KLASIFIKASI MENGGUNAKAN METODE SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK DETEKSI AWAL RISIKO DIABETES MELITUS

by Puspita Kartikasari

Submission date: 10-May-2023 11:48PM (UTC+0700)
Submission ID: 2089611653
File name: 37093-83803-2-PB.pdf (564.08K)
Word count: 4213
Character count: 25389

ISSN: 2339-2541
JURNAL GAUSSIAN, Volume 11, Nomor 3, Tahun 2022 Halaman 386-396
Online di: https://ejournal3.undip.ac.id/index.php/gaussian/

KLASIFIKASI MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE* DAN *RANDOM FOREST* UNTUK DETEKSI AWAL RISIKO DIABETES MELITUS

Chea Zahrah Vaganza Junus[1*], Tarno[2], Puspita Kartikasari [3]
[1,2,3] Departemen Statistika, Fakultas Sains dan Matematika, Universitas Diponegoro
***email:** cheazahrah@gmail.com

DOI: 10.14710/j.gauss.11.3.386-396

Article Info:
Received: 2022-06-30
Accepted: 2022-09-03
Available Online: : 2023-01-03

Keywords:
Diabetes Melitus; Machine Learning; Classification; Support Vector Machine; Random Forest

Abstract: Diabetes Mellitus is one of the four leading causes of death and therefore possible treatments are of crucial importance to the world leaders. Prevention and control of Diabetes Mellitus are often done by implementing a healthy lifestyle. Thus, both people with risk factors and people diagnosed with Diabetes Mellitus can control their disease in order to prevent complications or premature death.. For a proper education and standardized disease management the early detection of Diabetes Mellitus is necessary, which led to this conducted study about the classification of early detection of Diabetes Mellitus risk by utilizing the use of Machine Learning. The classification algorithms used are the Support Vector Machine and Random Forest where the performance analysis of the two methods will be seen in classifying Diabetes Mellitus data. The type of data used in this study is secondary data obtained from the official website of the UCI Machine Learning Repository consisting of 520 diabetes patient data taken from Sylhet Diabetic Hospital in Bangladesh with 16 independent variables and 1 dependent variable. The dependent variable categorizes the test result into positive and negative Diabetes Mellitus classes. The results of this study indicate that the Random Forest classification algorithm produces a better classification performance on Accuracy (98.08%), Recall (97.87%), Precision (98.92), and F1_Score (88.40%).

1 PENDAHULUAN

Penyakit Tidak Menular (PTM) telah menjadi masalah kesehatan masyarakat yang cukup meresahkan. Hal ini ditandai dengan bergesernya pola penyakit secara epidemiologi dari penyakit menular yang cenderung menurun ke penyakit tidak menular yang secara global meningkat di dunia dan telah menduduki salah satu dari empat penyakit penyebab kematian juga kasus terbanyak yang menjadi target tindak lanjut oleh para pemimpin dunia adalah penyakit Diabetes Melitus. Tidak hanya menyebabkan kematian prematur di seluruh dunia, Diabetes Melitus juga menjadi penyebab utama kebutaan, penyakit jantung dan gagal ginjal (Kemenkes RI, 2020).

Pencegahan dan pengendalian Diabetes Melitus kerap dilakukan dengan menerapkan pola hidup sehat, dalam hal ini agar orang yang memiliki faktor risiko dan penderita Diabetes Melitus dapat mengendalikan penyakitnya sehingga tidak terjadi komplikasi atau kematian dini. Upaya pencegahan dan pengendalian ini dilakukan melalui edukasi dan tatalaksana sesuai standar. Deteksi dini Diabetes Melitus sangat diperlukan, dengan demikian penelitian ini dilakukan untuk klasifikasi deteksi awal risiko Diabetes Melitus dengan memanfaatkan penggunaan *Machine Learning*.

Klasifikasi merupakan suatu pekerjaan yang melakukan penilaian terhadap suatu objek data untuk masuk dalam suatu kelas tertentu dari sejumlah kelas yang tersedia (Prasetyo, 2012). Terdapat berbagai macam metode klasifikasi, yaitu CART (*Classification And Regression Trees*), CHAID (*Chi-Square Automatic Interaction Detection*), *Random Forest*, SVM (*Support Vector Machine*) dan metode klasifikasi lainnya. Metode SVM

dikenal sebagai metode klasifikasi yang memiliki nilai akurasi yang tinggi. Menurut (Prasetyo, 2012) konsep klasifikasi dengan SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah kelas data pada *input space*. Metode klasifikasi nonparametrik yang juga sering digunakan adalah Classification and Regresssion Trees (CART). Metode *Random Forest* adalah pengembangan dari metode CART, yaitu dengan menerapkan metode *bootstrap aggregating* (*bagging*) dan *random feature selection* (Breiman, 2001). Teknik dasar dari metode *Random Forest* adalah *Decision Tree*. *Random Forest* memberikan akurasi yang bagus dalam klasifikasi, dapat menangani data training yang jumlahnya besar, dan juga efektif untuk mengatasi data yang tidak lengkap.

Mengingat betapa pentingnya deteksi dini Diabetes Melitus maka masih terdapat peluang bagi penulis untuk melakukan penelitian mengenai analisis performa yang dihasilkan dari metode *Support Vector Machine* dan *Random Forest* dalam klasifikasi deteksi awal risiko Diabetes Melitus.

2 TINJAUAN PUSTAKA

Menurut American Diabetes Association (ADA, 2010) Diabetes Melitus merupakan salah satu kelompok penyakit metabolik yang ditandai oleh hiperglikemia karena gangguan sekresi insulin, kerja insulin, atau keduanya. Keadaan hiperglikemia kronis dari diabetes berhubungan dengan kerusakan jangka panjang, gangguan fungsi dan kegagalan berbagai organ, terutama mata, ginjal, saraf, jantung, dan pembuluh darah. Diabetes Melitus adalah penyakit serius kronis yang terjadi baik ketika pankreas tidak menghasilkan cukup insulin (hormon yang mengatur gula darah, atau glukosa), atau ketika tubuh tidak dapat secara efektif menggunakan insulin yang dihasilkan (World Health Organization, 2016).

Konsep klasifikasi dengan SVM dapat dijelaskan secara sederhana sebagai usaha mencari hyperplane terbaik. Hyperplane merupakan pemisah terbaik antara kedua kelas data, yang dapat ditentukan dengan mengukur titik maksimal dari margin hyperplane. Margin merupakan jarak antara hyperplane dengan data terdekat pada masing-masing kelas. Data yang berada paling dekat dengan hyperplane terbaik disebut sebagai support vector (Prasetyo, 2012).

Diberikan himpunan $X = \{x_1, x_2, \cdots, x_n\}$ dengan $x_i \in R^n$, dimana i $= 1,2, \ldots, t$ adalah data *training*, untuk $y_i \in \{-1, +1\}$ menyatakan label kelas, sehingga data berupa pasangan $(x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t)$ merupakan himpunan data training dari dua kelas yang akan diklasifikasikan dengan *Support Vector Machine*.

Diasumsikan kedua kelas -1 dan +1 dapat terpisah secara sempurna oleh *hyperplane* berdimensi n, yang didefinisikan sebagai :

$$(\boldsymbol{w}.\boldsymbol{x}) + b = 0 \quad (1)$$

$\boldsymbol{w}$ dan b merupakan parameter model, dimana b merupakan simpangan atau kesalahan yang konsisten dalam memperkirakan sebuah nilai. Untuk mendapatkan fungsi pemisah terbaik adalah dengan mencari fungsi pemisah yang terletak ditengah-tengah antara dua bidang pembatas kelas, sama dengan memaksimalkan margin atau jarak antara dua set objek dari kelas yang berbeda (Santosa, 2007). Selanjutnya diformulasikan kedalam persamaan *quadratic programming* dengan meminimalkan invers persamaan :

$$min \frac{1}{2}\|\boldsymbol{w}\|^2 \text{ dimana } \|\boldsymbol{w}\|^2 = \boldsymbol{w}^T\boldsymbol{w} \quad (2)$$

dengan syarat :

$$y_i[(\boldsymbol{w}.\boldsymbol{x}) + b] \geq 1 \quad ; i = 1,2, \ldots, t \quad (3)$$

Optimalisasi ini dapat diselesaikan dengan *Lagrange Multiplier*.

$$L(\boldsymbol{w}, b, \alpha) = \frac{1}{2}\boldsymbol{w^T w} - \sum_{i=1}^{t} \alpha_i \{y_i[\boldsymbol{w^T}.\boldsymbol{x_i} + b] - 1\} \quad (4)$$

Nilai optimal dari persamaan (4) dapat dihitung dengan mencari turunan pertama dari fungsi lagrange terhadap variabel w dan b dan disamakan dengan nol (Santosa, 2007), sehingga diperoleh syarat optimal dari fungsi lagrange multiplier adalah seperti persamaan (5) dan (6) :

$$\frac{\delta L}{\delta b} = 0 \rightarrow -\sum_{i=1}^{t} \alpha_i y_i = 0 \quad (5)$$

$$\frac{\delta L}{\delta \boldsymbol{w}} = 0 \rightarrow \sum_{i=1}^{t} \alpha_i \boldsymbol{x_i} y_i = \boldsymbol{w} \quad (6)$$

dimana $\boldsymbol{w^T w}$ dapat dijabarkan seperti persamaan (7) :

$$\boldsymbol{w^T w} = \sum_{i=1}^{t} \alpha_i y_i [\boldsymbol{w^T}.\boldsymbol{x_i}] = \sum_{i=1}^{t}\sum_{j=1}^{t} y_i y_j \alpha_i \alpha_j (\boldsymbol{x_i^T}.\boldsymbol{x_j}) \quad (7)$$

maka persamaan (4) akan berubah menjadi dualitas Lagrange Multiplier seperti persamaan (8) :

$$L_d = \sum_{i=1}^{t} \alpha_i - \frac{1}{2}\sum_{i=1}^{t}\sum_{j=1}^{t} y_i y_j \alpha_i \alpha_j (\boldsymbol{x_i^T}.\boldsymbol{x_j}) \quad (8)$$

Kemudian fungsi dual pada persamaan (8) dimaksimalkan, sehingga diperoleh seperti persamaan (9)

$$max_\alpha L_d = max \sum_{i=1}^{t} \alpha_i - \frac{1}{2}\sum_{i=1}^{t}\sum_{j=1}^{t} y_i y_j \alpha_i \alpha_j (\boldsymbol{x_i^T}.\boldsymbol{x_j}) \quad (9)$$

dengan batasan $\alpha_i \geq 0$; $i = 1,2,\dots,t$ dan $\sum_{i=1}^{t} \alpha_i y_i = 0$

fungsi keputusan yang dihasilkan hanya dipengaruhi oleh *Support Vector*. *Hyperplane* (fungsi pemisah) diperoleh dengan rumus seperti persamaan (10) :

$$f(z) = sign\ (\boldsymbol{w}.\boldsymbol{z} + b) = \left(\sum_{i=1}^{p} \alpha_i y_i \boldsymbol{x_i}.\boldsymbol{z}\right) + b \quad (10)$$

dengan p merupakan jumlah data yang menjadi *Support Vector*, $\boldsymbol{x_i}$ merupakan *Support Vector*, $\boldsymbol{z}$ merupakan data testing yang akan diprediksi kelasnya, dan $\boldsymbol{x_i}.\boldsymbol{z}$ merupakan *dot-product* antara $\boldsymbol{x_i}$ dan $\boldsymbol{z}$ (Prasetyo, 2012).

Metode SVM juga dapat digunakan dalam kasus non-separable dengan memperluas formulasi yang terdapat pada kasus linier. Masalah optimasi sebelumnya baik pada fungsi obyektif maupun kendala dimodifikasi dengan mengikutsertakan variabel Slack $\xi > 0$. Variabel slack merupakan sebuah ukuran kesalahan klasifikasi. Formulasinya seperti persamaan (11) (Gunn, 1998).

$$y_i[(\boldsymbol{w}^T x_i) + b] \geq 1 - \xi_i\ , i = 1,2,3,\dots,t \quad (11)$$

sehingga persamaan (11) menjadi seperti persamaan (12)

$$(\boldsymbol{w}, \xi) = \frac{1}{2}(\boldsymbol{w^T w}) + c\sum_{i=1}^{t} \xi_i \quad (12)$$

Usaha untuk meminimalkan kesalahan klasifikasi yang dinyatakan dengan variabel *slack* (ξ_i) sementara dalam waktu yang bersamaan juga memaksimalkan margin (Santosa, 2007).

Dalam dunia nyata (*real world problem*) pada umumnya masalah data yang diperoleh jarang yang bersifat linear, banyak yang bersifat *non* linear (Nugroho, 2003). Pada SVM terdapat sebuah fungsi kernel, yaitu fungsi yang digunakan untuk menyelesaikan problem *non* linear. Kernel berfungsi memungkinkan untuk mengimplementasikan suatu model pada ruang dimensi lebih tinggi (ruang fitur). Fungsi pemisah (hyperplane) untuk klasifikasi non-linier menggunakan fungsi kernel adalah seperti persamaan (13)

$$f(z) = sign\left(\sum_{i=1}^{p} \alpha_i\, y_i K(\boldsymbol{x_i}, \boldsymbol{z}) + b\right) \quad (13)$$

Berikut merupakan fungsi kernel yang populer dan sering digunakan :

Tabel 1. Fungsi kernel yang sering digunakan

Jenis Kernel	Definisi Fungsi
Linear	$K(\boldsymbol{x}_i, \boldsymbol{z}) = (\boldsymbol{x}_i^T . \boldsymbol{z})$
Polynomial	$K(\boldsymbol{x}_i, \boldsymbol{z}) = (\boldsymbol{x}_i^T . \boldsymbol{z} + 1)^{\mathrm{d}}$
Radial Basis Function (RBF)	$K(\boldsymbol{x}_i, \boldsymbol{z}) = \exp(-\gamma \left\|\boldsymbol{x}_i - \boldsymbol{z}\right\|^2)$

Pada tahun 1995, Tin Kam Ho dari Bell Labs pertama kali memperkenalkan *Random Forest*. Menurutnya, *Random Forest* adalah algoritma pembelajaran *ensemble* untuk tahapan *data mining* dengan tugas klasifikasi dan regresi (Ho, 1995). *Random Forest* menggunakan teknik *bagging* untuk membangun *ensemble decision tree*. Pada penelitiannya, Breiman menyebutkan beberapa kelebihan dari *Random Forest* diantaranya dapat menghasilkan hasil akhir klasifikasi yang baik dan error yang lebih rendah, mampu mengatasi jumlah data yang cukup banyak secara lebih baik dan adalah salah satu algoritma yang efektif dalam permasalahan *missing data* (Breiman, 2001).

Algoritma atau prosedur dalam membangun *Random Forest* pada gugus data yang terdiri dari n amatan dan terdiri atas q variabel independen, berikut dibawah merupakan tahapan penyusunan dan pendugaan menggunakan *Random Forest* (Breiman, 2001; Breiman & Cutler, 2003) :

1. Lakukan *boostrap* yaitu proses penarikan sampel acak berukuran *n* dengan *replacement* (pengembalian).
2. Pilih *m* variabel secara *random* dari *q* variabel, dimana $m \leq q$. Biasanya ukuran *m* terbaik dipilih melalui aproksimasi dari akar kuadrat dari total jumlah *q* variabel, yaitu $\sqrt{q}$. Nilai *m* juga dapat diperoleh dari dua kali nilai akar kuadrat dari total jumlah *q* variabel ($m = 2\sqrt{q}$) dan setengah dari nilai akar kuadrat dari total jumlah *q* variabel ($m = \frac{1}{2}\sqrt{q}$).
3. Setelah dilakukan pemilihan *m* secara *random*, maka pohon ditumbuhkan tanpa *pruning* (pemangkasan). Lalu lakukan tahap *random feature selection* yaitu pada setiap *node* (simpul), selanjutnya dilakukan pemecahan simpul terbaik dalam suatu pohon dengan ukuran *Gini Index*. Penghitungan *Gini Index* dilakukan pada *m* variabel terpilih. Apabila node *D* dipisah menjadi dua partisi node D_1 dan D_2, maka *Gini Index* dari node yang dipisah dihitung menggunakan persamaan (14) dan (15):
$$Gini_{split}(D) = \frac{N_1}{N} Gini(D_1) + \frac{N_2}{N} Gini(D_2) \quad (14)$$
$$Gini(D) = 1 - \sum_{l=1}^{k} s_l^2 \quad (15)$$
dengan s_l adalah probabilitas dari kelas *l* pada node *D*, $l = 1,2,\ldots,k$ dan N adalah jumlah total sampel data pada node *D*.
4. Ulangi langkah 1 dan 2 sebanyak *n* kali, sehingga membentuk *forest* yang terdiri atas *n- tree* (terbentuk pohon sebanyak "n").

Salah satu algoritma yang dapat digunakan untuk pengambilan keputusan adalah algoritma grid search. Algoritma grid search adalah algoritma yang sering diaplikasikan untuk melakukan hyperparameter optimization karena pengaplikasiannya yang cenderung mudah (Bergstra & Bengio, 2012). Perbedaan antara grid search dan random search adalah

pada grid search untuk mencari parameter terbaik algoritma ini mencoba semua kombinasi yang ada sedangkan random search hanya mengambil beberapa kombinasi secara acak.

Akurasi klasifikasi merupakan ukuran ketepatan klasifikasi yang menunjukkan performansi teknik klasifikasi secara keseluruhan (Nugroho, 2003). Semakin tinggi akurasi klasifikasi berarti performansi teknik klasifikasi juga semakin baik. Umumnya cara mengukur kinerja klasifikasi menggunakan matriks konfusi. Matriks konfusi mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya.

Tabel 2. *Confusion Matrix*

	Predicted Positive Class	*Predicted Negative Class*
Actual Positive Class	TP (*True Positive*)	FN (*False Negative*)
Actual Negative Class	FP (*False Positive*)	TN (*True Negative*)

Ukuran dalam mengevaluasi kinerja model berdasarkan *confusion matrix* ada berbagai macam, diantaranya adalah *accuracy, precision,* dan *F1 score*. *Accuracy* dalam *confusion matrix* adalah presentase kebenaran dari hasil prediksi pada data *testing*. *Precision* merupakan ukuran proporsi prediksi TP. *Accuracy, specificity, precision,* dan *F1 score* dapat diperoleh sebagai persamaan (16),(17) dan (18) berikut:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (16)$$

$$Precision = \frac{TP}{FP+TP} \quad (17)$$

$$F1\_Score = \frac{2xPrecisionxRecall}{Precision+Recall} \quad (18)$$

Dataset dengan kelas tidak seimbang dan *error cost* yang sama maka bisa menggunakan ukuran kinerja berupa *error rate* yaitu *error rate* = 1- *accuracy*. Namun, dalam klasifikasi data kelas yang tidak seimbang nilai *error rate* bukanlah ukuran performa yang baik, maka ukuran performa hasil klasifikasi pada kasus seperti ini biasanya diukur dengan *recall* seperti persamaan (19) (Chawla *et al*., 2002).

$$Recall = \frac{TP}{TP+FN} \quad (19)$$

3 METODOLOGI PENELITIAN

Jenis data yang digunakan dalam penelitian ini adalah data sekunder yang diperoleh dari website resmi UCI *Machine Learning Repository*. Data yang digunakan terdiri dari 520 data pasien Diabetes yang diambil dari *Sylhet Diabetic Hospital* di Bangladesh. Variabel dalam penelitian ini terdiri atas 1 variabel respon (Y) dan 16 variabel penjelas (X).

Software atau alat bantu yang digunakan untuk pengolahan data ini adalah python dengan menggunakan *Google Colaboratory* (Colab) yang merupakan pengembangan *environment* python yang dijalankan dalam browser menggunakan *Google Cloud* dan *Microsoft Excel* 2019. Data yang telah diperoleh kemudian dianalisis sebagai berikut :

1. Input data
2. Melakukan *pre-processing*.
3. Melakukan klasifikasi risiko awal Diabetes Melitus dengan menggunakan metode *Support Vector Machine* (SVM) dengan tahapan sebagai berikut :
   a. Menentukan fungsi kernel yang digunakan beserta nilai-nilai parameter

b. Membagi data *training* dan data *testing* dengan *holdout validation*
c. Melakukan *tuning hyperparameter Support Vector Machine* (SVM)
d. Melakukan pengukuran performa klasifikasi

4. Melakukan klasifikasi risiko awal Diabetes Melitus dengan menggunakan metode *Random Forest* dengan tahapan sebagai berikut :
   a. Membagi data *training* dan data *testing* dengan *holdout validation*
   b. Melakukan *tuning hyperparameter Random Forest*
   c. Membentuk model pohon klasifikasi *Random Forest*
   d. Melakukan pengukuran performa klasifikasi
5. Melihat performa klasifikasi metode SVM dan *Random Forest*

4 Hasil dan Pembahasan

Dari 520 data dapat diketahui bahwa sebanyak 320 pasien atau sebesar 61,5% terdeteksi Diabetes Melitus. Sedangkan untuk yang tidak terdeteksi Diabetes Melitus adalah sebanyak 200 pasien atau sebesar 38,5%. Sebelum melakukan implementasi algortima, akan dilakukan tahap data *pre-processing* yang bertujuan untuk menyiapkan data guna meningkatkan kualitas dari data agar menghasilkan prediksi yang akurat. Dari data menunjukkan tidak ada missing value pada semua variabel independen dan variabel dependen sehingga tidak perlu dilakukan penghapusan baris maupun estimasi nilai

Dataset yang digunakan terdiri dari 1 variabel numerik dan 15 variabel kategorik. Karena hanya ada 1 variabel numerik yaitu umur sehingga tidak perlu dilakukan transformasi karena tidak ada pembanding untuk menyamakan satuan parameternya. Perlu dilakukannya transformasi apabila terdapat lebih dari satu variabel dengan satuan parameter yang berbeda.

Langkah-langkah pengklasifikasian menggunakan metode *Support Vector Machine* (SVM) telah dijelaskan pada bab sebelumnya, oleh karena itu akan dilakukan pembentukan model yang sesuai dengan langkah-langkah tersebut. Hal pertama yang dilakukan untuk membentuk suatu model adalah dengan menentukan parameter terbaiknya, berikut merupakan tahapan pengerjaan untuk pembentukan model :

1. Pada *Support Vector Machine* (SVM) menggunakan kernel *Radial Basis Function* (RBF) parameter yang ditentukan adalah nilai C (cost) dan γ (gamma).
2. Dalam penentuan parameter terbaik dari model, dapat dilakukan *tuning* secara simultan dengan *grid search* yaitu dengan mencobakan semua kombinasi parameter yang telah ditentukan. Pada penelitian ini, parameter yang akan dicobakan adalah sebagai berikut :

Tabel 3. Parameter *Support Vector Machine (SVM)*

Parameter	Nilai Parameter
C (Cost)	0.01,0.1,1,10,100
γ (Gamma)	0.0001,0.001,0.01,0.1, 1

Tuning parameter dilakukan untuk mencobakan semua kombinasi parameter yang telah ditetapkan sehingga terdapat 25 kombinasi. Proses *tuning* parameter ini menggunakan 5-*cross validation*, sehingga didapatkan 125 *running model*. Untuk menentukan parameter terbaik didasarkan pada nilai akurasi tertingginya. Proses *tuning* parameter ini menghasilkan model terbaik pada parameter dengan nilai C = 100 dan Gamma = 0.1.

3. Model yang dibangun menggunakan *Support Vector Machine* (SVM) memiliki nilai C = 100 dan Gamma = 0.1. Data *train* dan data *test* yang telah dibagi sebelumnya menggunakan metode *holdout validation* akan diproses untuk melakukan prediksi.

Proporsi data *training* : *testing* yang digunakan pada penelitian ini adalah 70% : 30%. Hasil klasifikasi pada data *train* dan *test* ditampilkan pada tabel berikut :

Tabel 4. *Confusion Matrix* Hasil Klasifikasi *Support Vector Machine* (SVM) dengan Proporsi *Training-Testing* 70%:30%

Aktual Training	Prediksi Training		Aktual Testing	Prediksi Testing	
	Negatif	Positif		Negatif	Positif
Negatif	130	0	Negatif	68	2
Positif	0	234	Positif	12	74

Tabel 4 menampilkan hasil dari Metode *Support Vector Machine* yang menunjukkan bahwa data *test* sudah menghasilkan klasifikasi yang berimbang yaitu tidak hanya mengelompokkan data ke dalam kelas positif atau negatif saja. Performa klasifikasi dapat dihitung secara manual seperti dibawah ini dan hasil perhitungan menggunakan *software python* dilihat pada tabel 5.

$$\text{Akurasi} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{74+68}{74+2+68+12} = 0{,}91025641 = 91{,}03\%$$

$$Recall = \frac{TP}{TP+FN} = \frac{74}{74+12} = 0{,}86046511 = 86{,}05\%$$

$$Precision = \frac{TP}{FP+TP} = \frac{74}{2+74} = 0{,}97368421 = 97{,}37\%$$

$$F1\_Score = \frac{2xPrecisionxRecall}{Precision+Recall} = \frac{2x0{,}97368421x0{,}86046511}{0{,}97368421+0{,}86046511} = 0{,}91358024 = 91{,}36\%$$

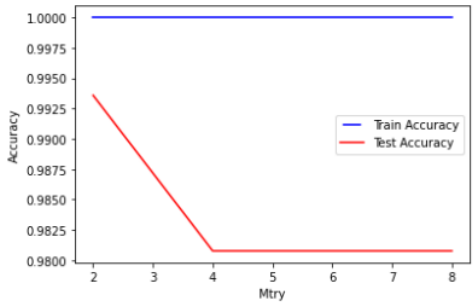
Tabel 5. Ukuran Kebaikan Model *Support Vector Machine*

Ukuran Kebaikan Model	Training-Testing (70%-30%)	
	Data Training	Data Testing
Akurasi	100%	91,03%
Recall	100%	86,05%
Precision	100%	97,37%
F1_score	100%	91,36%

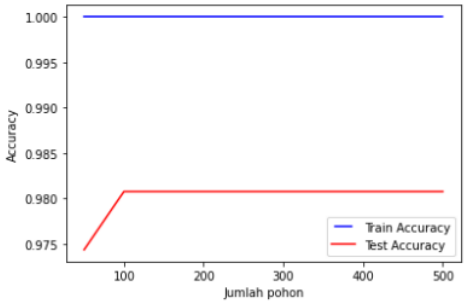
Tabel Tabel 5 menunjukkan bahwa untuk proporsi *training-testing* sebesar 70%-30% menghasilkan akurasi dari data *training-testing* sebesar 100% dan 91% artinya, sebesar 91% pasien yang benar diprediksi terdeteksi Diabetes Melitus dan yang tidak terdeteksi Diabetes Melitus dari keseluruhan pasien. Nilai *recall* untuk data *training-testing* menunjukkan angka 100% dan 86%. Nilai *recall* sudah diatas 70% dan tidak jauh dengan akurasi model sehingga model yang terbentuk bisa dikatakan mampu melakukan klasifikasi dengan tepat pada kedua kelas atau dapat dikatakan sebesar 86% pasien yang diprediksi terdeteksi Diabetes Melitus dibandingkan keseluruhan pasien yang bener terdeteksi Diabetes Melitus. Nilai *precision* untuk data *training-testing* berkisar 100% dan 97% dimana nilai *precision* menunjukkan ketepatan model dalam memprediksi kelas positif Diabetes Melitus dengan benar artinya sebesar 97% pasien yang benar terdeteksi Diaebetes Melitus dari keseluruhan pasien yang diprediksi terdeteksi Diabetes Melitus. Kemudian nilai *F1_score* untuk data *training-testing* juga menunjukkan nilai 100% dan 91%. Data *testing* digunakan untuk mengukur ketepatan klasifikasi di luar data pemodelan.

Langkah-langkah pengklasifikasian menggunakan metode Random Forest (RF) telah dijelaskan pada bab sebelumnya, oleh karena itu akan dilakukan pembentukan model yang sesuai dengan langkah-langkah tersebut. Hal pertama yang dilakukan untuk membentuk suatu model adalah dengan menentukan parameter terbaiknya, berikut merupakan tahapan pengerjaan untuk pembentukan model:

1. Pada *Random Forest* (RF) parameter yang ditentukan adalah nilai dari *mtry (m)* yang merupakan banyaknya jumlah pemilah dalam satu pohon dan *ntree (n)* yang merupakan banyaknya pohon yang akan ditumbuhkan dalam satu *forest*. Oleh karena itu, dilakukan plot dengan mencoba beberapa nilai untuk *mtry* dan *ntree*.



Gambar 1. Akurasi Model *Random Forest* pada Jumlah *Mtry*



Gambar 2. Akurasi Model *Random Forest* pada *Ntree*

Gambar 5 menunjukkan pengaruh perubahan jumlah pemilah terhadap nilai akurasi. Nilai akurasi pada data *train* tampak stabil sedangkan data *test* mulai konvergen pada jumlah pemilah 4. Gambar 6 menunjukkan pengaruh perubahan jumlah pohon terhadap nilai akurasi. Semakin banyak jumlah pohon cenderung menghasilkan nilai akurasi yang lebih baik. Nilai akurasi pada data *train* tampak stabil tetapi data *test* mulai konvergen pada jumlah pohon 100.

2. Dalam penentuan parameter terbaik dari model, dapat dilakukan *tuning* secara simultan dengan *grid search* yaitu dengan mencobakan semua kombinasi parameter yang telah ditentukan. Pada penelitian ini, parameter yang akan dicobakan adalah sebagai berikut:

Tabel 6. Parameter *Random Forest*

Parameter	Nilai Parameter
Mtry	4,5,6,7,8
Ntree	100,150,200,250,500

Tuning parameter dilakukan untuk mencobakan semua kombinasi parameter yang telah ditetapkan sehingga terdapat 25 kombinasi. Proses tuning parameter ini menggunakan 5-*cross validation*, sehingga didapatkan 125 *running model*. Untuk menentukan parameter terbaik didasarkan pada nilai akurasi tertingginya. Proses *tuning parameter* ini menghasilkan model terbaik pada parameter dengan nilai *mtry* = 4 dan *ntree* = 500.

3. Model yang Model yang dibangun menggunakan *Random Forest* (RF) memiliki pohon keputusan sebanyak 500 dan setiap pohon memiliki 4 variabel bebas yang dipilih sebagai pemilah. Data *train* dan data *test* yang telah dibagi sebelumnya menggunakan metode *holdout validation* akan diproses untuk melakukan prediksi. Proporsi data *training* : *testing* yang digunakan pada penelitian ini adalah 70% : 30%. Hasil klasifikasi data *train* dan *test* ditampilkan pada tabel berikut :

Tabel 7. *Confusion Matrix* Hasil Klasifikasi *Random Forest* (RF) dengan Proporsi *Training-Testing* 70%:30%

Aktual Training	Prediksi Training		Aktual Testing	Prediksi Testing	
	Negatif	Positif		Negatif	Positif
Negatif	138	0	Negatif	61	1
Positif	0	226	Positif	2	92

Tabel 7 menampilkan hasil dari metode Random Forest yang menunjukkan bahwa data test sudah menghasilkan klasifikasi yang berimbang yaitu tidak hanya mengelompokkan data ke dalam kelas positif atau negatif saja. Performa klasifikasi dapat dihitung secara manual seperti dibawah ini dan hasil perhitungan menggunakan software python dilihat pada tabel 8.

$$\text{Akurasi} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{92+61}{92+1+61+2} = 0{,}98076923 = 98{,}08\%$$

$$Recall = \frac{TP}{TP+FN} = \frac{92}{92+2} = 0{,}9787234 = 97{,}87\%$$

$$Precision = \frac{TP}{FP+TP} = \frac{92}{1+92} = 0{,}98924731 = 98{,}92\%$$

$$F1\_Score = \frac{2xPrecisionxRecall}{Precision+Recall} = \frac{2x0{,}98924731x0{,}9787234}{0{,}98924731+0{,}9787234} = 0{,}98395722 = 98{,}40\%$$

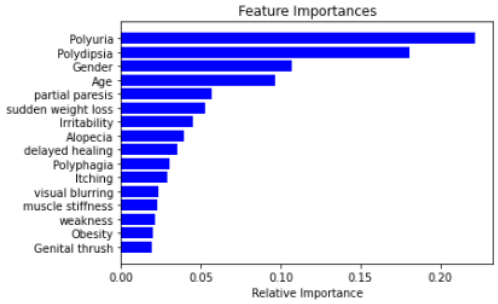
Tabel 8. Ukuran Kebaikan Model *Random Forest*

Ukuran Kebaikan Model	*Training-Testing* (70%-30%)	
	Data *Training*	Data *Testing*
Akurasi	100%	98,08%
Recall	100%	97,87%
Precision	100%	98,92%
F1_score	100%	98,40%

Tabel 8 menunjukkan bahwa untuk proporsi training-testing sebesar 70%-30% menghasilkan akurasi dari data training-testing sebesar 100% dan 98% artinya, sebesar 98% pasien yang benar diprediksi terdeteksi Diabetes Melitus dan yang tidak terdeteksi Diabetes Melitus dari keseluruhan pasien. Nilai recall untuk data training-testing menunjukkan angka 100% dan 98%. Nilai recall sudah diatas 70% dan tidak jauh dengan akurasi model sehingga model yang terbentuk bisa dikatakan mampu melakukan klasifikasi dengan tepat pada kedua

kelas atau dapat dikatakan sebesar 98% pasien yang diprediksi terdeteksi Diabetes Melitus dibandingkan keseluruhan pasien yang bener terdeteksi Diabetes Melitus. Nilai precision untuk data training-testing berkisar 100% dan 99% dimana nilai precision menunjukkan ketepatan model dalam memprediksi kelas positif Diabetes Melitus dengan benar artinya sebesar 99% pasien yang benar terdeteksi Diaebetes Melitus dari keseluruhan pasien yang diprediksi terdeteksi Diabetes Melitus. Kemudian nilai F1_score untuk data training-testing juga menunjukkan nilai 100% dan 98%. Data testing digunakan untuk mengukur ketepatan klasifikasi di luar data pemodelan.

Variabel yang paling berpengaruh terhadap klasifikasi kelas positif atau negatif Diabetes Melitus bisa diketahui dengan membentuk diagram *Feature Importances* dari model *Random Forest* yang sudah dibangun. Feature Importances pada Random Forest ini dihitung menggunakan *Mean Decrease Gini* (MDG). Nilai MDG suatu variabel bebas didapatkan dengan menghitung rata-rata nilai gain variabel tersebut ketika terpilih sebagai pemilah pada setiap pohon. Semakin besar nilai (MDG) maka semakin besar pula pengaruh variabel bebas tersebut. *Feature Importances* pada model *Random Forest* dapat dilihat pada Gambar 3.



Gambar 3. Diagram *Feature Importances*

Feature Importance ini telah diurutkan berdasarkan tingkat kepentingan suatu variabel dalam mempengaruhi seorang pasien terdeteksi Diabetes Melitus masuk ke dalam kategori kelas positif Diabetes Melitus dan negatif Diabetes Melitus.

Dilihat dari *Feature Importances* pada gambar 7, maka dapat disimpulkan bahwa poliuria (*polyuria*) memiliki pengaruh sangat besar dalam menentukan seseorang terdeteksi Diabetes Melitus dari pada variabel lainnya. Variabel yang dianggap paling tidak mempengaruhi seseorang terdeteksi Diabetes Melitus berdasarkan *Feature Importances* adalah infeksi jamur pada kelamin *(genital thrush*).

Hasil evaluasi klasifikasi yang terbentuk pada penelitian ini berjumlah 2 yaitu dengan metode *Support Vector Machine* dan *Random Forest*. Ukuran kebaikan model berupa akurasi, *recall, precision*, dan *F1_score* telah didapatkan seperti yang sudah dijelaskan di pembahasan sebelumnya. Model terbaik adalah model yang memiliki nilai akurasi, *recall, precision*, dan *F1_score* tertinggi. Untuk performa klasifikasi antara kedua metode dapat dilihat pada tabel 9 berikut:

Tabel 9. Performa Klasifikasi *Support Vector Machine* dan *Random Forest*

Ukuran Kebaikan Model	*Support Vector Machine*	*Random Forest*
Akurasi	91,03%	98,08%
Recall	86,05%	97,87%
Precision	97,37%	98,92%
F1_Score	91,36%	98,40%

Berdasarkan Tabel 9, diperoleh informasi bahwa klasifikasi terbaik untuk data deteksi awal risiko Diabetes Melitus adalah dengan menggunakan metode *Random Forest*. Dari data proporsi data *training* 30:70 diperoleh rata-rata akurasi klasifikasi sebesar 98,08%.

5 KESIMPULAN

Berdasarkan hasil dan pembahasan pada bab sebelumnya, diperoleh kesimpulan sebagai berikut:

1. Performa klasifikasi dari metode *Support Vector Machine* menghasilkan nilai akurasi sebesar 91%, *recall* sebesar 86%, *precision* sebesar 97% dan *F1_Score* sebesar 91%.
2. Performa klasifikasi dari metode *Random Forest* menghasilkan nilai akurasi sebesar 98%%, *recall* sebesar 98%, *precision* sebesar 99% dan *F1_Score* sebesar 98%.
3. Berdasarkan performa klasifikasi dari kedua metode, *Random Forest* menghasilkan nilai akurasi yang lebih baik yaitu sebesar 98% untuk deteksi awal risiko Diabetes Melitus.

DAFTAR PUSTAKA

Bergstra, J. and Bengio, Y. (2012) 'Random search for hyper-parameter optimization', Journal of Machine Learning Research, 13, pp. 281–305.

Breiman, L. et al. (1993) Classification and Regression Trees. New York.

Breiman, L. (2001) 'Random Forest', Random Forest, 45, pp. 5–32. doi: 10.1023/A:1010933404324.

Nugroho, A. ., Wranto, A. . and Handoko, D. (2003) Support Vector Machine Teori dan Aplikasinya dalam Bioinformatika.

Prasetyo, E. (2012) 'Data Mining Konsep dan Aplikasi Menggunakan Matlab', in. Yogyakarta: Yogyakarta ANDI.

Santosa, B. (2007) Data Mining: Teknik Pemanfaatan Data untuk Keperluan Bisnis Teori & Aplikasi. Cet. 1. Yogyakarta: Yogyakarta Graha Ilmu , 2007.

Gunn, S. . (1998) Support Vector Machines for Classification and Regression. Southampton.

Kariadi (2009) Diabetes? Siapa Takut!! Panduan Lengkap untuk Diabetesi, Keluarganya, dan Profesional Medis. Bandung: Bandung Qanita.

KLASIFIKASI MENGGUNAKAN METODE SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK DETEKSI AWAL RISIKO DIABETES MELITUS

ORIGINALITY REPORT

9%	9%	0%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	garuda.kemdikbud.go.id Internet Source	6%
2	eprints2.undip.ac.id Internet Source	3%

Exclude quotes	On	Exclude matches	< 3%
Exclude bibliography	On		

KLASIFIKASI MENGGUNAKAN METODE SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK DETEKSI AWAL RISIKO DIABETES MELITUS

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11