# SKRIPSI

# DATA MINING HISTORI PENCARIAN RUTE ANGKOT

JOVAN GUNAWAN

NPM: 2011730029

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2014

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# BAB 1

# PENDAHULUAN

## 1.1 Latar Belakang

Pertumbuhan teknologi hingga saat ini telah menghasilkan banyak sekali data-data, namun sering kali pemilik data hanya menggunakan data tersebut seperlunya saja. Jika dilihat lebih rinci, sebenarnya jika data tersebut diolah lebih lanjut, dapat menghasilkan sesuatu yang lebih. Salah satu cara mengolah data tersebut adalah dengan menggunakan teknik *data mining*. Dengan menggunakan teknik *data mining* akan mempermudah menganalisa masalah, pengambilan kesimpulan, bahkan mempermudah konsumen dalam membeli jasa atau barang.

Tujuan utama dari *data mining* adalah *knowledge*. *Knowledge* merupakan suatu informasi yang berharga dan dapat dijadikan landasan untuk menganalisa atau membuat kesimpulan. Untuk mendapatkan *knowledge*, dapat dilakukan dengan cara melakukan pencarian *pattern* atau pola yang merupakan salah satu tahap dari *data mining*. Pola inilah yang akan memperlihatkan data manakah yang menarik dan dapat dijadikan *knowledge* yang akan digunakan untuk menganalisa data tersebut.

Pada penelitian *data mining* ini, penulis memiliki data *log* histori KIRI selama 1 bulan. Data tersebut akan diimplementasikan proses *data mining* untuk mendapatkan *pattern* dan *knowledge* yang terkandung pada data *log* KIRI. Data *log* tersebut memiliki 5 *field* untuk setiap *entry* sebagai berikut:

- statisticId, primary key dari entry
- verifier, mengidentifikasikan sumber dari pencarian ini
- *timestamp*, waktu ketika pengguna KIRI mencari rute angkot
- *type*, tipe fungsi yang digunakan
- additionalInfo, mencatat koordinat awal, koordinat akhir, dan banyak rute yang ditemukan pada pencarian ini

Berdasarkan hal diatas, penulis ingin mendapatkan pola yang menarik dan menghasilkan *knowledge* yang berguna dan dapat dipakai baik untuk KIRI ataupun pemerintah.

## 1.2 Perumusan Masalah

Dengan mengacu pada uraian diskripsi diatas, maka permasalahan yang dibahas dan diteliti oleh penulis adalah

- Bagaimana cara mengolah pola yang diperoleh dari *data log histori* KIRI agar pola menjadi menarik dan bermakna?
- Bagaimana membuat perangkat lunak untuk melakukan data mining pada data log history?

## 1.3 Tujuan

Penelitian ini bertujuan untuk

- Mencari pola dan informasi yang menarik dari *log histori* KIRI
- Perangkat lunak dapat melakukan data mining dari *log histori* KIRI

## 1.4 Batasan Masalah

Penelitian *data mining* yang diatas akan ditentukan batasan masalah yang diteliti berupa :

- Penelitian ini dibatasi hanya pada permasalahan pada penerapan *data mining* pada *data log* KIRI
- Data log yang merupakan masukan akan dibatasi sebanyak 10000 buah data

## 1.5 Metode Penelitian

Berikut adalah Metode Penelitian yang digunakan :

- Melakukan studi literatur tentang algoritma-algoritma yang berkaitan dengan pemrosesan *data mining*
- Melakukan penelitian *data mining* yang diterapkan pada *log* KIRI
- Merancang dan mengimplementasikan algoritma untuk *data mining*
- Mengimplementasikan pembangkit pola *data mining*
- Melakukan pengujian dan eksperimen

## 1.6 Sistematika Pembahasan

Sitematika pembahasan dalam penelitian ini adalah: Bab 1: Pendahuluan, berisi latar belakang dari penelitian ini, rumusan masalah yang timbul, tujuan yang ingin dicapain, ruang lingkup atau batasan masalah dari penelitian ini, serta metode penelitian yang akan digunakan dan sistematika pembahasan dari penelitian ini. Bab 2: Landasan Teori, berisi dasar teori mengenai *data mining*

dan *log* histori KIRI Bab 3: Analisa dasar teori yang akan digunakan untuk merancang aplikasi *data mining log* histori KIRI Bab 4: perancangan aplikasi *data mining log* histori KIRI Bab 5: kesimpulan

# BAB 2

# LANDASAN TEORI

## 2.1 *Data Mining*

*Data mining* merupakan merupakan proses yang melakukan pengambilan inti sari atau penggalian *knowledge* dari data yang besar dan merupakan salah satu langkah dari *knowledge discovery*.

Menurut [1], *knowledge discovery* dapat dibagi menjadi 7 tahap (gambar 2.1):

1. *Data cleaning*
2. *Data integration*
3. *Data selection*
4. *Data transformation*
5. *Data mining*
6. *Pattern Evaluation*
7. *Knowledge presentation*

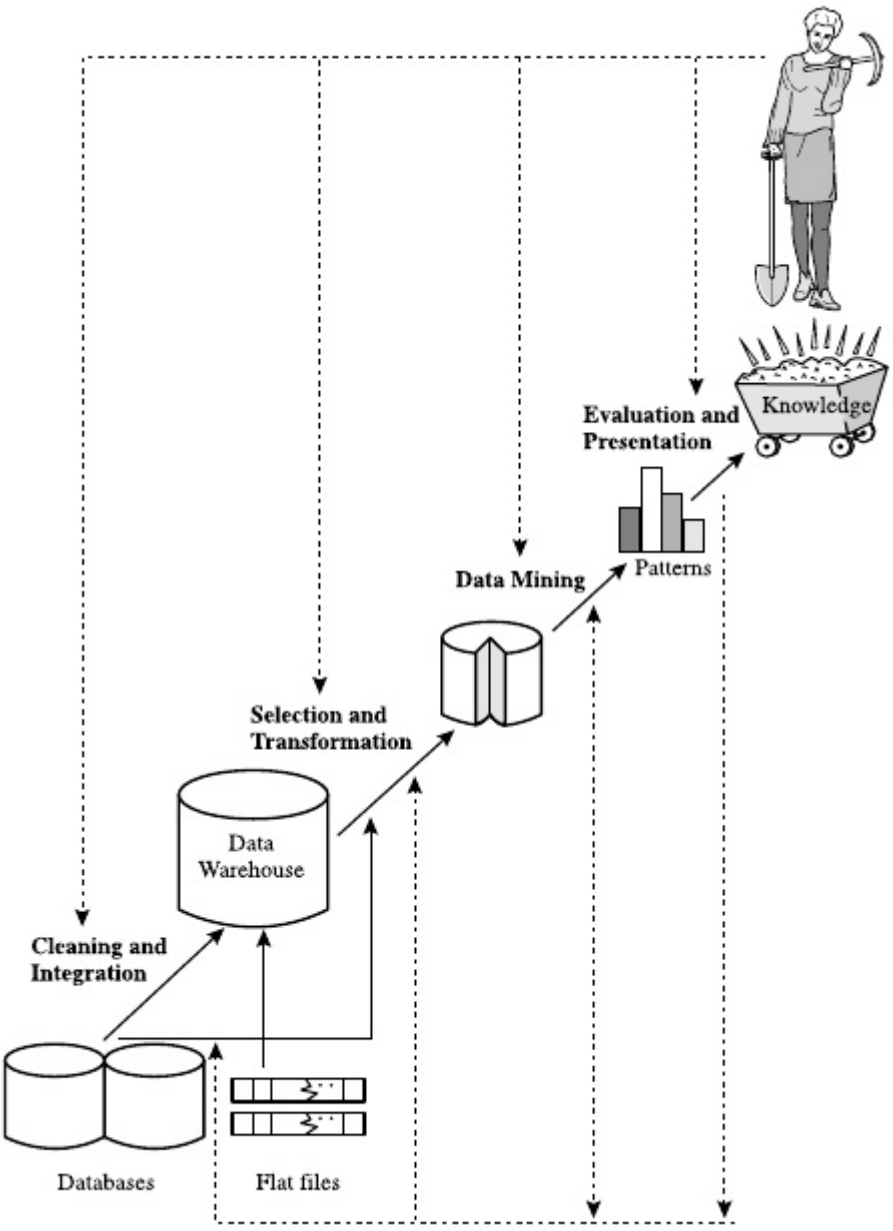
Tahap pertama hingga keempat merupakan bagian dari *data preprocessing*, dimana data-data disiapkan untuk dilakukan penggalian data. Tahap *data mining* merupakan tahap dimana melakukan penggalian data. Tahap keenam merupakan tahap pencarian pola yang merepresentasikan *knowledge*. Sedangkan tahap terakhir merupakan visualisasi dan representasi dari *knowledge* yang sudah diperoleh dari tahap sebelumnya.

### 2.1.1 *Data Cleaning*

*Data cleaning* merupakan tahap *data mining* untuk menghilangkan *missing value* dan *noisy data*. Pada umumnya, *data* yang diperoleh dari *database* terdapat nilai yang tidak sempurna seperti nilai yang hilang, nilai yang tidak valid atau salah ketik. Atribut dari suatu *database* yang tidak relevan atau redudansi bisa diatasi dengan menghapus atribut tersebut.

#### *Missing Values*

*Missing values* akan mengganggu proses *data mining* pada komputer dan dapat menghasilkan nilai akhir yang tidak sesuai. Terdapat beberapa teknik untuk mengatasi *missing values* yaitu




Gambar 2.1: Tahap *Data Mining*, [1]

- Membuang tuple yang mengandung nilai yang hilang
- Mengisi nilai yang hilang secara manual
- Mengisi nilai yang hilang dengan menggunakan nilai konstan yang bersifat umum
- Menggunakan nilai rata-rata dari suatu atribut untuk mengisi nilai yang hilang

#### *Noisy Data*

*Noisy data* merupakan nilai yang berasal dari error atau tidak valid. *Noisy data* dapat dihilangkan dengan menggunakan teknik *smoothing*. Terdapat 3 metode untuk menghilangkan *noisy data* yaitu

- *Binning*, merupakan metode pengisian data sesuai dengan proses yang dilakukan pada data tersebut
- *Regression*, merupakan metode yang mencari detail persamaan atribut untuk memprediksikan suatu nilai
- *Clustering*, merupakan metode pengelompokan dimana ditemukan *outliers* yang dapat dibuang

### 2.1.2 *Data Integration*

*Data integration* merupakan tahap menggabungkan data dari berbagai sumber. Sumber tersebut bisa termasuk beberapa *database*, *data cubes*, atau bahkan *flat data*. *Data cube* merupakan teknik pengambilan data-data dari *data warehouse* dan dilakukan operasi agregasi sesuai dengan kondisi tertentu (contoh, penjumlahan total penjualan per tahun dari 2005-2010). Sedangkan *flat data* merupakan data yang disimpan dengan cara apapun untuk merepresentasikan database model pada sebuah data baik berbentuk *plain text file* maupun *binary file*.

Tahap ini harus dilakukan secara teliti terutama ketika dalam memasangkan nilai-nilai yang berasal dari sumber yang berbeda. Pada tahap ini, perlu dilakukan identifikasi data apakah data tersebut dapat diturunkan atau tidak agar data yang diperoleh tidak terlalu besar. *Data integration* yang baik merupakan integrasi yang dapat memaksimalkan kecepatan dan meningkatkan akurasi dari proses *data mining*. Contoh studi kasus dari *data integration*, jika suatu perusahaan sepatu A memiliki dua pabrik dengan *database* lokal pada masing-masing pabrik, jika akan dilakukan *data mining* pada kedua *database* tersebut, maka kedua *database* akan digabung dan perlu diperhatikan serta diperbaiki nilai-nilai seperti *primary key*, atribut, dan lain-lain agar tidak terjadi *error* pada *database* yang sudah digabung. Proses dari penggabungan hingga perbaikan nilai-nilai pada kedua database tersebut adalah proses *data integration*.

### 2.1.3 *Data Selection*

Proses dimana data-data yang relevan dengan analisis akan diambil dari database dan data yang tidak relevan akan dibuang. Sebagai contoh kasus, jika akan dilakukan analisa mengenai nilai mahasiswa dalam satu semester, atribut pada tabel nilai sebagai berikut

- NPMMahasiswa

- NamaMahasiswa
- JenisKelamin
- Alamat
- MataKuliah
- NilaiART
- NilaiUTS
- NilaiUAS

Maka, atribut yang berpotensi diambil adalah MataKuliah, NilaiART, NilaiUTS, NilaiUAS, sedangkan atribut yang akan dibuang adalah NPMMahasiswa, NamaMahasiswa JenisKelamin, dan Alamat karena tidak terlalu berhubungan dengan analisa.

### 2.1.4 *Data Transformation*

*Data transformation* merupakan tahap pengubahan data agar siap dilakukan proses *data mining*. *Data transformation* bisa melibatkan:

- *Smoothing*, proses untuk membuang *noise* seperti yang dilakukan pada tahap *data cleaning*
- *Aggregation*, proses mengganti nilai-nilai menjadi suatu nilai yang dapat mewakili nilai sebelumnya
- *Generalization*, proses dimana membuat suatu nilai yang bersifat khusus menjadi nilai yang bersifat umum
- *Normalization*, proses dimana suatu nilai dapat diubah skalanya menjadi nilai yang lebih kecil dan spesifik
- *Attribute construction*, proses membuat atribut baru yang berasal dari beberapa atribut untuk membantu proses data mining

#### *Smoothing*

*Smoothing* merupakan bagian dari *data cleaning* untuk menghilangkan *noise* pada database. Teknik dari *smoothing* adalah *binning*, *regression*, dan *clustering*. Penjelasan teknik *smoothing* dapat dilihat pada 2.1.1, bagian *noisy data*.

#### *Aggregation*

*Aggregation*, dimana suatu kesimpulan atau hasil dari *aggregation operation* yang disimpan dalam database. Contoh studi kasus, jika terdapat suatu database dari toko A, kita dapat menggunakan operasi *aggregation* untuk mencari total pendapatan dengan rentang hari tertentu.

#### *Generalization*

*generalization*, dimana suatu data yang memiliki nilai *primitive* atau *low level* diubah menjadi *high level* dengan menggunakan konsep hirarki. Contoh studi kasus, nilai pada atribut umur dapat dikelompokkan menjadi muda, dewasa, tua.

#### *Normalization*

Atribut dapat dinormalisasi dengan memberi skala pada nilainya sehingga nilai tersebut menjadi suatu range yang lebih spesifik dan kecil seperti 0,0 sampai 1,0. Dua teknik nnormalisasi yaitu, *min-max normalization* dan *z-score normalization*. *Min-max normalization* akan mengubah semua nilai menjadi nilai dengan skala tertentu. Dengan menggunakan rumus

$$\nu' = \frac{\nu - min_A}{max_A - min_A}(newMax_A - newMin_A) + newMin_A$$

Contoh kasus, misalkan nilai minimun dan maximum dari suatu pendapatan adalah 12.000 dan 98.000, akan diubah menjadi berskala antara 0,0 sampai 1,0. Jika ada nilai pendapat yang baru, yaitu 73.600, maka akan menjadi

$$\frac{73.600 - 12.000}{98.000 - 12.000}(1,0 - 0) + 0 = 0,716$$

*z-score normalization* merupakan normalisasi berdasarkan nilai rata-rata dan standar deviasi dari nilai-nilai atribut dengan cara

$$\nu' = \frac{\nu - \overline{A}}{\sigma_A}$$

Contoh kasus, misal nilai rata-rata dan standar deviasi dari nilai-nilai atribut pendapatan adalah 54.000 dan 16.000. Dengan *z-score*, jika ada nilai pendapatan baru yaitu 73600, maka akan diubah menjadi

$$\frac{73.600 - 54.000}{16.000} = 1,225$$

#### *Attribute Construction*

*Attribute Construction* merupakan teknik menambahkan atribut baru yang berdasarkan dari atribut yang sudah ada guna menambah akurasi. Contoh kasus, dibuat atribut baru bernama area berdasarkan atribut panjang dan lebar.

### 2.1.5 *Data Mining*

#### *Classification and Prediction*

*Classification* merupakan pemodelan yang dibangun untuk memprediksikan label kategori, seperti "‘baik"’, "‘cukup"’, dan "‘buruk"’ dalam sistem penilaian sikap seorang siswa atau "‘mini bus"’, "‘bus"’, atau "‘sedan"’ dalam kategori tipe mobil. Kategori tersebut dapat direpresentasikan dengan menggunakan nilai diskret. Nilai diskret merupakan nilai yang terpisah dan berbeda, seperti 1 atau 5. Kategori yang direpresentasikan oleh nilai diskret maka akan menjadi nilai yang terurut dan

tidak memiliki arti, seperti 1,2,3 untuk merepresentasikan kategori tipe mobil "'mini bus"', "'bus"', dan "'sedan"'.

*Prediction* merupakan model yang dibangun untuk meramalkan fungsi nilai kontinu atau *ordered value*. *Ordered value* merupakan nilai yang terurut dan berlanjut. Contoh studi kasus untuk pemodelan prediction adalah seorang marketing ingin meramalkan seberapa banyak konsumen yang akan belanja di sebuah toko dalam waktu satu bulan. Pemodelan tersebut disebut *predictor*. *Regression Analysis*, merupakan metodologi statistik yang digunakan untuk *numeric prediction*. *Classification* dan *numeric prediction* merupakan dua jenis utama dalam masalah prediksi.

*Data Classification* merupakan proses untuk melakukan klasifikasi. *Data classification* memiliki dua tahap proses, yaitu *learning step* dan tahap klasifikasi seperti pada ilustrasi di gambar 2.2. *Learning step* merupakan langkah pembelajaran, di mana algoritma klasifikasi membangun *classification rules* (yang berisi syarat atau aturan sebuah nilai masuk ke dalam kategori tertentu) dengan cara menganalisis *training set* yang merupakan *database tuple*. Karena pembuatan *classification rules* menggunakan *training set*, yang dikenal juga sebagai *supervised learning*. Pada tahap kedua, dilakukan proses klasifikasi nilai berdasarkan *classification rules* yang sudah dibangun dari tahap pertama.

### *Decision Tree*

Salah satu cara pembuatan *classification rules* pada *Data Classification* adalah dengan membuat *decision tree* (pohon keputusan). *Decision tree* merupakan *flowchart* yang berbentuk pohon, dimana setiap node internal (*nonleaf* node) merupakan hasil test dari atribut, setiap cabang merepresentasikan output dari test, dan setiap node daun memiliki *class label*. Bagian paling atas dari pohon disebut *root node*. Contoh studi kasus, pohon keputusan untuk menentukan apakah seorang konsumen akan membeli komputer atau tidak (ilustrasi pohon keputusan pada gambar 2.3)

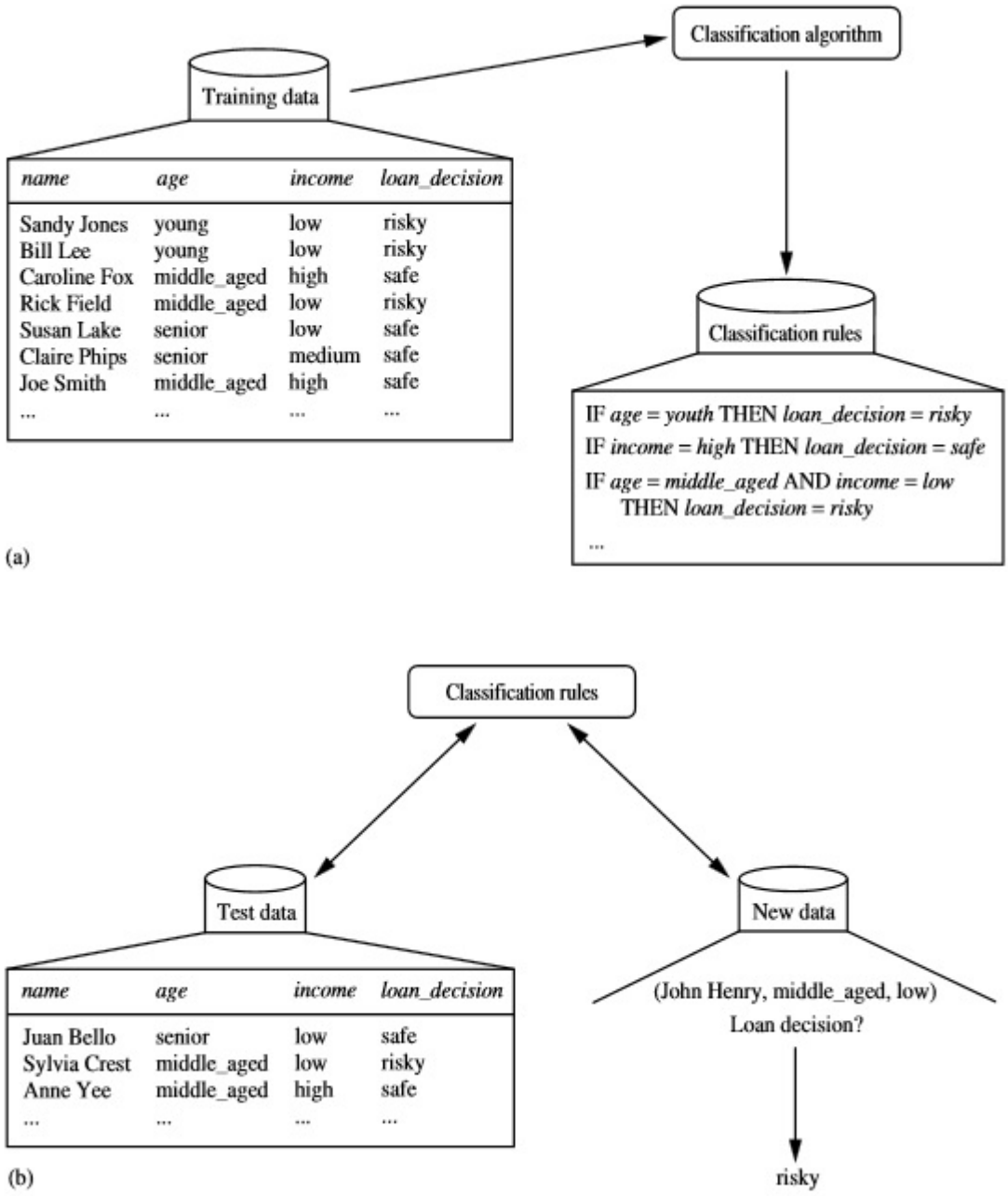
### *Decision Tree Induction*

*Decision tree induction* merupakan pelatihan pohon keputusan dari tupel pelatihan kelas berlabel. Terdapat tiga teknik untuk membuat *decission tree* diantaranya adalah ID3 dan C4.5. Teknik tersebut menggunakan pendekatan *greedy* yang merupakan *decission tree* yang dibangun secara *top-down recursive divide and conquer*. Algoritma yang diperlukan secara umum sama, hanya berbeda pada *attribute_selection_method*. Berikut algoritma untuk membuat pohon keputusan dari suatu tupel pelatihan.

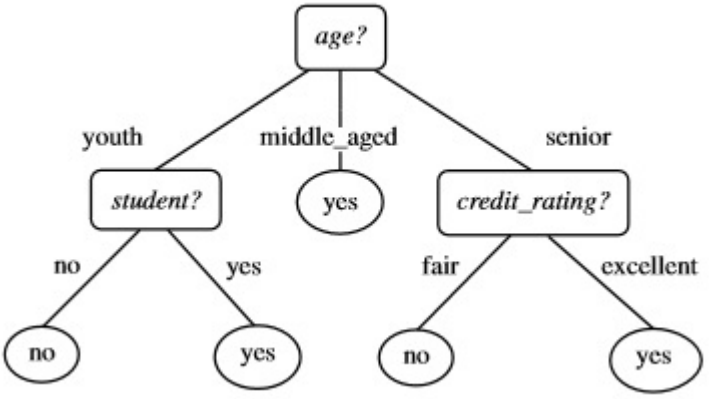
Input:

- Partisi data, D, merupakan set data pelatihan dan kelas label
- *attribute_list*, merupakan set dari atribut kandidat
- *Attribute_selection_method*, prosedur untuk menentukan *splitting criterion*. Pada input ini, terdapat juga data *splitting_attribute* dan mungkin salah satu dari *split point* atau *splitting subset*

Output: pohon keputusan
Method:




Gambar 2.2: Tahap *data classification*, [1]




Gambar 2.3: Contoh *decision tree*, [1]

(1) create a node N;

(2) if tuples in D are all of the same class, C then

(3) return N as a leaf node labeled with the class C;

(4) if attribute_list is empty then

(5) return N as leaf node labeled with the majority class in D; //majority voting

(6) apply Attribute_selection_method(D, atribute_list) to find the "'best"' splitting_criterion;

(7) label node N with splitting_criterion;

(8) if splitting_attribute is discrete valued and multiway splits allowed then //not restricted to binary trees

(9) attribute_list <- attribute_list - splitting_attribute; //remove splitting_attribute

(10)for each outcome j of splitting_criterion // partition the tuples and from subtrees for each partition

(11)let Dj be the set of data tuples in D satisfying outcome j; //a partition

(12) if Dj is empty then

(13) attach a leaf labeled with the majority class in D to node N;

(14) else attach the node returned by generate_decision_tree(Dj, attribute_list) to node N;

endfor

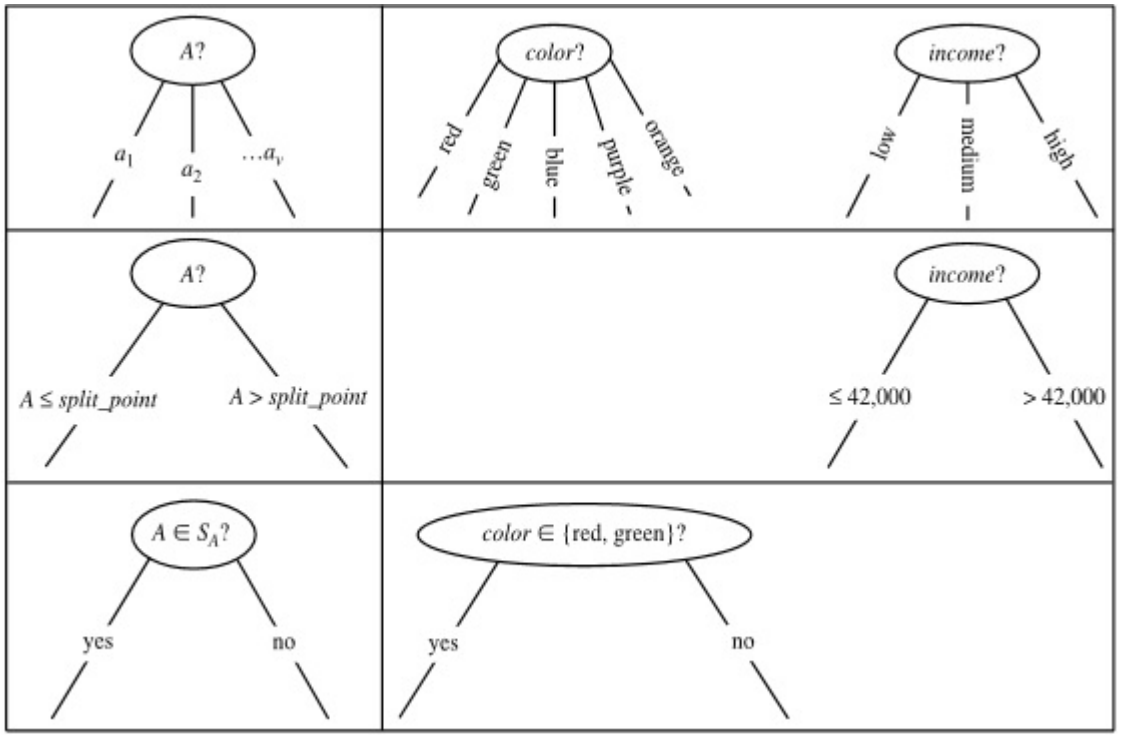
(15) return N;

Pohon keputusan akan dimulai dengan satu node, yaitu N, merepresentasikan tuple pelatihan pada D (langkah 1)

Jika tuple di D memiliki kelas yang sama semua, maka node N akan menjadi daun dan diberi label dari kelas tersebut (langkah 2 dan 3). Perlu diketahui bahwa langkah 4 dan 5 akan mengakhiri kondisi.

Jika tuple di D ada kelas yang berbeda, maka algoritma akan memanggil *attribute_selection_method* untuk menentukan *splitting criterion*. *Splitting criterion* akan menentukan atribut pada node N yang merupakan nilai terbaik untuk memecah nilai atribut pada tuple ke dalam kelas masing-masing. (langkah 6)

Node N akan diisi dengan hasil dari *splitting criterion* (langkah 7). Kemudian kriteria tersebut agak dibentuk cabangnya masing-masing sesuai pada langkah 10 dan 11. Terdapat tiga kemungkinan bentuk kriteria jika A merupakan *splitting_attribute* yang memiliki nilai unik seperti {a1,




Gambar 2.4: Jenis-jenis *split point*, [1]

a2, ..., av} seperti pada gambar 2.4, yaitu,

1. *Discrete valued*: cabang yang dihasilkan memiliki kelas dengan nilai diskret. Karena kelas yang dihasilkan diskret dan hanya memiliki nilai yang sama pada cabang tersebut, maka *attribut_list* akan dihapus (langkah 8 dan 9)

2. *Continuous values*: cabang yang dihasilkan memiliki jarak nilai untuk memenuhi suatu kondisi (contoh: A <= split_point), dimana nilai *split_point* adalah nilai pembagi yang dikembalikan oleh *attribute_selection_method*

3. *Dicrete valued and a binary tree*: cabang yang dihasilkan adalah dua berupa nilai iya atau tidak dari "'apakah A anggota Sa"', dimana Sa merupakan subset dari A, yang dikembalikan oleh *Attribute_selection_method*

Kemudian, akan dipanggil kembali algoritma *decision tree* untuk setiap nilai hasil pembagian pada tuple, Dj (langkah 14).

Rekursif tersebut akan berhenti ketika salah satu dari kondisi terpenuhi, yaitu

1. Semua tuple pada partisi D merupakan bagian dari kelas yang sama.

2. Sudah tidak ada atribut yang dapat dilakukan pembagian lagi (dilakukan pengecekan pada langkah 4). Disini, akan dilakukan *majority voting* (langkah 5) yang akan mengkonversi node N menjadi *leaf* dan diberi label dengan kelas yang terbanyak pada D.

3. Sudah tidak ada tuple yang dapat diberi cabang, Dj sudah kosong (langkah 12) dan *leaf* akan dibuat dengan *majority class* pada D (langkah 13).

Pada langkah 15, akan dikembalikan nilai *decision tree* yang telah dibuat.

subsubsection*Attribute Selection Measure*

*Attribute Selection Measure* merupakan suatu hirarki untuk pemilihan *splitting criterion* yang terbaik yang memisah partisi data (D), tuple pelatihan kelas label ke dalam kelas masing-masing. *Attribute Selection Measure* menyediakan peringkat untuk setiap atribut pada training tuple. Jika *splitting criterion* merupakan nilai *continous* atau *binary trees*, maka nilai *split point* dan *splitting subset* harus ditentukan sebagai bagian dari *splitting criterion*. Contoh dari *attribute selection measure* adalah *information gain*, *gain ratio*, dan *gini index*.

Notasi yang digunakan adalah sebagai berikut. D merupakan data partisi, set pelatihan dari *class-labeled* tuple. Jika label kelas atribut memiliki m nilai yang berbeda yang mendifinisikan m kelas yang berbeda, Ci (for i=1,...,m). Ci,d menjadi kelas tuple dari Ci di D. |D| dan |Ci,d| merupakan banyak tuple pada D dan Ci,d.

### *Information Gain*

*Information* menurut Claude Shannon dalam *information theory* adalah ukuran *pure* dari suatu data. ID3 menggunakan *information gain* sebagai *attribute selection measure* yang melakukan pemilihan atribut berdasarkan informasi yang terkandung dalam pesan. Cara ID3 mendapatkan *information gain* dengan menggunakan *entropy*. *Entropy* adalah ukuran *impurity* dari suatu data. Cara mendapatkan nilai *entropy* adalah

$$Info(D) = -\sum_{i=1}^{m} pi \log_2(pi)$$

Dimana pi merupakan probabilitas tuple pada D terhadap class Ci, dapat diperoleh dengan |ci,d|/|D|. Info(D) merupakan nilai rata-rata *entropy* dari suatu label kelas pada tuple D. Untuk mengetahui atribut mana yang paling baik untuk dijadikan *splitting attribute*, adalah dengan cara menghitung nilai *entrophy* dari suatu atribut kemudian diselisihkan dengan nilai *entropy* dari D. Jika pada tuple D, memiliki atribut A dengan v nilai yang berbeda, maka menghitung *entropy* dari suatu atribut adalah

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

|Dj|/D merupakan angka yang menghitung bobot dari suatu partisi. Semakin kecil nilai dari InfoA(Dj), maka atribut tersebut masih memerlukan informasi, semakin besar nilai InfoA(Dj), semakin tinggi pula tingkat *pure* dari suatu partisi.

Setelah mendapatkan nilai Info(D) dan InfoA(Dj), *information gain* dapat diperoleh dari selisih nilai Info(D) dan InfoA(Dj)

$$Gain(A) = Info(D) - Info_A(D)$$

contoh kasus untuk ID3, dalam pencarian*information gain*

Pada tabel 2.1, terdapat *training set*, D. Atribut kelas label merupakan dua nilai yang berbeda yaitu *yes* atau *no*, maka dari itu, nilai m = 2. C1 diisi dengan kelas label bernilai *yes*, sedangkan

Tabel 2.1: Contoh training set

| RID | age | income | student | credit_rating | Class: buys_computer |
|---|---|---|---|---|---|
| 1 | youth | high | no | fair | no |
| 2 | youth | high | no | excellent | no |
| 3 | middle_aged | high | no | fair | yes |
| 4 | senior | medium | no | fair | yes |
| 5 | senior | low | yes | fair | yes |
| 6 | senior | low | yes | excellent | no |
| 7 | middle_aged | low | yes | excellent | yes |
| 8 | youth | medium | no | fair | no |
| 9 | youth | low | yes | fair | yes |
| 10 | senior | medium | yes | fair | yes |
| 11 | youth | medium | yes | excellent | yes |
| 12 | middle_aged | medium | no | excellent | yes |
| 13 | middle_aged | high | yes | fair | yes |
| 14 | senior | medium | no | excellent | no |

C2 diisi dengan kelas label bernilai *no*. Terdapat sembilan tuple atribut kelas label dengan nilai *yes* dan lima tuple dengan nilai *no*. Untuk dapat menentukan *splitting criterion*, *information gain* harus dihitung untuk setiap atribut terlebih dahulu. Perhitungan *entropy* untuk D adalah

$$Info(D) = -\frac{9}{14}\log 2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940bits$$

Setelah diperoleh nilai *entropy* dari D, kemudian akan dihitung nilai *entropy* atribut dimulai dari atribut age. Pada kategori *youth*, terdapat dua tuple dengan kelas *yes* dan tiga tuple dengan kelas *no*. Untuk kategori *middle_ age*, terdapat empat tuple dengan kelas *yes* dan nol tuple dengan kelas *no*. Pada kategori *senior*, terdapat tiga dengan kelas *yes* dan dua dengan kelas *no*. Perhitungan nilai *entropy* atribut *age* terhadap D sebagai berikut

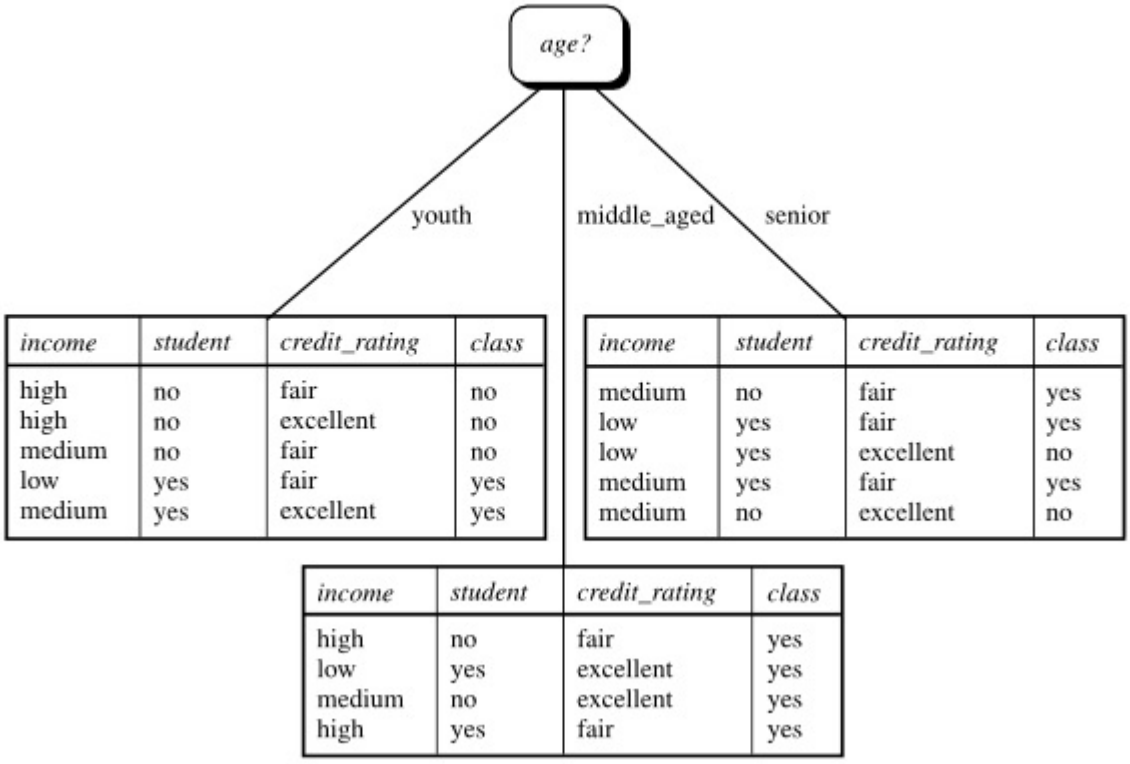
$$Info_{age}(D) = \frac{5}{14}\times(-\frac{2}{5}\log_2\frac{2}{5}-\frac{3}{5}\log_2\frac{3}{5})+\frac{4}{14}\times(-\frac{4}{4}\log_2\frac{4}{4}-\frac{0}{4}\log_2\frac{0}{4})+\frac{5}{14}\times(-\frac{3}{5}\log_2\frac{3}{5}-\frac{2}{5}\log_2\frac{2}{5}) = 0.694bits$$

Setelah mendapatkan *entropy* dari atribut *age*, maka nilai *gain information* dari atribut *age* adalah

$$Gain_{(age)} = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246bits$$

Dengan melakukan hal yang sama, dapat diperoleh nilai *gain* untuk atribut *income* adalah 0.029 *bits*, untuk nilai *gain(student)* adalah 0.151 *bits*, dan *gain(credit_rating)* = 0.048 *bits*. Karena nilai *gain* dari atribut *age* merupakan nilai terbesar diantara semua atribut, maka atribut *age* dipilih menjadi *splitting attribute*. Setelah ditentukan, node N akan membentuk cabang berdasarkan nilai dari atribut *age* seperti pada gambar 2.5.

Untuk atribut yang merupakan nilai *continuous*, harus dicari nilai *split point* untuk A. Nilai-nilai dari dua angka yang bersebelahan dapat diambil nilai tengahnya untuk dijadikan *split-point*. Jika terdapat v nilai yang berbeda dari A, maka akan terdapat v-1 kemungkinan *split point*. Kemudian nilai *split point* akan dijadikan sebagai nilai pembagi, sebagai contoh: A <= *split-point* merupakan




| *income* | *student* | *credit_rating* | *class* |
|---|---|---|---|
| high | no | fair | no |
| high | no | excellent | no |
| medium | no | fair | no |
| low | yes | fair | yes |
| medium | yes | excellent | yes |

| *income* | *student* | *credit_rating* | *class* |
|---|---|---|---|
| medium | no | fair | yes |
| low | yes | fair | yes |
| low | yes | excellent | no |
| medium | yes | fair | yes |
| medium | no | excellent | no |

| *income* | *student* | *credit_rating* | *class* |
|---|---|---|---|
| high | no | fair | yes |
| low | yes | excellent | yes |
| medium | no | excellent | yes |
| high | yes | fair | yes |

Gambar 2.5: Hasil cabang dari atribut *age*, [1]

cabang pertama, dan A > *split-point* merupakan cabang kedua.

### *Gain Ratio*

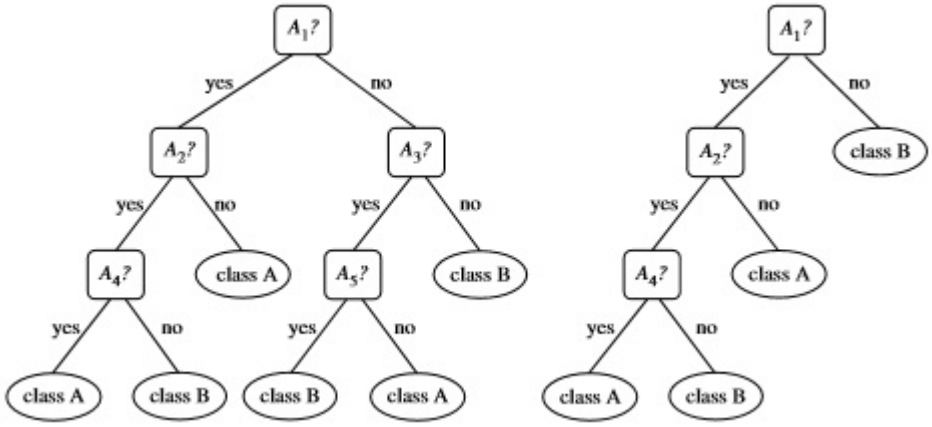
*Information gain* akan memiliki nilai yang baik jika suatu atribut memiliki banyak nilai yang berbeda, namun hal itu tidak selalu bagus. Sebagai contoh kasus, jika nilai id suatu table yang memiliki nilai unik, maka akan terdapat banyak sekali cabang. Namun setiap cabang hanya akan berisi satu tuple dan bersifat *pure*, maka nilai *entropy* yang dihasilkan adalah 0. Oleh karena itu, informasi yang diperoleh pada atribut ini akan bernilai maksimum namun tidak akan berguna untuk *classification* [1].

C4.5, menggunakan nilai tambahan dari *information gain* yaitu *gain ratio*, yang dapat mengatasi permasalahan *information gain* tentang nilai yang banyak namun tidak baik untuk *classification*. C4.5 melakukan teknik normalisasi terhadap *gain information* dengan menggunakan *split information* yang memiliki rumus sebagai berikut:

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2(\frac{|D_j|}{|D|})$$

Setelah mendapatkan nilai *split info* dari suatu atribut, dapat diperoleh nilai *gain ratio* dengan rumus sebagai berikut:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$




Gambar 2.6: `Decision tree` yang belum dipotong dan yang sudah dipotong, [1]

Nilai dari *gain ratio* terbesar yang akan dipilih. Perlu diketahui [1] jika nilai hasil mendekati 0, maka rasio menjadi tidak stabil, oleh karena itu, *gain information* yang dipilih harus besar, minimal sama besarnya dengan nilai rata-rata dari semua test yang diperiksa.

Contoh studi kasus, akan dilakukan perhitungan *gain ratio* dengan menggunakan training set pada tabel 2.1. Dapat dilihat pada atribut *income* memiliki tiga partisi yaitu *low*, *medium*, dan *high*. Terdapat empat tuple dengan nilai *low*, enam tuple dengan nilai *medium*, dan empat tuple dengan nilai *high*. Untuk menghitung *gain ratio*, perlu dihitung nilai *split information* terlebih dahulu dengan cara:

$$SplitInfo_A(D) = -\frac{4}{14} \times \log_2(\frac{4}{14}) - \frac{6}{14} \times \log_2(\frac{6}{14}) - \frac{4}{14} \times \log_2(\frac{4}{14}) SplitInfo_A(D) = 0.926 bits$$

Jika nilai *gain information* dari *income* adalah 0.029, maka, dapat diperoleh *gain rasio* dari *income* adalah

$$GainRasio(D) = \frac{0.029}{0.926} = 0.031 bits$$

### Tree Pruning

*Tree pruning* merupakan proses pemotongan *decision tree* agar lebih efisien dan tidak terlalu mempengaruhi nilai keputusan yang dihasilkan. *decision tree* yang sudah dipotong akan lebih kecil ukuran pohonnya, tidak serumit dengan pohon yang asli, namun lebih mudah untuk diproses. *Decision tree* yang sudah dipotong memiliki kecepatan serta ketepatan mengklasifikasikan yang lebih baik [1]. Perbedaan *decision tree* yang sudah dipotong dan belum dapat dilihat pada gambar 2.6.

Terdapat dua pendekatan dalam melakukan *pruning*, yaitu *prepruning* dan *postpruning*.

Pada *prepruning*, pemotongan pohon dilakukan dengan cara menahan dan tidak melanjutkan pembuatan cabang atau partisi dari sebuah node, dan membuat node tersebut menjadi *leaf*.

Pada *postpruning*, pemotongan pohon dilakukan ketika *decision tree* sudah selesai dibangun dengan cara menggubah cabang pohon menjadi *leaf*.

### 2.1.6 *Pattern Evaluation*

*Pattern evaluation* merupakan tahap mengidentifikasi apakah *pattern* atau pola tersebut menarik dan merepresentasikan *knowledge* berdasarkan beberapa *interestingness measures*. Suatu *pattern* atau pola dapat dinyatakan menarik apabila

- mudah dimengerti oleh manusia
- valid untuk data percobaan maupun data yang baru
- memiliki potensi atau berguna
- merepresentasikan *knowledge*

### 2.1.7 *Knowledge Presentation*

*Knowledge presentation* merupakan tahap representasi dan visualisasi terhadap *knowledge* yang merupakan hasil dari *knowledge discovery*.

## 2.2 Log Histori KIRI

KIRI memiliki log histori yang melakukan pencatatan untuk setiap user ketika menggunakan KIRI. *Log* tersebut memiliki 5 *field* untuk setiap *entry* sebagai berikut:

- logId, primary key dari entry
- APIKey, mengidentifikasikan sumber dari pencarian ini
- *Timestamp* (UTC), waktu ketika pengguna KIRI mencari rute angkot menggunakan waktu UTC / GMT
- *Action*, tipe dari log yang dibuat.
- AdditionalData, mencatat data-data yang berhubungan sesuai dengan nilai atribut*action*

LogId merupakan *field* dengan tipe data int dengan batas 6 karakter yang digunakan sebagai *primary key* dari tabel tersebut. LogId diisi dengan menggunakan fungsi *increment integer*. *Increment integer* merupakan fungsi untuk pengisian data pada database dengan menambahkan nilai 1 dari nilai yang terakhir kali diisi. APIKey merupakan *field* dengan tipe data varchar yang digunakan untuk memeriksa pengguna KIRI ketika menggunakan KIRI. *Timestamp* (UTC) merupakan *field* dengan tipe data *timestamp* yang digunakan untuk mencatat waktu penggunaan KIRI oleh user, diisi dengan menggunakan fungsi *current time*. *Current time* merupakan fungsi untuk pengisian data pada database dengan mengambil waktu pada komputer ketika record dibuat. *Action* merupakan *field* dengan tipe data varchar yang digunakan untuk memeriksa fungsi apa yang dipanggil dari API KIRI. Terdapat beberapa tipe pada *field* ini, yaitu /

- *ADDAPIKEY*, *action* yang dicatat ke dalam log ketika fungsi pembuatan *API key* yang baru dipanggil.
- *FINDROUTE*, *action* yang dicatat ketika user melakukan pencarian rute

- *LOGIN*, *action* yang dicatat ketika developers melakukan login dengan menggunakan *API key*
- *NEARBYTRANSPORT*, *action* yang dicatat ketika user mencari transportasi di daerah rute sedang dicari
- *PAGELOAD*, *action* yang dicatat ketika user memasuki halaman KIRI
- *REGISTER*,*action* yang dicatat ketika developers melakukan pendaftaran pada KIRI *API key*
- *SEARCHPLACE*, *action* yang dicatat ketika user memanggil fungsi pencarian lokasi dengan menggunakan nama tempat
- *WIDGETERROR*, mencatat log tersebut ketika user menerima error dari *widget*
- *WIDGETLOAD*, mencatat log tersebut ketika user mengdownload widget

AdditionalData, merupakan *field* dengan tipe data varchar yang digunakan untuk mencatat informasi yang dibutuhkan sesuai dengan *field action*.

# BAB 3

# ANALISA

Pada bab ini, akan dilakukan analisa terhadap data yang akan diproses menggunakan *data mining* dan perangkat lunak yang akan dibuat untuk melakukan proses data tersebut.

## 3.1 Analisis Data

Pada bab ini, akan dilakukan analisa *preprocessing data* yang meliputi *data selection* dan *data transformation*. Tahap *data cleaning* dan *data integration* dapat dilewat karena hingga saat ini, database yang digunakan belum ditemukan *noise* atau *missing value*, dan hanya satu tabel (yaitu tabel *statistics*) pada database KIRI yang digunakan pada penelitian ini.

### 3.1.1 *Data Selection*

Pada penelitian ini, akan dilakukan proses *data mining* mengenai lokasi keberangkatan dan tujuan dari seorang user yang menggunakan aplikasi KIRI. Oleh karena itu, pada atribut *action*, nilai yang akan dipilih hanya *FINDROUTE*. Karena seluruh *action* bernilai satu jenis yaitu *FINDROUTE*, maka atribut tersebut dapat dihilangkan. Selain itu, atribut logId dan APIKey tidak akan dimasukan ke dalam proses karena tidak memiliki hubungan dengan lokasi keberangkatan dan tujuan dari seorang user.

Dari analisis diatas, maka atribut yang dipilih untuk diproses ke dalam *data mining* adalah

- *Timestamp* (UTC)
- *AdditionalData*

Berikut contoh data dari atribut tersebut dapat dilihat pada tabel 3.1

Tabel 3.1: Contoh data *log* KIRI setelah *data selection*

| **Timestamp (UTC)** | **AdditionalData** |
|---|---|
| 2/1/2014 0:11 | -6.8972513,107.6385574/-6.91358,107.62718/1 |
| 2/1/2014,0:13 | -6.8972513,107.6385574/-6.91358,107.62718/1 |
| 2/1/2014 0:16 | -6.90598,107.59714/-6.90855,107.61082/1 |
| 2/1/2014 0:18 | -6.9015366,107.5414474/-6.88574,107.53816/1 |
| 2/1/2014 0:25 | -6.90608,107.61530/-6.89140,107.61060/2 |
| 2/1/2014 0:27 | -6.89459,107.58818/-6.89876,107.60886/2 |
| 2/1/2014 0:28 | -6.89459,107.58818/-6.86031,107.61287/2 |

Pada atribut *additionalData*, jika nilai atribut *action* adalah *FINDROUTE*, maka nilai *addtional data* memiliki tiga bagian yang dibatasi dengan '/'. Ketiga bagian tersebut adalah

1. Nilai latitude dan longitude dari lokasi keberangkatan yang dipilih oleh user
2. Nilai latitude dan longitude dari lokasi tujuan yang dipilih oleh user
3. Nilai yang menunjukkan banyak jalur yang dihasilkan oleh sistem KIRI

### 3.1.2 *Data Transformation*

Pada atribut yang dipilih, nilai dari atribut *timestamp* dan *additionaldata* perlu dilakukan transformasi agar program dapat membaca dan memproses data lebih cepat.

Pada atribut *timestamp*, data akan diubah menjadi tiga atribut, yaitu:

- Tanggal, atribut ini akan menunjukkan tanggal ketika user KIRI memanggil *action FINDROUTE*
- Hari, atribut ini akan menunjukkan hari ketika user KIRI memanggil *action FINDROUTE*
- Jam, atribut ini akan menunjukkan jam ketika user KIRI memanggil *action FINDROUTE*

Pada atribut *additionalData*, data akan diubah menjadi tiga atribut, yaitu:

- Keberangkatan, atribut ini berisi nilai latitude dan longitude dari lokasi keberangkatan yang dipilih oleh user
- Tujuan, atribut ini berisi Nilai latitude dan longitude dari lokasi tujuan yang dipilih oleh user
- Jalur, atribut ini berisi Nilai yang menunjukkan banyak jalur yang dihasilkan oleh sistem KIRI

Dari analisis diatas, banyak atribut dari tabel *statistics* akan menjadi enam, yaitu:

- Tanggal
- Hari
- Jam
- Keberangkatan
- Tujuan
- Jalur

Contoh hasil data transformasi jika input merupakan data dari tabel 3.1 dapat dilihat pada tabel 3.2.

## 3.2 Analisis Perangkat Lunak

Tabel 3.2: Contoh hasil data transformasi

| **Tanggal** | **Hari** | **Jam** | **Keberangkatan** | **Tujuan** | **Jalur** |
|---|---|---|---|---|---|
| 2/1/2014 | Sabtu | 0:11 | -6.8972513,107.6185574 | -6.91358,107.62718 | 1 |
| 2/1/2014 | Sabtu | 0:13 | -6.8972513,107.6385574 | -6.91358,107.62718 | 1 |
| 2/1/2014 | Sabtu | 0:16 | -6.90598,107.59714 | -6.90855,107.61082 | 1 |
| 2/1/2014 | Sabtu | 0:18 | -6.9015366,107.5414474 | -6.88574,107.53816 | 1 |
| 2/1/2014 | Sabtu | 0:25 | -6.90608,107.61530 | -6.89140,107.61060 | 2 |
| 2/1/2014 | Sabtu | 0:27 | -6.89459,107.58818 | -6.89876,107.60886 | 2 |
| 2/1/2014 | Sabtu | 0:28 | -6.89459,107.58818 | -6.86031,107.61287 | 2 |

# DAFTAR REFERENSI

# LAMPIRAN A

## 100 DATA PERTAMA DARI *LOG* HISTORI KIRI

| **LogId** | **APIKey** | **Timestamp (UTC)** | **Action** | **AddionalData** |
|---|---|---|---|---|
| 113909 | E5D9904F0A8B4F99 | 2/1/2014 0:07 | PAGELOAD | /5.10.83.30/ |
| 113910 | E5D9904F0A8B4F99 | 2/1/2014/ 0:07 | PAGELOAD | /5.5.83.49/ |
| 113911 | E5D9904F0A8B4F99 | 2/1/2014/ 0:09 | PAGELOAD | /5.10.83.30/ |
| 113912 | E5D9904F0A8B4F99 | 2/1/2014 0:10 | PAGELOAD | /5.10.83.88/ |
| 113913 | E5D9904F0A8B4F99 | 2/1/2014 0:10 | PAGELOAD | /5.10.83.58/ |
| 113914 | A44EB361A179A49E | 2/1/2014 0:11 | SEARCHPLACE | taman+fot/10 |
| 113915 | A44EB361A179A49E | 2/1/2014 0:11 | FINDROUTE | -6.8972513,107.6385574/-6.91358,107.62718/1 |
| 113916 | E5D9904F0A8B4F99 | 2/1/2014 0:12 | PAGELOAD | /5.10.83.24/ |
| 113917 | 81CC9E4AD224357E | 2/1/2014 0:13 | WIDGETLOAD | /192.95.25.92/ |
| 11318 | A44EB361A179A49E | 2/1/2014 0:13 | SEARCHPLACE | taman+f/10 |
| 113919 | A44EB361A179A49E | 2/1/2014 0:13 | FINDROUTE | -6.8972513,107.6385574/-6.91358,107.62718/1 |
| 113920 | D0AB08D956A351E4 | 2/1/2014 0:15 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113921 | D0AB08D956A351E4 | 2/1/2014 0:16 | SEARCHPLACE | istanta/0 |
| 113922 | D0AB08D956A351E4 | 2.1.2014 0:16 | SEARCHPLACE | istaba/0 |
| 113923 | D0AB08D956A351E4 | 2/1/2014 0:16 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113924 | D0AB08D956A351E4 | 2/1/2014 0:17 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |

| | | | | |
|---|---|---|---|---|
| 113925 | A44EB361A179A49E | 2/1/2014 0:18 | SEARCHPLACE | kantor+po/10 |
| 113926 | A44EB361A179A49E | 2/1/2014 0:18 | SEARCHPLACE | kantor+pos/10 |
| 113927 | A44EB361A179A49E | 2/1/2014 0:18 | SEARCHPLACE | kantor+pos+ci/10 |
| 113928 | A44EB361A179A49E | 2/1/2014 0:18 | SEARCHPLACE | kantor+pos+cimahi/10 |
| 113929 | A44EB361A179A49E | 2/1/2014 0:18 | FINDROUTE | -6.7185828,107.0150728/-6.918881548242062,107.60667476803064/1 |
| 113930 | A44EB361A179A49E | 2/1/2014 0:18 | FINDROUTE | -6.9015366,107.5414474/-6.88574,107.53816/1 |
| 113931 | E5D9904F0A8B4F99 | 2/1/2014 0:22 | PAGELOAD | /5.10.83.49/ |
| 113932 | E5D9904F0A8B4F99 | 2/1/2014 0:22 | PAGELOAD | /180.253.140.219/ |
| 113933 | E5D9904F0A8B4F99 | 2/1/2014 0:24 | PAGELOAD | /180.253.140.219/ |
| 113934 | E5D9904F0A8B4F99 | 2/1/2014 0:25 | PAGELOAD | /180.253.140.219/ |
| 113935 | E5D9904F0A8B4F99 | 2/1/2014 0:25 | FINDROUTE | -6.90608,107.61530/-6.89140,107.61060/2 |
| 113936 | E5D9904F0A8B4F99 | 2/1/2014 0:26 | PAGELOAD | /118.137.96.28/ |
| 113937 | E5D9904F0A8B4F99 | 2/1/2014 0:26 | FINDROUTE | -6.89459,107.58818/-6.89876,107.60886/2 |
| 113938 | E5D9904F0A8B4F99 | 2/1/2014 0:27 | FINDROUTE | -6.90608,107.61530/-6.89140,107.61060/2 |
| 113939 | E5D9904F0A8B4F99 | 2/1/2014 0:28 | FINDROUTE | -6.89977,107.62706/-6.89140,107.61060/2 |
| 113940 | E5D9904F0A8B4F99 | 2/1/2014 0:28 | FINDROUTE | -6.89459,107.58818/-6.86031,107.61287/2 |
| 113941 | D0AB08D956A351E4 | 2/1/2014 0:28 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113942 | A44EB361A179A49E | 2/1/2014 0:29 | FINDROUTE | -6.9172304,107.6042556/-6.92663,107.63644/1 |
| 113943 | A44EB361A179A49E | 2/1/2014 0:29 | FINDROUTE | -6.9172448,107.6042255/-6.92663,107.63644/1 |
| 113944 | D0AB08D956A351E4 | 2/1/2014 0:30 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113945 | D0AB08D956A351E4 | 2/1/2014 0:32 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113946 | D0AB08D956A351E4 | 2/1/2014 0:33 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113947 | A44EB361A179A49E | 2/1/2014 0:35 | SEARCHPLACE | jalan+asia+af/8 |
| 113948 | A44EB361A179A49E | 2/1/2014 0:35 | FINDROUTE | -6.9172448,107.6042255/-6.92163,107.61046/1 |
| 113949 | A44EB361A179A49E | 2/1/2014 0:35 | SEARCHPLACE | taman+fotog/10 |

| | | | | |
|---|---|---|---|---|
| 113950 | A44EB361A179A49E | 2/1/2014 0:36 | FINDROUTE | -6.917321,107.6043132/-6.921568846707516,107.61015225201845/1 |
| 113951 | E5D9904F0A8B4F99 | 2/1/2014 0:38 | PAGELOAD | /5.10.83.68/ |
| 113952 | E5D9904F0A8B4F99 | 2/1/2014 0:38 | PAGELOAD | /5.10.83.28/ |
| 113953 | E5D9904F0A8B4F99 | 2/1/2014 0:40 | PAGELOAD | /206.53.152.81/m |
| 113954 | E5D9904F0A8B4F99 | 2/1/2014 0:40 | FINDROUTE | -6.90598,107.59714/-6.91728,107.60417/1 |
| 113955 | E5D9904F0A8B4F99 | 2/1/2014 0:41 | PAGELOAD | /5.10.83.30/ |
| 113956 | E5D9904F0A8B4F99 | 2/1/2014 0:40 | PAGELOAD | /5.10.83.28/ |
| 113957 | E5D9904F0A8B4F99 | 2/1/2014 0:55 | PAGELOAD | /5.10.83.99/ |
| 113958 | D0AB08D956A351E4 | 2/1/2014 1:00 | SEARCHPLACE | babd/1 |
| 113959 | D0AB08D956A351E4 | 2/1/2014 1:00 | SEARCHPLACE | babdu/1 |
| 113960 | D0AB08D956A351E4 | 2/1/2014 1:00 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113961 | D0AB08D956A351E4 | 2/1/2014 1:09 | FINDROUTE | -6.38355,106.919975/-6.85029,107.58496/1 |
| 113962 | D0AB08D956A351E4 | 2/1/2014 1:10 | FINDROUTE | -6.90598,107.59714/-6.85029,107.58496/1 |
| 113963 | D0AB08D956A351E4 | 2/1/2014 1:10 | FINDROUTE | -6.90598,107.59714/-6.90855,107.61082/1 |
| 113964 | E5D9904F0A8B4F99 | 2/1/2014 1:10 | PAGELOAD | /5.10.83.49/ |
| 113965 | D0AB08D956A351E4 | 2/1/2014 1:12 | SEARCHPLACE | tea/10 |
| 113966 | A44EB361A179A49E | 2/1/2014 1:15 | SEARCHPLACE | taman+pustaka/10 |
| 113967 | A44EB361A179A49E | 2/1/2014 1:15 | SEARCHPLACE | taman+pustaka+besarn/8 |
| 113968 | A44EB361A179A49E | 2/1/2014 1:15 | FINDROUTE | -6.9135911,107.6272095/-6.90179,107.62301/1 |
| 113969 | E5D9904F0A8B4F99 | 2/1/2014 1:16 | PAGELOAD | /36.72.98.13/ |
| 113970 | E5D9904F0A8B4F99 | 2/1/2014 1:17 | PAGELOAD | /120.173.21.110/m |
| 113971 | E5D9904F0A8B4F99 | 2/1/2014 1:17 | SEARCHPLACE | jalan+abdrrahman+saleh/10 |
| 113972 | E5D9904F0A8B4F99 | 2/1/2014 1:17 | FINDROUTE | -6.90872,107.62253/-6.90774,107.60908/1 |
| 113973 | A44EB361A179A49E | 2/1/2014 1:19 | FINDROUTE | -6.9158359,107.6101751/-6.90691,107.62259/1 |
| 113974 | E5D9904F0A8B4F99 | 2/1/2014 1:19 | FINDROUTE | -6.91335,107.64827/-6.86198,107.59193/1 |
| 113975 | E5D9904F0A8B4F99 | 2/1/2014 1:25 | PAGELOAD | /114.79.13.124/ |

| | | | | |
|---|---|---|---|---|
| 113976 | E5D9904F0A8B4F99 | 2/1/2014 1:25 | PAGELOAD | /5.10.83.24/ |
| 113977 | E5D9904F0A8B4F99 | 2/1/2014 1:25 | FINDROUTE | -6.91485,107.59123/-6.91593,107.65588/1 |
| 113978 | E5D9904F0A8B4F99 | 2/1/2014 1:26 | PAGELOAD | /5.10.83.82/ |
| 113979 | E5D9904F0A8B4F99 | 2/1/2014 1:28 | FINDROUTE | -6.91593,107.65588/-6.91485,107.59123/1 |
| 113980 | A44EB361A179A49E | 2/1/2014 1:29 | FINDROUTE | -6.9250709,107.6204635/-6.91728,107.60417/1 |
| 113981 | A44EB361A179A49E | 2/1/2014 1:35 | FINDROUTE | -6.9252132,107.6200288/-6.91728,107.60417/1 |
| 113982 | A44EB361A179A49E | 2/1/2014 1:36 | FINDROUTE | -6.922427886995373,107.61768691241741/-6.91728,107.60417/1 |
| 113983 | E5D9904F0A8B4F99 | 2/1/2014 1:36 | FINDROUTE | -6.91431,107.63921/-6.94024,107.71550/1 |
| 113984 | E5D9904F0A8B4F99 | 2/1/2014 1:37 | PAGELOAD | /5.10.83.98/ |
| 113985 | A44EB361A179A49E | 2/1/2014 1:37 | FINDROUTE | -6.921635413232821,107.61909071356058/-6.91728,107.60417/1 |
| 113986 | E5D9904F0A8B4F99 | 2/1/2014 1:38 | FINDROUTE | -6.88936,107.57533/-6.92600,107.63628/1 |
| 113987 | E5D9904F0A8B4F99 | 2/1/2014 1:39 | PAGELOAD | http://www.kiri.travel/m/r/?qs=trans+studi... |
| 113988 | E5D9904F0A8B4F99 | 2/1/2014 1:39 | FINDROUTE | -6.92600,107.63628/-6.88936,107.57533/1 |
| 113989 | A44EB361A179A49E | 2/1/2014 1:41 | SEARCHPLACE | terminal+ta/10 |
| 113990 | A44EB361A179A49E | 2/1/2014 1:41 | FINDROUTE | -6.9158359,107.6101751/-6.90658,107.61623/1 |
| 113991 | A44EB361A179A49E | 2/1/2014 1:42 | FINDROUTE | -6.9158359,107.6101751/-6.90658,107.61623/1 |
| 113992 | D0AB08D956A351E4 | 2/1/2014 1:50 | FINDROUTE | -6.38355,106.919975/-7.08933734335005,107.562576737255/1 |
| 113993 | A44EB361A179A49E | 2/1/2014 1:51 | SEARCHPLACE | taman+ci/10 |
| 113994 | A44EB361A179A49E | 2/1/2014 1:51 | SEARCHPLACE | taman+cilaki/10 |
| 113995 | E5D9904F0A8B4F99 | 2/1/2014 1:52 | PAGELOAD | /206.53.152.33/m |
| 113996 | E5D9904F0A8B4F99 | 2/1/2014 1:52 | FINDROUTE | -6.90598,107.59714/-6.91728,107.60417/1 |
| 113997 | A44EB361A179A49E | 2/1/2014 1:54 | FINDROUTE | -6.901306,107.6214169/-6.90336,107.62235/1 |
| 113998 | A44EB361A179A49E | 2/1/2014 1:54 | FINDROUTE | -6.901306,107.6214169/-6.90336,107.62235/1 |
| 113999 | E5D9904F0A8B4F99 | 2/1/2014 | PAGELOAD | /5.10.83.27/ |

| | | | | |
|---|---|---|---|---|
| 114000 | 308201BB30820124 | 2/1/2014 1:15 | SEARCHPLACE | riau+jucntion/10 |
| 114001 | 308201BB30820124 | 2/1/2014 1:56 | FINDROUTE | -6.90687,107.61239/-6.89032,107.57961/2 |
| 114002 | E5D9904F0A8B4F99 | 2/1/2014 1:57 | PAGELOAD | /118.99.112.66/ |
| 114003 | 308201BB30820124 | 2/1/2014 1:57 | FINDROUTE | -6.90687,107.61239/-6.90159,107.60442/1 |
| 114004 | 308201BB30820124 | 2/1/2014 1:57 | FINDROUTE | -6.90687,107.61239/-6.89032,107.57961/2 |
| 114005 | E5D9904F0A8B4F99 | 2/1/2014 1:58 | FINDROUTE | -6.88211,107.60378/-6.90774,107.60908/1 |
| 114006 | A44EB361A179A49E | 2/1/2014 1:59 | FINDROUTE | -6.9212516,107.6196466/-6.91728,107.60417/1 |
| 114007 | 308201BB30820124 | 2/1/2014 1:59 | FINDROUTE | -6.90687,107.61239/-6.91486,107.60824/1 |
| 114008 | 687C44EB2424285D | 2/1/2014 1:59 | WIDGETLOAD | http://www.cendekialeadershipschool.sc... |
| 114009 | E5D9904F0A8B4F99 | 2/1/2014 2:00 | FINDROUTE | -6.88166,107.61561/-6.90774,107.60908/1 |