IMPLEMENTASI WEB SERVICE DAN ANALISIS KINERJA ALGORITMA KLASIFIKASI DATA MINING UNTUK MEMPREDIKSI DIABETES MELLITUS

Doni Setyawan
Fakultas Ilmu Komputer, Program Studi Teknik Informatika
Universitas Widya Dharma
Email: donnisoft@gmail.com

Agustinus Suradi
Fakultas Ilmu Komputer, Program Studi Manajemen Informatika
Universitas Widya Dharma
Email: simpati2000@mailcity.com

ABSTRAK

Salah satu penyakit yang ditimbulkan akibat kesalahan pola gaya hidup adalah *Diabetes Mellitus* (DM). Gejala penyakit diabetes sering dilalaikan oleh kebanyakan orang, sehingga mereka cenderung untuk mengabaikannya dan tidak mau melakukan *medical check up*. Di Indonesia jumlah penderita DM terus mengalami peningkatan dari tahun ke tahun. *World Health Organization* (WHO) memperkirakan jumlah penderita DM tipe 2 di Indonesia akan mengalami peningkatan secara signifikan hingga 21,3 juta jiwa pada tahun 2030 mendatang. Ternyata dengan bantuan ilmu *data mining*, data pasien diabetes dapat digunakan untuk memprediksi apakah sesorang positif diabetes atau tidak. Tahapan awal dilakukan *preprocessing* data untuk menangani *missing* dan *non numeric value*s. Kemudian *traning* dan *testing* menggunakan *k-fold cross validation* dengan algoritma *K-Nearest Neighbors* (KNN), *random forest* dan *naive bayesian*. Pengujian dilakukan dengan menghitung *accuracy*, *sensitivity* dan *specificity*. Dari hasil uji 10-*fold cross validation* diperoleh rata-rata akurasi tertinggi ketika menggunakan *naive bayesian* yaitu 75,65%, sedangkan KNN 75,53% dan *random forest* 73,69%. Perhitungan *sensitivity* dan *specificity* dengan membagi 786 data menjadi 594 data training dan 192 data testing. Untuk KNN diperoleh *sensitivity* 56,72% dan *specificity* 78,68%, *random forest* diperoleh *sensitivity* 53,73% dan *specificity* 86,4%, sedangkan *naive bayesian* diperoleh *sensitivity* 62,69% dan *specificity* 84%. Implementasi *restful web service* diterapkan pada model dengan akurasi tertinggi yaitu *naive bayesian* dengan format json sebagai *return value*.

Kata kunci: diabetes**,** KNN, *random forest, naive bayesian, web service*.

ABSTRACT

One of the diseases caused by the errors of lifestyle pattern is Diabetes Mellitus (DM). Symptoms of diabetes are often neglected by most people, so they tend to ignore it and do not want to do a medical checkup. In Indonesia the number of DM patients continues to increase from year to year. World Health Organization (WHO) estimates the number of people with type 2 diabetes mellitus in Indonesia will increase significantly to 21.3 million people in the coming 2030. Apparently with the help of data mining science, data about diabetes patients can be used to predict whether a person has diabetes or not. The initial stage is preprocessing data to handle missing and non numeric values. The initial stage is preprocessing data to handle missing and non numeric values. Then the training and testing used k-fold cross validation with K-Nearest Neighbors (KNN) algorithm, random forest and naive Bayesian. Testing is done by calculating accuracy, sensitivity and specificity. From the results of 10-fold cross validation test obtained the highest average accuracy when using naive Bayesian is 75.65%, while KNN 75.53% and random forest 73.69%. The calculation of sensitivity and specificity by dividing 786 data into 594 training data and 192 testing data. For KNN, 56.72% sensitivity and 78.68% specificity, random forest obtained 53.73% sensitivity and 86.4% specificity, while naive Bayesian obtained a sensitivity of 62.69% and 84% specificity. Restful web service implementation is applied to the model with the highest accuracy, i.e. naive Bayesian with json format as the return value.

Keywords: DM**,** KNN, *random forest, naive bayesian, web service.*

1. PENDAHULUAN

Kesehatan merupakan salah satu nikmat berharga dalam kehidupan manusia, dimana dengan sehat aktivitas kehidupan sehari-sehari seperti bekerja, berumah tangga dan bermasyarakat dapat berjalan

dengan baik. Tentunya kesehatan tidak datang dengan sendirinya, tetapi harus dijaga dan diupayakan dalam pola kehidupan seperti pola makan, bekerja, istirahat, olahraga dan pola rohani seseorang. Sayangnya trend kehidupan modern telah membuat orang mengabaikan pola hidup yang sehat. Salah satu penyakit yang ditakuti akibat kesalahan pola gaya hidup adalah *Diabetes Mellitus* (DM) atau sering dikenal dengan diabetes adalah penyakit atau gangguan metabolisme kronis dimana jumlah insulin yang dibutuhkan tubuh dan jumlah insulin yang dihasilkan tubuh tidak sesuai sehingga menyebabkan peningkatan glukosa dalam darah [1]. Gejala penyakit diabetes sebenarnya dapat dikenali, diantaranya sering merasa lapar, tubuh mudah lelah, sering buang air kecil, sering merasa haus, penglihatan menjadi kabur, luka sulit sembuh dan berat badan menurun drastis [2]. Gejala tersebut sering dilalaikan oleh kebanyakan orang, sehingga mereka cenderung untuk mengabaikannya dan tidak mau melakukan *medical check up*.

DM sampai saat ini masih menjadi persoalan kesehatan serius dunia, termasuk di Indonesia yang berada di urutan ke-4 dengan prevalensi diabetes tertinggi di dunia setelah India, China, dan Amerika Serikat. Bahkan, jumlah penderita diabetes terus mengalami peningkatan dari tahun ke tahun, terutama untuk DM tipe 2. *World Health Organization* (WHO) memperkirakan jumlah penderita DM tipe 2 di Indonesia akan mengalami peningkatan secara signifikan hingga 21,3 juta jiwa pada tahun 2030 mendatang [3]. Dengan bantuan teknologi informasi khususnya bidang *data mining*, data mengenai pasien diabetes dapat digunakan untuk memprediksi apakah sesorang menderita diabetes atau tidak.

Data mining adalah proses menemukan pola-pola yang menarik dan pengetahuan dari sekumpulan data yang berjumlah besar. Sumber data dapat berupa basis data, data *warehouse*, web, repositori informasi lainnya ataupun data yang mengalir pada sebuah sistem secara dinamis [4]. Ada banyak algoritma *data mining* yang dapat digunakan untuk memprediksi seseorang menderita diabetes atau tidak. Beberapa penelitian sebelumnya membandingkan kinerja algoritma *decision tree*, KNN, *random forest* dan *Support Vector Machine* (SVM) untuk mengklasifikasi pasien penderita diabetes. Feature yang digunakan berjumlah 8 yaitu berapa kali telah hamil (*number of times pregnant*), Konsentrasi glukosa plasma (*glucose tolerance test*), tekanan darah diastolik (mm Hg), *triceps skin fold thickness* (mm), serum insulin 2 jam (mu U / ml), indeks massa tubuh (berat dalam kg / (tinggi dalam m) ^ 2), fungsi silsilah diabetes, usia (tahun). Pengujian dengan menghitung *accuracy*, *sensitivity* dan *specificity*. Nilai *accuracy* 100 % diperoleh dari algoritma KNN dengan k=1 dan *random forest* [5]. Kinerja KNN juga terbukti unggul dibandingkan dengan *binary logistic regression*, *multilayer perceptron* dimana dengan observasi sebanyak 100 data dengan 7 atribut yaitu *PG concentration, Diastolic BP, Tri Fold Thick, Serum Ins, BMI, DP function, age* dan *disease*. Hasil pengujian *accuracy* dengan *binary logistic regression* 69 %, *multilayer perceptron* 71 % dan KNN 80 % [6].

Algoritma *naive bayesian* juga digunakan untuk mengklasifikasi penyakit diabetes, proses *training* dan *testing* menggunakan *k-fold cross validation*. Akurasi tertinggi diperoleh pada 7 – *fold* dengan rata-rata akurasi 83,89 % [7]. Kinerja algoritma naive bayesian juga diuji dengan optimasi parameter menggunakan algoritma genetika dan terbukti memperoleh akurasi yang lebih tinggi 2,74 % dibandingkan dengan tanpa optimasi parameter [8]. Dibandingkan dengan algoritma C4.5, *naive bayesian* juga memberikan hasil yang lebih baik [9]. Pengujian kinerja naive bayesian juga dilakukan dengan berbagai variasi ukuran data *training* dan *testing* yang menghasilkan akurasi terendah 39 % dan tertinggi 80 % [10]. Berdasar pada hasil penelitian sebelumnya, maka dalam penelitian akan membandingkan kinerja algoritma KNN, *random forest* dan *naive bayesian* dengan 10-*fold cross validation*, model dengan akurasi terbaik akan diimplementasikan dalam bentuk *restful webservice* agar dapat diakses aplikasi lain.

1.1 K-Nearest Neighbors (KNN)

KNN adalah algoritma yang menyimpan semua data *training* yang ada dan mengklasifikasikan data baru berdasar ukuran similaritas, misalnya fungsi jarak. Data baru akan diklasifikaikan ke dalam kelas dari mayoritas data *training* (*neighbors*) yang terdekat [5]. Jarak data baru dengan *neighbors* dapat diukur dengan e*uclidean distance* disajikan pada persamaan (1) [11]. Dimana x_i adalah titik data baru dan x_j adalah titik neighbor, sedangkan *r* adalah jumlah dimensi.

$$dist(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ir} - x_{jr})^2} \quad (1)$$

Tahapan algoritma KNN adalah [12]:

a) Menentukan jumlah k (neigbors), misalnya k = 10.
b) Menghitung jarak antara semua data *training* dengan data baru.
c) Mengurutkan hasil pengukuran jarak dari yang terkecil
d) Mengambil 10 pengukuran dengan jarak yang terdekat

e) Menentukan kelas data baru sesuai dengan kelas data *training* yang mayoritas terdekat dari 10 pengukuran tersebut.

1.2 Random Forest

Random forest merupakan algoritma yang menggunakan *multiple learning* untuk mendapatkan hasil prediksi yang lebih baik, berikut adalah tahapan algortima *random forest* [13]:

a) Gambar n_{tree} sampel *bootstrap* dari data training
b) Untuk setiap sampel *bootstrap*, buat pohon klasifikasi dengan ketentuan : pada setiap *node*, daripada memilih pemisahan (*split*) terbaik diantara semua prediktor, secara acak menyampel m_{try} prediktor dan memilih pemisahan terbaik diantara variabel tersebut.
c) Prediksi data baru dengan menggabungkan prediksi-prediksi dari pohon n_{tree}, yaitu *vooting* dari masing-masing *tree*, kemudian dipilih yang mayoritas.

Estimasi tingkat kesalahan dapat diperoleh berdasarkan *data training* dengan cara [13]:

a) Pada setiap iterasi *bootsrap*, prediksi *out of bag* (OOB) data yaitu data yang tidak ada pada sampel *bootstrap* dengan cara menggunakan pohon yang tumbuh dengan sampel *bootstrap*.
b) Gabungkan prediksi-prediksi OOB.

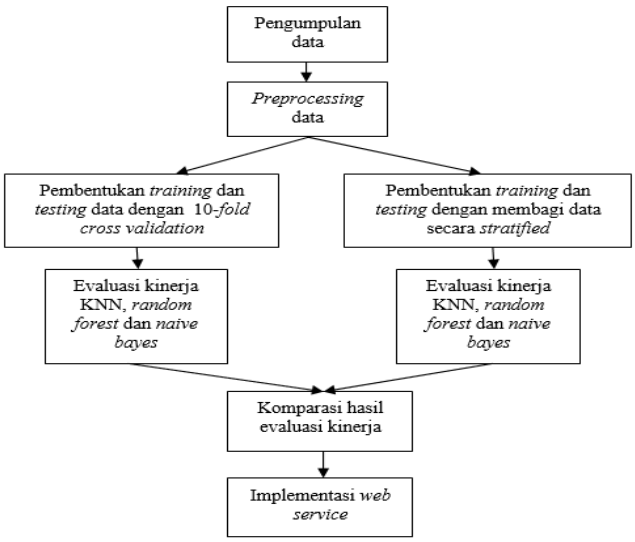
1.3 Naive Bayesian

Supervised learning dapat dilihat dari sudut pandang probabilitas, *naive bayes* merupakan klasifikasi probabilitas yang berdasar pada teorema *bayes* dengan asumsi ketidakbergantungan antar prediktor, yang berarti keberadaan suatu fitur dalam sebuah kelas tidak berhubungan dengan keberadaan fitur yang lain. Data baru akan diberi label dari kelas yang memiliki probabilitas tertinggi. Persamaan (2) menyajikan perhitungan probabilitas suatu kelas, dimana Pr($C = c_j$) adalah jumlah data training dengan kelas c_j dibagi dengan jumlah seluruh data training dan Pr($A_i = a_i \mid C = c_j$) adalah jumlah data training dengan atribut $A_i = a_i$ pada kelas c_j dibagi dengan jumlah data training dengan kelas c_j [11].

$$c = \arg\max_{c_j} \Pr\left(C = c_j\right) \prod_{i=1}^{|A|} \Pr\left(A_i = a_i \mid C = c_j\right) \quad (2)$$

2. METODOLOGI PENELITIAN

Tahapan dalam penelitian ini meliputi pengumpulan data, *preprocessing data*, membagi data *training* dan *testing* menggunakan *k-fold cross validation* dan membagi data *training* dan *testing* dengan *stratified sampling*, kemudian membandingkan akurasi *k-fold cross validation* dan akurasi *stratified sampling* pada algoritma KNN, *random forest* dan *naive bayesian*. Model dengan akurasi tertinggi akan diimplementasikan dalam bentuk *restful webservice*, sebagai mana disajikan pada Gambar 1.



Gambar 1. Tahapan Penelitian

2.1 Pengumpulan Data

Data yang digunakan dalam penelitian ini merupakan data sekunder bersumber dari Pima Indian Diabetes tipe 3 *dataset*, UCI *Machine Learning Repository*. Total data berjumlah 786 dengan rincian sebanyak 500 pasien tidak terdeteksi terkena penyakit diabetes dan 268 pasien terdeteksi penyakit diabetes. Gambaran data pasien penyakit diabetes tipe 3 dapat dilihat pada Tabel 1.

Tabel 1. Data penyakit diabetes tipe 3 [14]

No	*Berapa kali hamil*	*Konsen trasi glukosa*	*Tekanan darah*	*Lipatan kulit*	*Serum insulin*	*Massa tubuh*	*Diabetes type 3 silsilah fungsi*	*Usia*	*Kelas*
1	6	14	148	35	0	33,6	0,627	50	1
2	1	85	66	29	0	26,6	0,351	31	0
3	8	18	64	0	0	23,3	0,672	32	1
4	1	89	66	23	94	28,1	0,167	21	0
5	0	13	40	35	168	43,1	2,288	33	1
6	5	11	74	0	0	25,6	0,201	30	0
7	3	78	50	32	88	31,0	0,248	26	1
8	10	115	115	0	0	35,3	0,134	29	0
9	2	19	70	45	543	30,5	0,158	53	1
10	8	12	96	0	0	0,0	0,232	54	1

2.2 Preprocessing Data

Tahapan selanjutnya adalah *preprocessing* data yang terdiri seleksi integritas data, transformasi data dan pembentukan fitur matrik *X* dan vektor target *y*. Seleksi integritas data akan melakukan pengecekan terhadap *missing value* untuk setiap atribut dari data penelitian, jika ditemukan terdapat baris yang memiliki *missing value* untuk atributnya maka akan diisi dengan nilai rata-rata (*mean*) dari kolom atribut tersebut. Transformasi data memastikan semua nilai data sudah numerik sehingga dapat digunakan untuk proses *training* dan *testing*. Dengan melihat nilai dari suatu atribut, jika ditemukan nilai yang bukan numerik maka akan dirubah menjadi numerik. Pembentukan fitur matrik *X* dan vektor target *y* akan memisahkan atribut / fitur dan target variabel dari Tabel 1. Fitur matrik *X* akan menyimpan data mulai dari kolom berapa kali hamil sampai dengan kolom usia sebagaimana disajikan pada Tabel 2, sedangkan vektor target *y* hanya terdiri dari sebuah kolom yaitu kolom kelas sebagaimana disajikan pada Tabel 3.

Tabel 2. Fitur matrik *X*

No	*Berapa kali hamil*	*Konsen trasi glukosa*	*Tekanan darah*	*Lipatan kulit*	*Serum insulin*	*Massa tubuh*	*Diabetes type 3 silsilah fungsi*	*Usia*
1	6	14	148	35	0	33,6	0,627	50
2	1	85	66	29	0	26,6	0,351	31
3	8	18	64	0	0	23,3	0,672	32
4	1	89	66	23	94	28,1	0,167	21
5	0	13	40	35	168	43,1	2,288	33
6	5	11	74	0	0	25,6	0,201	30
7	3	78	50	32	88	31,0	0,248	26
8	10	115	115	0	0	35,3	0,134	29
9	2	19	70	45	543	30,5	0,158	53
10	8	12	96	0	0	0,0	0,232	54

Tabel 3. Vektor target *y*

Kelas
1
0
1
0
1

Kelas
0
1
0
1
1

2.3 Pembentukan Data Training Dan Testing

Pembentukan data *training* dan *testing* dalam penelitian akan membandingkan 2 pendekatan, 10-*fold cross validation* dan *stratified sampling* pada vektor target *y*, hal ini bertujuan untuk memastikan apakah algoritma yang unggul dengan pendekatan 10-*fold cross validation* juga unggul dengan pendekatan *stratified sampling.*

2.3.1 Training Dan Testing Dengan 10-Fold Cross Validation

Pada tahapan ini akan dilakukan pembentukan *data training* dan *testing* dengan 10-*fold cross validation.* Dalam pendekatan ini *dataset* berjumlah 786 data akan dibagi secara proporsional menjadi 10 bagian (*fold*) sebagaimana disajikan pada Gambar 2. Selanjutnya proses uji akan dilakukan sebanyak 10x, dimana masing-masing *fold* pada gilirannya akan menjadi data *training* dan pada giliran yang lain akan menjadi data *testing* sebagaimana disajikan pada Tabel 4.

Fold ke 1 (F1)	Fold ke 2 (F2)	Fold ke 3 (F3)	Fold ke 4 (F4)	Fold ke 5 (F5)	Fold ke 6 (F6)	Fold ke 7 (F7)	Fold ke 8 (F8)	Fold ke 9 (F9)	Fold ke 10 (F10)

Gambar 2. Pembagian Dataset Menjadi 10 Fold

Tabel 4. Pembentukan data *training* dan *testing* 10-*fold*

Eksprerimen ke	*Data Training*	*Data Testing*
1	F2, F3, F4, F5, F6, F7, F8, F9, F10	F1
2	F1, F3, F4, F5, F6, F7, F8, F9, F10	F2
3	F1, F2, F4, F5, F6, F7, F8, F9, F10	F3
4	F1, F2, F3, F5, F6, F7, F8, F9, F10	F4
5	F1, F2, F3, F4, F6, F7, F8, F9, F10	F5
6	F1, F2, F3, F4, F5, F7, F8, F9, F10	F6
7	F1, F2, F3, F4, F5, F6, F8, F9, F10	F7
8	F1, F2, F3, F4, F5, F6, F7, F9, F10	F8
9	F1, F2, F3, F4, F5, F6, F7, F8, F10	F9
10	F1, F2, F3, F4, F5, F6, F7, F8, F9	F10

2.3.2 Training Dan Testing Dengan Stratified Sampling

Pendekatan *stratified sampling* akan memisah 786 data menjadi data *training* dan *testing* dengan pembagian 75% yaitu 576 *record* untuk data *training* dan 25 % yaitu 192 *record* untuk data *testing*, dengan *stratified* pada vektor target *y* artinya jumlah data yang bernilai 1 dan 0 akan diproporsionalkan.

2.4 Evaluasi Kinerja

Evaluasi kinerja akan dilakukan pada masing – masing pendekatan dengan metrik *accuracy*, *sensitivity* dan *specificity*. Pada pendekatan *k-fold cross validation*, kinerja algoritma KNN, *random forest* dan *naive bayes* akan diuji dengan menghitung akurasi pada setiap eksperimen dari Tabel 4 dengan persamaan (3) [11]. Rata-rata akurasi diperoleh dengan menghitung jumlah total akurasi pada setiap eksperimen dan membaginya dengan 10.

$$Accuracy = \frac{number\ of\ correct\ classifications}{total\ number\ of\ test\ cases} \quad (3)$$

Pada pendekatan *stratified sampling* memanfaatkan *confusion matrix* untuk menghitung *accuracy*, *sensitivity* dan *specificity* sebagaimana disajikan pada persamaan (4), (5) dan (6) [5]. *Confusion matrix* disajikan pada Gambar 3 [5].

total data *testing* = 192		Predicted	
		Negative	Positive
Actual	Negative	TN	FP
	Positive	FN	TP

Gambar 3. *Confusion Matrix* [5]

Keterangan :
TN (True Negatif) : jumlah data *testing* yang sebenarnya negatif dan terprediksi negatif.
TP (True Positif) : jumlah data *testing* yang sebenarnya positif dan terprediksi positif.
FN (False Negatif) : jumlah data *testing* yang sebenarnya positif dan terprediksi negatif.
FP (False Positif) : jumlah data *testing* sebenarnya negatif dan terprediksi positif.

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

$$Sensitivity = \frac{TP}{TP + FN} \quad (5)$$

$$Specificity = \frac{TN}{TN + FP} \quad (6)$$

2.5 Komparasi Hasil Evaluasi Kinerja

Hasil evaluasi kinerja algoritma KNN, *random forest* dan *naive bayes* dengan *training* dan *testing* 10-*fold cross validation* selanjutnya akan dibandingkan untuk mengetahui algoritma manakah yang menghasilkan akurasi terbaik, kemudian untuk memastikan juga akan dibandingkan dengan akurasi dari hasil uji dengan *stratified sampling*.

2.6 Implementasi Web Service

Algoritma dengan hasil akurasi terbaik selanjutnya akan diimplementasikan dalam bentuk *restful web service* menggunakan flask python *web framework*. Web service akan diimplementasikan dengan fungsi *predict*, yang akan menerima parameter masukan berformat json dengan memberikan nilai 'pregn', 'gluc', 'bp', 'sk', 'ins', 'bmi', 'ped', 'age' seperti nilai-nilai pada Tabel 2. *Response value* juga berformat json dengan nilai 1 berarti positif diabetes dan nilai 0 berarti negatif diabetes.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Kinerja KNN, Random Forest Dan Naive Bayes Dengan 10-Fold Cross Validation

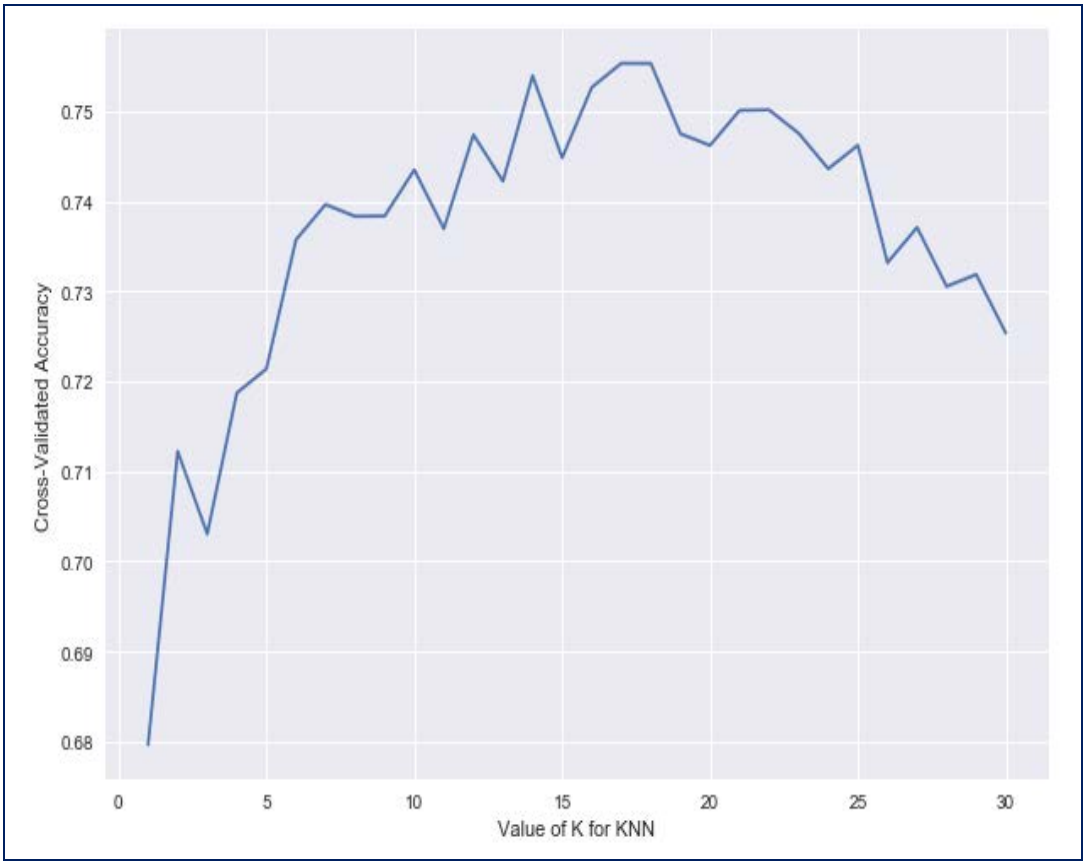
Kinerja KNN dipengaruhi oleh jumlah *neighbors* yang digunakan dalam proses *training*, sehingga untuk mendapatkan *classifier* KNN yang optimum maka akan dilakukan ujicoba dengan jumlah *neighbors* dari 1 sampai 30 dengan proses *training* dan *testing* untuk KNN akan dilakukan sebanyak 30 kali, dimana pada setiap *training* dan *testing* menggunakan 10-*fold cross validation*, kemudian akan dipilih *classifier* KNN dengan jumlah *neighbors* yang menghasilkan akurasi paling tinggi. Gambar 4 menampilkan grafik rata-rata akurasi mulai dari jumlah *neighbors* 1 sampai 30, terlihat nilai akurasi semakin tinggi mengikuti jumlah neighbors, tetapi nilai akurasi menurun ketika neighbors melebihi 20, sedangkan nilai akurasi maksimal diperoleh ketika jumlah neighbors = 17, yaitu 0.755297334245 (75,53 %). Hasil algoritma *random forest* memberikan kinerja yang lebih rendah dibandingkan KNN. Tabel 5 menyajikan akurasi dari 10 eksperimen dengan rata-rata akurasi 0.73696 (73, 69 %). Hasil yang baik diperoleh oleh algoritma *naive bayes* dengan rata-rata akurasi 0.75649 (75, 65 %), Tabel 6 menyajikan hasil uji akurasi naive bayes dengan 10 eksperimen. Berdasarkan hasil uji akurasi KNN, random forest, dan naive bayesian dengan 10-*fold cross validation* diperoleh akurasi tertinggi 75,65 % dengan *naive bayesian*.

Tabel 5. Uji akurasi *random forest* dengan 10-*fold cross validation*

Uji akurasi eksperimen ke									
1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
0.67532	0.81818	0.71428	0.7012	0.68831	0.74025	0.74025	0.83116	0.71052	0.75

Tabel 6. Uji akurasi *naive bayes* dengan 10-*fold cross validation*

Uji akurasi eksperimen ke									
1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
0.72727	0.75324	0.79220	0.71428	0.71428	0.7922	0.76623	0.80519	0.72368	0.77631



Gambar 4. Grafik Rata-Rata Akurasi KNN Dengan Jumlah Neighbors 1-30

3.2 Hasil Kinerja KNN, Random Forest Dan Naive Bayes Dengan Stratified Sampling

Pada pendekatan *stratifed sampling* akan menggunakan *confusion matrix* untuk menghitung *accuracy*, *sensitivity* dan *specificity*. Tabel 7 menyajikan *confusion matrix* dari pengujian menggunakan KNN. Sesuai persamaan (4), (5) dan (6) maka diperoleh nilai *accuracy* 75,52 %, *sensitivity* 56,72 % dan *specificity* 78,68 %. Tabel 8 menyajikan *confusion matrix* dari pengujian menggunakan *random forest*, sehingga diperoleh nilai *accuracy* 75 %, *sensitivity* 53,73 % dan *specificity* 86,4 %.

Tabel 7. *Confusion matrix* untuk KNN

Total data testing = 192		***Predicted***	
		Negative	***Positive***
Actual	*Negative*	107	18
	Positive	29	38

Tabel 8. *Confusion matrix* untuk *random forest*

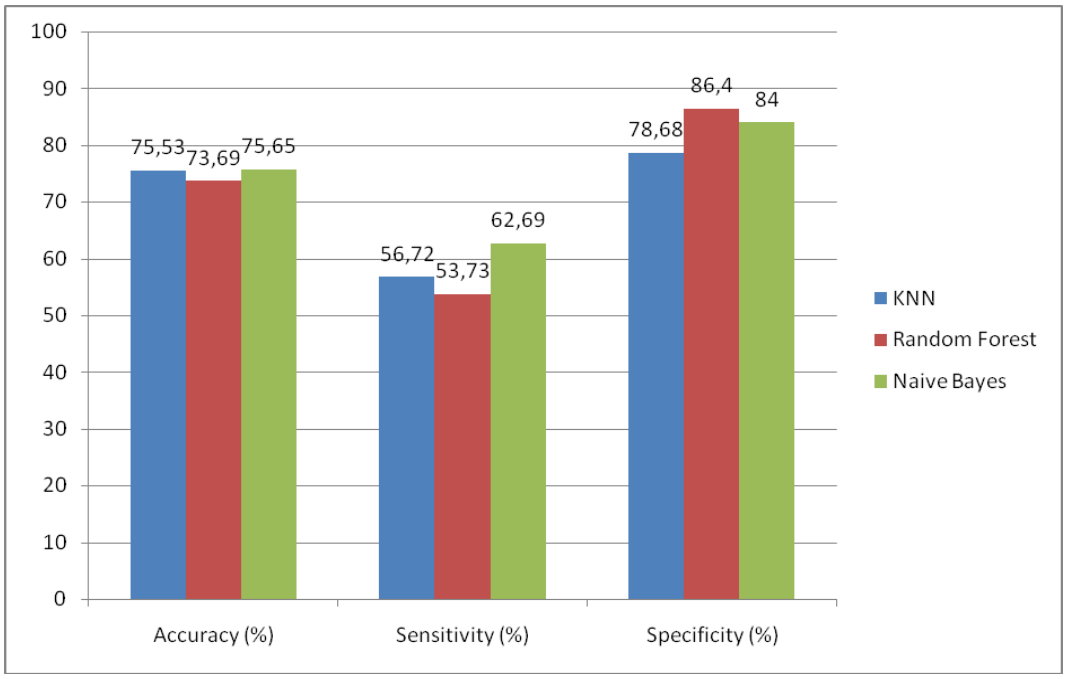
Total data testing = 192		***Predicted***	
		Negative	***Positive***
Actual	*Negative*	108	17
	Positive	31	36

Tabel 9 menyajikan *confusion matrix* dari pengujian menggunakan *naive bayes*, sehingga diperoleh nilai *accuracy* 76,56 %, *sensitivity* 62,69 % dan *specificity* 84 %.

Tabel 9. *Confusion matrix* untuk *naive bayes*

Total data testing = 192		***Predicted***	
		Negative	***Positive***
Actual	*Negative*	105	20
	Positive	25	42

Dari hasil uji ketiga algoritma terbukti naive bayes menghasilkan akurasi yang tertinggi, baik dengan 10-*fold cross validation* maupun *stratified sampling*. Gambar 5 menampilkan grafik perbandingan kinerja ketiga algoritma, nilai *accuracy* yang ditampilkan mengambil dari hasil 10-*fold cross validation*.



Gambar 5. Perbandingan Kinerja KNN, *Random Forest* Dan *Naive Bayes*

3.3 Implementasi Web Service

Implementasi *web service* menggunakan flask *web framework*, *library* pandas untuk membaca file csv dan membentuk fitur matrix *X* dan vektor target *y*. Penggunan *library* scikit-learn agar dapat menggunakan modul naive bayes, serta untuk melakukan pembagian data *training* dan *testing* dengan fungsi train_test_split. Modul terakhir yang digunakan adalah pickle untuk menyimpan *classifier* yang telah dibuat ke dalam disk, agar tidak melakukan *training* berulang-ulang untuk setiap *request* yang dikirim. Listing program 1 berfungsi untuk membuat *naive bayes classifier*, kemudian menyimpannya dengan nama nb_pimadiebets.pkl agar dapat dipanggil oleh *web service*. Listing program 2 menampilkan kode pembuatan *web service* yang diimplementasikan pada fungsi *predict*, sedangkan Listing program 3 menampilkan contoh kode aplikasi dalam bahasa python yang melakukan request ke *web service* dengan mengirimkan parameter 'pregn':6,'gluc':148,'bp':72,'sk':35,'ins':0,'bmi':33.6,'ped':0.627,'age':50, dimana akan menghasilkan variabel output dengan nilai {'results': [1]} yang berarti positif diabetes.

Listing program 1. Pembuatan *naive bayes classifier*

```
url = 'pima-indians-diabetes.csv'
col_names = ['pregnant', 'glucose', 'bp', 'skin',
 'insulin', 'bmi', 'pedigree', 'age', 'label']
pima = pd.read_csv(url, header=None, names=col_names)
feature_cols = ['pregnant', 'glucose', 'bp', 'skin',
 'insulin', 'bmi', 'pedigree', 'age']
X = pima[feature_cols]
y = pima.label
X_train, X_test, y_train, y_test =
train_test_split(X, y, stratify=y, test_size=0.25, random_state = 0)
nb = GaussianNB()
nb.fit(X_train, y_train)
pickle.dump(nb, open("nb_pimadiabtes.pkl","wb"))
```

Listing program 2. Implementasi *web service* dengan flask

```
import numpy as np
from flask import Flask, request, abort, jsonify
import pickle

nbclassifier = pickle.load(open("nb_pimadiabtes.pkl","rb"))
app = Flask(__name__)

@app.route('/api', methods=['POST'])
def predict():
    data = request.get_json(force = True)
    predict_request = [data['pregn'],data['gluc'],data['bp']
    data['sk'],data['ins'],data['bmi'],data['ped'],data['age']]
    predict_request = np.array(predict_request)
    y_result = nbclassifier.predict(predict_request)
    output = [int(y_result[0])]
    return jsonify(results=output)

if __name__ == '__main__':
    app.run(debug = True)
```

Listing program 3. Contoh kode yang mengakses *web service*

```
import json
import requests
url = "http://127.0.0.1:5000/api"
```

Listing program 3. Lanjutan

```
data= json.dumps({'pregn':6,'gluc':148,'bp':72,'sk':35,
'ins':0,'bmi':33.6,
'ped':0.627,'age':50})
r = requests.post(url, data)
output = r.json()
```

4. KESIMPULAN

Dari penelitian yang telah dilakukan, diperoleh beberapa kesimpulan, yaitu dengan pendekatan 10-*fold cross validation* diperoleh nilai akurasi tertinggi ketika menggunakan *naive bayes* yaitu 75,65 %, sedangkan nilai akurasi untuk KNN 75,53 % dengan jumlah neighbors = 17 dan *random forest* 73, 69 %. Naive bayes juga terbukti unggul ketika menggunakan pendekatan *stratified sampling* dengan akurasi 75, 56 %. *Sensitivity* terbaik juga dihasilkan *naive bayes* yaitu 62, 69 %, sedangkan *specificity* terbaik dihasilkan oleh *random forest* yaitu 86,4 %. Penelitian selanjutnya dapat menggunakan *hybrid methode* seperti *naive bayesian* dengan *neural network* ataupun KNN dengan SVM untuk melakukan klasifikasi, diperlukan juga melakukan *feature engineering* terhadap fitur-fitur yang digunakan untuk mendapatkan akurasi yang lebih maksimal.

DAFTAR PUSTAKA

[1] Kuncoro, S. (2015). Apa Itu Diabetes Mellitus: Penyebab, Gejala, Pengobatan, http://www.pasiensehat.com/2015/01/pengertian-diabetes-mellitus-adalah.html, 23 Januari 2015, diakses 26 Agustus 2017.
[2] Ana. (2015). *29 Gejala Diabetes Melitus Awal Pada Pria dan Wanita*. http://halosehat.com/penyakit/diabetes/gejala-diabetes. 6 November 2015. diakses 26 Agustus 2017.
[3] Ika. (2016). *60 Persen Penderita Diabetes Tidak Sadar Mengidap Diabetes*. https://www.ugm.ac.id/id/berita/11467-60.persen.penderita.diabetes.tidak.sadar.mengidap.diabetes. 6 April 2016. diakses 26 Agustus 2017.
[4] Han, J., Kamber, M. dan Pei, J. (2012). *Data Mining: Concepts and Techniques 2e*. Morgan Kaufmann Publishers. San Francisco.
[5] Kandhasamy, J.P. and Balamurali, S. 2015. "Performance analysis of classifier models to predict diabetes mellitus". *Procedia Computer Science*. 47. pp.45-51
[6] Selvakumar, S., Kannan, K. S., & GothaiNachiyar, S. 2017. "Prediction of Diabetes Diagnosis Using Classification Based Data Mining Techniques". *International Journal of Statistics and Systems*. 12(2). 183-188.
[7] Prasetya, T.W.H. 2016. "Klasifikasi Diagnosa Diebetes Mellitus Dengan Penerapan Metode Naive Bayesian Clasifier". *Skripsi*. Program Studi Teknik Informatika Universitas Sanata Dharma. Yogyakarta.
[8] Handayanna, F., Rinawati, R., Arisawati, E., & Dewi, L. S. 2017. "Prediksi Penyakit Diabetes Menggunakan Naive Bayes Dengan Optimasi Parameter Menggunakan Algoritma Genetika". *Konferensi Nasional Ilmu Sosial & Teknologi*. 1(1).
[9] Fatmawati, F. 2016. "Perbandingan Algoritma Klasifikasi Data Mining Model C4. 5 Dan Naive Bayes Untuk Prediksi Penyakit Diabetes". *Jurnal Techno Nusa Mandiri*. 13(1). 50-59.
[10] Qurnia, I. L., Prasetyo, E., & Zainal, R. F. 2017. "Classification Of Diabetes Disease Using Naive Bayes Case Study: Siti Khadijah Hospital". *Journal Of Electrical Engineering And Computer Sciences*. Vol 1 Number 2. 1(2)
[11] Liu, B. (2011). *Web Data Mining Exploring Hyperlinks, Contents, and Usage Data 2nd*. Springer. New York.
[12] Larose, D. (2005). *Discovering Knowledge in Data: An Introduction to Data Mining*. John Wiley & Sons. Hoboken. NJ.
[13] Liaw, A., & Wiener, M. (2002). *Classification and regression by randomForest*. R news, 2 (3), 18-22.
[14] Sigillito, V. (1990). *Pima Indians Diabetes Data Set*. https://archive.ics.uci.edu/ml/datasets/pima+indians+diabetes. 9 Mei 1990. di akses 20 Agustus 2017.