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

P rtumbuhan t knologi hingga saat ini t lah m nghasilkan banyak s kali data-data, namun s ring kali p milik data hanya m nggunakan data t rs but s p rlunya saja. Jika dilihat l bih rinci, s b - narnya jika data t rs but diolah l bih lanjut, dapat m nghasilkan s suatu yang l bih. Salah satu cara m ngolah data t rs but adalah d ngan m nggunakan t knik *data mining*. D ngan m nggunakan t knik *data mining* akan m mp rmudah m nganalisa masalah, p ngambilan k simpulan, bahkan m mp rmudah konsum n dalam m mb li jasa atau barang.

Tujuan utama dari *data mining* adalah *knowledge* [1]. *Knowledge* m rupakan suatu informasi yang b rharga dan dapat dijadikan landasan untuk m nganalisa atau m mbuat k simpulan. Untuk m ndapatkan *knowledge*, dapat dilakukan d ngan cara m lakukan p ncarian *pattern* atau pola yang m rupakan salah satu tahap dari *data mining*. Pola inilah yang akan m mp rlihatkan data manakah yang m narik dan dapat dijadikan *knowledge* yang akan digunakan untuk m nganalisa data t rs but.

Pada p n litian *data mining* ini, p nulis m miliki data *log* histori KIRI s lama 1 bulan. Data t rs but akan diimpl m ntasikan pros s *data mining* untuk m ndapatkan *pattern* dan *knowledge* yang t rkandung pada data *log* KIRI. Data *log* t rs but m miliki 5 *field* untuk s tiap *entry* s bagai b rikut:

statisticId, primary k y dari ntry

v rifi r, m ngid ntifikasikan sumb r dari p ncarian ini

timestamp, waktu k tika p ngguna KIRI m ncari rut angkot

type, tip fungsi yang digunakan

additionalInfo, m ncatat koordinat awal, koordinat akhir, dan banyak rut yang dit mukan pada p ncarian ini

B rdasarkan hal diatas, p nulis ingin m ndapatkan pola yang m narik dan m nghasilkan *knowledge* yang b rguna dan dapat dipakai baik untuk KIRI ataupun p m rintah.

1.2 Perumusan Masalah

Dengan mengacu pada uraian diskripsi diatas, maka permasalahan yang dibahas dan ditliti 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* histori?

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 ditliti berupa :

Penelitian ini dibatasi hanya pada permasalahan pada penerapan *data mining* pada *data log* KIRI

Data *log* yang digunakan untuk mining merupakan log satu bulan dari KIRI

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 kesimpulan

1.6 Sistematika Pembahasan

Sistematika 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 T ori, b risi dasar t ori m ng nai *data mining*, *data cleaning*, *data integration*, *data selection*, *data transform*, *decision tree*, *pattern evaluation*, *knowledge presentation* dan *log* histori KIRI

BAB 3: B risi analisa dasar t ori yang akan digunakan, analisa data s rta tahap *preprocessing* data yang akan digunakan, s rta analisa m rancang aplikasi *data mining log* histori KIRI b rikut diagram *use case*, sk nario, dan diagram k las

BAB 4: B risi p rancangan dari aplikasi *data mining log* histori KIRI yang akan dibangun

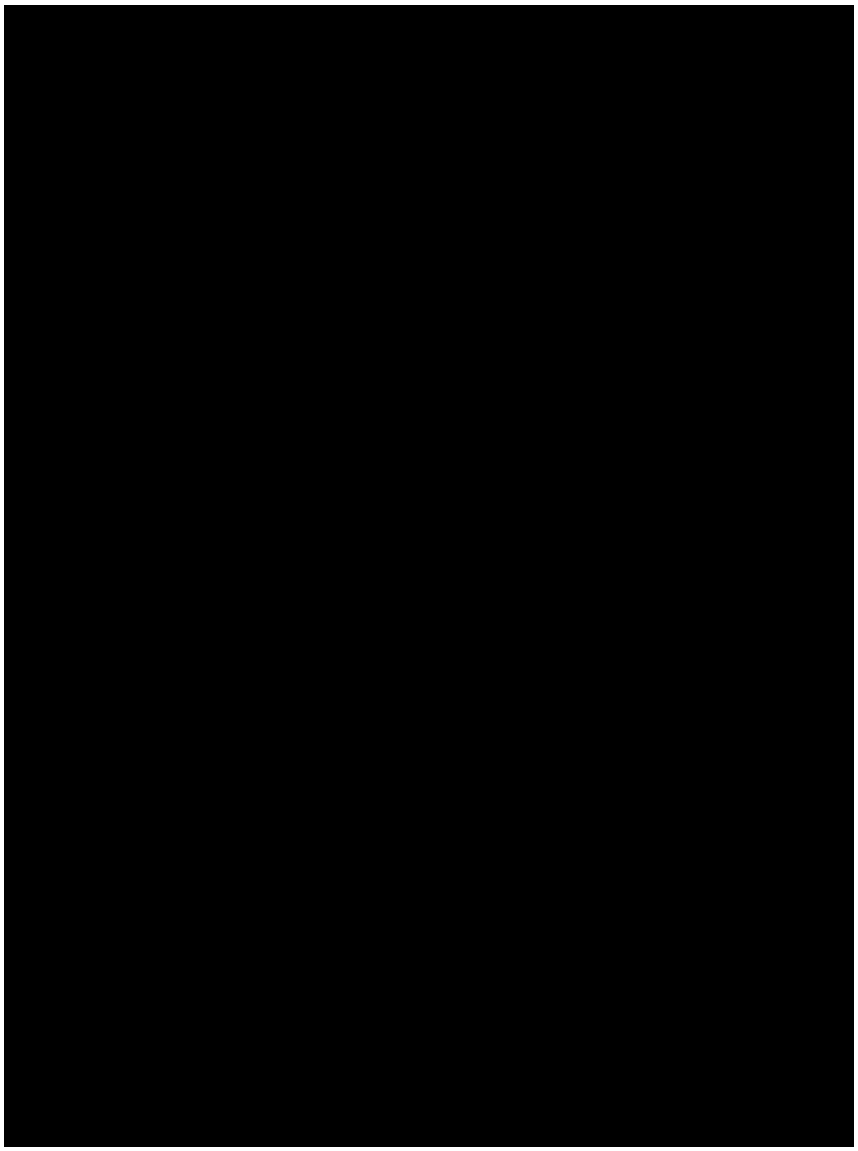
BAB 5: B risi hasil yang dip rol h dan k simpulan dari p n litian *data mining log* histori KIRI

BAB 2

LANDASAN TEORI

2.1 *Data Mining*

Data mining m rupakan m rupakan pros s yang m lakukan p ngambilan inti sari atau p nggalian *knowledge* dari data yang b sar dan m rupakan salah satu langkah dari *knowledge discovery*.



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

M nurut [1], *knowledge discovery* dapat dibagi m njadi 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 p rtama hingga k mpat m rupakan bagian dari *data preprocessing*, dimana data-data disiapkan untuk dilakukan p nggalian data. Tahap *data mining* m rupakan tahap dimana m lakukan p nggalian data. Tahap k nam m rupakan tahap p ncarian pola yang m r pr s ntasikan *knowledge*. S dangkan tahap t rakhir m rupakan visualisasi dan r pr s ntasi dari *knowledge* yang sudah dip rol h dari tahap s b lumnya.

2.1.1 *Data Cleaning*

Data cleaning m rupakan tahap *data mining* untuk m nghilangkan *missing value* dan *noisy data*. Pada umumnya, *data* yang dip rol h dari *database* t rdapat nilai yang tidak s mpurna s p rti nilai yang hilang, nilai yang tidak valid atau salah k tik. Atribut dari suatu *database* yang tidak r l van atau r dudansi bisa diatasi d ngan m nghapus atribut t rs but. Contoh studi data yang m miliki *missing value* dan *noisy data* dapat dilihat pada tab l 2.1

Tab l 2.1: Tab l m ngandung *missing value* dan *noisy*

IdP njualan	NamaBarang	Costum r	Harga	BanyakBarang
1	Mous	Elvin	45000	2
2	K yboard	All ria	-35000	1
3	Monitor		225000	1

Dapat dilihat, pada idP njualan 2, harga dari k yboard adalah -35000, itu m rupakan *noisy* kar na tidak mungkin nilai harga suatu barang dibawah 0. Pada idP njualan 3, kolom *costumer* tidak m miliki nilai, dan itu m rupakan *missing value*.

Missing Values

Missing values akan m ngganggu pros s *data mining* pada komput r dan dapat m nghasilkan nilai akhir yang tidak s suai. T rdapat b b rapa t knik untuk m ngatasi *missing values* yaitu

M mbuang tupl yang m ngandung nilai yang hilang

M ngisi nilai yang hilang s cara manual

M ngisi nilai yang hilang d ngan m nggunakan nilai konstan yang b rsifat umum

M nggunakan nilai rata-rata dari suatu atribut untuk m ngisi nilai yang hilang

Noisy Data

Noisy data m rupakan nilai yang b rasal dari rror atau tidak valid. *Noisy data* dapat dihilangkan d ngan m nggunakan t knik *smoothing*. T rdapat 3 m tod untuk m nghilangkan *noisy data* yaitu

Binning, m rupakan m tod p ngisian data s suai d ngan pros s yang dilakukan pada data t rs but

Regression, m rupakan m tod yang m ncari d tail p rsamaan atribut untuk m mpr diksikan suatu nilai

Clustering, m rupakan m tod p ng lompokan dimana dit mukan *outliers* yang dapat dibuang

2.1.2 *Data Integration*

Data integration m rupakan tahap m nggabungkan data dari b rbagai sumb r. Sumb r t rs but bisa t rmasuk b b rapa *database*, *data cubes*, atau bahkan *flat data*. *Data cube* m rupakan t knik p ngambilan data-data dari *data warehouse* dan dilakukan op rasi agr gasi s suai d ngan kondisi t rt ntu (contoh, p njumlahan total p njualan p r tahun dari 2005-2010). S dangkan *flat data* m rupakan data yang disimpan d ngan cara apapun untuk m r pr s ntasikan databas mod l pada s buah data baik b rb ntuk *plain text file* maupun *binary file*.

Tahap ini harus dilakukan s cara t liti t rutama k tika dalam m masangkan nilai-nilai yang b rasal dari sumb r yang b rb da. Pada tahap ini, p rlu dilakukan id ntifikasi data apakah data t rs but dapat diturunkan atau tidak agar data yang dip rol h tidak t rlalu b sar. *Data integration* yang baik m rupakan int grasi yang dapat m maksimalkan k c patan dan m ningkatkan akurasi dari pros s *data mining*. Contoh studi kasus dari *data integration*, jika suatu p rusahaan s patu A m miliki dua pabrik d ngan *database* lokal pada masing-masing pabrik, jika akan dilakukan *data mining* pada k dua *database* t rs but, maka k dua *database* akan digabung dan p rlu dip rhatikan s rta dip rbaiki nilai-nilai s p rti *primary key*, atribut, dan lain-lain agar tidak t rjadi *error* pada *database* yang sudah digabung. Pros s dari p nggabungan hingga p rbaikan nilai-nilai pada k dua databas t rs but adalah pros s *data integration*.

2.1.3 *Data Selection*

Pros s dimana data-data yang r l van d ngan analisis akan diambil dari databas dan data yang tidak r l van akan dibuang. S bagai contoh kasus, jika akan dilakukan analisa m ng nai nilai mahasiswa pada tab l nilai yang m miliki atribut s bagai b rikut:

NPMMahasiswa

NamaMahasiswa

J nisK lamin

Alamat

MataKuliah

NilaiART

NilaiUTS

NilaiUAS

Maka, atribut yang b rpot nsi diambil adalah MataKuliah, NilaiART, NilaiUTS, NilaiUAS, s - dangkan atribut yang akan dibuang adalah NPMMahasiswa, NamaMahasiswa J nisK lamin, dan Alamat kar na tidak t rlalu b rhubungan d ngan analisa.

2.1.4 *Data Transformation*

Data transformation m rupakan tahap p ngubahan data agar siap dilakukan pros s *data mining*. *Data transformation* bisa m libatkan:

Smoothing, pros s untuk m mbuang *noise* s p rti yang dilakukan pada tahap *data cleaning*

Aggregation, pros s m ngganti nilai-nilai m njadi suatu nilai yang dapat m wakili nilai s b - lumnya

Generalization, pros s dimana m mbuat suatu nilai yang b rsifat khusus m njadi nilai yang b rsifat umum

Normalization, pros s dimana suatu nilai dapat diubah skalanya m njadi nilai yang l bih k cil dan sp sifik

Attribute construction, pros s m mbuat atribut baru yang b rasal dari b b rapa atribut untuk m mbantu pros s data mining

Smoothing

Smoothing m rupakan bagian dari *data cleaning* untuk m nghilangkan *noise* pada databas . T knik dari *smoothing* adalah *binning*, *regression*, dan *clustering*. P nj lasan t knik *smoothing* dapat dilihat pada 2.1.1, bagian *noisy data*.

Aggregation

Aggregation, dimana suatu k simpulan atau hasil dari *aggregation operation* yang disimpan dalam databas . Contoh studi kasus, jika t rdapat suatu databas dari toko A, kita dapat m nggunakan op rasi *aggregation* untuk m ncari total p ndapatan d ngan r ntang hari t rt ntu.

Generalization

generalization, dimana suatu data yang m miliki nilai *primitive* atau *low level* diubah m njadi *high level* d ngan m nggunakan kons p hirarki. Contoh studi kasus, nilai pada atribut umur dapat dik lompokkan m njadi muda, d wasa, tua.

Normalization

Atribut dapat dinormalisasi d ngan m mb ri skala pada nilainya s hingga nilai t rs but m njadi suatu rang yang l bih sp sifik dan k cil s p rti 0,0 sampai 1,0. Dua t knik nnormalisasi yaitu, *min-max normalization* dan *z-score normalization*. *Min-max normalization* akan m ngubah s mua nilai m njadi nilai d ngan skala t rt ntu. D ngan m nggunakan rumus

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

Contoh kasus, misalkan nilai minimun dan maximum dari suatu p ndapatan adalah 12.000 dan 98.000, akan diubah m njadi b rskala antara 0,0 sampai 1,0. Jika ada nilai p ndapat yang baru, yaitu 73.600, maka akan m njadi

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

z-score normalization m rupakan normalisasi b rdasarkan nilai rata-rata dan standar d viasi dari nilai-nilai atribut d ngan cara

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

Contoh kasus, misal nilai rata-rata dan standar d viasi dari nilai-nilai atribut p ndapatan adalah 54.000 dan 16.000. D ngan *z-score*, jika ada nilai p ndapatan baru yaitu 73600, maka akan diubah m njadi

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

Attribute Construction

Attribute Construction m rupakan t knik m nambahkan atribut baru yang b rdasarkan dari atribut yang sudah ada guna m nambah akurasi. Contoh kasus, dibuat atribut baru b rnama ar a b rdasarkan atribut panjang dan l bar.

2.1.5 *Data Mining*

Pada tahap ini, akan dilakukan pros s *data mining* d ngan m nggunakan input data yang sudah dipros s pada tahap s b lumnya (*data cleaning, data selection, data integration,* dan /*data transformation*).

Classification and Prediction

Classification m rupakan p mod lan yang dibangun untuk m mpr diksikan lab l kat gori, s p rti "'baik"', "'cukup"', dan "'buruk"' dalam sist m p nilaian sikap s orang siswa atau "'mini bus"', "'bus"', atau "'s dan"' dalam kat gori tip mobil. Kat gori t rs but dapat dir pr s ntasikan d ngan m nggunakan nilai diskr t. Nilai diskr t m rupakan nilai yang t rpisah dan b rb da, s p rti 1 atau 5. Kat gori yang dir pr s ntasikan ol h nilai diskr t maka akan m njadi nilai yang t rurut dan

tidak m miliki arti, s p rti 1,2,3 untuk m r pr s ntasikan kat gori tip mobil "'mini bus"', "'bus"', dan "'s dan"'.

Prediction m rupakan mod l yang dibangun untuk m ramalkan fungsi nilai kontinu atau *ordered value*. *Ordered value* m rupakan nilai yang t rurut dan b rlanjut. Contoh studi kasus untuk p mod lan pr diction adalah s orang mark ting ingin m ramalkan s b rapa banyak konsum n yang akan b lanja di s buah toko dalam waktu satu bulan. P mod lan t rs but dis but *predictor*. *Regression Analysis*, m rupakan m todologi statistik yang digunakan untuk *numeric prediction*. *Classification* dan *numeric prediction* m rupakan dua j nis utama dalam masalah pr diksi.

Data Classification m rupakan pros s untuk m lakukan klasifikasi. *Data classification* m miliki dua tahap pros s, yaitu *learning step* dan tahap klasifikasi s p rti pada ilustrasi di gambar 2.2. *Learning step* m rupakan langkah p mb lajaran, di mana algoritma klasifikasi m mbangun *classification rules* (yang b risi syarat atau aturan s buah nilai masuk k dalam kat gori t rt ntu) d ngan cara m nganalisis *training set* yang m rupakan *database tuple*. Kar na p mbuatan *classification rules* m nggunakan *training set*, yang dik nal juga s bagai *supervised learning*. Pada tahap k dua, dilakukan pros s klasifikasi nilai b rdasarkan *classification rules* yang sudah dibangun dari tahap p rtama.

Decision Tree

Salah satu cara p mbuatan *classification rules* pada *Data Classification* adalah d ngan m mbuat *decision tree* (pohon k putusan). *Decision tree* m rupakan *flowchart* yang b rb ntuk pohon, dimana s tiap nod int rnal (*nonleaf* nod) m rupakan hasil t st dari atribut, s tiap cabang m r pr s ntasikan output dari t st, dan s tiap nod daun m miliki *class label*. Bagian paling atas dari pohon dis but *root node*. Contoh studi kasus, pohon k putusan untuk m n ntukan apakah s orang konsum n akan m mb li komput r atau tidak (ilustrasi pohon k putusan pada gambar 2.3)

Decision Tree Induction *Decision tree induction* m rupakan p latihan pohon k putusan dari tup l p latihan k las b rlab l. T rdapat b b rapa t knik untuk m mbuat *decission tree* dua diantaranya adalah ID3 dan C4.5. ID3 m rupakan t knik p mbuatan *decision tree* d ngan m manfaatkan *entropy* dan *gain info* untuk m n ntukan atribut yang t rbaik untuk nod pada *decision tree*. S dangkan C4.5 m rupakan t knik lanjutan dari ID3 yang m nggunakan *gain ratio* untuk m lakukan p ng c kan pada nilai *gain info*.K dua t knik t rs but m nggunakan p nd katan *greedy* yang m rupakan *decission tree* yang dibangun s cara *top-down recursive divide and conquer*. Algoritma yang dip rlukan s cara umum sama, hanya b rb da pada *attribute_selection_method*. B rikut algoritma untuk m mbuat pohon k putusan dari suatu tup l p latihan.

Require: Partisi data, D, m rupakan s t data p latihan dan k las lab l

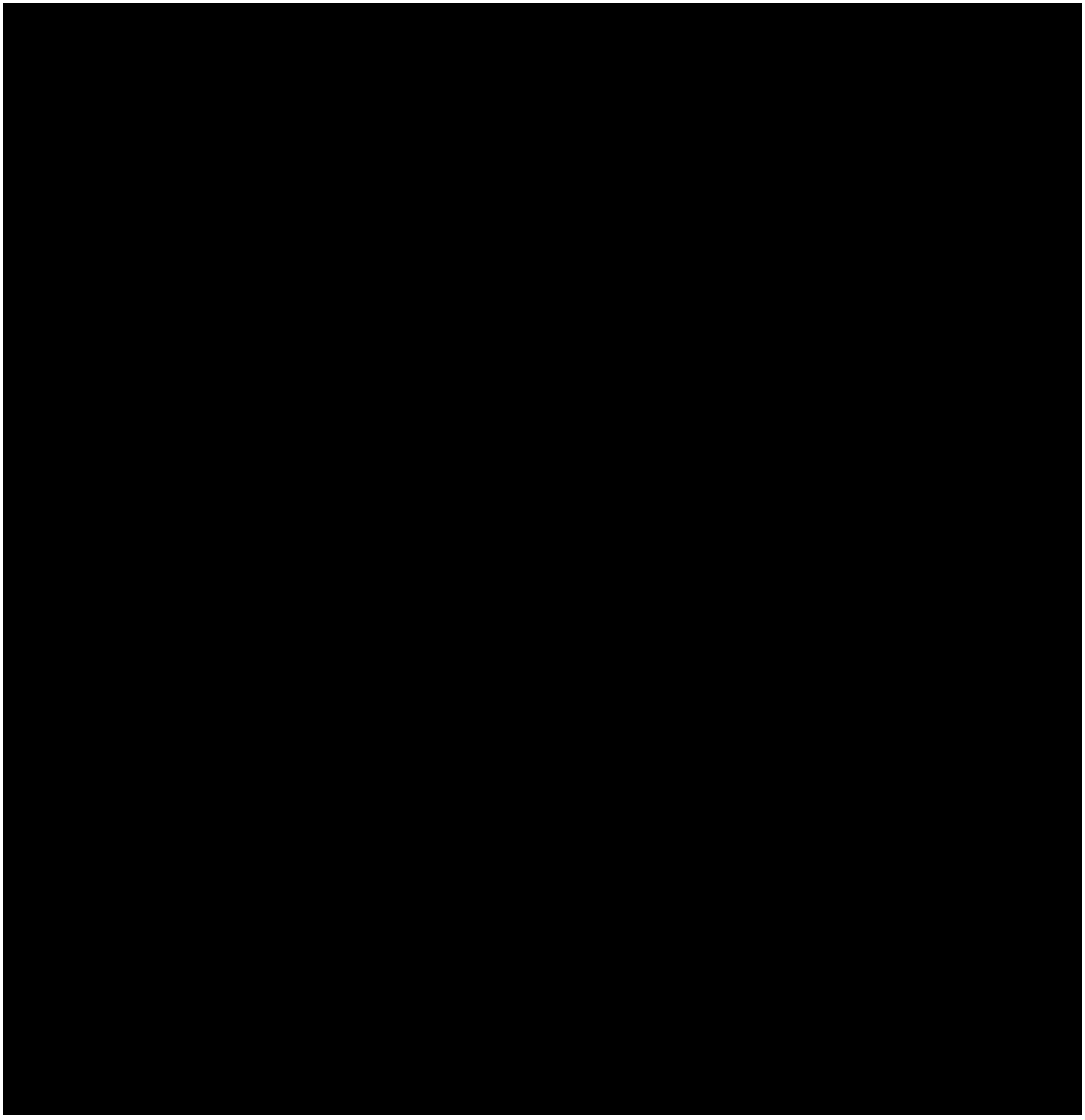
Require: *attribute_list*, m rupakan s t dari atribut kandidat

Require: *Attribute_selection_method*, pros dur untuk m n ntukan *splitting criterion*. Pada input ini, t rdapat juga data *splitting_attribute* dan mungkin salah satu dari *split point* atau *splitting subset*

Ensure: Pohon k putusan

1: M mbuat nod N;

2: **if** tupl pada D m rupakan k las yang sama, C **then**



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

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

3: **return** N s bagai nod daun d ngan lab l k las C;
4: **end if**
5: **if** attribut _list tidak ada nilai atau kosong **then**
6: **return** N s bagai nod daun d ngan lab l k las yang t rpaling banyak pada D; {majority voting}
7: **end if**
8: m manggil m thod Attribut _s l ction_m thod(D, atribut _list) untuk m ncari nilai t rbaik splitting_crit rion;
9: m namakan nod N d ngan splitting_crit rion;
10: **if** splitting_attribut m rupakan nilai discr t and multiway splits diizinkan **then**
11: attribut _list attribut _list - splitting_attribut ; {m nghapus splitting_attribut }
12: **end if**
13: **for all** hasil j dari splitting_crit rion **do**
14: Dj m rupakan himpunan data tup l D yang s suai d ngan j;
15: **if** Dj tidak ada nilai atau kosong **then**
16: m lampirkan daun yang dib ri lab l d ngan k las mayoritas di D k nod N;
17: **else**
18: m lampirkan nod yang dik mbalikan ol h g n rat _d cision_tr (Dj, attribut _list) k nod N;
19: **end if**
20: **end for**
21: **return** N;

Pohon k putusan akan dimulai d ngan satu nod , yaitu N, m r pr s ntasikan tupl p latihan pada D (baris 1)

Jika tupl di D m miliki k las yang sama s mua, maka nod N akan m njadi daun dan dib ri lab l dari k las t rs but (baris 2 sampai 4). P rlu dik tahui bahwa baris 5 sampai 7 akan m ngakhiri kondisi.

Jika tupl di D ada k las yang b rb da, maka algoritma akan m manggil *attribute_selection_method* untuk m n ntukan *splitting criterion*. *Splitting criterion* akan m n ntukan atribut pada nod N yang

Gambar 2.4: J nis-j nis *split point*, [1]

m rupakan nilai t rbaik untuk m m cah nilai atribut pada tupl k dalam k las masing-masing. (baris 8)

Nod N akan diisi d ngan hasil dari *splitting criterion* (baris 9). K mudian krit ria t rs but agak dib ntuk cabangnya masing-masing s suai pada baris 13 dan 14. T rdapat tiga k mungkinan b ntuk krit ria jika A m rupakan *splitting_attribute* yang m miliki nilai unik s p rti $\{a_1, a_2, ..., a_v\}$ s p rti pada gambar 2.4, yaitu,

1. *Discrete valued*: cabang yang dihasilkan m miliki k las d ngan nilai diskr t. Kar na k las yang dihasilkan diskr t dan hanya m miliki nilai yang sama pada cabang t rs but, maka *attribut_list* akan dihapus (baris 10 sampai 12)

2. *Continuous values*: cabang yang dihasilkan m miliki jarak nilai untuk m m nuhi suatu kondisi (contoh: A <= split_point), dimana nilai *split_point* adalah nilai p mbagi yang dik mbalikan ol h *attribute_selection_method*

3. *Dicrete valued and a binary tree*: cabang yang dihasilkan adalah dua b rupa nilai iya atau tidak dari "'apakah A anggota S_a"', dimana S_a m rupakan subs t dari A, yang dik mbalikan ol h *Attribute_selection_method*

K mudian, akan dipanggil k mbali algoritma *decision tree* untuk s tiap nilai hasil p mbagian pada tupl , D_j (baris 18).

R kursif t rs but akan b rh nti k tika salah satu dari kondisi t rp nuhi, yaitu

1. S mua tupl pada partisi D m rupakan bagian dari k las yang sama.

2. Sudah tidak ada atribut yang dapat dilakukan p mbagian lagi (dilakukan p ng c kan pada baris 4). Disini, akan dilakukan *majority voting* (baris 6) yang akan m ngkonv rsi nod N m njadi *leaf* dan dib ri lab l d ngan k las yang t rbanyak pada D.

3. Sudah tidak ada tupl yang dapat dib ri cabang, D_j sudah kosong (baris 15) dan *leaf* akan dibuat d ngan *majority class* pada D (baris 16).

Pada baris 21, akan dik mbalikan nilai *decision tree* yang t lah dibuat.
subsubs ction*Attribute Selection Measure*

Attribute Selection Measure m rupakan suatu hirarki untuk p milihan *splitting criterion* yang t rbaik yang m misah partisi data (D), tupl p latihan k las lab l k dalam k las masing-masing. *Attribute Selection Measure* m ny diakan p ringkat untuk s tiap atribut pada training tupl . Jika *splitting criterion* m rupakan nilai *continous* atau *binary trees*, maka nilai *split point* dan *splitting subset* harus dit ntukan s bagai bagian dari *splitting criterion*. Contoh dari *attribute selection measure* adalah *information gain*, *gain ratio*, dan *gini index*.

Notasi yang digunakan adalah s bagai b rikut. D m rupakan data partisi, s t p latihan dari *class-labeled* tupl . Jika lab l k las atribut m miliki m nilai yang b rb da yang m ndifinisikan m k las yang b rb da, C_i (for i=1,...,m). $C_{i,d}$ m njadi k las tupl dari C_i di D. |D| dan $|C_{i,d}|$ m rupakan banyak tupl pada D dan $C_{i,d}$.

ID3

ID3 m rupakan t knik untuk m mbuat *decision tree* d ngan m nggunakan *information gain* s bagai *attribute selection measure* untuk m milih atribut. Cara ID3 m ndapatkan *information gain* d ngan m nggunakan *entropy*. *Entropy* adalah ukuran *impurity* dari suatu data. Cara m ndapatkan nilai *entropy* adalah

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

Dimana pi m rupakan probabilitas tupl pada D t rhadap class C_i, dapat dip rol h d ngan $|C_{i,d}|/|D|$. Info(D) m rupakan nilai rata-rata *entropy* dari suatu lab l k las pada tupl D. Untuk m ng tahui atribut mana yang paling baik untuk dijadikan *splitting attribute*, adalah d ngan cara m nghitung nilai *entrophy* dari suatu atribut k mudian dis lisihkan d ngan nilai *entropy* dari D. Jika pada tupl D, m miliki atribut A d ngan v nilai yang b rb da, maka m nghitung *entropy* dari suatu atribut adalah

$$Info_A(D) = \sum_{j=1}^{v} \frac{jD_jj}{jDj} \quad Info(D_j)$$

$|D_j|/D$ m rupakan angka yang m nghitung bobot dari suatu partisi. S makin k cil nilai dari $Info_A(D)$, maka atribut t rs but masih m m rlukan informasi, s makin b sar nilai $Info_A(D)$, s makin tinggi pula tingkat *pure* dari suatu partisi.

S t lah m ndapatkan nilai Info(D) dan $Info_A(D)$, *information gain* dapat dip rol h dari s lisih nilai Info(D) dan $Info_A(D)$

$$Gain(A) = Info(D) \quad Info_A(D)$$

contoh kasus untuk ID3, dalam p ncarian*information gain*

Tab l 2.2: Contoh training s t

RID	umur	p ndapatan	siswa	r siko_kr dit	Class: m mb li_komput r
1	muda	tinggi	tidak	cukup	tidak
2	muda	tinggi	tidak	baik	tidak
3	r maja	tinggi	tidak	cukup	ya
4	d wasa	s dang	tidak	cukup	ya
5	d wasa	r ndah	ya	cukup	ya
6	d wasa	r ndah	ya	baik	tidak
7	r maja	r ndah	ya	baik	ya
8	muda	s dang	tidak	cukup	tidak
9	muda	r ndah	ya	cukup	ya
10	d wasa	s dang	ya	cukup	ya
11	muda	s dang	ya	baik	ya
12	r maja	s dang	tidak	baik	ya
13	r maja	tinggi	ya	cukup	ya
14	d wasa	s dang	tidak	baik	tidak

Pada tab l 2.2, t rdapat *training set*, D. Atribut k las lab l m rupakan dua nilai yang b rb da yaitu ya dan tidak, maka dari itu, nilai m = 2. C_1 diisi d ngan k las lab l b rnilai ya, s dangkan C_2 diisi d ngan k las lab l b rnilai tidak. T rdapat s mbilan tupl atribut k las lab l d ngan nilai ya dan lima tupl d ngan nilai tidak. Untuk dapat m n ntukan *splitting criterion*, *information gain* harus dihitung untuk s tiap atribut t rl bih dahulu. P rhitungan *entropy* untuk D adalah

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

S t lah dip rol h nilai *entropy* dari D, k mudian akan dihitung nilai *entropy* atribut dimulai dari atribut umur. Pada kat gori muda, t rdapat dua tupl d ngan k las ya dan tiga tupl d ngan k las tidak. Untuk kat gori r maja, t rdapat mpat tupl d ngan k las ya dan nol tupl d ngan k las tidak. Pada kat gori d wasa, t rdapat tiga d ngan k las ya dan dua d ngan k las tidak. P rhitungan nilai *entropy* atribut umur t rhadap D s bagai b rikut

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

S t lah m ndapatkan *entropy* dari atribut umur, maka nilai *gain information* dari atribut umur adalah

$$Gain_{(umur)} = Info(D) \quad Info_{age}(D) = 0.940 \quad 0.694 = 0.246bits$$

D ngan m lakukan hal yang sama, dapat dip rol h nilai *gain* untuk atribut p ndapatan adalah 0.029 *bits*, untuk nilai *gain*(siswa) adalah 0.151 *bits*, dan *gain*(r siko_kr dit) = 0.048 *bits*. Kar na

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

nilai *gain* dari atribut umur m rupakan nilai t rb sar diantara s mua atribut, maka atribut umur dipilih m njadi *splitting attribute*. S t lah dit ntukan, nod N akan m mb ntuk cabang b rdasarkan nilai dari atribut umur s p rti pada gambar 2.5.

Untuk atribut yang m rupakan nilai *continuous*, harus dicari nilai *split point* untuk A. Nilai-nilai dari dua angka yang b rs b lahan dapat diambil nilai t ngahnya untuk dijadikan *split-point*. Jika t rdapat v nilai yang b rb da dari A, maka akan t rdapat v-1 k mungkinan *split point*. K mudian nilai *split point* akan dijadikan s bagai nilai p mbagi, s bagai contoh: A <= *split-point* m rupakan cabang p rtama, dan A > *split-point* m rupakan cabang k dua.

C4.5

Information gain akan m miliki nilai yang baik jika suatu atribut m miliki banyak nilai yang b rb - da, namun hal itu tidak s lalu bagus. S bagai contoh kasus, jika nilai id suatu tabl yang m miliki nilai unik, maka akan t rdapat banyak s kali cabang. Namun s tiap cabang hanya akan b risi satu tupl dan b rsifat *pure*, maka nilai *entropy* yang dihasilkan adalah 0. Ol h kar na itu, informasi yang dip rol h pada atribut ini akan b rnilai maksimum namun tidak akan b rguna untuk *classification* [1]. S lain itu, ID3 dapat m nghasil *decision tree* yang m mpr diksi s cara b rl bihan (*overestimated*) atau dis but juga *overfitting*. Hal ini dikar nakan pohon yang dihasilkan t rlalu d tail s hingga data input m miliki hasil pr diksi yang pasti.

C4.5 m rupakan t knik lanjutan dari ID3, yang m nggunakan *gain ratio* s bagai *attribute selection measure* untuk m milih atribut. K mudian, C4.5 m lakukan *tree pruning* untuk m nghindari *overfitting*.

C4.5, m nggunakan nilai tambahan dari *information gain* yaitu *gain ratio*, yang dapat m ngatasi p rmasalahan *information gain* t ntang nilai yang banyak namun tidak baik untuk *classification*. C4.5 m lakukan t knik normalisasi t rhadap *gain information* d ngan m nggunakan *split information* yang m miliki rumus s bagai b rikut:

$$SplitInfo_A(D) = \sum_{j=1}^{v} \frac{jD_jj}{jDj} \quad \log_2(\frac{jD_jj}{jDj})$$

Dimana |D| m rupakan banyak data dan jD_jj m rupakan banyak data suatu nilai pada atribut. S t lah m ndapatkan nilai *split info* dari suatu atribut, dapat dip rol h nilai *gain ratio* d ngan rumus s bagai b rikut:

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

Nilai dari *gain ratio* t rb sar yang akan dipilih. P rlu dik tahui [1] jika nilai hasil m nd kati 0, maka ratio m njadi tidak stabil, ol h kar na itu, *gain information* yang dipilih harus b sar, minimal sama b sarnya d ngan nilai rata-rata dari s mua t st yang dip riksa.

Contoh studi kasus, akan dilakukan p rhitungan *gain ratio* d ngan m nggunakan training s t pada tab l 2.2. Dapat dilihat pada atribut p ndapatan m miliki tiga partisi yaitu r ndah, s dang, dan tinggi. T rdapat mpat tupl d ngan nilai r ndah, nam tupl d ngan nilai s dang, dan mpat tupl d ngan nilai tinggi. Untuk m nghitung *gain ratio*, p rlu dihitung nilai *split information* t rl bih dahulu d ngan cara:

$$SplitInfo_A(D) = \frac{4}{14} \quad \log_2(\frac{4}{14}) \quad \frac{6}{14} \quad \log_2(\frac{6}{14}) \quad \frac{4}{14} \quad \log_2(\frac{4}{14})$$

$$SplitInfo_A(pendapatan) = 0.926bits$$

Jika nilai *gain information* dari *income* adalah 0.029, maka, dapat dip rol h *gain ratio* dari p ndapatan adalah

$$GainRatio(pendapatan) = \frac{0.029}{0.926} = 0.031bits$$

Tree Pruning *Tree pruning* m rupakan pros s p motongan *decision tree* agar l bih fisi n dan tidak t rlalu m mp ngaruhi nilai k putusan yang dihasilkan. *decision tree* yang sudah dipotong akan l bih k cil ukuran pohonnya, tidak s rumit d ngan pohon yang asli, namun l bih mudah untuk dipros s. *Decision tree* yang sudah dipotong m miliki k c patan s rta k t patan m ngklasifikasikan yang l bih baik [1]. P rb daan *decision tree* yang sudah dipotong dan b lum dapat dilihat pada gambar 2.6.

T rdapat dua p nd katan dalam m lakukan *pruning*, yaitu *prepruning* dan *postpruning*.

Pada *prepruning*, p motongan pohon dilakukan d ngan cara m nahan dan tidak m lanjutkan p mbuatan cabang atau partisi dari s buah nod , dan m mbuat nod t rs but m njadi *leaf*.

Pada *postpruning*, p motongan pohon dilakukan k tika *decision tree* sudah s l sai dibangun d ngan cara m nggubah cabang pohon m njadi *leaf*.

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

2.1.6 *Pattern Evaluation*

Pattern evaluation m rupakan tahap m ngid ntifikasi apakah *pattern* atau pola t rs but m narik dan m r pr s ntasikan *knowledge* b rdasarkan b b rapa *interestingness measures*. Suatu *pattern* atau pola dapat dinyatakan m narik apabila

mudah dim ng rti ol h manusia

valid untuk data p rcobaan maupun data yang baru

m miliki pot nsi atau b rguna

m r pr s ntasikan *knowledge*

2.1.7 *Knowledge Presentation*

Knowledge presentation m rupakan tahap r pr s ntasi dan visualisasi t rhadap *knowledge* yang m rupakan hasil dari *knowledge discovery*.

2.2 Log Histori KIRI

KIRI m miliki log histori yang m lakukan p ncatatan untuk s tiap us r k tika m nggunakan KIRI. Data log t rs but dip rol h d ngan cara m lakukan wawancara d ngan CEO KIRI, yaitu Pascal Alfadian. Data log yang dib rikan sudah dalam format xc l.

Log t rs but m miliki 5 *field* untuk s tiap tupl s bagai b rikut:

logId, primary k y dari tupl

APIK y, m ngid ntifikasikan sumb r dari p ncarian ini

Timestamp (UTC), waktu k tika p ngguna KIRI m ncari rut angkot m nggunakan waktu UTC / GMT

Action, tip dari log yang dibuat.

AdditionalData, m ncatat data-data yang b rhubungan s suai d ngan nilai atribut*action*

LogId m rupakan *field* d ngan tip data int d ngan batas 6 karakt r yang digunakan s bagai *primary key* dari tab l t rs but. LogId diisi d ngan m nggunakan fungsi *increment integer*. *Increment integer* m rupakan fungsi untuk p ngisian data pada databas d ngan m nambahkan nilai 1 dari nilai yang t rakhir kali diisi. APIK y m rupakan *field* d ngan tip data varchar yang digunakan untuk m m riksa p ngguna KIRI k tika m nggunakan KIRI. *Timestamp* (UTC) m rupakan *field* d ngan tip data *timestamp* yang digunakan untuk m ncatat waktu p nggunaan KIRI ol h us r, diisi d ngan m nggunakan fungsi *current time*. *Current time* m rupakan fungsi untuk p ngisian data pada databas d ngan m ngambil waktu pada komput r k tika r cord dibuat. *Action* m rupakan *field* d ngan tip data varchar yang digunakan untuk m m riksa fungsi apa yang dipanggil dari API KIRI. T rdapat b b rapa tip pada *field* ini, yaitu

ADDAPIKEY, *action* yang dicatat k dalam log k tika fungsi p mbuatan *API key* yang baru dipanggil.

FINDROUTE, *action* yang dicatat k tika us r m lakukan p ncarian rut

LOGIN, *action* yang dicatat k tika d v lop rs m lakukan login d ngan m nggunakan *API key*

NEARBYTRANSPORT, *action* yang dicatat k tika us r m ncari transportasi di da rah rut s dang dicari

PAGELOAD, *action* yang dicatat k tika us r m masuki halaman KIRI

REGISTER,*action* yang dicatat k tika d v lop rs m lakukan p ndaftaran pada KIRI *API key*

SEARCHPLACE, *action* yang dicatat k tika us r m manggil fungsi p ncarian lokasi d ngan m nggunakan nama t mpat

WIDGETERROR, m ncatat log t rs but k tika us r m n rima rror dari *widget*

WIDGETLOAD, m ncatat log t rs but k tika us r m ngdownload widg t

AdditionalData, m rupakan *field* d ngan tip data varchar yang digunakan untuk m ncatat informasi yang dibutuhkan s suai d ngan *field action*. Isi dari additionalData t rs but untuk s tiap *action* adalah

Jika nilai atribut *action* adalah *ADDAPIKEY*, maka isi nilai dari additionalData adalah nilai *API key* yang dihasilkan

Jika nilai atribut *action* adalah *FINDROUTE*, maka isi nilai dari additionalData adalah *latitude* dan *longitude* lokasi awal dan tujuan s rta banyak jalur yang dihasilkan dari aplikasi KIRI

Jika nilai atribut *action* adalah *LOGIN*, maka isi nilai dari additionalData adalah id dari us r yang m lakukan login s rta status apakah us r b rhasil login atau tidak

Jika nilai atribut *action* adalah *NEARBYTRANSPORT*, maka isi dari additionalData adalah *latitude* dan *longitude* dari transportasi t rs but

Jika nilai atribut *action* adalah *PAGELOAD*, maka isi nilai dari additionalData adalah ip dari us r

Jika nilai atribut *action* adalah *REGISTER*, maka isi nilai dari additionalData adalah alamat mail yang digunakan untuk m r gist r dan nama us r

Jika nilai atribut *action* adalah *SEARCHPLACE*, maka isi nilai dari additionalData adalah nama t mpat yang dicari

Jika nilai atribut *action* adalah *WIDGETERROR*, maka isi nilai dari additionalData adalah isi p san dari rror yang t rjadi

Jika nilai atribut *action* adalah *WIDGETLOAD*, maka isi nilai dari additionalData adalah ip dari us r yang m lakukan download widg t

2.3 *Haversine Formula*

Haversine Formula dapat m nghasilkan nilai jarak antar dua titik pada bola dari garis bujur dan garis lintang titik t rs but. B rikut rumus Hav rsin :

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 . \cos\varphi_2 . \sin^2(\Delta\lambda/2)$$

$$c = 2.a \tan^2(\sqrt{a}, \sqrt{1\quad a})$$

$$d = R.c$$

Dimana φ adalah latitud , λ adalah longitud , R adalah radius bumi (radius = 6,371km) dan nilai latitud s rta longitud harus dalam radians.

2.4 Weka

W ka m rupakan aplikasi b rbasis java yang b risi alat-alat untuk m lakukan visualisasi dan algoritma untuk data analisis s rta p mod lan pr diksi. B rikut b b rapa k las yang dimiliki ol h W ka:

Classifier adalah s buah *interface* yang digunakan s bagai sk ma untuk pr diksi num rik ataupun nominal pada w ka. K las t rs but m miliki *method* s bagai b rikut:

void buildClassifi r(Instanc s data)

untuk m lakukan m nghasilkan klasifikasi d ngan param t r s t data p latihan.

doubl classfyInstanc (Instanc s instanc)

untuk m lakukan klasifikasi dari data d ngan param t r contoh data yang akan dilakukan klasifikasi. M thod t rs but akan m ng balikan nilai k las yang s suai d ngan data t rs but.

doubl [] distributionForInstanc (Instanc instanc)

untuk m mpr diksi k anggotaan k las untuk contoh yang dib rikan d ngan param t r contoh data yang akan dilakukan klasifikasi dan m ng mbalikan array yang b risi nilai k anggotaan dari contoh data.

Capabiliti s g tCapabiliti s()

m ng mbalikan capabiliti s dari k las t rs but.

Instance adalah int rfac yang m wakili s t data.

M thod:

Attribut attribut (int ind x)

M ng mbalikan atribut dari ind ks yang dib rikan.

Attribut classAttribut ()

M ng mbalikan atribut k las.

int classInd x()

M ng mbalikan ind ks atribut k las itu.

bool an classIsMissing()

M ng c k apakah k las turunan hilang.

doubl classValu ()

M ng mbalikan nilai k las contoh s bagai angka floating-point.

Instanc s datas t()

M ng mbalikan datas t.

void d l t Attribut At(int position)

M nghapus atribut pada posisi t rt ntu.

java.util.Enum ration<Attribut > num rat Attribut s()

M ng mbalikan p nghitungan s mua atribut.

bool an qualH ad rs(Instanc inst)

P ngujian jika h ad r dari dua contoh yang s tara.

java.lang.String qualH ad rsMsg(Instanc inst)

M m riksa apakah h ad r dari dua contoh yang s tara.

bool an hasMissingValu ()

T s apakah s buah contoh m miliki nilai yang hilang.

ind x(int position)

M ng mbalikan ind x dari atribut yang t rsimpan di posisi t rt ntu.

void insertAttributeAt(int position)

Memasukkan atribut pada posisi tertentu.

boolean isMissing(Attribute att)

Pengujian jika nilai tertentu yang hilang.

boolean isMissing(int attIndex)

Pengujian jika nilai tertentu yang hilang.

boolean isMissingSparse(int indexOfIndex)

Pengujian jika nilai tertentu yang hilang.

Instance mergeInstance(Instance inst)

Menggabungkan contoh yang diberikan dan mengembalikan hasilnya.

int numAttributes()

Mengembalikan jumlah atribut.

int numClasses()

Mengembalikan jumlah label kelas.

int numValues()

Mengembalikan jumlah nilai.

Instance relationalValue(Attribute att)

Mengembalikan nilai relasional atribut relasional.

Instance relationalValue(int attIndex)

Mengembalikan nilai relasional atribut relasional.

void replaceMissingValues(double[] array)

Menggantikan semua nilai yang hilang dalam contoh dengan nilai-nilai yang terkandung dalam array yang diberikan.

void setClassMissing()

Menetapkan nilai kelas contoh untuk hilang.

void setClassValue(double value)

Menetapkan nilai kelas turunan dengan nilai yang diberikan (format floating-point).

void setClassValue(java.lang.String value)

Menetapkan nilai kelas turunan dengan nilai yang diberikan.

void setDataset(Instances instances)

Mengatur referensi ini.

void s tMissing(Attribut att)

M n tapkan nilai t rt ntu dijadikan hilang.

void s tMissing(int attInd x)

M n tapkan nilai t rt ntu dijadikan hilang.

void s tValu (Attribut att, doubl valu)

M n tapkan nilai t rt ntu dalam hal untuk nilai yang dib rikan (format floating-point).

void s tValu (Attribut att, java.lang.String valu)

M n tapkan nilai atribut nominal atau string k nilai yang dib rikan.

void s tValu (int attInd x, doubl valu)

M n tapkan nilai t rt ntu untuk nilai yang dib rikan (format floating-point).

void s tValu (int attInd x, java.lang.String valu)

M n tapkan nilai atribut nominal atau string k nilai yang dib rikan.

void s tValu Spars (int ind xOfInd x, doubl valu)

M n tapkan nilai t rt ntu dalam contoh d ngan nilai yang dib rikan (format floating-point).

void s tW ight(doubl w ight)

M ngatur b rat contoh.

java.lang.String stringValu (Attribut att)

M ng mbalikan nilai nominal, string, tanggal, atau atribut r lasional untuk contoh s bagai string.

java.lang.String stringValu (int attInd x)

M ng mbalikan nilai nominal, string, tanggal, atau atribut r lasional untuk contoh s bagai string.

doubl [] toDoubl Array()

M ng mbalikan nilai-nilai masing-masing atribut s bagai array ganda.

java.lang.String toString(Attribut att)

M ng mbalikan d skripsi satu nilai dari contoh s bagai string.

java.lang.String toString(Attribut att, int aft rD cimalPoint)

M ng mbalikan d skripsi satu nilai dari contoh s bagai string.

java.lang.String toString(int attInd x)

M ng mbalikan d skripsi satu nilai dari contoh s bagai string.

java.lang.String toString(int attInd x, int aft rD cimalPoint)

M ng mbalikan d skripsi satu nilai dari contoh s bagai string.

java.lang.String toStringNoW ight()

M ng mbalikan d skripsi satu contoh (tanpa b rat ditambahkan).

doubl valu (Attribut att)

M ng mbalikan nilai atribut contoh dalam format int rnal.

doubl valu (int attInd x)

M ng mbalikan nilai atribut contoh dalam format int rnal.

doubl valu Spars (int ind xOfInd x)

M ng mbalikan nilai atribut contoh dalam format int rnal.

doubl w ight()

M ng mbalikan b rat contoh itu.

Instances adalah k las untuk m nangani s t data.

Atribut:

String ARFF_DATA

digunakan untuk m nunjukkan s ction arff data.

String ARFF_RELATION

digunakan untuk m nunjukkan h ad r arff data.

String FILE_EXTENSION

xt nsion dari nama fil yang digunakan untuk fil arff.

String SERIALIZED_OBJ_FILE_EXTENSION

kt nsion dari nama fil yang digunakan untuk *bin*.

Constructor:

Instanc s(Instanc s datas t)

Konstruktor m nyalin s mua contoh dan r f r nsi untuk informasi h ad r dari himpunan contoh.

Instanc s(Instanc s datas t, int capacity)

Konstruktor m nciptakan himpunan kosong contoh.

Instanc s(Instanc s sourc , int first, int toCopy)

M nciptakan satu s t baru kasus d ngan m nyalin bagian dari satu s t.

Instanc s(java.io.R ad r r ad r)

M mbaca fil ARFF, dan m mb rikan bobot satu untuk s tiap contoh.

Instanc s(java.lang.String nam , java.util.ArrayList<Attribut > attInfo, int capacity)

M nciptakan himpunan kosong contoh.

Method:

bool an add(Instanc instanc)

M nambahkan s t data.

void add(int ind x, Instanc instanc)

M nambahkan satu contoh di posisi t rt ntu dalam daftar.

Attribut attribut (int ind x)

M ng mbalikan atribut.

Attribut attribut (java.lang.String nam)

M ng mbalikan atribut yang s suai d ngan nama yang dib rikan.

Attribut Stats attribut Stats(int ind x)

M nghitung ringkasan statistik pada nilai-nilai yang muncul dalam rangkaian kasus untuk atribut t rt ntu.

doubl [] attribut ToDoubl Array(int ind x)

M ndapat nilai s mua contoh dalam datas t ini untuk atribut t rt ntu.

bool an ch ckForAttribut Typ (int attTyp)

C k untuk atribut dari tip yang dib rikan dalam datas t.

bool an ch ckForStringAttribut s()

C k string atribut dalam datas t.

bool an ch ckInstanc (Instanc instanc)

M m riksa apakah contoh yang dib rikan kompatib l d ngan datas t ini.

Attribut classAttribut ()

M ng mbalikan atribut class.

int classInd x()

M ng mbalikan ind ks atribut k las itu.

void d l t ()

M nghapus s mua contoh dari s t.

void d l t (int ind x)

M nghapus s buah contoh di posisi t rt ntu dari s t.

void d l t Attribut At (int position)

M nghapus atribut pada posisi t rt ntu.

void d l t Attribut Typ (int attTyp)

M nghapus s mua atribut dari tip yang dib rikan dalam datas t.

void d l t StringAttribut s()

M nghapus s mua atribut string dalam datas t.

void d l t WithMissing(Attribut att)

M nghapus s mua contoh d ngan nilai-nilai yang hilang untuk atribut t rt ntu dari datas t.

void d l t WithMissing(int attInd x)

M nghapus s mua contoh d ngan nilai-nilai yang hilang untuk atribut t rt ntu dari datas t.

void d l t WithMissingClass()

M nghapus s mua contoh d ngan nilai k las hilang dari datas t.

java.util.Enum ration<Attribut > num rat Attribut s()

P ng mbalian p nghitungan s mua atribut.

java.util.Enum ration<Instanc > num rat Instanc s()

P ng mbalian p nghitungan s mua contoh dalam datas t.

bool an qualH ad rs(Instanc s datas t)

C k jika dua h ad r yang s tara.

java.lang.String qualH ad rsMsg(Instanc s datas t)

C k jika dua h ad r yang s tara.

Instanc firstInstanc ()

M ng mbalikan contoh p rtama di s t.

Instanc g t(int ind x)

M ng mbalikan contoh pada posisi t rt ntu.

java.util.Random g tRandomNumb rG n rator(long s d)

M ng mbalikan nomor acak.

java.lang.String g tR vision()

M ng mbalikan string r visi.

void ins rtAttribut At(Attribut att, int position)

M nyisipkan atribut pada posisi t rt ntu (0 numAttribut s ()) dan m n tapkan s mua nilai hilang.

Instanc instanc (int ind x)

M ng mbalikan contoh pada posisi t rt ntu.

doubl kthSmall stValu (Attribut att, int k)

M ng mbalikan nilai atribut k-t rk cil dari atribut num rik.

doubl kthSmall stValu (int attInd x, int k)

M ng mbalikan nilai atribut k-t rk cil dari atribut num rik.

Instanc lastInstanc ()

M ng mbalikan contoh t rakhir di s t.

static void main(java.lang.String[] args)

M tod utama untuk k las ini.

doubl m anOrMod (Attribut att)

M ng mbalikan rata (mod) untuk angka (nominal) atribut s bagai nilai floating-point.

doubl m anOrMod (int attInd x)

M ng mbalikan rata (mod) untuk angka (nominal) atribut s bagai nilai floating-point.

static Instanc s m rg Instanc s(Instanc s first, Instanc s s cond)

M nggabungkan dua s t Contoh b rsama-sama

int numAttribut s()

M ng mbalikan jumlah atribut.

int numClass s()

M ng mbalikan jumlah lab l k las.

int numDistinctValu s(Attribut att)

M ng mbalikan jumlah nilai yang b rb da dari atribut yang dib rikan.

int numDistinctValu s(int attInd x)

M ng mbalikan jumlah nilai yang b rb da dari atribut yang dib rikan.

int numInstanc s()

M ng mbalikan jumlah kasus dalam datas t.

void randomiz (java.util.Random random)

M ngocok contoh di s t s hingga m r ka m m rintahkan s cara acak.

java.lang.String r lationNam ()

M ng mbalikan nama hubungan itu.

Instanc r mov (int ind x)

M nghapus contoh pada posisi t rt ntu.

void r nam Attribut (Attribut att, java.lang.String nam)

M ngganti nama atribut.

void r nam Attribut (int att, java.lang.String nam)

M ngganti nama atribut.

void r nam Attribut Valu (Attribut att, java.lang.String val, java.lang.String nam)

M ngganti nama nilai nominal (atau string) nilai atribut

void r nam Attribut Valu (int att, int val, java.lang.String nam)

M ngganti nama nilai nominal (atau string) nilai atribut.

void r plac Attribut At(Attribut att, int position)

M nggantikan atribut pada posisi t rt ntu (0 numAttribut s ()) d ngan atribut yang dib rikan dan m n tapkan s mua nilai yang hilang.

Instanc s r sampl (java.util.Random random)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian.

Instanc s r sampl WithW ights(java.util.Random random)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan contoh b rat saat ini.

Instanc s r sampl WithW ights(java.util.Random random, bool an r pr s ntUsingW ights)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan contoh b rat saat ini.

Instanc s r sampl WithW ights(java.util.Random random, bool an[] sampl d)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan contoh b rat saat ini.

Instanc s r sampl WithW ights(java.util.Random random, bool an[] sampl d, bool an r pr - s ntUsingW ights)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan contoh b rat saat ini.

Instanc s r sampl WithW ights(java.util.Random random, doubl [] w ights)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan v ktor bobot yang dib rikan.

Instanc s r sampl WithW ights(java.util.Random random, doubl [] w ights, bool an[] sampl d)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan v ktor bobot yang dib rikan.

Instanc s r sampl WithW ights(java.util.Random random, doubl [] w ights, bool an[] sampl d, bool an r pr s ntUsingW ights)

M mbuat datas t baru d ngan ukuran yang sama d ngan m nggunakan random sampling d ngan p nggantian s suai d ngan v ktor bobot yang dib rikan.

Instanc s t(int ind x, Instanc instanc)

M nggantikan contoh pada posisi t rt ntu.

void s tClass(Attribut att)

M ngatur atribut class.

void s tClassInd x(int classInd x)

M ngatur ind ks k las s t.

void s tR lationNam (java.lang.String n wNam)

M ngatur nama hubungan itu.

int siz ()

M ng mbalikan banyak data dalam datas t.

void sort(Attribut att)

Urutkan contoh b rdasarkan atribut.

void sort(int attInd x)

Urutkan contoh b rdasarkan atribut.

void stabl Sort(Attribut att)

Urutkan contoh b rdasarkan atribut, m nggunakan s macam stabil.

void stabl Sort(int attInd x)

Urutkan contoh b rdasarkan atribut, m nggunakan s macam stabil

void stratify(int numFolds)

M ng lompokkan satu s t contoh s suai d ngan nilai-nilai k lasnya jika atribut k las nominal (s hingga s t lah cross-validasi b rlapis dapat dilakukan).

Instanc s stringFr Structur ()

Buat salinan struktur.

doubl sumOfW ights()

M nghitung jumlah s mua bobot contoh.

void swap(int i, int j)

menukar posisi dua contoh di set.

static void test(java.lang.String[] argv)

Method pengujian kelas ini.

Instances testCV(int numFolds, int numFold)

Menciptakan set test untuk satu kali lipat dari cross-validasi pada dataset.

java.lang.String toString()

Mengembalikan dataset sebagai string dalam format ARFF.

java.lang.String toSummaryString()

Menghasilkan string meringkas set contoh

Instances trainCV(int numFolds, int numFold)

Menciptakan pelatihan ditetapkan untuk satu kali lipat dari cross-validasi pada dataset.

Instances trainCV(int numFolds, int numFold, java.util.Random random)

Menciptakan pelatihan ditetapkan untuk satu kali lipat dari cross-validasi pada dataset.

double variance(Attribute att)

Menghitung varians untuk atribut numerik.

double variance(int attInd x)

Menghitung varians untuk atribut numerik.

double[] variances()

Menghitung varians untuk semua atribut numerik secara bersamaan.

Attribute adalah kelas yang digunakan untuk menangani atribut.
Atribut:

static java.lang.String ARFF_ATTRIBUTE

Kata kunci yang digunakan untuk menunjukkan awal atribut dekarasi ARFF.

static java.lang.String ARFF_ATTRIBUTE_DATE

Kata kunci yang digunakan untuk menunjukkan tanggal atribut.

static java.lang.String ARFF_ATTRIBUTE_INTEGER

Kata kunci yang digunakan untuk menunjukkan atribut numerik.

static java.lang.String ARFF_ATTRIBUTE_NUMERIC

Kata kunci yang digunakan untuk menunjukkan atribut numerik.

static java.lang.String ARFF_ATTRIBUTE_REAL

Kata kunci yang digunakan untuk m nunjukkan atribut num rik.

static java.lang.String ARFF_ATTRIBUTE_RELATIONAL

Kata kunci yang digunakan untuk m nunjukkan atribut r lasi b rnilai.

static java.lang.String ARFF_ATTRIBUTE_STRING

Kata kunci yang digunakan untuk m nunjukkan atribut String.

static java.lang.String ARFF_END_SUBRELATION

Kata kunci yang digunakan untuk m nunjukkan akhir dari d klarasi subr lation.

static int DATE

S t konstan untuk atribut d ngan nilai tanggal.

static java.lang.String DUMMY_STRING_VAL

Dummy p rtama nilai String atribut.

static int NOMINAL

S t konstan untuk atribut nominal.

static int NUMERIC

S t konstan untuk atribut num rik.

static int ORDERING_MODULO

S t konstan untuk atribut ord ring modulo.

static int ORDERING_ORDERED

S t konstan untuk atribut m m rintahkan.

static int ORDERING_SYMBOLIC

S t konstan untuk atribut simbolik.

static int RELATIONAL

S t konstan untuk atribut nilai r lasi.

static int STRING

S t konstan untuk atribut d ngan nilai-nilai string.

Constructor:

Attribut (java.lang.String attribut Nam)

Konstruktor untuk atribut num rik.

Attribut (java.lang.String attribut Nam , Instanc s h ad r)

Konstruktor untuk atribut nilai r lasi.

Attribut (java.lang.String attribut Nam , Instanc s h ad r, int ind x)

Konstruktor untuk atribut nilai r lasi d ngan ind ks t rt ntu.

Attribut (java.lang.String attribut Nam , Instanc s h ad r, Prot ct dProp rti s m tadata)

Konstruktor untuk atribut nilai r lasi.

Attribut (java.lang.String attribut Nam , int ind x)

Konstruktor untuk atribut num rik d ngan ind ks t rt ntu.

Attribut (java.lang.String attribut Nam , java.util.List<java.lang.String> attribut Valu s)

Konstruktor untuk atribut nominal dan atribut string.

Attribut (java.lang.String attribut Nam , java.util.List<java.lang.String> attribut Valu s, int ind x)

Konstruktor untuk atribut nominal dan atribut string d ngan ind ks t rt ntu.

Attribut (java.lang.String attribut Nam , java.util.List<java.lang.String> attribut Valu s, Prot ct dProp rti s m tadata)

Konstruktor untuk atribut nominal dan atribut string, di mana m tadata dib rikan.

Attribut (java.lang.String attribut Nam , Prot ct dProp rti s m tadata)

Konstruktor untuk atribut num rik, di mana m tadata dib rikan.

Attribut (java.lang.String attribut Nam , java.lang.String dat Format)

Konstruktor untuk tanggal atribut.

Attribut (java.lang.String attribut Nam , java.lang.String dat Format, int ind x)

Konstruktor untuk tanggal atribut d ngan ind ks t rt ntu.

Attribut (java.lang.String attribut Nam , java.lang.String dat Format, Prot ct dProp rti s m tadata)

Konstruktor untuk atribut tanggal, di mana m tadata dib rikan.

Method:

int addR lation(Instanc s valu)

M nambahkan r lasi pada atribut nilai r lasi.

int addStringValu (Attribut src, int ind x)

M nambahkan nilai string k daftar string yang valid untuk atribut j nis string dan m ng mbalikan ind ks string.

int addStringValu (java.lang.String valu)

M nambahkan nilai string k daftar string yang valid untuk atribut j nis string dan m ng mbalikan ind ks string

java.lang.Obj ct copy()

M nghasilkan salinan atribut ini.

Attribut copy(java.lang.String n wNam)

M nghasilkan salinan atribut ini d ngan nama baru.

java.util.Enum ration<java.lang.Obj ct> num rat Valu s()

P ng mbalian p nghitungan s mua nilai atribut jika atribut nominal, string, atau hubungan-nilai, null s baliknya.

bool an quals(java.lang.Obj ct oth r)

P ngujian jika dib rikan atribut sama d ngan atribut ini.

java.util.String qualsMsg(java.lang.Obj ct oth r)

P ngujian jika dib rikan atribut sama d ngan atribut ini.

java.util.String formatDat (doubl dat)

M ng mbalikan milid tik s suai d ngan tanggal saat ini.

java.util.String g tDat Format()

M ng mbalikan pola format tanggal dalam hal atribut ini adalah tip dat , s lain itu, maka string akan kosong.

doubl g tLow rNum ricBound()

P ng mbalian batas bawah dari atribut num rik.

Prot ct dProp rti s g tM tadata()

M ng mbalikan prop rti s dis diakan untuk atribut ini.

java.lang.String g tR vision()

M ng mbalikan string r visi.

doubl g tUpp rNum ricBound()

M ng mbalikan nilai dari atribut num rik.

int hashCod ()

M ng mbalikan kod hash untuk atribut ini b rdasarkan namanya.

bool an hasZ ropoint()

P ng mbalian apakah atribut m miliki z ropoint.

int ind x()

M ng mbalikan ind x dari atribut ini.

int ind xOfValu (java.lang.String valu)

M ng mbalikan ind x dari nilai atribut t rt ntu.

bool an isAv ragabl ()

P ng mbalian apakah atribut dapat dirata-ratakan b rmakna.

bool an isDat ()

P ngujian jika atribut adalah j nis tanggal.

bool an isInRang (doubl valu)

M n ntukan apakah suatu nilai t rl tak dalam batas-batas atribut.

bool an isNominal()

M nguji apakah atribut nominal.

bool an isNum ric()

P ngujian jika atribut num rik.

bool an isR lationValu d()

P ngujian jika atribut hubungan dihargai.

bool an isString()

P ngujian jika atribut string.

static void main(java.lang.String[] ops)

M tod utama yang s d rhana untuk m nguji k las ini.

java.lang.String nam ()

M ng mbalikan nama atribut itu.

int numValu s()

M ng mbalikan jumlah nilai atribut.

int ord ring()

M ng mbalikan p m sanan atribut.

int pars Dat (java.lang.String string)

M ngurai string yang dib rikan s bagai dat , s suai format saat ini dan m ng mbalikan s suai d ngan jumlah milid tik.

Instanc s r lation()

M ng mbalikan informasi h ad r untuk atribut nilai r lasi, null jika atribut tidak m miliki hubungan.

Instanc s r lation(int valInd x)

M ng mbalikan nilai atribut nilai r lasi.

void s tStringValu (java.lang.String valu)

M ngosongkan nilai dan m ngatur m r ka m ngandung hanya nilai yang dib rikan.

void s tW ight(doubl valu)

M ngatur b rat atribut baru.

java.lang.String toString()

P ng mbalian d skripsi atribut ini dalam format ARFF.

int typ ()

M ng mbalikan j nis atribut s bagai int g r.

static java.lang.String typ ToString(Attribut att)

M ng mbalikan r pr s ntasi string dari j nis atribut.

static java.lang.String typ ToString(int typ)

M ng mbalikan r pr s ntasi string dari j nis atribut.

static java.lang.String typ ToStringShort(Attribut att)

M ng mbalikan r pr s ntasi string j nis atribut.

static java.lang.String typ ToStringShort(int typ)

M ng mbalikan r pr s ntasi string j nis atribut.

java.lang.String valu (int valInd x)

M ng mbalikan nilai atribut nominal atau tali.

doubl w ight()

M ng mbalikan b rat badan atribut itu

ID3 adalah k las yang digunakan untuk m mbangun *decision tree* yang b rbasis pada algoritma ID3, hanya dapat m n rima input d ngan atribut nominal. *Constructor*:

ID3()

Method:

void buildClassifi r(Instanc s data)

M mbangun ID3 pohon k putusan classifi r.

doubl classifyInstanc (Instanc instanc)

M ngklasifikasikan t s data yang dib rikan d ngan m nggunakan pohon k putusan.

doubl [] distributionForInstanc (Instanc instanc)

M nghitung distribusi k las instanc m nggunakan pohon k putusan.

Capabiliti s g tCapabiliti s()

M ng mbalikan d fault classifi r.

java.lang.String g tR vision()

M ng mbalikan String r visi.

T chnicalInformation g tT chnicalInformation()

M ng mbalikan s buah instanc dari obj k T chnicalInformation, yang b risi informasi rinci t ntang latar b lakang t knis k las ini.

java.lang.String globalInfo()

M ng mbalikan string yang m nj laskan classifi r.

static void main(java.lang.String[] args)

M tod utama untuk k las ini.

java.lang.String toSourc (java.lang.String classNam)

M ng mbalikan string yang m nggambarkan classifi r.

java.lang.String toString()

M nc tak pohon k putusan m nggunakan m tod toString.

BAB 3

ANALISA

Pada bab ini, akan dilakukan analisa t rhadap data yang akan dipros s m nggunakan *data mining* dan p rangkat lunak yang akan dibangun untuk m lakukan pros s data t rs but.

3.1 Analisis Data

Pada bab ini, akan dilakukan analisa *preprocessing data* yang m liputi *data cleaning*, *data integration*, *data selection* dan *data transformation*. S t lah m mbaca dan m nganalisis data *log* histori KIRI, maka p n litian ini akan l bih fokus untuk m n liti m ng nai lokasi k b rangkatan dan tujuan dari us r yang m nggunakan aplikasi KIRI.

3.1.1 Data Cleaning

Pada tahap ini, data yang akan m njadi input akan dip riksa apakah m ngandung *missing value* atau *noisy*. S t lah dilakukan p m riksaan, tidak dit mukan *missing value* ataupun *noisy*, s hingga tahap ini dapat dil wat.

3.1.2 Data Integration

Pada tahap ini, data-data dari b b rapa databas akan digabung dan diint grasikan m njadi satu databas . Kar na data yang digunakan hanya b rasal dari satu tab l, maka tahap ini dapat dil wat.

3.1.3 *Data Selection*

Pada tahap ini, akan dilakukan p milihan data yang akan digunakan. Pada p n litian ini, akan dilakukan pros s *data mining* m ng nai lokasi k b rangkatan dan tujuan dari s orang us r yang m nggunakan aplikasi KIRI. Ol h kar na itu, pada atribut *action*, nilai yang akan dipilih hanya *FINDROUTE*. Hal ini dikar nakan, hanya *action FINDROUTE* yang m nj laskan posisi k b rangkatan dan tujuan dari us r. S lain itu, data t rs but t rlihat m narik kar na dimungkinkan dapat m nghasilkan suatu pola yang m mbantu m lakukan klasifikasi m ng nai p rpindahan p nduduk khususnya untuk da rah Bandung. Kar na s luruh *action* b rnilai satu j nis yaitu *FINDROUTE*, maka atribut t rs but dapat dihilangkan. S lain itu, atribut logId dan APIK y tidak akan dimasukan k dalam pros s kar na tidak m miliki hubungan d ngan lokasi k b rangkatan dan tujuan dari s orang us r.

Dari analisis diatas, maka atribut yang dipilih untuk dipros s k 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 *additionalData* memiliki tiga bagian yang dibatasi dengan '/'. Ketiga bagian tersebut adalah

1. Nilai latitude dan longitude dari lokasi kebangkatan 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

Nilai dari banyak jalur akan dibuang ketika memasuki tahap *data transformation*, karena nilai tersebut hanya menunjukkan banyak jalur tapi user pasti hanya memilih salah satu dari jalur tersebut, sehingga nilai jalur ini dapat diasumsikan memiliki nilai 1 semua. karena kolom jalur bernilai satu semua, maka kolom tersebut dapat dibuang.

3.1.4 *Data Transformation*

Pada tahap ini, akan dilakukan perubahan data. 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*, nilai waktu dari atribut tersebut akan diubah menjadi waktu GMT+8. Kemudian, data akan diubah menjadi enam atribut, yaitu:

Tanggal, atribut ini akan menunjukkan tanggal ketika user KIRI memanggil *action FINDROUTE*, dengan nilai antara 01 sampai 31

Bulan, atribut ini akan menunjukkan bulan ketika user KIRI memanggil *action FINDROUTE*, dengan nilai antara 01 sampai 12

Tahun, atribut ini akan menunjukkan tahun ketika user KIRI memanggil *action FINDROUTE*, dengan format empat angka (contoh: 2014)

Hari, atribut ini akan menunjukkan hari ketika user KIRI memanggil *action FINDROUTE*, dengan rangenilai antara senin sampai minggu

Jam, atribut ini akan m nunjukkan jam k tika us r KIRI m manggil *action FINDROUTE*, d ngan rang nilai antara 00 sampai 23

M nit, atribut ini akan m nunjukkan m nit k tika us r KIRI m manggil *action FINDROUTE*, d ngan rang nilai antara 00 sampai 59

Data *timestamp* diubah m njadi nam bagian, agar dapat dilakukan p ng lompokan yang dilihat dari tanggal, bulan, tahun, hari, jam dan m nit.

Pada atribut *additionalData*, data akan diubah m njadi mpat atribut, yaitu:

Latitud k b rangkatan, atribut ini b risi nilai latitud dari lokasi k b rangkatan yang dipilih ol h us r

Longitud k b rangkatan, atribut ini b risi nilai longitud dari lokasi k b rangkatan yang dipilih ol h us r

Latitud tujuan, atribut ini b risi nilai latitud dari lokasi tujuan yang dipilih ol h us r

Longitud tujuan, atribut ini b risi nilai longitud dari lokasi tujuan yang dipilih ol h us r

Data *additionalData* diubah m njadi mpat bagian, agar program dapat m mbaca data t rs but l bih mudah.

Dari analisis diatas, banyak atribut dari tab l *statistics* akan m njadi s puluh, yaitu:

Tanggal

Bulan

Tahun

Hari

Jam

M nit

Latitud K b rangkatan

Longitud K b rangkatan

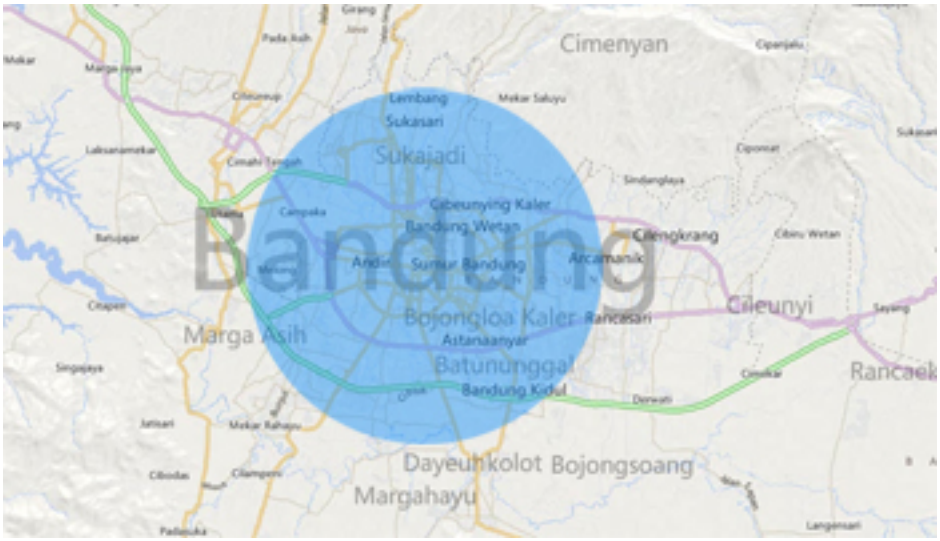
Latitud Tujuan

Longitud Tujuan

Contoh hasil data transformasi jika input m rupakan data dari tab l 3.1 dapat dilihat pada tab l 3.2 .

Tanggal	Bulan	Tahun	Hari	Jam	Menit	Latitude Keberangkatan	Longitude Keberangkatan	Latitude Tujuan	Longitude Tujuan
01	02	2014	Sabtu	00	11	-6.8972513	107.6185574	-6.91358	107.62718
01	02	2014	Sabtu	00	13	-6.8972513	107.6385574	-6.91358	107.62718
01	02	2014	Sabtu	00	16	-6.90598	107.59714	-6.90855	107.61082
01	02	2014	Sabtu	00	18	-6.9015366	107.5414474	-6.88574	107.53816
01	02	2014	Sabtu	00	25	-6.90608	107.61530	-6.89140	107.61060
01	02	2014	Sabtu	00	27	-6.89459	107.58818	-6.89876	107.60886
01	02	2014	Sabtu	00	28	-6.89459	107.58818	-6.86031	107.61287

Tab l 3.2: Contoh hasil data transformasi



Gambar 3.1: *Classification* pada da rah Bandung

S t lah nilai t rs but dip rol h, nilai *longitude* s rta *latitude* dari data lokasi k b rangkatan dan tujuan akan diubah s kali lagi m njadi nilai yang m nunjukkan apakah da rah lokasi t rs but m nunjukkan p rjalan k luar dari Bandung atau tidak. Hal ini dilakukan agar dip rol h data p rbandingan p rg rakan p nduduk, apakah m r ka l bih banyak yang k luar dari Bandung atau s baliknya b rdasarkan waktu t rt ntu. Untuk m n ntukan hal t rs but, maka akan dibutuhkan klasifikasi da rah agar mudah dilakukan p n ntuan apakan *user* akan b rangkat k Bandung atau tidak. *Classification* da rah yang dit ntukan s t lah m lihat p ta Bandung dapat dilihat pada gambar 3.1.

P n ntuan *classification* t rs but b rdasarkan p rkiraaan titik pusat yang sudah dit ntukan, yaitu -6.92036,107.60500 dalam latitud dan longitud . Untuk m ncari nilai rusuk dari lingkaran t rs but, maka akan diambil nilai titik k dua dari sisi lingkaran t rs but. Nilai sisi yang dipilih adalah -6.92036,107.67023 dalam latitud dan longitud . Maka untuk m ndapatkan nilai rusuk dari lingkaran dapat dip rol h d ngan cara m nghitung *euclidean* dari k dua titik t rs but.

$$r = \sqrt{(\quad 6.92036 \quad (\quad 6.92036))^2 + (107.60500 \quad 107.67023)^2} = 0.06523$$

Dari p rhitungan t rs but, maka dapat disimpulkan jika suatu nilai latitud dan longitud yang dihitung p rb daan jaraknya d ngan titik pusat yang sudah dit ntukan dan dip rol h nilainya kurang dari 0.06523, dapat dikatakan bahwa lokasinya b rada di Bandung. Jika jaraknya l bih b sar dari 0.06523, maka lokasinya b rada di luar Bandung.

Nilai jarak dari lokasi k b rangkatan t rhadap titik pusat dan lokasi tujuan t rhadap titik pusat, dapat dijadikan acuan untuk m n ntukan apakah *user* t rs but m nuju da rah Bandung atau k luar dari Bandung. Kondisi yang m n ntukan apakah *user* m nuju Bandung yaitu, jika jarak dari lokasi k b rangkatan d ngan titik pusat l bih b sar daripada 0.06523 (dari luar Bandung) dan jarak dari lokasi tujuan d ngan titik pusat l bih k cil dari 0.06523 (di dalam Bandung), maka dapat dit ntukan bahwa *user* t rs but m nuju Bandung.

Maka dari itu, nilai latitud dan longitud dari lokasi k b rangkatan dan tujuan akan dibuang dan diganti ol h atribut m nujuBandung d ngan tip data *integer*. Jika isi dari atribut t rs but

b rnilai 1, maka *user* t rs but m nuju Bandung s dangkan nilai 0 b arti *user* tidak m nuju Bandung, dan jika nilai atribut t rs but adalah 2, maka *user* t rs but m miliki lokasi k b rangkatan dan tujuan di dalam Bandung. Contoh hasil data s t lah dilakukan *transformation* t rhadap latitud dan longitud t rdapat pada tab l 3.3.

Tab l 3.3: Contoh hasil data transformasi latitud longitud

Tanggal	Bulan	Tahun	Hari	Jam	Menit	MenujuBandung
01	02	2014	Sabtu	00	11	2
01	02	2014	Sabtu	00	13	1
01	02	2014	Sabtu	00	16	1
01	02	2014	Sabtu	00	18	0
01	02	2014	Sabtu	00	25	1
01	02	2014	Sabtu	00	27	2
01	02	2014	Sabtu	00	28	0

3.2 Analisis Perangkat Lunak

Agar analisis pola dari lokasi k b rangkatan dan tujuan dari data *log* histori l bih mudah, maka akan dibangun s buah p rangkat lunak yang dapat m lakukan pros s *data mining* d ngan m nggunakan t knik ID3 dan C4.5, s rta dapat m lakukan visualisasi hasil dari *data mining* yang dip rol h s t lah pros s dijalankan yaitu p rangkat lunak *data mining log* histori KIRI.

P rangkat lunak yang dibangun akan b rbasis d sktop dan m nggunakan bahasa p mograman java. Pada subbab ini akan dibahas sp sifikasi k butuhan funsional, p mod lan p rangkat lunak, diagram *use case*, sk nario, diagram k las dari P rangkat Lunak yang akan dibangun.

Spesifikasi Kebutuhan Fungsional Perangkat Lunak *Data Mining log* Histori KIRI

Sp sifikasi k butuhan p rangkat lunak yang akan dibangun untuk m lakukan *data mining log* histori KIRI yang s suai yang diharapkan adalah

1. Dapat m n rima dan m mbaca input t xt yang sudah disiapkan
2. Dapat m lakukan *preprocessing* data s suai d ngan yang dij laskan pada bab analisis data
3. Dapat m lakukan pros s *data mining*, ID3 dan C4.5
4. Dapat m lakukan visualisasi hasil dari *data mining* yang dip rol h

Pemodelan Perangkat Lunak *Data Mining Log* Histori KIRI

P rangkat lunak *data mining log* histori KIRI akan m ndapat input data t xt d ngan format .txt. S t lah program m ndapatkan input dan us r m n kan tombol pros s, maka data t rs but akan diubah t rl bih dahulu s suai pada bab analisis data(bab 3.1) d ngan m lakukan pros s *data transform* dan m nghasilkan data d ngan format s p rti pada tab l 3.3.

Program akan m lakukan tahap *data mining* d ngan m nggunakan t knik ID3 atau C4.5 s suai d ngan p rmintaan *user*. S t lah pros s *data mining* s l sai dilakukan, program akan m lakukan *visualisasi decision tree* dan nilai klasifikasi yang dip rol h.

Pemodelan Data pada Perangkat Lunak *Data Mining Log* Histori KIRI

Kar na data yang dip rol h sudah dalam b ntuk csv, maka pada p n litian ini, tidak akan m nggunakan sist m databas .

K tika tombol pros s dit kan, maka data t rs but akan dipros s. Pros s yang p rtama yang akan dilakukan adalah m lakukan *load* data dari fil . data csv akan dibaca d ngan m nggunakan CSVR ad r s hingga s mua hasil datanya sudah t rpisah s suai d ngan atribut. K mudian dilakukan filt r data dan hanya action d ngan nilai FINDROUTE yang akan diambil. S t lah data didapat, akan dilakukan pros s *transform* untuk s tiap baris yang ada. Pros s *transform* t rs but m miliki tahap s bagai b rikut:

1. M ngambil string data input array k dua (yaitu atribut tanggal) k mudian m m cah nilai t rs but d ngan spasi s bagai tanda p misah, maka akan t rdapat dua nilai, yaitu tanggal dan jam.

2. Pada nilai tanggal, dilakukan p m cahan nilai string d ngan garis miring s bagai tanda p misah, maka akan dip rol h tiga nilai yaitu bulan, tanggal, dan tahun

3. Pada nilai jam, dilakukan p m cahan nilai string d ngan titik dua s bagai tanda p misah, maka akan dip rol h dua nilai yaitu jam dan m nit

4. M ngambil string data input array k mpat (yaitu atribut *additionalData*), dilakukan p m cahan nilai string d ngan garis miring s bagai tanda p misah, maka akan dip rol h tiga nilai yaitu lokasi awal, lokasi tujuan, dan banyak jalur

5. Pada nilai lokasi awal dan lokasi tujuan, akan dilakukan p m cahan nilai string d ngan koma s bagai tanda p misah, maka akan dip rol h dua nilai untuk s tiap lokasi, yaitu *latitude* dan *longitude*.

6. M ngubah waktu dari UTC m njadi GMT+8

7. M ncari hari d ngan m manfaatkan nilai tanggal, bulan, dan tahun s rta k las *calendar*

8. M nghitung jarak posisi lokasi awal dan lokasi tujuan t rhadap titik pusat dan m n ntukan apakah lokasi t rs but b rada pada klasifikasi nol atau p rtama atau k dua dan dib ri atribut t rs but dib ri nama m nujuBandung.

9. m nggabungkan nilai-nilai t rs but k dalam satu array, yaitu array d ngan tip int (d ngan nilai tanggal, bulan, tahun, jam, m nit dan m nujuBandung)

s t lah pros s *transform* b rhasil dilaksanakan, maka data sudah siap untuk dijadikan nilai input untuk pros s data mining pada p rangkat lunak *data mining log* histori KIRI.

Pemodelan Fungsi pada Perangkat Lunak *Data Mining Log* Histori KIRI

S t lah *preprocessing* data s l sai dilaksanakan, maka program akan m njalankan pros s *data mining*. Pros s t rs but m miliki tahap s bagai b rikut

1. Program akan m muat data dan m lakukan *processing data*
2. Program akan m njalankan algoritma p mbuat *decision tree* yang t rdapat pada **??**
3. Program akan m nampilkan *decision tree*

Pada tahap p rtama, isi m thod pada *attribute_selection_method* akan m miliki tahap pros s s bagai b rikut

1. Program akan m nghitung nilai ntropy *class*
2. Program akan m nghitung nilai ntropy dan m ndapatkan nilai *gain info* untuk s tiap atribut pada *attribute_list*
3. Jika *user* m milih untuk m nggunakan algoritma C4.5, maka program akan m nghitung *splitInfo* dan m nghitung *gainRasio*
4. Program akan m milih atribut yang t rbaik untuk dijadikan *node* (jika ID3 maka nilai *gainInfo* yang akan digunakan untuk m milih atribut, jika C4.5 maka nilai *gain Rasio* yang akan digunakan untuk m milih atribut)
5. Program akan m ng mbalikan *node* yang dipilih b rs rta nilai tupl yang t rdapat pada cabang masing-masing

3.2.1 Diagram *Use Case* Perangkat Lunak *Data Mining Log* Histori KIRI

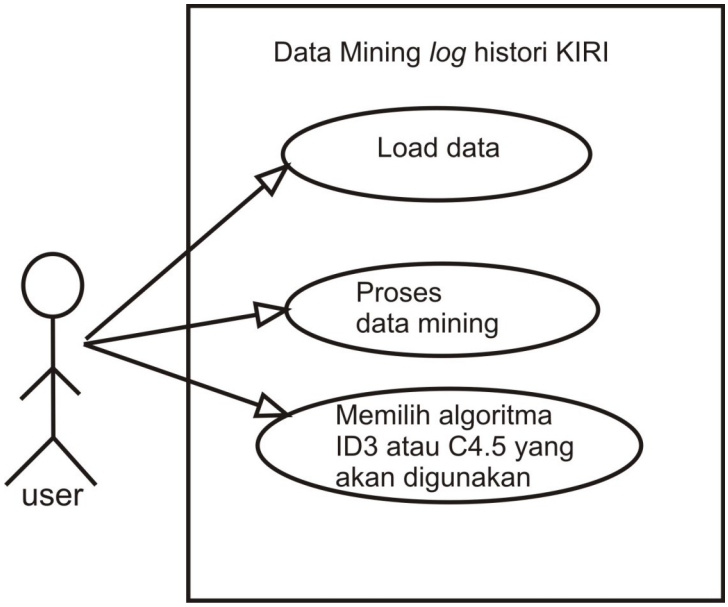
Diagram *use case* m rupakan diagram yang m nd skripsikan sist m d ngan lingkungannya. Pada p n litian ini, lingkungan yang pada sist m yang dibangun adalah *user*. B rdasarkan analisa yang t lah dilakukan, maka *user* dapat m lakukan:

M lakukan *load* data yang digunakan s bagai input data d ngan cara m masukan alamat data pada program

M milih algoritma yang akan digunakan, t rdapat dua algoritma, yaitu ID3 dan C4.5

M lakukan pros s *data mining* d ngan input data dari alamat data yang sudah dimasukan. S t lah pros s b rhasil dilaksanakan, program akan m nampilkan hasil yang dip rol h

Diagram *use case* saat *user* m njalankan p rangkat lunak *data mining log* histori KIRI dapat dilihat pada gambar 3.2.



Gambar 3.2: Diagram *Use Case* P rangkat Lunak *Data Mining Log* Histori KIRI

Tab l 3.5: Sk nario M lakukan *load* Data

Nama	Load data
Aktor	*User*
D skripsi	M masukan alamat data yang akan dijadikan s bagai input program
Kondisi awal	*Textbox* b lum t risi
Kondisi akhir	*Textbox* sudah t risi d ngan alamat data
Sk nario utama	*User* m masukan alamat data pada t xtbox
Eks spi	Data tidak dit mukan

Tab l 3.6: Sk nario M lakukan *Data Mining*

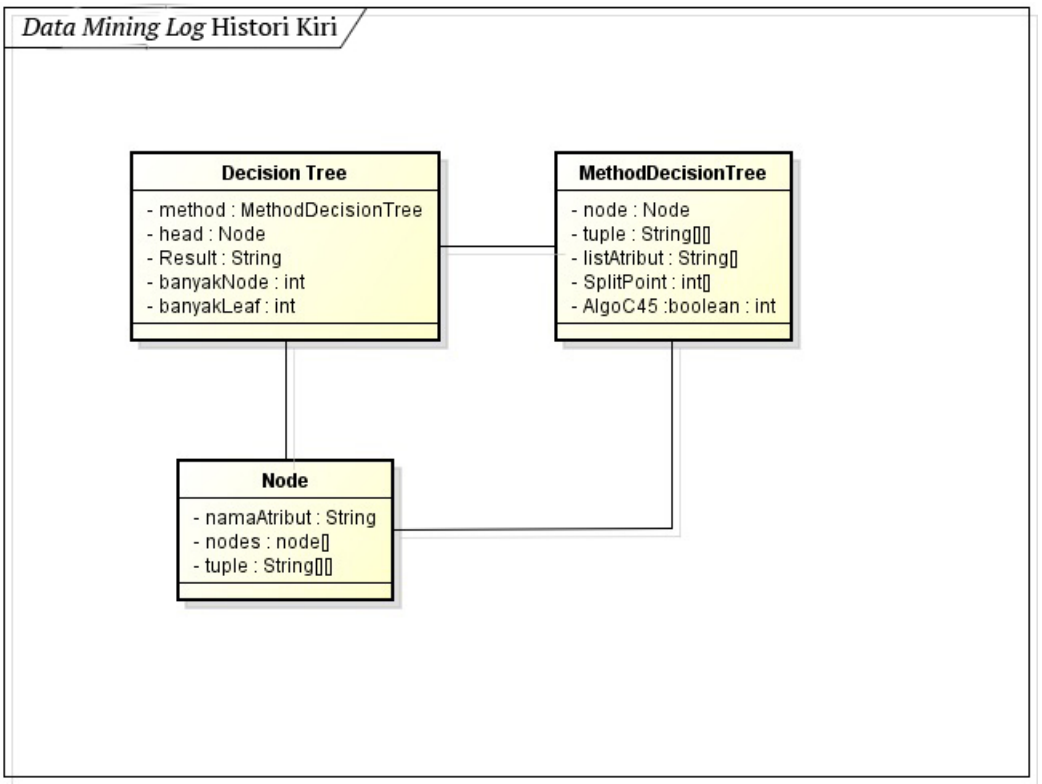
Nama	Pros s *Data Mining*
Aktor	*User*
D skripsi	M n kan tombol pros s pada *interface*
Kondisi awal	*Textbox* b lum t risi
Kondisi akhir	*Textbox* sudah t risi d ngan hasil *data mining*
Sk nario utama	*User* m n kan tombol pros s
Eks spi	Data tidak dit mukan atau data tidak dapat dipros s

Tab l 3.7: Sk nario M milih Algoritma yang Akan Digunakan

Nama	M milih algoritma ID3 atau C4.5
Aktor	*User*
D skripsi	Us r m milih algoritma yang akan dipakai
Kondisi awal	*Radiobutton* t rpilih pada ID3
Kondisi akhir	*Radiobutton* t rpilih pada ID3 atau C4.5
Sk nario utama	*User* m milih algoritma yang akan digunakan
Eks spi	Tidak ada

3.2.2 Diagram kelas Perangkat Lunak *Data Mining Log* Histori KIRI

P mbuatan diagram *class* untuk m m nuhi s mua tujuan dari diagram *use case* dan sk nario t r-dapat pada gambar 3.3.



Gambar 3.3: Diagram *Class* P rangkat Lunak *Data Mining Log* Histori KIRI

B rikut d skripsi k las diagram *class*:

DecisionTree, m rupakan k las utama yang akan m njalankan algoritma p mbuatan pohon

MethodDecisionTree, m rupakan k las yang m njalankan algoritma p milihan atribut untuk p mbuatan pohon (pada p n litian ini, algoritma yang dapat dipilih adalah ID3 dan C4.5)

Node, m rupakan k las yang digunakan s bagai struktur data untuk *decision tree*

DAFTAR REFERENSI

[1] Data Mining *Data Mining Concepts and Techniques* 2006 : Jiaw i Han and Mich lin Kamb r

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/-6.921568846707516,107.61015225201845/1-6.917321,107.6043132/-2513.13707516,107.61015225201845/1

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.trav l/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	t rminal+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.c nd kial ad rshipschool.sc...
114009	E5D9904F0A8B4F99	2/1/2014 2:00	FINDROUTE	-6.88166,107.61561/-6.90774,107.60908/1