khazanah informatika

Jurnal Ilmu Komputer dan Informatika

Evaluasi Performa Pemecahan Database dengan Metode Klasifikasi pada Data Preprocessing Data Mining

Dedi Gunawan[1*]
[1]Program Studi Informatika
Universitas Muhammadiyah Surakarta
[*]dedi.gunawan@ums.ac.id

ABSTRAK

Secara umum data transaksi akan dianalisa menggunakan teknik data mining seperti *classification, clustering*, ataupun *prediction* agar bisa memberikan informasi yang bernilai lebih kepada pemilik data. Analisa data transaksi akan menjadi tidak mudah jika ukuran data yang dimiliki sangat besar sehingga perlu dilakukan data *preprocessing* terlebih dahulu. Data *preprocessing* merupakan proses mempersiapkan data seperti membersihkan data dari *noise* ataupun merubah format data. Salah satu teknik data *preprocessing* untuk mengatasi ukuran database yang besar adalah dengan membagi database menjadi beberapa bagian sehingga akan mempercepat proses *scanning* data saat algoritma data mining diterapkan. Database bisa dipartisi menjadi beberapa bagian berdasarkan kasifikasi jenis item dari transaksi yang dilakukan oleh konsumen ataupun partisi secara otomatis dengan membagi beberapa bagian tanpa melihat item di dalamnya. Dalam penelitian ini kami akan membandingkan hasil kinerja dari kedua jenis model partisi database tersebut. Hasil perbandingan kinerja diukur dari waktu komputasi dan jumlah memori yang terpakai dalam proses partisi database. Berdasarkan hasil pengujian partisi database dengan teknik klasifikasi item membutuhkan waktu yang lebih tinggi yaitu 7000 milidetik dari pada partisi secara otomatis dengan waktu 500 milidetik dengan jumlah data transaksi 10.000. Sedangkan penggunaan memori komputer yang diperlukan adalah 0.35 MB untuk partisi otomatis dan 0.013 MB untuk partisi dengan klasifikasi item.

Kata Kunci : Partisi database, database transaksi, data mining, data preprocessing.

1. Pendahuluan

Database adalah sebuah kumpulan informasi yang terstruktur. Sebuah database terdiri dari kumpulan file dalam sistem komputer [1]. Database biasanya digunakan oleh suatu organisasi untuk menyimpan informasi yang berkaitan dengan bisnis proses dari organisasi tersebut seperti, penggajian karyawan, menejemen pelanggan dan inventarisasi. Selain itu, data yang tersimpan di dalam database tidak akan memberikan manfaat banyak bagi pemilik data apabila tidak dilakukan usaha data untuk mencari informasi tersembunyi yang lebih berharga di dalam database. Salah satu langkah yang bisa dilakukan untuk memperoleh nilai lebih dari data adalah dengan menerapkan teknik data mining. Data mining merupakan serangkaian proses yang terdiri dari persiapan data, penerapan algoritma data mining dan menampilkan hasil proses secara visual [2]. Proses pertama kali yang harus dilakukan adalah menyiapkan data untuk proses data mining.

Adapun tantangan yang dihadapi oleh pemilik data untuk mempersiapkan data sebelum proses data mining adalah ukuran data yang besar dan tentunya memerlukan waktu yang lama untuk proses analisa mengingat performa database akan menurun seiring dengan pertumbuhan datanya [7]. Data yang dikoleksi dalam jangka waktu yang lama tentu ukurannya akan menjadi sangat besar dan membutuhkan proses yang tidak sederhana untuk mengolahnya. Sehingga untuk memudahkan proses pengolahan data yang sangat besar tersebut diperlukan teknik pemecahan database yang bertujuan agar proses *scanning* data saat implementasi algoritma data mining bisa dilakukan dengan cepat. Banyak cara yang bisa dilakukan untuk memecah database diantaranya adalah database bisa dipartisi berdasarkan jumlah partisi yang sesuai dengan keinginan pemilik data, ataupun dipecah berdasarkan pengelompokan item dari transaksi yang dilakukan oleh konsumen. Dalam tulisan ini penulis hanya menganalisa serta membandingkan waktu komputasi dan besarnya *resource* yang digunakan oleh dua metode di atas yang merupakan langkah awal dari proses data mining sebelum algoritma data mining yang sesungguhnya diterapkan. Dengan mengetahui perbandingan performa pemecahan database dengan kedua teknik diatas kita bisa menentukan teknik mana yang paling sesuai dengan kondisi yang dialami oleh pemilik data khususnya untuk data *preprocessing*.

2. Tinjauan Pustaka

2.1. Partisi Database

Tujuan partisi database adalah untuk mempermudah dalam proses pengolahan data. Mengingat pentingnya

proses ini maka banyak studi dilakukan untuk mendukung proses partisi data base [6]. Secara sederhana keuntungan memecah database adalah data akan terbagi menjadi beberapa bagian yang ukuran file lebih kecil sehingga jika pemrosesan dilakukan secara paralel dengan menggunakan sistem terdistribusi melalui jaringan komputer maka jaringan tersebut tidak akan terbebani ketika proses transfer data. Terlebih lagi, data yang berukuran kecil tentunya akan lebih cepat dibaca oleh komputer dari pada data yang berukuran besar. Secara umum terdapat dua model partisi yang digunakan untuk memecah database yaitu partisi database secara horizontal dan partisi database secara vertikal.

2.2. Partisi Horizontal

Partisi horizontal merupakan format database yang berupa {*TID : item*}. *TID* merupakan identitas dari sebuah transaksi yang bersifat *unique* dan item merupakan kumpulan item atau itemset yang dimiliki oleh *TID* tertentu dalam setiap transaksi [3]

Tabel. 1. *Horizontal Database*

TID	Item
1	a b c d f g h
2	a b d e
3	b c d f g h
4	a b c f h
5	c d e i
6	a c f i
7	b c f g
8	c d f h i
9	a f i
10	a c e f h

2.3 Partisi Vertikal

Partisi vertikal database merupakan kebalikan dari partisi horizontal dimana format yang digunakan adalah {*item : TID_set*}. Item merupakan kode item atau nama item yang dimiliki oleh *TID*. Sedangkan *TID_set* merupakan kumpulan dari identitas transaksi yang memiliki kesamaan item.

Tabel. 2. *Vertical Database*

Item	TID_set
a	{T1, T4, T5, T7, T8, T9}
b	{T1, T4, T5}
c	{T3, T7, T9,T10}
d	{T1, T3, T5, T6, T8, T10}
e	{T1, T4}
f	{ T5, T6, T8}
g	{ T7, T8, T9}
h	{ T8, T10}
i	{T7, T9,T10}
j	{ T3, T5}

Penelitian [4] menyebutkan bahwa ada persoalan yang timbul ketika data transaksi dipartisi secara vertikal ketika algoritma data mining seperti *Association Rule* diterapkan untuk menganalisa data transaksi tersebut. Hal ini terjadi karena transaksi tersebar dibeberapa lokasi yang berbeda, di mana setiap lokasi mempunyai atribut yang berbeda-beda.

3. Metode Penelitian

Seperti yang disebutkan pada bagian awal bahwa penelitian ini hanya fokus pada data *proprocessing* yaitu partisi database. Dalam penelitian ini kami mengguanakan model pemecahan database secara horizontal di mana database transaki berasal dari IBM Synthetic data generator [5], karena database akan didistribusikan ke beberapa server secara paralel ketika akan dilakukan proses mining dengan algoritma data mining seperti *association rule* dan *market basket analysis*. Ilustrasi dari pemecahan database secara horizontal adalah sebagai berikut, misalkan kita mempunyai sebuah database **DB** dan akan dipecah untuk dikirim ke beberapa **n** server, maka $\mathbf{DB} = \mathbf{DB}_1 \cup \mathbf{DB}_2 \cup \mathbf{DB}_3 \cup ... \cup \mathbf{DBn}$ sehingga $\mathbf{DB}_i$ akan dikirim ke server *Si* dimana $(1 \leq i \leq n)$.

3.1. Partisi Database Secara Otomatis

Database bisa dipartisi sesuai dengan keinginan pemilik data, sehingga pemilik data memiliki kebebasan dalam menentukan berapa jumlah partisi yang diinginkan tanpa memperhatikan item pada masing-masing transaksi.

1) Algoritma Partisi Database Otomatis

Langkah pertama dari algoritma partisi otomatis adalah user memasukkan database *D* yang akan dipartisi, kemudian menginputkan jumlah partisi *n* yang diinginkan oleh pemilik data. Selanjutnya database akan dibaca untuk menentukan berapa jumlah transaksi yang terdapat di dalamnya $|D|$. Langkah berikutnya adalah menentukan jumlah transaksi yang akan disimpan pada setiap partisi $|D_n|$. Proses ini dihitung berdasarkan *modulo operation* dimana jumlah total transaksi keseluruhan dibagi dengan jumlah partisi yang diinputkan oleh user. Jika sisa hasil bagi antara jumlah transaksi dengan jumlah partisi adalah nol, maka setiap partisi akan mendapatkan jumlah transaksi yang sama. Akan tetapi apabila setelah perhitungan tersebut terdapat sisa hasil bagi maka sisa tersebut akan dimasukkan ke partisi yang pertama sehingga partisi pertama akan memiliki jumlah transaki yang sedikit lebih besar dari partisi yang lainnya. Langkah paling akhir adalah setiap partisi disimpan ke dalam buffer dan dituliskan kedalam dataset. Dataset hasil partisi inilah yang nantinya akan diproses oleh algoritma data mining. Sebagai contoh, misalkan di dalam suatu database terdapat 1000 transaksi, dan pemilik data menginginkan untuk memecah database menjadi tiga bagian, maka 1000 mod 3 menghasilkan 333 transaksi dengan 1 sisa transaksi, sehingga setiap partisi akan memiliki 333 transaksi dan partisi pertama akan ditambahkan dengan sisa transaksi, hasilnya adalah partisi pertama memiliki 334 transaksi, partisi kedua dan ketiga masing-masing memiliki 333 transaksi.

input: *D, n*
output: D_n
1. Masukkan dataset *D*
2. Masukkan *n*
3. for *i* <= *D i*++
4. |*D*| = i
5. menentukan jumlah transaksi untuk setiap partisi
   a. x = |D| % n
   b. |*Dn*| = (|D| / n) + x
6. for each *n*
   a. simpan *Dn* ke dalam buffer
   b. simpan buffer ke dalam file dataset

Gambar 1. Algoritma partisi otomatis

3.2. Partisi Database dengan Klasifikasi

Teknik klasifikasi menggunakan pendekatan sistematis untuk mengelompokan data berdasarkan kemiripan tertentu. Teknik ini sudah banyak digunakan untuk beberapa aplikasi seperti, pengelompokan dokumen berdasarakan kata kunci tertentu, mengelompokkan konsumen berdasarkan tempat tinggalnya, mengkategorikan berita, dan masih banyak aplikasi lain yang bisa memanfaatkan teknik klasifikasi. Dengan menggunakan algoritma ini data akan dikelompokkan menjadi beberapa kelas berdasarkan item yang dikehendaki oleh pemilik data.

Secara sederhana algoritma pengklasifikasian item akan melakukan proses *scanning* terhadap semua transaki yang ada di dalam database. Selanjutnya transaksi ini akan dikelompokkan menjadi beberapa kelas sesuai dengan kemiripan item yang ada pada setiap transaksi. Jika sebuah transaksi terdapat item yang bisa dianggap sebagai item kunci maka transaksi tersebut akan disimpan kedalam kelas yang sama. Item kunci ini merupakan item utama yang digunakan untuk mengelompokkan transaksi berdasarakan ada atau tidaknya item tersebut di dalam suatu data transaksi. Misalkan, pemilik data berkeinginan mengklasifikasikan transaksi berdasarkan ada atau tidaknya itemset 'a b' dan 'b c' pada sebuah transaksi, maka setiap transaksi yang mengandung itemset 'a b' akan dimasukkan ke dalam kelas tertentu yang berbeda dengan transaksi yang memiliki itemset 'b c'. Sehingga hasil akhir dari partisi dataset adalah dataset yang memiliki kesamaan itemset pada setiap transaksinya.

Table 3. Parameter Testing

Parameter	Value	Range
Jumlah partisi	3	2-5
Jumlah itemset kunci	3	1-3
Jumlah item rata-rata per transaksi	6	1-10
Jumlah transaksi	-	10K-50K

Baris pertama dan kedua pada algoritma di atas merupakan fase inisialisasi, dimana database dimasukkan bersamaan dengan item yang yang merupakan kata kunci oleh pemilik data. Selanjutnya pada baris ketiga setiap transaki akan dianalisa apakah di dalamnya terdapat item kunci atau tidak. Kemudian, jika sebuah transaksi mengandung item kunci maka akan dimasukkan ke partisi database yang berisi transaksi yang sama. Pada langkah selanjutnya setiap transaksi akan dimasukkan ke dalam kelas yang sesuai dengan item transaksi. Setelah itu data pada masing-masing kelas akan disimpan kedalam buffer. Kemudian pada langkah terakhir, semua kelas yang ada di buffer akan dituliskan ke dalam file partisi database.

4. Analisa Hasil Pengujian

Database yang dipakai untuk melakukan pengujian algoritma merupakan database hasil dari IBM database generator [5], sedangkan algoritma diimplementasikan menggunakan bahasa pemrograman java. Pengujian database dilakukan dengan menggunakan komputer dengan sistem operasi windows 7, RAM 2 Gb, hard disk 500 Gb.

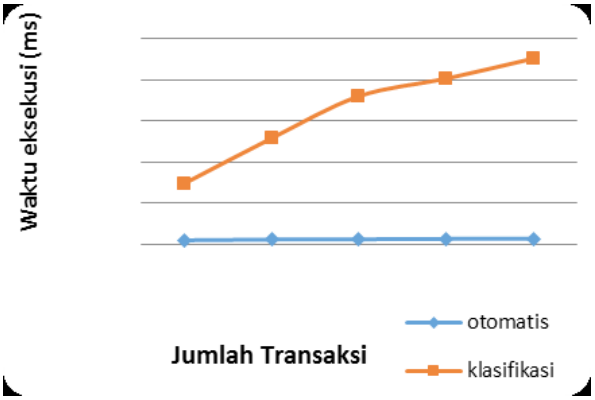
Tabel 3 berisi parameter testing yang merupakan nilai untuk variabel testing, tujuan dari ditentukannya parameter testing adalah agar proses testing lebih spesifik.

input: D, i_set
output: partisi database
1. Membaca database D
2. Masukkan item kunci i_set
3. Menentukan kelas transaksi Ti
4. if i_set terdapat pada transaksi Ti
5. Tambahkan Ti ke kelas Ci
6. for each kelas Ci
7. simpan Ci ke dalam buffer
8. simpan Ci ke dalam file

Gambar 2. Algoritma partisi data dengan klasifikasi item

4.1. Waktu Komputasi

Waktu komputasi digunakan untuk menghitung berapa lama sebuah perintah atau algoritma diselesaikan oleh komputer.

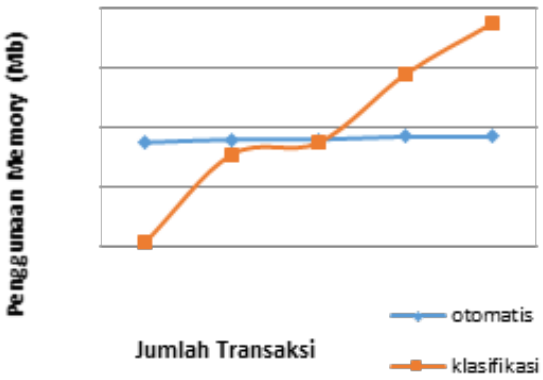


Gambar. 3. Grafik waktu komputasi

Hasil komputasi pada gambar.2 menunjukkan bahwa waktu yang dibutuhkan untuk melakukan partisi menggunakan teknik klasifikasi item transaksi lebih besar dari pada waktu yang diperlukan untuk melakukan partisi otomatis, hal ini disebabkan karena pada algoritma klasifikasi berdasarkan item transaksi harus membaca dan menentukan terlebih dahulu apakah item kunci terdapat pada transaksi tertentu atau tidak. Sedangkan untuk algoritma partisi otomatis, transaksi tidak dianalisa, melainkan algoritma tersebut langsung membagi database berdasarkan jumlah yang dikehendaki oleh pemilik data tanpa memperdulikan item yang ada pada masing-masing transaksi

4.2. Penggunaan Memori Komputer

Memori yang digunakan pada masing-masing algoritma memberikan gambaran bahwa untuk melakukan partisi database setiap algoritma membutuhkan jumlah memori tertentu. Grafik pada gambar.3 menunjukkan perbandingan jumlah memory yang terpakai pada masing-masing algoritma. Memori yang terpakai pada algoritma klasifikasi berdasarkan item transaksi mengalami peningkatan sesuai dengan jumlah transaksi yang ada dalam database. Hal ini terjadi karena semakin banyak transaksi maka semakin banyak juga transaksi yang harus diklasifikasikan sesuai dengan itemnya. Sedangkan untuk partisi secara otomatis jumlah memory yang terpakai relatif tidak terpengaruh oleh jumlah transaksi.



Gambar. 4. Grafik pengguanaan memory

5. Kesimpulan

Hasil analisa dari grafik pada gambar 3 dan 4 menunjukan bahwa, algoritma partisi otomatis memberikan kebebasan kepada pemilik data untuk mempartisi database sesuai dengan keinginan mereka, selain itu waktu yang dibutuhkan untuk melakukan partisi lebih rendah dibandingkan dengan algoritma partisi berdasarkan klasifikasi item transaksi. Penggunaan *memory* pada partisi berdasarkan algoritma partisi otomatis relatif stabil dari pada partisi dengan teknik klasifikasi item transaksi. Secara keseluruhan algoritma partisi otomatis memberikan keuntungan dari segi fleksibilitas jumlah *cluster*, waktu komputasi dan penggunaan memory yang relatif lebih rendah. Meskipun sebenarnya algoritma klasifikasi item transaksi bisa lebih unggul ketika algoritma data mining seperti hukum asosiasi dan frequent itemset diterapkan mengingat item-item transaksi sudah terkelompokkan ke dalam satu *cluster*.

6. Persantunan

Hasil penelitian yang dipublikasi ini merupakan sebagian dari thesis penulis tingkat master pada Department computer science and information engineering, National Dong Hwa University.

7. Daftar Pustaka

[1] Powel, Gavin. Beginning Database Design, Wiley Publishing, Inc. 2006

[2] Bestein, A., Provost, F., Hill, S., Toward intelligent assistance for a data mining process: an ontology-based approach for cost-sensitive classification, Knowledge and Data Engineering Volume:17 503 - 518 April 2005

[3] Han, Jiawey., Kamber, M. Data Mining Concept and Technique . Elsevier publishing. 2006.

[4] Kantarcioglu, Murat., Clifton, C,. Privacy-preserving Distributed Mining of Association Rules on Horizontally Partitioned Data . IEEE transaction on knowledge and data engineering. 2003

[5] IBM Synthetics data generator, http://www.philippe-fournier-viger.com/spmf/datasets/

[6] Chauhan, Sonam S, Deskmukh P R., Literature Review on Information Extraction by Partitioning. International Journal of Computer Science and Mobile Computing. Vol 2. 2013.

[7] L. Hong, M. Lu, W. Hong. A Business Computing System Optimization Research on the Efficiency of Database Queries. International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery. 2013.