BAB 6

KESIMPULAN DAN SARAN

6.1 Kesimpulan

Pada skripsi ini telah berhasil dibangun perangkat lunak yang terdiri dari beberapa modul yang menerapkan algoritma klasifikasi *naive bayes* berbasis *MapReduce* untuk dapat berjalan pada sistem terdistribusi Hadoop. Terdapat 2 modul yang dibuat untuk pembuatan model klasifikasi *naive bayes* dan melakukan evaluasi terhadap model *naive bayes* yang sudah dibuat. Sedangkan 2 modul lainnya dibuat dengan berbasiskan web untuk melakukan pengelolaan input file dalam HDFS dan klasifikasi untuk tiap kasus yang di-input-kan oleh user secara manual.

Berdasarkan hasil pengujian perangkat lunak dan eksperimen yang telah dilakukan pada Bab 5, dapat disimpulkan bahwa :

- Penggunaan Maven sebagai *dependency management* membantu memudahkan pengambilan *library - library* yang dibutuhkan dari repository *central* milik Maven pada perangkat lunak yang dibangun.
- Penggunaan Thymeleaf sebagai *server side - templating engine* memudahkan perangkat lunak dalam mengirimkan HTML beserta atribut - atribut yang dibutuhkan (sebelumnya diproses pada server) kepada *client* (*browser*).
- Perangkat lunak untuk menerapkan algoritma klasifikasi *naive bayes* yang dibuat berbasiskan *MapReduce* pada lingkungan terdistribusi Hadoop telah diuji kebenaran-nya dengan melakukan perbandingan terhadap perhitungan manual pada bagian 5.4 dan menghasilkan nilai yang cukup sama. Perbedaan hanya terjadi pada pembulatan bilangan desimal pada angka dibelakang koma.
- Algoritma klasifikasi *naive bayes* pada sistem terdistribusi Hadoop dapat menangani data yang termasuk ke dalam golongan *big data*.
- Eksperimen yang telah dilakukan untuk menguji efisiensi dan kecepatan perangkat lunak pada variabel - variabel yang berubah. Berikut ini adalah variabel-variabel beserta kesimpulan hasil eksperimen :

  (a) Ukuran blok HDFS
  Pada data berukuran 1GB, Semakin besar ukuran blok HDFS untuk file input maka waktu yang dihabiskan untuk menjalankan proses perangkat lunak berbasis *MapReduce* semakin sedikit. Tetapi, perlu diperhatikan bahwa pemilihan ukuran blok file input akan sangat berpengaruh terhadap besaran ukuran file input itu sendiri. Karena, jika ukuran blok HDFS lebih besar atau sama dengan ukuran file itu sendiri, maka file tidak akan didistribusikan kepada tiap *datanode*.

  (b) Ukuran data
  Ukuran data mempengaruhi waktu eksekusi perangkat lunak. Meskipun perbedaan waktu terjadi mungkin karena performa komputer yang jika mengolah data lebih banyak akan memanas dan sedikit melambat.

(c) Jumlah Atribut
Jumlah atribut mempengaruhi waktu eksekusi perangkat lunak. Hal tersebut dikarenakan algoritma klasifikasi *naive bayes* melakukan perhitungan jumlah frekuensi untuk tiap nilai atribut prediktor terhadap tiap nilai atribut kelas.

(d) Tipe Atribut Prediktor
Perbedaan waktu eksekusi data yg memiliki atribut numerik dibandingkan hanya memiliki atribut diskret memiliki perbedaan yang sangat signifikan. Untuk data yang memiliki atribut numerik memiliki waktu eksekusi yang jauh lebih lama dibandingkan jika seluruh datanya hanya berisi atribut diskrit. Hal tersebut disebabkan karena perhitungan komputasi yang melibatkan atribut numerik memiliki kompleksitas `O(2n)` yang menyebabkan waktu eksekusi akan 2 kali lipat lebih lama daripada atribut diskrit dan melakukan penulisan ke dalam HDFS (untuk output) untuk setiap iterasi.Perbedaan ukuran data yang dapat ditangani oleh perangkat lunak dengan spesifikasi yang sudah dijelaskan jika semua atribut berupa diskret dengan yang memiliki atribut numerik terlihat sangat jauh berbeda. Untuk data yang memiliki atribut numerik hanya mampu memproses hingga ukurannya mencapai 1,11GB saja, selebihnya akan terkena limit pada memori *node* pekerja yang terbatas. Hal tersebut terjadi karena untuk atribut numerik akan melakukan penyimpanan objek `double` sebanyak 2 kali lipat dari data asli, yang membutuhkan memori sebesar 8 bytes untuk setiap objeknya. Sehingga menyebabkan limit memori yang diperlukan melebihi kapasitas memori yang dimiliki oleh *node* pekerja.

Berdasarkan hasil eksperimen juga didapat bahwa perangkat lunak yang dibangun memiliki waktu eksekusi cukup cepat untuk menangani perhitungan dengan atribut prediktor bertipe diskrit. Tetapi, untuk atribut prediktor bertipe numerik, perangkat lunak yang dibangun belum memiliki waktu eksekusi yang cukup cepat, dikarenakan hanya mampu menjalankan program dengan data yang maksimal berukuran sebesar 1.11GB saja pada spesifikasi perangkat keras yang sudah dijelaskan pada Subbab 5.1.

6.2 Saran Penelitian Lanjutan

Hadoop merupakan platform *open source* yang cocok untuk melakukan proses terhadap data yang berukuran besar, memiliki skalabilitas yang baik, dan juga memiliki mekanisme penanganan error(*fault tolerance*) yang bagus dapat memberikan solusi bagi industri untuk mengambil informasi dari *big data* dengan waktu yang cukup singkat. Model dari pemrograman *MapReduce* juga cukup mudah dimengerti dan diimplementasikan, dengan pemahaman yang tidak terlalu mendalam mengenai detil dari desain internal pada Hadoop, orang sudah dapat membuat program *MapReduce*. Dengan adanya Hadoop streaming, saat ini program *MapReduce* sudah dapat dibuat menggunakan bahasa lain seperti dengan skrip Python dsb. Hal ini dapat cukup menghilangkan batasan bahasa pemrograman dalam membuat program berbasis *MapReduce* untuk dapat dieksekusi pada lingkungan Hadoop.

Algoritma klasifkasi *naive bayes* merupakan salah satu algoritma dalam teknik penambangan data dan *machine learning* yang dapat sangat bermanfaat pada era *big data* seperti saat ini. Pada era *big data*, teknik penambangan data dapat digunakan oleh perusahaan besar dalam mengubah data yang sudah tidak berguna menjadi sebuah informasi yang sangat berharga, sehingga dapat membantu perusahaan dalam membuat suatu keputusan.

Penelitian pada skripsi kali ini hanya merupakan sebuah langkah kecil dalam memanfaatkan teknik penambangan data berbasis *MapReduce* pada sistem terdistribusi Hadoop. Penelitian selanjutnya diharapkan dapat melakukan lebih banyak lagi penerapan teknik/algoritma penambangan data dan/atau *machine learning* pada sistem terdistribusi hadoop mengguanakan pemrograman berbasis *MapReduce*. Selain itu, dapat juga dilakukan pemahaman lebih dalam mengenai internal desain dari sistem Hadoop untuk dapat memanfaatkan sumber daya dari *cluster* yang kita miliki

dengan maksimal. Seperti misalnya, menganalisis jumlah *mapper* dan *reducer* yang optimal untuk digunakan pada suatu proses *MapReduce* untuk memaksimalkan penggunaan sumber daya dan meningkatkan efisiensi waktu pada pemrosesan data berukuran besar yang kita miliki.

Adapun penelitian yang dapat dilanjutkan jika mengacu pada penelitian ini adalah sebagai berikut:

- Dapat dilakukan penambahan teknik penanganan untuk atribut bertipe numerik dengan menghitung nilai standard deviasi menggunakan pendekatan aproksimasi, dimana nilai standard deviasi didapat pada iterasi sebelumnya yang mengakibatkan nilai tersebut tidak mutlak kebenarannya tetapi dapat membantu perangkat lunak dalam mempercepat waktu eksekusi.
- Dapat dilakukan penambahan teknik penanganan untuk atribut bertipe numerik dengan memodifikasi alur kerja dari framework *MapReduce* pada *Hadoop* untuk mengulangi fase reduce setelah perhitungan rata - rata didapat pada fase reduce sebelumnya menggunakan `Chain Mapper`[1]
- Dapat dilakukan penambahan untuk menerapkan teknik *predictive regression* pada perangkat lunak yang sudah dibangun untuk melakukan prediksi terhadap atribut numerik.
- Dapat dilakukan pengembangan untuk dapat melakukan eksekusi program berbasis *MapReduce* pada sistem terdistribusi hadoop menggunakan GUI (*Graphical User Interface*) yang telah dibuat untuk modul kelola input dan klasifikasi. Sehingga, akan memudahkan *end-user*(misalnya: pegawai pada perusahaan) untuk menjalankan program berbasis *MapReduce*.
- Dapat dilakukan penambahan untuk menerapkan beberapa teknik penanganan atribut bertipe numerik seperti metode binning dsb.
- Melakukan eksperimen pada cluster yang lebih besar yang dapat mengikutsertakan lebih banyak node dari saat ini.
- Menganalisis variabel - variabel pada Hadoop yang dapat dikelola secara manual untuk mengoptimalkan kinerja dari program *MapReduce* yang berjalan.

[1] *Chain Mapper* merupakan fitur dari *MapReduce* untuk menggunakan lebih dari satu kelas mapper pada satu pekerjaan. Sumber: http://hadoop.apache.org/docs/r2.7.2/api/org/apache/hadoop/mapreduce/lib/chain/ChainMapper.html

DAFTAR REFERENSI

[1] Piateski, G. dan Frawley, W. (1991) *Knowledge Discovery in Databases*. MIT Press, Cambridge, MA, USA.

[2] ZaÃrane, O. R. (2015) Chapter I: Introduction to data mining. https://webdocs.cs.ualberta.ca/~zaiane/courses/cmput695/F07/slides/ch1-695-F07.pdf. [Online; diakses 14-September-2015].

[3] Michael J. A. Berry, G. L. (1997) *Data Mining Techniques. For Marketing, Sales, and Customer Support*. Verlag John Wiley And Sons, Inc.

[4] Zikopoulos, P., Eaton, C., dkk. (2011) *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media.

[5] network, J. (2012) Introduction to big data - infrastructure and networking considerations. http://www.one.com.vn/sites/default/files/file-attached/catalog/introduction_to_big_data_-_infrastructure_and_networking_considerations.pdf. [Online; diakses 09-April-2017].

[6] Lam, C. (2010) *Hadoop in Action*, 1st edition. Manning Publications Co., Greenwich, CT, USA.

[7] Dean, J. dan Ghemawat, S. (2004) Communications of the acm. *MapReduce: simplified data processing on large clusters*, pp. 107–113.

[8] S. Ghemawat, H. G. dan Leung, S. (2003) Proceedings of the nineteenth acm symposium on operating systems principles. *The Google file system*, London, UK.

[9] Holmes, A. (2012) *Hadoop in Practice*. Manning Publications Co., Greenwich, CT, USA.

[10] Garg, B. (2013) Design and development of naive bayes classifier. *Circulation*, **701**.

[11] Ro, D. dan Pe, H. (1973) Pattern classification and scene analysis.

[12] Langley, P., Iba, W., Thompson, K., dkk. (1992) An analysis of bayesian classifiers. *Aaai*, pp. 223–228.

[13] Rish, I. (2001) An empirical study of the naive bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence*, pp. 41–46. IBM New York.

[14] Peugh, J. L. dan Enders, C. K. (2004) Missing data in educational research: A review of reporting practices and suggestions for improvement. *Review of educational research*, **74**, 525–556.

[15] Setiawan, E. B. (2009) Pemilihan ea framework. *Seminar Nasional Aplikasi Teknologi Informasi (SNATI)*.

[16] Toha, M. (2010) Implementasi framework spring mvc untuk pembuatan sistem informasi manajemen e commerce. Disertasi. Universitas Sebelas Maret.

[17] Gunawan, E. (2015) Mengenal apache maven. http://www.erikgunawan.com/mengenal-apache-maven/. [Online; diakses 09-April-2017].

[18] Cogoluègnes, A. (2013) Introducing the thymeleaf template engine.

[19] Joseph, J. (2016) How to treat missing values in your data. http://www.datasciencecentral.com/profiles/blogs/how-to-treat-missing-values-in-your-data-1. [Online; diakses 09-April-2017].

[20] Oracle (2015) Generic types. https://docs.oracle.com/javase/tutorial/java/generics/types.html. [Online; diakses 19-April-2017].

[21] Walter Savitch, K. M. (2012) *Absolute Java*, 5 edition 5th Edition. Addison Wesley.