Telematika: Jurnal Informatika dan Teknologi Informasi
Vol. 21, No. 2, Juni 2024, pp.145-157
ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

Prediction Of Type 2 Diabetes Mellitus Using The K-Nearest Neighbor Algorithm

Prediksi Penyakit Diabetes Melitus Type 2 menggunakan Algoritma *K-Nearest Neighbor*

Uning Lestari[1], Amir Hamzah[2], Franco Albertino Karel Paays[3]

[1,2,3] Informatika, Universitas AKPRIND Indonesia, Indonesia

[1]*uning@akprind.ac.id, [2]amir@akprind.ac.id, [3]albertinofranco06@gmail.com
**: Penulis korenspondensi* (*corresponding author*)

Informasi Artikel
Received: December 2023
Revised: March 2024
Accepted: April 2024
Published: June 2024

Abstract

Purpose: High blood sugar causes diabetes mellitus (DM), a metabolic disorder. DM affects human metabolism and causes many complications, such as heart disease, kidney problems, skin disorders, and slow healing. Therefore, using machine learning algorithms to implement an automatic diabetes diagnosis system is crucial for predicting DM.
Design/methodology/approach: This research created a DM disease prediction system using machine learning with the K-Nearest Neighbor algorithm. The National Institute of Diabetes and Digestive and Kidney Diseases, Hospital Frankfurt, Germany, and the results of health surveys and medical research are the sources of two separate datasets used in the Kaggle platform data. The stages in Machine Learning include data merging, data cleaning, and data splitting
Findings/result: This research produces the best prediction model at a ratio of 70:30 on test data with the highest accuracy values, namely 78.20%, precision 77.80%, recall 78.20%, and F1 78.0%. Test results with K Folding Cross validation produced an average accuracy of 73.88%. This shows that the prediction model is quite good although its accuracy still needs to be improved.
Originality/value/state of the art: This research creates a prediction model for diabetes mellitus type 2 using two different datasets with 9 features. It makes a Machine Learning model using the KNN algorithm by importing the KneighborClassifier and evaluating it using accuracy, precision, recall and F1 score and K Folding cross-validation to determine modelling accuracy.

Keywords:prediction; Diabetes Mellitus; K-Nearest Neighbor
Kata kunci: Prediksi, Diabetes Melitus; K-Nearest Neighbor

Abstrak

Tujuan: Diabetes melitus (DM) merupakan gangguan metabolisme yang disebabkan tingginya gula darah. DM mempengaruhi metabolisme manusia, dan penyakit ini menyebabkan banyak komplikasi seperti penyakit jantung, masalah ginjal, kelainan kulit dan juga penyembuhan yang lambat. Oleh karena itu penting untuk memprediksi penyakit DM menggunakan sistem diagnosis diabetes otomatis, yang dapat diimplementasikan menggunakan algoritma *machine learning.*

Perancangan/metode/pendekatan: Pada penelitian ini telah dibuat sistem prediksi penyakit DM menggunakam *machine learning* dengan algoritma The K-Nearest Neighbor (KNN). Data yang digunakan berasal dari platform Kaggle dengan dua dataset berbeda yang bersumber dari hasil survey kesehatan dan penelitian medis *National Institute of Diabetes and Digestive and Kidney Diseases dan Hospital Frankfurt, Germany*. Tahapan dalam Machine Learning dilakukan tahapan penggabungan data, pembersihan data, splitting dataset, pemodelan dengan algoritma KNN dan eavluasi pemodelan hasil prediksi.

Hasil: penelitian ini menghasilkan model prediksi terbaik pada rasio 70:30 pada data testing dengan nilai akurasi tertinggi yaitu 78.20%, precision 77,80%, recall 78.20%, dan F1 78.0%. Hasil uji dengan K Folding Cross validation menghasilkan rata-rata akurasi 73,88%. Hal ini menunjukkan model prediksi sudah cukup baik, walaupun masih perlu ditingkatkan lagi akurasinya

Keaslian/ *state of the art*: Penelitian ini membuat model prediksi penyakit diabetes melitus tipe 2 dengan menggunakan dua dataset yang berbeda dengan 9 fitur. Pembuatan model *Machine Learning* menggunakan algoritma KNN dengan mengimpor *KneighborClassifier* dan evaluasi model mengunakan *accuracy, precision, recall* dan *F1 score. K Folding Cross Validation* digunakan untuk menentukan uji akurasi pemodelan.

1. Pendahuluan

Diabetes melitus (DM) adalah gangguan metabolisme yang ditandai dengan peningkatan kadar gula darah (*hiperglikemia)* akibat kelainan sekresi insulin, kerja insulin, atau keduanya. Menurut definisi organisasi kesehatan dunia (WHO), diabetes melitus adalah penyakit *degeneratif* kronis yang disebabkan oleh produksi insulin yang tidak mencukupi di pankreas, atau ketidakmampuan tubuh untuk menggunakan insulin yang dihasilkannya secara efektif, sehingga menyebabkan *hiperglikemia* (kadar gula darah tinggi) sebagai indikator utama [1]. Penyakit diabetes ini merupakan penyakit berbahaya yang mengalami peningkatan signifikan yang dipengaruhi

berbagai faktor seperti tekanan darah tinggi, kadar gula berlebih, berat badan, riwayat keturunan diabetes, usia, jumlah kehamilan seseorang, ketebalan lipatan kulit, jumlah kadar insulin dalam tubuh, kurangnya aktivitas fisik dan pola hidup, serta diet tidak sehat [2]. Terdapat dua jenis utama dalam penyakit diabetes melitus yaitu diabetes tipe 1 atau *Juvenile Diabetes* yang disebabkan oleh gangguan sistem kekebalan tubuh dan diabetes Tipe 2 yang disebabkan oleh tingkat insulin yang dihasilkan pankreas tidak cukup. Diabetes tipe 2 paling umum terjadi ketika insulin diproduksi oleh pankreas, produksi yang tidak mencukupi atau sebaliknya mengakibatkan tubuh tidak dapat menggunakan insulin tersebut.
Data International Diabetes Federation (IDF) tahun 2021 mencatat 537 juta orang dewasa atau 1 dari 10 orang hidup dengan diabetes di seluruh dunia, dan menyebabkan 6,7 juta kematian atau 1 kematian setiap 5 detik. IDF pada 2021 juga menyebutkan bahwa Indonesia berada di posisi ke-5 dengan jumlah pengidap diabetes sebanyak 19,47 juta, prevalensi diabetes sebesar 10,6 persen. Pada tahun 2010 jumlah penderita diabetes melitus tipe 2 di Indonesia sebanyak 8,4 juta jiwa dan pada tahun 2030 diperkirakan akan meningkat menjadi 21,3 juta jiwa. Tingginya penderita diabetes melitus tipe 2 di Indonesia menunjukan bahwa masyarakat Indonesia masih kurang waspada dengan bahayanya penyakit diabetes melitus yang dapat mengakibatkan kematian. Gaya hidup masyarakat yang kurang sehat atau tidak seimbang menjadi salah satu faktor penyebab seseorang mengidap penyakit diabetes tipe 2, meskipun tidak mungkin untuk menyembuhkan penyakit ini sepenuhnya, adalah mungkin untuk hidup sehat jika dicegah atau diobati dengan benar. Mendiagnosis diabetes pada tahap awal sangat penting untuk menghindari risiko berkembangnya diabetes yang lebih parah [3].

Prediksi atau *forecasting* adalah cara untuk memprediksi apa yang paling mungkin terjadi di masa depan yang dilakukan secara sistematis berdasarkan informasi masa lalu dan saat ini, dan bertujuan untuk mengurangi perbedaan antara peristiwa yang terjadi dan hasil prediksi [4]. Prediksi ini sangat cocok diterapkan pada bidang kesehatan, khususnya pada prediksi deteksi penyakit diabetes melitus. Seorang ahli diabetologist harus mampu untuk secara kritis menganalisis beberapa faktor yang mempengaruhi untuk mendiagnosis diabetes. Ketidakmampuan untuk memahami data dalam jumlah yang besar dapat menyebabkan diagnosis yang salah, Untuk mengatasi masalah tersebut maka diperlukan bantuan teknologi yang dapat membantu memprediksi atau mendiagnosis diabetes agar lebih akurat [5]. Teknologi *machine learning* adalah aplikasi *Artificial Intelligence* (AI) dalam ilmu komputer yang berfokus pada penggunaan data dan algoritma untuk meniru cara manusia belajar. *Machine learning* dapat digunakan dalam bidang kesehatan seperti seperti analisis citra medis, pengembangan obat, diagnosis dan prognosis, *screening* penyakit, dan prediksi epidemi. *Machine learning* dapat membantu untuk mendeteksi deteksi dini penyakit, evaluasi hasil pengobatan, dan pengembangan tindakan kesehatan [6].

Penelitian tentang prediksi penyakit diabetes melitus ini telah dilakukan oleh peneliti-peneliti sebelumnya. Beberapa Algoritma juga telah diterapkan untuk melakukan prediksi atau *forecasting* yang dapat diaplikasikan ke dalam *machine learning* untuk membantu dalam proses diagnosis diabetes melitus adalah algoritma Decision Tree, Random Forest, K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Regresi Logistik, C45, dan Bayesian Classification. Penelitian Dwivedi *et al* [7] menggunakan klasifikasi untuk memprediksi penyakit DM menggunakan enam algoritma yaitu Classification tree, Support Vector Machine, Logistic Regression, Naïve Bayes, dan Artificial Neural Network (ANN). Akurasi klasifikasi sebesar 77

dan 78% dicapai masing-masing dengan ANN dan Regresi Logistik, dengan ukuran F 1 sebesar 0,83 dan 0,84. Dewi Rahma Ente *et al* [8] membangun sistem identifikasi dan melihat hubungan antara faktor yang mempengaruhi penyakit diabetes melitus menggunakan algoritma C4.5. Hasil menunjukkan ada empat faktor yang mempengaruhi prediksi status pasien DM: gula darah setelah makan, kolesterol LDL, trigliserida, dan berat badan. Penelitian Nurlaelatul Maulidah *et al* [9] menggunakan metode SVM dan Naive Bayes untuk mengetahui akurasi hasil diagnosa diabetes. Nilai akurasi model SVM dan Naive Bayes masing-masing 78,04% dan 76,98%. Laela Ismail et al [10] telah mengembangkan Intelligent Diabetes Mellitus Prediction Framework (IDMPF) menggunakan *machine learning* dan evaluasi model prediksi menggunakan Decision Tree (DT)-based Random Forest (RF) dan SVM. Usha, V.[11] telah melakukan penelitian tentang akurasi prediksi penyakit diabetes melitus untuk menilai tingkat keberhasilan prediksi beberapa metode pembelajaran seperti Convolutional Neural Networks dan Support Vector Machines. Dengan menggunakan *RandomForestClass* menghasilkan tingkat akurasi 91%. Penelitian prediksi penyakit diabetes melitus tipe-2 telah dilakukan juga oleh B. Ahamed *et al* [12] dengan menggunakan metode Logistic Regression, XGBoost, Gradient Boosting, Decision Trees, Extratrees, Random Forest, and Light Gradient Boosting Machine (LGBM). Hasil prediksi menunjukkan bahwa klasifikasi LGBM memiliki akurasi tertinggi dibandingkan dengan algoritma lainnya.yaitu 95,20%. Mohideen *et al* [13] telah berhasil mengusulkan peningkatan tingkat akurasi prediksi penyakit diabetes melitus secara otomatis menggunakan algoritma Otimizes Gaussian Naive Bayes (OGNB), Hasil klasifikasi Gaussian Naïve Bayes yang dioptimalkan mencapai akurasi pengklasifikasi 81,85%.

Permasalahan prediksi penyakit diabetes melitus tipe 2 merupakan permasalahan klasifikasi, sehingga pada penelitian ini penulis menggunakan algoritma K-Nearest Neighbor (KNN) untuk pemodelannya. Algoritma KNN ini dipilih karena mudah diimplemantasikan dan cocok untuk dataset sederhana. Proses evaluasi model mengunakan matrik *accuracy, precision, recall* dan *F1 score. K Folding Cross Validation* digunakan untuk menentukan uji akurasi pemodelan.

2. Metode Penelitian

2.1. Dataset Penelitian

Data yang digunakan pada penelitian ini berasal dari platform *Kaggle Data* dengan dua dataset sumber yang berasal dari hasil survey kesehatan dan penelitian medis National Institute of Diabetes and Digestive and Kidney Diseases dan Hospital Frankfurt, Germany. Dataset tersebut adalah dataset (*https://www.kaggle.com/datasets/akshaydattatraykhare/diabetes-dataset)* berjumlah 768 data dan dataset (*https://www.kaggle.com/datasets/johndasilva/diabetes*) sebanyak 2000 data. Terdapat sembilan atribut yang digunakan adalah *Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI (Body Mass Index), Diabetes Pedigree Function, Age, dan Outcome.* Keterangan fitur-fitur tersebut dapat dilihat pada Tabel 1.

Tabel 1. Fitur-fitur dataset pasien diabetes melitus

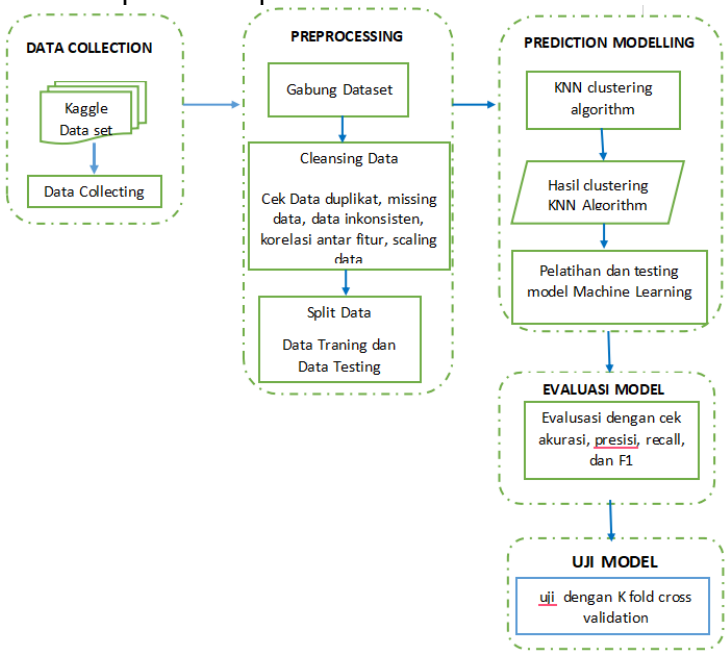
No	Fitur	Keterangan
1	Pregnancies	data jumlah kehamilan pasien
2	Glucose	data kadar gula darah pasien
3	Blood Pressure	data tekanan darah pasien
4	Skin Thickness	data ketebalan kulit pasien
5	Insulin	data jumlah kadar insulin

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

6	BMI (Body Mass Index)	data indeks massa tubuh pasien
7	Diabetes Pedigree Function	Resiko Diabetes turunan dari keluarga
8	Age	data umur pasien
9	Outcome	data hasil prediksi penyakit diabetes (1= DM, 0= tidak DM)

2.2. Tahapan Penelitian

Dalam pelaksanaan penelitian ini, terdiri beberapa tahapan yaitu diawali dengan tahap pengumpulan data (*data collection*), *preprocessing*, pembuatan model prediksi dan terakhir tahap evaluasi model. Proses *data collection* dilakukan dengan mengambil dataset pasien diabates melitus dari *Kaggle Dataset*. Pada tahap ini diambil dua dataset sumber pasien diabetes melitus. Kemudian dilanjutkan tahap *preprocessing* yaitu membersihkan, mengubah, dan mempersiapkan dataset agar sesuai untuk analisis lebih lanjut. Data yang dilakukan *preprocessing* adalah data yang duplikat, *missing* data, data *inkonsisten*, *data outlier*, dan data yang tidak seimbang. Tahap selanjutnya adalah *splitting* dengan membagi dataset menjadi data *training* dan data *testing*. Kemudian dilanjutkan proses pembuatan model *machine learning* menggunakan algoritma KNN dengan mengimpor *KNeighborClassifier* dari *library* Python yaitu *scikit-learn*. Tahap selanjutnya adalah pelatihan model *machine learning* dengan mempelajari data untuk menemukan pola dataset yang dimasukan dan membuat aturan-aturan dari dataset. Setelah pelatihan model sudah dilakukan maka model siap digunakan untuk melakukan *forecasting* atau prediksi penyakit diabetes. Selanjutnya model *machine learning* akan dievaluasi dengan menggunakan accuracy, precision, recall and F1 score. Jika nilai evaluasi model tidak sesuai maka akan dilakukan proses *preprocessing* untuk memperbaiki data atau mengubah parameter nilai K. Setelah dilakukan evaluasi model maka selanjutnya adalah tahap menentukan hasil evaluasi terbaik dari *forecasting* yang sudah dilakukan sebelumnya dan menganalisa performa model untuk menentukan apakah model perlu perbaikan kembali atau tidak. Tahapan penelitian dapat dilihat pada Gambar 1



Gambar 1. Tahapan Penelitian

2.3. Model Prediksi dengan K-Nearest Neighbor (KNN)

K-Nearest Neighbor (KNN) adalah metode untuk klasifikasi terhadap objek berdasarkan data pembelajaran (*neighbor*) yang jaraknya paling dekat dengan objek tersebut. Dekat atau jauhnya *neighbor* biasanya dihitung berdasarkan pengukuran jarak. Algoritma KNN bekerja dengan mencari K tetangga terdekat berdasarkan jarak ruang objek dari data baru. Dekat atau jauhnya jarak dengan tetangganya dapat dihitung berdasarkan rumus jarak Euclidean. Nilai K merupakan parameter yang digunakan untuk menentukan jumlah tetangga terdekat yang digunakan dalam proses klasifikasi. Nilai K adalah bilangan bulat positif. Besarnya K menentukan berapa banyak tetangga terdekat yang diambil dalam proses prediksi. Sebelum menjalankan algoritma KNN, nilai K harus ditentukan terlebih dahulu. Pemilihan nilai K yang tepat sangat penting karena mempengaruhi efisiensi dan akurasi model. Jika nilai K terlalu kecil, model menjadi sangat sensitif terhadap noise atau outlier dalam data, sehingga prediksi menjadi tidak stabil. Sebaliknya, jika nilai K terlalu besar, model kurang peka terhadap variasi lokal dalam data dan memberikan prediksi yang terlalu umum. Nilai-K dapat dipilih menggunakan beberapa metode, seperti validasi silang atau mencoba beberapa nilai-K dan memilih yang terbaik berdasarkan evaluasi kinerja model [14].

K-Nearest Neighbor adalah algoritma yang sederhana dan mudah diimplementasikan tetapi KNN memiliki beberapa kelemahan, seperti sensitivitas terhadap skala fitur, sensitivitas terhadap data yang redundant atau tidak seimbang, dan performa yang dapat melambat saat volume data meningkat. Pemilihan nilai K yang tepat juga merupakan faktor penting dalam keberhasilan algoritma KNN [15].

Tahapan algoritma KNN adalah sebagai berikut:

- Tentukan parameter K (jumlah tetangga paling dekat).
- Menghitung kuadrat jarak euclidian (*euclidean distance*) masing-masing obyek terhadap data sampel yang diberikan dengan persamaan :

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

Dimana:

d(x,y) = jarak.
i = indek atribut
n = jumlah data
x_i = atribut dari data ke-i (i=1,2,3...,n)
y_i = atribut dari pusat cluster ke-i (i=1,2,3,....,n)

- Mengurutkan objek-objek tersebut ke dalam kelompok yang mempunyai jarak euclidian terkecil
- Proses evaluasi kinerja: setelah model KNN dilatih, kinerjanya dievaluasi menggunakan confusion matrix, yang dari matrik tersebut dapat dihitung akurasi, presisi, recall dan skor F1 dengan formula sebagai berikut :

Table 2. Confusion Matrix dari hasil klasifikasi 2 kelas

	Nilai aktual

Nilai Prediksi	Positive	Negative
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

- Selanjutnya dihitung parameter kinerja klasifikasi sebagai berikut :

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$Akurasi = \frac{TP+TN}{TP+FP+TN+FN} \quad (4)$$

- Untuk membandingkan kinerja dua algoritma diperlukan kombinasi ukuran yang memadukan Precision dan Recall dalam satu ukuran, yaitu skor F1 dengan formula :

$$F1 = \frac{2*Precision*Recall}{Precision+Recall} \quad (5)$$

- Dalam perhitungan nilai positive diwakili nilai numerik 1 (menderita diabetes) dan nilai negative numerik 0 (tidak menderita diabetes)

3. Hasil dan Pembahasan

3.1. Preprocessing Data

Pengolahan data pada penelitian ini menggunakan Google Collaboratory yang merupakan tools pengolah data dan melakukan permodelan menggunakan bahasa pemrograman *Python.*

Sebelum dilakukan tahap preprocessing, terlebih dahulu dilakukan *exploratory* analisis data untuk mengetahui karakteristik dari data akan digunakan. Pada tahap ini dilakukan pengecekan nilai statistik dataset seperti nilai rata-rata, jumlah data, standar deviasi, nilai minimal, nilai maksimal. Pada nilai statistik kedua dataset ternyata memiliki nilai 0 yang merata di setiap fitur yang nantinya akan mempengaruhi hasil peramalan. Nilai 0 pada dataset dapat diganti dengan nilai rata-rata dan median pada tahap preprocessing data.

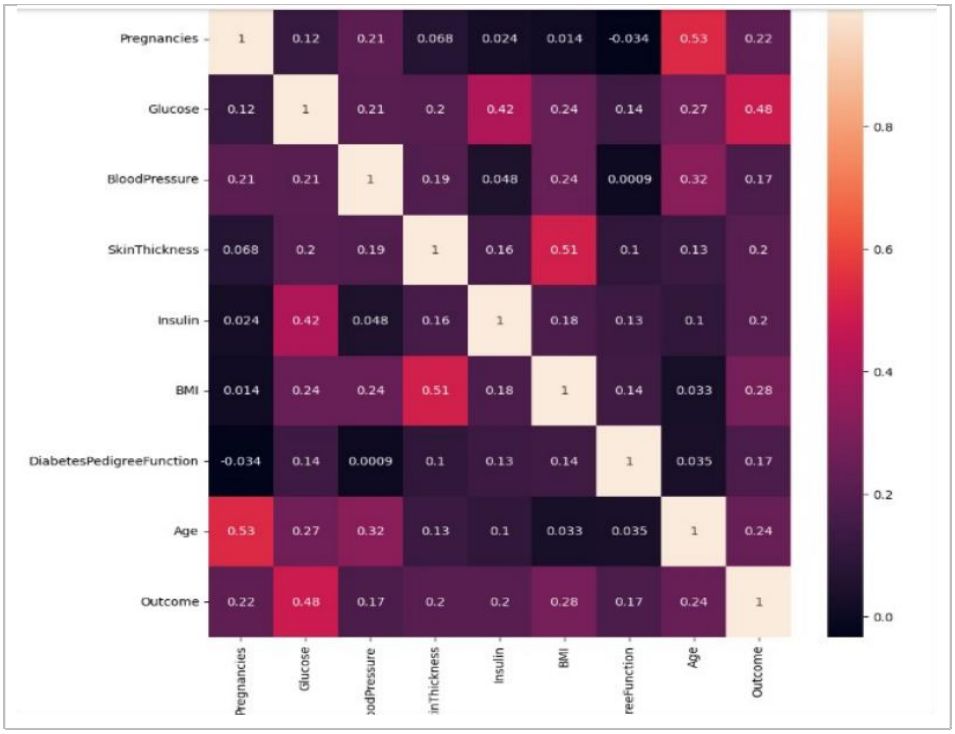
Selain nilai statistik akan mengecek data duplikat dan data kosong atau null yang ada dalam 2 dataset. Pada dataset *df_pima* tidak adanya nilai *null* dalam dataset serta nilai yang duplikat di dalam dataset. Pada dataset *df_db* tidak didapati data bernilai null tetapi memiliki data duplikat yang cukup banyak dengan jumlah 1256 data. data duplikat ini dapat diatasi dengan menggunakan perintah *drop_duplicated()* untuk menghilangkan data duplikat

Tahap *preprocessing* data dilakukan untuk membersihkan data, mengatasi missing value, dan memanipulasi data untuk mempersiapkan dataset sebelum membangun model *machine learning*. Tahap *preprocessing* data dimulai dengan menggabungkan kedua dataset menjadi satu agar lebih mudah memanipulasi data dan membersihkan data dengan menggunakan perintah *concat([df1,df2]).* Penggabungan kedua dataset menghasilkan jumlah data 1512 data dengan 9 kolom fitur. Tahap selanjutnya dalam *preprocessing* adalah mengecek jumlah nilai 0 yang ada dalam dataset untuk kolom atribut *pregnancies, glucose, blood pressure, skinthickness, insulin,* dan BMI. Pada atribut *pregnancies* terdapat 144 kasus (14,65%), atribut *glucose* (0.64%) , atribut *bloodpressure* memiliki 36 kasus (4.63%), atribut skinthickness memiliki 229 kasus (48.46%) dan kolom BMI memiliki 11 kasus (1.41%). Selanjutnya Nilai

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

0 pada kolom atribut yang telah dicek akan dimanipulasi dengan nilai *Nan* kecuali atribut *pregnancies.* Nilai *Nan* pada setiap atribut dapat diisi dengan nilai rata-rata atau nilai median dari masing-masing atribut. Pada atribut *glucose* dan *bloodpressure* nilai *Nan* akan diisi dengan nilai rata-rata dari masing-masing atribut sedangkan atribut *skinthickness, insulin,* dan *BMI* akan diisi dengan nilai median dari masing-masing atribut.

Tahap selanjutnya dalam proses preprocessing data adalah mengecek korelasi antar fitur dalam dataset dengan menggunakan *heatmap* dengan perintah *sns.heatmap()*. Hasil visualisasi korelasi fitur dengan *heatmap* dapat dilihat pada Gambar 2.



Gambar 2. Visualisasi korelasi fitur dengan *heatmap*

Pada Gambar 2 menunjukkan bahwa fitur *pragnancies dan age* memiliki korelasi yang cukup kuat sekitar 0.53. Fitur *Glucose* dan *Outcome* menunjukkan korelasi positif sekitar 0.48, berarti kadar *glukosa* lebih tinggi cenderung terkait dengan hasil yang lebih positif (kemungkinan besar terkait dengan diabetes). Fitur BMI dan *outcame* memiliki korelasi positif senilai 0.28, menunjukkan bahwa BMI yang lebih tinggi cenderung terkait dengan hasil positif. Fitur *insulin* dan *SkinThickness* menunjukkan korelasi positif sekitar 0.51, menunjukkan hubungan yang signifikan antara kadar *insulin* dan ketebalan kulit.

Variabel dengan korelasi yang lebih tinggi (baik positif maupun negatif) akan menjadi indikator yang lebih kuat dalam analisis lebih lanjut atau model prediksi. Visualisasi *heatmap* korelasi dapat mengidentifikasi hubungan penting antara variabel dalam dataset dan membuat keputusan tentang fitur mana yang paling relevan untuk analisis atau pembuatan model lebih lanjut.

Tahap berikutnya dalam proses preprocessing data adalah *scaling* data yang berfungsi untuk mengubah rentang data supaya mempunyai rentang sama serupa atau terstandardisasi. Pada proses *scaling* data menggunakan teknik *standardscaler()* dimana mengubah skala data

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

pada setiap atribut fitur yang dimiliki dataset sehingga memiliki rata-rata 0 dan standar deviasi 1.

Tahap terakhir dalam proses *preprocessing* data penelitian ini adalah membagi dataset (*splitting*) menjadi 2 yaitu x dan y dimana Y digunakan menyimpan atribut fitur *Outcome* dan X untuk sisa atribut fitur kecuali Outcome

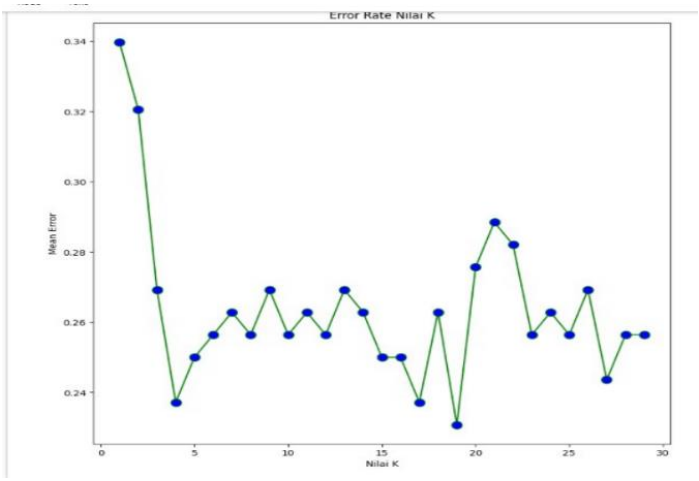
3.2. Splitting Data

Tahap Splitting Data dimulai dengan membagi dataset menjadi 2 bagian yaitu data training dan data testing. Pada penelitian ini dilakukan 3 kali splitting data dengan rasio yang berbeda yaitu 80:20, 70:30, dan 60:40 dimana 80%, 70%, dan 60% untuk data training dan 20%, 30% ,40 % data untuk data testing. Splitting data menggunakan *train_test_split* dari *scikit-learn*

3.3. Pemodelan K-Nearest Neighbor (KNN)

Pembuatan model *machine learning* dilakukan pada tahap ini. Model *machine learning* menggunakan 2 parameter nilai K yaitu nilai K yang diperoleh secara random dan parameter nilai K terbaik. Proses pembuatan model dimulai dengan menentukan nilai K selanjutnya menginisialisasi *KneighborClassifier* dan memasukan nilai K sebagai parameter didalamnya dan dilakukan proses *training* data untuk melatih model mempelajari dataset yang digunakan. Penggunaan *library* random untuk mendapatkan nilai K secara acak kemudian setelah menginisialisasi KNeighborClassifier maka model akan melakukan *training* data dengan menggunakan perintah *fit()* dan memasukan parameter berapa data *training* dan data testing dengan rasio pertama yaitu 80:20 yang telah dilakukan pada proses *splitting* data.

Tahap berikutnya adalah proses peramalan data testing dan mencari nilai akurasi model. Uji coba pertama menggunakan nilai *k* terbaik rasio 80:20. Pencarian nilai *k* terbaik dilakukan menggunakan perulangan nilai *k* dari 1sampai 30 dan mencari nilai error rate terkecil. Pada Gambar 3 menampilkan visualisasi nilai error rate pada *splitting* data 80.30. Pada visualisasi tersebut menunjukkan Nilai K terbaik dengan nilai error rate terendah pada nilai 19, nilai ini yang akan dihitung untuk mencari nilai akurasi yang optimal.



Gambar 3. Visualisasi Nilai K Terbaik dengan nilai *Error rate* terendah model 80:20

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

Setelah nilai *k* terbaik ditemukan maka proses inisialisasi *KNeighborClassifier* dapat dilakukan kembali dan melakukan peramalan ulang. Pada Gambar 4 menampilkan hasil running program dengan menggunakan nilai k sebesar 19 didapatkan akurasi model sebesar 76.92% pada rasio 80:20 %. Tahap berikutnya adalah pembuatan model dengan rasio berbeda yaitu 70:30 dan 60:40. Tahapan yang sama seperti yang dilakukan sebelumnya pada rasio 80:20. Hasil pemodelan dengan splitting data rasio 80:20, 70:20 dan 60:40 dapat dilihat pada Tabel 2.

```
[64] k=19

[65] #Inisialisasi KNN
     knn = KNeighborsClassifier(k)
     #Pembuatan Model
     model = knn.fit(x1_train,y1_train)

[66] y_pred = model.predict(x1_test)
     score = []
     score = model.score(x1_test, y1_test)

[67] print("Jumlah Nilai K : ",k,)
     print("Akurasi Model  : ",score*100 ," %")

     Jumlah Nilai K :  19
     Akurasi Model  :  76.92307692307693  %
```

Gambar 4. Hasil running program akurasi nilai k terbaik

Tabel 2. Hasil pemodelan nilai k dan akurasi data testing

Splitting	**K random**		**K terbaik (nilai k= 1-30)**	
	Nilai K	Akurasi	Nilak K	Akurasi
80:20	20	72.43%	19	76.92%
70:30	13	74,78%	23	78.20%
60:40	3	71.4%	21	76.28%

3.4. Evaluasi Model K-Nearest Neighbor (KNN)

Sebelum dilakukan evaluasi model terlebih dahulu dilakukan validasi model. Pada tahap validasi ini menggunakan teknik validasi *K folding cross validation*. Validasi ini dilakukan agar mengurangi bias karena akan terjadi bias jika hanya menggunakan satu set data saja. Pembagian dataset dibagi menjadi K subset atau "folds" yang kira-kira sama besar. Misalnya, jika K=5, maka dataset dibagi menjadi 5 subset. Pada penelitian ini dilakukan ujicoba dengan nilai k=5, dan hasil validasi tersebut menghasilkan rata-rata akurasi sebesar 73,88%. Hasil uji validasi tersebut dapat dilihat pada Tabel 4.

Tabel 4. Hasil uji validasi dengan *K folding cross validation*

k	**Akurasi**
k1	73,7
k2	66,6
k3	75,6
k4	80
k5	73,5
rata-rata akurasi	**73,88**

Selanjutnya evaluasi kinerja algoritma dilakukan dengan membagi data menjadi data training dan data testing menggunakan tiga jenis splitting data seperri disajikan pada Tabel 3.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

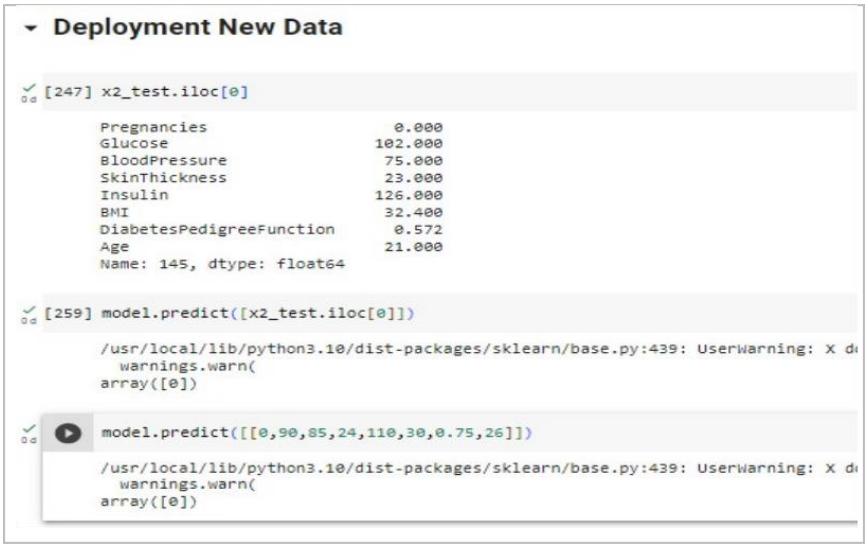
Selanjutnya pada setiap kombinasi dihitung nilai *Precision, Recall, Akurasi* dan skor *F1* menggunakan rumus pada persamaan (2)-(5). Dari Tabel 3 dapat dilihat nilai performa model terbaik data testing pada rasio 70:30, yaitu pada kombinasi tersbeut memiliki nilai Akurasi dan F1 yang tertinggi dibandingkan dengan kombinasi yang lain.

Tabel 3. Hasil Nilai Akurasi dan F1 untuk data training dan data testing

Kombinasi Train:Test	**Data Training**				**Data Testing**			
	Precision	**Recall**	**Akurasi**	**F1**	**Precision**	**Recall**	**Akurasi**	**F1**
80:20	71,20%	73,40%	72,60%	72,3%	75,80%	76,70%	76,92%	76,2%
70:30	73,10%	74,50%	**74,20%**	**73,8%**	77,80%	78,20%	**78,20%**	**78,0%**
60:40	69,70%	72,10%	71,80%	70,9%	76,20%	76,12%	76,28%	76,2%

Langkah terakhir adalah uji coba dengan memasukan data baru untuk melakukan peramalan dengan model yang sudah dibuat. Ujicoba ini bertujuan untuk melihat apakah model yang sudah dibuat mempunyai akurasi yang baik dalam memprediksi penyakit jika dimasukkan data baru yang tidak termasuk data training. Kode program untuk ujicoba ini adalah sebagai berikut:

```
model.predict([[0,90,85,24,110,30,0.75,26]])
```



Gambar.5. Hasil Peramalan Model 70:30 dengan data baru

Pada Gambar 5 dilakukan percobaan dengan menggunakan data input (x) pada indeks ke 0 dalam dataset yang sudah dilakukan pada splitting data 70:30 dan setelah dilakukan peramalan dihasilkan label outcome 0 atau non diabetes. Hasil prediksi ini sudah sesuai dengan label target.

4. Kesimpulan dan Saran

Hasil Model machine learning yang dihasilkan dalam penelitian ini dengan 3 parameter rasio pembagian data training dan data testing 80:20, 70:30, dan 60:40 memiliki hasil akurasi 76.92%,78,20%, 76,2%. Akurasi yang tertinggi pada splitting data 70:30 yaitu 78.20%. Akurasi yang dihasilkan dalam penelitian ini masih bisa ditingkatkan lagi untuk hasil peramalan yang dilakukan oleh model. Untuk meningkatkan tingkat kepercayaan model maka

harus ditingkatkan akurasi modelnya. Hal-hal yang dapat dilakukan untuk meningkatkan akurasi nya antara lain treatment data seperti menghilangkan outlier yang masih ada, mengurangi noise dan mengatasi imbalance data. Selain treatment data dapat dilakukan seleksi fitur dengan memilah kembali atribut yang dapat dihilangkan yaitu atribut-atribut yang kurang mempengaruhi nilai peramalan model. Setelah melakukan treatment data yang lebih baik dapat dilakukan validasi silang yang ketat untuk mengukur performa model yang lebih akurat untuk mencegah overfitting dan underfitting pada data pelatihan. Optimasi model dengan *hyper tuning parameter* juga dapat dilakukan salah satunya dengan *GridsearchCV* untuk menemukan parameter terbaik untuk metode K-Nearest Neighbor seperti nilai k, *weights, metric,* dan *leaf size*. Pemanfaatan algoritma lain juga dapat dilakukan untuk memperoleh hasil akurasi yang lebih baik seperti *LightGBM* dan algoritma lainnya.

Daftar Pustaka

[1] R. G. Ginting, E. Girsang, J. B. Ginting, en H. Hartono, "Analisis Determinan Dan Prediksi Penyakit Diabetes Melitus Tipe 2 Menggunakan Metode Machine Learning: Scoping Review", *J. Matern. Kebidanan*, 2022.

[2] Q. R. Cahyani, M. J. Finandi, J. Rianti, D. L. Arianti, en A. D. P. Putra, "Prediksi Risiko Penyakit Diabetes menggunakan Algoritma Regresi Logistik", *JOMLAI J. Mach. Learn. Artif. Intell.*, 2022.

[3] F. Fitriyani, "Prediksi Diabetes Menggunakan Algoritma Naive Bayes dan Greedy Forward Selection", *J. Nas. Teknol. dan Sist. Inf.*, 2021.

[4] R. J. Hyndman en G. Athanasopoulos, "Forecasting: Principles and Practice, 2nd edition", *OTexts: Melbourne, Australia.*, 2019. .

[5] H. Wu, S. Yang, Z. Huang, J. He, en X. Wang, "Type 2 diabetes mellitus prediction model based on data mining", *Informatics Med. Unlocked*, 2018.

[6] A. Sumathi en S. Meganathan, "Machine learning based pattern detection technique for diabetes mellitus prediction", *Concurr. Comput. Pract. Exp.*, 2022.

[7] A. K. Dwivedi, "Analysis of computational intelligence techniques for diabetes mellitus prediction", *Neural Comput. Appl.*, 2018.

[8] D. R. Ente, S. A. Thamrin, S. Arifin, H. Kuswanto, en A. Andreza, "Klasifikasi Faktor-Faktor Penyebab Penyakit Diabetes Melitus Di Rumah Sakit Unhas Menggunakan Algoritma C4.5", *Indones. J. Stat. Its Appl.*, 2020.

[9] N. Maulidah, R. Supriyadi, D. Y. Utami, F. N. Hasan, A. Fauzi, en A. Christian, "Prediksi Penyakit Diabetes Melitus Menggunakan Metode Support Vector Machine dan Naive Bayes", *Indones. J. Softw. Eng.*, 2021.

[10] L. Ismail en H. Materwala, "IDMPF: intelligent diabetes mellitus prediction framework using machine learning", *Appl. Comput. Informatics*, 2021.

[11] V. Usha en N. R. Rajalakshmi, "Insights into Diabetes Prediction: A Multi-Algorithm Machine Learning Analysis", in *Proceedings of the 4th International Conference on Smart Electronics and Communication, ICOSEC 2023*, 2023.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v21i2. 12384

[12] B. S. Ahamed, M. S. Arya, en A. O. Nancy V, "Prediction of Type-2 Diabetes Mellitus Disease Using Machine Learning Classifiers and Techniques", *Frontiers in Computer Science*. 2022.

[13] D. F. M. Mohideen, J. S. S. Raj, en R. S. P. Raj, "Regression Imputation and Optimized Gaussian Naive Bayes Algorithm for an Enhanced Diabetes Mellitus Prediction Model", *Brazilian Arch. Biol. Technol.*, 2021.

[14] S. Uddin, I. Haque, H. Lu, M. A. Moni, en E. Gide, "Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction", *Sci. Rep.*, 2022.

[15] S. Deng, L. Wang, S. Guan, M. Li, en L. Wang, "Non-parametric Nearest Neighbor Classification Based on Global Variance Difference", *Int. J. Comput. Intell. Syst.*, 2023.