**SKRIPSI**

**SISTEM KONTROL LAMPU PENGATUR LALU LINTAS MENGGUNAKAN ALGORITMA C4.5 DENGAN DISKRITISASI**

*TRAFFIC LIGHT CONTROL SYSTEM USING C4.5 ALGORITHM WITH DISCRETIZATION*



Muhammad Ilham Ibadurrohman

14/366093/PA/16211

**PROGRAM STUDI ILMU KOMPUTER**

**DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS GADJAH MADA**

**2021**

**SKRIPSI**

**SISTEM KONTROL LAMPU PENGATUR LALU LINTAS MENGGUNAKAN ALGORITMA C4.5 DENGAN DISKRITISASI**

*TRAFFIC LIGHT CONTROL SYSTEM USING C4.5 ALGORITHM WITH DISCRETIZATION*

Diajukan untuk memenuhi salah satu syarat memperoleh derajat

Sarjana Komputer



Muhammad Ilham Ibadurrohman

14/366093/PA/16211

**PROGRAM STUDI ILMU KOMPUTER**

**DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS GADJAH MADA**

**2021**

# **HALAMAN PERSETUJUAN**

**SKRIPSI**

SISTEM KONTROL LAMPU PENGATUR LAMPU PENGATUR LALU LINTAS DENGAN ALGORITMA C4.5 DENGAN DISKRITISASI

Telah dipersiapkan dan disusun oleh:

Muhammad Ilham Ibadurohman

14/366093/PA/16211

Telah dipertahankan di depan Tim Penguji

pada tanggal \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Susunan Tim Penguji

Dr. Suprapto, M.I.Kom.

Pembimbing I Penguji I

Sri Mulyana, Drs., M.Kom

Pembimbing II Penguji II

# **PERNYATAAN BEBAS PLAGIASI**

Saya yang bertanda tangan dibawah ini:

Nama : Muhammad Ilham Ibadurrohman

NIM : 14/366093/PA/16211

Tahun Terdaftar : 2014

Program Studi : Ilmu Komputer

Fakultas : Matematika dan Ilmu Pengetahuan Alam

Dengan ini saya menyatakan bahwa Skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Dengan demikian saya menyatakan bahwa Skripsi ini bebas dari unsur plagiasi dan apabila di kemudian hari terbukti merupakan plagiasi dari hasil karya penulis lain. Maka penulis bersedia menerima sanksi akadmikdan/atau sanksi hokum yang berlaku.

Yogyakarta, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Penulis

# **KATA PENGANTAR**

Alhamdulillah. Segala puji bagi Allah SWT yang telah melimpahkan rahmat dan nikmatnya sehingga skripsi ini bisa selesai. Shalawat dan salam semoga senantiasa dilimpahkan kepada Rasulullah Muhammad SAW.

Selesainya penulisan laporan penelitian skripsi ini tentunya tidak lepas dari bantuan berbagai pihak. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Bapak dan ibu tercinta atas segala dukungannya selama menempuh studi S1.
2. Bapak Dr. Suprapto, M.I.Kom. dan bapak Drs. Sri Mulyana, M.Kom selaku dosen pembimbing yang telah membimbing, mengarahkan, memotivasi sehingga laporan ini dapat diselesaikan.
3. Mas Hanif Rahmawan yang telah bersedia membantu dan membagikan ilmunya kepada penulis untuk menyelesaikan skripsi ini.
4. Teman-teman seperjuangan di S1 Ilmu Komputer UGM Angkatan Tahun 2014 terutama Syafina, Ruli, Emma, Achmad, dan Daniel yang selalu memberikan semangat untuk mengerjakan skripsi ini.
5. Mbak Asti dan suami yang setia menjadi teman curhat.
6. Semua pihak yang sudah membantu penulisan laporan ini.

Saya menyadari bahwa Skripsi ini masih memiliki banyak kekurangan sehingga saran dan kritik yang membangun senantiasa saya harapkan. Namun saya tetap berharap, laporan ini tetap bisa memberi manfaat bagi para pembaca.

Yogyakarta, \_\_\_\_\_\_\_\_\_\_\_\_ 2021

Penulis,

Muhammad Ilham Ibadurrohman

**Plus Ultra  
 - National Motto of Spain**

**DAFTAR ISI**

[HALAMAN PERSETUJUAN i](#_Toc61440366)

[PERNYATAAN BEBAS PLAGIASI ii](#_Toc61440367)

[KATA PENGANTAR iii](#_Toc61440368)

[DAFTAR ISI v](#_Toc61440369)

[DAFTAR GAMBAR vi](#_Toc61440370)

[DAFTAR TABEL vii](#_Toc61440371)

[INTISARI viii](#_Toc61440372)

[ABSTRACT ix](#_Toc61440373)

[BAB I 10](#_Toc61440374)

[1.1. Latar Belakang 10](#_Toc61440375)

[1.2. Rumusan 11](#_Toc61440377)

[1.3. Batasan 12](#_Toc61440378)

[1.4. Tujuan 12](#_Toc61440379)

[1.5. Manfaat 12](#_Toc61440380)

[BAB II 13](#_Toc61440381)

[BAB III 15](#_Toc61440384)

[3.1. Data Mining 15](#_Toc61440385)

[3.1.1. Diskritisasi Berbasis Entropi 17](#_Toc61440387)

[3.2. Decision Tree 21](#_Toc61440390)

[3.2.1. Algoritma C4.5 23](#_Toc61440393)

[3.3. Lampu Pengatur Lalu Lintas 25](#_Toc61440394)

[3.3.1. Metode Perhitungan Durasi Sinyal 25](#_Toc61440395)

[3.4. *Cross* *Validation* 27](#_Toc61440397)

[BAB IV 28](#_Toc61440399)

[4.1. Analisis Permasalahan 28](#_Toc61440400)

[4.2. Rancangan Umum Sistem 28](#_Toc61440401)

[4.2.1. Classifier 29](#_Toc61440403)

[4.2.2. Output Module 36](#_Toc61440405)

[4.3. Pengujian 37](#_Toc61440406)

[BAB V 38](#_Toc61440408)

[5.1. Classifier 38](#_Toc61440409)

[BAB VI 43](#_Toc61440415)

[DAFTAR PUSTAKA 47](#_Toc61440416)

# **DAFTAR GAMBAR**

[Gambar 3.1 Proses KDD pada database PAGEREF \_Toc61440386 \h17](#_Toc61440386)

[Gambar 3.2 Langkah-langkah diskritisasi PAGEREF \_Toc61440388 \h18](#_Toc61440388)

[Gambar 3.3 Ilustrasi diskritisasi berbasis entropi PAGEREF \_Toc61440389 \h19](#_Toc61440389)

[Gambar 3.4 Struktur decision tree PAGEREF \_Toc61440391 \h21](#_Toc61440391)

[Gambar 3.5 Ilustrasi decision tree resiko kredit PAGEREF \_Toc61440392 \h22](#_Toc61440392)

[Gambar 4.1 Diagram flowchart rancangan sistem PAGEREF \_Toc61440402 \h29](#_Toc61440402)

[Gambar 4.2 Ilustrasi decision tree hasil dari algoritma C4.5 PAGEREF \_Toc61440404 \hError: Reference source not found](#_Toc61440404)

[Gambar 5.1 Algoritma pengubahan dataset PAGEREF \_Toc61440410 \h39](#_Toc61440410)

[Gambar 5.2 Algoritma pembagian data latih dan data uji PAGEREF \_Toc61440411 \h40](#_Toc61440411)

[Gambar 5.3 Algoritma normalisasi data PAGEREF \_Toc61440412 \h41](#_Toc61440412)

[Gambar 5.4 Algoritma pengurutan data PAGEREF \_Toc61440413 \hError: Reference source not found](#_Toc61440413)

[Gambar 5.5 Algoritma Dougherty PAGEREF \_Toc61440414 \hError: Reference source not found](#_Toc61440414)

# **DAFTAR TABEL**

[Tabel 1.1 Pertambahan Jumlah Kendaraan Bermotor di Kabupaten Sleman Tahun 2014-2016 PAGEREF \_Toc61440376 \h10](#_Toc61440376)

[Tabel 2.1 Tabel Perbandingan Tinjauan Pustaka terkait Sistem Kontrol Lalu Lintas PAGEREF \_Toc61440382 \h14](#_Toc61440382)

[Tabel 2.2 Tabel Perbandingan Tinjauan Pustaka terkait Algoritma PAGEREF \_Toc61440383 \h14](#_Toc61440383)

[Tabel 3.1 Nilai setiap kendaraan bermotor PAGEREF \_Toc61440396 \h25](#_Toc61440396)

[Tabel 3.2: Ilustrasi 4-fold-cross validation PAGEREF \_Toc61440398 \h27](#_Toc61440398)

[Tabel 4.1 Confusion Matrix PAGEREF \_Toc61440407 \h38](#_Toc61440407)

# **INTISARI**

Peningkatan populasi manusia secara tidak langsung menyebabkan bertambahnya jumlah kendaraan. Pertambahan jumlah kendaraan dapat meningkatkan resiko terjadinya kemacetan yang dapat menganggu kelancaran lalu lintas. Algoritma lampu pengatur lalu lintas yang banyak digunakan saat ini hanya mengubah rencana waktu sinyal pada kondisi dan waktu tertentu, bukan berdasarkan pada besar arus kendaraan pada simpang. Algoritma ini efektif untuk simpang dengan resiko kemacetan rendah, namun tidak untuk simpang yang memiliki resiko kemacetan tinggi.

Algoritma yang digunakan pada penelitian ini merupakan kombinasi dari algoritma diskritisasi dan algoritma C4.5 untuk proses *data mining* dan konstruksi *decision tree*. Algoritma diskritisasi yang digunakan merupakan algoritma *entropy based discretization* (EBD). Di mana algoritma EBD melakukan diskritisasi pada dataset *traffic flow* berdasarkan nilai *information entropy*-nya. Dan algoritma C4.5 melakukan konstruksi *decision tree* berdasarkan dari data hasil diskritisasi.

<placeholder kesimpulan>

Kata kunci : *decision tree*, algoritma C4.5, *entropy-based discretization*

# **ABSTRACT**

The human population has increased exponentially over the years, and this phenomenon indirectly impacts the vehicle population over the years. The increasing number of vehicles also means an increased risk of congestion and traffic jams that could affect the overall traffic flow. The current algorithm used by most traffic light controller only change its signal timing plan at certain time and conditions. This kind of algorithm is effective on an intersection with a low risk of congestion and traffic jams, but not so for an intersection with a higher risk of congestion and traffic jams.

The algorithm that used in this research is a combination of entropy-based discretization and the C4.5 algorithm for data mining and constructing a decision tree. An entropy-based discretization algorithm discretizes the traffic flow based on its information entropy value. The C4.5 algorithm is used to construct a decision tree based on discretized data.

< conclusion placeholder>

Keyword: entropy-based discretization, C4.5 algorithm, decision tree

# **BAB I**

**PENDAHULUAN**

## **Latar Belakang**

Pertambahan populasi manusia secara tidak langsung menyebabkan bertambahnya jumlah kendaraan baik bermotor maupun tidak bermotor. Berdasarkan survey yang dilakukan oleh Badan Pusat Statistik Kabupaten Sleman dari tahun 2014 sampai 2016, menunjukkan bahwa jumlah kendaraan bermotor pada tahun 2014 mencapai 567.654 unit sedangkan pada tahun 2016 mencapai 929.739 unit. Rincian data tersebut terdapat pada Tabel 1.1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabel 1.1 Pertambahan Jumlah Kendaraan Bermotor di Kabupaten Sleman Tahun 2014-2016** | | | |
| Jenis Kendaraan Bermotor | Tahun | | |
| 2014 | 2015 | 2016 |
| Mobil Penumpang | 76.290 | 84.886 | 167.864 |
| Bus | 996 | 7.092 | 7.384 |
| Mobil Barang | 14.360 | 17.829 | 18.757 |
| Sepeda Motor | 476.008 | 671.528 | 735.734 |
| **Jumlah** | **567.654** | **781.315** | **929.739** |

Berdasarkan Tabel 1.1 di atas, terdapat pertambahan jumlah kendaraan bermotor di Kabupaten Sleman dari tahun 2014 hingga 2016 sebanyak 362.085 unit. Peningkatan jumlah kendaraan tersebut dapat menyebabkan peningkatan besar arus lalu lintas kendaraan yang berpotensi meningkatkan kemungkinan terjadinya kemacetan lalu lintas jika tidak diantisipasi.

Kemacetan lalu lintas merupakan masalah utama di kota-kota seluruh dunia, terutama di kota-kota besar. Menurut Koukol, et al. (2014), kemacetan lalu lintas tersebut disebabkan oleh algoritma yang diterapkan masih berupa aturan matematika sederhana. Berdasarkan Manual Kapasitas Jalan Indonesia (1997), algoritma lampu pengatur lalu lintas yang digunakan saat ini hanya mengubah rencana waktu sinyal pada kondisi dan waktu tertentu, misalnya pada kondisi lalu-lintas puncak pagi, puncak sore dan lewat puncak, dan bukan berdasarkan pada besar arus kendaraan pada simpang. Algoritma jenis ini efektif untuk simpang dengan resiko macet rendah dan tidak mengalami peningkatan resiko kemacetan yang signifikan. Namun efektifitas dan efisiensi tersebut berkurang untuk simpang yang memiliki resiko kemacetan tinggi dan berpotensi mengalami peningkatan resiko kemacetan yang disebabkan oleh jumlah kendaraan yang meningkat.

Dikutip dari artikel di *website* Dinas Perhubungan Kabupaten Sleman, sistem *Area Traffic Control System* (ATCS) merupakan integrasi dari 3 sistem yakni sistem kontrol, sistem informasi, dan sistem pengawasan lalu lintas. Sistem ini sudah mulai diterapkan di beberapa simpang di D.I Yogyakarta. Sistem kontrol tersebut bekerja berdasarkan pada persentase kepadatan di tiap lajur. Perangkat kamera berbasis mikrokontroler yang digunakan untuk menangkap gambar di sebuah persimpangan, harus selalu terkoneksi dengan komputer yang terdapat pada ruang kendali. Karena sistem kontrol bersifat *remote*,sistem jenis ini sangat sensitif terhadap ketersediaan sumber energi listrik dan sinyal koneksi serta rentan terhadap *human error.*

## **Rumusan**

Setiap tahunnya, jumlah kendaraan di perkotaan mengalami pertambahan. Jika pertambahan tersebut tidak disertai dengan perkembangan teknologi sistem lampu pengatur lalu lintas yang bertujuan untuk menangani dan mengurai kemacetan, dapat meningkatkan resiko terjadinya kemacetan.

Teknologi sistem kontrol lampu pengatur lalu lintas konvensional yang digunakan pada simpang masih menggunakan algoritma matematika sederhana yang tidak dapat beradaptasi terhadap besar arus lalu lintas. Pada umumnya, lampu pengatur lalu lintas konvensional hanya melakukan perubahan rencana waktu sinyal berdasarkan waktu dan kondisi yang telah ditetapkan sebelumnya,. Algoritma jenis ini efektif untuk simpang dengan resiko macet rendah dan kondisi macet pada waktu yang telah diketahui dan ditetapkan sebelumnya, namun tidak efektif jika terjadi lonjakan besar arus kendaraan yang terjadi di luar waktu dan kondisi yang telah ditetapkan tersebut.

Menurut Kulkarni dan Waingankar (2007), pengembangan suatu algoritma sistem pengawasan dan kontrol lalu lintas yang adaptif dan cerdas diharapkan dapat menangani dan mengantisipasi situasi lalu lintas yang berubah-ubah. Contohnya dengan sistem yang berdasarkan pada algoritma Machine Learning.

## **Batasan**

Dalam penelitian ini terdapat beberapa batasan masalah, yakni:

1. Besar arus yang digunakan merupakan besar arus pada pendekat.
2. Jenis pendekat adalah jenis pendekat terlawan yang membolehkan gerakan ke kanan.
3. Jenis lampu pengatur lalu lintas berupa lampu pengatur lalu lintas 3 warna.
4. Dataset yang digunakan merupakan open dataset.
5. Data yang terdapat pada dataset merupakan data besar arus lalu lintas selama 24 jam dalam 1 minggu.
6. Besar tambahan durasi lampu hijau berkisar antara 25-40% sesuai yang disarankan dalam MKJI.

## **Tujuan**

Tujuan dari penelitian ini adalah untuk mengembangkan sebuah subsistem untuk sistem kontrol lampu pengatur lalu lintas dengan memanfaatkan nilai besar arus kendaraan per jam yang kemudian diklasifikasikan menggunakan metode klasifikasi *decision tree*. Hasil klasifikasi tersebut kemudian diinferensikan menjadi sebuah nilai tambahan durasi lampu hijau.

## **Manfaat**

Manfaat yang diharapkan dari penelitian ini adalah dapat memberikan gambaran tentang sistem kontrol lalu lintas berdasarkan metode klasifikasi dan sebagai alternatif solusi untuk sistem kontrol lalu lintas yang adaptif.

# **BAB II**

**TINJAUAN PUSTAKA**

Rithesh, et al. (2018) dalam jurnalnya melakukan penelitian penerapan *decision tree* dalam sistem kontrol lalu lintas otomatis. Dalam penelitiannya, input untuk algoritma *decision tree* didapatkan dengan cara melakukan ekstraksi gambar *real-time* menggunakan metode *image processing* dengan algoritma *Canny* *edge detection*. Kriteria untuk sistem kontrol lalu lintas adalah kemiripan antara gambar *real-time­* dengan gambar referensi.

Marve, et al. (2016) dalam jurnalnya menjelaskan bahwa penyempurnaan *timing* sinyal lalu lintas merupakan praktek manajemen kemacetan yang cukup populer. Peningkatan penyempurnaan *timing* sinyal dapat mencakup perubahan sederhana dalam rencana pengaturan *timing* atau sinkronisasi sinyal yang dikendalikan oleh sebuah komputer

Rahmawan (2017) dalam tesisnya melakukan penerapan algoritma C4.5 dengan PCA dan diskritisasi untuk menentukan rekomendasi pelatihan pengembangan diri bagi PNS. Terdapat tiga metode yang diterapkan pada penelitian ini, yakni algoritma C4.5, kombinasi PCA dan C4.5, dan kombinasi PCA, diskritisasi, dan C4.5 untuk melakukan *data mining*.

Elisa (2017) dalam papernya melakukan penelitian penerapan algoritma C4.5 dalam *data mining* untuk mengidentifikasi faktor-faktor penyebab kecelakaan kerja konstruksi. Penelitian tersebut dilakukan dalam 3 tahap: mencaridanmengidentifikasi *pattern* menggunakan algoritma *KDD*, pencarian lebih lanjut menggunakan algoritma *data mining*, dan klasifikasi menggunakan algoritma C4.5

Ringkasan tinjauan pustaka yang telah diuraikan di atas dapat dilihat pada Tabel 2.1 dan Tabel 2.2:

|  |  |  |
| --- | --- | --- |
| **Tabel 2.1 Tabel Perbandingan Tinjauan Pustaka terkait Sistem Kontrol Lalu Lintas** | | |
| Peneliti | Metode | Keterangan |
| Rithesh, et al.  (2018) | *Decision Tree*, *Image Processing* menggunakan *Canny* *edge detection* | **Tujuan:** Mengatur sistem kontrol lampu lalu lintas secara otomatis.  Input didapatkan dengan cara melakukan ekstraksi gambar *real-time* menggunakan metode *image processing* dengan algoritma *Canny* *edge detection.* |
| Marve, et al.  (2016) | Sinkronisasi sinyal yang dikendalikan oleh sebuah komputer | Meningkatan dan menyempurnakan *timing* sinyal. |
| **Tabel 2.2 Tabel Perbandingan Tinjauan Pustaka terkait Algoritma** | | |
| Peneliti | Metode | Keterangan |
| Rahmawan  (2017) | C4.5, PCA, diskritisasi | **Tujuan:** Menentukan rekomendasi pelatihan pengembangan diri bagi PNS.  Terdapat tiga metode *data mining* yang diterapkan pada penelitian ini, yakni algoritma C4.5, kombinasi PCA-C4.5, dan kombinasi PCA-diskritisasi-C4.5 |
| Elisa  (2017) | C4.5 | **Tujuan:** Mengidentifikasi faktor-faktor penyebab kecelakaan kerja konstruksi.  Penelitian dilakukan dalam 3 tahap:   * Pencariandanidentifikasi *pattern* menggunakan algoritma *KDD*. * Pencarian lebih lanjut menggunakan algoritma *data mining*. * klasifikasi menggunakan algoritma C4.5 |

# **BAB III**

**LANDASAN TEORI**

## **Data Mining**

Rahmawan (2017) menjelaskan bahwa *data mining* merupakan proses ekstraksi informasi berharga dari sebuah dataset secara otomatis dan merupakan bagian utama dari proses *knowledge discovery in database* (KDD). Sumber dataset yang digunakan dalam *data mining dapat* diperoleh dari *database, data warehouse, web,* media penyimpanan, atau aliran data dinamis dalam sebuah sistem.

Dalam bukunya, Maimon dan Rokach (2010) menjelaskan bahwa proses KDD dapat dibagi ke dalam 9 tahap:

1. **Memahami domain aplikasi**

Merupakan tahap persiapan dalam proses KDD. Pada tahap ini dilakukan pengambilan keputusan yang berkaitan dengan transformasi data, algoritma yang digunakan, representasi, dll. Selain itu juga dilakukan pemahaman dan pendefinisian atas *goals* dari *end-user* dan *environment* di mana proses KDD akan dilakukan.

1. **Memilih dan membangun dataset**

Berdasarkan pada *goals* yang telah didefinisikan sebelumnya, dilakukan pencarian dan pengumpulan data termasuk data tambahan yang diperlukan. Kemudian data-data tersebut diintegrasikan ke dalam 1 dataset, termasuk atribut-atribut yang akan digunakan untuk proses KDD.

1. ***Preprocessing* dan *data cleansing***

Kehandalan dataset ditingkatkan dengan melakukan *data cleansing* yang meliputi penanganan *missing values* serta menghilangkan *noise* dan *outlier*.

1. **Transformasi data**

Transformasi data dilakukan untuk mendapatkan data yang lebih baik. Metode yang digunakan antara lain adalah metode reduksi dimensi (contoh: seleksi dan ekstraksi fitur) dan metode transformasi atribut (contoh: diskritisasi atribut numerik).

1. **Menentukan jenis proses *data mining* yang sesuai**

Pada tahap ini dilakukan pemilihan jenis proses *data mining* yang akan dilakukan. Misalkan: klasifikasi, regresi, atau *clustering*. Tahap ini sangat bergantung pada *goals* KDD dan langkah-langkah sebelumnya.

1. **Memilih algoritma *data* *mining* yang akan digunakan**

Pada tahap ini, ditentukan metode yang digunakan untuk mencari pola-pola pada data (termasuk *multiple inducers*) secara spesifik dan disesuaikan dengan kebutuhan KDD.

1. **Implementasi algoritma *data* *mining***

Eksekusi algoritma *data* *mining* dilakukan berulang kali hingga didapatkan hasil yang memuaskan dengan cara melakukan pengaturan parameter pada tiap perulangan.

1. **Evaluasi**

Pada tahap ini dilakukan evaluasi dan menginterpretasi pola yang didapatkan dari proses *data mining*. Kemudian dilakukan dokumentasi pada *knowledge* yang didapatkan agar dapat digunakan kembali.

1. **Menggunakan *knowledge* yang diperoleh dari proses *data mining***

Proses KDD telah selesai, *knowledge* yang diperoleh dari proses KDD dapat diimplementasikan pada sistem lainnya.

Ilustrasi tahapan proses KDD tersebut dapat dilihat pada Gambar 3.1:

|  |
| --- |
|  |
| Gambar 3.1 Proses KDD pada database |

### **Diskritisasi Berbasis Entropi**

Dalam jurnal yang ditulis oleh Dash, et al. (2011), dijelaskan bahwa diskritisasi merupakan bagian tahap *pre-processing* padaKDDyang digunakan untuk mengubah fitur kontinu menjadi diskrit. Diskritisasi tersebut akan membagi variabel kontinu ke dalam kategori-kategori. Tujuan utama dari diskritisasi adalah mengurangi jumlah nilai atribut dari sebuah atribut kontinu dengan membaginya ke dalam beberapa interval nilai berdasarkan titik potong (*cut point*) yang telah ditentukan. Penentuan titik potong tersebut dapat ditentukan sesuai dengan *rule* yang ditetapkan oleh user atau menggunakan sebuah algoritma komputasi. Salah satu algoritma yang digunakan untuk menentukan titik potong adalah dengan menggunakan algoritma *enthropy based discretization* (EBD).

Diskritisasi yang dilakukan pada data yang bersifat kontinu memiliki beberapa keuntungan, yakni:

* **Penggunaan memori yang lebih sedikit:**

Penggunaan memori yang lebih sedikit dapat meringankan beban komputasi sehingga proses KDD dapat dilakukan lebih cepat dan menggunakan *resource* yang lebih sedikit.

* **Gambaran pemetaan yang lebih jelas.**

Proses diskritisasi melakukan pemetaan pada data kontinu ke dalam interval-interval berdasarkan titik potong yang ditentukan oleh algoritma diskritisasi untuk menghasilkan data diskrit.

* **Data lebih mudah dipahami, digunakan, dan dijelaskan.**

Pemetaan data kontinu ke dalam interval-interval dapat mengurangi jumlah nilai atribut dan menjadikan dataset lebih sederhana. Sehingga dataset tersebut dapat lebih mudah dipahami, digunakan, dan dijelaskan.

* **Proses klasifikasi dapat dilakukan lebih efisien dan akurat.**

Karena data hasil diskritisasi lebih sederhana dan kecil daripada data kontinu, maka proses klasifikasi dengan data hasil diskritisasi dapat dilakukan secara lebih efisien dan akurat.

Langkah-langkah proses diskritisasi dapat dilihat pada Gambar 3.2:

|  |
| --- |
|  |
| **Gambar 3.2 Langkah-langkah diskritisasi** |

Diskritisasi dibagi menjadi dua jenis berdasarkan waktu proses dan penggunaan label data, perbedaan dari kedua jenis tersebut adalah:

* **Berdasarkan waktu proses:**

Berdasarkan jenis waktu prosesnya Diskritisasi dibagi menjadi diskritisasi global dan diskritisasi lokal. Diskritisasi global dilakukan sebelum proses induksi, sedangkan diskritisasi lokal dilakukan saat proses induksi.

* **Berdasar penggunaan label data:**

Berdasarkan jenis penggunaan label datanya, diskritisasi dibedakan menjadi dua, yakni diskritisasi *unsupervised* dan *supervised*. Pada diskritisasi *unsupervised*,label data tidak dilibatkan dalam proses partisi, sedangkan pada diskritisasi *supervised*, label data dilibatkan dalam proses partisi.

Salah satu jenis algoritma diskritisasi yang berjenis *supervised* adalah algoritma *entropy based discretization* (EBD). Algoritma tersebut merupakan algoritma diskritisasi dengan mekanisme *top-down*. Algoritma ini bertujuan untuk mendapatkan partisi yang mengandung baris data dari kelas yang sama sebanyak mungkin. Sehingga nilai entropi digunakan untuk dapat mencapai tujuan tersebut.

Peluang algoritma ini untuk dapat meningkatkan akurasi cukup besar karena algoritma ini menggunakan informasi kelas dalam menentukan titik potong. Ilustrasi diskritisasi berbasis entropi dapat dilihat pada Gambar 3.3:

|  |
| --- |
|  |
| **Gambar 3.3 Ilustrasi diskritisasi berbasis entropi** |

Fayyad dan Irani (2013) menjelaskan bahwa pada proses EBD, data yang akan didiskritisasi diurutkan terlebih dahulu. Kemudian nilai yang menjadi batas dari 2 kelas dijadikan sebagai kandidat titik potong. Kandidat titik potong dengan nilai *information entropy* terendah akan dipilih sebagai titik potong sehingga akan didapatkan dua buah partisi. Kemudian kedua partisi tersebut kemudian dipartisi lagi secara rekursif sampai kriteria pemberhentian tercapai.

Masing-masing kandidat titik potong dihitung *information entrophy*-nya dengan menggunakan persamaan (3.1), dengan merupakan data yang akan dipotong dengan titik potong T pada atribut A, sementara dan adalah data dari dua interval yang menggunakan titik potong T.

|  |  |
| --- | --- |
|  | (3.1) |

Nilai entropi didapatkan menggunakan persamaan (3.2). Di mana adalah perbandingan data sampel pada kelas dan jumlah data dalam . Sedangkan adalah jumlah kelas yang terdapat dalam .

|  |  |
| --- | --- |
|  | (3.2) |

Kriteria pemberhentian yang digunakan pada penelitian ini adalah kriteria MDLP. Fayyad dan Irani (1993) mengusulkan sebuah kriteria pemberhentian dengan menggunakan MDLP (*Minimum Description Length Principle*) yang akan menghentikan proses partisi ketika . Sehingga ketika kriteria tersebut terpenuhi, proses partisi akan ditolak. Nilai diperoleh dari persamaan (3.4) di mana adalah jumlah partisi.

|  |  |
| --- | --- |
|  | (3.4) |

Nilai merupakan jumlah kelas pada himpunan . Sehingga adalah jumlah kelas pada himpunan .

|  |  |
| --- | --- |
|  | (3.5) |

Dari persamaan (3.5) tersebut, nilai digunakan pada persamaan (3.6) untuk mendapatkan nilai dengan adalah jumlah data dalam S.

|  |  |
| --- | --- |
|  | (3.6) |

## **Decision Tree**

Bethaningtyas, et al. (2019) dalam jurnalnya menjelaskan bahwa algoritma *decision tree* adalah metode klasifikasi yang direpresentasikan oleh sebuah struktur pohon yang memiliki struktur hierarki yang terdiri dari *nodes* dan *direceted edges.* Dalam sebuah *decision* tree, terdapat tiga jenis *node,* yakni:

* ***Root node* :**

Merupakan *node* pada bagian atas *tree. Root node* merupakan sebuah *node* yang tidak memiliki *incoming edges* dan *zero* atau disebut *outgoing edges*.

* ***Internal node* :**

Merupakan *node* percabangan dari *root node*. Sifat dari *node* ini adalah memiliki satu *input* dan minimum 2 *output*.

* ***Leaf node :***

Merupakan *node* akhir dalam sebuah *decision tree* sehingga disebut juga sebagai *terminal nodes*. Sifat dari *node* ini hanya memiliki satu *input* dan tidak memiliki *output*.

Dalam sebuah *decision tree*, setiap *internal node* membagi ruang menjadi dua atau lebih sesuai dengan fungsi diskrit dari input atribut nilai. Dalam kasus yang paling umum dan sederhana, setiap tes menganggap sebagai atribut tunggal, sehingga ruang dipartisi kosong disesuaikan dengan nilai atribut. Klasifikasi menggunakan *decision tree* dilakukan oleh routing dari simpul akar sampai tiba di simpul daun. Struktur dari *decision tree* dapat dilihat pada Gambar 3.4:

|  |
| --- |
|  |
| **Gambar 3.4 Struktur decision tree** |

Wu dan Kumar (2009) dalam bukunya menjelaskan bahwa sebuah *decision tree* merupakan rangkaian dari pertanyaan-pertanyaan yang disusun secara sistematis sehingga masing-masing pertanyaan mengenai sebuah atribut dapat dijawab berdasarkan nilai dari atribut tersebut. Pada Gambar 3.5 yang menggambarkan tentang pohon keputusan untuk penentuan resiko kredit, pertanyaan mengenai nilai *savings*, dijawab berdasarkan nilai atribut *savings* sehingga muncul 3 cabang.

|  |
| --- |
|  |
| **Gambar 3.5 Ilustrasi decision tree resiko kredit** |

Larose (2014) menyatakan bahwa dalam sebuah *decision tree* terdapat beberapa persyaratan yang harus dipenuhi, yakni:

1. Algoritma *decision tree* adalah algoritma yang berjenis *supervised learning* sehingga data latih harus memiliki variabel target.
2. Algoritma *decision tree* membutuhkan kuantitas data latih yang cukup banyak dan bervariasi karena *decision tree* belajar dari contoh. Sehingga, jika contoh kurang sistematis untuk sebagian baris yang bisa didefinisikan, maka proses klasifikasi dan prediksi untuk bagian tersebut akan bermasalah atau tidak mungkin dilakukan.
3. Nilai variabel target harus bersifat *discrete*. Pohon keputusan tidak dapat diaplikasikan pada variabel target yang bersifat *continuous*.

Metode klasifikasi *decision tree* memiliki nilai keakuratan yang bergantung pada entitas yang diklasifikasikan. Dalam penelitian yang dilakukan oleh Nai-aruna dan Moungmaia (2015), nilai keakuratan prediksi resiko *diabetes mellitus* menggunakan metode klasifikasi *decision tree* mencapai nilai 85%. Namun pada penelitian yang dilakukan oleh Rangkuti, et al. (2015) nilai keakuratan untuk mengklasifikasi pola batik yang sulit seperti mega mendung berkisar antara 30-40%.

### **Algoritma C4.5**

Wu dan Kumar (2009) dalam bukunya menjelaskan, algoritma C4.5 merupakan algoritma yang digunakan untuk menyelesaikan masalah klasifikasi dalam *machine learning* dan *data mining*. Algortima C4.5 dikembangkan oleh J.Ross Quinlan sebagai *extension* dari algoritma ID3 yang telah dikembangkan sebelumnya. Algoritma C4.5 digunakan untuk men-*generate* sebuah pohon keputusan dengan dengan model pembelajaran terawasi. Algoritma ini menggunakan konsep *information gain* atau *entropy reduction* untuk memilih kriteria *split* yang optimal.

Rahmawan (2017) menjelaskan bahwa algoritma C4.5 memiliki model pembangunan yang bersifat *top-down*. Langkah-langkah pembangunan dari model pembangunan tersebut adalah:

1. Memilih *root node*.
2. Menerapkan metode *split selection* untuk memilih *split criterion* yang terbaik dan membagi data pelatihan berdasar node/atribut yang terpilih. Algoritma C4.5 menggunakan kriteria *gain ratio* sebagai *default*, tetapi juga dapat menggunakan kriteria *information gain* (atau disebut *gain*). *Information gain* merupakan selisih *information entropy* sebelum dilakukan pemisahan dan sesudah dilakukan pemisahan. Akan tetapi, *information gain* bias terhadap *multivalued attribute*. Sehingga untuk mengatasi hal tersebut, algoritma C4.5 menggunakan *gain ratio* yang merupakan normalisasi dari nilai *gain*. Nilai *gain ratio* didapatkan menggunakan persamaan (3.7) dengan k merupakan jumlah partisi dalam S.

|  |  |  |
| --- | --- | --- |
|  |  | (3.7) |

1. Cek apakah kriteria pemberhentian sudah terpenuhi. Jika kriteria tersebut sudah terpenuhi, maka pembangunan *decision tree* akan dihentikan. Jika kriteria tersebut tidak terpenuhi, maka dilakukan pengulangan langkah ke-2 dengan memilih sebuah node untuk pemisahan.

Pemberhentian dengan menggunakan kriteria pemberhentian berdasarkan homogenitas data dilakukan jika setiap *leaf node* memiliki data yang homogen. Data disebut homogen jika seluruh data pada *leaf node* tersebut mempunyai nilai target yang sama.

Dalam kasus tertentu, homogenitas data pada *leaf node* sulit untuk tercapai karena adanya *noise* pada data yang diklasifikasi. Maka pada kasus tersebut pemberhentian dilakukan jika homogenitas data lebih kecil dari nilai ambang *threshold* tertentu misal .

Algoritma C4.5 menangani atribut *continuous* dengan cara membagi nilai atribut menjadi dua bagian berdasarkan suatu nilai ambang batas. Nilai ambang batas dicari yang terbaik yaitu nilai ambang batas yang dapat memaksimalkan *gain ratio*. Semua nilai di atas nilai ambang batas dimasukkan ke dalam bagian pertama, dan nilai lainnya dimasukkan ke dalam bagian kedua (Wu dan Kumar, 2009).

## **Lampu Pengatur Lalu Lintas**

Lampu pengatur lalu lintas merupakan suatu alat yang berfungsi sebagai pengatur lalu lintas pada simpang. Dalam artikel yang ditulis McLachlan (2011) di situs [www.theguardian.com](http://www.theguardian.com/), disebutkan bahwa sistem kontrol lampu pengatur lalu lintas pertama ditemukan oleh John Peake Knight yang merupakan seorang insinyur sinyal jalur kereta api di Inggris. Sistem kontrol lampu pengatur lalu lintas ini dipasang di luar gedung Parlemen Inggris pada tahun 1868 dan terlihat seperti pengatur jalur kereta api biasa, dengan lengan semaphore dan lampu gas berwarna merah dan hijau untuk pengoperasian di malam hari.

Dikutip dari blog pemerintah negara bagian Utah (2016), lampu pengatur lalu lintas modern dikembangkan oleh Lester Farnsworth Wire di tahun 1912. Sistem merah-hijau di pasang di negara bagian Ohio pada tahun 1914. Sementara itu, sistem tiga-warna yang di operasikan secara manual lewat tower yang berada di tengah simpang di pasang di New York pada tahun 1918. Alat ini baru muncul di Inggris pada tahun 1925 di simpang St. James Street dan Piccadily. Sistem otomatis yang bekerja berdasarkan interval waktu di pasang di Wolverhampton pada tahun 1926.

Di Indonesia, menurut UU No. 22 Tahun 2009 Tentang Lalu Lintas dan Angkutan Jalan, dalam pasal 1 ayat (19) dijelaskan bahwa: Alat Pemberi Isyarat Lalu Lintas adalah perangkat elektronik yang menggunakan isyarat lampu yang dapat dilengkapi dengan isyarat bunyi untuk mengatur lalu lintas orang dan/atau kendaraan di simpang atau pada ruas jalan.

### **Metode Perhitungan Durasi Sinyal**

Dalam Manual Kapasitas Jalan Indonesia (MKJI) tahun 1997 dalam Bab 2 tentang Simpang Bersinyal dijelaskan prosedur perhitungan waktu sinyal pada simpang bersinyal. Prosedur tersebut dibagi menjadi 5 langkah, yakni:

* **Langkah A: Data Masukan**

Pada langkah ini dilakukan pendataan mengenai karakteristik fisik simpang seperti ukuran simpang, pengaturan lalu lintas, kondisi lingkungan, kondisi arus, nilai jumlah kendaraan per jam untuk setiap jenis kendaraan bermotor, dan gerakan lalu lintasnya. Nilai jumlah arus/jam yang memiliki satuan smp/jam tiap lengan simpang didapatkan dengan operasi perkalian nilai jumlah tiap jenis kendaraan per jam dengan nilai emp jenis kendaraan. Operasi tersebut digambarkan oleh persamaan (3.8) dengan adalah jenis kendaraan.

|  |  |
| --- | --- |
|  | (3.8) |

Sedangkan nilai untuk setiap kendaraan dapat dilihat pada Tabel 3.3

|  |  |  |
| --- | --- | --- |
| **Tabel 3.1 Nilai setiap kendaraan bermotor** | | |
| Tipe kendaraan |  | |
| Pendekat terlindung | Pendekat terlawan |
| LV | 1,0 | 1,0 |
| HV | 1,3 | 1,3 |
| MC | 0,2 | 0,4 |

* **Langkah B: Penggunaan Sinyal**

Pada tahap ini ditentukan fase sinyal, durasi waktu antar-hijau, waktu hilang total, dan waktu merah semua. Fase sinyal ditentukan berdasarkan titik konflik pada simpang. Durasi waktu antar-hijau ditentukan berdasarkan lebar jalan rata-rata. Sedangkan waktu hilang total merupakan ulah waktu antar-hijau pada tiap fase. Dan waktu merah semua merupakan fungsi dari kecepatan, jarak kendaraan yang berangkat dan yang datang dari garis henti sampai ke titik konflik, dan panjang kendaraan yang berangkat.

* **Langkah C: Penentuan Waktu Sinyal**

Pada tahap ini waktu sinyal ditentukan berdasarkan tipe pendekat, lebar pendekat efektif, arus jenuh dasar, dan faktor-faktor penyesuaian. Kemudian dicari rasio arus, waktu siklus, dan waktu hijaunya.

* **Langkah D: Kapasitas**

Pada tahap ini dilakukan pengukuran kapasitas masing-masing pendekat dan ditentukan cara-cara untuk meningkatkan kapasitasnya.

* **Langkah E: Perilaku Lalu Lintas**

Pada langkah ini dilakukan pengukuran faktor-faktor yang mempengaruhi perilaku lalu lintas seperti panjang antrian, nilai angka henti dan tundaan lalu lintas. Waktu sinyal yang dihitung dengan buku manual ini disarankan untuk digunakan pada sinyal dengan kendali waktu-tetap bagi kondisi lalu-lintas yang digunakan sebagai data masukan. Untuk keperluan pemasangan di lapangan, supaya berada pada sisi yang aman terhadap fluktuasi lalu-lintas, maka disarankan satu penambahan waktu hijau sebesar 10% secara proporsional dan penambahan waktu siklus yang sepadan. Jika penentuan waktu digunakan untuk pengaturan aktuasi lalu-lintas waktu hijau maksimum sebaiknya ditentukan 25-40% lebih besar dari pada waktu hijau jika menggunakan kendali waktu-tetap.

## ***Cross*-*Validation***

Han, et al. (2012) menjelaskan, salah satu variasi dari teknik pengujian *cross-validation* adalah *k-fold-cross-validation*. Teknik tersebut dilakukan dengan membagi dataset menjadi set *training* dan set *testing*. Data asli dibagi menjadi k-bagian yang disebut *fold*, Di mana setiap *fold* memiliki ukuran yang mirip. Pelatihan dan pengujian dilakukan sebanyak k-kali. Pada iterasi pertama, *fold* 1 digunakan sebagai data *testing* dan *fold* 2 sampai dengan *fold* k digunakan sebagai data *training*. Pada iterasi kedua, *fold* 2 digunakan sebagai data *testing*, dan *fold* 1, *fold* 3, sampai dengan *fold* k digunakan sebagai data *training*. Proses dilakukan sebanyak k-kali sehingga seluruh fold pernah digunakan sebagai sebagai data training tepat sebanyak 1 kali. Perkiraan akurasi menggunakan metode ini adalah keseluruhan klasifikasi yang benar dari k-iterasi dibagi jumlah baris pada data awal. Gambaran proses tersebut dapat dilihat pada Tabel 3.2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabel 3.2 Ilustrasi 4-fold-cross validation** | | | | |
| Iterasi | Fold 1 | Fold 2 | Fold 3 | Fold 4 |
| Iterasi ke-1 | Lat | Lat | Lat | Uji |
| Iterasi ke-2 | Lat | Lat | Uji | Lat |
| Iterasi ke-3 | Lat | Uji | Lat | Lat |
| Iterasi ke-4 | Uji | Lat | Lat | Lat |

# **BAB IV**

**ANALISIS DAN RANCANGAN SISTEM**

## **Analisis Sistem**

### **Data Pemetaan Arus Lalu Lintas**

Data yang digunakan dalam penelitian ini adalah data jumlah arus per jam merupakan data *time series* yang diambil dari *open dataset* dengan durasi maksimum 48 jam. Data besar arus per jam dibagi menjadi data latih dan data uji dengan rasio 75% dan 25%. Masing-masing data latih dan uji dinormalisasi menggunakan *library* eksternal *scikit-learn.*

Pada penelitian ini dilakukan klasifikasi besar arus per jam pada pendekat menggunakan *decision tree* yang dibangun menggunakan algoritma C4.5. Besar arus per jam didapatkan dengan cara menghitung jumlah kendaraan dengan nilai *emp* setiap kendaraan menggunakan persamaan (3.8). Data latih kemudian didiskritisasi menggunakan algoritma EBD. Hasil diskritisasi dilakukan untuk konstruksi model *decision tree* menggunakan algoritma C4.5. model decision tree diuji menggunakan data uji. Hasil pengujian diinferensikan sebagai besaran tambahan durasi lampu hijau. Berdasarkan MKJI, besaran tambahan durasi yang disarankan berkisar antara 25-40%. Model *decision tree* diuji performanya menggunakan metode *4-fold-cross-validation.*

## **Rancangan Umum**

Sistem yang akan dibangun dibagi menjadi *classifier, database* dan *output module. Classifier* meliputi bagian *pre-processing* dan *classifier*. Bagian ini mengklasifikasikan tingkat kemacetan sebuah pendekat berdasarkan nilai arus kendaraan per jam, sedangkan *output module* berfungsi menampilkan output tambahan durasi lampu hijau dengan mengubah aturan *decision tree* menjadi satuan detik. Database menyimpan data hasil *preprocessing* dan klasifikasi. Diagram *flowchart* dari rancangan sistem dapat dilihat pada Gambar (4.1).

|  |
| --- |
|  |
| **Gambar 4.1 Diagram flowchart rancangan sistem** |

## **Classifier**

Tingkat kemacetan berbanding lurus dengan nilai arus kendaraan per jam dan dibagi menjadi 4 kategori: ramai, padat, merayap, dan macet. Nilai arus kendaraan per jam didapatkan dengan mengoperasikan data hasil pengolahan video ATCS dengan operasi (3.8) dengan nilai untuk pendekat berjenis terlindung. Nilai arus kendaraan per jam tersebut kemudian diklasifikasi menggunakan *decision tree* yang dibangun menggunakan algoritma C4.5.

* **Diskritisasi**

Proses diskritisasi yang digunakan pada penelitian ini adalah diskritisasi berbasis entropi dengan kriteria pemberhentian partisinya menggunakan 2 jenis kriteria yakni jumlah interval dan MDLP. Diskritisasi berbasis entropi merupakan *supervised discretization* sehingga pada prosesnya melibatkan label dari masing-masing data.

Untuk kriteria pemberhentian berupa jumlah interval, nilai jumlah interval tersebut didapatkan dengan cara mengoperasikan jumlah data sampel dengan persamaan (3.3). Sehingga, berdasarkan persamaan tersebut jumlah interval maksimum untuk 126 data adalah 4 interval.

Untuk mendapatkan titik potong, nilai *fitur* diurutkan dari kecil ke besar. Setelah diurutkan, dilakukan perhitungan nilai *information enthropy* untuk setiap kandidat titik potong menggunakan persamaan (3.1). Titik potong yang digunakan adalah kandidat titik potong yang memiliki nilai *information entropy* terkecil, dan nilai untuk titik potong merupakan rata-rata dari dua buah nilai menjadi kandidat titik potong. Jumlah interval maksimum yang dibutuhkan untuk proses diskritisasi EBD adalah 4. Maka untuk mendapatkan 4 interval, diperlukan 3 level diskritisasi dimana diskritisasi level kedua dilakukan pada hasil diskritisasi level pertama dan seterusnya. Dataset pada tabel 4.1 digunakan sebagai contoh proses diskritisasi.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tabel 4.1 Dataset *training*** | | | | | |
| No | timestamp | Qmc | Qhv | Qlv | Qmv |
| 1 | 09/10/2015 12:00 | 0.681204895 | 0.409015025 | 0.781223377 | 0.757802221 |
| 2 | 09/10/2015 16:00 | 0.833071854 | 0.534223706 | 0.919577441 | 0.902939319 |
| 3 | 07/10/2015 15:00 | 0.899592093 | 0.577629382 | 0.894871358 | 0.899465818 |
| 4 | 05/10/2015 02:00 | 0.030749922 | 0.028380634 | 0.048219458 | 0.043681911 |
| 5 | 10/10/2015 05:00 | 0.060872294 | 0.011686144 | 0.01482365 | 0.021235724 |
| 6 | 07/10/2015 16:00 | 0.836209602 | 0.677796327 | 0.754131879 | 0.790418925 |
| 7 | 09/10/2015 18:00 | 0.953247568 | 0.524207012 | 0.867439087 | 0.881808852 |
| 8 | 06/10/2015 18:00 | 0.847819266 | 0.357262104 | 0.754472653 | 0.759788958 |
| 9 | 06/10/2015 04:00 | 0.001568874 | 0.013355593 | 0.004600443 | 0.003565602 |
| 10 | 06/10/2015 06:00 | 0.158770003 | 0.058430718 | 0.035270063 | 0.058233777 |
| … | … | … | … | … | … |
| 117 | 11/10/2015 00:00 | 0.159397553 | 0.116861436 | 0.264781053 | 0.241553076 |
| 118 | 05/10/2015 21:00 | 0.416692815 | 0.247078464 | 0.570625319 | 0.534221883 |
| 119 | 05/10/2015 18:00 | 0.737056793 | 0.365609349 | 0.89793832 | 0.85284985 |
| 120 | 09/10/2015 10:00 | 0.933165987 | 0.549248748 | 0.947009712 | 0.942450397 |
| 121 | 10/10/2015 17:00 | 0.951992469 | 0.664440735 | 0.924177884 | 0.939779485 |
| 122 | 07/10/2015 10:00 | 0.796360213 | 0.539232053 | 0.746634861 | 0.763749276 |
| 123 | 04/10/2015 01:00 | 0.080012551 | 0.053422371 | 0.093201568 | 0.089245303 |
| 124 | 08/10/2015 18:00 | 0.751804205 | 0.512520868 | 0.709831317 | 0.725119731 |
| 125 | 09/10/2015 21:00 | 0.39974898 | 0.282136895 | 0.746634861 | 0.670885743 |
| 126 | 05/10/2015 10:00 | 0.667085033 | 0.437395659 | 0.658033737 | 0.663215094 |

Tabel 4.2, 4.3, dan 4.4 merupakan contoh hasil diskritisasi yang dilakukan pada fitur Qmc **:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabel 4.2 Titik potong pertama untuk Qmc** | | | | |
|  | **Label** | **IE Qmc** | | **Nilai TP** |
| 0.7072482 | 2 | 1.5621554 |  |  |
| 0.70787575 | 2 | 1.5612212 |  |
| 0.72764355 | 1 | 1.5214379 | 0.7285849 |
| 0.7295262 | 3 | 1.5506461 |
| 0.7320364 | 2 | 1.5491841 |  |
| 0.73391905 | 3 |  |  |

Nilai titik potong antara nilai 0.72764355 dan 0.7295262 adalah rata-rata dari kedua nilai tersebut, yakni 0.7285849. Nilai 0.7285849 merupakan titik potong dengan nilai *information enthropy* terendah sehingga dipilih sebagai titik potong untuk diskritisasi level pertama. Nilai *information enthropy* pada titik potong tersebut didapatkan menggunakan persamaan dibawah:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabel 4.3 Titik potong kedua untuk Qmc** | | | | |
|  | **Label** | **IE Qmc** | | **Nilai TP** |
| 0.44116724 | 1 |  | 1.191595 |  |
| 0.44650141 | 1 | 1.1839311 |  |
| 0.44681519 | 1 | 1.1745481 | 0.4549733 |
| 0.46313147 | 2 | 1.2246012 |
| 0.47003452 | 1 | 1.2140383 |  |
| 0.47285849 | 2 |  |  |

Diskritisasi level kedua dilakukan pada hasil diskritisasi level pertama dengan memecah interval diatas titik potong 0.7285849. Nilai titik potong ditentukan diantara nilai 0.44681519 dan 0.46313147 dimana nilai titik potong tersebut adalah 0.4549733. Nilai *information enthropy* pada titik potong tersebut didapatkan menggunakan persamaan di bawah:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabel 4.4 Titik potong ketiga untuk Qmc** | | | | |
|  | **Label** | **IE Qmc** | | **Nilai TP** |
| 0.07718858 | 0 | 0.5412396 |  |  |
| 0.08001255 | 0 | 0.5005573 | 0.0815814 |
| 0.0831503 | 1 | 0.6030365 |
| 0.13837465 | 1 | 0.670531 |  |
| 0.1390022 | 0 | 0.6300591 |  |
| 0.14872921 | 1 |  |  |

Diskritisasi level ketiga dilakukan pada hasil diskritisasi level kedua diatas titik potong 0.4549733. Nilai titik potong yang ditentukan diantara nilai 0.08001255 dan 0.0831503 adalah 0.0815814. Nilai *information enthropy* untuk titik potong tersebut adalah:

Proses dikritisasi dilakukan pada semua *fitur* dan hasilnya dibuatkan tabel seperti Tabel 4.5.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tabel 4.5 Dataset hasil diskritisasi** | | | | | |
| No | timestamp | Qmc | Qhv | Qlv | Qmv |
| 1 | 09/10/2015 12:00 | 2 | 3 | 2 | 2 |
| 2 | 09/10/2015 16:00 | 3 | 3 | 3 | 3 |
| 3 | 07/10/2015 15:00 | 3 | 3 | 3 | 3 |
| 4 | 05/10/2015 02:00 | 0 | 0 | 0 | 0 |
| 5 | 10/10/2015 05:00 | 0 | 0 | 0 | 0 |
| 6 | 07/10/2015 16:00 | 3 | 3 | 2 | 3 |
| 7 | 09/10/2015 18:00 | 3 | 3 | 3 | 3 |
| 8 | 06/10/2015 18:00 | 3 | 2 | 2 | 2 |
| 9 | 06/10/2015 04:00 | 0 | 0 | 0 | 0 |
| 10 | 06/10/2015 06:00 | 1 | 0 | 0 | 0 |
| … | … | … | … | … | … |
| 117 | 11/10/2015 00:00 | 1 | 1 | 1 | 1 |
| 118 | 05/10/2015 21:00 | 1 | 1 | 1 | 1 |
| 119 | 05/10/2015 18:00 | 3 | 2 | 3 | 3 |
| 120 | 09/10/2015 10:00 | 3 | 3 | 3 | 3 |
| 121 | 10/10/2015 17:00 | 3 | 3 | 3 | 3 |
| 122 | 07/10/2015 10:00 | 3 | 3 | 2 | 3 |
| 123 | 04/10/2015 01:00 | 0 | 0 | 0 | 0 |
| 124 | 08/10/2015 18:00 | 3 | 3 | 2 | 2 |
| 125 | 09/10/2015 21:00 | 1 | 2 | 2 | 2 |
| 126 | 05/10/2015 10:00 | 2 | 3 | 2 | 2 |

Sedangkan untuk pemberhentian menggunakan kriteria MDLP, proses *splitting* akan berhenti jika nilai . Penerapan kriteria pada level diskritisasi dilakukan dengan menghitung nilai entropi interval 1 dan interval 2 menggunakan persamaan (3.2). Penerapan kriteria tersebut pada diskritisasi untuk Qmc pada diskritisasi level 1 dapat dilihat pada perhitungan berikut:

)

Hasil perhitungan tersebut digunakan untuk menghitung nilai menggunakan persamaan (3.5). Nilai kemudian digunakan untuk mendapatkan nilai menggunakan persamaan (3.6). Nilai δ tersebut kemudian dibandingkan dengan nilai yang didapatkan menggunakan persamaan (3.4). Proses perhitungan-perhitungan tersebut dapat dilihat pada persamaan dibawah:

Dari hasil perhitungan diatas, perbandingan nilai dan yang dihasilkan memperlihatkan bahwa nilai tidak lebih kecil dari , maka proses splitting bisa diterima. Proses splitting dilakukan terus untuk setiap level diskritisasi yang terbentuk hingga kriteria pemberhentian tercapai. Perhitungan ini dilakukan ke semua fitur kecuali timestamp*.*

* **Algoritma C4.5**

Langkah pertama pada algoritma C4.5 adalah menentukan *root node.* Penentuan *root node* dilakukan dengan cara mencari atribut yang memiliki *gain ratio* tertinggi yang dicari menggunakan persamaan (3.7). Contoh hasil perhitungan *gain ratio* untuk data diskrit pada Tabel 4.5 pada atribut dijelaskan sebagai berikut:

Perhitungan tersebut dilakukan untuk semua *fitur.* Hasil perhitungan terhadap semua fitur menunjukkan *fitur* memiliki *gain ratio* terbesar sehingga ditetapkan sebagai *root node*. Setelah *root node* didapatkan, data pelatihan dibagi sesuai *root node* terpilih yakni . Sebuah *leaf node* dianggap homogen jika nilai entropinya berjumlah 0. Jika *leaf node* sudah homogen, maka tidak perlu dilakukan pemisahan lagi. Namun jika belum maka masih perlu dilakukan pemisahan dengan melakukan perhitungan yang sama seperti perhitungan sebelumnya. Pemisahan dilakukan sampai *decision tree* memiliki nilai entropi 0.

Pada *fitur* , *leaf node* ditetapkan pada nilai 1 karena memiliki nilai entropi 0 yang berarti sudah homogen. Sedangkan nilai 0, 2, dan 3 belum homogen sehingga perlu dilakukan pemisahan. Pemilihan *fitur* terbaik untuk melakukan pemisahan pada data yang bernilai ,, dilakukan seperti perhitungan diatas dan pemisahan dilakukan terus sampai seluruh *leaf node* bernilai homogen. Pohon keputusan yang terbentuk dari data pada Tabel 4.5 dapat dilihat pada Gambar 4.2

|  |
| --- |
|  |
| **Gambar 4.2 Ilustrasi *decision tree* dari algoritma C4.5** |

Langkah yang dilakukan setelah *decision tree* terbentuk adalah mengubahnya ke dalam bentuk aturan seperti berikut:

1. Jika , maka 0
2. Jika , , , 0, maka 0
3. Jika , , , 0, maka 0
4. Jika , , , , maka 1
5. Jika maka 1, 1
6. Jika , , , 1, maka 1
7. Jika , , , 1, maka 1
8. Jika , , , , maka 1
9. Jika , , , , maka 1
10. Jika , , , , maka 2
11. Jika , , , , maka 2
12. Jika , , , , maka 2
13. Jika , , , , maka 2
14. Jika , , 1, 2, maka 2
15. Jika , , 2, 2, maka 2
16. Jika , , , , maka 2
17. Jika , , , , maka 3
18. Jika , , , , maka 3
19. Jika , , , , maka 3
20. Jika , , , , maka 3
21. Kelas default adalah 0

Nilai merupakan nilai kesimpulan dari rule tersebut. Sehingga jika nilai adalah 2 maka nilai kesimpulannya adalah 2. Nilai kesimpulan merupakan nilai tingkat kemacetan pada pendekat pada suatu waktu tertentu.

## **Output Module**

Bagian *output module* merupakan mesin inferensi sederhana dimana mesin inferensi melakukan penentuan besar tambahan durasi lampu hijau. Besar tambahan durasi lampu hijau didapatkan dengan cara memproyeksikan aturan *decision tree* ke dalam persentase tambahan durasi lampu hijau. Sebagai contoh, rule yang menghasilkan nilai akhir 0 akan mendapatkan 0% tambahan durasi, nilai akhir 1 mendapatkan 25% tambahan durasi, nilai akhir 2 mendapatkan 30% tambahan durasi, dan nilai akhir 3 mendapatkan 40% tambahan durasi.

## **Pengujian**

Pengujian dilakukan pada bagian *classifier.* Pengujian diawali dari internal sub sistem untuk mendapatkan konfigurasi terbaik untuk satu jenis pelatihan dengan cara menentukan jumlah *fitur* dan *split point* yang dapat menghasilkan performa terbaik. Untuk mendapatkan konfigurasi terbaik, semua fitur diujicoba menggunakan *4-fold-cross-validation*. Mulai dari penggunaan 1 fitur hingga 3 fitur. Pengujian dilakukan menggunakan *4-fold-cross-validation* sehingga dihasilkan 4 aturan dimana 1 *fold* menghasilkan 1 aturan.

Pengukuran performa dilakukan dengan menghitung *precision*, *recall*, keakuratan, dan ­*F-measure*. Nilai *precision* dicari menggunakan persamaan (4.1):

|  |  |
| --- | --- |
|  | (4.1) |

Nilai recall didapatkan menggunakan persamaan (4.2):

|  |  |
| --- | --- |
|  | (4.2) |

Nilai keakuratan didapatkan menggunakan persamaan (4.3)

|  |  |
| --- | --- |
|  | (4.3) |

Dan nilai ­F-measure didapatkan dengan menggunakan persamaan (4.4)

|  |  |
| --- | --- |
|  | (4.4) |

Semua perhitungan tersebut berdasarkan *confusion matrix* seperti pada Tabel 4.1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabel 4.1 Confusion Matrix** | | | |
|  |  | Nilai sebenarnya | |
|  |  | True | False |
| Nilai prediksi | True | True Positive (TP) | False Positive (FP) |
| False | False Negative (FN) | True Negative (TN) |

# **BAB V**

**IMPLEMENTASI**

## **Classifier**

Pada proses konstruksi *classifier*, terdapat beberapa *library* eksternal yang akan digunakan, yakni:

1. Numpy ([www.numpy.org](http://www.numpy.org/))
2. Pandas (<https://pandas.pydata.org/>)
3. Scikit-learn (<http://scikit-learn.org/>)

* **Preprocessing**

Proses konstruksi *classifier* dimulai dengan tahap *preprocessing* yang mengubah data besar arus per jam menjadi data smp/jam dan melakukan normalisasi. Untuk mendapatkan data smp/jam tersebut, dataset besar arus per jam untuk setiap jenis kendaraan dioperasikan dengan nilai emp untuk jenis pendekat terlawan menggunakan persamaan (3.8).

Jenis pendekat terlawan merupakan jenis pendekat yang mengizinkan gerakan belok kanan untuk pendekat yang saling berlawanan arah. Jenis pendekat ini digunakan karena jenis pendekat ini adalah jenis pendekat yang paling umum digunakan di kota Yogyakarta. Penjelasan operasi persamaan (3.8) dataset dapat dilihat pada potongan kode pada Gambar 5.1.

|  |
| --- |
|  |
| Gambar 5.1 Algoritma pengubahan data besar arus/jam menjadi smp/jam |

Setelah data besar arus kendaraan diubah menjadi data smp/jam, dataset untuk masing-masing jenis kendaraan digabungkan menggunakan algoritma pada Gambar 5.2.

|  |
| --- |
|  |
| Gambar 5.2 Algoritma penggabungan dataset |

Setelah dataset digabungkan dilakukan konstruksi fitur yang merupakan jumlah total arus. Fitur ini ditambahkan karena dalam perhitungan durasi sinyal lampu pengatur lalu lintas yang tercantum dalam MKJI, fitur ini berperan penting. Dalam perhitungan ini, fitur tersebut digunakan sebagai penentu nilai akhir *decision tree*. Algoritma ini dijelaskan oleh Gambar 5.3

|  |
| --- |
|  |
| Gambar 5.3 Algoritma konstruksi fitur baru |

Dataset kemudian dipotong menjadi data *training* dan data *testing* secara random dengan rasio pembagian untuk data latih sebesar 75% dan data uji sebesar 25% dari keseluruhan jumlah data. Pemotongan tersebut dilakukan menggunakan algoritma *train\_test\_split* seperti pada gambar 5.4.

|  |
| --- |
|  |
| Gambar 5.4 Algoritma pembagian data latih dan data uji |

Kemudian masing-masing data latih dan data uji dinormalisasi menggunakan library Scikit-learn yang dijelaskan oleh Gambar 5.5.

|  |
| --- |
|  |
| Gambar 5.5 Algoritma normalisasi data |

* **Diskritisasi**

Metode diskritisasi yang digunakan pada penelitian ini merupakan metode *entropy based discretization* (EBD) atau bisa juga disebut *entropy based binning* (EBB). Metode diskritisasi berbasis entropi merupakan *supervised discretization* sehingga pada prosesnya melibatkan label dari masing-masing data. Proses *labelling* dilakukan menggunakan bantuan Microsoft Excel seperti diilustrasikan oleh Gambar 5.6

|  |
| --- |
|  |
| Gambar 5.6 Hasil *labelling* menggunakan Microsoft Excel |

Proses diskritisasi dilakukan menggunakan *library* *mdlp-discretization* yang dikembangkan oleh Henry Lin (2017). Kutipan kode yang digunakan untuk menggunakan library tersebut ditunjukkan oleh Gambar 5.7.

|  |
| --- |
|  |
| Gambar 5.7 Algoritma diskritisasi data |

*Library* tersebut menggunakan kriteria pemberhentian MLDP sehingga proses diskritisasi hanya akan berhenti jika nilai .

* **Decision Tree**

Metode konstruksi *decision tree* yang digunakan adalah algoritma C4.5. Algoritma C4.5 merupakan algoritma konstruksi *decision tree* yang bersifat *supervised learning* sehingga diperlukan kolom label proses *training*-nya. Kolom label diberi nama header “*Decision”* karena *library* *chefboost* hanya menerima kolom label dengan nama header tersebut. Berbeda dengan *library mdlp-discretization*, *library chefboost* hanya menerima label berupa *string*, sehingga isi dari kolom label “*Decision”* perlu diganti dari *integer* ke *string*, proses tersebut dijelaskan oleh Gambar 5.8 berikut.

|  |
| --- |
|  |
| Gambar 5.8 Penggantian nilai isi kolom label dari *integer* ke *string* |

Potongan kode untuk menjalankan algoritma *learning* dan konstruksi *decision tree* dari file *training* hasil diskritisasi dapat dilihat pada Gambar 5.9 berikut.

|  |
| --- |
|  |
| Gambar 5.9 Algoritma C4.5 menggunakan library chefboost |

# **BAB VI**

**HASIL DAN PEMBAHASAN**

* 1. **Preprocessing**
* **Mengubah dataset besar arus per jam menjadi smp/jam**

Dalam penelitian ini terdapat tiga dataset yang masing-masing mewakili nilai besar arus per jam untuk sepeda motor (MC), kendaraan ringan seperti mobil penumpang dan mobil pickup (LV), dan kendaraan berat seperti truk dan bus (HV). Masing-masing dataset tersebut diubah menjadi dataset smp per jam menggunakan algoritma pada Gambar 5.1. Hasil dari operasi algoritma tersebut ditampilkan pada Gambar 6.1.

|  |
| --- |
|  |
| **Gambar 6.1 Hasil pengubahan dataset besar arus per jam menjadi smp/jam** |

* **Menggabungkan dataset**

Setelah ketiga dataset tersebut diubah, dilakukan penggabungan menjadi 1 dataset. Proses penggabungan dilakukan menggunakan algoritma dari Gambar 5.2 dan hasilnya dijelaskan oleh Gambar 6.2.

|  |
| --- |
|  |
| **Gambar 6.2 Hasil penggabungan dataset** |

* **Konstruksi fitur**

Fitur merupakan hasil penjumlahan fitur , , dan . Fitur ini berfungsi menentukan nilai akhir pada *decision tree*. Hasil penjumlahan tersebut dapat dilihat pada Gambar 6.3

|  |
| --- |
|  |
| **Gambar 6.3 Hasil penggabungan dataset beserta fitur baru** |

* **Membagi dataset**

Hasil penggabungan dataset kemudian dibagi menjadi data training dan data testing menggunakan algoritma dari Gambar 5.3. Hasil pembagian tersebut dapat dilihat pada Gambar 6.4. Dari hasil pembagian tersebut dihasilkan data training sebanyak 126 baris dan data testing sebanyak 42 baris.

|  |
| --- |
|  |
| **Gambar 6.4 Pemotongan dataset menjadi data *training* dan data *testing*** |

* **Normalisasi data**

Normalisasi dilakukan secara terpisah untuk setiap fitur baik pada data training maupun data testing. Normalisasi dilakukan menggunakan algoritma dari Gambar 5.4. Hasil dari normalisasi tersebut ditampilkan pada Gambar 6.4.

|  |
| --- |
|  |
| **Gambar 6.4 Normalisasi data** |

* 1. **Diskritisasi**

Hasil diskritisasi dari data *training* yang telah di diskritisasi adalah *array* dari fitur Qmc, Qhv, Qlv dan Qmv seperti pada Gambar 6.5 berikut.

|  |
| --- |
|  |
| **Gambar 6.5 Hasil Diskritisasi** |

* 1. **Decision Tree**

Bukti bahwa *decision tree* telah dikonstruksi ditunjukkan oleh Gambar 6.6 berikut

|  |
| --- |
|  |
| **Gambar 6.5 Hasil Diskritisasi** |

# **BAB VII**

**KESIMPULAN**

* 1. **Kesimpulan**

Kesimpulan yang dapat diambil dari penelitian ini adalah:

1. Nilai arus lalu lintas bernilai cukup besar harus dinormalisasi sebelum dilakukan proses diskritisasi agar komputasinya lebih ringan.
2. Proses diskritisasi dilakukan karena data hasil normalisasi bersifat kontinu, algoritma C4.5 tidak dapat melakukan proses *training* menggunakan data kontinu.
3. Pendekatan menggunakan algoritma diskritisasi dan C4.5 dapat dijadikan alternatif dalam pengembangan sistem Smart Traffic.

# **DAFTAR PUSTAKA**

Badan Pusat Statistik Kabupaten Sleman, 2020, *Jumlah Kendaraan Bermotor yang Terdaftar menurut Jenisnya di Kabupaten Sleman, 2014*, <https://slemankab.bps.go.id/statictable/2017/01/03/71/jumlah-kendaraan-bermotor-yang-terdaftar-menurut-jenisnya-di-kabupaten-sleman-kecuali-milik-tni-2014.html>, diakses 14 November 2020

Badan Pusat Statistik Kabupaten Sleman, 2020, *Jumlah Kendaraan Bermotor yang Terdaftar menurut Jenisnya di Kabupaten Sleman, 2015-2016*, <https://slemankab.bps.go.id/statictable/2017/11/15/249/jumlah-kendaraan-bermotor-yang-terdaftar-menurut-jenisnya-di-kabupaten-sleman-kecuali-milik-tni-2015-2016.html>, diakses 14 November 2020

Bethaningtyas, H., Suwandi, Anggraini, C.D., 2019, Sistem Klasifikasi Kondisi Pita Suara dengan Metode Decision Tree, *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, Vol. 8, No. 2, pp. 168-174

Dash, R., Paramguru, R.L. & Dash, R., 2011, Comparative Analysis of Supervised and Unsupervised Discretization Techniques. *International Journal of Advances in* *Science and Technology*, Vol.2, No. 3, pp. 29–37.

Dinas Perhubungan Kabupaten Sleman, 2017, *Area Traffic Control System*, <https://perhubungan.slemankab.go.id/area-traffic-control-system/>, diakses 31 Desember 2020

Direktorat Jenderal Bina Marga, 1997, *Manual Kapasitas Jalan Indonesian (MKJI)*, Jakarta: Kementrian PUPR

Fayyad, U.M. & Irani, K.B., 1993, Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning, In: *13th International Conference on Artificial Intelligence*, Chambe’ry: IJCAI, pp. 1022–1027

Elisa, E., 2017, Analisa dan Penerapan Algoritma C4.5 Dalam Data Mining Untuk Mengidentifikasi Faktor-Faktor Penyebab Kecelakaan Kerja Kontruksi PT. Arupadhatu Adisesanti, *Jurnal Online Informatika*, Vol. 2, No. 1, pp. 36-41

KBBI, 2018, *Arti kata Kendaraan Bermotor menurut KBBI*, <https://kbbi.kata.web.id/kendaraan-bermotor/>, diakses 10 September 2019

Larose, D.T. & Larose, C.D., 2014, *Discovering Knowledge in data An Introduction to Data Mining* 2nd ed., New Jersey: John Wiley & Sons, Inc.

Lin, H., 2017, *Minimum Description Length Binning*, <https://github.com/hlin117/mdlp-discretization.git>, diakses 20 Juni 2021

McLahlan, A., 2011, *Who invented traffic lights and where were the first ones situated?*, <https://www.theguardian.com/notesandqueries/query/0,5753,-1460,00.html>, diakses pada tanggal 28 Agustus 2019

Maimon, O. & Rokach, L., 2010, *Data Mining and Knowledge Discovery Handbook* 2nd edition, Springer, New York, Amerika Serikat

Nai-arun, N. & Moungmai, R. (2015) Comparison of Classifiers for the Risk of Diabetes Prediction, In: *7th* *International Conference on Advances in Information Technology*, Bangkok: Elsevier B.V, pp. 132-142

Rahmawan, H., 2017, *Penentuan Rekomendasi Pelatihan Pengembangan Diri Bagi Pegawai Negeri Sipil Menggunakan Algoritma C4.5 Dengan Principal Component Analysis Dan Diskritisasi*, Yogyakarta: Universitas Gadjah Mada

Rangkuti, A.H., Rasjid, Z.E., Santoso, D. (2015) Batik Image Classification Using Treeval And Treefit As Decision Tree Function In Optimizing Content Based Batik Image Retrieval, In: *International Conference on Computer Science and Computational Intelligence*, Jakarta: Elsevier B.V, pp. 577-583

Rithesh, R.N., Vignesh R., Anala, M. R., 2018, Autonomous Traffic Signal Control using Decision Tree, *International Journal of Electrical dan Computer Engineering*, Vol. 8, No. 3, pp. 1522-1529

UDOT Communications, 2016, *Happy Birthday Lester Wire*, <https://blog.udot.utah.gov/2016/09/happy-birthday-lester-wire/>, diakses 22 Januari 2021

UU RI No. 22 Tahun 2009 Tentang Lalu Lintas dan Angkutan Jalan

Wu, X. & Kumar, V., 2009, *The Top Ten Algorithms In Data Mining*, Boca Raton: CRC Press