

**Tugas Akhir Mata Kuliah EC7010
(Dosen Dr. Ir. Budi Raharjo)**

PENERAPAN *DATA MINING* UNTUK IDS

Oleh

LIA ZALILIA

NIM: 23205322

Program Studi Teknik Elektro



**Institut Teknologi Bandung
2007**

ABSTRAK

PENERAPAN DATA MINING UNTUK IDS

PROGRAM STUDI TEKNIK ELEKTRO

Oleh

Lia Zalilia

NIM : 23205322

Seiring dengan menurunnya biaya pengolahan informasi dan akses internet, organisasi semakin meningkat dan berpotensi terhadap ancaman *cyber* seperti *network intrusion* (penggangguan pada jaringan).

Intrusion adalah tindakan yang mencoba membypass mekanisme keamanan sistem komputer. *Intrusion* disebabkan oleh penyerang yang mengakses sistem dari internet.

Untuk membedakan aktivitas *traffic network* yang ganjil dengan yang normal itu sulit dan membosankan. Seorang analis harus memeriksa seluruh data yang besar dan luas untuk menemukan urutan yang anomali (ganjil) pada koneksi *network*. Untuk mendeteksi *network intrusion* diperlukan suatu cara yang dapat mendeteksi *network intrusion* yang luas dan besar.

Dalam paper ini diungkapkan cara mendeteksi *network intrusion* dengan menggunakan teknik dari *machine learning (data mining)* untuk membuat aturan (*rule*) untuk sistem pendeteksian gangguan. Metoda yang digunakan untuk menghasilkan aturan adalah klasifikasi dengan algoritma *decision tree*.

Decision tree dapat digunakan untuk mengelompokkan peristiwa pada jaringan berdasarkan atribut. Setiap peristiwa pada jaringan akan diturunkan ke dalam bagian yang unik oleh *decision tree*. Urutan peristiwa pada jaringan akan dipetakan pada urutan keterhubungan bagian. Dengan membangaun rules berdasarkan urutan bagian menghasilkan tanda *intrusion* yang dapat mendeteksi segala usaha untuk melakukan *intrusion*.

Kata kunci : *intrusion, data mining, decision tree*.

DAFTAR ISI

	Halaman
ABSTRAK	i
DAFTAR ISI	ii
1. Latar Belakang	1
2. IDS	2
3. Data Mining	4
3.1 Definisi <i>Data Mining</i>	4
3.2 Proses KDD	5
3.3 Tugas <i>Data Mining</i>	6
3.4 Teknik-Teknik <i>Data Mining</i>	7
A. <i>Association Rule Mining</i>	7
B. <i>Clustering</i>	7
C. Klasifikasi	8
4. <i>Decision Tree</i>	12
4.1 Tipe simpul pada <i>tree</i>	12
4.2 Metoda untuk Mengekspresikan Kondisi Tes Atribut	13
4.3 Algoritma Induksi <i>Decision Tree</i>	15
4.4 Membangun <i>Decision Tree</i>	16
4.5 Ukuran untuk Memilih <i>Split</i> Terbaik	18
5. Menerapkan <i>Data Mining</i> untuk IDS	20
5.1 Menghitung <i>Split</i> Terbaik	21
5.2 Membangun <i>Decision Tree</i> Berdasarkan Perhitungan	26
5.3 <i>Pruning Decision Tree</i>	27
5.4 Mengklasifikasikan Kemungkinan <i>Intrusion</i> dengan Data Baru	27

	Halaman
5.5 Mengekstrak <i>Classification Rules</i> dari <i>Decision Tree</i>	28
6. Kesimpulan	28
DAFTAR PUSTAKA	29

PENERAPAN *DATA MINING* UNTUK IDS

1. Latar Belakang

Perkembangan internet yang semakin luas secara tidak langsung berdampak pada kehidupan sosial, bahkan bagi segelintir orang menimbulkan pemikiran untuk memanfaatkan teknologi internet untuk tindakan kriminal. Bagi sebagian orang internet adalah inspirasi dan motivasi untuk melakukan kejahatan baik yang bersifat financial, political, militer ataupun hanya sebagai pemuasan diri untuk eksistensi di dunia maya ini.

Dengan melihat kenyataan seperti di atas sangatlah penting bagi kita untuk memproteksi jaringan yang dikelola dari penyusupan orang-orang yang bermaksud ingin memperoleh informasi untuk maksud jahat. Bagi seorang administrator jaringan apabila jaringan yang dikelola tersebut masih dalam skala kecil dan dapat ditangani tentunya tidak terlalu bermasalah. Akan tetapi hal ini akan sangat bermasalah kalau jaringan yang dikelola sangat besar, luas dan tidak dapat dijangkau dalam waktu dekat. Mendeteksi orang atau aktivitas yang memungkinkan melakukan penyusupan di dalam jaringan seperti itu tentunya membosankan. Contohnya menggunakan *firewall* sebagai penjaga jaringan. *Firewall* hanya bertindak sebagai pagar yang mengelilingi jaringan. Dia tidak dapat mengenali apakah itu merupakan suatu serangan atau bukan. Maka untuk itu diperlukan suatu sistem yang secara otomatis dapat mengenali serangan tersebut.

Intrusion detection system (IDS) dapat mengenali serangan yang tidak dapat di kenali oleh *firewall*. IDS juga dapat mengenali penyerang yang baru yang sedang membangun serangan terhadap jaringan. Akan tetapi IDS tidak dapat mendeteksi semua usaha serangan. Maka dari itu diperlukan suatu cara yang dapat mendeteksi semua serangan secara otomatis dalam jaringan yang besar. Cara tersebut dengan menerapkan algoritma *data mining* pada IDS.

2. IDS

Sejak komputer dihubungkan bersama dalam suatu jaringan, keamanan jaringan menjadi suatu isu penting. Ditambah evolusi internet yang sangat cepat maka masalah keamanan ini menjadi sangat penting.

Intrusion adalah usaha untuk membypass sistem komputer^[6].

IDS mengumpulkan dan memonitor sistem operasi dan aktivitas data pada jaringan, juga menganalisis informasi untuk menjelaskan keadaan selama terjadinya penyerangan^[6].

Berdasarkan tempat dimana data dianalisis IDS diklasifikasikan kedalam.

1. *Network based system*

sistem ditempatkan pada jaringan, dekat dengan sistem atau sistem dimonitor. Sistem menguji network traffic apakah dapat diterima sesuai batasan tertentu.

2. *Host based system*

Dijalankan pada sistem yang dimonitor. Sistem diuji untuk menjelaskan keadaan aktivitas apakah diterima atau tidak.

Network based system menguji *event* (kejadian) sebagai pertukaran paket informasi antara komputer yang ada, *Host based system* menguji *event* seperti *file* yang telah diakses atau telah dieksekusi.

IDS diklasifikasikan dalam 2 kategori berdasarkan bagaimana data dianalisis.

1. *Misuse detection*

Sistem mempelajari pola penyerangan yang ada dan sudah dikenal. Pola ini dipelajari dengan memeriksa seluruh data yang datang untuk menemukan tipe *intrusion*. Metoda ini tidak mampu mendeteksi serangan baru yang polanya belum diketahui.

2. *Anomaly detection*

Pola dipelajari dari data normal. Data yang tidak terlihat dicek dan dicari penyimpangan dari pola yang telah dipelajari. Metoda ini tidak mampu mengidentifikasi tipe serangan.

Misuse detection mempunyai keuntungan tidak hanya mendeteksi *intrusion* tetapi mengenali tipe *intrusion*. Kekurangannya adalah tipe *intrusion* baru tidak dapat dideteksi. Dengan kata lain meskipun *anomaly detection* memiliki kekurangan tidak dapat mendeteksi tipe *intrusion* tetapi *intrusion* baru dapat dideteksi.

Gabungan 2 metoda ini akan sangat membantu, misalnya *anomaly detection* digunakan untuk mencari kemungkinan *intrusion* dan kemudian tipe *intrusion* dapat diidentifikasi dengan *misuse detection*.

Berdasarkan waktu kapan audit data dianalisis terdapat 2 kemungkinan:

1. *off line* IDS
2. *on line* IDS

Off line IDS hanya dapat mendeteksi serangan setelah terjadi penyerangan. *Real time* IDS dapat menangkah usaha penyerangan sebelum status sistem disepakati, tetapi *real time* IDS harus dijalankan bersamaan dengan sistem aplikasi lain yang akan berpengaruh buruk terhadap *throughput*. IDS tidak dapat mendeteksi semua usaha serangan, penyebab utamanya karena hanya skenario *intrusion* saja yang dapat terlihat. Algoritma *Data Mining* diterapkan untuk menganalisis log data *off line mode*, sehingga anomali dapat ditelusuri, dapat dianalisis oleh orang yang ahli, dan kemudian pola untuk menelusuri serangan yang baru dapat dihitung, dan dapat diinstallkan ke dalam *real time* IDS.

3. Data Mining

3.1 Definisi Data Mining

Kemajuan dalam pengumpulan data dan teknologi penyimpanan yang cepat memungkinkan organisasi menghimpun jumlah data yang sangat luas. Alat dan teknik analisis data yang tradisional tidak dapat digunakan untuk mengekstrak informasi dari data yang sangat besar. Untuk itu diperlukan suatu metoda baru yang dapat menjawab kebutuhan tersebut. *Data mining* merupakan teknologi yang menggabungkan metoda analisis tradisional dengan algoritma yang canggih untuk memproses data dengan volume besar.

Terdapat beberapa pengertian yang berkaitan dengan *data mining* dari beberapa referensi sebagai berikut.

1. *Data mining* adalah mencocokkan data dalam suatu model untuk menemukan informasi yang tersembunyi dalam basisdata^[1].
2. *Data mining* merupakan aplikasi suatu algoritma untuk menggali informasi bermanfaat dari dalam basisdata^[2].
3. *Data mining* adalah proses menemukan pola-pola didalam data, dimana proses penemuan tersebut dilakukan secara otomatis atau semi otomatis dan pola-pola yang ditemukan harus bermanfaat^[3].
4. *Data mining* adalah proses penemuan informasi yang berguna pada penyimpanan data yang besar secara otomatis^[7].
5. *Data mining* atau *Knowledge Discovery in Databases* (KDD) adalah pengambilan informasi yang tersembunyi, dimana informasi tersebut sebelumnya tidak dikenal dan berpotensi bermanfaat. Proses ini meliputi sejumlah pendekatan teknis yang berbeda, seperti *clustering*, *data summarization*, *learning classification rules*^[2].

Secara sederhana *data mining* adalah ekstraksi informasi atau pola yang penting atau menarik dari data yang ada di basisdata yang besar. Dalam jurnal ilmiah, *data mining* juga dikenal dengan nama KDD.

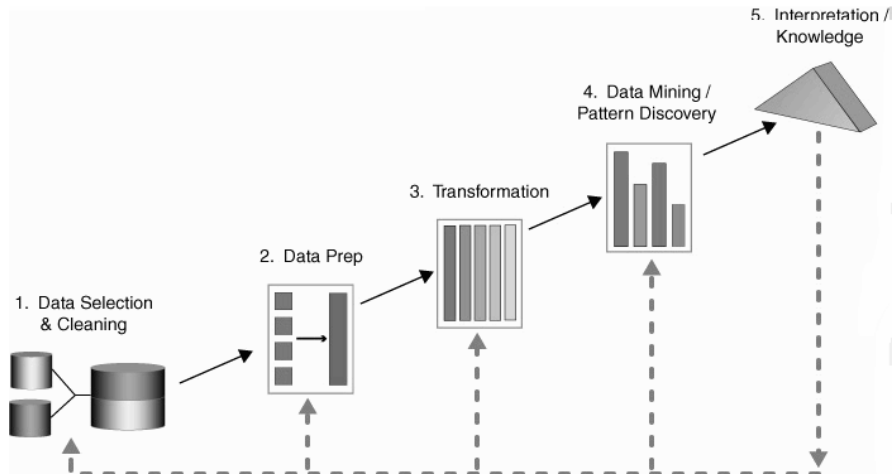
3.2 Proses KDD

Data Mining dapat digambarkan sebagai suatu proses menemukan pengetahuan baru yang menarik, seperti pola , asosiasi, aturan, perubahan, keganjilan dan struktur penting dari sejumlah besar data yang disimpan pada bank data dan tempat penyimpanan informasi lainnya.

Secara umum, proses KDD terdiri dari langkah-langkah dibawah ini^[2]:

1. pemilihan data (*data selection*), pemilihan data relevan yang didapat dari basisdata;
2. pembersihan data (*data cleaning*), proses menghilangkan *noise* dan data yang tidak konsisten atau data tidak relevan;
3. pengintegrasian data (*data integration*), penggabungan data dari berbagai basisdata ke dalam satu basisdata baru;
4. transformasi data, data diubah atau digabung ke dalam format yang sesuai untuk diproses dalam *data mining*;
5. *data mining*, suatu proses di mana metoda diterapkan untuk menemukan pengetahuan berharga dan tersembunyi dari data;
6. evaluasi pola (*pattern evaluation*), untuk mengidentifikasi pola-pola menarik untuk di representasikan kedalam *knowledge based*;
7. representasi pengetahuan (*knowledge presentation*), visualisasi dan penyajian pengetahuan mengenai teknik yang digunakan untuk memperoleh pengetahuan yang diperoleh pengguna.

Proses KDD seperti terlihat pada Gambar 3.1 dibawah ini.



Gambar 3.1 *Data mining* sebagai salah satu bagian dalam proses KDD^[2].

Proses *data mining* selalu berhubungan dengan pengguna atau *knowledge base*. Pola-pola yang menarik akan dipresentasikan kepada pengguna dan kemudian disimpan sebagai pengetahuan yang baru di dalam *knowledge base*.

3.3 Tugas *Data Mining*

Umumnya tugas *data mining* dibagi dalam 2 kategori.

- Tugas prediktif
Sasaran pada tugas ini adalah memprediksikan nilai atribut tertentu berdasarkan nilai atribut yang lain. Atribut yang diprediksi dikenal sebagai target atau variabel yang tergantung pada variabel lain, atribut yang digunakan selama membuat prediksi dikenal sebagai penjelasan (*explanatory*) atau variabel yang bebas.
- Tugas deskriptif
Sasaran pada tugas ini adalah memperoleh pola (kecenderungan korelasi, *cluster* dan anomali) yang menyimpulkan hubungan dalam data. Tugas deskriptif *data mining* memerlukan teknik *postprocessing* untuk validasi dan kejelasan hasil.

3.4 Teknik-Teknik dalam *Data Mining*

Data mining adalah serangkaian proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual. Perlu diingat bahwa kata *mining* sendiri berarti usaha untuk mendapatkan sedikit data berharga dari sejumlah besar data dasar. Karena itu *data mining* sebenarnya memiliki akar yang panjang dari bidang ilmu seperti kecerdasan buatan (*artificial intelligent*), *machine learning*, statistik dan basisdata. Beberapa teknik yang sering disebut-sebut dalam literatur *data mining* antara lain yaitu *association rule mining*, *clustering*, *klasifikasi*, *neural network*, *genetic algorithm* dan lain-lain.

A. *Association Rule Mining*

Association rule mining adalah teknik *mining* untuk menemukan aturan asosiatif antara suatu kombinasi atribut.

Contoh dari aturan asosiatif dari analisa pembelian di suatu pasar swalayan diketahui berapa besar kemungkinan seorang pelanggan membeli roti bersamaan dengan susu. Dengan pengetahuan tersebut pemilik pasar swalayan dapat mengatur penempatan barangnya atau merancang strategi pemasaran dengan memakai kupon diskon untuk kombinasi barang tertentu.

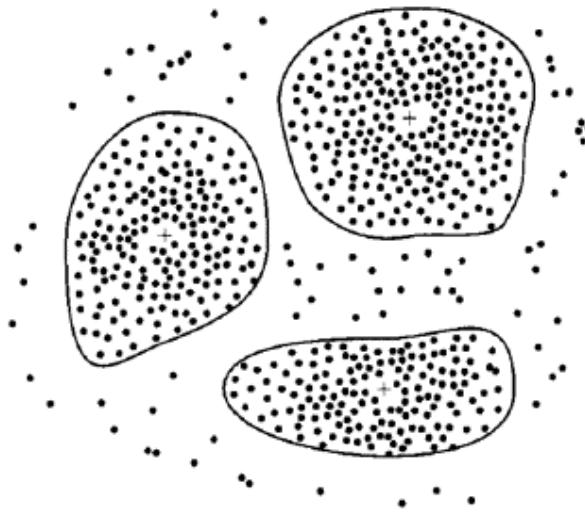
Penting tidaknya suatu aturan asosiatif dapat diketahui dengan dua parameter, *support* yaitu prosentasi kombinasi atribut tersebut dalam basisdata dan *confidence* yaitu kuatnya hubungan antar atribut dalam aturan asosiatif.

B. *Clustering*

Berbeda dengan *association rule mining* dan *klasifikasi* dimana kelas data telah ditentukan sebelumnya, *clustering* melakukan pengelompokan data tanpa berdasarkan kelas data tertentu. Bahkan *clustering* dapat dipakai untuk memberikan label pada kelas data yang belum diketahui. Karena itu *clustering* sering digolongkan sebagai metode *unsupervised learning*.

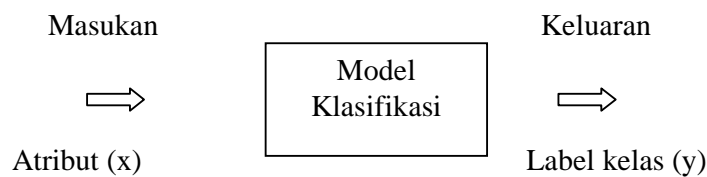
Prinsip dari *clustering* adalah memaksimalkan kesamaan antar anggota satu kelas dan meminimumkan kesamaan antar *cluster*. *Clustering* dapat dilakukan pada data yang memiliki beberapa atribut yang dipetakan sebagai ruang multidimensi.

Ilustrasi dari *clustering* dapat dilihat di Gambar 2 dimana lokasi, dinyatakan dengan bidang dua dimensi, dari pelanggan suatu toko dapat dikelompokkan menjadi beberapa *cluster* dengan pusat *cluster* ditunjukkan oleh tanda positif (+). Banyak algoritma *clustering* memerlukan fungsi jarak untuk mengukur kemiripan antar data, diperlukan juga metoda untuk normalisasi bermacam atribut yang dimiliki data.



Gambar 3.2 *Clustering*^[3].

C. Klasifikasi



Gambar 3.3 Klasifikasi sebagai suatu tugas memetakan atribut x ke dalam label kelas y ^[7].

Masukan data untuk klasifikasi adalah kumpulan *record*. Setiap *record* dikenal sebagai *instance* atau contoh yang ditandai oleh *tuple* (x,y) dimana x adalah atribut dan y adalah atribut khusus yang menunjukkan label kelas (disebut juga kategori atau atribut target).

Tabel 3.1 menunjukkan contoh data yang digunakan untuk mengklasifikasikan IP Port yang normal dan *intrusion*. Label kelas haruslah atribut diskrit. Hal ini merupakan karakteristik kunci yang membedakan klasifikasi dengan regresi sebagai pemodelan *predictive task* dimana pada regresi y adalah atribut *continuous*.

a. Definisi klasifikasi

Klasifikasi adalah tugas dari pembelajaran fungsi target f yang memetakan setiap atribut x terhadap satu label kelas yang sudah ditentukan $y^{[Tan, P.]}$. Fungsi target disebut juga model klasifikasi. Model klasifikasi yang digunakan terdiri dari.

- Pemodelan deskriptif

Dapat bertindak sebagai suatu alat yang bersifat menjelaskan untuk membedakan antara objek dengan klas yang berbeda.

Tabel 3.1 Pemodelan deskriptif^[6].

IP Port	Nama Sistem	Kategori
004020	Artemis	Normal
004020	Apollo	<i>Intrusion</i>
002210	Artemis	Normal
002210	Apollo	<i>Intrusion</i>
000010	Artemis	Normal
000010	Apollo	Normal

- Pemodelan prediktif

Model klasifikasi juga dapat menggunakan prediksi label kelas yang belum diketahui recordnya. Seperti terlihat pada Gambar 3.3, sebuah model klasifikasi dapat di perlakukan sebagai kotak hitam yang secara otomatis menandai sebuah label kelas ketika disajikan dengan set atribut yang belum diketahui recordnya. Andaikan diberikan karakteristik dari IP dengan sistem yang baru Win.

Tabel 2 Pemodelan prediktif^[6].

IP Port	Nama Sistem	Kategori
000010	Win	?

Dari data pada Tabel 3.1 model klasifikasi dapat dibuat untuk menjelaskan kelas yang dimiliki IP. Teknik klasifikasi yang sering sesuai untuk memprediksi atau menjelaskan dataset adalah kategori binary atau nominal.

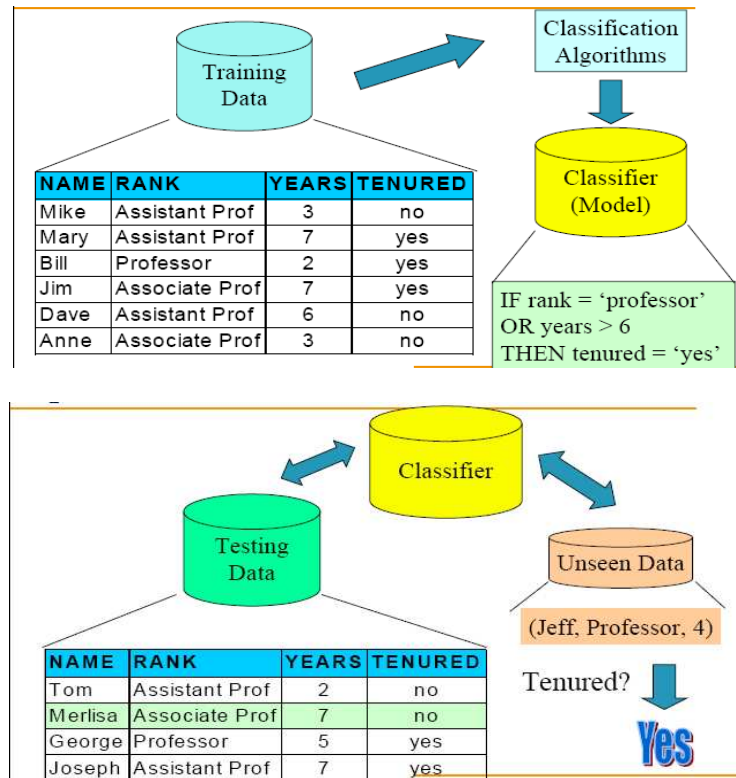
b. Pendekatan umum untuk menyelesaikan masalah klasifikasi

Sebuah teknik klasifikasi atau *classifier* adalah pendekatan yang sistematis untuk membuat klasifikasi model dari kumpulan input data. Contohnya terdiri dari *decision tree classifier*, *rule-based classifier*, *neural network*, dan *naive bayes classifier*.

Setiap teknik mengerjakan sebuah algoritma pembelajaran untuk mengidentifikasi model paling sesuai dengan hubungan antara set atribut dan label kelas dari masukan data.

Model yang dihasilkan oleh algoritma pembelajaran kedua-duanya harus cocok dengan masukan data yang baik dan dengan tepat memprediksikan label kelas dari *record* yang belum pernah terlihat sebelumnya. Oleh karena itu kunci obyektif dari algoritma pembelajaran adalah membuat model yang baik dan mempunyai kemampuan yang sama.

c. Tahapan proses pembuatan model klasifikasi



Gambar 3.4 tahapan proses klasifikasi^[3].

Pada Gambar 3.4 terdiri dari pembuatan model dan penggunaan model. Pembuatan model menguraikan sebuah set dari penentuan kelas-kelas sebagai:

- setiap *tuple* diasumsikan sudah mempunyai kelas sudah dikenal seperti ditentukan oleh label kelas atribut,
- kumpulan *tuple* yang digunakan untuk membuat model disebut kumpulan pelatihan (*training set*),
- model direpresentasikan sebagai *classification rules*, *decision tree* atau formula matematika.

Penggunaan model menguraikan pengklasifikasian masa yang akan datang atau obyek yang belum ketahui, yaitu taksiran keakuratan dari model yang terdiri dari :

- label yang telah diketahui dari contoh tes dibandingkan dengan hasil klasifikasi dari model,
- nilai keakuratan adalah prosentase dari kumpulan contoh tes yang diklasifikasikan secara tepat oleh model,
- kumpulan tes tidak terikat pada kumpulan pelatihan,
- jika akurasi diterima, gunakan model untuk mengklasifikasikan data *tuple* yang label kelasnya belum diketahui.

4. Decision Tree

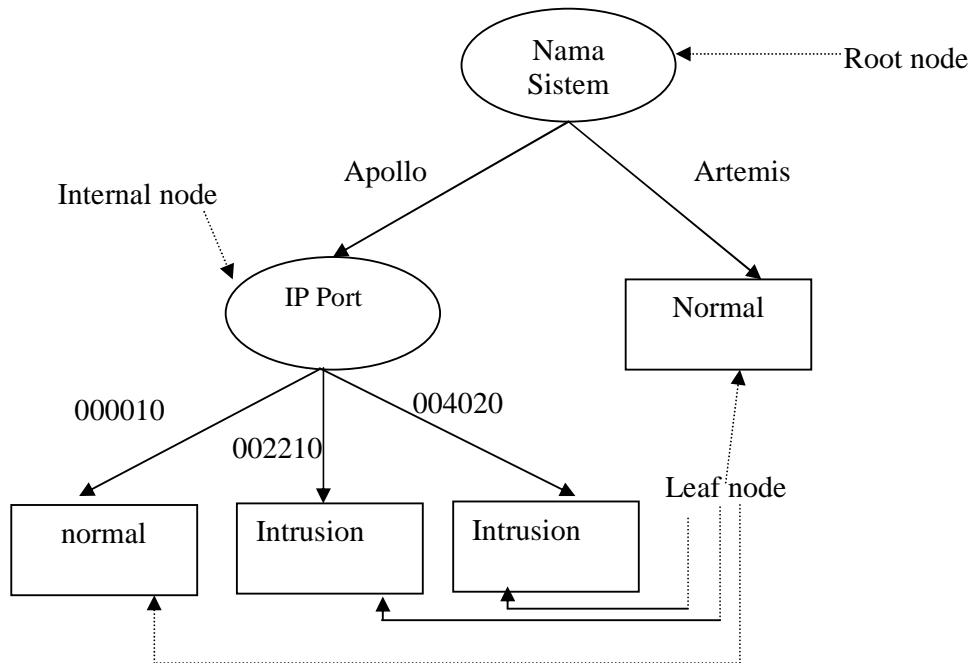
Salah satu metoda *Data Mining* yang umum digunakan adalah *decision tree*. *Decision tree* adalah struktur *flowchart* yang menyerupai *tree* (pohon), dimana setiap simpul internal menandakan suatu tes pada atribut, setiap cabang merepresentasikan hasil tes, dan simpul daun merepresentasikan kelas atau distribusi kelas^[3]. Alur pada *decision tree* di telusuri dari simpul akar ke simpul daun yang memegang prediksi kelas untuk contoh tersebut. *Decision tree* mudah untuk dikonversi ke aturan klasifikasi (*classification rules*).

4.1 Tipe Simpul Pada Tree

Tree mempunyai 3 tipe simpul yaitu:

- simpul akar dimana tidak ada masukan *edge* dan 0 atau lebih keluaran *edge* (tepi),
- simpul internal, masing-masing 1 masukan *edge* dan 2 atau lebih *edge* keluaran,
- simpul daun atau simpul akhir, masing-masing 1 masukan *edge* dan tidak ada *edge* keluaran.

Pada *decision tree* setiap simpul daun menandai label kelas. Simpul yang bukan simpul akhir terdiri dari akar dan simpul internal yang terdiri dari kondisi tes atribut pada sebagian *record* yang mempunyai karakteristik yang berbeda. Simpul akar dan simpul internal ditandai dengan bentuk oval dan simpul daun ditandai dengan bentuk segi empat^[3].



Gambar 3.5 *Decision tree* untuk masalah klasifikasi *intrusion*^[6].

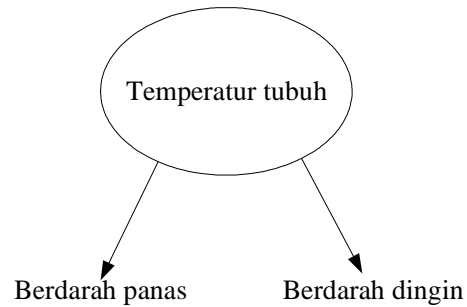
Mengklasifikasikan tes *record* secara langsung telah membangun *decision tree*. Dimulai dari simpul akar, kemudian terapkan kondisi tes pada *record* dan ikuti sesuai cabang berdasarkan hasil dari tes. Hal ini berlaku juga untuk simpul internal dimana suatu kondisi tes baru akan diterapkan pada simpul daun. Label kelas akan berhubungan dengan simpul daun pada *record* yang ditugaskan.

4.2 Metode untuk Mengekspresikan Kondisi Tes Atribut

Algoritma induksi *decision tree* harus menyediakan sebuah metoda untuk mengekspresikan tes atribut dan berhubungan dengan hasil untuk setiap atribut yang berbeda.

- Atribut biner

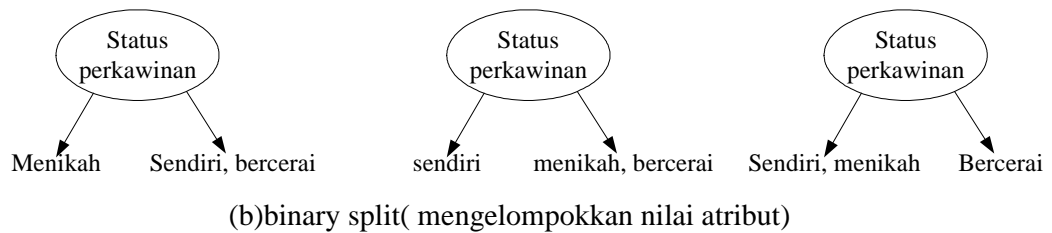
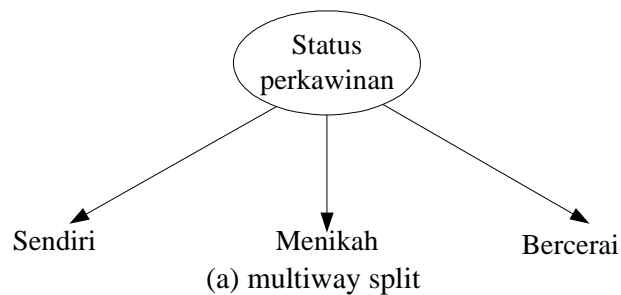
Kondisi tes untuk atribut biner menghasilkan 2 hasil keluaran.



Gambar 3.6 kondisi tes untuk atribut biner^[7].

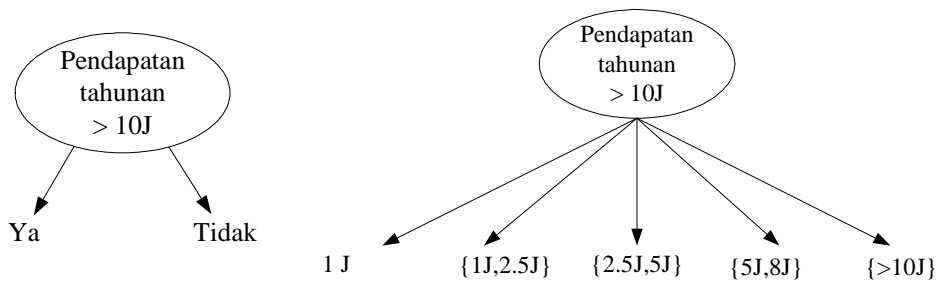
- Atribut nominal

Atribut nominal memiliki beberapa nilai, kondisi tes dapat diekspresikan ke dalam *split* dua cara atau banyak cara. Untuk *split* banyak cara jumlah hasil keluaran berdasarkan perbedaan nilai untuk atribut yang berhubungan.



Gambar 3.7 kondisi tes untuk atribut nominal^[7].

- Atribut ordinal
Atribut ordinal juga dapat menghasilkan *split* dua cara atau banyak cara.
- Atribut *continuous*
Untuk atribut *continuous* kondisi tes dapat diekspresikan sebagai sebuah tes perbandingan ($A < v$) atau ($A \geq v$) dengan hasil keluaran biner atau interval dengan hasil keluaran dalam bentuk $v_i \leq A < v_{i+1}$ untuk $i = 1, \dots, k$.

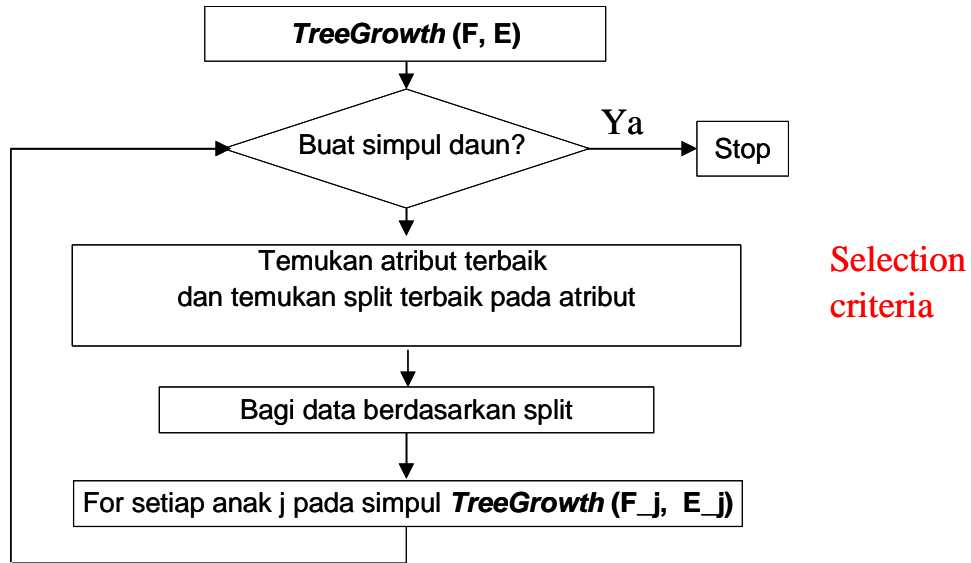


Gambar 3.8 Kondisi tes untuk atribut *continuous*^[7].

4.3 Algoritma Induksi *Decision Tree*

Rangka algoritma induksi *decision tree* disebut *TreeGrowth* yang terlihat pada algoritma di bawah ini. Masukan terdiri dari *record* pelatihan (*training record*) E dan atribut F. Cara kerja algoritma yaitu dengan memilih atribut yang terbaik untuk memisahkan data secara rekursif dan mengembangkan simpul daun pada tree sampai ditemui kriteria untuk berhenti .

Algoritma induksi *decision tree*^[4]



Gambar 3.9 Algoritma induksi *decision tree*^[4].

4.4 Membangun *Decision Tree*

Salah satu algoritma yang digunakan untuk membangun *decision tree* yang berbasis algoritma induksi *decision tree* seperti ID3, C4.5 dan CART adalah algoritma Hunt.

Pada algoritma Hunt *decision tree* tumbuh dalam struktur perulangan dengan membagi *record* pelatihan ke dalam bagian yang berurutan. T adalah *record* pelatihan yang berhubungan dengan simpul t dan $C = \{C_1, C_2, \dots, C_t\}$ adalah label kelas.

Metoda Hunt untuk membangun *decision tree*:

Diberikan pelatihan kumpulan T dengan kelas $C = \{C_1, C_2, \dots, C_t\}$,

Sebuah *tree* dibangun dengan melakukan pengujian secara sebagai berikut^[6].

T berisi satu atau lebih kelas yang sama C_j .

Sebuah simpul daun dibuat untuk T yang menotasikan kelas kepunyaan C_j .

T tidak berisi kasus apapun .

Sebuah simpul daun dibuat untuk T tetapi kelas yang dimilikinya harus dipilih dari luar sumber. Algoritma C4.5 memilih kelas yang frekuensinya tertinggi pada simpul orang tua.

T berisi kasus dengan kelas lebih dari satu .

Temukan sebuah pengujian yang akan memecah T ke dalam kasus dengan satu kelas.

Pengujian ini berdasarkan nilai atribut tunggal, dan pilihan seperti itu menghasilkan satu atau lebih keluaran $\{O_1, O_2, \dots, O_n\}$.

kumpulan T kemudian dipecah ke dalam bagian $\{T_1, T_2, \dots, T_n\}$ yang berakibat kumpulan T_i berisi semua kasus dalam T dengan keluaran O_i . Algoritma dikerjakan secara rekursif untuk semua bagian T .

Di bawah ini adalah definisi perulangan pada algoritma Hunt.

Langkah 1

Jika semua *record* dalam *decision tree* mempunyai kelas yang sama C_t , maka t adalah simpul daun yang diberi label sebagai C_t .

Langkah 2

Jika *decision tree* terdiri dari *record* yang mempunyai lebih dari 1 kelas, maka sebuah kondisi tes atribut dipilih untuk membagi *record* ke dalam bagian yang terkecil. Sebuah simpul anak dibuat untuk setiap hasil pada kondisi tes dan *record* pada T akan didistribusikan ke simpul anak berdasarkan hasil. Algoritma kemudian akan berulang pada setiap simpul anak.

Algoritma Hunt akan bekerja jika setiap kombinasi pada nilai atribut ada dalam data pelatihan dan setiap kombinasi memiliki label kelas yang unik. Ada kondisi tambahan untuk menangani beberapa kasus.

1. Kemungkinan untuk beberapa simpul anak yang terbuat dalam langkah 2 menjadi kosong. Contoh tidak ada *record* yang berhubungan dengan simpul tersebut. Ini dapat terjadi jika tidak satu pun *record* pelatihan mempunyai nilai kombinasi atribut yang berhubungan dengan beberapa simpul tersebut. Dalam kasus ini simpul menerangkan simpul daun dengan label kelas yang sama sebagai kelas mayoritas dari *record* pelatihan yang berhubungan dengan simpul orang tua.
2. Pada tahap 2, jika semua *record* berhubungan dengan C_i mempunyai nilai atribut yang identik (kecuali untuk label kelas), maka tidak mungkin memotong atau membagi *record* tersebut. Pada kasus ini simpul adalah simpul daun yang mempunyai label kelas yang sama sebagai kelas mayoritas pada *record* pelatihan yang berhubungan dengan simpul tersebut.

4.5 Ukuran untuk Memilih *Split* Terbaik

Pemilihan atribut pada algoritma induksi *decision tree* menggunakan ukuran berdasarkan *entropy* yang dikenal dengan *information gain* sebagai sebuah *heuristic* untuk memilih atribut yang merupakan bagian terbaik dari contoh ke dalam kelas. Semua atribut adalah bersifat kategori yang bernilai diskrit. Atribut dengan nilai *continuous* harus didiskritkan.

Ukuran *information gain* digunakan untuk memilih tes atribut pada setiap simpul dalam *tree*. Atribut dengan informasi tertinggi (nilai pengurangan *entropy* yang terbesar) dipilih sebagai tes atribut untuk simpul tersebut. Atribut ini meminimalkan informasi yang dibutuhkan untuk mengklasifikasikan contoh pada proses pembagian dan mencerminkan ketidakmurnian (*impurity*).

Misalkan s adalah kumpulan dari s contoh data. Andaikan atribut label kelas mempunyai m nilai berbeda yang menjelaskan m nilai kelas yang berbeda, C_i (*for* $i = 1, \dots, m$). Misalkan s_i menjadi jumlah contoh S dalam kelas C_i .

Informasi yang dibutuhkan untuk mengklasifikasikan contoh yang diberikan adalah sebagai berikut.

$$I(s_1, s_2, \dots, s_m) = - \sum_{i=1}^m p_i \log_2(p_i) \quad \dots(4.1)$$

dimana p_i adalah kemungkinan contoh kepunyaan kelas C_i dan diperkirakan oleh $s_i|s$. Catatan fungsi log basis 2 digunakan semenjak informasi dikodekan dalam bit-bit.

Misalkan atribut A mempunyai nilai v yang berbeda, $\{a_1, a_2, \dots, a_v\}$. Atribut A dapat digunakan untuk membagi S kedalam v bagian $\{S_1, S_2, \dots, S_v\}$ dimana S_j berisi contoh di S yang mempunyai nilai a_j dari A . Jika A terpilih sebagai tes atribut maka bagian ini akan sesuai dengan pertumbuhan cabang dari simpul yang berisi S . *Entropy* atau informasi berdasarkan pembagian ke dalam bagian A sebagai berikut.

$$E(A) = \sum_{j=1}^v \frac{s_{ij} + \dots + s_{mj}}{s} I(s_{ij}, \dots, s_{mj}) \quad \dots(4.2)$$

bentuk $\frac{s_{ij} + \dots + s_{mj}}{s}$ adalah bobot dari bagian j dan merupakan jumlah contoh pada subbagian dibagi oleh total jumlah contoh dalam S . Nilai *entropy* terkecil adalah kemurnian (*purity*) terbesar pada pembagian subbagian. Catatan untuk subbagian s_j ,

$$I(s_{1j}, s_{2j}, \dots, s_{mj}) = - \sum_{i=1}^m p_{ij} \log_2(p_{ij}) \quad \dots(4.3)$$

dimana $p_{ij} = \frac{s_{ij}}{|S_j|}$ adalah probabilitas pada contoh S_j kepunyaan kelas C_i .

Pengkodean informasi yang akan diperoleh dari percabangan pada A adalah

$$Gain(A) = I(s_1, s_2, \dots, s_m) - E(A) \quad \dots(4.4)$$

Dengan kata lain, $Gain(A)$ adalah reduksi yang diharapkan dalam *entropy* yang disebabkan oleh pengetahuan nilai pada atribut A .

Algoritma menghitung *information gain* pada setiap atribut. Simpul A dibuat dan dilabelkan dengan atribut, cabang dibuat untuk setiap nilai atribut. Atribut dengan nilai *gain* terbesar dipilih sebagai tes atribut (simpul akar).

5. Menerapkan *Data Mining* untuk IDS

Teknik data mining diterapkan pada *network based* IDS untuk melindungi *military subnetwork*. Setiap *military subnetwork* adalah suatu pemeriksaan yang menyaring dan membukukan *traffic network* ke dalam database pusat. Sebuah *rule set* digunakan untuk menganalisis *archived* data untuk menemukan pola *intrusive*. Pola yang ditemukan terlihat sederhana, seperti melihat aktivitas yang berlebihan seperti koneksi dari IP address yang mempunyai kebiasaan *intrusive*. Contoh dari tipe *intrusion* seperti ini adalah serangan yang rendah dan lama yang berisi kebiasaan *intrusive* selama berjam-jam, berhari-hari atau berminggu-minggu yang dimulai dari berbagai jaringan. Data mining dapat diterapkan pada masalah ini untuk mengembangkan *human pattern recognition*.

Contoh *traffic network* pada suatu jaringan sebagai berikut

Tabel 5.1 *Traffic network*^[6].

Source IP	Dest IP	Source Port	Dest Port	Protocol	<i>Intrusion</i>
123.202.72.109	225.142.187.12	001360	000080	IP	true
123.202.72.109	225.142.187.12	001425	000080	IP	true
123.202.72.109	225.142.187.12	001488	000080	IP	true
123.202.72.109	225.142.187.12	001559	000080	IP	true
123.202.72.109	225.142.187.12	001624	000080	IP	true
123.202.72.109	225.142.187.12	002156	000080	IP	true
123.202.72.109	225.142.187.12	002158	000080	IP	true
225.142.175.75	150.216.191.119	001624	000080	IP	true
225.142.187.19	125.250.187.19	004207	000025	IP	true
233.167.15.65	225.142.187.12	004607	000025	IP	false
233.167.15.65	225.142.187.12	004690	000025	IP	false
139.61.51.70	225.142.187.12	001052	000021	IP	false
142.142.5.113	225.142.187.12	001572	000080	IP	false

5.1 Menghitung *Split* Terbaik

Langkah 1. menghitung nilai informasi untuk data *traffic network*.

Ada 13 *instance* dengan label kelas *true* = 9 dan *false* = 4 maka nilai informasinya adalah

$$I(s_1, s_2) = I(9, 5) = -\frac{9}{14} \log_2 \frac{9}{14} - \frac{4}{14} \log_2 \frac{4}{14} = 0,890$$

Langkah 2 menghitung nilai informasi untuk setiap atribut.

1. Source IP

- 123.202.72.109 true = 7, false = 0

$$I(7, 0) = 0$$

- 225.142.175.75 true = 1, false = 0

$$I(1, 0) = 0$$

- 225.142.187.19 true = 1, false = 0

$$I(1, 0) = 0$$

- 233.167.15.65 true = 0, false = 2

$$I(0, 2) = 0$$

- 139.61.51.70 true = 0, false = 1

$$I(0, 1) = 0$$

- 142.142.5.113 true = 0, false = 1

$$I(0, 1) = 0$$

2. Dest IP

- 225.142.187.12 true = 7, false = 4

$$I(3, 4) = -\frac{7}{11} \log_2 \frac{7}{11} - \frac{4}{11} \log_2 \frac{4}{11} = 0,946$$

- 150.216.191.119 true = 1, false = 0

$I(1,0)=0$

- 125.250.187.19 true = 1, false = 0

$I(1,0)=0$

3. Source Port

- 001360 true = 1, false = 0

$I(1,0)=0$

- 001425 true = 1, false = 0

$I(1,0)=0$

- 001425 true = 1, false = 0

$I(1,0)=0$

- 001428 true = 1, false = 0

$I(1,0)=0$

- 001559 true = 1, false = 0

$I(1,0)=0$

- 001624 true = 2, false = 0

$I(2,0)=0$

- 002156 true = 1, false = 0

$I(1,0)=0$

- 002158 true = 1, false = 0

$I(1,0)=0$

- 004207 true = 1, false = 0

$I(1,0)=0$

- 004607 true = 0, false = 1

$$I(0,1)=0$$

- 004690 true = 0, false = 1

$$I(0,1)=0$$

- 001052 true = 0, false = 1

$$I(0,1)=0$$

- 001572 true = 0, false = 1

$$I(0,1)=0$$

4. Dest Port

- 000080 true = 8, false = 1

$$I(4,2)=-\frac{8}{9}\log_2\frac{8}{9}-\frac{1}{9}\log_2\frac{1}{9}=0,503$$

- 000025 true = 1, false = 2

$$I(2,2)=-\frac{1}{3}\log_2\frac{1}{3}-\frac{2}{3}\log_2\frac{2}{3}=0,198$$

- 000021 true = 0, false = 1

$$I(0,1)=0$$

Langkah 3 menghitung *entropy* untuk setiap atribut

- Entropy Source IP

$$E=\frac{7}{13}I[7,0]+\frac{1}{13}I[1,0]+\frac{1}{13}I[1,0]+\frac{2}{13}I[0,2]+\frac{1}{13}I[0,1]+\frac{1}{13}I[0,1]=0$$

- Entropy Dest IP

$$E = \frac{11}{13} I[7,4] + \frac{1}{13} I[1,0] + \frac{1}{13} I[1,0] = 0,800$$

- Entropy Source Port

$$E = \frac{1}{13} I[1,0] + \frac{1}{13} I[1,0] + \frac{1}{13} I[1,0] + \frac{1}{13} I[1,0] + \frac{2}{13} I[2,0] + \frac{1}{13} I[1,0] + \frac{1}{13} I[1,0] + \frac{1}{13} I[1,0] + \frac{1}{13} I[0,1] + \frac{1}{13} I[0,1] + \frac{1}{13} I[0,1] + \frac{1}{13} I[0,1] + \frac{1}{13} I[0,1] = 0$$

- Entropy Dest Port

$$E = \frac{9}{13} I[8,1] + \frac{3}{13} I[1,2] + \frac{1}{13} I[0,1] = 0,560$$

Langkah 4 menghitung *gain* untuk setiap atribut

- Gain Source IP

$$G = 0,890 - 0 = 0,890$$

- Gain Dest IP

$$G = 0,890 - 0,800 = 0,090$$

- Gain Source Port

$$G = 0,890 - 0 = 0,890$$

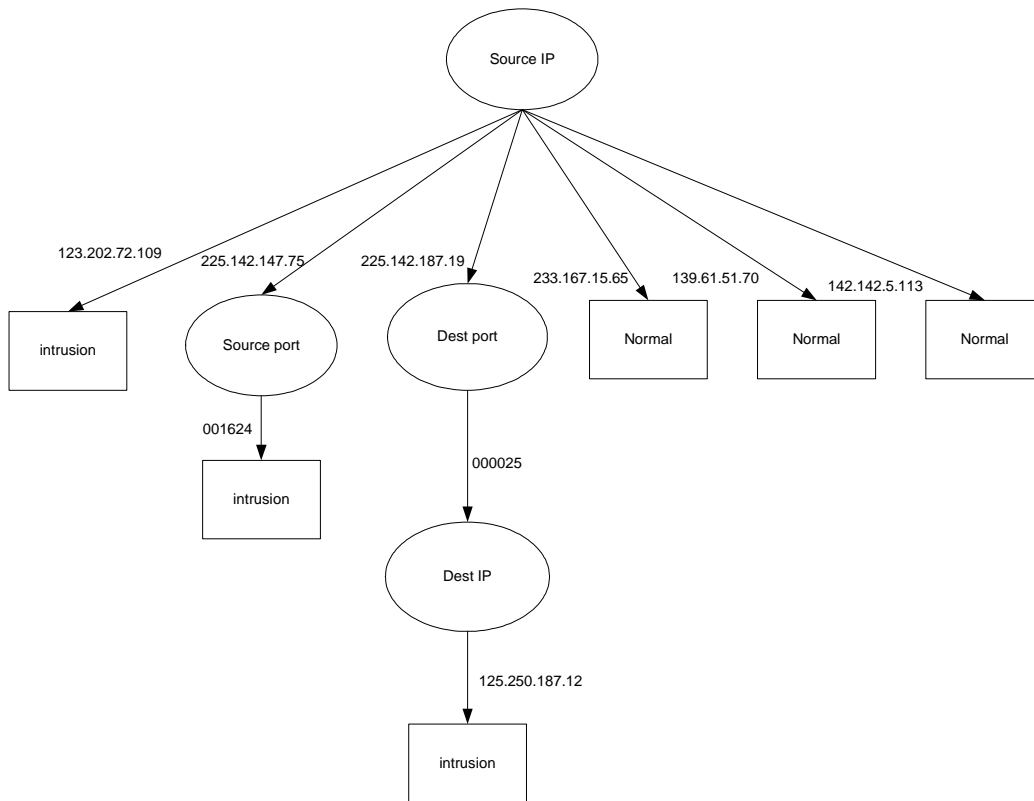
- Gain Dest Port

$$G = 0,890 - 0,560 = 0,330$$

Tabel 5.2 Hasil perhitungan.

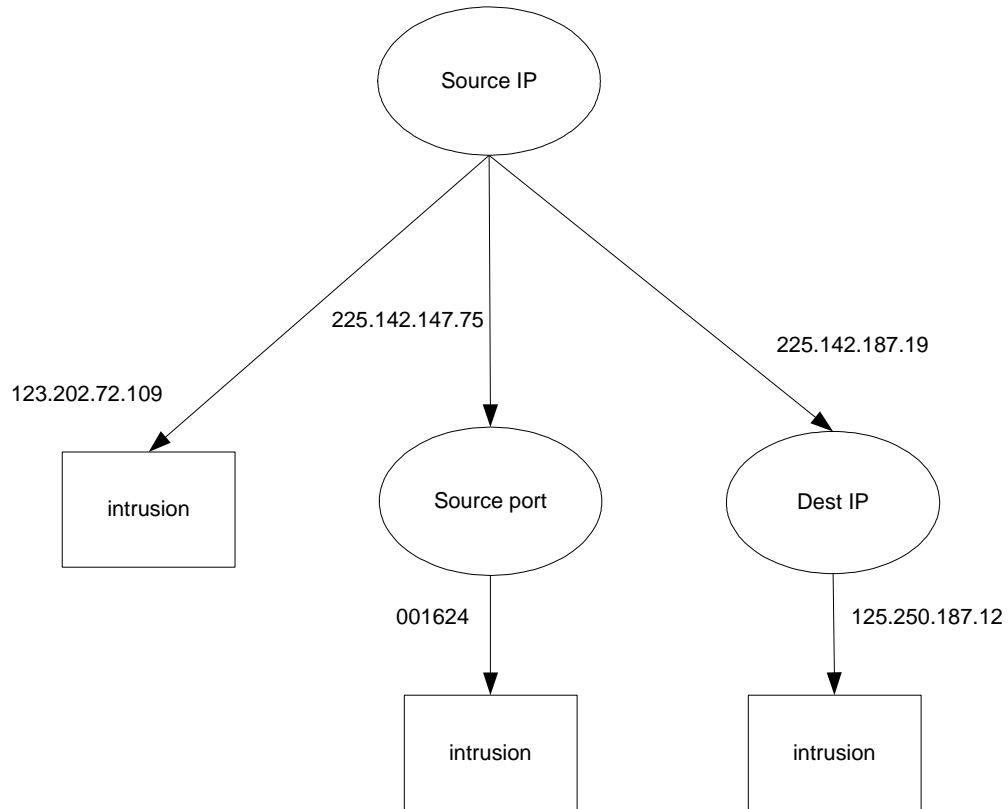
Tes	Info	Entropy	Gain
Contoh (s)	$I(s_1, s_2) = 0,890$	-	-
Source IP	123.202.72.109 = 0 225.142.175.75 = 0 225.142.187.19 = 0 233.167.15.65 = 0 139.61.51.70 = 0 142.142.5.113 = 0	0	0,890
Dest IP	225.142.187.12 = 0,946 150.216.191.119 = 0 125.250.187.19 = 0	0,800	0,090
Source Port	001360 = 0 001425 = 0 001425 = 0 001428 = 0 001559 = 0 001624 = 0 002156 = 0 002158 = 0 004207 = 0 004607 = 0 004690 = 0 001052 = 0 001572 = 0	0	0,890
Dest Port	000080 = 0,503 000025 = 0,198 000021 = 0	0,560	0,330

5.2 Membangun *Decision Tree* Berdasarkan Perhitungan



Gambar 5.1 *decision tree traffic network* untuk IDS

5.3 Pruning Decision Tree



Gambar 5.2 Pruning decision tree traffic network untuk IDS

5.4 Mengklasifikasikan Kemungkinan *Intrusion* dengan Data Baru

Diberikan data baru *traffic network* dengan ciri-ciri sebagai berikut :

Tabel 5.3 Data baru *traffic network* tanpa label kelas

Source IP	Dest IP	Source Port	Dest Port	Protocol	<i>Intrusion</i>
123.202.72.109	225.142.187.12	001360	000021	IP	?
225.142.147.75	225.142.187.12	001624	000080	IP	?
225.142.187.19	225.142.187.12	001052	000025	IP	?

DAFTAR PUSTAKA

- [1] Dunham, H. Margareth (2002), *Data Mining: Introductory and Advanced*, Prentice Hall.
- [2] Fayyad, U., Piatetsky-Shapiro, G. dan Smyth, P. (1996), *From Data Mining to Knowledge Discovery in Databases*, AAAI and The MIT Pres, 37-53.
- [3] Han, J., Kamber, M. (2001), *Data Mining: Concepts and Techniques*, Morgan Kaufman.
- [4] Kohavi, R., Quinlan (1999), *Decision Tree Discovery*, AAAI and The MIT Pres, 1-16.
- [5] Weenke, L at. All (2001), *Real Time Data Mining based- Intrusion Detection*, Proceeding DARPA.
- [6] Sinclair, C., Pierce, L., Matzner, S. (2000), *An Application of Machine Learning to Network Intrusion Detection*, Applied Research Laboratory Technical Report No.859 dan 875, Applied Research Laboratory, The University of Texas at Austin.
- [7] Tan P. N., Steinbach, M., Kumar, V. (2006), *Introduction to Data Mining*, Addison Wesley.