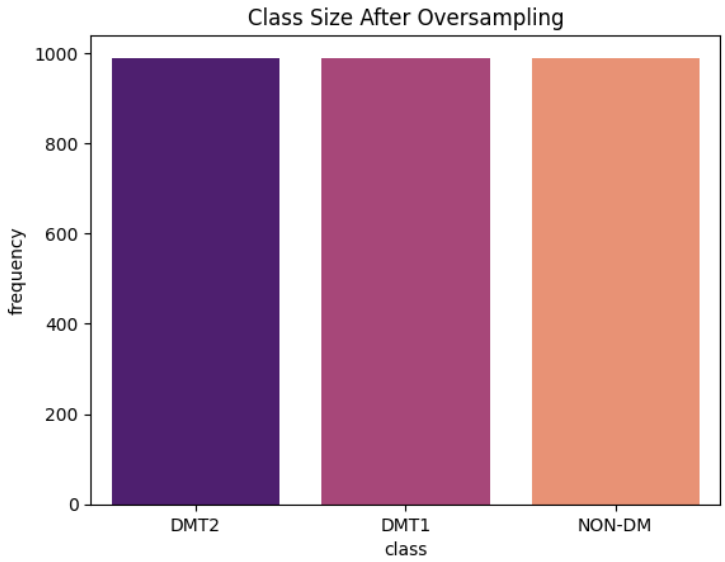
Menurut Tabel 4.2, terlihat bahwa jumlah data sekuens DNA dalam penelitian ini tidak seimbang, yang dapat memengaruhi performa model klasifikasi. Untuk menyeimbangkan data tersebut, digunakan teknik oversampling dengan metode SMOTE (Synthetic Minority Over-sampling Technique), yaitu menyamakan jumlah sampel data dari kelas minoritas dengan cara memperbanyak data hingga sebanding dengan kelas mayoritas.

SMOTE menghasilkan data sintetis baru dengan memperhatikan kesamaan karakteristik dari data kelas minoritas. Pertama, SMOTE memilih sejumlah sampel dari kelas minoritas dalam dataset. Selanjutnya, untuk setiap sampel yang dipilih, SMOTE mencari tetangga terdekatnya dalam ruang fitur. Setelah menemukan tetangga terdekat, SMOTE menghasilkan sampel sintetis

di antara sampel minoritas tersebut dan tetangganya, dengan meletakkan sampel baru ini pada garis yang menghubungkan kedua titik dalam ruang fitur. Sampel sintetis ini kemudian ditambahkan ke dataset untuk menambah data kelas minoritas dan menyeimbangkan jumlahnya. Berikut adalah grafik yang menunjukkan distribusi data pada setiap kelas setelah oversampling dengan SMOTE.

```
from imblearn.over_sampling import SMOTE

smote = SMOTE(random_state=42)

X_os, y_os = smote.fit_resample(X_padded, y)

OSclasses = pd.DataFrame(y_os)
OSclasses.columns = ['classes']

plt.subplots(figsize=(6,4))
sns.countplot(OSclasses, x='classes', palette='inferno')
plt.show()
```

Gambar 4.5 *Code* Frekuensi Data Setelah *Oversamping*



Gambar 4.6 Hasil Oversampling Menggunakan SMOTE

Tabel 4.5 Frekuensi Data Sekuens DNA Sebelum dan Setelah *Oversampling*

Sebelum *Oversampling*		
DMT1	DMT2	NON-DM
145	443	989
Setelah *Oversampling*		
DMT1	DMT2	NON-DM
989	989	989

Setelah melakukan *oversampling*, terlihat pada Gambar 4 bahwa frekuensi semua kelas sekarang telah seimbang. Tabel 4.5 menunjukkan bahwa pada kelas minoritas telah ditambahkan data sintetis baru menggunakan metode SMOTE, sehingga ketiga kelas memiliki jumlah yang sama, yaitu masing-masing 989 data. Dengan demikian, total jumlah data dalam dataset menjadi 2.967.

4.2 Klasifikasi *Support Vector Machine*

4.2.1 Pembagian Data *Training* dan Data *Testing*

Proses klasifikasi diawali dengan membagi data menjadi dua bagian, yaitu data training dan data testing, dengan rasio 90:10. Pemilihan rasio ini didasarkan pada hasil uji coba sebelumnya yang menunjukkan bahwa model yang dilatih dengan perbandingan ini mencapai akurasi tinggi. Data training, yang terdiri dari beberapa variabel serta kelas target, digunakan sebagai sampel untuk membangun model klasifikasi menggunakan algoritma SVM non-linear. Sementara itu, data testing berfungsi untuk menilai akurasi model klasifikasi melalui confusion matrix yang dihasilkan. Dari tabel 4.5 diketahui bahwa jumlah data dalam dataset 2.967, kemudian dibagi dengan rasio 90:10 menghasilkan 2670 sampel untuk pelatihan, 297 sampel untuk pengujian.

4.2.2 Penyelesaian Manual Model SVM

Dalam perhitungan manual, dilakukan klasifikasi SVM menggunakan kernel RBF yang dinyatakan pada Persamaan (2.13), dengan contoh parameter σ (sigma = 10). Pemilihan parameter σ berdasarkan *Cross-validation* memastikan bahwa parameter yang dipilih mampu bekerja dengan baik pada data baru (Nugraha & Sasongko, 2022). Pada contoh ini, sebanyak 3 data dari 2670 data DNA digunakan untuk perhitungan manual, dengan rincian data DNA yang disajikan dalam Tabel 4.6.

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_3)

X_tokenized = tokenizer.texts_to_sequences(X_3)

max_dna = max([len(s.split()) for s in X_3])
print('max DNA length {}'.format(max_dna))

X_padded = pad_sequences(X_tokenized, maxlen=max_dna, padding='post')
print({X_padded.shape[0]})

print(f"X-before tokenized: {X[5]}")
print()
print(f"X-after tokenized: {X_tokenized[5]}")
print()
```

Gambar 4.7 *Code* Untuk Menampilkan Data

Tabel 4.6 Sampel Data Sekuens DNA

No	Sekuens DNA hasil tokenizer	Label
$\boldsymbol{x_1}$	41, 49, 35, 6, 10, 12, 41, 52, 17, 21, 26, 8, 4, 16, 24, 10	DMT2
$\boldsymbol{x_2}$	18, 27, 22, 32, 37, 61, 59, 32, 37, 39, 11, 7, 21, 17, 21, 26	DMT1
$\boldsymbol{x_3}$	41, 52, 26, 7, 20, 31, 63, 59, 14, 55, 64, 52, 26, 11, 7, 21	DMT2
⋮	⋮	⋮
$\boldsymbol{x_{2670}}$	14, 10, 13, 48, 45, 18, 5, 37, 39, 42, 37, 18, 21, 25, 27, 14	DMT2

Langkah awal dalam perhitungan klasifikasi SVM adalah menentukan nilai matriks kernel $\boldsymbol{K}$. Matriks kernel $\boldsymbol{K}$ dihitung dengan ukuran $n \times n$, di mana n adalah jumlah data yang digunakan. Persamaan yang digunakan untuk menghitung kernel $\boldsymbol{K}$ adalah Persamaan (2.13). Pada perhitungan ini, parameter yang digunakan adalah σ = 10, sehingga diperoleh perhitungan sebagai berikut:

$$K(\boldsymbol{x}_i.\boldsymbol{x}_j) = e^{-\frac{\left||x_i - x_j|\right|^2}{2\sigma^2}}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_1) = e^{-\frac{0}{2*10^2}} = 1$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = \left||\boldsymbol{x}_1 - \boldsymbol{x}_2|\right|^2$$

$$= \begin{array}{c}(41-18)^2 + (49-27)^2 + (35-22)^2 + (6-32)^2 + (10-37)^2 \\ +(12-61)^2 + (41-59)^2 + (52-32)^2 + (17-37)^2 + (21-39)^2 \\ +(26-11)^2 + (8-7)^2 + (4-21)^2 + (16-17)^2 \\ +(24-21)^2 + (10-26)^2 \\ = 7217\end{array}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_2) = e^{-\frac{7217}{2*10^2}} = e^{-43,51} \approx 2{,}1305 \times 10^{-19}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_3) = \left||\boldsymbol{x}_1 - \boldsymbol{x}_3|\right|^2$$

$$= \begin{array}{c}(41-41)^2 + (49-52)^2 + (35-26)^2 + (6-7)^2 + (10-20)^2 \\ +(12-31)^2 + (41-63)^2 + (52-59)^2 + (17-14)^2 \\ +(21-55)^2 + (26-64)^2 + (8-52)^2 + (4-26)^2 \\ +(16-11)^2 + (24-7)^2 + (10-21)^2 \\ = 6549\end{array}$$

$$K(\boldsymbol{x}_1.\boldsymbol{x}_3) = e^{-\frac{6549}{2*10^2}} = e^{-3,2745} \approx 6{,}0121 \times 10^{-19}$$

$$K(\boldsymbol{x}_2.\boldsymbol{x}_3) = \left||\boldsymbol{x}_2 - \boldsymbol{x}_3|\right|^2$$

$$= \begin{array}{c}(18-41)^2 + (27-52)^2 + (22-26)^2 + (32-7)^2 + (37-20)^2 \\ +(61-31)^2 + (59-63)^2 + (32-59)^2 + (37-14)^2 \\ +(7-52)^2 + (21-26)^2 + (17-11)^2 + (21-7)^2 + (26-21)^2 \\ +(39-55)^2 + (11-64)^2 \\ = 9630\end{array}$$

$$K(\boldsymbol{x}_2.\boldsymbol{x}_3) = e^{-\frac{9630}{2*10^2}} = e^{-4815} \approx 1{,}2266 \times 10^{-19}$$

Nilai kernel tersebut akan dihitung untuk seluruh data sampel yang ada sehingga didapatkan matriks kernel K yang ditunjukkan sebagai berikut:

$$\boldsymbol{K} = \begin{bmatrix} 1 & 2{,}1305 \times 10^{-19} & 6{,}0121 \times 10^{-19} \\ 2{,}1305 \times 10^{-19} & 1 & 1{,}2266 \times 10^{-19} \\ 6{,}0121 \times 10^{-19} & 1{,}2266 \times 10^{-19} & 1 \end{bmatrix}$$

Dengan menggantikan produk skalar $\phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$ dalam persamaan dualitas Lagrange (2.15) dengan kernel K, persamaan tersebut menjadi:

$$\max_{\alpha} L_{dual} = \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}(\alpha_1^2 \boldsymbol{K}_{11} y_1 y_1 + \alpha_1 \alpha_2 \boldsymbol{K}_{12} y_1 y_2 + \alpha_1 \alpha_3 \boldsymbol{K}_{13} y_1 y_3$$

$$+ \alpha_2^2 \boldsymbol{K}_{22} y_2 y_2 + \alpha_2 \alpha_3 \boldsymbol{K}_{23} y_2 y_3 + \alpha_3^2 \boldsymbol{K}_{33} y_3 y_3)$$

Dengan batasan:

$$\alpha_i \in [0.C]. \text{misalkan } C = 10$$

Pemilihan rentang nilai C untuk SVM adalah 10^{-2}dan 10^4 (Huang dkk., 2007).

$$\sum_{i=1}^{3} \alpha_i y_i = 0 \text{ atau } \alpha_1 - \alpha_2 + \alpha_3 = 0$$

$$\alpha_2 + \alpha_3 = \alpha_1$$

$$\max_{\alpha} L_{dual} = \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}(\alpha_1^2 + (-2{,}1305 \times 10^{-19})\alpha_1 \alpha_2$$

$$+ (6{,}0121 \times 10^{-19})\alpha_1 \alpha_3 + \alpha_2^2 - (1{,}2266 \times 10^{-19})\alpha_2 \alpha_3 + \alpha_3^2)$$

Hitung turunan dari lagrangian:

$$\frac{\partial L}{\partial \alpha_1} = 1 - \alpha_1 - (2{,}1305 \times 10^{-19})\alpha_2 + (6{,}0121 \times 10^{-19})\alpha_3 = 0$$

$$\alpha_1 = 1 - (2{,}1305 \times 10^{-19})\alpha_2 + (6{,}0121 \times 10^{-19})\alpha_3$$

$$\frac{\partial L}{\partial \alpha_2} = 1 - \alpha_2 - (2{,}1305 \times 10^{-19})\alpha_1 - (1{,}2266 \times 10^{-19})\alpha_3 = 0$$

$$\alpha_2 = 1 - (2{,}1305 \times 10^{-19})\alpha_1 - (1{,}2266 \times 10^{-19})\alpha_3$$

$$\frac{\partial L}{\partial \alpha_3} = 1 - \alpha_3 - (6{,}0121 \times 10^{-19})\alpha_1 + (1{,}2266 \times 10^{-19})\alpha_2 = 0$$

$$\alpha_3 = 1 - (6{,}0121 \times 10^{-19})\alpha_1 + (1{,}2266 \times 10^{-19})\alpha_2$$

Dengan menggantikan nilai-nilai α_1, α_2 dan α_3 secara bertahap ke dalam persamaan lainnya, kita dapat menemukan solusi untuk ketiga variabel tersebut. Kemudian untuk mencari nilai dari fungsi *Lagrange Multiplier* akan dilakukan dengan bantuan bahasa pemrograman python pada *google collab* dan dihasilkan $\alpha_1 = 0{,}66666666. \alpha_2 = 1{,}33333332 \; dan \; \alpha_3 = 0{,}66666666$

Setelah nilai parameter $\alpha_1. \alpha_2 \; dan \; \alpha_3$ ditemukan, langkah selanjutnya adalah mencari nilai bias b untuk mendapatkan hasil yang diinginkan.

$$b = y_i - \sum_{i=1}^{n} y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

$$b = y_1 - y_1 \alpha_1 K(\boldsymbol{x}_1, \boldsymbol{x}_1) + y_2 a_2 K(\boldsymbol{x}_1, \boldsymbol{x}_2) + y_3 a_3 K(\boldsymbol{x}_1, \boldsymbol{x}_3)$$

$$b = 1 - 1 \times 0.66666666 \times 1 + (-1) \times 1.333333332 \times 2{,}1305 \times 10^{-19}$$

$$+ 1 \times 0{,}6666666 \times 6{,}0121 \times 10^{-19}$$

Karena $2{,}1305 \times 10^{-19}$ dan $6{,}0121 \times 10^{-19}$ sangat kecil dibandingkan dengan $0{,}6666666$, didapatkan nilai untuk perhitungan

$$b \approx 1 - 0{,}66666666$$

$$b \approx 0{,}33333334$$

Selanjutnya, setelah nilai parameter α dan b ditemukan maka proses klasifikasi dengan menggunakan algoritma SVM akan dapat dilakukan dengan memasukkan nilai parameter pada persamaan:

$$f(\boldsymbol{x}) = \sum_{i.j=1}^{n} y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) + b$$

$$g(\boldsymbol{x}) = sign(f(\boldsymbol{x}))$$

1. Hitung untuk $\boldsymbol{x}_1$

$$f(\boldsymbol{x}_1) = y_1\alpha_1 K(\boldsymbol{x}_1, \boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_1, \boldsymbol{x}_2) + y_3\alpha_3 K(\boldsymbol{x}_1, \boldsymbol{x}_3) + b$$

$$= 0{,}6666666 \times 1 \times 1 + 1{,}333333332 \times -1 \times (2{,}1305 \times 10^{-19})$$

$$+ 0{,}6666666 \times 1 \times 6{,}0121 \times 10^{-19} + 0{,}33333334$$

$$= 0{,}999994$$

2. Hitung untuk $\boldsymbol{x}_2$

$$f(\boldsymbol{x}_2) = y_1\alpha_1 K(\boldsymbol{x}_2, \boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_2, \boldsymbol{x}_2) + y_3\alpha_3 K(\boldsymbol{x}_2, \boldsymbol{x}_3) + b$$

$$= 0{,}6666666 \times 1 \times (2{,}1305 \times 10^{-19}) + 1{,}333333332 \times -1 \times 1$$

$$+ 0{,}6666666 \times 1 \times (1{,}2266 \times 10^{-19}) + 0{,}33333334$$

$$= -0{,}9999999$$

3. Hitung untuk $\boldsymbol{x}_3$

$$f(\boldsymbol{x}_3) = y_1\alpha_1 K(\boldsymbol{x}_3, \boldsymbol{x}_1) + y_2\alpha_2 K(\boldsymbol{x}_3, \boldsymbol{x}_2) + y_3\alpha_3 K(\boldsymbol{x}_3, \boldsymbol{x}_3) + b$$

$$= 0{,}6666666 \times 1 \times (2{,}1305 \times 10^{-19})$$

$$+ 1{,}333333332 \times -1 \times (1{,}2266 \times 10^{-19})$$

$$+ 0{,}6666666 \times 1 \times 1 + 0{,}33333334$$

$$= 0{,}999999$$

Berdasarkan perhitungan manual menggunakan algoritma SVM di atas, sampel data diklasifikasikan ke dalam kelas positif (+1) untuk data $\boldsymbol{x}_1$ dan $\boldsymbol{x}_3$, serta kelas negatif (-1) untuk data $\boldsymbol{x}_2$. Proses perhitungan tersebut akan diterapkan pada seluruh data pelatihan guna memperoleh model klasifikasi SVM yang optimal dalam mengklasifikasikan sekuens DNA manusia pada penderita dan non-penderita diabetes mellitus.

4.2.3 Hasil Klasifikasi SVM

Penelitian ini bertujuan untuk melakukan klasifikasi menggunakan Support Vector Machine (SVM) non-linier dan mencari nilai akurasi terbaik. SVM akan diuji menggunakan tiga jenis fungsi kernel: RBF, sigmoid, dan polinomial, dengan derajat (degree) untuk kernel polinomial yang bervariasi, yaitu 3, 4, dan 5. Parameter C yang akan diuji terdiri dari 0.01; 0.1; 1; 10; dan 100, sesuai dengan yang direkomendasikan oleh penelitian (Huang dkk., 2007).

Proses uji coba ini melibatkan pembentukan model SVM menggunakan data pelatihan (*training* data), yang kemudian diuji untuk melihat nilai akurasinya pada data pengujian (*testing* data). Setiap kernel diuji menggunakan lima nilai parameter C berbeda, dan hasil akurasi untuk kernel RBF, sigmoid, dan polinomial akan dicatat pada tabel 4.7 berikut.

```
for kernel in kernels:
    for C in C_values:
        svm = SVC(kernel=kernel, C=C)
        svm.fit(X_train, y_train)
        # Predict and evaluate
        y_pred = svm.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        report = classification_report(y_test, y_pred)
        # Print results
        print(f'Kernel: {kernel}, C={C}')
        print(f'Accuracy: {accuracy:.2f}')
        print(f'Classification Report:\n{report}')
        print('-' * 60)
C_values = [0.01, 0.1, 1, 10, 100]
kernels = ['rbf', 'poly', 'sigmoid']
print("Processing and training for 3-mers...")
process_and_train_svm(data, k=3, kernels=kernels, C_values=C_values)
print("Processing and training for 4-mers...")
process_and_train_svm(data, k=4, kernels=kernels, C_values=C_values)
print("\nProcessing and training for 5-mers...")
process_and_train_svm(data, k=5, kernels=kernels, C_values=C_values)
```

Gambar 4.8 *Code* Hasil Akurasi

Tabel 4.7 Hasil Akurasi Model Support Vector Machine

k-mers	Kernel	$C = 0{,}01$	$C = 0{,}1$	$C = 1$	$C = 10$	$C = 100$
3-*mers*	RBF	52%	68%	92%	95%	95%
3-*mers*	Polinomial	53%	74%	92%	94%	93%
3-*mers*	sigmoid	45%	57%	79%	84%	76%
4-*mers*	RBF	53%	67%	93%	96%	95%
4-*mers*	Polinomial	52%	75%	91%	95%	94%
4-*mers*	sigmoid	47%	58%	84%	86%	77%
5-*mers*	RBF	53%	70%	92%	95%	95%
5-*mers*	Polinomial	52%	73%	93%	93%	93%
5-*mers*	sigmoid	47%	59%	85%	83%	78%

Hasil uji coba menunjukkan berbagai nilai akurasi dari beberapa percobaan menggunakan kernel RBF, sigmoid, dan polinomial. Berdasarkan Tabel 4.7, akurasi tertinggi dicapai oleh kernel RBF dengan 4-mers dan parameter ($C = 10$), menghasilkan akurasi tertinggi sebesar 96%. Model ini dianggap sangat baik, karena nilai parameter cost (C) yang kecil cenderung menghasilkan margin yang lebih lebar dengan mengabaikan data yang kurang sesuai pada data pelatihan, sementara nilai cost (C) yang besar lebih menyesuaikan data pelatihan, sehingga dapat mengurangi kesalahan klasifikasi.

4.3 Evaluasi Model SVM

Evaluasi akurasi model klasifikasi dapat dilakukan dengan menggunakan *K-Fold Cross Validation*, dengan k = 10 *fold*. Dalam metode ini, setiap bagian data akan berperan sebagai data uji satu kali, sementara data sisanya digunakan untuk data latih. Dengan demikian, model klasifikasi akan dilatih dan diuji sebanyak 10 kali. Hasil akurasi, presisi, dan recall yang diperoleh melalui 10-*Fold Cross Validation* dapat dilihat pada tabel 4.8.

```
for train_index, test_index in skf.split(X_os, y_os):
    X_train, X_test = X_os[train_index], X_os[test_index]
    y_train, y_test = y_os[train_index], y_os[test_index]

    # Train SVM with specified kernel and C
    svm = SVC(kernel=kernel, C=C)
    svm.fit(X_train, y_train)

    # Predict and evaluate
    y_pred = svm.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    report = classification_report(y_test, y_pred)
    cm = confusion_matrix(y_test, y_pred)

    # Print results for each fold
    print(f"\nResults for Fold {fold}, {k}-mers, Kernel: {kernel}, C={C}")
    print(f'Accuracy: {accuracy:.2f}')
    print(f'Classification Report:\n{report}')
    print(f'Confusion Matrix:\n{cm}')
    accuracies.append(accuracy)
    fold += 1

# Print overall accuracy
print(f"\nAverage Accuracy over 10 folds: {np.mean(accuracies):.2f}")
```

Gambar 4.9 *Code Croos Validation*

Tabel 4.8 Hasil 10-*Fold Validation*

Parameter *k*	Presisi	*Recall*	Akurasi
k = 1	96%	96%	96%
k = 2	94%	94%	94%
k = 3	94%	94%	94%
k = 4	92%	92%	91%
k = 5	91%	91%	91%
k = 6	94%	94%	94%
k = 7	96%	96%	96%
k = 8	94%	94%	94%
k = 9	95%	95%	95%
k = 10	94%	94%	94%
Rata-rata	94%	94%	94%

Tabel 4.9 Perbandingan Dengan Penelitian Terdahulu

Metode & Autor	Dataset	Akurasi	Presisi	Recall
SVM (Hamed dkk., 2023)	Panjang Sekuens DNA lebih dari 12 juta	96%	91%	94%
SVM (Mahesh, 2020)	Sekuens DNA	98.4%	93.7%	92.3%
SVM (Al Kindhi dkk., 2018)	DNA Hepatitis C	99%	100%	100%
SVM (Penelitian ini)	Sekuens DNA manusia DMT1, DMT2 dan Non DM	94%	94%	94%

Berdasarkan Tabel 4.8, diperoleh rata-rata presisi, *recall* dan akurasi sebesar 94%. Tabel 4.9 menunjukan beberapa hasil dari penelitian terdahulu. Sehingga rata-rata 94% sudah tergolong sangat kompetitif, dikarenakan pada penelitian ini memiliki hasil presisi dan *recall* lebih tinggi dibandingkan dengan penelitian (Hamed dkk., 2023) dan penelitian (Mahesh, 2020). Hasil akurasi terbaik dimiliki oleh penelitian (Al Kindhi dkk., 2018), kemudian presisi dan *recall* penelitian ini lebih baik dari penelitian (Hamed dkk., 2023) dan penelitian (Mahesh, 2020).

4.4 Analisis Performa Model Klasifikasi

Klasifikasi pada seluruh data sekuens DNA manusia, baik penderita maupun non-penderita diabetes mellitus, akan dilakukan menggunakan algoritma *Support Vector Machine* (SVM) dalam bahasa pemrograman Python pada platform *notebook Kaggle*. Algoritma SVM yang diterapkan bersifat non-linier dengan parameter $C = 10$, *k-mer* = 4 dan kernel tipe *Radial Basis Function* (RBF). Setelah model SVM terbentuk, evaluasi performa dilakukan menggunakan Confusion Matrix, dan hasil evaluasi performa model ini dianalisis melalui metrik-metrik berikut:

Tabel 4.10 Hasil Confusion Matrix Data Sekuens DNA

Confusion Matrix		Prediksi		
		DMT1	DMT2	NONDM
Aktual	DMT1	99	0	0
	DMT2	5	88	6
	NONDM	0	1	98

Berdasarkan tabel dari *confusion matrix* menunjukan bahwa 99 data DNA diabetes tipe 1 diprediksi benar sebagai diabetes tipe 1, 5 data DNA diabetes tipe 2 diprediksi salah sebagai diabetes tipe 1, 88 data DNA diabetes tipe 2 diprediksi benar benar sebagai diabetes tipe 2, 6 data DNA diabetes tipe 2 diprediksi salah sebagai non-diabetes, 98 data DNA non-diabetes diprediksi benar sebagai non-diabetes, dan 1 data DNA non-diabetes diprediksi salah sebagai diabetes tipe 2. Kemudian, dari hasil prediksi diatas didapatkan nilai performa sebagai berikut:

1. Akurasi

$$\text{Akurasi} = \frac{99 + 88 + 98}{99 + 5 + 88 + 6 + 1 + 98} * 100\%$$

$$= 96\%$$

2. Presisi

$$\text{Presisi}_{\text{DMT1}} = \frac{TP_{DMT1}}{TP_{DMT1} + FP_{DMT1}} = \frac{99}{99 + 5} * 100\% = 95\%$$

$$\text{Presisi}_{\text{DMT2}} = \frac{TP_{DMT2}}{TP_{DMT2} + FP_{DMT2}} = \frac{88}{88 + 1} * 100\% = 99\%$$

$$\text{Presisi}_{\text{NONDM}} = \frac{TP_{\text{NONDM}}}{TP_{\text{NONDM}} + FP_{\text{NONDM}}} = \frac{98}{98 + 6} * 100\% = 94.23\%$$

$$\text{Presisi} = \frac{\text{Presisi}_{\text{DMT1}} + \text{Presisi}_{\text{DMT2}} + \text{Presisi}_{\text{NONDM}}}{3} = 96\%$$

3. Recall

$$\text{Recall}_{\text{DMT1}} = \frac{TP_{DMT1}}{TP_{DMT1} + FN_{DMT1}} = \frac{99}{99} * 100\% = 100\%$$

$$\text{Recall}_{\text{DMT2}} = \frac{TP_{DMT2}}{TP_{DMT2} + FN_{DMT2}} = \frac{88}{88 + 5 + 6} * 100\% = 89\%$$

$$\text{Recall}_{\text{NONDM}} = \frac{TP_{\text{NONDM}}}{TP_{\text{NONDM}} + FN_{\text{NONDM}}} = \frac{98}{98 + 1} * 100\%$$

$$= 98.98\%$$

$$\text{Recall} = \frac{\text{Recall}_{\text{DMT1}} + \text{Recall}_{\text{DMT2}} + \text{Recall}_{\text{NONDM}}}{3} = 96\%$$

4. F1-Score

$$\text{F1}_{\text{DMT1}} = \frac{2 * \text{Presisi}_{\text{DMT1}} * \text{Recall}_{DMT1}}{\text{Presisi}_{\text{DMT1}} + \text{Recall}_{DMT1}}$$

$$= \frac{2 * 0.95 * 1}{0.95 + 1} * 100\% = 98\%$$

$$\text{F1}_{\text{DMT2}} = \frac{2 * \text{Presisi}_{\text{DMT2}} * \text{Recall}_{DMT2}}{\text{Presisi}_{\text{DMT2}} + \text{Recall}_{DMT2}}$$

$$= \frac{2 * 0.99 * 0.89}{0.99 + 0.89} \text{x} 100\% = 94\%$$

$$\text{F1}_{\text{NONDM}} = \frac{2 * \text{Presisi}_{\text{NONDM}} * \text{Recall}_{\text{NONDM}}}{\text{Presisi}_{\text{NONDM}} + \text{Recall}_{\text{NONDM}}}$$

$$= \frac{2 * 0.9423 * 0.99}{0.9423 + 0.99} * 100\% = 97\%$$

$$\text{F1} = \frac{\text{F1}_{\text{DMT1}} + \text{F1}_{\text{DMT2}} + \text{F1}_{\text{NONDM}}}{3} = 96\%$$

Berdasarkan hasil perhitungan tersebut, model klasifikasi sekuens DNA manusia untuk penderita dan non-penderita diabetes mellitus dengan algoritma SVM berhasil mencapai akurasi prediksi sebesar 96%. Hal ini menunjukkan bahwa SVM mampu mengklasifikasikan sekuens DNA manusia dengan sangat baik. Selanjutnya, rata-rata nilai presisi untuk ketiga kelas tercatat sebesar 96%, yang berarti dari total 297 sekuens DNA, sebanyak 285 sekuens berhasil diklasifikasikan secara akurat sesuai kelasnya. Selain itu, nilai recall rata-rata untuk ketiga kelas

sebesar 96%, yang menunjukkan bahwa dari 297 sekuens DNA, sebanyak 285 sekuens dapat diidentifikasi dengan tepat oleh model. Rata-rata F1-Score yang diperoleh adalah 96%, yang mengindikasikan bahwa model klasifikasi dengan algoritma SVM ini mampu menjaga keseimbangan antara presisi dan recall.

4.5 Kajian Islam

Diabetes mellitus, atau sering disebut dengan penyakit kencing manis, merupakan penyakit kronis yang ditandai dengan tingginya kadar gula dalam darah. Penyakit ini dapat menyebabkan berbagai komplikasi serius, seperti kerusakan saraf, gangguan pada penglihatan, penyakit jantung, serta gangguan ginjal. Pola hidup yang sehat, seperti menjaga pola makan dan berolahraga secara teratur, sangat penting untuk mencegah dan mengendalikan diabetes. Dalam ajaran Islam, kesehatan adalah karunia Allah yang harus dijaga dan disyukuri, sebagaimana Allah SWT berfirman dalam Al-Qur'an, "Dan janganlah kamu menjatuhkan dirimu sendiri ke dalam kebinasaan" (QS. Al-Baqarah: 195). Ayat ini mengingatkan kita untuk menjaga kesehatan sebagai bentuk tanggung jawab atas tubuh yang telah Allah amanahkan kepada kita.

Penelitian tentang klasifikasi penyakit diabetes mellitus menggunakan metode SVM menunjukkan bahwa metode ini mampu memberikan hasil yang akurat dalam mendiagnosis penyakit tersebut, dengan tingkat akurasi mencapai 94%. Hal ini sejalan dengan hadis yang diriwayatkan oleh Abu Hurairah RA, bahwa Rasulullah SAW bersabda, "Sesungguhnya Allah tidak menurunkan penyakit kecuali Dia juga menurunkan penawarnya." Hadis ini menunjukkan bahwa dengan ilmu pengetahuan dan teknologi, manusia dapat mencari solusi dan penanganan

yang lebih baik terhadap penyakit.

Kemampuan metode SVM dalam mengklasifikasikan diabetes mellitus diharapkan dapat membantu tenaga medis dalam memberikan diagnosis yang lebih akurat dan cepat, sehingga memungkinkan pasien mendapatkan penanganan yang tepat. Penelitian ini menunjukkan bahwa ilmu pengetahuan modern, terutama di bidang kecerdasan buatan, dapat memberikan kontribusi positif dalam meningkatkan layanan kesehatan. Manfaatnya sangat besar, tidak hanya dalam meningkatkan kualitas hidup pasien, tetapi juga dalam mengurangi angka komplikasi dan kematian akibat diabetes. Semoga penelitian ini dapat dikembangkan lebih lanjut dan menjadi manfaat bagi masyarakat dalam mencegah dan mengelola diabetes sesuai dengan prinsip Islam untuk menjaga kesehatan dan kebugaran tubuh sebagai bentuk rasa syukur kepada Allah SWT.

BAB V

KESIMPULAN

5.1 Kesimpulan

Berdasarkan rumusan masalah dan pembahasan pada bab sebelumnya, kesimpulan yang diperoleh adalah sebagai berikut:

1. Metode SVM dengan kernel RBF memberikan hasil klasifikasi terbaik dalam mendiagnosis penyakit diabetes. Model ini menggunakan parameter $C = 10$ dan k-mers sebesar 4. Berdasarkan analisis menggunakan confusion matrix, hasilnya menunjukkan bahwa 99 data DNA pasien diabetes tipe 1 berhasil diprediksi dengan benar sebagai diabetes tipe 1. Terdapat 11 data DNA pasien diabetes tipe 2 yang diprediksi salah sebagai DNA diabetes tipe 1 dan non-diabetes, sementara 88 data DNA pasien diabetes tipe 2 berhasil diprediksi dengan benar sebagai diabetes tipe 2. Selain itu, terdapat 1 data DNA pasien non-diabetes yang diprediksi salah sebagai diabetes tipe 2, dan 98 data DNA pasien non-diabetes diprediksi benar sebagai non-diabetes.
2. Tingkat akurasi model dalam mendiagnosis penyakit diabetes menggunakan SVM dengan kernel RBF, Rata-rata mencapai 94% untuk akurasi, presisi dan *recall*. Hasil akurasi terbaik dimiliki oleh penelitian (Al Kindhi dkk., 2018), kemudian presisi dan *recall* penelitian ini lebih baik dari penelitian (Hamed dkk., 2023) dan penelitian (Mahesh, 2020). Hasil ini menunjukkan bahwa metode SVM memiliki kemampuan yang baik dalam klasifikasi diagnosis penyakit diabetes. Berdasarkan penelitian ini, SVM dapat menjadi alternatif model klasifikasi yang digunakan untuk mendiagnosis penyakit diabetes.

5.2 Saran

Berikut adalah beberapa saran yang dapat menjadi acuan untuk penyempurnaan penelitian di masa mendatang:

1. Disarankan untuk menguji model ini pada dataset DNA yang lebih beragam, termasuk data dari pasien dengan jenis diabetes lainnya atau dari latar belakang genetik yang berbeda. Langkah ini akan membantu mengukur sejauh mana model dapat digeneralisasi dalam berbagai kondisi dan memperluas penggunaan model SVM untuk klasifikasi penyakit lainnya dalam bidang medis.
2. Apabila model klasifikasi menunjukkan hasil yang konsisten dan akurat, model ini berpotensi menjadi dasar dalam pengembangan aplikasi klinis untuk mendukung diagnosis dini diabetes berbasis data DNA. Namun, perlu dilakukan pengujian lebih lanjut dengan data berukuran lebih besar serta evaluasi terhadap aspek keamanan, privasi, dan efisiensi pemrosesan data agar aplikasi ini dapat digunakan secara *real-time*.

DAFTAR PUSTAKA

Agus, R. (2018). Dasar-Dasar Biologi Molekuler: Basics of Molecular Biology (IND SUB) . *CELEBES MEDIA PERKASA.*, *Vol.1*.

Akbar, A. F. R., Rifaannudin, M., Badrun, M., & Nuraini, N. M. (2023). Nafs Wahidah Dalam Al-Qur'an Al-Karim Menurut Zaghlul Raghib Muhammad An-Najjar. *ZAD Al-Mufassirin*, *5*(1), 60–77. https://doi.org/10.55759/zam.v5i1.70

Akkaya, U. M., & Kalkan, H. (2021). Classification of DNA Sequences with k-mers Based Vector Representations. *Proceedings - 2021 Innovations in Intelligent Systems and Applications Conference, ASYU 2021*. https://doi.org/10.1109/ASYU52992.2021.9599084

Al Kindhi, B., Sardjono, T. A., & Purnomo, M. H. (2018). Optimasi Support Vector Machine untuk Memprediksi Adanya Mutasi pada DNA Hepatitis C Virus. *Jurnal Nasional Teknik Elektro Dan Teknologi Informasi (JNTETI)*, *7*(3). https://doi.org/10.22146/jnteti.v7i3.441

Anang, K., & Mike, P. (2022). *Klasifikasi Kondisi Finansial pada Perusahaan Sektor Finansial yang Terdaftar di BEI Periode Tahun 2020 Menggunakan Support Vector Machine*.

Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., & Ridella, S. (2022). *The "K" in K-fold Cross Validation*. http://www.i6doc.com/en/livre/?GCOI=28001100967420.

Awad, M., & Khanna, R. (2015). Efficient learning machines: Theories, concepts, and applications for engineers and system designers. In *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Apress Media LLC. https://doi.org/10.1007/978-1-4302-5990-9

Choiriyati, N., Arkeman, Y., & Kusuma, W. A. (2020). Deep learning model for metagenome fragment classification using spaced k-mers feature extraction. *Jurnal Teknologi Dan Sistem Komputer*, *8*(3), 234–238. https://doi.org/10.14710/jtsiskom.2020.13407

Cortes, C., Vapnik, V., & Saitta, L. (1995). Support-Vector Networks Editor. In *Machine Leaming* (Vol. 20). Kluwer Academic Publishers.

El Naqa, I., & Murphy, M. J. (2015). What Is Machine Learning? In *Machine Learning in Radiation Oncology* (pp. 3–11). Springer International Publishing. https://doi.org/10.1007/978-3-319-18305-3_1

El-Attar, N. E., Moustafa, B. M., & Awad, W. A. (2022). Deep learning model to detect diabetes mellitus based on dna sequence. *Intelligent Automation and*

Soft Computing, *31*(1), 325–338. https://doi.org/10.32604/IASC.2022.019970

Ghaddar, B., & Naoum-Sawaya, J. (2018). High dimensional data classification and feature selection using support vector machines. *European Journal of Operational Research*, *265*(3), 993–1004. https://doi.org/10.1016/j.ejor.2017.08.040

Hadits riwayat Al-Bukhari no. 5678 dari Abu Hurairah

Hamed, B. A., Ibrahim, O. A. S., & Abd El-Hafeez, T. (2023). Optimizing classification efficiency with machine learning techniques for pattern matching. *Journal of Big Data*, *10*(1). https://doi.org/10.1186/s40537-023-00804-6

Han, J., kamber, micheline, & pei, jian. (2012). *Data Mining Third Edition*.

Huang, C. M., Lee, Y. J., Lin, D. K. J., & Huang, S. Y. (2007). Model selection for support vector machines via uniform design. *Computational Statistics and Data Analysis*, *52*(1), 335–346. https://doi.org/10.1016/j.csda.2007.02.013

Indah, M., Pengaturan, S. :, & Gen, E. (2007). *PENGATURAN EKSPRESI GEN Dr. MUTIARA INDAH SARI NIP: 132 296 973 2007*.

Kemenag. (2024a). *Qur'an Kemenag.* https://quran.kemenag.go.id/quran/per-ayat/surah/2?from=195&to=195

Kemenag. (2024b). *Qur'an Kemenag.* https://quran.kemenag.go.id/quran/per-ayat/surah/24?from=45&to=45

Lathifah, N. L. (2017). *HUBUNGAN DURASI PENYAKIT DAN KADAR GULA DARAH DENGAN KELUHAN SUBYEKTIF PENDERITA DIABETES MELITUS The Relationship Between Duration Disease and Glucose Blood Related to Subjective Compliance in Diabetes Mellitus*. https://doi.org/10.20473/jbe.v5i2.2017.231-239

M. R. Adrian1, M. P. Putra2, M. H. Rafialdy3, & N. A. Rakhmawati4. (2021). *Perbandingan Metode Klasifikasi Random Forest dan SVM Pada Analisis Sentimen PSBB 7099-25292-1-PB.*

Mahesh, B. (2020). Machine Learning Algorithms - A Review. *International Journal of Science and Research (IJSR)*, *9*(1), 381–386. https://doi.org/10.21275/art20203995

Maulana Al-Fakih, M., Lammada Teknik Elektro, I., Singaperbangsa Karawang Jl Ronggo Waluyo, U. H., Timur, T., & Barat, J. (2024). ANALISIS PERBANDINGAN KINERJA ALGORITMA MLPCLASSIFIER DAN DECISION TREE CLASSIFICATION PADA SISTEM INSPEKSI ALAT KESEHATAN. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Issue 4).

Muhammad, H. F. L., Sulistyoningrum, D. C., Kusuma, R. J., Dewi, A. L., & Permatasari, I. K. (2021). *Buku Ajar Nutrigenomik dan Nutrigenetik Bagi Mahasiswa Gizi*. UGM PRESS.

Munasatya, N., & Novianto, S. (2020). Natural Language Processing untuk Analisis Sentimen Presiden Jokowi Menggunakan Multi Layer Perceptron Natural Language Processing for President Jokowi Sentiment Analysis using Multi Layer Perceptron. In *Agustus* (Vol. 19, Issue 3). https://t.co/dV56DeVJSA

Nasution, F., Azwar Siregar, A., & Tinggi Kesehatan Indah Medan, S. (2021). FAKTOR RISIKO KEJADIAN DIABETES MELLITUS (Risk Factors for The Event of Diabetes Mellitus). *Jurnal Ilmu Kesehatan*, *9*(2).

Nolly, R. A., Fitria, A., & Saputra S, K. (2023). Penerapan Algoritma K-Nearest Neighbors untuk Klasifikasi Fragmen Metagenom Berdasarkan Ekstraksi Fitur K-Mers. *Informatika Mulawarman : Jurnal Ilmiah Ilmu Komputer*, *17*(1), 52. https://doi.org/10.30872/jim.v17i1.5779

Nugraha, W., & Sasongko, A. (2022). *SISTEMASI: Jurnal Sistem Informasi Hyperparameter Tuning pada Algoritma Klasifikasi dengan Grid Search Hyperparameter Tuning on Classification Algorithm with Grid Search* (Vol. 11, Issue 2). http://sistemasi.ftik.unisi.ac.id

Putra, R. F., Zebua, R. S. Y., & Budiman, B. (2023). Data Mining: Algoritma dan Penerapannya. . *PT. Sonpedia Publishing Indonesia.*

Rumpun, J., Kesehatan, I., Jurnal, H., Enikmawati, A., Mar'atus Sholihah, A., Sarifah, S., Diii, P., & Surakarta, M. (2022). *PENGARUH KULIT MANGGIS TERHADAP PENURUNAN KADAR GULA DARAH PADA PENDERITA DIABETES MELLITUS*. *2*(2).

Saptadi, I. N. T. S. K., Umalihayati, S., & MT, M. , M. Z. (2024). *DATA MINING*. Cendikia Mulia Mandiri.

Sari, N., & Purnama, A. (2019). *Aktivitas Fisik dan Hubungannya dengan Kejadian Diabetes Melitus*.

Sutoyo, E., Asri Fadlurrahman, M., Telekomunikasi Jl Terusan Buah Batu, J., Dayeuhkolot, K., Bandung, K., & Barat, J. (2020). *JEPIN (Jurnal Edukasi dan Penelitian Informatika) Penerapan SMOTE untuk Mengatasi Imbalance Class dalam Klasifikasi Television Advertisement Performance Rating Menggunakan Artificial Neural Network*.

Whiting, D. R., Guariguata, L., Weil, C., & Shaw, J. (2011). IDF Diabetes Atlas: Global estimates of the prevalence of diabetes for 2011 and 2030. *Diabetes Research and Clinical Practice*, *94*(3), 311–321. https://doi.org/10.1016/j.diabres.2011.10.029

Widyanto, R. M., Muslihah, N. , Rahmawati, I. S., Raras, T. Y. M., Dini, C. Y., & Maulidiana, A. R. (2021). Gizi Molekuler. *Universitas Brawijaya Press.*

Zaki, M. J., & JR. Wagner Meira. (2020). *Data Mining and Machine Learning, Fundamental Concepts and Algorithms*. Cambridge University Press.

Zhang, J., Hung, G.-C., Nagamine, K., Li, B., Tsai, S., & Lo, S.-C. (2016). Development of Candida -Specific Real-Time PCR Assays for the Detection and Identification of Eight Medically Important Candida Species . *Microbiology Insights*, *9*, MBI.S38517. https://doi.org/10.4137/mbi.s38517

LAMPIRAN

Lampiran 1 Hasil Klasifikasi Sekuens DNA Manusia Penderita Diabetes

Actual	Predicted	Correct
1	1	True
2	2	True
1	1	True
2	2	True
2	2	True
0	0	True
0	0	True
2	2	True
2	2	True
0	0	True
0	0	True
2	2	True
0	0	True
2	2	True
1	2	False
0	0	True
2	2	True
1	1	True
2	2	True
1	1	True
2	2	True
1	1	True
2	2	True
2	2	True
1	1	True
1	1	True
2	2	True
0	0	True
0	0	True
0	0	True
1	1	True
2	2	True
0	0	True
0	0	True
0	0	True
1	1	True
2	2	True
1	1	True
1	1	True
2	2	True

0	0	True
0	0	True
0	0	True
0	0	True
0	0	True
0	0	True
2	2	True
0	0	True
0	0	True
2	2	True
0	0	True
1	1	True
2	2	True
0	0	True
1	1	True
2	2	True
0	0	True
2	2	True
0	0	True
2	2	True
1	1	True
1	1	True
0	0	True
2	2	True
1	1	True
1	1	True
2	2	True
2	2	True
2	2	True
2	2	True
1	1	True
2	2	True
2	2	True
2	2	True
0	0	True
0	0	True
0	0	True
1	1	True
0	0	True
1	1	True
0	0	True
1	0	False
1	1	True
1	1	True
0	0	True
1	0	False

2	2	True
0	0	True
1	1	True
2	2	True
2	2	True
2	2	True
1	1	True
1	1	True
2	2	True
2	2	True
2	2	True
2	2	True
0	0	True
1	1	True
2	2	True
1	1	True
1	1	True
0	0	True
0	0	True
2	2	True
0	0	True
0	0	True
0	0	True
2	2	True
2	2	True
2	2	True
2	2	True
0	0	True
2	2	True
2	2	True
1	1	True
0	0	True
1	1	True
0	0	True
2	2	True
2	2	True
2	2	True
0	0	True
1	1	True
1	2	False
0	0	True
0	0	True
0	0	True
0	0	True
0	0	True
0	0	True

0	0	True
0	0	True
2	2	True
1	1	True
2	2	True
1	1	True
2	2	True
2	2	True
0	0	True
0	0	True
2	2	True
0	0	True
1	1	True
0	0	True
1	1	True
0	0	True
0	0	True
0	0	True
1	1	True
0	0	True
0	0	True
1	1	True
1	1	True
1	1	True
1	1	True
2	2	True
1	1	True
1	2	False
1	1	True
0	0	True
0	0	True
1	1	True
1	1	True
1	1	True
2	2	True
1	0	False
2	2	True
1	1	True
2	2	True
2	2	True
1	1	True
0	0	True
1	1	True
2	2	True
0	0	True
1	1	True

2	2	True
1	1	True
1	1	True
1	1	True
1	1	True
1	1	True
2	2	True
1	1	True
0	0	True
1	1	True
2	2	True
2	2	True
1	1	True
1	1	True
2	2	True
0	0	True
0	0	True
2	2	True
0	0	True
0	0	True
2	2	True
2	2	True
1	1	True
1	1	True
1	1	True
1	1	True
1	1	True
0	0	True
2	2	True
2	2	True
0	0	True
0	0	True
1	1	True
2	2	True
2	2	True
2	2	True
0	0	True
0	0	True
0	0	True
1	1	True
2	2	True
0	0	True
0	0	True
2	2	True
1	1	True
1	1	True

0	0	True
0	0	True
0	0	True
2	2	True
2	2	True
1	1	True
0	0	True
2	2	True
1	1	True
2	1	False
2	2	True
1	1	True
1	1	True
2	2	True
2	2	True
0	0	True
0	0	True
1	1	True
0	0	True
1	1	True
0	0	True
0	0	True
0	0	True
1	1	True
0	0	True
0	0	True
2	2	True
0	0	True
2	2	True
2	2	True
0	0	True
2	2	True
1	0	False
2	2	True
0	0	True
1	1	True
1	1	True
0	0	True
1	1	True
2	2	True
2	2	True
2	2	True
2	2	True
2	2	True
0	0	True
2	2	True

2	2	True
2	2	True
0	0	True
1	1	True
0	0	True
1	2	False
2	2	True
0	0	True
1	2	False
2	2	True
1	0	False
1	1	True
1	1	True
1	1	True
1	2	False
2	2	True
1	1	True
1	1	True
1	1	True
1	1	True
0	0	True
1	1	True
0	0	True
1	1	True
1	1	True
0	0	True
2	2	True

No	Actual	Predicted	Correct
14	1	2	False
81	1	0	False
85	1	0	False
125	1	2	False
159	1	2	False
167	1	0	False
233	2	1	False
256	1	2	False
275	1	2	False
278	1	2	False
280	1	0	False
284	1	2	False

Lampiran 2 *Script Code Python*

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings

import tensorflow as tf
tf.random.set_seed(1234)

warnings.filterwarnings('ignore')

import os
def load_data():
    for dirname, _, filenames in os.walk('/kaggle/input'):
        for filename in filenames:
            if filename.endswith('.csv'):
                data = pd.read_csv(os.path.join(dirname, filename))
                return data

data = load_data()

# avoid OOM error by set the GPU memory consumption growth
gpus = tf.config.experimental.list_physical_devices('GPU')
print(gpus)

# print(gpus)
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)

# menampilkan dataset
data.head()

# check the datasize
sns.countplot(data, x='class', palette='inferno')
plt.show()
# ==================== PREPROCESSING
====================
# split into k-mers
# memecah sekuens dna menjadi mers
def kmers(seq, size):
    kmers = [seq[x:x+size].lower() for x in range(len(seq)-size+1)]
    return kmers
```

```
data['3mers'] = data.apply(lambda x: kmers(x['sequence'], size=3), axis=1)
data['4mers'] = data.apply(lambda x: kmers(x['sequence'], size=4), axis=1)
data['5mers'] = data.apply(lambda x: kmers(x['sequence'], size=5), axis=1)
data['6mers'] = data.apply(lambda x: kmers(x['sequence'], size=6), axis=1)

# split X, y data
X_3= list(data['3mers'])
X_3= [(' '.join(X_3[seq])) for seq in range(len(X_3))]
y = data['class'].copy()
print(f"3-mers: {X_3[3]}")
X=list(data['4mers'])

# tokenize each kmer pairs
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_3)

X_tokenized = tokenizer.texts_to_sequences(X_3)

max_dna = max([len(s.split()) for s in X_3])
print('max DNA length {}'.format(max_dna))

X_padded = pad_sequences(X_tokenized, maxlen=max_dna, padding='post')
print({X_padded.shape[0]})

print(f"X-before tokenized: {X[5][:16]}")
print()
print(f"X-after tokenized: {X_tokenized[5][:16]}")
print()

# encode the classes
from sklearn.preprocessing import OneHotEncoder, LabelEncoder

oh = LabelEncoder()

labels = np.array(y)
labels = oh.fit_transform(labels)

print(f"Class before encoded: \n{y[:20]}")
print()
print(f"Class after encoded: \n{labels[:20]}")
```

```
from imblearn.over_sampling import SMOTE

# Membuat objek SMOTE
smote = SMOTE(random_state=42)

# Lakukan oversampling pada data yang sudah bebas dari outlier
X_os, y_os = smote.fit_resample(X_padded, y)

# see the classes after oversampling
OSclasses = pd.DataFrame(y_os)
OSclasses.columns = ['classes']

plt.subplots(figsize=(6,4))
sns.countplot(OSclasses, x='classes', palette='inferno')
plt.show()

import numpy as np
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_os, y_os, test_size=0.2,
random_state=42, stratify=y_os)

print(f'X_train size is {X_train.shape}')
print(f'y_train size is {y_train.shape}')
print(f'X_test size is {X_test.shape}')
print(f'y_test size is {y_test.shape}')
```

```
import numpy as np
import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from imblearn.over_sampling import SMOTE
import seaborn as sns
import matplotlib.pyplot as plt

# ====================================
# Helper function to create k-mers
def kmers(seq, size):
    return [seq[x:x + size].lower() for x in range(len(seq) - size + 1)]

# Function to preprocess and train SVM with given parameters
def process_and_train_svm(data, k, kernels, C_values):
    # Split into k-mers
    data[f'{k}mers'] = data.apply(lambda x: kmers(x['sequence'], size=k),
axis=1)

    # Prepare X and y
    X = list(data[f'{k}mers'])
    X = [(' '.join(X[seq])) for seq in range(len(X))]
    y = data['class'].copy()

    # Tokenize
    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(X)
    X_tokenized = tokenizer.texts_to_sequences(X)
    max_dna = max([len(s.split()) for s in X])
    X_padded = pad_sequences(X_tokenized, maxlen=max_dna,
padding='post')

    # Encode labels
    label_encoder = LabelEncoder()
    labels = np.array(y)
    labels = label_encoder.fit_transform(labels)

    # Apply SMOTE
    smote = SMOTE(random_state=42)
    X_os, y_os = smote.fit_resample(X_padded, labels)
```

```
    # Train-test split
    X_train, X_test, y_train, y_test = train_test_split(X_os, y_os, test_size=0.1,
random_state=42, stratify=y_os)

    # Train SVM for each kernel and C value
    for kernel in kernels:
        print(f"\n================ Kernel: {kernel}
================")
        for C in C_values:
            svm = SVC(kernel=kernel, C=C)
            svm.fit(X_train, y_train)

            # Predict and evaluate
            y_pred = svm.predict(X_test)
            accuracy = accuracy_score(y_test, y_pred)
            report = classification_report(y_test, y_pred)

            # Print results
            print(f'Kernel: {kernel}, C={C}')
            print(f'Accuracy: {accuracy:.2f}')
            print(f'Classification Report:\n{report}')
            print('-' * 60)

# ====================================
# Load your dataset here
# Assuming `data` DataFrame has columns 'sequence' and 'class'

# Parameter values
C_values = [0.01, 0.1, 1, 10, 100]
kernels = ['rbf', 'poly', 'sigmoid']

# Run the processing and training for 4-mers
print("Processing and training for 3-mers...")
process_and_train_svm(data, k=3, kernels=kernels, C_values=C_values)

# Run the processing and training for 4-mers
print("Processing and training for 4-mers...")
process_and_train_svm(data, k=4, kernels=kernels, C_values=C_values)

# Run the processing and training for 5-mers
print("\nProcessing and training for 5-mers...")
process_and_train_svm(data, k=5, kernels=kernels, C_values=C_values)
```

RIWAYAT HIDUP

M. Wildan Nuril Akmal atau lebih dikenal Wildan, lahir di Kabupaten Rembang pada 27 Juli 2003. Penulis merupakan anak pertama dari pasangan Bapak Afif Luthfi dan Ibu Mutimmatul Aliyah serta kakak dari M. Adly Habib Akmal. Penulis telah menempuh pendidikan mulai dari TK Muslimat NU yang lulus pada tahun 2009, dilanjutkan menempuh pendidikan sekolah dasar di MIS An Nashriyah Lasem dan lulus pada tahun 2015. Kemudian penulis melanjutkan pendidikan sekolah menengah pertama di MTS Negeri 1 Lasem dan lulus pada tahun 2018. Selanjutnya menempuh pendidikan sekolah menengah atas di MAN 2 Rembang dan lulus pada tahun 2021. Pada tahun yang sama, penulis melanjutkan pendidikan di Universitas Islam Negeri Maulana Malik Ibrahim Malang pada Program Studi Matematika, Fakultas Sains dan Teknologi. Selama menempuh pendidikan tinggi, penulis aktif mengikuti beberapa kegiatan. Penulis bergabung dalam organisasi HMPS "Integral" Matematika selama satu periode sebagai anggota Divisi Penerbitan dan Jurnalistik. Penulis juga pernah menjadi asisten praktikum Pemodelan Matematika selama satu semester. Pada tahun 2024 penulis berpartisipasi sebagai peserta Riset Kompetitif Mahasiswa Fakultas Sains dan Teknologi, dan mengikuti *Internasional Conference on Green Technology* (ICGT) 2024. Selain itu, penulis juga mengikuti kegiatan di luar kampus seperti pelatihan dan seminar.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : M. Wildan Nuril Akmal
NIM : 210601110096
Fakultas / Jurusan : Sains dan Teknologi / Matematika
Judul Skripsi : Implementasi Algoritma *Support Vector Machine* Pada Model Klasifikasi Sekuens DNA Manusia Studi Kasus: Penderita Diabetes Mellitus
Pembimbing I : Ari Kusumastuti, M.Si., M. Pd.
Pembimbing II : Mohammad Nafie Jauhari, M.Si.

No	Tanggal	Hal	Tanda Tangan
1.	13 September 2024	Konsultasi Topik dan Data	1.
2.	17 September 2024	Konsultasi Bab I, II, dan III	2.
3.	26 September 2024	Konsultasi Bab I, II, dan III	3.
4.	30 september 2024	ACC Bab I, II, dan III	4.
5.	01 Oktober 2024	Konsultasi Kajian Agama Bab I dan II	5.
6.	02 Oktober 2024	ACC Kajian Agama Bab I dan II	6.
7.	04 Oktober 2024	ACC Seminar Proposal	7.
8.	14 Oktober 2024	Konsultasi Revisi Seminar Proposal	8.
9.	17 Oktober 2024	Konsultasi Revisi Seminar Proposal	9.
10.	21 Oktober 2024	Konsultasi Bab IV dan V	10.
11.	24 Oktober 2024	Konsultasi Bab IV dan V	11.
12.	29 Oktober 2024	ACC Bab IV dan V	12.
13.	06 November 2024	Konsultasi Kajian Agama Bab IV	13.
14.	07 November 2024	ACC Kajian Agama Bab IV	14.
15.	08 November 2024	ACC Seminar Hasil	15.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

16.	15 November 2024	Konsultasi Revisi Seminar Hasil	16.
17.	21 November 2024	ACC Sidang Skripsi	17.
18	20 Desember 2024	ACC Keseluruhan	18.

Malang, 27 Desember 2024
Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005