Klasifikasi Penyakit Diabetes Melitus Menggunakan *Adaboost Classifier*

**Ginanjar Abdurrahman1**

*1* Jurusan Teknik Informatika, Universitas Muhammadiyah Jember

*Email:* *1abdurrahmanginanjar@unmuhjember.ac.id*

(Naskah masuk: dd mmm yyyy, diterima untuk diterbitkan: dd mmm yyyy)

ABSTRAK

Diabetes Melitus (DM) merupakan penyakit dengan ciri tingginya gula darah sewaktu dengan nilai ≥200 mg/dL, dan nilai gula darah puasa ≥126 mg/dL. Klasifikasi merupakan algoritma *supervised learning* untuk menemukan pola atau fungsi untuk mendeskripsikan dan memisahkan suatu kelas data dengan kelas data yang lainnya. *Adaboost (Adaptive Boosting)* merupakan salah satu algoritma *machine learning* yang dapat membangun pengklasifikasi kuat dengan mengombinasikan beberapa pengklasifikasi lemah*.* Algoritma ini juga dapat beradaptasi dengan data dan metode pengklasifikasi lainnya. Selain itu, algoritma ini juga dapat mengurangi tingkat error dari *pengklasifikasi lemah* sehingga dapat menaikkan nilai akurasi dari algoritma pembelajaran yang ada (*boosting).* Pada penelitian ini akan dilakukan klasifikasi penyakit DM menggunakan algoritma *AdaBoost Classifier* untuk menentukan apakah seseorang diklasifikasikan sebagai pasien diabetes atau sehat. Dataset diambil dari repositori *UCI Machine Learning*, terdiri dari 8 variabel input, 1 variabel target, serta 768 record. Hasil klasifikasi algoritma *Adaboost Clasifier* pada dataset hasil *imputing* *mean* menghasilkan akurasi sebesar 80.09 %, untuk dataset hasil *imputing* *median* menghasilkan akurasi sebesar 76.19 %, Sedangkan dataset hasil *imputing* modus, output yang dihasilkan sama dengan dataset yang mengandung missing values yang mengakibatkan algoritma *Adaboost* tidak bisa berjalan. Hal ini dikarenakan algoritma Adaboost sangat sensitif terhadap keberadaan *missing values*. Python mengenali NaN sebagai nilai modus untuk beberapa fitur, dengan demikian, nilai NaN tetap ada

**Kata kunci**: gula darah, diabetes, klasifikasi, prevalensi, machine learning, adaboost

*ABSTRACT*

Diabetes mellitus (DM) is a disease characterized by high blood sugar at a time with a value of ≥200 mg / dL, and a value of fasting blood sugar ≥126 mg / dL. Classification is a supervised learning algorithm to find patterns or functions to describe and separate a data class from other data classes. Adaboost (Adaptive Boosting) is a machine learning algorithm that can build strong classifiers by combining several weak classifiers. This algorithm can also adapt to other data and classification methods. In addition, this algorithm can also reduce the error rate of weak classifiers so that it can increase the accuracy value of the existing learning algorithm (boosting). In this study, the classification of DM will be carried out using the AdaBoost Classifier algorithm to determine whether a person is classified as a diabetic or healthy patient. The dataset is taken from the UCI Machine Learning repository, consisting of 8 input variables, 1 target variable, and 768 records. The classification results of the Adaboost Clasifier algorithm on the mean imputing dataset yields an accuracy of 80.09%, for the median imputing dataset it produces an accuracy of 76.19%, while the imputed mode dataset results in the same output as the dataset containing missing values ​​which results in the Adaboost algorithm unable to run . This is because the Adaboost algorithm is very sensitive to the presence of missing values. Python recognizes NaN as the mode value for some features, thus, the NaN value remains.

**Keywords**: *glucose, diabetes, classification, machine learning, adaboost*

# PENDAHULUAN

Diabetes melitus merupakan penyakit dengan ciri tingkat gula darah sewaktu ≥200 mg/dL, dan kadar gula darah puasa ≥126 mg/dL (Misnadiarly(Hestiana, 2017)). International Diabetes Federation (IDF) menjadikan Diabetes sebagai penyakit paling mamatikan urutan ke-tujuh di dunia dengan prevalensi 1.9%. Pada tahun 2013, penderita diabetes dunia mencapai 382 juta jiwa, dengan 95% diantaranya adalah DM tipe 2.

*Machine learning* merupakan suatu teknik untuk meniru cara mesin (komputer) untuk “belajar” dari data *(learn from data)* (Lukman & Marwana, 2014). Python merupakan Bahasa pemrograman untuk data *analyst*, data *scientist*, juga data *engineer* dalam implementasi machine learning (Purwadhika, 2019).

Klasifikasi merupakan algoritma *supervised learning* untuk menemukan pola atau fungsi untuk memisahkan suatu kelas data dengan kelas data yang lainnya. Biasanya algoritma klasifikasi digunakan untuk prediksi data yang belum memiliki kelas data tertentu (Bimo et al., 2020).

*Adaboost (Adaptive Boosting)* merupakan salah satu algoritma dalam *machine learning* yang diperkenalkan oleh Yoav Freund dan Robert Schapire. Algoritma ini membangun pengklasifikasi kuat *(strong classifier)* dengan mengombinasikan beberapa pengklasifikasi lemah *(weak classifier).* Algoritma ini juga dapat beradaptasi dengan data dan algoritma *classifier* lainnya, sehingga disebut *adaptive*. Selain itu, algoritma ini juga dapat mengurangi *error* dari penklasifikasi lemahsehingga dapat menaikkan akurasi dari setiap algoritma pembelajaran yang ada sehingga algoritma ini bernama *boosting* (Imaduddin & Tawakal, 2015)*.*

Pada penelitian ini akan mengklasifikasikan penyakit DM dengan algoritma *AdaBoost Classifier* untuk mengklasifikasikan seseorang sebagai pasien diabetes atau sehat. Dipilihnya algoritma Adaboost dalam penelitian ini dikarenakan kelebihan *AdaBoost* dalam membangun pengklasifikasi kuat dengan mengombinasikan pengklasifikasi lemah, Algoritma ini juga dapat beradaptasi dengan data dan metode pengklasifikasilainnya. Selain itu, algoritma ini juga dapat mengurangi tingkat *error* dari *pengklasifikasi lemah* sehingga dapat menaikkan akurasi dari algoritma pembelajaran yang ada. Dataset dalam penelitian adalah dataset diabetes dari repositori UCI. Dataset ini terdiri dari 8 variabel input, 1 variabel target, serta 768 record

# PENELITIAN TERKAIT

Penelitian yang dilakukan (Imaduddin & Tawakal, 2015) untuk mendeteksi dan mengklasifikasikan daun secara otomatis menggunakan algoritma Adaboost dan SVM.Pada penelitian ini dilakukan deteksi daun otomatis sekaligus mengenali objek daun. Metode AdaBoost digunakan untuk deteksi otomatis letak dan jenis daun. Performa deteksi AdaBoost dari penelitian ini memiliki akurasi 84,23 %, sedangkan klasifikasi dengan SVM memiliki akurasi 71 %.

Penelitian selanjutnya (Oktanisa & Supianto, 2018) bertujuan membandingkan kinerja 9 algoritma klasifikasi terhadap tanggapan nasabah.Pada penelitian ini bertujuan membandingkan 9 algoritma klasifikasi pada tanggapan nasabahdari dataset Bank Direct Marketing. Tujuan penelitian ini adalah untuk mengetahui keefektifan model klasifikasi dalam mengklasifikasikan target pada dataset *Bank Direct Marketing*. Algoritma yang digunakan adalah *SVM, Adaboost, Naïve Bayes, Constant, KNN, Tree, Random Forest, SGD, dan CN2 Rule. Preprocessing* yang dilakukan adalah menghapus *missing value* dan ekstraksi fitur dari dataset.Pada tahap evaluasi dilakukan Teknik 10 *fold cross validation*. Setelah diuji, diperoleh hasil klasifikasi terbaik adalah model Tree dengan CA 0.97, *precission* 0.95, dan *recall* 0.98.

Selain sebagai classifier, AdaBoost juga dapat digunakan untuk meningkatkan kinerja algoritma *classifier*. Seperti penelitian (Rohman et al., 2017) dengan tujuan memprediksi penyakit jantung dengan algoritma C4.5. Performa algoritma C4.5 ditingkatkan dengan Algoritma AdaBoost yang diimplementasikan pada data penderita penyakit jantung. Berdasarkan confusion matrix dan kurva ROC, metode C45 berbasis Adaboost diperoleh akurasi 86.6%, nilai AUC yang diperoleh 0.96 dan seetelah dipotimasi dengan algoritma *AdaBoost* nilai akurasinya menjadi 92,24% nilai AUC 0.982

# METODE PENELITIAN

Secara umum, langkah-langkah penelitian ditampilkan pada Gambar 1 berikut:

Studi Literatur

Dataset

*Klasifikasi :*

*AdaBoost*

Identifikasi Masalah

Akurasi

Selesai

*Preprocessing:*

*Handling* *Missing Values*

*Impute:*

*Mean*

*Impute :*

*Median*

*Impute :*

*Modus*

# HASIL DAN PEMBAHASAN

**4.1 *Preprocessing Data***

*Preprocessing* yang dilakukan hanya penanganan *missing values* saja. Hal ini dikarenakan, gangguan data yang ada hanyalah *missing values.* Dari dataset yang digunakan, untuk fitur kelas keputusan sudah dalam bentuk boolean nol dan satu, bukan dalam karakter string berupa kata, sehingga tidak perlu dilakukan *binarization*. Dari dataset yang digunakan, terdapat *missing values* dari bebarapa fitur, yakni pada fitur: *Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, dan BMI.* Langkah-langkah dalam *preprocessing* selengkapnya dijelaskan sebagai berikut:

1. Mengidentifikasi banyaknya *missing values*

Untuk mengidentifikasi banyaknya *missing values* pada setiap fitur dengan sintaks *isna().sum()* yang ada pada library *python 3.* Pada *dataset* *default*, nilai *missing values* dituliskan dengan angka 0 (nol) pada setiap fitur kelas input, sehingga pada saat dilakukan identifikasi *missing values* menggunakan sintaks *isna().sum()* angka nol tersebut tidak teridentifikasi sebagai nilai *missing values*, karena dianggap mempunyai nilai yakni nol. Oleh karena itu, nilai *missing value* tersebut perlu diubah dulu menjadi kosong (NaN) dengan sintaks *replace* nol menjadi NaN yang merupakan salah satu *library* dari *Numpy*. Tampilan awal identifikasi *missing values* saat masih bernilai nol dapat dilihat pada Tabel 1. Sedangkan tampilan identifikasi nilai *missing values* setelah angka nol sebagai representasi *missing values* diubah menjadi *NaN* disajikan pada Tabel 2.

Tabel 1 Identifikasi *missing values* pada *dataset default*

|  |  |
| --- | --- |
| Fitur | Banyaknya *missing values* |
| *Kehamilan**Glukosa**Tekanan Darah**Ketebalan Kulit**Insulin**BMI**Fungsi Pedigree Diabetes**Usia* | 00000000 |

Tabel 2 Identifikasi *missing values* pada dataset setelah nilai 0 sebagai representasi *missing values* diubah menjadi *NaN*.

|  |  |
| --- | --- |
| Fitur | Banyaknya *missing values* |
| *Kehamilan**Glukosa**Tekanan Darah**Ketebalan Kulit**Insulin**BMI**Fungsi Pedigree Diabetes**Usia* | 1115352273741100 |

Adapun tampilan Dataset setelah nilai nol sebagai representasi missing values diubah menjadi NaN disajikan pada Gambar 5.

Gambar 5 Dataset kelas input setelah nilai nol sebagai representasi missing values diubah menjadi NaN

***4.2 Imputing* *missing values***

Teknik *impute* yang digunakan adalah impute menggunakan *mean*, *median* dan *modus* dari setiap fitur. Tampilan dataset setelah nilai missing values diimpute menggunakan nilai mean, median dan modus dapat dilihat pada Gambar 6, 7 dan 8.



Gambar 6 Imputing Missing Values dengan Mean



Gambar 7 Imputing Missing Values dengan Median



Gambar 8 Imputing Missing Values dengan Modus

**4.3. Klasifikasi *Adaboost***

**4.3.1 Uji coba mengunakan dataset yang mengandung *NaN* sebagai representasi missing values**

Algoritma *Adaboost* sangat sensitif terhadap keberadaan *missing values*, hal ini dapat dilihat pada saat menjalankan algoritma dengan keberadaan *missing values (NaN)*, terdapat keterangan *ValueError: Input contains NaN.* Adapun tampilan eksekusi algoritma pada python 3 ketika masih terdapat missing values pada Gambar 9.



Gambar 9 Tampilan output python 3 ketika masih terdapat missing values

Oleh karena itu perlu dilakukan teknik *imputing* *missing values*. Dalam penelitian ini, akan digunakan 3 teknik *imputing* yakni *imputing* menggunakan nilai *mean*, nilai *median*, dan nilai *modus* dari setiap fitur, kemudian setiap dataset hasil *imputing* *mean, median, dan modus* diklasifikasikan menggunakan algoritma *Adaboost Classifier* dan dilihat nilai akurasinya

**4.3.2 Uji coba mengunakan dataset hasil imputing *mean***

. Hasil klasifikasi algoritma *Adaboost Clasifier* pada dataset hasil *imputing* *mean* menghasilkan akurasi sebesar 80.09 %.

**4.3.3 Uji coba mengunakan dataset hasil imputing *median***

 Dataset hasil *imputing* *median* menghasilkan akurasi sebesar 76.19 %

**4.3.4 Uji coba mengunakan dataset hasil imputing *modus***

Dataset hasil *imputing* modus, output yang dihasilkan sama dengan dataset yang mengandung NaN sebagai representasi *missing values*. Hal ini dikarenakan, aplikasi mengenali NaN sebagai nilai modus untuk beberapa fitur, dengan demikian, nilai NaN tetap ada.

# KESIMPULAN

1. Hasil klasifikasi algoritma *Adaboost Clasifier* pada dataset hasil *imputing* *mean* menghasilkan akurasi sebesar 80.09 %
2. Dataset hasil *imputing* *median* menghasilkan akurasi sebesar 76.19 %
3. Dataset hasil *imputing* modus, output yang dihasilkan sama dengan dataset yang mengandung *missing values* yang mengakibatkan algoritma *Adaboost* tidak bisa berjalan. Hal ini dikarenakan algoritma Adaboost sangat sensitif terhadap keberadaan *missing values*. Python mengenali NaN sebagai nilai modus untuk beberapa fitur, dengan demikian, nilai NaN tetap ada

# DAFTAR PUSTAKA

Bimo, P., Setio, N., Retno, D., Saputro, S., & Winarno, B. (2020). Klasifikasi dengan Pohon Keputusan Berbasis Algoritme C4.5. *PRISMA, Prosiding Seminar Nasional Matematika*, *3*, 64–71.

Hestiana, D. W. (2017). Journal of Health Education. *Journal of Health Education*, *25*(1), 57–60. https://doi.org/10.1080/10556699.1994.10603001

Imaduddin, Z., & Tawakal, H. A. (2015). *Deteksi dan klasifikasi daun menggunakan metode adaboost dan svm*. 6–8.

Lukman, A., & Marwana. (2014). Machine Learning Multi Klasifikasi Citra Digital. *Konferensi Nasional Ilmu Komputer (KONIK)*, *December 2014*, 1–6.

Oktanisa, I., & Supianto, A. A. (2018). Perbandingan Teknik Klasifikasi Dalam Data Mining Untuk Bank a Comparison of Classification Techniques in Data Mining for. *Teknologi Informasi Dan Ilmu Komputer*, *5*(5), 567–576. https://doi.org/10.25126/jtiik20185958

Purwadhika, S. S. (2019). *Apa Itu Python dan Fungsinya di Dunia Nyata?* https://medium.com/purwadhikaconnect/apa-itu-python-dan-fungsinya-di-dunia-nyata-d5b533117c63

Rohman, A., Suhartono, V., & Supriyanto, C. (2017). Penerapan Agoritma C4.5 Berbasis Adaboost Untuk Prediksi Penyakit Jantung. *Jurnal Teknologi Informasi*, *13*, 13–19.