BAB 2
LANDASAN TEORI

2.1 Diabetes Mellitus

Diabetes mellitus merupakan kelompok penyakit yang ditandai dengan tingginya kadar gula (*glukosa*) di dalam darah akibat gangguan pada metabolisme. *Glukosa* adalah sumber energi utama bagi tubuh, sementara insulin adalah hormon yang memungkinkan *glukosa* masuk ke dalam sel-sel tubuh yang digunakan sebagai energi. Dalam kasus diabetes mellitus, tubuh mengalami masalah dengan produksi insulin, penggunaan insulin, atau bahkan bisa jadi keduanya, yang mengakibatkan konsentrasi *glukosa* dalam darah menjadi tinggi.

Terdapat tiga jenis diabetes mellitus yang umum dikenal:

1) Diabetes Tipe 1

Diabetes tipe 1 merupakan bentuk diabetes autoimun di mana sistem kekebalan tubuh menyerang dan merusak sel-sel pankreas yang memproduksi insulin. Ini menyebabkan produksi insulin menjadi sangat terbatas atau bahkan berhenti sama sekali. Ini biasanya muncul pada usia muda, seringkali sebelum usia 30 tahun, meskipun bisa muncul juga pada segala usia. Penderita diabetes tipe 1 memerlukan suntikan insulin setiap hari untuk menjaga kadar *glukosa* dalam darah tetap normal. Penyebab pasti diabetes tipe 1 belum sepenuhnya dipahami, tetapi faktor genetik dan lingkungan diyakini berperan dalam perkembangannya.

2) Diabetes Tipe 2

Diabetes tipe 2 adalah bentuk diabetes yang lebih umum dan biasanya terjadi pada usia dewasa, meskipun semakin sering terjadi pada anak-anak dan remaja karena peningkatan obesitas. Pada diabetes tipe 2, tubuh tidak merespons insulin dengan baik (resistensi insulin) atau tidak memproduksi cukup insulin untuk memenuhi kebutuhan tubuh. Faktor risiko diabetes tipe 2 meliputi obesitas, pola makan tidak sehat, kurangnya aktivitas fisik, dan riwayat turunan diabetes pada keluarga.

Pengobatan untuk diabetes tipe 2 dapat melibatkan perubahan gaya hidup, obat-obatan, atau kombinasi keduanya. Pada tahap awal, penderita diabetes tipe 2 mungkin dapat mengendalikan kadar gula darah melalui perubahan gaya hidup dan makanan, tetapi dalam beberapa kasus, mungkin memerlukan obat antidiabetes atau insulin jika kondisinya memburuk.

3) Diabetes Gestasional

Diabetes gestasional adalah kondisi diabetes yang terjadi selama kehamilan. Ini terjadi karena peningkatan hormon saat kehamilan yang menyebabkan resistensi insulin. Diabetes gestasional biasanya muncul pada trimester kedua kehamilan dan bisa berlangsung hingga kelahiran. Biasanya, diabetes gestasional dapat diatasi dengan mengikuti rencana makan yang sehat dan aktif secara fisik. Namun, jika tidak ditangani dengan baik, dapat menyebabkan komplikasi pada ibu dan bayi. Setelah melahirkan, kebanyakan wanita yang mengalami diabetes gestasional akan kembali normal, namun mereka juga memiliki risiko lebih tinggi untuk terkena diabetes tipe 2 di masa mendatang.

Sementara diabetes tipe 1 dan tipe 2 adalah kondisi kronis yang berlangsung seumur hidup, diabetes gestasional terjadi selama kehamilan dan biasanya membaik setelah melahirkan. Semua jenis diabetes memerlukan manajemen dan perawatan yang tepat untuk mengendalikan kadar *glukosa* darah dan mencegah komplikasi.

2.2 *Data Mining*

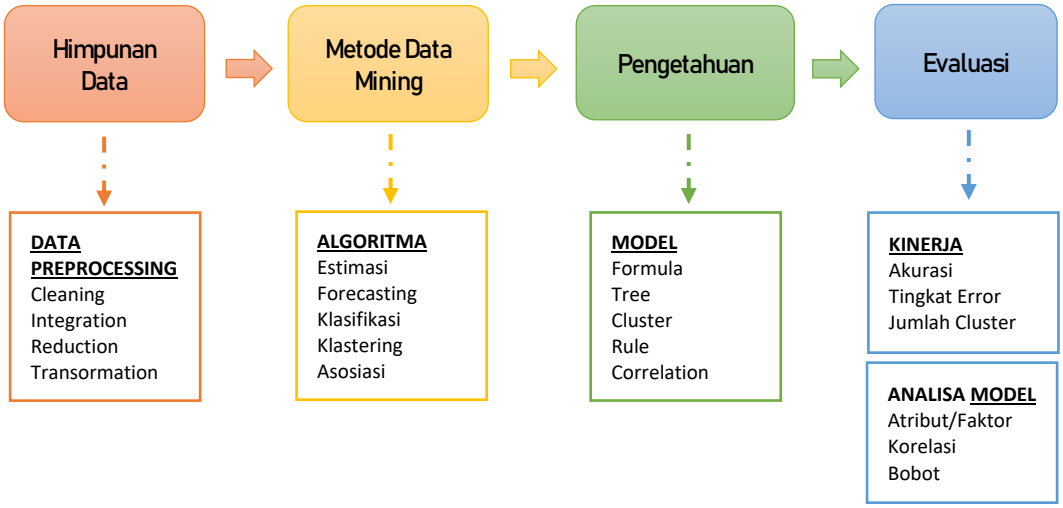
Menurut Larose [26], *data mining* mengacu pada proses penemuan pola, hubungan, atau informasi yang berharga dari kumpulan data yang besar dan kompleks. *Data mining* mencakup berbagai teknik dan algoritma statistik, matematika, kecerdasan buatan untuk mengeksplorasi data, mengidentifikasi pola, dan membuat prediksi atau model berdasarkan data yang ada. Tujuan utama dari *data mining* ini adalah untuk menggali pengetahuan tersembunyi dari data, membantu dalam pengambilan keputusan, memahami perilaku dari pengguna, atau membuat rekomendasi yang relevan.

Terdapat lima teknik *data mining* yang umum digunakan [26][27]: Estimasi, Forecasting, Klasifikasi, Klastering, dan Asosiasi. Dalam penerapannya, kelima teknik ini memiliki perbedaan: Estimasi merupakan proses memperkirakan nilai numerik berdasarkan data, Forecasting untuk memperkirakan nilai masa depan, Klasifikasi pengelompokan data ke kelas-kelas yang sudah ditentukan, Klastering pengelompokan data yang didasarkan pada kesamaan fitur, sementara Asosiasi mencari hubungan antara item atau peristiwa dari data.

Tabel 2.1 – *Role Data Mining* [27]

ROLE	ATTRIBUTE			LABEL		attribute correlation
	numeric	nominal	time-series	numeric	nominal	
Estimation	✓			✓		
Forecasting	✓		✓	✓		
Classification	✓	✓			✓	
Clustering	✓					
Association	✓					✓

Secara sederhana, penerapan *data mining* (*role*) [27] dapat dilakukan dengan cara melihat karakteristik dari *dataset* yang akan diolah. Untuk kategori *supervised-learning*, yang pengolahan datanya memerlukan *class/label*, dapat kita terapkan teknik estimasi, *forecasting*, ataupun klasifikasi. Sementara untuk klastering dan asosiasi, teknik ini tidak memerlukan label (*unsupervised-learning*), seperti yang ditunjukkan pada Tabel 2.1.

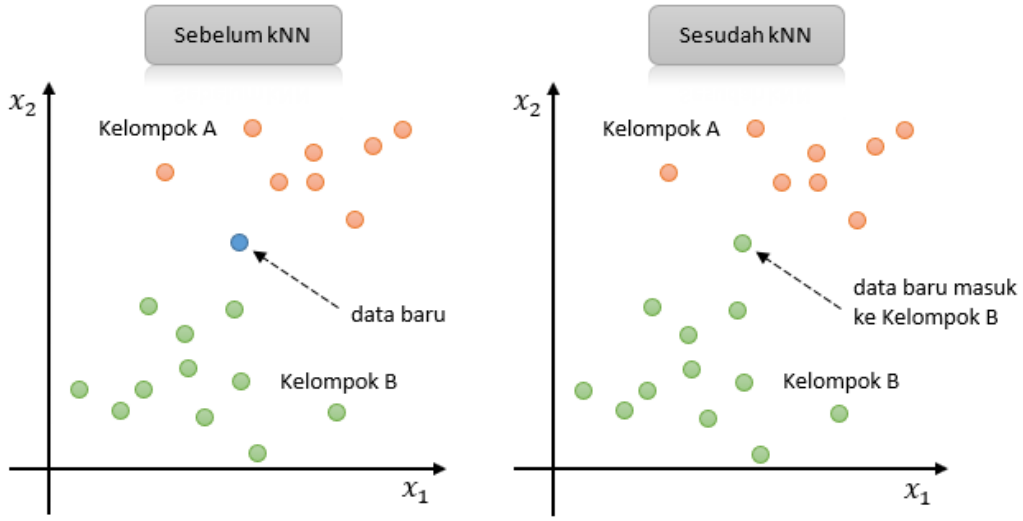


Gambar 2.1 – Proses *Data Mining* [27]

Data mining seringkali merupakan bagian dari proses yang lebih besar dalam analisis data yang dikenal sebagai "KDD" (*Knowledge Discovery in Databases*). Melibatkan beberapa tahapan yang dimulai dari pembersihan data (*data cleaning*), penerapkan model algoritma, evaluasi kinerja dan model, sampai interpretasi hasil (Gambar 2.1). Dalam penelitian ini penulis akan menerapkan teknik klasifikasi *data mining* menggunakan algoritma k-Nearest Neighbor (k-NN), Random Forest, dan Deep Learning untuk mengolah *dataset*.

2.2.1 k-Nearest Neighbor

k-Nearest Neighbor (atau, kNN) adalah algoritma pembelajaran berbasis instan (*instance-based*) yang digunakan dalam klasifikasi dan regresi. Prinsip Kerja kNN adalah mencari k tetangga terdekat dari data yang ingin diklasifikasi/diprediksi dan memanfaatkan mayoritas voting atau rata-rata nilai dari tetangga tersebut untuk mengambil keputusan.

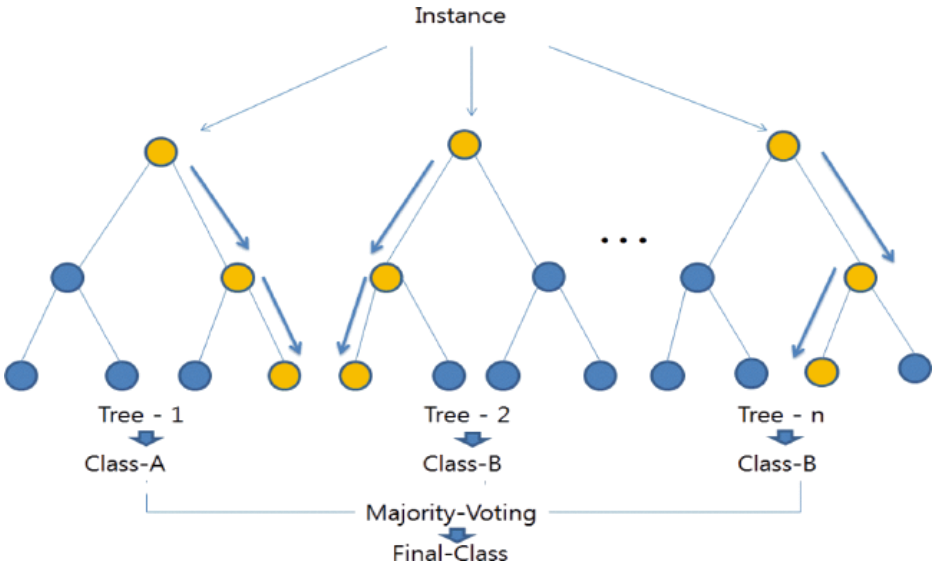


Gambar 2.2 – Prinsip dasar dari k-Nearest Neighbor

kNN tidak melatih model seperti halnya algoritma lainnya; sebaliknya, kNN menyimpan semua data latih sebagai "*knowledge base*". Karena itu kNN kurang efisien untuk data latih yang besar karena ia harus menghitung jarak ke seluruh data latih. Meski demikian kNN dapat memperlihatkan fitur interpolasi lokal yang baik karena prediksi didasarkan pada tetangga terdekat.

2.2.2 Random Forest

Random Forest (RF) adalah algoritma *ensemble learning* yang digunakan untuk mengklasifikasi, regresi, dan tugas-tugas lainnya. Prinsip Kerja dari Random Forest adalah membangun banyak pohon (*tree*) keputusan dari *subsample data* dengan menggunakan *bootstrapping* dan menggabungkan hasil prediksi dari semua pohon ini melalui *voting* mayoritas (klasifikasi) atau rata-rata (regresi). Random Forest cenderung mengurangi overfitting karena kombinasi dari banyak pohon. Meskipun lebih efisien daripada kNN, Random Forest masih memerlukan lebih banyak waktu pelatihan daripada beberapa algoritma lainnya. Selain itu, Random Forest biasanya menunjukkan fitur interpolasi global karena menggabungkan prediksi dari banyak pohon (*decision tree*).

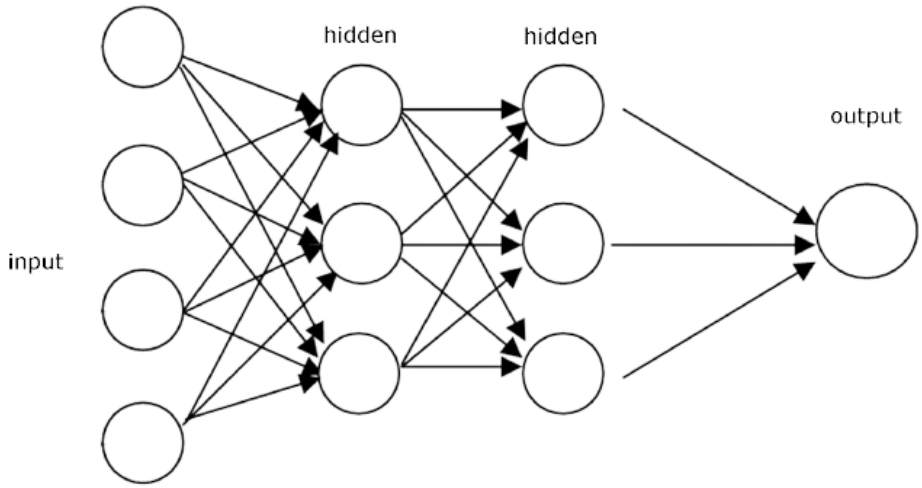


Gambar 2.3 – Random Forest bekerja dengan membangun beberapa pohon keputusan (*decision tree*)

2.2.3 Deep Learning

Deep Learning (DL) adalah sub-bidang dari *Machine Learning* yang menggunakan model *neural network* untuk mempelajari representasi data yang kompleks. Deep Learning dapat secara otomatis mengekstraksi representasi fitur dari data, yang memungkinkannya untuk menangani data yang sangat kompleks dan berukuran besar. Untuk pelatihan model yang efektif, Deep Learning memerlukan jumlah data latih yang besar. Dengan demikian, pelatihan model Deep Learning memerlukan

sumber daya komputasi yang kuat, seperti GPU dan TPU, sehingga membutuhkan waktu dan biaya yang signifikan.



Gambar 2.4 – Model Deep Learning pada Rapidminer yang digunakan adalah jenis jaringan *Multi-layer Feed-forward*

Deep Learning pada Rapidminer, yang digunakan dalam penelitian ini, merupakan jenis model jaringan saraf *multi-layer feed-forward* yang dilatih dengan penurunan gradien stokastik menggunakan propagasi pembalik. *Stochastic Gradient Descent* (SGD) ini merupakan metode optimisasi yang sering digunakan dalam pelatihan model jaringan saraf tiruan, termasuk pada jaringan *multi-layer feed-forward*. SGD digunakan untuk menemukan nilai parameter model yang dapat meminimalkan fungsi kerugian (*loss function*). Dalam konteks jaringan saraf, *back-propagation* merupakan algoritma yang umumnya digunakan untuk menghitung gradien fungsi kerugian ini terhadap setiap parameter model.

Dalam memilih algoritma untuk suatu tugas tertentu, sangat penting untuk mempertimbangkan karakteristik *dataset*, ukuran data latih, kompleksitas masalah, dan sumber daya komputasi yang tersedia. Masing-masing algoritma memiliki kelebihan dan kelemahan yang berbeda, dan pemilihan algoritma yang tepat dapat sangat mempengaruhi kinerja dan efisiensi pemodelan.

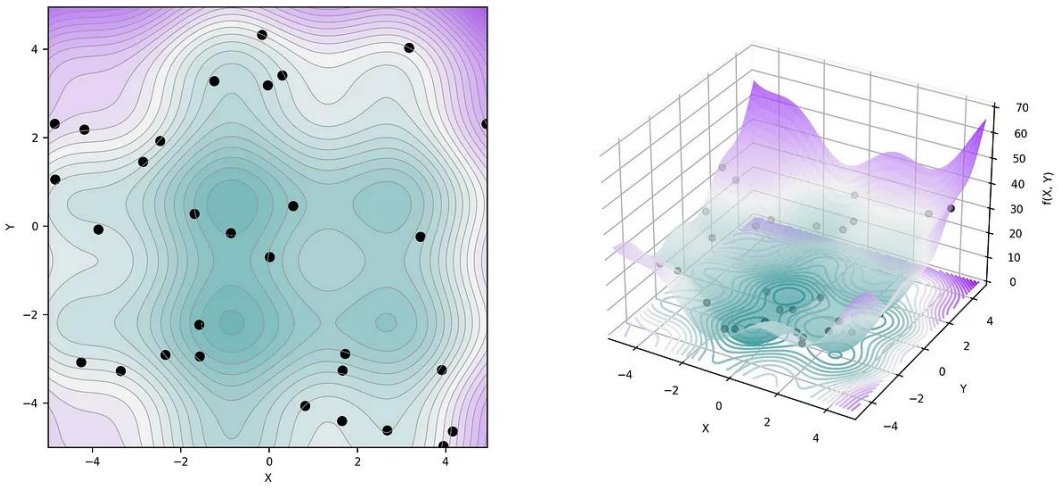
2.3 Particle Swarm Optimization

Particle Swarm Optimization (PSO) adalah salah satu metode optimisasi yang dapat digunakan untuk meningkatkan kinerja algoritma klasifikasi dalam konteks *data mining*. PSO terinspirasi dari perilaku kelompok burung yang mencari makanan (Gambar 2.5).

Gambar 2.5 – Kawanan burung jalak [57]

Ini adalah algoritma optimisasi yang menggambarkan sekumpulan partikel, bagaikan kelompok burung, yang bergerak bersama untuk mencari solusi optimal dalam ruang pencarian. Bila kita analogikan, beberapa prinsip dasar dari PSO ini mencakup:

1) **Partikel (Burung):** Dalam PSO, setiap partikel mewakili satu individu dalam populasi. Tiap partikel memiliki posisi dan kecepatan yang memungkinkannya bergerak dalam ruang pencarian.
2) **Solusi Optimal (Makanan):** Tujuan dari PSO adalah menemukan "*makanan*," yang dalam konteks optimisasi adalah solusi optimal dari masalah yang sedang diselesaikan.
3) ***Swarm* (Interaksi Kawanan):** Partikel dalam PSO berinteraksi satu sama lain. Mereka berkomunikasi dan saling berbagi informasi mengenai solusi yang mereka temukan, mirip seperti kelompok burung yang bergerak bersama untuk mencari makanan.



Gambar 2.6 – Inisialisasi awal partikel-partikel PSO [28]

Awal proses, PSO memulai dengan menginisialisasi sejumlah n populasi partikel (perhatikan ilustrasi Gambar 2.6), di mana setiap partikel memiliki posisi atau nilai-nilai yang diacak dalam rentang domain nilai yang akan dieksplorasi. Setiap partikel akan menyimpan informasi *personal best*-nya (atau posisi terbaik yang pernah ia temukan saat melakukan eksplorasi) dan juga *global best* (posisi terbaik yang pernah ditemukan oleh seluruh kawanan). Posisi-posisi terbaik ini membentuk pengetahuan kolektif atau "*swarm knowledge*", dan menjadi informasi yang digunakan oleh partikel-partikel ini untuk membimbing pergerakan mereka menemukan solusi optimal (fungsi objektif), atau “makanan” tadi.

~ Persamaan 1 ~

$$v_i^{t+1} = \underbrace{\omega \times v_i^t}_{\text{inertia term}} + \underbrace{c_1 \times r_1 \times (Pb_i - x_i^t)}_{\text{cognition term}} + \underbrace{c_2 \times r_2 \times (Gb - x_i^t)}_{\text{social term}}$$

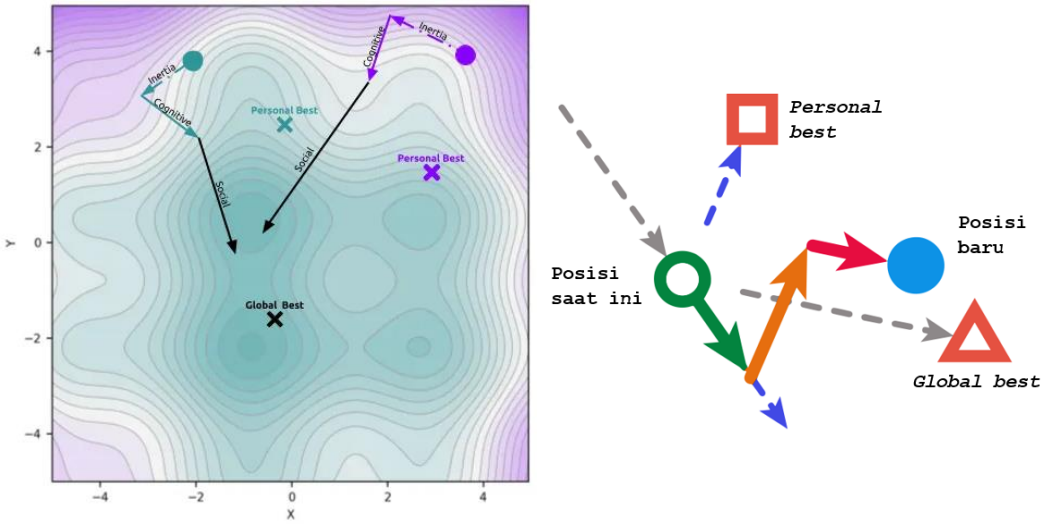
~ Persamaan 2 ~

$$x_i^{t+1} = x_i^t + v_i^{t+1}$$

Dalam tiap kali iterasi, setiap partikel bergerak di dalam ruang pencarian berdasarkan pengalaman pribadinya dan juga pengalaman kawanannya. Gerakan ini diatur oleh pembaruan posisi (Persamaan 2) dan kecepatan partikel (Persamaan 1) [29]. Kecepatan (*velocity*) tiap partikel di*update* dengan mempertimbangkan tiga faktor utama: *inertia term*, *cognition term*, dan *social term* (Gambar 2.7).

yang mana,

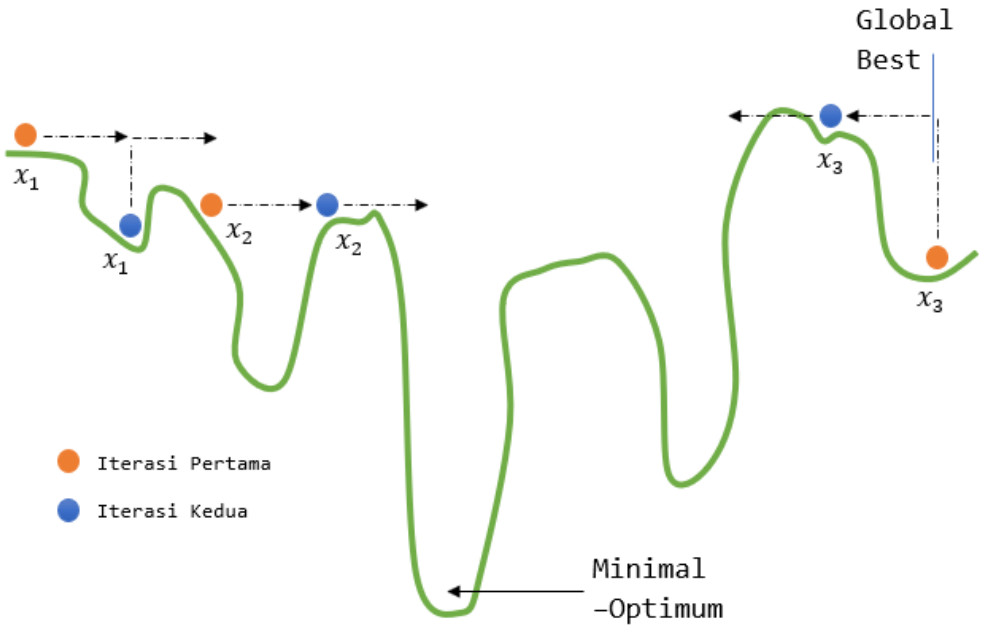
v_i^{t+1}	:	adalah kecepatan partikel ke-i pada iterasi berikutnya $(t+1)$
v_i^t	:	kecepatan partikel ke-i pada iterasi saat ini (t)
Pb_i	:	*personal best* / posisi terbaik yang pernah dicapai partikel ke-i
Gb	:	*global best* atau posisi terbaik yang pernah dicapai oleh seluruh kawanan (*swarm*) atau kelompok partikel
x_i^t	:	posisi saat ini dari partikel ke-i
ω	:	faktor *inertia* menentukan seberapa besar kemampuan partikel mempertahankan arah gerak sebelumnya
c_1	:	faktor *cognition* yang mengatur seberapa besar partikel cenderung bergerak ke arah *personal best*-nya sendiri
c_2	:	faktor *social* yang mengarahkan partikel menuju *global best* yang ditemukan oleh *swarm*
r_1 dan r_2	:	nilai acak antara 0 dan 1 yang digunakan untuk menambahkan elemen stokastisitas pada proses optimisasi



Gambar 2.7 – *Inertia term*, *cognition term*, dan *social term* partikel [28][30]

Inertia term (diwakili oleh panah hijau) menentukan seberapa besar partikel mempertahankan arah geraknya yang sebelumnya. *Cognition term* (panah jingga) mengatur seberapa besar partikel cenderung bergerak ke arah *personal best*-nya sendiri, sementara *social term* (panah merah) mengarahkan partikel menuju *global best* yang ditemukan oleh seluruh kawanan. Nilai ω, c_1, dan c_2 adalah parameter-parameter penting yang mengatur *seberapa besar* nilai pengaruh dari masing-masing *term* ini dalam menentukan pergerakan partikel. Sehingga gabungan dari

ketiganya menghasilkan vektor yang akan menentukan posisi partikel selanjutnya (ikon bulat biru). Tapi meski partikel ada kecenderungan bergerak ke arah *global best* yang telah ditemukan kawanan, mereka juga tetap memperhatikan pencapaian terbaik mereka sendiri atau *personal best*-nya. Alasan di balik ini adalah, bahwa meskipun *global best* mungkin menunjukkan arah yang menjanjikan, tidak selalu terjadi bahwa itu merupakan langkah terbaik bagi setiap partikel. Untuk memahami ini mari kita perhatikan pergerakan partikel dalam ilustrasi yang lebih sederhana berikut (Gambar 2.8).

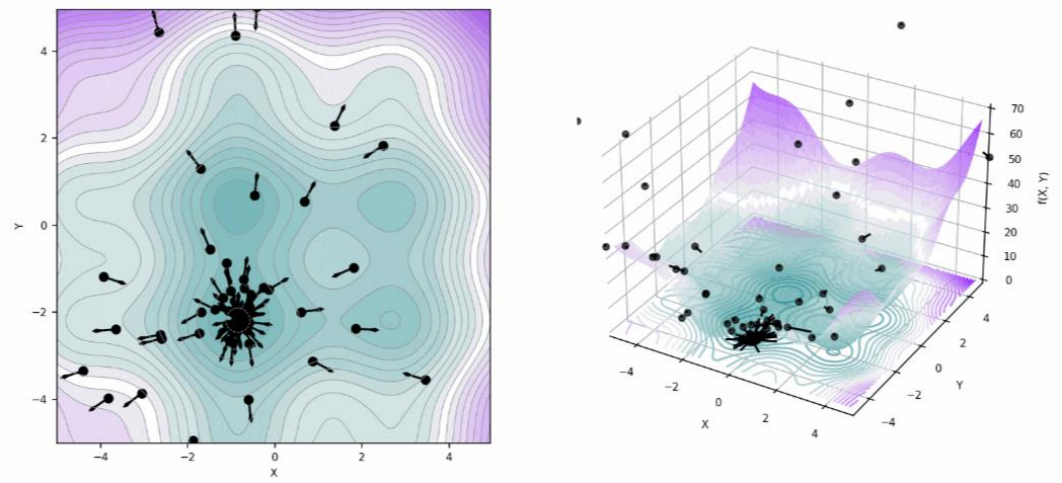


Gambar 2.8 – Arah pergerakan partikel [31]

Personal best masing-masing partikel bisa saja lebih dekat ke tujuan daripada *global best.* Pada kasus minimal optimum seperti Gambar 2.8, bila kita perhatikan posisi partikel x_2 bukan *global best* –baik di iterasi pertama ataupun kedua, tapi posisi x_2 ini sebenarnya lebih dekat ke tujuan. Dalam konteks ini, setiap partikel akan berfungsi sebagai agen yang mencoba menavigasi ruang pencarian: Mencari tahu apakah mereka lebih dekat ke tujuan secara individual, atau bila-ada-arah-yang-lebih-baik yang telah ditunjukkan oleh kawanannya (*swarm*). Sehingga PSO menciptakan sistem adaptif di mana setiap partikel ini dapat mempertimbangkan pengetahuan pribadinya dan informasi global untuk menemukan solusi optimal secara kolaboratif. Dengan demikian, meskipun setiap partikel ini tidak memiliki

pengetahuan penuh tentang lingkungannya, akan tetapi interaksi diantara mereka memungkinkan algoritma untuk menemukan solusi yang efektif melalui eksplorasi dan eksploitasi informasi yang tersedia.

Setelah pembaruan posisi dan kecepatan, partikel dievaluasi ulang dengan menghitung nilai fungsi objektif baru untuk posisi yang telah diperbarui. Fungsi objektif adalah tujuan optimisasi yang ingin dicapai dan digunakan untuk mengukur sejauh mana solusi saat ini memenuhi kriteria optimisasi. Tujuannya adalah untuk mencapai nilai fungsi objektif optimal dengan meminimalkan atau memaksimalkan nilai sesuai dengan kebutuhan masalah. Meminimalkan atau memaksimalkan nilai ini bergantung pada masalah yang dihadapi. Misalnya, dalam masalah optimisasi biaya kita mungkin ingin meminimalkan fungsi objektif ini, sementara dalam masalah optimisasi keuntungan, kita mungkin ingin memaksimalkannya. Dalam penelitian ini fungsi objektifnya adalah akurasi.



Gambar 2.9 – Partikel-partikel bergerak menuju solusi terbaik [28]

Setelah menghitung nilai fungsi objektif, partikel juga diberi skor berdasarkan hasil dari fungsi objektif tadi. Skor ini disebut sebagai skor fitness, dan digunakan untuk mengukur seberapa baik partikel dalam mencapai solusi optimal. Partikel dengan skor fitness yang lebih baik akan memiliki kinerja yang lebih baik dalam pencarian. Jadi intinya, fungsi objektif adalah tujuan matematis yang ingin dicapai dalam optimisasi PSO ini, sementara skor *fitness*-nya digunakan untuk mengukur seberapa baik partikel-partikel mendekati pencapaian tujuan ini.

PSO berakhir ketika iterasi telah maksimal, atau kondisi berhenti tertentu tercapai. Selama iterasi, PSO cenderung mengarahkan partikel ke arah solusi yang lebih baik secara kolektif. Hasil akhir adalah posisi yang terbaik, yang berhasil ditemukan selama iterasi yang memiliki nilai fungsi objektif terbaik.

2.4 Evaluasi Kinerja Model Klasifikasi

Evaluasi dalam *data mining* adalah proses yang melibatkan pengukuran kinerja dan evaluasi model atau algoritma yang digunakan dalam analisis data. Evaluasi adalah langkah penting dalam siklus *data mining* dan membantu kita memahami sejauh mana model atau algoritma kita bekerja dengan baik dalam menghasilkan prediksi atau informasi yang berharga dari data.

Confusion Matrix dan kurva ROC adalah dua teknik yang penting dalam mengukur kinerja dari model klasifikasi, dan masing-masing menyajikan bentuk informasi yang berbeda.

2.4.1 Confusion Matrix

Confusion matrix adalah tabel yang digunakan untuk mengevaluasi kinerja model klasifikasi. Ini menyajikan jumlah hasil prediksi yang benar dan yang salah pada model suatu *dataset*. Confusion matrix terdiri dari empat sel, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

Tabel 2.2 – Nilai kondisi/prediksi dari Confusion Matrix

CONFUSION MATRIX		Diprediksi?	
		Positif	Negatif
Kenyataan terhadap Hasil (T/F)	*True* – Benar	TP	TN
	False – Salah	FP	FN

- *True Positive* (TP): adalah jumlah dari nilai-nilai yang diprediksi bernilai positif dan benar–positif (*true*), karena kenyataan (*real*-nya) nilai-nilai ini *memang* bernilai positif. Sehingga bila ada yang diprediksi negatif, maka tidak akan dimasukkan ke dalam kumpulan karena *real*-nya positif.

- *True Negative* (TN): Kenyataan-nya negatif, diprediksi negatif.
- *False Positive* (FP): Berarti kenyataan-nya negatif tapi diprediksi positif, sehingga hasilnya *false* (jumlah dari nilai-nilai yang salah).
- *False Negative* (FN): Kenyataan-nya positif, tapi diprediksi negatif.

Tiga perhitungan yang umum digunakan dalam metrik ini: Akurasi, presisi, dan *recall*. Untuk mengukur sejauh mana sebuah model berhasil memprediksi dengan benar keseluruhan kelas, kita gunakan perhitungan akurasi.

$$akurasi = \frac{TP + TN}{TP + TN + FP + FN}$$

Akurasi bisa menjadi metrik yang baik ketika kelas positif dan negatif memiliki distribusi yang seimbang. Namun, akurasi tidak selalu mencerminkan kinerja yang baik jika kelas-kelas tersebut tidak seimbang.
Presisi mengukur sejauh mana prediksi positif model yang benar (*true*).

$$presisi = \frac{TP}{TP + FP}$$

Presisi berguna ketika kita ingin menghindari membuat prediksi positif yang salah. Ini bermanfaat dalam situasi di mana biaya atau dampak *false positive* tinggi.

Recall (Sensitivitas atau *True Positive Rate*) mengukur sejauh mana model mampu mendeteksi atau menangkap *instance* positif.

$$recall = \frac{TP}{TP + FN}$$

Recall berguna ketika fokus utama adalah mendeteksi sebanyak mungkin *instance* positif. Ini bermanfaat dalam situasi di mana kita ingin menghindari *false negative*.

Ketiga metrik ini saling terkait dan bisa saling bertentangan tergantung pada konteks dan tujuan aplikasi. Misalnya, meningkatkan presisi dapat menyebabkan penurunan *recall*, dan sebaliknya. Oleh karena itu, pemilihan metrik evaluasi harus mempertimbangkan kebutuhan spesifik dari tugas klasifikasi yang dihadapi.

2.4.2 Kurva AUC-ROC

Kurva ROC adalah grafik yang digunakan untuk mengukur kinerja dari model klasifikasi pada berbagai tingkat ambang batas (*threshold*) dalam konteks masalah klasifikasi biner. Ini mengilustrasikan hubungan antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR) seiring dengan variasi ambang batas.

Bila *Confusion Matrix* memberikan pemahaman lebih rinci tentang kinerja suatu model klasifikasi melalui tabel yang menunjukkan jumlah prediksi yang benar dan yang salah. Maka, kurva ROC (*Receiver Operating Characteristic*) memberikan gambaran visual tentang kinerja model klasifikasi dengan memplot hubungan antara tingkat *true positive* (TP) dan *false positive* (FP) pada berbagai nilai *threshold*.

Kurva ROC umumnya memiliki sumbu X yang menunjukkan tingkat *false positive rate* (1 - *specificity*) dan sumbu Y yang menunjukkan tingkat *true positive rate* (sensitivitas). Kurva ini membantu mengukur sejauh mana model dapat membedakan antara kelas positif dan negatif. Selain itu, area di bawah kurva ROC (AUC-ROC) sering digunakan sebagai metrik untuk menilai kinerja model secara keseluruhan. Semakin besar nilai AUC-ROC (dalam rentang 0 hingga 1), semakin baik model tersebut dalam membedakan antara kelas positif dan negatif.

Tabel 2.3 – Klasifikasi level nilai AUC-ROC [32][33]

RANGE AUC-ROC	CLASSIFICATION
0.9 – 1.0	*Excellent*
0.8 – 0.9	*Good*
0.7 – 0.8	*Fair*
0.6 – 0.7	*Poor*
< 0.6	*Failure*

Dengan menggunakan kurva AUC-ROC, kita dapat memilih *threshold* yang sesuai dengan kebutuhan aplikasi atau toleransi terhadap *false positive* dan *false negative*. Dengan demikian secara keseluruhan, kedua metode: *Confusion Matrix* dan kurva AUC-ROC memberikan wawasan yang berharga tentang kinerja model klasifikasi, baik dalam tabel maupun secara visual.

2.4.3 Uji Beda t-Test

Dalam pengujian t-test, Hipotesis Nol (H_0) menyatakan bahwa tidak ada perbedaan yang signifikan antara dua kondisi atau algoritma yang dibandingkan. Sebaliknya Hipotesis alternatif (H_a) menyatakan: ada perbedaan yang signifikan.

- Jika nilai-$p \leq$ tingkat signifikansi, kita menolak hipotesis nol. Artinya, kita menyimpulkan bahwa ada perbedaan yang signifikan kondisi atau algoritma yang dibandingkan.
- Jika nilai-$p >$ tingkat signifikansi, maka kita gagal menolak hipotesis nol. Artinya, kita tidak memiliki cukup bukti statistik untuk menyatakan bahwa ada perbedaan signifikan.

Tingkat signifikansi (*alpha*) yang umum digunakan adalah 0.05 atau 5%. Nilai ini menjadi acuan umum, tetapi pada beberapa analisis statistik atau bidang penelitian tertentu, tingkat signifikansi yang berbeda mungkin digunakan, seperti 0.01 (1%) atau 0.10 (10%).

Jadi, bila dalam penelitian ini hasil uji t-test nanti memberikan nilai-p yang lebih kecil dari atau sama dengan *alpha* (0.05), maka kita akan menolak hipotesis nol dan menyimpulkan bahwa ada perbedaan signifikan antara algoritma-algoritma yang dibandingkan. Nilai 0.05 disini adalah batasan atau ambang batas yang akan kita gunakan sebagai tingkat signifikansi.

2.5 Beberapa Kajian Penelitian Yang Pernah Dilakukan

Berikut beberapa *dataset* diabetes *public* yang umum digunakan dalam berbagai penelitian (Tabel 2.4). Diurutkan berdasarkan tahun rilisnya: *Dataset* Pima Indian masih merupakan *dataset* yang paling umum dan paling banyak digunakan hingga saat ini, meski telah dirilis dari hampir setengah abad yang lalu. Sampai dengan *dataset* Sylhet yang baru dirilis di tahun 2020.

Dataset BIT Mesra terbanyak ketiga yang memiliki jumlah keberagaman atribut, setelah *dataset* 130 US Hospitals dan BRFSS Survey Data. Keragaman atribut ini tentu mempengaruhi kinerja dari model yang akan diprediksi dan juga efisiensi komputasi yang dibutuhkan [34][35].

Tabel 2.4 – *Dataset* yang umum digunakan dalam penelitian diabetes

NO	DATASET	REC | ATR	SOURCE	AUTHORS
1	Pima Indians – 1988	768 | 9	National Institute of Diabetes and Digestive and Kidney Diseases. *https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database*	Jack W. Smith, J.E. Everhart, W.C. Dickson, W.C. Knowler, and R.S. Johannes
2	130 US Hospitals – The data set represents 10 years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks.	101.766 | 47	Center for Clinical and Translational Research, Virginia Commonwealth University. *https://archive.ics.uci.edu/ml/datasets/Diabetes+130-S+hospitals+for+years+1999-2008*	Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, and John N. Clore
3	BRFSS Survey Data – 2015	253.680 | 22	The Centers for Disease Control and Prevention (CDC), - US *https://www.kaggle.com/datasets/alexteboul/ diabetes-health-indicators-dataset*	The Centers for Disease Control and Prevention (CDC), - US
4	Hospital Frankfurt – 2000	2000 | 9	Hospital Frankfurt, Hesse - Germany. *https://www.kaggle.com/datasets/johndasilva/diabetes*	John Da Silva
5	BIT Mesra – 2019	952 | 18	Department of Computer Science and Engineering, BIT Mesra, Ranchi-835215, - India. *https://www.kaggle.com/datasets/tigganeha4/diabetes-dataset-2019*	Neha Prerna Tigga and Dr. Shruti Garg
6	Sylhet – 2020	520 | 17	Direct questionnaires from the patients of Sylhet Diabetes Hospital in Sylhet, - Bangladesh. *https://archive.ics.uci.edu/dataset/529/early+stage+diabetes+risk+prediction+dataset*	M. Faniqul Islam, Rahatara Ferdousi, Sadikur Rahman, Humayra Yasmin Bushra

Lebih lanjut, *dataset* Pima Indian merupakan *dataset* yang dikumpulkan dari populasi Suku Indian Pima (Akimel O'odham). Suku ini merupakan kelompok suku asli Amerika yang mendiami wilayah Arizona dan Sonora Amerika Utara. Populasi *sample* yang diambil hanya terdiri dari wanita saja. Berbeda dengan *dataset* Sylhet dan BIT Mesra, yang memiliki populasi *sample* yang terdiri dari pria dan wanita. *Dataset* Sylhet ini dikumpulkan dari pasien rumah sakit di Bangladesh, sementara *dataset* BIT Mesra berasal dari Ranchi India. Untuk *dataset* Hospital Frankfurt, atribut-atributnya sama persis dengan atribut yang ada pada *dataset* Pima Indian. Hanya saja jumlah *record* datanya lebih banyak, lebih dari 2 kali lipatnya.

Tabel 2.5 – Kajian dari beberapa penelitian yang pernah dilakukan

NO	PENELITI	ALGORITMA	DATASET USED | METODE OPTIMASI	AKURASI
1	Raghavendra S, dkk. 2020. [8]	Naive Bayes, k-Nearest Neighbor, Logistic Regression, Neural Network, Random Forest	DS. 1 | Forward Selection and Backward Elimination	NB=78.69%, kNN=80.51%, LR=82.81%, NN=83.11%, RF=84.10%
2	Widya A, dkk. 2021. [14]	Support Vektor Machine, Naive Bayes, Random Forest	DS. 6 | –	SVM=94.80%, NB=91.92%, RF=97.88%
3	Ali Yasar. 2021. [17]	Decision Tree, Random Forest, Support Vector Machine, k-Nearest Neighbor, Feed Forward Neural Networks (FFNN)	DS. 6 | Particle Swarm Optimization (PSO), Tree Seed Algorithm (TSA), Crow Search Algorithm (CSA), Slime Mould Algorithm (SMA), and Artificial Bee Colony (ABC)	PSO + SVM = 97.50%, TSA + SVM = 96.15%, CSA + FFNN = 99.04%, SMA + FFNN = 94.23%, ABC + SVM = 96.73%
4	Jobeda J. Khanam, dkk. 2021. [10]	Decision Tree, k-Nearest Neighbor, Random Forest, Naive Bayes, Adaboost, Linear Regression, Support Vector Machine	DS. 1 | Feature selection: Remove atribut under 0,2 coefficient correlation with output	DT=74.24%, kNN=75.10%, RF=74.96%, NB=75.53%, AB=73.96%, LR=76.82%, SVM=76.82%

NO	PENELITI	ALGORITMA	DATASET USED | METODE OPTIMASI	AKURASI
5	Xiaohua Li, dkk. 2021. [36]	k-Nearest Neighbor	DS. 1 | Genetic algorithm (GA), K-means, Particle Swarm Optimization (PSO), Harmony (HR)	GA-Kmeans + kNN =88.02%, GA-PSO-Kmeans + kNN =89.64%, HR-Kmeans + kNN =91.65%,
6	Aishwarya Jakka, dkk. 2019. [37]	K-Nearest-Neighbor, Decision Tree, Naïve Bayes, Support Vector Machine, Logistic Regression, Random Forest	DS. 1 | –	kNN=73.43%, DT=70.31%, NB=75.52%, SVM=65.63%, LR=77.60%, RF=74.30%
7	Huma Naz, dkk. 2020. [25]	Artificial Neural Network, Naive Bayes, Decision Tree, Deep Learning	DS. 1 | Data validation using Shuffled Sampling	ANN=90.34%, NB=76.33%, DT=96.62%, DL=98.07%
8	Ti'jay Goudjerkan, dkk. 2019. [38]	Random Forest Gini, Convolutional Neural Network, Recurrent Neural Network, Multilayer Perceptron,	DS. 2 | –	RFG=94%, CNN=92%, RNN=81%, MLP=95%
9	Zahid Ullah, dkk. 2022. [39]	k-Nearest-Neighbor, Random Forest, XGBoost, Bagging, AdaBoost	DS. 3 | –	kNN=98.36%, RF=95.59%, XGBoost=95.02%, Bagging=94.66%, AdaBoost=94.35%,
10	Mahendra K. Gourisaria, dkk. 2022. [40]	Logistic Regression, k-Nearest-Neighbor, Support Vector Machine, Naive Bayes, Decision Tree, Random Forest, Artificial Neural Network	DS. 4 dan DS. 6 | Feature Extraction using: Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA)	kNN=98.20% (DS. 4) RF=99.20% (DS. 6)

Dengan mempelajari jurnal-jurnal dari penelitan yang dilakukan sebelumnya (Tabel 2.5), kita bisa melihat bahwa ada banyak sekali topik bahasan mengenai diabetes ini. Sementara semua pengujian yang dilakukan ini menerapkan berbagai macam varian algoritma klasifikasi yang ada, *dataset* yang digunakan pun beragam

(lihat Tabel 2.4). Raghavendra S, dkk [8] berhasil meningkatkan akurasi model mengunakan metode Forward Selection and Backward Elimination terhadap lima algoritma. Akurasi tertinggi pada algoritma Random Forest (RF) sebesar 84,10%. Hasil ini lebih baik bila dibandingkan dengan penelitian yang dilakukan [10] dengan akurasi RF sebesar 74,96% yang menerapkan fitur seleksi dengan cara membuang semua atribut yang hanya memiliki koefisian korelasi terhadap *output* kurang dari 0,2; atau penelitian yang dilakukan [37] yang tidak menerapkan metode optimasi apapun -dengan perolehan akurasi RF hanya sebesar 74,30%. Semua penelitian ini diujikan pada *dataset* yang sama: Pima Indians.

Meski demikian metode yang diterapkan oleh [10] ini mungkin lebih cocok dengan menggunakan algoritma kNN, dengan akurasi sebesar 75,10%, lebih tinggi dari penelitian [8] yang hanya memperoleh akurasi kNN sebesar 80,51%. Tapi [36] masih bisa meningkatkan lagi akurasi kNN ini menjadi 89,64% dengan menerapkan metode optimasi PSO.

Kemudian ada penelitian yang dilakukan [14], memperoleh akurasi 97,88% menggunakan RF. Akan tetapi hasil ini diujikan pada *dataset* yang berbeda: *dataset* Sylhet, dan tanpa menggunakan metode optimasi apapun. Lalu penelitian [40] yang memperoleh akurasi RF 99,20% dengan menerapkan metode fitur ekstraksi PCA, juga pada *dataset* Sylhet. Selain RF, [14] juga menggunakan algoritma Suport Vector Machine (SVM) pada *dataset* ini, dengan perolehan akurasi sebesar 94,80%. Lalu [17] meningkatkan akurasinya menjadi 97,50% dengan optimasi PSO.

Beberapa peneliti lainnya menggunakan *dataset* yang berbeda, atau *dataset* yang sama tetapi dengan algoritma yang lain. Seperti [38], memperoleh akurasi tertinggi 95% menggunakan Multilayer Perceptron pada *dataset* 130 US Hospitals. Kemudian [39] yang memperoleh akurasi tertinggi 95,59% menggunakan RF pada *dataset* BRFSS Survey Data. Atau pun penelitian yang dilakukan [25] pada *dataset* Pima Indians tapi dengan algoritma Deep Learning (akurasi algoritma tertinggi) dengan akurasi 98,07%.