BAB III

LANDASAN TEORI

3.1 *Data, Informasi, Pengetahuan*

Data adalah bilangan, terkait dengan angka angka atau atribut atribut yang bersifat kuantitas, yang berasal dari hasil observasi, eksperimen, atau kalkulasi. Informasi adalah data di dalam satu konteks tertentu. Informasi merupakan kumpulan data dan terkait dengan penjelasan, interpretasi, dan berhubungan dengan materi lainnya mengenai objek, peristiwa peristiwa atau proses tertentu. Sementara itu, pengetahuan adalah informasi yang telah diorganisasi, disintesisksn, diringkaskan untuk meningkatkan pengertian, kesadaran atau pemahaman (Bergeron,2003).

3.2 *Data Mining*

Data mining adalah proses klasifikasi otomatis kasus berdasarkan pola data yang diperoleh dari dataset. Sejumlah algoritma telah dikembangkan dan diimplementasikan untuk mengekstrak informasi dan menemukan pola pengetahuan yang mungkin berguna untuk mendukung keputusan. Data mining juga dikenal sebagai KDD *(knowledge discovery in databases)* (Singhal & Jena, 2013). Berikut adalah tahapan dalam data mining (Mabrur & Lubis, 2012):

1. **Pemilihan (Data Selection)**

   pemilihan data dari sekumpulan data operasional perlu dilakukan sebelum tahap penggalian informasi dalam KDD dimulai.

2. **Pemrosesan awal *(preprocessing)***

   Sebelum proses *data mining* dapat dilaksanakan, perlu dilakukan proses *cleaning* dengan tujuan untuk membuang duplikasi data, memeriksa data yang inkonsisten, dan memperbaiki kesalahan pada data, seperti kesalahan cetak (tipografi). Juga dilakukan proses *enrichment*, yaitu proses "memperkaya" data yang sudah ada dengan data atau informasi lain yang relevan dan diperlukan untuk KDD, seperti data atau informasi eksternal.

3. **Transformation**

   proses *coding* pada data yang telah dipilih, sehingga data tersebut sesuai untuk proses *data mining*. Proses *coding* dalam KDD merupakan proses kreatif dan sangat tergantung pada jenis atau pola informasi yang akan dicari dalam database.

4. ***Data Mining***

   proses mencari pola atau informasi menarik dalam data terpilih dengan menggunakan teknik atau metode tertentu.

5. ***Interpretation/Evaluation***

   pola informasi yang dihasilkan dari proses *data mining* perlu ditampilkan dalam bentuk yang mudah dimengerti oleh pihak yang berkepentingan. Tahap ini merupakan bagian dari proses KDD yang disebut dengan *interpretation*. Tahap ini mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesa yang ada sebelumnya atau tidak.

3.3 Preprocessing Data

Data yang belum diproses disebut data mentah, dimana data harus disiapkan terlebih dahulu sebelum dapat dipakai dalam suatu proses. Data mentah atau data real cenderung mengandung kesalahan atau mengandung nilai-nilai yang menyimpang dari yang diharapkan (Kumar & Chadha, 2012). Data yang mengandung kesalahan dikarenakan data tersebut tidak lengkap ataupun tidak konsisten. Ketidaklengkapan data terjadi karena adanya atribut data yang tidak tersedia, hilangnya nilai untuk beberapa data (atribut) karena adanya penghapusan data yang dianggap tidak penting. Sedangkan data dianggap tidak konsisten karena pada saat pengumpulan data adanya instrumen yang rusak karena kesalahan manusia(human error) ataupun kesalahan komputer, adanya ketidaksamaan (tidak konsisten) dalam penamaan suatu data dengan data yang lain, yang merupakan suatu data yang sama (Han & Kamber, 2006).

Preprocessing data merupakan langkah penting dalam proses penemuan pengetahuan, karena keputusan-keputusan yang berkualitas harus didasarkan pada data yang berkualitas (Kumar & Chadha, 2012). *Preprocessing data* sering kali digunakan untuk mengurangi kesalahan data dan sistematis bias dalam data mentah sebelum analisis apapun terjadi (Tong et al., 2011). Tugas utama dari *preprocessing data*, antara lain :

1. **Pembersihan Data**

   Pembersihan data dilakukan dengan mengisi nilai yang hilang, mengidentifikasi atau menghapus data yang salah dan menyelesaikan ketidaksamaan atau inkonsistensi data.

2. **Integrasi Data**

   Integrasi data adalah penggabungan data dari berbagai sumber penyimpanan data untuk menjadi suatu kesatuan data yang koheren.

3. **Transformasi Data**

   Transformasi data dilakukan dengan proses normalisasi.

4. **Reduksi Data**

   Reduksi data merupakan perolehan representasi penurunan volume tetapi menghasilkan hasil analisis yang sama atau mirip.

5. **Diskritisasi Data**

   Diskritisasi data merupakan bagian dari reduksi data, tetapi dengan kepentingan tertentu, terutama untuk data *numeric* (Han & Kamber,2006).

3.4 Noisy Data Filtering

Noise merupakan kesalahan acak dalam nilai atribut. Dalam dataset yang sangat besar, *noise* dapat datang dalam berbagai bentuk (J.Roiger & W.Geatz, 2003). Berikut beberapa cara menangani *noise* data dengan teknik *smoothing* :

1. **Metode *Binning***

   Pertama mengurutkan data, kemudian mempartisi kedalam (*equidepth*) bin dan selanjutnya dapat di *smoothing* dengan bin means, bin median dan bin boundaries.

2. **Metode *Clustering***

   Dengan metode ini outlier dapat dideteksi oleh *clustering*, dimana nilai-nilai yang sama akan

disusun dalam kelompok-kelompok, dan nilai diluar himpunan cluster dipertimbangkan.

3. **Metode Kombinasi komputer dan inspeksi manusia**
   Dengan metode ini komputer mendeteksi nilai yang mencurigakan, yang kemudian diperiksa oleh manusia.
4. **Metode Regression**
   Dengan metode ini data dapat diperhalus dengan pas data ke fungsi, seperti dengan regresi. regresi linear melibatkan menemukan jalur terbaik untuk menyesuaikan dua variabel, sehingga satu variabel dapat digunakan untuk memprediksi lainnya (Han & Kamber, 2001).

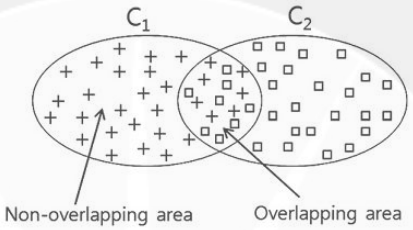
3.5 Kompleksitas Data

Karakteristik dataset pada penelitian ini menjadi penting karena secara signifikan akan mempengaruhi kinerja algoritma. Berkenaan dengan konsep tumpang tindih / "*overlap concept*", maka akan dianalisis kompleksitas data untuk menentukan hubungan antara kompleksitas dataset dan kinerja algorirma klasifikasi. *Overlap* sebagai pengukuran kompleksitas terbatas pada dua aspek penting yang cukup mewakili gagasan *overlap* dalam kumpulan data. Dua buah pengukuran *overlap* yang dipilih adalah *volume of overlap region* (F2) dan *overlap feature efficiency*(F3).

volume of overlap region (f2) dari kelas dianggap berdasarkan maksimum dan minimun nilai dari setiap atribut kelas. Perhitungan dapat berbeda menggunakan setiap fitur, maksimum nilai dan minimum untuk masing-masing kelas seperti pada persamaan (3.1). Overlap

sebagai pengukuran kompleksitas, ditemukan karena merupakan hasil dari berbagai faktor termasuk kesalahan empiris pengukuran, metode komputasi yang tidak memadai, dan ambiguitas kognitif.

$$F_2 = \prod_i \frac{MIN\left(max\left(f_i, c_1\right), max\left(f_i, c_2\right)\right)}{MAX\left(max\left(f_i, c_1\right), max\left(f_i, c_2\right)\right)} \frac{MAX\left(min\left(f_i, c_1\right), min\left(f_i, c_2\right)\right)}{MIN\left(min\left(f_i, c_1\right), min\left(f_i, c_2\right)\right)} \quad (3.1)$$



Gambar 3.1 *Non-overlapping and Overlapping area of cluster c_1 and c_2*

Kompleksitas dataset pada penelitian ini terkait dengan masalah perbedaan. Kompleksitas dapat didefinisikan sebagai kesulitan algoritma klasifikasi untuk menentukan batas keputusan. Kinerja dari algoritma klasifikasi dipengaruhi oleh kompleksitas dataset. Struktur kelas juga bisa menjadi karakteristik penting bagi masalah perbedaan. Selain itu dapat merepresentasikan kompleksitas dataset karena sesuai dengan tingkat algoritma klasifikasi (Ho & Basu, 2002). Sehubungan dengan penelitian ini maka dipilih 5 buah dataset yang digunakan dalam penelitian, sesuai dengan tujuan penelitian kompleksitas dataset dipilih dalam berbagai tingkatan yaitu dari rendah ke tinggi.

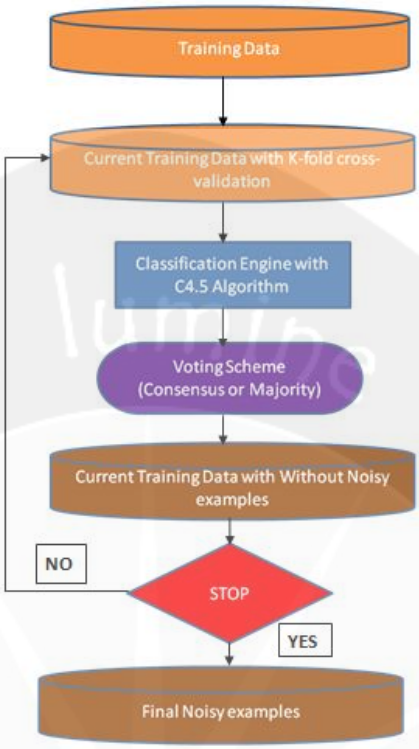
Tabel 3.1. Karakteristik dataset dan Pengukuran Kompleksitas untuk Pengujian Overlap Clustering (Ho, 2006)

Dataset	#Attributes	#Clusters	#Data	F2	F3
Wine	13	3	178	0.001	0.564
Iris	4	3	150	0.114	0.500
Wisconsin	9	2	683	0.217	0.350
Pima	8	2	768	0.251	0.651
Haberman	3	2	306	0.718	0.029

Pada tabel 3.1 diketahui bahwa nilai semakin besar nilai *volume of overlap region* (F2) maka kompleksitas data semakin besar, hal ini berbanding terbalik dengan nilai *overlap feature efficiency*(F3), yaitu semakin kecil nilai *overlap feature efficiency*(F3), maka kompleksitas data semakin besar.

3.6 Iterative Partitioning Filter

Iterative partitioning filter adalah salah satu algoritma *Noisy Data Filtering* pada *preprocesing data*. *Iterative partitioning filter* mengahapus *noisy examples* di beberapa iterasi. Dalam setiap iterasi data pelatihan dibagi menjadi n bagian, dan algoritma C4.5 dibangun disetiap subset ini untuk mengevaluasi semua *examples*. Kemudian semua contoh kesalahan klasifikasi dihapus (menggunakan skema majority atau consensus) dan iterasi baru dimulai (Saez et al., 2016). Pada gambar 2.1 menunjukkan diagram algoritma *iterative partitioning filter*.



Gambar 3.2 Diagram Algoritma Iterative Partitioning Filter

Pada gambar 2.1 diketahui bahwa *Iterative Partitioning Filter* menggunakan pohon keputusan C4.5 pada *classification engine*, karena algoritma ini bekerja dengan baik sebagai filter untuk *noisy* data dan umumnya menghasilkan hasil yang baik pada berbagai dataset yang besar (Quinlan, 1993).

Algoritma C4.5 merupakan kelompok algoritma *Decision Tree*. Algoritma ini mempunyai input berupa *training samples* dan *samples* berupa data contoh yang akan digunakan untuk membangun sebuah *tree* yang telah diuji kebenarannya. *Decision tree* itu sendiri adalah *flow-chart* seperti struktur *tree*, dimana tiap internal *node* menunjukkan sebuah *test* pada sebuah atribut, tiap cabang menunjukkan hasil dari *test* dan *leaf node*

menunjukkan *class-class* atau *class distribution* (Br Ginting et al., 2014). Tahapan Algoritma Decision Tree C4.5 :

1. Menyiapkan data training
2. Menentukan akar dari pohon
3. Menghitung nilai Gain :

$$Entropy(S) = \sum_{i=1}^{n} -pi * log_2 pi \quad \ldots\ldots\ldots (1)$$

4. Ulangi langkah ke-2 hingga semua tupel terpartisi

$$Gain(S, A) = S - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * S_i \quad \ldots\ldots\ldots (2)$$

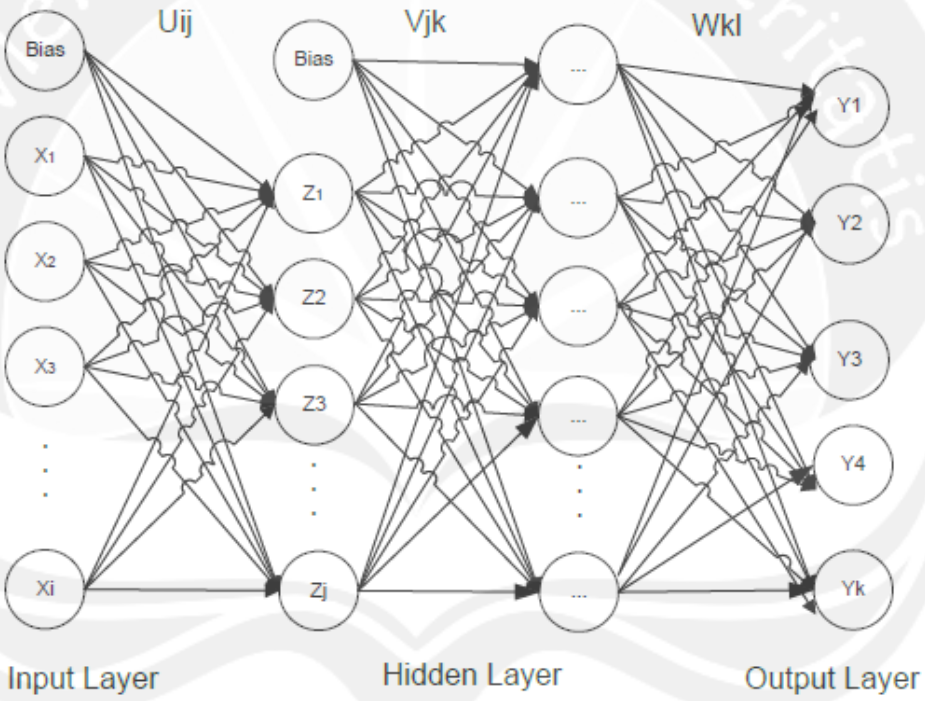
Proses partisi pohon keputusan akan berhenti saat semua tupel dalam node N mendapat kelas yang sama dan atau tidak ada atribut di dalam tupel yang dipartisi lagi dan atau tidak ada tupel didalam cabang yang kosong (Swastina, 2013).

Selanjutnya pada *Iterative Partitioning Filter* terdapat dua buah skema untuk penyaringan noise, bila menggunakan skema majority maka menghilangkan sebuah *instance* bila kesalahan klasifikasi lebih dari 50 % dari pengklasifikasi, sedangkan bila menggunakan skema consensus maka menghilangkan *noisy example* jika kesalahan klasifikasi oleh semua pengklasifikasi. Proses iterasi berakhir setelah kriteria tercapai, sebagaimana yang telah didefinisikan (Zhu et al., August 2003).

3.7 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan adalah sistem komputasi dimana arsitektur dan operasi diilhami dari pengetahuan tentang sel saraf biologis didalam otak, yang merupakan salah satu representasi buatan dari otak manusia yang

selalu mencoba menstimulasi proses pembelajaran pada otak manusia tersebut. Jaringan Syaraf Tiruan (JST) dapat digambarkan sebagai model matematis dan komputasi untuk fungsi aproksimasi non-linear, klasifikasi data cluster dan regresi non-parametrik atau sebuah stimulasi dari koleksi model saraf biologi. Arsitektur yang digunakan untuk pengenalan pola adalah arsitektur pada Multi Layer Perceptron (MLP) seperti pada gambar 2.2



Gambar 3.3 Arsitektur Multi Layer Perceptron (Fahlman & Hinton, 1987)

Arsitektur MLP terdiri dari *input layer* (X_i), *hidden layer* (Z_j), dan output layer (Y_k). Koneksi antar *layer* dihubungkan dengan bobot U_{ij} merupakan bobot dari *input layer* (X_i) ke *hidden layer*(Z_j). *Wkl* merupakan bobot dari *hidden layer*(Z_j) ke *output* layer (Y_k) (Haryati et al., 2016).

3.8 Algoritma Backpropagation

Backpropagation merupakan pelatihan yang terawasi dengan menggunakan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan *neuron-neuron* yang ada pada lapisan tersembunyi. Algoritma *Backpropagation* merupakan *error* keluaran untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error* ini,tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu (Haryati et al., 2016).

3.9 Knowledge Extraction based on Evolutionary Learning (KEEL)

KEEL (Pengetahuan Ekstraksi berdasarkan Evolusioner Learning) merupakan perangkat lunak java open source (GPLv3) yang dapat digunakan untuk sejumlah pengetahuan yang besar dari tugas penemuan data yang berbeda. KEEL menyediakan GUI sederhana berdasarkan aliran data untuk merancang eksperimen dengan dataset yang berbeda dan algoritma kecerdasan komputasi (menyita perhatian khusus untuk algoritma evolusioner) untuk menilai perilaku algoritma. Ini berisi berbagai algoritma ekstraksi pengetahuan klasik, teknik (training set pilihan, pilihan fitur, diskritisasi, metode imputasi untuk nilai-nilai yang hilang, dan lain-lain), berdasarkan algoritma pembelajaran kecerdasan komputasi, model hybrid, metodologi statistik preprocessing untuk percobaan kontras dan sebagainya. Selain itu, KEEL telah dirancang dengan dua tujuan yaitu penelitian dan pendidikan.

Gambar 3.4 Tampilan awal keel

3.10 Waikato Environment for Knowledge Analysis (WEKA)

Weka merupakan perangkat lunak yang menyediakan layanan untuk melakukan pengolahan data dalam *data mining*. Perangkat lunak ini berbasis *open source* dan dibuat menggunakan Java. WEKA dibuat dan dikembangkan oleh Universitas Waikato di Selandia Baru. Weka merupakan prangkat lunak gratis yang tersedia dibawah *General Public License*. Perangkat ini memiliki fasilitas untuk melakukan *preprocessing data, classification, regression, clustering, association rules, dan visualization.*

Gambar 3.5 Tampilan awal Weka

WEKA memiliki empat jenis *test option* yang dapat digunakan untuk melakukan proses klasifikasi. Proses yang ditangani adalah proses pelatihan dan pengujian. Keempat jenis *test option* tersebut yaitu :

a. *Use training set*

   Klasifikasi ini menggunakan satu data untuk melakukan pelatihan. Lalu dari seluruh data yang telah dilatih sebelumnya juga digunakan untuk proses pengujian.

b. *Supplied test set*

   Klasifikasi ini dilakukan evaluasi dengan cara memprediksi seberapa baik satu dataset yang diambil dari sebuah data tertentu yang memang sudah disediakan untuk pengujian. Proses pelatihan akan dilakukan terlebih dahulu dengan data latih kemudian proses pengujian akan dilakukan dengan data uji yang berbeda dengan data yang dilatih pada klasifikasi.

c. *Cross-validation*

Klasifikasi ini dilakukan evaluasi dengan *cross-validation* dan menggunakan jumlah *fold* yang tertentu yang dapat diinputkan manual. Pada *cross-validation* akan ada pilihan beberapa *fold* yang akan digunakan. Nilai *fold* default aplikasi yang diberikan adalah 10. Proses pengujian akan dilakukan sebanyak nilai fold yang diberikan serta akan dibentuk subset sebanyak nilai *fold*. Kemudian proses pengujian akan dilakukan menggunakan sebuah subset yang terbentuk dan sisanya akan digunakan untuk proses pelatihanya.

d. *Percentage split*

Klasifikasi ini dilakukan evaluasi dengan melakukan pembagian data antara data uji dan data latih pada satu dataset dengan menggunakan prosentase. Prosentase yang diinputkan akan digunakan untuk proses pelatihan dan sisanya akan digunakan untuk proses pengujian. Proses ini biasanya dilakukan untuk dengan perbandingan 2/3 data untuk pelatihan dan 1/3 data untuk proses pengujian atau nilai k = 66%.