BAB 2
TINJAUAN PUSTAKA

2.1 Penyakit Diabetes

Diabetes melitus adalah penyakit gangguan metabolik yang ditandai oleh kenaikan gula darah akibat penurunan sekresi insulin oleh sel beta pankreas dan atau gangguan fungsi insulin (resistensi insulin) [10]. Biasanya penyakit ini ditandai dengan terjadinya hiperglikemia dan gangguan metabolisme karbohidrat, lemak, dan protein yang dihubungkan dengan kekurangan secara absolut atau relatif dari kerja dan atau sekresi insulin [10]. Diabetes merupakan suatu penyakit yang menahun yang ditandai dengan kadar glukosa pada darah (gula darah) melebih dari nilai normal yaitu 200 mg/dl, dan kadar gula darah pada saat puasa di atas atau sama dengan 126 mg/dl [11].

Diabetes Melitus sering kali dikaitkan dengan gangguan sistem *mikrovaskular* dan *makrovaskular*, gangguan *neuropatik*, dan *lesi dermopatik* [12]. Adapun contoh penyakitnya antara lain nefropati, diabetik retinopati (kebutaan), neuropati, penyakit jantung koroner (PJK), gagal jantung kongestif, dan stroke [10]

Penyakit diabetes dapat menyerang siapa saja, segala lapisan umur dan tanpa memandang status sosial ekonomi. Jumlah penderita diabetes di Indonesia terus meningkat sampai tahun 2019 terakhir Indonesia menempati posisi ketujuh terbanyak penderita diabetes. Berdasarkan data dari WHO, pada tahun 2016 terjadi sebanyak 48.300 orang yang berusia 30-69 tahun mengidap penyakit diabetes [2].

2.2 Klasifikasi Penyakit Diabetes

Dari sekian banyak kasus diabetes ini, berdasarkan tipenya penyakit diabetes ini dibedakan menjadi tiga tipe, yakni :

1. Diabetes tipe-1

   Diabetes tipe-1 adalah penyakit autoimun kronis yang berhubungan dengan kehancuran selektif sel beta pankreas yang memproduksi insulin.

Timbulnya penyakit klinis merupakan tahap akhir dari kerusakan sel beta yang mengarah ke tipe 1 [13].

2. Diabetes tipe-2

   Diabetes tipe 2 adalah penyakit gangguan metabolik yang di tandai oleh kenaikan gula darah akibat penurunan sekresi insulin oleh sel beta pankreas dan atau gangguan fungsi insulin [10].

3. Diabetes gestasional

   Diabetes gestasional adalah kehamilan normal yang disertai peningkatan resistensi insulin. Faktor risiko diabetes gestasional yaitu riwayat keluarga, obesitas, dan glikosuria [14].

2.3 Gejala Klinis

Berikut ini merupakan gejala yang umumnya dirasakan oleh penderita diabetes menurut Tobing [15]:

1) Buang air kecil menjadi lebih sering. Tingginya kadar gula dalam darah yang dikeluarkan melalui ginjal selalu diiringi oleh cairan dari tubuh sehingga air kencing yang dikeluarkan menjadi lebih banyak. Haus dan banyak minum. Banyaknya urine yang keluar menyebabkan cairan tubuh berkurang sehingga kebutuhan akan air minum meningkat.
2) Mudah merasa lelah. Hal ini disebabkan oleh berkurangnya glukosa pada jaringan dan sel di dalam tubuh. Kadar gula yang tinggi tidak bisa masuk dengan optimal karena menurunnya fungsi insulin. Itulah sebabnya penderita diabetes kekurangan energi.
3) Merasa pusing, berkeringat, dan sulit berkonsentrasi. Hal tersebut disebabkan oleh menurunnya kadar gula dalam darah. Setelah seorang penderita diabetes mengonsumsi gula, akan terjadi peningkatan reaksi pada pankreas yang dapat menimbulkan hipoglikemik..
4) Berat badan meningkat yang disebabkan oleh metabolisme karbohidrat yang terganggu karena hormon lain juga terganggu.
5) Gatal-gatal yang terjadi karena lapisan kulit yang mengering akibat dari gangguan regulasi cairan tubuh..

6) Gangguan imunitas. Meningkatnya kadar glukosa dalam darah menyebabkan penderita Diabetes rentan terhadap infeksi. Hal tersebut disebabkan oleh menurunnya fungsi sel-sel darah putih.
7) Adanya gangguan pada penglihatan yang disebabkan oleh perubahan cairan dalam lensa mata. Pandangan akan tampak kabur karena terjadi kelumpuhan pada otot mata.
8) Terjadinya gejala polyneuropathy atau gangguan sensorik pada saraf peripheral di kaki dan tangan.

2.4 Kecerdasan Buatan

Menurut Oxford Dictionary kecerdasan buatan atau Artficial Intelligence (AI) merupakan teori dan perkembangan sistem komputer untuk dapat melakukan pekerjaan yang biasanya membutuhkan kecerdasan manusia seperti, persepsi visual, pengenalan suara, pembuat keputusan, dan penerjemahan antar bahasa [16]. Tujuan dari AI ini adalah untuk membuat sistem yang ahli di mana sistem menunjukkan perilaku cerdas, belajar, menunjukkan, menjelaskan, dan memberi saran kepada penggunanya. Selain itu AI dibuat untuk mengimplementasikan kecerdasan manusia ke dalam mesin di mana menciptakan sistem yang memahami, berpikir, belajar, dan berperilaku seperti manusia [17].

2.5 Machine Learning

Salah satu program komputer yang dapat belajar sendiri dari masukan yang tersedia dan merupakan cabang dari *Artificial Intelligence* (AI) disebut dengan *Machine Learning* atau pembelajaran mesin. Jika pembelajaran adalah proses mengubah pengalaman menjadi keahlian, maka masukan ke dalam algoritma dikatakan sebagai pengalaman sedangkan keluaran adalah keahliannya [18]. *Machine Learning* hadir untuk membantu menemukan solusi terhadap berbagai permasalahan dalam visualisasi, pengenalan suara, dan robotika [19].

Machine Learning telah bercabang menjadi beberapa sub bidang yang berhubungan dengan berbagai jenis pembelajaran. Dalam machine learning pembelajaran dibagi menjadi 3, yaitu supervised learning, unsupervised learning, dan reinforcement learning [18].

1.5.1 Supervised Learning

Supervised Learning merupakan pembelajaran yang tujuannya untuk mempelajari pemetaan dari masukan ke keluaran yang nilai-nilainya sudah disediakan [19]. *Supervised Learning* memerlukan data training untuk melakukan pelatihan dan data testing untuk melakukan pengujian. Pembelajaran jenis ini dibagi menjadi dua, yaitu:

a. Regresi

   Regresi merupakan pembelajaran yang melakukan pemetaan dari variabel masukan ke variabel keluaran yang berupa nilai nyata atau kontinu [19]. Contoh dari pembelajaran ini adalah memprediksi umur seseorang yang dilihat dari foto atau gambar wajah.

b. Klasifikasi

   Klasifikasi merupakan pembelajaran yang melakukan pemetaan dari variabel masukan ke variabel keluaran [19]. Hampir sama dengan regresi, namun keluaran pada klasifikasi ini disebut dengan label atau kategori. Contohnya adalah memprediksi seorang pasien penyakit ginjal, apakah pasien tersebut menderita penyakit ginjal kronis atau tidak.

1.5.2 Unsupervised Learning

Unsupervised Learning merupakan pembelajaran yang tujuannya untuk menemukan keteraturan dalam masukan [19]. Dengan adanya struktur ke ruang input, terbentuklah pola-pola tertentu. Metode yang digunakan pada pembelajaran ini adalah *clustering*, tujuannya untuk menemukan pengelompokan masukan.

2.6 Pembersihan Data

Pembersihan data merupakan proses yang dilakukan sebelum masuk pada proses pengaplikasian dari machine learning. Pembersihan data berisi beberapa proses yang tujuannya untuk melakukan pengenalan dan perbaikan pada data yang akan diteliti [20]. Perlunya pembersihan pada data yang akan diteliti disebabkan karena data yang masih mentah cenderung tidak siap untuk digunakan. Kasus yang sering terjadi adalah adanya *missing values* pada data.

Missing value pada data berasal dari data-data yang atributnya tidak memiliki nilai informasi atau kosong. Informasi ini tidak diperoleh dimungkinkan

karena proses yang terjadi saat penggabungan data. Untuk mengatasi data yang salah dapat dilakukan membuang semua fitur yang tersedia dari data atau mengganti nilai dari fitur yang kosong dengan rata-rata fitur tersebut [21].

2.7 Normalisasi Data

Salah satu tahap penting dalam pembangunan *machine learning* yaitu pemrosesan awal atau *preprocessing*, ada beberapa metode yang dilakukan untuk pemrosesan awal salah satu contohnya adalah normalisasi data. Dalam praktiknya, dalam pembangunan *machine learning* sering dihadapkan dengan fitur nilai yang terletak dalam nilai jangkauan yang berbeda. Akibatnya, fitur dengan nilai jangkauan yang besar mempunyai pengaruh besar dalam fungsi daripada dengan fitur dengan nilai jangkauan yang kecil [21]. Untuk menangani masalah tersebut, bisa digunakan teknik normalisasi fitur, sehingga fitur semua fitur akan memiliki jangkauan yang sama. Untuk menskalakan nilai dari fitur dapat digunakan teknik *min-max normalization* dengan menggunakan persamaan (2.1).

$$f(x_i) = \frac{x_i - min_A}{max_A - min_A}(max_B - min_B) + min_B \quad \textbf{(2.1)}$$

2.8 K-Nearest Neighbor

Prinsip kerja K-Nearest Neighbor (KNN) adalah mencari jarak terdekat antara data yang akan dievaluasi dengan K tetangga terdekatnya dalam data pelatihan [22]. Teknik ini termasuk dalam kelompok klasifikasi *non parametric*. Di sini tidak memperhatikan distribusi dari data yang ingin kita kelompokkan. Teknik ini sangat sederhana dan mudah diimplementasikan. Mirip dengan teknik *clustering*, yaitu dengan cara mengelompokkan suatu data baru berdasarkan jarak data baru itu ke beberapa tetangga terdekat.

Tujuan algoritma KNN adalah mengklasifikasikan obyek baru berdasarkan atribut dan *training sample*. Proses klasifikasi menggunakan voting terbanyak diantara klasifikasi dari k. Algoritma KNN menggunakan klasifikasi ketetanggaan sebagai nilai prediksi. Algoritma metode KNN sangatlah sederhana, bekerja berdasarkan jarak terpendek dari *data testing* ke *data training* untuk menentukan KNN-nya. Ada pun persamaan yang digunakan untuk menghitung jarak antar data

digunakan *Euclidean Distance*. Berikut persamaan (2.2) yang digunakan untuk menghitung *euclidean distance*.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \textbf{(2.2)}$$

Nilai k yang terbaik untuk algoritma ini tergantung pada data. Secara umum, nilai k yang tinggi akan mengurangi efek *noise* pada klasifikasi, tetapi membuat batasan antara setiap klasifikasi menjadi semakin kabur. Kasus khusus di mana klasifikasi diprediksikan berdasarkan *data training* yang paling dekat disebut algoritma Nearest Neighbor .

Kelebihan KNN (K-Nearest Neighbor) [23]:

1. Tangguh terhadap training data yang memiliki banyak noise.
2. Efektif apabila training datanya besar.

Kelemahan KNN (K-Nearest Neighbor) [23]:

1. KNN perlu menentukan nilai dari parameter k (jumlah dari tetangga terdekat).
2. Training berdasarkan jarak tidak jelas mengenai jenis jarak apa yang harus digunakan.
3. Atribut mana yang harus digunakan untuk mendapatkan hasil terbaik.
4. Biaya komputasi cukup tinggi karena diperlukan perhitungan jarak dari tiap data uji ke data latih pada keseluruhan proses *training*.

Ada pun algoritma KNN adalah sebagai berikut

1. Tentukan parameter K
2. Hitung jarak antara data yang akan dievaluasi dengan semua pelatihan menggunakan persamaan
3. Urutkan jarak yang terbentuk (urut naik)
4. Tentukan jarak terdekat sampai urutan K
5. Pasangkan kelas yang bersesuaian
6. Cari jumlah kelas dari tetangga yang terdekat dan tetapkan kelas tersebut sebagai kelas data yang akan dievaluasi.

2.9 K-Support Vector Nearest Neighbor

K-Support Vector Nearest Neighbor (KSVNN) merupakan metode reduksi data latih yang didasarkan pada metode Template Reduction K-Nearest Neighbor (TRKNN) dan Support Vector Nearest Neighbor (SV-KNN), dengan prinsip K tetangga terdekat pada setiap data latih [21]. Tidak ada proses clustering yang dilakukan pada sisa data latih yang dihasilkan dan juga belum ada pembobotan pada data latih yang didapatkan sebagai *support vector*, sehingga proses pelatihan menjadi cepat [3]. Untuk mencari support vector pada K-SVNN berbeda dengan SV-KNN, untuk K-SVNN mempunyai persamaan sendiri untuk mencari *support vector*. Dengan berkurangnya data latih diharapkan proses prediksi menjadi lebih cepat dan akurat [3].

KSVNN berusaha mengurangi data latih berdasarkan properti skor dan properti relevansi/derajat signifikansi pada setiap data latih berdasarkan prinsip K tetangga terdekat [21]. Setiap data latih mempunyai kedua properti tersebut. Properti skor ada dua, yaitu *Left Value* (LV) dan *Right Value* (RV). Nilai LV untuk kelas yang sama, sedangkan nilai RV untuk kelas yang berbeda.

Ada pun berikut ini adalah algoritma KSVNN [24].

1. Melakukan inisialisasi *D* merupakan set data latih, K sebagai jumlah tetangga terdekat, T merupakan *threshold*, *Significant Degree* (SD), *Left Value* (LV) dan *Right Value* (RV) untuk semua data latih = 0.
2. Untuk setiap $d_i \in D$ data latih, lakukan langkah 3 hingga 5.
3. Menghitung jarak dari d_i ke data latih lain. Untuk mendapatkan jarak digunakan *Euclidean Distance* atau persamaan (2.2).
4. Pilih *dt* sebagai K tetangga terdekat.
5. Untuk setiap data dalam d_t, jika memiliki kelas yang sama, berikan nilai 1 pada *LV*, jika tidak berikan nilai 1 pada *RV*.
6. Untuk setiap data d_i, hitung *SD* menggunakan persamaan (2.3) di bawah ini.

$$SD_i = \begin{cases} 0, LV_i = RV_i = 0 \\ \frac{LV_i}{RV_i}, LV_i < RLV_i \\ \frac{RV_i}{LV_i}, LV_i > RV_i \\ 1, LV_i = RV_i \end{cases} \quad \textbf{(2.3)}$$

7. Pilih data dengan $SD \geq T$, simpan dalam variabel.

2.10 Matriks Konfusi

Sebuah sistem yang melakukan klasifikasi diharapkan dapat melakukan klasifikasi semua set data dengan benar, tetapi tidak dapat dipungkiri bahwa kinerja suatu sistem tidak dapat 100% benar sehingga sebuah sistem klasifikasi juga harus diukur kinerjanya. Umumnya, pengukuran kinerja klasifikasi dilakukan dengan matriks konfusi [21].

Matriks Konfusi (*Confusion Matrix*) adalah tabel yang memuat hasil klasifikasi. Jika terdapat elemen matriks konfusi untuk data klasifikasi dua buah kelas dinyatakan dengan f_{ij}, maka kelas i menyatakan jumlah data yang sebenarnya dan kelas j menyatakan hasil prediksi data tersebut. Tabel 2.1 merupakan contoh dari matriks konfusi.

Tabel 2.1 Contoh Matriks Konfusi

f_{ij}		Hasil Prediksi	
		i	j
Kelas Asli	i		
	j		

Ketepatan klasifikasi dapat dinilai dari hasil akurasi klasifikasi. Akurasi klasifikasi dapat menunjukan performa metode klasifikasi. Untuk menghitung akurasi dapat digunakan menggunakan persamaan (2.4).

$$Akurasi = \frac{Jumlah\ data\ diprediksi\ dengan\ benar}{Jumlah\ prediksi\ yang\ dilakukan} \times 100\% \quad \textbf{(2.4)}$$

2.11 Unified Modelling Language (UML)

Salah satu model perancangan dalam membangun aplikasi berorientasi objek yang paling sering digunakan adalah *Unified Modelling Language* (UML). UML sendiri adalah bahasa pemodelan untuk sistem atau perangkat lunak yang memiliki paradigma berorientasi objek.

UML adalah bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu aplikasi. UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek [25]. Ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri perangkat lunak. UML menyediakan beberapa macam diagram untuk memodelkan aplikasi berorientasi objek. Adapun beberapa diagram yang sering digunakan adalah *Use Case*, *Activity Diagram*, *Sequence Diagram*, dan *Class Diagram*.

2.11.1 Use Case Diagram

Use case diagram digunakan untuk memodelkan proses bisnis berdasarkan perspektif pengguna sistem. *Use case diagram* terdiri atas diagram untuk *use case* dan *actor*. *Actor* merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem [25].

Use case merepresentasikan operasi-operasi yang dilakukan oleh *actor*. *Use case* digambarkan berbentuk elips dengan nama operasi dituliskan di dalamnya. *Actor* yang melakukan operasi dihubungkan dengan garis lurus ke *use case*.

2.11.2 Activity Diagram

Sebuah activity diagram atau diagram aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana actor menggunakan sistem untuk melakukan aktivitas. Sama seperti state, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas [25].

Activity diagram pada dasarnya menggambarkan skenario secara grafis. Serta *activity diagram* cukup serupa dengan *flow chart*, yang membedakan mungkin hanya *swimlane* yang menunjukkan suatu *state* berada pada objek/kelas tertentu. Keunggulan dari *activity diagram* adalah bahwa diagram tersebut lebih

mudah dipahami dibanding skenario. Selain itu, dengan menggunakan *activity diagram*, dapat melihat dibagian manakah sistem dari suatu skenario akan berjalan.

2.11.3 Class Diagram

Class merepresentasikan suatu *Class* individu atau pun beberapa Class yang terkait satu sama lain. *Class diagram* menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas [25]. *Class diagram* membantu dalam memvisualisasikan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. Selama tahap desain, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat. Suatu *Class* memiliki tiga area pokok, yaitu nama (dan *stereotype*), atribut, dan metoda.

2.11.4 Sequence Diagram

Sequence diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. *Sequence diagram* merupakan salah satu diagram interaksi (interaction diagram) yang menjelaskan bagaimana sistem bekerja. Diagram ini diatur berdasarkan waktu. Objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut [25].

2.12 Python

Python adalah bahasa pemrograman komputer, sama seperti bahasa pemrograman lain, misalnya C, C++, C#, Pascal, Java, Kotlin, PHP, Perl dan lain-lain. Sebagai bahasa pemrograman. Python dapat dijalankan hampir pada semua platform, seperti GNU/Linux, Windows, dan Machintos. Python memiliki kosakata atau kata kunci, dan aturan tersendiri yang jelas berbeda dengan bahasa pemrograman lainnya [26]. Salah satu yang membedakan Python dengan bahasa pemrograman lain adalah Python dapat secara langsung mendeklarasikan suatu variabel tanpa menyebutkan tipe datanya.

Python diciptakan oleh Guido van Rossum pertama kali di Centrum Wiskunde & Informatica (CWI) di Belanda pada awal tahun 1990-an. Bahasa python terinspirasi dari bahasa pemrograman ABC. Python pertama kali

dikembangkan pada awal tahun 1990 oleh Guido van Rossum di Stiching Mathematisch Centrum. Pada tahun 1995, pengembangan bahasa pemrograman Python dilakukan di Corporation for National Research Initiatives. Saat ini Python Software Foundation bertugas sebagai pengembang Python [25].