BAB II

TINJAUAN PUSTAKA

2.1 Penelitian-Penelitian Sebelumnya

Pada *state of the art* ini diambil beberapa penelitian terdahulu sebagai panduan penulis untuk penelitian yang akan dilakukan, yang kemudian akan menjadi acuan dan perbandingan dalam melakukan penelitian ini. Dalam *state of the art* ini akan terdapat beberapa jurnal.

Tabel 2. 1 *State of The Art*

NO	Peneliti	Metode	Kelebihan	Kekurangan
1	Setiawati dkk, 2019 [6].	Implementasi *Decision Tree* Untuk Mendiagnosis Penyakit *Liver*.	Mendapatkan nilai akurasi sebesar 72.67%, model yang dikembangkan menunjukkan kinerja yang cukup baik dalam mengklasifikasikan penyakit *liver* berdasarkan dataset ILPD.	*Decision Tree* mungkin tidak optimal dalam menangani hubungan *non-linear* dan interaksi yang kompleks antara variabel, dibandingkan dengan metode *machine learning* yang lebih canggih seperti *Random Forest*.
2	Prabiantissa, 2021 [7].	Klasifikasi pada Dataset Penyakit Hati Menggunakan Algoritma *Support Vector Machine*, K-NN, dan *Naïve Bayes*	SVM menunjukkan performa terbaik dengan rata-rata akurasi 82,36%, menjadikannya algoritma yang diunggulkan untuk klasifikasi penyakit hati dalam penelitian ini.	Penelitian ini hanya menggunakan satu dataset (dataset penyakit hati dari UCI *Machine Learning Repository*), yang mungkin tidak cukup representatif untuk semua kasus penyakit hati, sehingga dapat Mempengaruhi kemampuan generalisasi hasil

NO	Peneliti	Metode	Kelebihan	Kekurangan
3	Desiani, 2022 [8].	Perbandingan Implementasi Algoritma *Naïve Bayes* dan K-NN pada Klasifikasi Penyakit Hati.	Algoritma K-NN memberikan nilai akurasi, presisi, serta *recall* yang lebih tinggi dibandingkan dengan algoritma *Naïve Bayes*. Maka algoritma terbaik yang dapat digunakan adalah K-NN.	Hasil yang menunjukan akurasi K-NN 100% mungkin menunjukan *overfitting* atau masalah lain yang tidak dijelaskan.
4	Patimah, dkk, 2021 [9].	Klasifikasi Penyakit Liver dengan Menggunakan Metode *Decision Tree*	Penelitian ini menghasilkan akurasi dari kombinasi berbagaiteknik, dengan jelas menunjukan bahwak-fold cross validation dengan standar scaler menghasilkan akurasi tertinggi sebesar 0.7333.	Tidak disebutkan apakah ada ketidakseimbangan kelas dalam dataset dan bagaimana ini ditangani. Teknik seperti SMOTE bisa digunakan untuk memperbaiki masalah ini, yang bisa meningkatkan akurasi model lebih lanjut.

2.2 Teori Pendukung

Dalam penelitian ini, penulis akan membahas teori-teori pendukung yang menjadi landasan utama dalam memahami fenomena yang diamati. Teori-teori tersebut diharapkan dapat memberikan penjelasan yang komprehensif mengenai hubungan yang terjadi antara variabel-variabel yang diteliti, serta menjabarkan bagaimana interaksi di antara variabel tersebut dapat mempengaruhi hasil penelitian. Pembahasan teori pendukung ini tidak hanya akan merangkum teori-teori yang relevan, tetapi juga menguraikan secara mendetail bagaimana teori-teori tersebut berperan penting dalam mendukung dan memperkuat dasar pemikiran serta arah analisis dalam penelitian ini.

2.2.1 Penyakit Hati

Hati merupakan organ yang sangat penting dalam pengaturan *homeostasis* tubuh meliputi metabolisme, biotransformasi, sintesis, penyimpanan dan *imunologi*. Sel - sel hati (*hepatosit*) mempunyai kemampuan regenerasi yang cepat. Oleh karena itu sampai batas tertentu, hati dapat mempertahankan fungsinya bila terjadi gangguan ringan. Pada gangguan yang lebih berat, terjadi gangguan fungsi yang serius dan akan berakibat fatal. Penyebab penyakit hati bervariasi, sebagian besar disebabkan oleh virus yang menular secara fekal-oral, parenteral, seksual, perinatal dan sebagainya. lain dari penyakit hati adalah akibat efek toksik dari obat-obatan, alkohol, racun, jamurdan lain-lain. Di samping itu juga terdapat beberapa penyakit hati yang belum diketahui pasti penyebabnya.

Jenis-jenis penyakit hati antara lain yaitu Hepatitis, *Liver*, Sirosis, Kanker Hati, *Jaundice* (penyakit kuning), Kegagalan Hati, Kolangitis, Leptospirosis dan Abses Hati. Penyakit-penyakit hati akut akan banyak mempengaruhi fungsi-fungsi hati, penyakit tesebut dapat diketahui dari gejala klinis maupun fisik yang timbul pada diri pasien, gejala klinis dapat diketahui dari apa yang dirasakan oleh pasien, sedangkan gejala fisik dapat diketahui dari keadaan tubuh pasien [3].

Organisasi Kesehatan Dunia (WHO) memperkirakan 354 juta orang di seluruh dunia hidup dengan hepatitis B atau C dan setiap tahun ada satu juta orang meninggal karena hepatitis. Di Indonesia diperkirakan ada sekitar 20 juta orang menderita hepatitis dengan prevalensi tertinggi pada kasus hepatitis B. CDA *Foundation* mencatat angka kematian akibat hepatitis B di Indonesia mencapai 51.100 setiap tahun dan kematian akibat hepatitis C sebesar 5.942 tiap tahun pada 2016. Menurut data BPJS Kesehatan, 2.159 orang meninggal karena sirosis dan kanker hati, yang merupakan dampak dari hepatitis kronis yang biasanya dialami orang dengan hepatitis B pada stadium lanjut, pada 2022.

WHO pada 2020 mengeluarkan resolusi bahwa penyakit hepatitis menjadi Namun, sulitnya mendeteksi penyakit ini menjadi tantangan yang harus diatasi karena biasanya pasien hepatitis diketahui ketika kondisinya sudah tingkat lanjut.

Selain mengungkap data yang mengkhawatirkan itu, laporan ini juga memaparkan tentang cara mencegah hepatitis dan menangani pasien hepatitis sehingga diharapkan penyakit ini dapat dideteksi sejak dini dan jumlah penderitanya berkurang [10].

2.2.2 Klasifikasi

Pemanfaatan teknologi tidak lepas dari kegiatan manusia sehari-hari. Komputer adalah salah satu teknologi yang digunakan untuk melakukan pengolahan atau komputasi data. pengolahan dan komputasi data pada teknologi memiliki banyak metode. Metode tersebut dikembangkan untuk memudahkan pengolahan data sesuai dengan macam atau tipe data yang akan diolah. Salah satu metode dalam pengolahan data adalah klasifikasi [20].

Klasifikasi merupakan cara pengelompokkan benda berdasarkan ciri – ciri yang dimiliki oleh objek klasifikasi [20]. Dalam prosesnya, klasifikasi dapat dilakukan dengan banyak cara baik secara manual ataupun dengan bantuan teknologi. Klasifikasi yang dilakukan secara manual adalah klasifikasi yang dilakukan oleh manusia tanpa adanya bantuan dari algortima cerdas komputer. Sedangkan klasifikasi yang dilakukan dengan bantuan teknologi, memiliki beberapa algoritma, diantaranya *Random Forest*, *Naïve Bayes*, *Support Vector Machine*, *Decission Tree*, dan *Fuzzy*.

Teknik klasifikasi merupakan salah satu tugas yang penting dalam *machine learning*. Sebuah pengklasifikasi dibuat dari sekumpulan data latih dengan kelas yang telah di tentukan. Klasifikasi merupakan pengelompokan fitur ke dalam kelas yang sesuai. Vektor fitur pelatihan tersedia dan telah diketahui kelas-kelasnya, kemudian vektor fitur pelatihan tersebut dimanfaatkan untuk merancang pemilah. Pengenalan pola ini disebut terbimbing, *supervised*. Seperti yang telah dinyatakan sejumlah klasifikasi teknik telah diusulkan dalam literatur. Terutama proses klasifikasi dibagi menjadi beberapa kategori yang berbeda, yang dinamakan sebagai keputusan berbasis pengklasifikasi [11].

2.2.3 *Machine Learning*

Machine Learning (ML) merupakan bidang studi yang fokus kepada desain dan analisis algoritma sehingga memungkinkan komputer untuk dapat belajar [12]. MLberisi sebuah algoritma yang bersifat (umum) dimana algoritma tersebut dapat menghasilkan sesuatu yang menarik atau bermanfaat dari sejumlah data tanpa harus menulis kode yang spesifik. Pada intinya, algoritma yang generik tersebut ketika diberikan sejumlah data maka ia dapat membangun sebuah aturan atau model atau inferensi dari data tersebut.

Sebagai contoh sebuah algoritma untuk mengenali tulisan tangan dapat digunakan untuk mendeteksi email yang berisi spam dan bukan spam tanpa mengganti kode. Algoritma yang sama ketika diberikan data pelatihan yang berbeda menghasilkan logika klasifikasi yang berbeda.

2.2.3.1 Jenis-Jenis *Machine Learning*

Machine Learning terdapat banyak jenis dan tipenya berdasarkan hasil yang akan didapatkan dengan melihat kondisi data yang diperoleh. Berikut adalah jenis-jenis *machine learning*:

1. *Supervised Learning*

   Metode *supervised learning* bisa juga disebut "*learning byexample*", yaitu metode melatih model ML di mana sebuah model diberikaninput data beserta dengan output berupa label dari input tersebut. Dengan kata lain, *supervised learning* dapat didefinisikan sebagai metode *training* model ML di mana model akan menganalisis data input untuk menghasilkan*prediction*, dan *prediction* ini akan dibandingkan dengan *label* yang diberikan untuk melatih model menghasilkan *label* yang tepat.

   Metode *learning* ini sangat umum digunakan dalam berbagai bidang ML, khususnya *regression* dan *classification*. Dalam bidang ML, *regression* adalah *task* untuk membuat prediksi yang berupa nilai kontinu. Beberapa contoh kasus *regression* ini adalah prediksi jumlah pembeli suatu produk atau perkiraan tingkat keparahan suatu penyakit yang diderita seseorang. Dilain sisi, *classification* adalah *task* yang berupa pengelompokkan data ke dalam kelompok dengan *label*. Dalam *task* ini, model ML akan

menganalisis fitur kunci yang membedakan data dari masing-masing kelompok seperti klasifikasi jenis penyakit, deteksi email spam dan sebagainya.

Beberapa jenis model ML yang umum digunakan dengan metode *supervised learning* meliputi *Linear Regression, Logistic Regression, Decision Tree* (DT), *Support Vector Machine* (SVM) dan salah satunya adalah *Random Forest* yang digunakan dalam penelitian ini.

2. *Unsupervised learning*

Berbeda dengan *supervised learning*, metode *unsupervised learning* tidak membutuhkan *label* untuk melatih model ML. Model yang dilatih dengan *unsupervised learning* diarahkan untuk menganalisis pola tersembunyi dari data input yang mereka terima, kemudian mengelompokkan data-data yang mirip. Karena tugasnya adalah mengelompokkan data yang mirip, metode ini tidak membutuhkan label pada data. Teknik *unsupervised learning* secara umum dapat digunakan untuk empat jenis kasus seperti *Clustering, Association, Anomaly Detection, Dimensionality Reduction* [13].

Machine Learning yang digunakan dalam penelitian ini adalah *classification* dari *supervised learning*. Algoritma ini membuat keputusan dan memberi label berdasarkan kelas data. *Classification* yang digunakan adalah *Random Forest*.

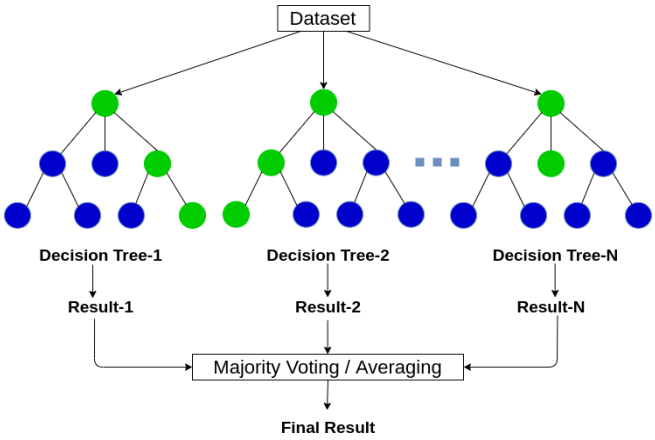
2.2.4 Algoritma *Random Forest*

Algoritma *Random Forest* adalah salah satu yang menggunakan pembelajaran *supervised*. Ini menghasilkan kumpulan pohon keputusan, yang sering dilatih menggunakan pendekatan "*bagging*". Inilah artinya jika mengacu pada "*forest*". Premis dasar di balik pendekatan *bagging* adalah bahwa menggabungkan beberapa model pembelajaran harus mengarah pada peningkatan produk akhir. *Random Forest classifier* merupakan salah satu metode yang digunakan untuk klasifikasi dan regresi. *Random Forest* juga dapat diartikan sebagai terbentuk dari serangkaian pohon keputusan atau *decision tree*. Dapat digunakan untuk memprediksi kategori dengan beberapa nilai yang mungkin dan

probabilitas keluaran yang dapat disesuaikan. Satu hal yang harus diperhatikan adalah *overfitting*. *Random Forest* adalah sejenis pembelajaran mesin yang membangun banyak pohon keputusan dan kemudian menggabungkan pohon-pohon untuk mendapatkan prediksi yang lebih akurat dan konsisten [14].

Random Forest merupakan metode yang efektif karena terdiri dari banyak pohon keputusan (*decision trees*). *Random Forest* menggabungkan hasil dari beberapa estimator yang dibuat oleh *decision tree*, performa *Random Forest* akan semakin baik jika semakin sering estimator yang digunakan. Model *Random Forest* menggunakan minimal 100 penduga (*estimators*) dan untuk mengukur kualitas dalam pembagian data menggunakan kriteria gini atau entropi.

Random Forest memiliki beberapa kelebihan, yaitu dapat meningkatkan hasil akurasi jika terdapat data yang hilang, dan untuk *resisting outliers*, serta efisien untuk penyimpanan sebuah data. Selain itu, *Random Forest* mempunyai proses seleksi fitur dimana mampu mengambil fitur terbaik sehingga dapat meningkatkan performa terhadap model klasifikasi. Dengan adanya seleksi fitur tentu *Random Forest* dapat bekerja pada *big data* dengan parameter yang kompleks secara efektif [5].



Gambar 2. 1 Model *Random Forest*

Proses pembentukan *Random Forest* dapat dilakukan dengan langkah-langkah, mengambil sampel *bootstrap* sebanyak *n-tree* dari data. Membangun pohon untuk setiap sampel *bootstrap*. Pada setiap simpul pohon, secara acak dipilih variabel untuk dipisahkan, dan pohon dibiarkan tumbuh hingga setiap node terminal

memiliki ukuran kasus yang memadai. Informasi dari masing-masing pohon dalam *n-tree* diagregasikan untuk memprediksi data baru, seperti melalui pemilihan mayoritas untuk klasifikasi. Menghitung tingkat kesalahan *out-of-bag* (OOB) dengan menggunakan data yang tidak termasuk dalam sampel *bootstrap*.

Algoritma pada *Random Forest* memerlukan data yang telah diberi label untuk mempermudah saat pemrosesan data. *Random Forest* dapat menangani model yang kompleks dan data yang relatif besar. Selain itu, *Random Forest* dapat mengatasi kompleksitas data teks bahasa Indonesia dengan memahami makna dari setiap kata lalu mengambil keputusan dari pemahaman tersebut. Akan tetapi, dikarenakan *Random Forest* menggunakan banyak *Decision Tree* ini mengakibatkan proses klasifikasi menjadi lambat dan cukup memakan waktu [15].

Algoritma *decision tree* mempunyai beberapa algoritma, yaitu ID3 berdasarkan nilai entropy dan CART berdasarkan nilai gini. CART merupakan *algoritma decision tree* selain ID3. Pada tahun 1984, sekelompok ahli statistik menerbitkan buku *Classification* and *Regression Trees* atau CART, yang menjelaskan generasi dari decision tree biner. Rumus dari pencarian nilai *impurity* pada algoritma CART. Berikut persamaannya:

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \qquad (2.1)$$

Dimana *Pi* adalah nilai peluang dari sebuah nilai tuple *D* pada suatu kelas dan *m* adalah jumlah label kelas [16].

2.2.5 *Synthetic Minority Oversampling Technique* (SMOTE)

Klasifikasi pada *imbalanced* dataset cukup sulit dilakukan karena terlalu sedikit data dari kelas minoritas untuk dapat menghasilkan model klasifikasi yang efektif. Untuk mengatasi hal tersebut, salah satu cara yang bisa dilakukan adalah melakukan oversampling yaitu melakukan duplikasi data dari kelas minoritas pada data latih agar menghasilkan model yang lebih baik [17].

Synthetic Minority Oversampling Technique (SMOTE) memilih data pada kelas minoritas secara acak kemudian mencari *k* data kelas minoritas terdekat dari data tersebut [17]. Dari *k* data kelas minoritas tersebut, akan dipilih satu data secara acak yang selanjutnya dihubungkan dengan data awal yang dipilih untuk

membentuk segmen garis pada ruang fitur. Data sintesis dihasilkan menggunakan kombinasi *convex* antara dua data yang dipilih secara acak tadi. Pendekatan ini dinilai cukup efektif karena data sintesis yang dibentuk relatif dekat dalam ruang fitur dengan data yang ada dari kelas minoritas. Prosedur ini dapat digunakan untuk membuat sebanyak mungkin data sintetik untuk kelas minoritas yang diperlukan.

Namun agar menghasilkan data yang lebih baik, teknik SMOTE menyarankan penggunaan under-sampling acak untuk menyeimbangkan distribusi kelas. Kekurangan dari SMOTE adalah data sintesis yang dibentuk tidak memperhatikan kelas mayoritas sehingga memungkinkan adanya tumpang tindih data.

2.2.6 Bahasa Pemograman *Pyhton*

Bahasa yang dipahami komputer adalah bahasa mesin. Pada mulanya manusia memberi intruksi kepada komputer langsung menggunakan bahasa mesin yaitu, bahasa asli komputer. Karena pemberian intruksi ini dianggap kurang efesien dan rumit, maka diciptakanlah media perantara untuk menerjemahkan komunikasi manusia dengan komputer. Media perantara itu disebut dengan bahasa pemrograman. Seiring dengan meningkatnya kompleksitas masalah yang ditangani oleh komputer, dibutuhkan bahasa pemrograman yang lebih handal. Para ahli kemudian menciptakan bahasa pemrograman yang lebih mudah pemberian intruksinya sehingga lahirlah bahasa pemrograman tingkat tinggi (*high level languange*). Yang termasuk dalam kelompok ini adalah bahasa pemrograman *Python* [21].

Python di anggap sebagai bahasa yang paling banyak digunakan di dalam bidang *Machine Learning* dan *Deep Learning* hal ini dikarenakan selain penulisan sintaksis yang mudah *Python* juga didorong oleh komunitas yang besar, selain itu *python* juga memiliki banyak *library* yang sangat mendukung *Machine Learning* dan *Deep Learning* berikut beberapa contoh *library python* yang dapat digunakan untuk *Machine Learning* dan *Deep Learning* seperti:

1. *Scikit Learn*

   Library ini menyediakan banyak algoritma pembelajaran tanpa pengawasan dan pengawasan. Itu dibangun di atas beberapa teknologi yang mungkin sudah di kenal, seperti *NumPy*, panda, dan *Matplotlib*

2. *Pandas*

   Library Python paling populer yang digunakan untuk analisis data dengan dukungan untuk struktur data yang cepat, fleksibel, dan ekspresif yang dirancang untuk bekerja pada data "relasional" atau "berlabel".*Pandas* hari ini adalah *library* yang tak terelakkan untuk menyelesaikan analisis data dunia nyata yang praktis dengan *Python.*

3. *Matplotbib*

   Library yang digunakan untuk visualisasi data. Visualisasi data memiliki peranan penting untuk memahami data secara lebih mendalam sebelum melakukan *data processing* dan melatihnya dalam program *machine learning* [18].

2.2.7 *Confusion Matrix*

Confusion Matrix adalah tabel yang digunakan untuk mengevaluasi kinerja dari suatu sistem klasifikasi atau model prediksi. Matriks inimemungkinkan kita untuk melihat seberapa baik model tersebut melakukan klasifikasi pada set data uji dengan membandingkan hasil prediksi dengan nilai sebenarnya [19]. Tabel 2.2 berikut menunjukan gambaran *confusion matrix*:

Tabel 2. 2 *Confusion Matrix*

		Skor Terkini	
		(Positive)	**(Negative)**
Skor Perkiraan	**(Positive)**	TP	FP
	(Negative)	FN	TN

Keterangan pada Tabel 2.2 terdapat empat nilai yang dikeluarkan yaitu:

1. *True Positive* (TP) adalah jumlah data positif yang diklasifikasikan dengan benar sebagai nilai positif.
2. *False Negative* adalah jumlah data negatif yang salah diklasifikasikan sebagai nilai positif.
3. *False Positive* adalah jumlah data positif yang salah diklasifikasikan sebagai nilai negatif.
4. *True Negative* (TN) adalah jumlah data negatif yang diklasifikasikan dengan benar sebagai nilai negatif.

Berikut ini adalah evaluasi yang dihasilkan dari metode *confusion matrix*:

1. *Accuracy*

   Merupakan presentase jumlah data yang dilakukan pada klasifikasi atau prediksi secara benar oleh algoritma.

   Rumus :

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \times 100\% \quad (2.2)$$

2. *Precision*

   Nilai dari ketepatan dari metode yang digunakan dalam klasifikasi. Nilai tersebut menunjukkan banyaknya data yang dapat terklasifikasi dikelas yang benar dalam beberapa pengujian.

Rumus :

$$Precission\ (positive) = \frac{TP}{(TP+FP)} \times 100\% \quad (2.3)$$

$$Precission\ (negative) = \frac{TN}{(TN+FN)} \times 100\% \quad (2.4)$$

3. *Recall*

   Nilai yang dapat mengukur hasil berapa presentase data yang terklasifikasikan dengan benar, adapun rumus yang digunakan untuk menghitung nilai *recall*.

   Rumus :

$$Recall\ (positive) = \frac{TP}{(TP+FN)} \times 100\% \quad (2.5)$$

$$Recall\ (negative) = \frac{TN}{(TN+FP)} \times 100\% \quad (2.6)$$

4. *F-1 Score*

   F-1 Score menggambarkan perbandingan rata-rata *precision* dan *recall* yang dibobotkan. *Accuracy* tepat kita gunakan sebagai acuan performansi algoritma jika dataset kita memiliki jumlah data *False Negatif* dan *False Positif* yang sangat mendekati *(symmetric).* Namun jika jumlahnya tidak mendekati, maka sebaiknya kita menggunakan *F1 Score* sebagai acuan.

   Rumus :

$$F1\ Score = 2 \times \frac{Recall \times Precission}{Recall + Precission} \quad (2.7)$$