Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) p-ISSN: 2355-7699
Vol. 4, No. 3, September 2017, hlm. 180-184 e-ISSN: 2528-6579

OPTIMASI *NAÏVE BAYES CLASSIFIER* DENGAN MENGGUNAKAN *PARTICLE SWARM OPTIMIZATION* PADA DATA IRIS

Husin Muhamad[1], Cahyo Adi Prasojo[2], Nur Afifah Sugianto[3], Listiya Surtiningsih[4], Imam Cholissodin[5]

[1,2,3,4,5] Fakultas Ilmu Komputer Universitas Brawijaya
Email: [1]husin.muh@outlook.com, [2]cahyo.ap@outlook.co.id, [3]afifahnur30@gmail.com,
[4]listiyasurtiningsing@gmail.com, [5]imamcs@ub.ac.id

(Naskah masuk: 12 Januari 2017, diterima untuk diterbitkan: 25 September 2017)

Abstrak

Klasifikasi adalah proses identifikasi obyek kedalam sebuah kelas, kelompok, atau kategori berdasarkan karakteristik yang telah ditentukan sebelumnya. Secara singkat, klasifikasi merupakan pengelompokan obyek berdasarkan kelompoknya yang biasanya disebut dengan kelas (*class*). Tak hanya klasifikasi, proses pengelompokkan obyek juga dapat dilakukan dengan menggunakan teknik *clustering* yang merupakan pengelompokan obyek berdasarkan kemiripan antar obyek. Salah satu metode klasifikasi yang sering digunakan adalah *Naïve Bayes Classifier*. Menurut beberapa penelitian, *Naïve Bayes Classifier* memiliki beberapa kelebihan yaitu, cepat dalam proses perhitungan, algoritma yang sederhana dan akurasi yang tinggi. Namun probabilitas pada *Naïve Bayes Classifier* tidak bisa mengukur seberapa besar tingkat keakuratan sebuah prediksi, hasil akurasi metode ini juga masih kurang jika dibandingkan dengan metode C4.5, selain itu metode naïve bayes juga memiliki kelemahan pada seleksi atribut. Untuk menyelesaikan permasalahan tersebut, algoritma *particle swarm optimization* (PSO) dapat digunakan untuk melakukan pembobotan atribut untuk meningkatkan akurasi *naïve bayes classifier*.

Kata kunci: *Naïve Bayes Classifier, Particle Swarm Optimization, klasifikasi, pembobotan atribut.*

Abstract

Classification is the process of identifying objects into a class, group or category based on the predetermined characteristics. In other words, classification is a process to group objects based on their class. Grouping objects can be done not only by classification but also by clustering, which is grouping objects according to the similarity between objects. One of the most frequently used methods for classification is Naïve Bayes Classifier. According to some researchers, Naïve Bayes methods has its strength which is a simple and fast algorithm that can acquire a high accuracy. However, the probability of Naïve Bayes methods cannot measure the level of accuracy of a prediction, the accuracy of the results of this method is still less than the C4.5 method, and Naïve Bayes method has a deficiency on the selection of attributes. To solve this problem, Particle Swarm Optimization Algorithm (PSO) can be used to give weight to attributes to improve the accuracy of Naïve Bayes Classifier.

Keywords: *Naïve Bayes Classifier, Particle Swarm Optimization, classification, attribute weighting.*

1. PENDAHULUAN

Klasifikasi adalah proses pengidentifikasian obyek ke dalam sebuah kategori, kelas atau kelompok berdasarkan prosedur, definisi dan karakteristik yang telah ditentukan sebelumnya (U.S Fish and Wildlife Service, 2013). Klasifikasi bertujuan untuk menempatkan objek yang ditugaskan hanya ke salah satu kategori yang disebut kelas (Bramer, 2007). Tak hanya klasifikasi, proses pengelompokkan obyek juga dapat dilakukan dengan menggunakna teknik *clustering*. *Clustering* merupakan pengelompokan obyek berdasarkan kemiripan antar obyek. Perbedaan antara klasifikasi dan *clustering* terletak pada proses pengelompokan obyek. Jika pada klasifikasi proses pengelompokan obyek dilakukan dengan membagi obyek berdasarkan kelompok / kategori yang telah didefinisikan sebelumnya, makaproses pengelompokan obyek pada *clustering* dilakukan dengan melihat kemiripan antar obyek, sehingga kategori belum terdefinisi sebelumnya. Salah satu metode klasifikasi yang sering digunakan adalah *Naïve Bayes Classifier* yang pertama kali dikemukakan oleh Revered Thomas Bayes. Penggunaan *Naïve Bayes Classifier* sudah dikenalkan sejak tahun 1702-1761. Menurut Lewis, Hand dan Yu, *Naïve Bayes Classifier* merupakan pendekatan yang sangat sederhana dan sangat efektif untuk pelatihan klasifikasi (Lewis, 1998) (Hand and Yu, 2001). Sedangkan Kononenko dan Langley menyimpulkan bahwa *Naïve Bayes Classifier* merupakan kemungkinan label kelas data atau bisa diasumsikan sebagai atribut kelas yang diberi label (Kononenko, 1990) (Langley, 1994).

Penelitian terkait penggunaan *Naïve Bayes Classifier* telah banyak dilakukan. Salah satunya penelitian yang dilakukan oleh Hamzah pada tahun 2012. Menurut Hamzah, Naïve Bayes memiliki beberapa kelebihan, yaitu cepat dalam perhitungan, algoritma yang sederhana dan berakurasi tinggi (Hamzah, 2012). Selain itu, penelitian yang dilakukan oleh Henny Leidiyana juga mengungkapkan bahwa algoritma NBC hasil akurasinya masih kurang dibandingkan menggunakan algoritma C4.5 (Leidiyana, 2012). Hal ini dikarenakan dalam C4.5 seluruh atribut diseleksi yang kemudian dibagi menjadi himpunan bagian yang lebih kecil, namun jika data berukuran besar dengan banyak atribut maka model yang terbentuk menjadi rumit dan sulit dipahami, sehingga perlu dilakukan pemangkasan yang dapat mengurangi akurasi (Wu et al, 2009). Sedangkan *Naïve Bayes Classifier* lebih tepat diterapkan pada data yang besar dan dapat menangani data yang tidak lengkap (*missing value*) serta kuat terhadap atribut yang tidak relevan dan noise pada data. Akan tetapi, *Naïve Bayes Classifier* juga memiliki kelemahan dimana sebuah probabilitas tidak bisa mengukur seberapa besar tingkat keakuratan sebuah prediksi. Selain itu, *Naïve Bayes Classifier* juga memiliki kelemahan pada seleksi atribut sehingga dapat mempengaruhi nilai akurasi. Oleh karena itu, *Naïve Bayes Classifier* perlu dioptimasi dengan cara memberikan bobot pada atribut agar *Naïve Bayes Classifier* dapat bekerja lebih efektif.

Untuk menyelesaikan permasalahan tersebut, algoritma *Particle Swarm Optimization* (PSO) dapat digunakan untuk melakukan pembobotan atribut untuk meningkatkan akurasi *Naïve Bayes Classifier*. Data yang digunakan pada paper ini adalah data iris yang diambil dari *UCI Machine Learning* yang terdiri dari 150 dataset yang terbagi menjadi 3 kelas dan 4 atribut, yaitu *sepal length*, *sepal width*, *petal length* dan *petal width*. Dengan adanya algoritma *Particle Swarm Optimization* dalam optimasi diharapkan akan menambah akurasi dari *Naïve Bayes Classifier*.

2. DASAR TEORI

2.1 Penjelasan Dataset

Data yang digunakan pada penelitian ini adalah data Iris yang diambil dari *UCI Machine Learning*. Data yang digunakan dibagi menjadi dua bagian yaitu *data training* dan *data testing.* Data tersebut terdiri dari 150 dataset yang terbagi menjadi 3 kelas dan 4 atribut, yaitu:

Class:
1. Iris Sentosa
2. Iris VersiColor
3. Iris Virginica

Atribut:
1. *Sepal Length*
2. *Sepal Width*
3. *Petal Length*
4. *Petal Width*

2.2 Klasifikasi

Klasifikasi adalah proses pengidentifikasian obyek ke dalam sebuah kategori, kelas atau kelompok berdasarkan prosedur, definisi dan karakteristik yang telah ditentukan sebelumnya. Klasifikasi bertujuan untuk menempatkan objek yang ditugaskan hanya ke salah satu kategori yang disebut kelas.

2.3 Algoritma *Naïve Bayes Clasifier*

Naïve Bayes Clasifier atau disebut juga dengan *Bayesian Classification* merupakan metode pengklasifikasian statistik yang didasarkan pada teorema *bayes* yang dapat digunakan untuk memprediksi probabilitas keanggotaan suatu kelas. *Bayesian Classification* terbukti memiliki akurasi dan kecepatan yang tinggi saat diaplikasikan ke dalam *database* yang besar.

Bentuk umum teorema *bayes* adalah sebagai berikut:

$$P(H|X) = \frac{P(X|H)\,P(H)}{P(X)} \quad (1)$$

Dimana :
X = Data dengan kelas yang belum diketahui
H = Hipotesa data X merupakan suatu kelas spesifik
P(H|X) = Probabilitas hipotesis H berdasarkan kondisi X (posterior probability)
P(H) = Probabilitas hipotesis H (prior probability)

Peluang bersyarat atribut kategorikal dinyatakan dalam bentuk sebagai berikut:

$$P(A_i|C_j) = \frac{|A_{ij}|}{N_{Cj}} \quad (2)$$

Dimana $|A_{ij}|$ adalah jumlah contoh pelatihan dari kelas A_i yang menerima nilai C_j. Jika hasilnya adalah nol, maka menggunakan pendekatan berikut:

$$P(A_i|C_j) = \frac{n_C + n_{equiv}\,p}{n + n_{equiv}} \quad (3)$$

Dimana n adalah total dari jumlah hasil dari kelas C_j. n_C adalah jumlah contoh pelatihan dari kelas A_i yang menerima nilai C_j. n_{equiv} adalah nilai konstan dari ukuran sampel yang ekuivalen. *P* adalah peluang estimasi *prior*, $P = 1/k$ dimana k adalah jumlah kelas dalam variabel target.

Peluang bersyarat atribut kontinu dinyatakan dalam bentuk berikut:

$$P(A_i|C_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}}} exp^{[-\frac{(A_i-\mu_{ij})^2}{2(\sigma_{ij})^2}]} \quad (4)$$

Parameter μ_{ij} dapat diestimasi berdasarkan sampel *mean* A_i untuk seluruh hasil pelatihan yang dimiliki kelas C_j. Dengan cara sama, $(\sigma_{ij})^2$ dapat diestimasi dari sampel varian (s^2) hasil pelatihan tersebut.

2.4 *Particle Swarm Optimization (PSO)*

Particle Swarm Optimization (PSO) adalah metode optimasi global yang diperkenalkan oleh Kennedy dan Eberhart pada tahun 1995 berdasarkan penelitian terhadap perilaku kawanan burung dan ikan. Setiap partikel dalam *Particle Swarm Optimization* memiliki kecepatan partikel bergerak dalam ruang pencarian dengan kecepatan yang dinamis disesuaikan dengan perilaku historis mereka. Oleh karena itu, partikel memiliki kecenderungan untuk bergerak menuju daerah pencarian yang lebih baik selama proses pencarian.

Dalam algoritma PSO terdapat beberapa proses sebagai berikut:

1. **Inisialisasi**
   a. Inisialisasi kecepatan awal
      Pada iterasi ke-0, dapat dipastikan bahwa nilai kecepatan awal semua partikel adalah 0.
   b. Inisialisasi posisi awal partikel
      Pada iterasi ke-0, posisi awal partikel dibangkitkan dengan persamaan :

$$x = x_{min} + rand[0,1] \times (x_{max} - x_{min}) \quad (5)$$

   c. Inisialisasi pBest dan gBest
      Pada iterasi ke-0, pBest akan disamakan dengan nilai posisi awal partikel. Sedangkan gBest dipilih dari satu pBest dengan *fitness* tertinggi.
2. ***Update* kecepatan**
   Untuk melakukan *update* kecepatan, digunakan rumus berikut:

$$v_{i,j}^{t+1} = w.v_{i,j}^{t} + c_1.r_1\left(Pbest_{i,j}^{t} - x_{i,j}^{t}\right) + c_2.r_2\left(Gbest_{g,j}^{t} - x_{i,j}^{t}\right) \quad (6)$$

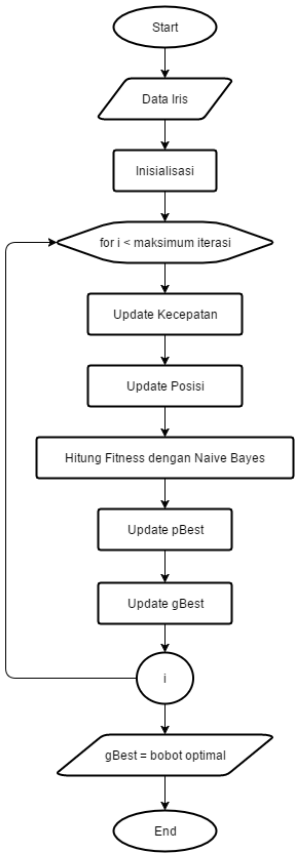
3. ***Update* posisi dan hitung *fitness***
   Untuk melakukan update posisi, digunakan rumus berikut:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + v_{i,j}^{t+1} \quad (7)$$

4. ***Update* pBest dan gBest**
   Dilakukan perbandingan antara pBest pada iterasi sebelumnya dengan hasil dari *update* posisi. *Fitness* yang lebih tinggi akan menjadi pBest yang baru. pBest terbaru yang memiliki nilai *fitness* tertinggi akan menjadi gBest yang baru.

3. PERANCANGAN DAN IMPLEMENTASI

Proses optimasi *Naïve Bayes Classifier* menggunakan algoritma *Particle Swarm Optimization* pada data iris ditunjukkan pada Gambar 1 berikut:



Gambar 1. Diagram Alir Kombinasi NBC dan PSO

Implementasi sistem klasifikasi menggunakan metode NBC dan PSO terdiri dari beberapa masukan yaitu jumlah partikel, jumlah iterasi, bobot inersia, konstanta kecepatan 1 dan 2. Jumlah partikel digunakan untuk menentukan banyaknya *popsize* pada PSO. Partikel direpresentasikan dengan bobot tiap atribut yang akan dioptimasi. Kemudian setiap *data training* dan *data testing* akan di kalikan dengan bobot.

Proses ini merupakan proses menghitung *fitness* dari data testing. Dengan mencari hasil klasifikasi setiap data dengan NBC dan kemudian dihitung akurasinya sebagai *fitness*.

```
1  public void hitFitness() {
2          double[][] dataTest =
3                getDataTest();
4          String[] hasilKlasifikasi = new
5                String[dataTest.length];
6
7     for(int
8     i=0;i<hasilKlasifikasi.length; i++){
9           hasilKlasifikasi[i] = new
10          Bayes(dataTest[i],
11          posisi).hasilKlasifikasi();
12    }
13
14    int fit = 0;
15    for(int i=0; i<17; i++) {
16         if(hasilKlasifikasi[i]
17         .equals("sentosa")) {
18            fit++;
19         }
20    }
21    for(int i=17; i<34; i++) {
22        if(hasilKlasifikasi[i]
23        .equals("versicolor")) {
24             fit++;
25        }
26    }
27    for(int i=34; i<51; i++){
28        if(hasilKlasifikasi[i]
29        .equals("virginica")) {
30             fit++;
31        }
32    }
33    fitness = fit;
34 }
```

Kode program 1. Perhitungan *fitness*

Penjelasan Kode Program 1:

1. Baris 1-12 merupakan proses mengambil hasil klasifikasi dengan menggunakan NBC
2. Baris 14-34 merupakan proses perhitungan *fitness* dengan menggunakan akurasi klasifikasi.

```
run:
Inisialisasi
==================================================

        Partikel 1 :
                Posisi : 0.2, 0.3, 0.4, 0.1,
                Kecepatan : 0.0, 0.0, 0.0, 0.0,
                Fitness : 36.0
        Partikel 2 :
                Posisi : 0.9, 0.2, 0.9, 0.1,
                Kecepatan : 0.0, 0.0, 0.0, 0.0,
                Fitness : 36.0
        Partikel 3 :
                Posisi : 0.7, 0.3, 0.7, 0.2,
                Kecepatan : 0.0, 0.0, 0.0, 0.0,
                Fitness : 47.0
```

Gambar 2. Hasil inisialisasi partikel awal

```
==========
      gBest
            Posisi : 0.8907999999999998, 0.4382999999999997, 0.7782399999999998, 0.5004,
            Kecepatan : 0.04419999999999996, 0.05639999999999985, -0.04868000000000011, 0.10619999999999997,
            Fitness : 47.0

BUILD SUCCESSFUL (total time: 1 second)
```

Gambar 3. Hasil *gBest* iterasi terakhir

4. PENGUJIAN DAN ANALISIS

4.1 Pengujian Jumlah Partikel

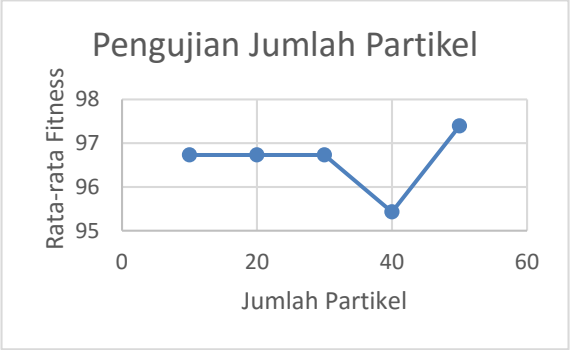
Pada pengujian ini, akan dilakukan pengujian jumlah partikel dengan kelipatan 10 dan iterasi 100 kali dengan perobaan sebanyak 3 kali di setiap pengujian.

Tabel 1. Pengujian Jumlah Partikel

Jumlah Partikel	Percobaan *fitness* ke - i			Rata-rata *fitness*
	1	2	3	
10	94.12	98.04	98.04	96.73
20	98.04	94.12	98.04	96.73
30	98.04	98.04	94.12	96.73
40	94.12	98.04	94.12	95.43
50	98.04	98.04	96.08	97.39

Pengujian ini dilakukan dengan beberapa jumlah partikel berbeda dengan kelipatan 10 seperti yang dicantumkan pada Tabel 1, dan untuk setiap jumlah partikel akan dijalankan sebanyak 3 kali percobaan. Dimana hasil akan dilihat adalah nilai rata-rata *fitness* tertinggi.

Berdasarkan hasil pengujian jumlah partikel dalam 3 kali percobaan pada Tabel 1, rata-rata *fitness* terbaik didapatkan pada partikel 50. Jumlah partikel tersebut menghasilkan rata-rata *fitness* tertinggi sebanyak 97.39. Hasil pengujian dapat dilihat dari grafik pada Gambar 4.



Gambar 4. Grafik pengujian jumlah partikel

4.2 Pengujian Kombinasi Parameter

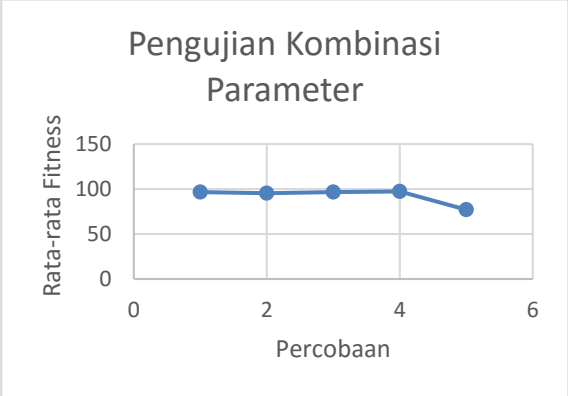
Pada pengujian ini, dilakukan uji coba kombinasi parameter C_1 dan C_2 dengan jumlah partikel 10, iterasi sebanyak 50 dan bobot (w) dengan nilai 1.

Tabel 2. Pengujian Kombinasi Parameter

C1	C2	Percobaan *fitness* ke – *i*			Rata-rata *fitness*
		1	2	3	
0.2	0.5	94.12	98.04	98.04	96.73
0.3	0.4	92.17	98.04	96.08	95.43
0.2	0.3	98.04	98.04	94.12	96.73
0.8	0.3	96.08	98.04	98.04	97.39
0.9	0.4	35.29	98.04	98.04	77.12

Pengujian ini dilakukan dengan kombinasi parameter yang diambil secara *random* dan untuk setiap kombinasi parameter akan dijalankan sebanyak 3 kali percobaan. Dimana hasil akan dilihat adalah nilai rata-rata *fitness* tertinggi.

Berdasarkan hasil pengujian kombinasi parameter dalam 3 kali percobaan pada Tabel 2, rata-rata *fitness* tertinggi didapatkan pada kombinasi parameter C1 dengan nilai 0.8 dan C2 dengan nilai 0.3. Kombinasi parameter tersebut menghasilkan rata-rata *fitness* tertinggi sebanyak 97.39. Hasil pengujian ini dapat dilihat dari grafik pada Gambar 5.



Gambar 5. Grafik pengujian kombinasi parameter

5. KESIMPULAN DAN SARAN

Pada optimasi *Naïve Bayes Classifier* dengan menggunakan *Particle Swarm Optimization* pada data iris. Klasifikasi dilakukan dengan menentukan bobot atribut optimum dengan menggunakan *Particle Swarm Optimization*. Hasil klasifikasi diperoleh dari *fitness* tertinggi.

Dalam implementasi ini, dilakukan 2 pengujian yaitu pengujian jumlah partikel dan pengujian kombinasi parameter. Pada pengujian jumlah partikel sebanyak 10 hingga 50 dengan percobaan sebanyak 3 kali didapatkan rata-rata *fitness* tertinggi sebanyak 97.39 pada partikel 50. Sedangkan, Pada pengujian kombinasi parameter, nilai kombinasi di bangkitkan secara *random* dengan percobaan sebanyak 3 kali didapatkan rata-rata *fitness* tertinggi sebanyak 97.39 pada kombinasi parameter C1 dengan nilai 0.9 dan C2 dengan nilai 0.3. Adapun saran dalam penelitian ini yaitu metode yang digunakan dapat dilanjutkan dengan metode lain atau mengganti metode optimasi lain untuk menghasilkan klasifikasi yang optimal.

6. DAFTAR PUSTAKA

BRAMER, MAX. 2007. *Principles of Data Mining*. Springer, London.

HAMZAH, A. 2012. Klasifikasi Teks dengan Naïve Bayes Classifier (NBC) untuk Pengelompokan Teks Berita dan Abstrak Akademik. *Proceedings Seminar Nasional Aplikasi Sains & Teknologi (SNAST) Periode III*. 3 Novermber, Yogyakarta, Indonesia.

HAND, DAVID J. & YU, KEMING. 2001. Idiot's Bayes: Not So Stupid after All?. *International Statistical Review*, 69 (3), 385-398.

LANGLEY & S. SAGE. 1994. Induction of Selective Bayesian Classifier. *Proceeding of The Tenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, US.

LEIDIYANA, H. 2012. Komparasi Algoritma Klasifikasi Data Mining dalam Penentuan Resiko Kredit Kepemilikan Kendaraan Bermotor. *Tesis Magister Ilmu Komputer*. Sekolah Tinggi Managemen Informatika dan Komputer Nusa Mandiri. Jakarta

LEWIS, D. 1998. Naïve Bayes at forty: The independence assumption in information retrieval. *Proceedings of the Tenth European Conference on Machine Learning*. April, Berlin, Germany. 4-15.

U.S FISH AND WILDLIFE SERVICE. 2013. *Definition of Terms and Phrases.* February 8, 2013. http://www.fws.gov/stand/devterms.html, diakses tanggal 1 Desember 2016.

WU, XINDONG & KUMAR, VIPIN. 2009. *The Top Ten Algorithms in Data Mining*. CRC Press, Boca Raton.