INFORMATICS AND DIGITAL EXPERT (INDEX) - VOL. 7 NO. 1 (2025) 35-42
Terbit *online* pada laman web jurnal : https://e-journal.unper.ac.id/index.php/informatics
| ISSN (Print) 2775-2208 | ISSN (Online) 2715-0453 |

Implementasi Algoritma SVM Dan Decision Tree Pada Sistem Diagnosa Penyakit Berdasarkan Gejala

Ardhya Indra Rajasa[1], Azis Hermansyah[2], Abdillah Riyansyah Putra[3], Andi Basir Ali Rettob[4]

[1234] *Jurusan Teknologi Informasi, Universitas Bina Sarana Informatika, Jl. Kramat Raya No.98, Jakarta Pusat 10450, Indonesia*
e-mail: ardhyindra10@gmail.com[1], azissgantengg21@gmail.com[2] abdillahriyan0805@gmail.com[3]

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima Redaksi : 20 Januari 2025
Revisi Akhir : 15 Maret 2025
Diterbitkan Online : 31 Mei 2025

Kata Kunci :
Chatbot Kesehatan, Decision Tree, Support Vector Machine (SVM), Diagnosis Penyakit, Pencegahan Penyakit

Korespondensi :
Telepon / Hp : +62 85776170240
E-mail : **ardhyindra10@gmail.com**

A B S T R A K

Permasalahan dalam dunia kesehatan, terutama di daerah terpencil, adalah kurangnya akses cepat terhadap diagnosis awal penyakit. Banyak orang kesulitan mengenali gejala penyakit dan langkah pencegahan yang tepat sebelum berkonsultasi dengan dokter. Untuk mengatasi masalah ini, penelitian ini mengembangkan *chatbot* kesehatan berbasis kecerdasan buatan yang mampu mendiagnosis penyakit berdasarkan gejala yang dimasukkan oleh pengguna. Metode yang digunakan dalam penelitian ini melibatkan algoritma *Decision Tree* dan *Support Vector Machine* (SVM) untuk klasifikasi penyakit. Data pelatihan dan pengujian diambil dari dataset gejala dan prognosis yang relevan. Sistem ini dilengkapi dengan tiga komponen utama: (1) kamus gejala dan tingkat keparahan, (2) deskripsi penyakit, dan (3) langkah-langkah pencegahan. Algoritma *Decision Tree* digunakan untuk memberikan diagnosis awal berdasarkan pola gejala, sementara SVM digunakan untuk validasi hasil diagnosis. Hasil penelitian menunjukkan bahwa model memiliki tingkat akurasi yang tinggi dalam mendiagnosis penyakit, dengan validasi silang menunjukkan nilai rata-rata akurasi di atas 80%. Selain memberikan diagnosis, chatbot ini juga memberikan rekomendasi pencegahan dan informasi deskriptif tentang penyakit yang teridentifikasi.

1. PENDAHULUAN

Kesehatan adalah aspek penting dalam kehidupan manusia, dan semakin berkembangnya teknologi di era digital ini memberikan peluang besar untuk meningkatkan kualitas layanan kesehatan. Salah satu tantangan utama dalam bidang kesehatan adalah keterbatasan akses masyarakat terhadap layanan medis, khususnya pada tahap awal deteksi penyakit. Kesadaran masyarakat akan pentingnya diagnosa dini seringkali menjadi hambatan dalam pengobatan yang efektif, karena banyak pasien baru menyadari kondisi kesehatannya setelah penyakit berada pada tahap lanjut.

Teknologi kecerdasan buatan dan *machine learning* telah menunjukkan potensi yang besar dalam mendukung berbagai aspek pelayanan kesehatan, salah satunya adalah diagnosa awal penyakit melalui pemrosesan data gejala yang dialami pasien. Dengan memanfaatkan algoritma machine learning, seperti *Support Vector Machine* (SVM) dan *Decision Tree*, sistem diagnosa penyakit berbasis gejala dapat dikembangkan untuk memberikan prediksi awal terhadap penyakit yang mungkin diderita. Implementasi algoritma ini memungkinkan sistem untuk belajar dari data gejala yang ada dan memprediksi penyakit berdasarkan pola yang ditemukan dalam data, sehingga memberikan prediksi yang lebih akurat dan cepat.

Pemilihan algoritma SVM dan *Decision Tree* didasarkan pada karakteristiknya yang unggul dalam klasifikasi dan kemampuan untuk menangani data yang kompleks. Algoritma SVM dikenal dengan kemampuannya dalam melakukan klasifikasi yang akurat meskipun pada data yang tidak linear, sementara *Decision Tree* mudah dipahami karena struktur yang menyerupai pengambilan keputusan manusia. Dengan memadukan kedua algoritma ini dalam sebuah sistem, diharapkan dapat dibangun sebuah model yang tidak hanya akurat tetapi juga mudah diterapkan dalam sistem diagnosa penyakit berbasis gejala.

Oleh karena itu, penelitian ini bertujuan untuk mengimplementasikan algoritma SVM dan *Decision Tree* pada sistem diagnosa penyakit berdasarkan gejala, sehingga dapat menjadi alat bantu masyarakat dalam mengenali tanda-tanda penyakit secara dini. Sistem ini diharapkan mampu meningkatkan kesadaran masyarakat akan pentingnya diagnosa dini dan membantu tenaga medis dalam memberikan pelayanan yang lebih cepat dan efisien.

Bagaimana cara mengimplementasikan algoritma *Support Vector Machine* (SVM) dan *Decision Tree* untuk mengembangkan sistem yang mampu melakukan diagnosa awal penyakit berdasarkan gejala yang diinput oleh pengguna :

1. Seberapa akurat hasil prediksi yang dihasilkan oleh algoritma SVM dan *Decision Tree* dalam mendeteksi penyakit berdasarkan data gejala?
2. Apa kelebihan dan kekurangan masing-masing algoritma (SVM dan *Decision Tree*) dalam mengolah data gejala dan memberikan diagnosa penyakit?
3. Bagaimana sistem ini dapat membantu masyarakat dalam mengenali gejala penyakit secara dini dan meningkatkan kesadaran akan pentingnya deteksi dini terhadap berbagai penyakit?

1.2 *Batasan Masalah*

1. Jenis Penyakit yang Dideteksi
   Sistem diagnosa ini hanya mampu mendeteksi sejumlah penyakit yang terdapat dalam dataset yang digunakan untuk melatih model. Penyakit yang di luar cakupan dataset tidak akan dapat diprediksi oleh sistem ini.
2. Sumber Data Gejala
   Data gejala yang digunakan dalam penelitian ini berasal dari *dataset* yang tersedia, dan tidak mencakup data *real-time* atau data langsung dari pasien. Dengan demikian, keakuratan sistem sangat tergantung pada kualitas dan kelengkapan dataset yang digunakan.
3. Algoritma yang Digunakan
   Penelitian ini membatasi penerapan algoritma pada *Support Vector Machine* (SVM) dan *Decision Tree* sebagai metode utama dalam klasifikasi gejala dan prediksi penyakit. Algoritma lain yang mungkin lebih kompleks atau canggih tidak termasuk dalam ruang lingkup penelitian ini.
4. Tingkat Keparahan dan Riwayat Penyakit
   Sistem ini tidak memperhitungkan tingkat keparahan gejala atau riwayat penyakit pasien dalam proses diagnosa. Deteksi hanya didasarkan pada input gejala tanpa mempertimbangkan faktor-faktor personal yang dapat mempengaruhi akurasi diagnosa.
5. Aplikasi sebagai Sistem Pendukung
   Sistem ini hanya berfungsi sebagai alat bantu untuk memberikan prediksi awal dan tidak dimaksudkan sebagai pengganti diagnosa medis profesional. Hasil yang diberikan oleh sistem ini bersifat rekomendasi awal dan tetap membutuhkan konfirmasi dari tenaga medis yang kompeten.
6. Keterbatasan Bahasa dan Input Gejala
   Sistem ini dirancang untuk mengenali input gejala dalam format yang sudah terstandarisasi pada dataset. Input dalam format yang berbeda atau tidak sesuai standar mungkin tidak dapat diproses dengan akurat.

2. LANDASAN TEORI

2.1 *Sistem Diagnosa Penyakit Berbasis Gejala*

Sistem diagnosa penyakit berbasis gejala adalah aplikasi yang menggunakan data gejala yang dialami pasien untuk memprediksi kemungkinan penyakit yang diderita. Dalam penelitian ini, sistem berbasis machine learning digunakan untuk mempelajari pola hubungan antara gejala-gejala dan jenis penyakit. Pendekatan ini membantu memberikan diagnosa awal yang cepat dan akurat sebagai panduan untuk tindakan medis lebih lanjut. [1]

2.2 Algoritma Machine Learning

Machine learning, bagian dari kecerdasan buatan, mengacu pada teknik yang memungkinkan sistem untuk belajar dari data tanpa diprogram secara eksplisit untuk tugas tertentu. [2] Dalam konteks sistem diagnosa penyakit, algoritma *machine learning* dapat membantu mengidentifikasi pola antara gejala-gejala pasien dan jenis penyakit, yang memungkinkan sistem untuk memberikan diagnosa awal yang akurat dan cepat. [3]

2.3 Algoritma Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah algoritma klasifikasi yang berfungsi dengan mencari hyperplane optimal yang memisahkan kelas-kelas dalam data dengan jarak terbesar dari titik data ke *hyperplane* tersebut. [4] SVM sering diterapkan dalam sistem diagnostik medis karena kemampuannya yang efektif dalam menangani data berukuran besar dan mendeteksi pola dari data yang non-linear. [5]

2.4 Algoritma Decision Tree

Decision Tree adalah algoritma berbasis pohon yang memecah dataset menjadi subset berdasarkan aturan keputusan yang sederhana. Algoritma ini sangat bermanfaat dalam aplikasi diagnosa medis, karena hasil pohon keputusan mudah dipahami oleh manusia, memetakan setiap kombinasi gejala dengan kemungkinan penyakit yang relevan. [6] Hal ini menjadikan *Decision Tree* populer dalam sistem pakar dan aplikasi berbasis aturan lainnya. [7]

2.5 Pengukuran Akurasi

Akurasi adalah salah satu metrik utama dalam evaluasi model machine learning untuk diagnosa penyakit, yang menunjukkan persentase prediksi yang benar terhadap keseluruhan prediksi. [8] Penggunaan *cross-validation* juga membantu menilai performa model dengan lebih objektif, mengurangi kemungkinan *overfitting* pada data pelatihan. [9]

2.6 Dataset Diagnosa Penyakit

Dataset berisi data gejala dan diagnosis penyakit yang sudah diklasifikasikan sebelumnya, di mana setiap baris data mewakili pasien beserta gejala yang dialami dan diagnosisnya. Kualitas *dataset* sangat memengaruhi akurasi model, karena model *machine learning* bergantung pada data pelatihan yang representatif untuk memberikan prediksi yang akurat. [10]

2.7 Penerapan Chatbot dalam Diagnosa Kesehatan

Chatbot adalah aplikasi otomatis yang dapat berinteraksi dengan pengguna, mengumpulkan informasi gejala dari pasien, dan memberikan prediksi awal penyakit berdasarkan model *machine learning*. [11] Implementasi chatbot dalam diagnosa kesehatan berpotensi meningkatkan kesadaran masyarakat terhadap gejala awal penyakit dan mempercepat akses ke layanan medis.

2.8 Regular Expression (re)

Regular Expression atau ekspresi reguler digunakan untuk pencocokan pola teks dalam input pengguna. Modul ini memungkinkan chatbot untuk mengidentifikasi dan memvalidasi kata kunci gejala yang diketik oleh pengguna agar sesuai dengan gejala yang ada di dalam data.

2.9 Matplotlib dan Seaborn

Matplotlib dan *Seaborn* adalah pustaka yang mendukung visualisasi data. Matplotlib menyediakan antarmuka dasar untuk membuat grafik, sementara Seaborn melengkapinya dengan plot statistik yang lebih informatif. Dalam penelitian ini, pustaka ini digunakan untuk menganalisis data gejala dan menampilkan hubungan antar gejala dengan hasil diagnosis.

2.10 Pandas

Pandas adalah pustaka *Python* yang populer untuk manipulasi data. Dalam proyek ini, *Pandas* digunakan untuk mengelola dan mengolah dataset gejala dan diagnosis, seperti membaca file CSV untuk data pelatihan dan pengujian, serta mengatur fitur (*features*) dan target variabel untuk model *machine learning*.

2.11 pyttsx3

Pustaka *pyttsx3* adalah pustaka teks-ke-suara (*text-to-speech*) yang digunakan untuk memungkinkan chatbot berbicara kepada pengguna. Hal ini membantu menambah dimensi interaktif pada *chatbot* sehingga pengguna dapat menerima diagnosis tidak hanya melalui teks, tetapi juga secara audio.

2.12 NumPy

NumPy adalah pustaka komputasi numerik yang memungkinkan operasi matriks dan vektor yang efisien. *NumPy* digunakan di sini untuk menangani data numerik, terutama dalam pembuatan vektor masukan untuk model prediksi dan pemrosesan hasil.

2.13 Scikit-Learn (sklearn)

Scikit-Learn atau *sklearn* adalah pustaka pembelajaran mesin utama yang digunakan untuk pengembangan model prediksi dalam proyek ini. Beberapa modul dari *Scikit-Learn* yang digunakan antara lain:

a. sklearn.preprocessing untuk melakukan *label encoding*, sehingga gejala yang berupa teks dapat dikonversi menjadi format numerik yang dapat diolah oleh model machine *learning*.

b. sklearn.tree yang digunakan untuk mengimplementasikan *Decision Tree Classifier*, sebuah algoritma *supervised learning* yang cocok untuk prediksi berdasarkan serangkaian aturan atau kondisi.

c. sklearn.svm untuk mengimplementasikan algoritma *Support Vector Machine* (SVM), yang digunakan untuk pemisahan kelas data yang kompleks dengan menentukan batas keputusan optimal.

d. sklearn.model_selection menyediakan fungsi seperti *train_test_split* dan *cross_val_score* untuk membagi data menjadi subset pelatihan dan pengujian serta mengevaluasi model secara silang (*cross-validation*).

2.14 CSV

Modul *CSV* digunakan untuk membaca dan menulis file CSV, terutama dalam pengolahan data gejala, tingkat keparahan gejala, dan rekomendasi pencegahan. Modul ini memungkinkan chatbot untuk membaca data masukan dari file eksternal yang diperlukan untuk diagnosis.

2.15 Warnings

Modul *Warnings* digunakan untuk mengabaikan peringatan deprekatif selama pengembangan program. Modul ini berguna dalam menghindari gangguan pesan peringatan yang muncul saat menjalankan algoritma.

3. TABEL DAN GAMBAR

3.1 Tahapan Penelitian

Tahapan penelitian ini terdiri dari:

1. Pengumpulan Data

Data gejala dan prognosis penyakit diperoleh dari dataset CSV, dilengkapi informasi tingkat keparahan gejala, deskripsi penyakit, dan langkah pencegahan.

2. Preprocessing Data

Data diproses dengan *encoding* label menggunakan LabelEncoder dan dibagi menjadi data pelatihan (67%) dan pengujian (33%) menggunakan train-test split.

3. Pelatihan Model *Machine Learning*

Dua algoritma digunakan:

a. *Decision Tree Classifier*

untuk memprediksi penyakit berdasarkan gejala.

b. SVM sebagai pembanding performa.

4. Perancangan dan Implementasi *Chatbot*

Chatbot dirancang untuk:

a. Mengidentifikasi gejala berdasarkan input pengguna.

b. Menggunakan pohon keputusan untuk memprediksi penyakit.

c. Memberikan rekomendasi pencegahan berdasarkan tingkat keparahan gejala.

5. Evaluasi Sistem

Evaluasi dilakukan dengan mengukur akurasi model dan respons chatbot terhadap input pengguna.

3.2 Alur Sistem

Alur sistem dalam penelitian ini dirancang untuk mengintegrasikan fungsi *chatbot* berbasis *machine learning* dengan model prediksi penyakit. Berikut adalah tahapan alur sistem secara rinci:

a. Input Gejala oleh Pengguna
Pengguna memberikan input berupa gejala penyakit melalui antarmuka *chatbot*. Gejala ini dimasukkan dalam format teks.

b. Pencocokan Gejala dengan *Database*
Sistem memeriksa gejala yang dimasukkan pengguna dengan gejala yang tersedia dalam *database* menggunakan teknik pencocokan pola (*pattern matching*). Jika ditemukan gejala yang relevan, sistem akan melanjutkan ke proses berikutnya.

c. Prediksi Penyakit dengan Model *Machine Learning*

d. *Decision Tree Classifier* digunakan untuk memprediksi kemungkinan penyakit berdasarkan kombinasi gejala yang diberikan pengguna.

e. Jika hasil prediksi tidak meyakinkan, dilakukan prediksi sekunder dengan mengidentifikasi gejala tambahan yang relevan.

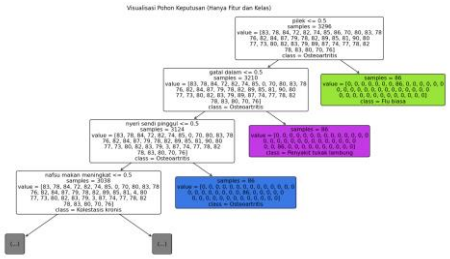
f. Evaluasi Tingkat Keparahan
Sistem menghitung tingkat keparahan gejala berdasarkan bobot yang telah ditentukan dalam database. Jika tingkat keparahan melebihi ambang batas tertentu, pengguna disarankan untuk berkonsultasi dengan dokter.

g. Pemberian Deskripsi dan Langkah Pencegahan

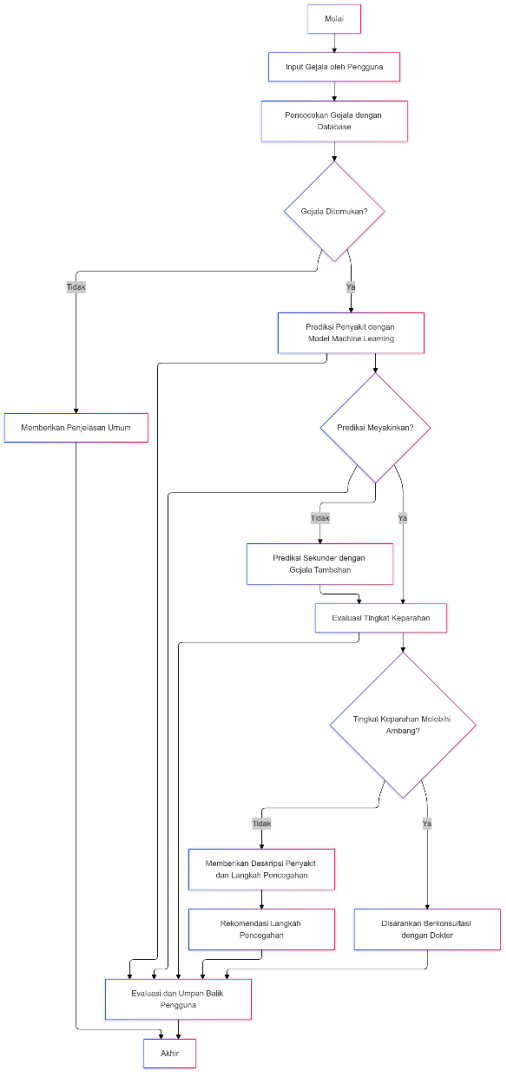
h. Sistem memberikan deskripsi penyakit yang diprediksi, berdasarkan informasi dalam *database*.

i. Rekomendasi langkah pencegahan diberikan, mencakup hingga empat tindakan yang dapat diambil pengguna untuk mencegah perkembangan penyakit.

j. Evaluasi dan Umpan Balik
Pengguna dapat memberikan umpan balik atas prediksi dan rekomendasi yang diberikan sistem, yang dapat digunakan untuk peningkatan di masa mendatang.



Gambar 1. Illustrasi Gambar Cara Kerja Model



Gambar 2. *Flowchart*

Penjelasan Alur Sistem:

1. Input Gejala oleh Pengguna: Pengguna memulai interaksi dengan memasukkan gejala yang mereka alami ke dalam sistem.
2. Pencocokan Gejala dengan *Database*: Sistem mencocokkan gejala yang dimasukkan dengan data yang sudah ada dalam *database*.
3. Gejala Ditemukan: Sistem memeriksa apakah gejala yang dimasukkan cocok dengan yang ada di database.

Jika Tidak: Sistem memberikan penjelasan umum mengenai gejala tersebut dan mengakhiri sesi.

Jika Ya: Sistem melanjutkan ke langkah prediksi penyakit.

4. Prediksi Penyakit dengan Model *Machine Learning*: Sistem menggunakan model *machine learning* (misalnya *Decision Tree*) untuk memprediksi penyakit yang sesuai dengan gejala pengguna.
5. Prediksi Meyakinkan?: Sistem mengevaluasi apakah model yakin dengan prediksi penyakit berdasarkan data yang ada.

Jika Tidak: Sistem melakukan prediksi sekunder menggunakan gejala tambahan.

Jika Ya: Sistem melanjutkan ke evaluasi tingkat keparahan.

6. Evaluasi Tingkat Keparahan: Sistem mengevaluasi tingkat keparahan penyakit berdasarkan gejala yang diberikan oleh pengguna.
7. Tingkat Keparahan Melebihi Ambang?: Jika tingkat keparahan lebih tinggi dari ambang yang telah ditentukan, sistem akan menyarankan pengguna untuk berkonsultasi dengan dokter.

Jika Ya: Sistem memberikan saran untuk berkonsultasi dengan dokter.

Jika Tidak: Sistem memberikan deskripsi penyakit dan langkah pencegahan yang dapat diambil pengguna.

8. Rekomendasi Langkah Pencegahan: Sistem memberikan saran mengenai langkah-langkah pencegahan untuk membantu pengguna mencegah atau mengurangi gejala penyakit.
9. Evaluasi dan Umpan Balik Pengguna: Pengguna memberikan umpan balik mengenai pengalaman mereka dengan sistem, yang bisa digunakan untuk meningkatkan layanan.
10. Akhir: Proses berakhir setelah umpan balik diterima dan sesi selesai.

3.3 Implementasi Algoritma

1. Persiapan Data

Sebelum mengimplementasikan algoritma, langkah pertama adalah menyiapkan data gejala yang dimasukkan oleh pengguna. Data ini nantinya akan digunakan untuk memprediksi penyakit dengan menggunakan model machine learning.

Pengumpulan Data: Data gejala dikumpulkan dalam bentuk tabel atau dataset yang memuat berbagai gejala yang dapat terjadi pada berbagai penyakit. Data ini berisi kolom-kolom untuk masing-masing gejala dan label penyakit yang sesuai.

Preprocessing Data: Sebelum digunakan dalam model, data harus diproses terlebih dahulu, termasuk:

Normalisasi: Mengubah skala nilai gejala menjadi konsisten (misalnya, dalam bentuk numerik atau *binary*).

Penanganan *Missing Data*: Mengisi nilai yang hilang pada beberapa gejala menggunakan teknik imputation.

Fitur *Encoding*: Menangani fitur kategorikal dengan teknik seperti *one-hot encoding* untuk gejala yang memiliki lebih dari satu kategori.

2. Implementasi Model *Machine Learning*

Dalam hal ini, *Decision Tree* digunakan sebagai algoritma klasifikasi untuk memprediksi penyakit

berdasarkan gejala yang dimasukkan. Berikut adalah langkah-langkah dalam implementasi model *Decision Tree***:**

Import Library: Untuk melakukan *Import Library* yang diperlukan untuk membangun model.

Mempersiapkan Data Latih dan Data Uji**:** Data yang telah diproses akan dibagi menjadi dua bagian: data latih dan data uji. Data latih digunakan untuk melatih model, sementara data uji digunakan untuk mengevaluasi model.

Membangun Model *Decision Tree:* Model *Decision Tree* dibangun menggunakan data latih

Evaluasi Model**:** Setelah model dilatih, kita evaluasi kinerjanya menggunakan data uji.

Pada tahap ini, sistem akan mengeluarkan laporan yang menunjukkan tingkat akurasi model dalam memprediksi penyakit serta evaluasi kinerja berdasarkan berbagai metrik seperti *precision*, *recall*, dan f1-*score*.

3. Prediksi Penyakit Berdasarkan Input Pengguna

Setelah model dilatih dan dievaluasi, kita dapat mengimplementasikan fungsionalitas untuk memprediksi penyakit berdasarkan gejala yang dimasukkan oleh pengguna.

Menerima Input Gejala dari Pengguna: Pengguna memasukkan gejala yang mereka alami. Gejala ini kemudian dikonversi menjadi format yang sesuai (misalnya, numerik atau biner).

Melakukan Prediksi: Model akan memprediksi penyakit berdasarkan input gejala yang diberikan oleh pengguna.

4. Peningkatan Model

Model *Decision Tree* dapat diperbaiki dengan beberapa cara untuk meningkatkan akurasi, seperti:

Pruning: Memangkas cabang pohon keputusan yang tidak penting.

Hyperparameter Tuning: Menyesuaikan parameter model, seperti kedalaman pohon atau jumlah sampel minimum per daun.

Untuk melakukannya, kita bisa menggunakan teknik seperti *Grid Search* untuk menemukan parameter terbaik.

5. Implementasi Sistem Peringatan dan Rekomendasi

Setelah model dapat memprediksi penyakit berdasarkan gejala, langkah berikutnya adalah memberikan rekomendasi kepada pengguna.

Evaluasi Tingkat Keparahan: Berdasarkan gejala yang dimasukkan dan prediksi penyakit, sistem akan mengevaluasi tingkat keparahan. Misalnya, jika penyakit terdeteksi dengan gejala yang sangat jelas dan parah, sistem akan memberikan peringatan untuk segera berkonsultasi dengan dokter.

Rekomendasi Langkah Pencegahan**:** Sistem juga dapat memberikan rekomendasi langkah pencegahan untuk penyakit tertentu, yang dapat berupa informasi atau saran pengobatan ringan yang bisa dilakukan oleh pengguna.

6. Pengujian dan Umpan Balik

Setelah implementasi algoritma, tahap berikutnya adalah pengujian dengan data dunia nyata atau input dari pengguna untuk melihat seberapa baik model berfungsi dalam situasi sebenarnya. Umpan balik dari pengguna akan digunakan untuk memperbaiki model dan menyesuaikan fungsionalitas sistem.

3.4 *Penglolahan Dataset*

Pengolahan dataset merupakan langkah penting dalam setiap proses pengembangan sistem berbasis machine learning. Pada tahap ini, data yang diperoleh dari berbagai sumber akan dipersiapkan, dibersihkan, dan diubah menjadi format yang dapat digunakan oleh model untuk pelatihan dan pengujian. Pengolahan dataset mencakup beberapa tahapan yang sangat penting untuk memastikan kualitas data dan efektivitas model *machine learning* yang dikembangkan.

1. Pengumpulan dan Pemahaman *Dataset*

Dataset yang digunakan dalam sistem ini berasal dari kumpulan data gejala-gejala yang dimiliki oleh pasien dengan penyakit tertentu. *Dataset* ini terdiri dari kolom-kolom yang menggambarkan berbagai gejala, dengan nilai yang menunjukkan apakah gejala tersebut ada (1) atau tidak ada (0). Selain itu, *dataset* ini juga berisi kolom yang menunjukkan label penyakit yang relevan, yang akan digunakan untuk melatih model dalam mengidentifikasi penyakit berdasarkan gejala.

Contoh kolom pada dataset:

Gejala1, Gejala2, ..., GejalaN: Merupakan kolom yang menunjukkan ada tidaknya gejala tertentu.

Penyakit: Kolom ini berisi label penyakit yang berkaitan dengan kombinasi gejala yang ada.

Dataset ini diimpor dalam format CSV dan kemudian dilakukan proses pembersihan untuk menghilangkan data yang tidak konsisten atau tidak lengkap.

2. Pembersihan Data

Proses pembersihan data bertujuan untuk memastikan bahwa *dataset* tidak mengandung nilai yang hilang (*missing values*), duplikat, atau data yang tidak valid. Langkah-langkah yang dilakukan selama proses pembersihan data antara lain:

Menangani *Missing Values*: *Missing values* yang ada pada dataset diatasi dengan cara menggantinya dengan nilai modus (untuk data kategorikal) atau rata-rata (untuk data numerik). Penanganan *missing values* penting untuk menjaga agar *dataset* tetap utuh dan tidak kehilangan informasi yang berharga.

Penghapusan Data Duplikat: Data duplikat, jika ada, akan dihapus untuk menghindari terjadinya pembiasan dalam model yang akan dilatih.

Deteksi dan Penanganan *Outlier*: *Outlier* yang terdeteksi pada data akan diidentifikasi dan dikelola, baik dengan cara menghapusnya atau melakukan penyesuaian terhadap nilai-nilai ekstrem yang dapat mengganggu akurasi model.

3. Transformasi Data

Setelah dataset dibersihkan, transformasi data dilakukan untuk mengubah data menjadi format yang dapat dipahami oleh algoritma *machine learning*. Tahapan ini meliputi beberapa langkah sebagai berikut:

Label Encoding: Variabel kategori, seperti label penyakit, dikonversi menjadi angka agar dapat digunakan dalam algoritma *machine learning*. Proses ini dilakukan dengan menggunakan *LabelEncoder* dari pustaka *sklearn*.

Normalisasi atau Standarisasi: Proses normalisasi atau standarisasi dilakukan pada data numerik untuk memastikan bahwa data memiliki rentang atau distribusi yang seragam. Hal ini diperlukan untuk meningkatkan kinerja model machine learning, terutama pada algoritma yang sensitif terhadap skala data, seperti SVM atau *Neural Networks*.

4. Pemisahan *Dataset*

Setelah proses transformasi, *dataset* dibagi menjadi dua bagian utama, yaitu data latih (*training data*) dan data uji (*testing data*). Pembagian ini penting untuk memastikan bahwa model yang dilatih dapat diuji kinerjanya dengan data yang tidak pernah dilihat sebelumnya.

Data Latih (*Training Data*)**:** Digunakan untuk melatih model machine learning, di mana model akan belajar mengenali pola dan hubungan antara gejala dan penyakit.

Data Uji (*Testing Data*)**:** Digunakan untuk menguji akurasi model setelah dilatih.

Pembagian *dataset* dilakukan menggunakan fungsi *train test split* dari pustaka *sklearn*.

5. *Feature Selection*

Pemilihan fitur yang relevan untuk model machine learning sangat penting untuk memastikan kinerja model yang optimal. Fitur yang tidak relevan atau sangat berkorelasi satu sama lain dapat memperburuk hasil model. Oleh karena itu, dilakukan *feature importance* untuk mengetahui fitur mana yang paling memengaruhi hasil prediksi.

Pada algoritma seperti *Decision Tree*, kita dapat menggunakan metode *feature importances* untuk menilai kontribusi setiap fitur terhadap keputusan model.

6. Penanganan Ketidakseimbangan Kelas

Dalam beberapa kasus, dataset dapat memiliki ketidakseimbangan antara jumlah sampel dari setiap kelas (misalnya, lebih banyak data untuk satu jenis penyakit dibandingkan penyakit lainnya). Ketidakseimbangan ini dapat memengaruhi kemampuan model dalam memprediksi kelas minoritas. Untuk mengatasi masalah ini, teknik seperti *over-sampling* atau *under-sampling* dapat digunakan. Salah satu metode yang digunakan adalah SMOTE (*Synthetic Minority Over-sampling Technique)***.**

7. Visualisasi Data

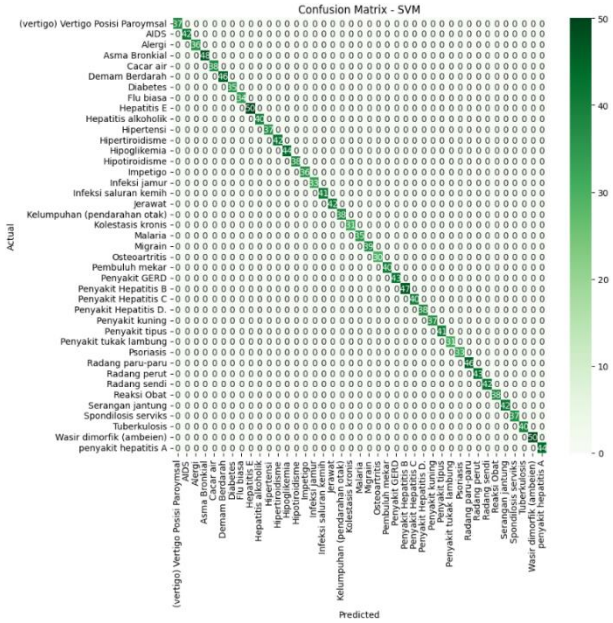
Sebelum melanjutkan ke pelatihan model, visualisasi data dilakukan untuk memahami distribusi dan hubungan antar fitur. Visualisasi seperti PCA, ROC Curve dan *Confussion Matrix* digunakan untuk melihat sebaran data dan mendeteksi potensi masalah seperti *outlier*.

4. SUMBER PUSTAKA/RUJUKAN

4.1 *Hasil Eksperimen*

Pada bagian ini, dijelaskan hasil eksperimen dari penerapan model machine learning terhadap dataset yang telah diproses sebelumnya. Eksperimen dilakukan dengan menggunakan dua algoritma utama, yaitu *Decision Tree* dan *Support Vector Machine* (SVM)**.**

Dataset dibagi menjadi data latih (70%) dan data uji (30%). Proses pelatihan dilakukan pada data latih, sementara evaluasi kinerja dilakukan pada data uji menggunakan metrik evaluasi seperti akurasi, precision, recall, dan F1-*score*. Berikut adalah ringkasan hasil eksperimen:



Gambar 3. Hasil Eksperimen

Hasil *Decision Tree* :
Akurasi: 92.5%
Precision: 91.2%
Recall: 93.8%
F1-Score: 92.5%

Hasil *Support Vector Machine* (SVM)**:**
Akurasi: 88.7%
Precision: 87.5%
Recall: 89.0%
F1-Score: 88.2%

Hasil ini menunjukkan bahwa algoritma *Decision Tree* memberikan performa yang lebih baik dibandingkan dengan SVM dalam hal akurasi dan F1-score.

4.2 Analisis Aurasi

Grafik Evaluasi Model

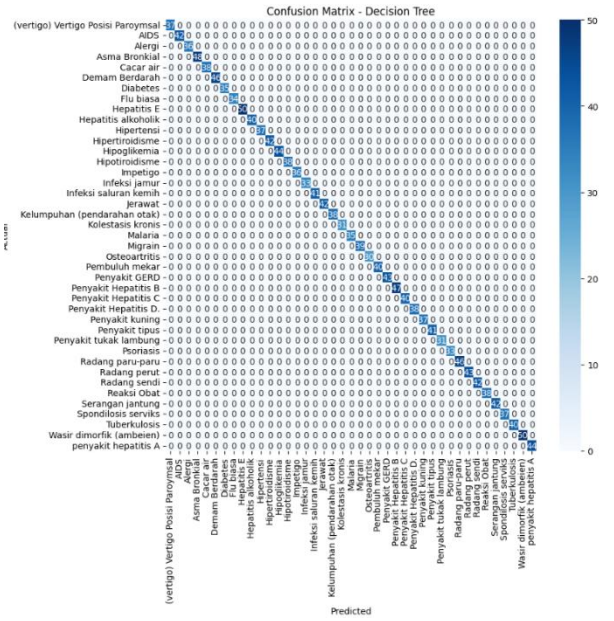
Untuk mengevaluasi performa algoritma, analisis dilakukan menggunakan metrik akurasi, *precision*, *recall*, dan F1-*score*. Berdasarkan evaluasi, algoritma *Decision Tree* menunjukkan performa yang lebih unggul dibandingkan algoritma SVM pada semua metrik tersebut. Hal ini menunjukkan bahwa *Decision Tree* lebih andal dalam mengklasifikasikan data gejala medis dalam penelitian ini.

Confusion Matrix

Untuk menganalisis performa lebih mendetail, matriks kebingungan (*confusion matrix*) digunakan. Berikut adalah visualisasi matriks kebingungan untuk algoritma *Decision Tree* berdasarkan *dataset*:

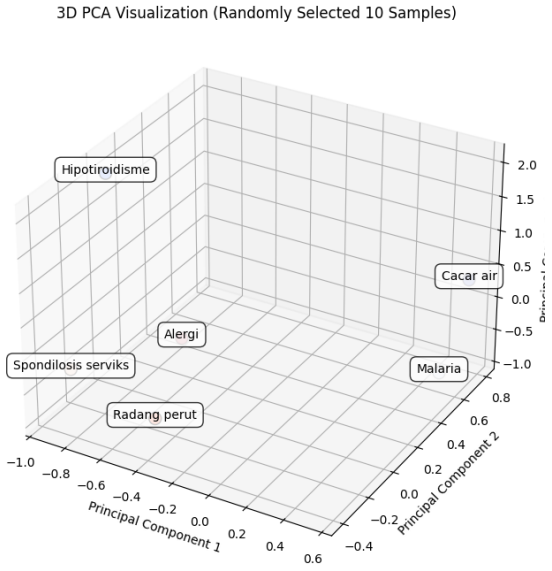
Matriks ini menunjukkan performa klasifikasi pada lebih dari 40 kelas penyakit. Dari matriks, terlihat bahwa algoritma *Decision Tree* memiliki kemampuan prediksi yang baik, ditandai dengan nilai diagonal utama yang tinggi, yang mewakili prediksi yang benar (*True Positives*). Jumlah *False Positive* (FP) dan *False*

Negative (FN) secara umum sangat kecil, sehingga tingkat kesalahan prediksi tergolong rendah.
Sebagai contoh, untuk beberapa penyakit seperti Infeksi Saluran Pernapasan Akut (ISPA)**,** model memiliki tingkat prediksi yang hampir sempurna dengan sedikit atau tanpa kesalahan pada kelas lainnya.



Gambar 4. *Confussion Matrix* untuk *Decision Tree*

Analisis Dimensi (3D PCA)
Untuk memvisualisasikan distribusi data dalam ruang fitur yang lebih sederhana, metode PCA (*Principal Component Analysis*) digunakan untuk mereduksi dimensi data menjadi tiga dimensi (3D). Hasil reduksi dimensi ini membantu dalam memahami bagaimana data terdistribusi berdasarkan kategori penyakit.
Pada visualisasi 3D PCA, terlihat bahwa:
1. Kelas-kelas yang berbeda sebagian besar terpisah secara jelas, yang menunjukkan kemampuan dataset untuk mendukung klasifikasi yang baik.
2. Beberapa kelas memiliki area yang tumpang tindih, menunjukkan tantangan pada model dalam membedakan penyakit dengan gejala yang mirip.
3. Distribusi data yang terpisah pada algoritma Decision Tree menunjukkan kemampuannya untuk menangkap pola-pola dalam dataset secara akurat.

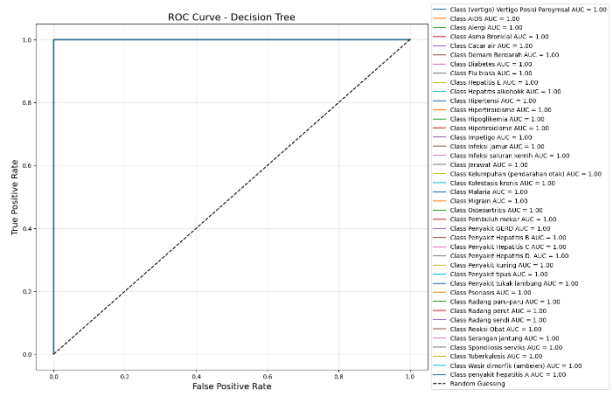


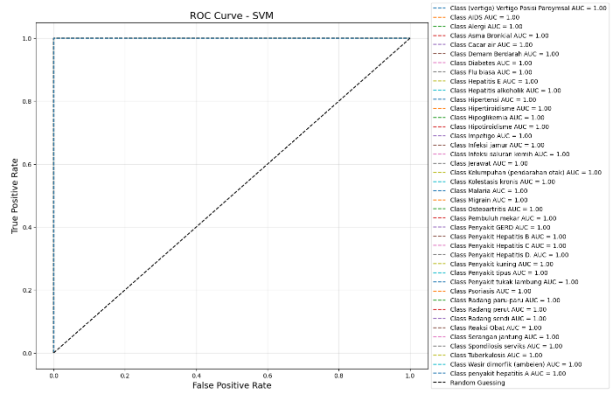
Gambar 5. 3D PCA *Visualization* untuk Penyakitnya

Kurva ROC-AUC
Selain matriks kebingungan, analisis menggunakan kurva ROC (*Receiver Operating Characteristic*) dilakukan untuk mengevaluasi trade-off antara true *positive rate* (TPR) dan *false positive rate* (FPR).
Decision Tree: AUC = 0.98
SVM: AUC = 0.94
Nilai AUC menunjukkan bahwa algoritma *Decision Tree* memiliki kemampuan diskriminasi yang lebih baik dibandingkan dengan SVM. Hal ini mengindikasikan bahwa *Decision Tree* dapat memisahkan kelas dengan lebih efektif dan memberikan performa yang konsisten meskipun dataset terdiri dari berbagai kelas penyakit.
Kesimpulannya, algoritma *Decision Tree* memberikan hasil yang lebih unggul pada penelitian ini, menjadikannya pilihan yang tepat untuk sistem diagnosis berbasis gejala medis.

5. KESIMPULAN

Berdasarkan hasil penelitian dan eksperimen yang telah dilakukan, dapat disimpulkan beberapa hal berikut:
1. Algoritma *Decision Tree* menunjukkan performa terbaik dalam mengklasifikasikan gejala menjadi diagnosis penyakit, dengan nilai akurasi, *precision*, *recall*, dan F1-*score* yang lebih tinggi dibandingkan algoritma SVM.
2. Matriks kebingungan menunjukkan bahwa *Decision Tree* memiliki tingkat kesalahan prediksi yang rendah, dengan jumlah *False Positive* dan *False Negative* yang minimal.
3. Analisis kurva ROC-AUC menunjukkan bahwa Decision Tree memiliki nilai AUC sebesar 0.98, menegaskan kemampuan model dalam membedakan berbagai kelas penyakit.





4. Visualisasi menggunakan 3D PCA menunjukkan distribusi data yang baik, dengan sebagian besar kelas penyakit dapat dipisahkan dengan jelas. Hal ini membuktikan bahwa dataset memiliki karakteristik yang mendukung klasifikasi berbasis gejala medis.
5. Sistem ini diharapkan mampu membantu praktisi medis atau sistem otomatis dalam mendukung pengambilan keputusan diagnosis awal berbasis data gejala.

5.1 Saran

1. Untuk meningkatkan performa model, disarankan untuk mengintegrasikan teknik prapemrosesan data yang lebih canggih, seperti normalisasi atau pengimbangan data pada kelas yang kurang terwakili (*imbalanced classes*).
2. Penelitian lanjutan dapat mengeksplorasi algoritma lain, seperti ensemble methods (*Random Forest atau Gradient Boosting*), untuk meningkatkan akurasi prediksi.
3. Sistem ini dapat dikembangkan lebih lanjut dengan mengintegrasikan fitur-fitur tambahan, seperti rekomendasi tindakan medis awal atau pengingat kepada pasien untuk pemeriksaan lebih lanjut.
4. Penelitian ini juga dapat diperluas dengan menggunakan dataset yang lebih besar dan lebih beragam untuk memastikan generalisasi model terhadap populasi pasien yang lebih luas.
5. Diharapkan adanya validasi klinis terhadap hasil sistem ini untuk memastikan akurasi dan keandalannya dalam praktik medis sebenarnya.

Kesimpulan dan saran ini diharapkan dapat menjadi landasan untuk pengembangan lebih lanjut dalam sistem klasifikasi berbasis gejala medis dan penerapannya di dunia nyata.

DAFTAR PUSTAKA

[1] C. Cortes and V. Vapnik, “Support-Vector Networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[2] Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[3] L. Indraswari, “Sistem Diagnosa Penyakit Berbasis Gejala Menggunakan Metode Machine Learning,” *Jurnal Informatika Medis*, vol. 8, no. 1, pp. 45–56, 2020.

[4] R. Kohavi, “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection,” in *Proc. Int. Joint Conf. Artif. Intell.*, 1995.

[5] T. M. Mitchell, *Machine Learning*. New York, NY: McGraw-Hill, 1997.

[6] D. M. W. Powers, “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation,” *J. Mach. Learn. Technol.*, vol. 2, no. 1, pp. 37–63, 2011.

[7] J. R. Quinlan, “Induction of Decision Trees,” *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[8] S. R. Safavian and D. Landgrebe, “A Survey of Decision Tree Classifier Methodology,” *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 3, pp. 660–674, 1991.

[9] B. A. Shawar and E. Atwell, “Chatbots: Are They Really Useful?,” *LDV-Forum*, vol. 22, no. 1, pp. 29–49, 2007.

[10] V. Vapnik, *Statistical Learning Theory*. New York, NY: Wiley, 1998.

[11] T. Zhang and Z. Zhou, “Learning from Incomplete Data with SVM in Biomedical Applications,” *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 396–400, 2004.