

PENGEMBANGAN MODEL *MACHINE LEARNING* UNTUK ANALISIS PREDIKSI KEMUNGKINAN KEJADIAN PATAH REL KERETA API DI INDONESIA

**Studi kasus: PT Kereta Api Indonesia di daerah operasi Sumatera Selatan
(DIVRE III dan DIVRE IV)**

LAPORAN TUGAS AKHIR

Disusun sebagai syarat kelulusan tingkat sarjana

oleh :

Nafisah Nurul Hakim

NIM: 18215045



**PROGRAM STUDI SISTEM DAN TEKNOLOGI INFORMASI
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG**

2020

Lembar Pengesahan

**Pengembangan Model *Machine Learning* untuk Analisis Prediksi Kemungkinan
Kejadian Patah Rel Kereta Api di Indonesia**

Tugas Akhir

Program Studi: Sarjana Sistem dan Teknologi Informasi

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

oleh :

Nafisah Nurul Hakim

NIM: 18215045

Telah disetujui dan disahkan sebagai laporan tugas akhir
di Bandung, 3 Agustus 2020

Pembimbing

Prof. Dr.Ir. Suhono Harso Supangkat, M.Eng.
NIP. 19621203 198811 1 001

Lembar Pernyataan Orisinalitas

Dengan ini saya menyatakan, Tugas Akhir ini adalah hasil karya sendiri dan semua referensi telah diacu dengan benar sesuai dengan kaidah penulisan ilmiah.

Bandung, 13 Agustus 2020

Nafisah Nurul Hakim
NIM. 18215045

ABSTRAK

Kereta api merupakan moda transportasi massal yang masih cukup diminati oleh masyarakat Indonesia hingga saat ini. Hal ini menyebabkan adanya urgensi terhadap penjaminan keselamatan penggunaan kereta api yang baik dari penyelenggaranya. Meskipun begitu, KNKT menemukan bahwa sebagian besar kecelakaan kereta api disebabkan oleh faktor prasarana, salah satunya adalah rel patah. Peninjauan lebih lanjut pada kejadian terkait memberikan hasil bahwa kejadian rel patah ini disebabkan oleh beberapa hal, di antaranya terdapat pada aspek manajemen dan organisasi.

Penelitian ini dilakukan untuk memberikan pandangan baru terhadap metode penyelesaian masalah pada aspek manajemen dan organisasi dengan menerapkan *machine learning* untuk melakukan penilaian kondisi rel. Metode penilaian yang dimaksud dalam penelitian ini adalah penggunaan model *machine learning* untuk melakukan prediksi kemungkinan terjadinya rel patah di suatu titik dengan kondisi-kondisi yang dimasukkan sebagai input pada model. Pengembangan model dilakukan menggunakan metodologi CRISP-DM dan beberapa teknik pemodelan yang saling dibandingkan hasilnya sehingga menghasilkan model paling tepat. Pengembangan model dilakukan menggunakan dataset kejadian patah rel pada tahun 2017 hingga 2019 yang disertai dengan detail teknis dan operasional. Lingkup lokasi observasi pada penelitian dibatasi pada daerah operasi kereta api di Sumatera Selatan, khususnya DIVRE III dan DIVRE IV.

Evaluasi pada hasil setiap model yang dilakukan di akhir penelitian memberikan kesimpulan bahwa random forest adalah teknik yang paling tepat untuk digunakan dalam melakukan pembuatan model analisis prediksi kejadian dan lokasi rel patah berdasarkan data yang digunakan.

Kata kunci: Kereta Api, Rel Patah, *Machine Learning*, CRISP-DM

KATA PENGANTAR

DAFTAR ISI

BAB I	PENDAHULUAN.....	I-1
I.1	Latar Belakang	I-1
I.2	Rumusan Masalah	I-2
I.3	Tujuan.....	I-3
I.4	Batasan Masalah.....	I-3
I.5	Sistematika Pembahasan.....	I-3
BAB II	STUDI LITERATUR	II-1
II.1	Machine Learning	II-1
II.2	Penelitian Terkait	II-9
BAB III	METODOLOGI.....	III-1
BAB IV	ANALISIS DAN PERANCANGAN	IV-1
IV.1	<i>Business Understanding</i>	IV-1
IV.2	<i>Data Understanding</i>	IV-4
IV.3	<i>Data Preparation</i>	IV-5
IV.4	<i>Modeling</i>	IV-5
IV.5	<i>Evaluation</i>	IV-6
IV.6	<i>Deployment</i>	IV-6
IV.6.1	Rancangan Sistem Prediksi Kejadian dan Lokasi Patah rel Kereta Api	IV-6
BAB V	IMPLEMENTASI dan evaluasi.....	V-1
V.1	<i>Data Preparation</i>	V-1
V.1.1	Kondisi data awal	V-1
V.1.2	Kondisi data akhir dan transformasi data	V-5
V.2	<i>Modeling</i>	V-7
V.2.1	<i>Data analysis and pre processing</i>	V-7
V.2.2	Pemodelan awal.....	V-14
V.2.3	Pemodelan akhir	V-16
V.3	<i>Evaluation</i>	V-22
V.4	<i>Deployment</i>	V-27
BAB VI	PENUTUP	VI-1
VI.1	Kesimpulan	VI-1
VI.2	Saran.....	VI-2
DAFTAR PUSTAKA		x
LAMPIRAN.....		xiii

DAFTAR GAMBAR

Gambar 1 <i>Machine Learning Approach</i> (Geron, 2017)	II-1
Gambar 2 Dataset yang sudah diberi label untuk <i>supervised learning</i> (Geron, 2017)	II-2
Gambar 3 <i>Dataset pembelajaran tanpa label untuk unsupervised learning</i> (Geron, 2017)	II-3
Gambar 4 <i>Semi supervised Learning</i> (Geron, 2017)	II-3
Gambar 5 <i>Reinforcement learning</i> (Geron, 2017)	II-4
Gambar 6 <i>Online learning</i> (Geron, 2017)	II-7
Gambar 7 <i>Instance-based learning</i> (Geron, 2017)	II-8
Gambar 8 <i>Model-based learning</i> (Geron, 2017)	II-8
Gambar 9 Hasil penelitian berjudul <i>Multivariate Statistical Model for Predicting Occurrence and Location of Broken Rails</i> (Dick, Barkan, Chapman, & Stehly, 2003)	II-10
Gambar 10 Hasil penelitian berjudul <i>A Prediction Model for Broken Rails and An Analysis of Their Economic Impact</i> (Schafer & Barkan, 2008)	II-11
Gambar 11 Hasil perbandingan metodologi <i>data mining</i> dalam penelitian KDD, SEMMA, and CRISP-DM: A Parallel Overview (Azevedo & Santos, 2008)	III-2
Gambar 12 Rangkaian proses metodologi CRISP-DM (Chapman, et al., 2000)	III-4
Gambar 13 Faktor penyebab kecelakaan kereta api (KNKT, 2016)	IV-2
Gambar 14 Gambaran umum sistem prediksi kejadian dan lokasi patah rel kereta api	IV-7
Gambar 15 Data murni kejadian rel patah	V-2
Gambar 16 Data murni <i>passing tonnage</i> DIVRE III 2017	V-2
Gambar 17 Data murni <i>passing tonnage</i> DIVRE IV 2017	V-3
Gambar 18 Data murni <i>passing tonnage</i> DIVRE III 2019	V-3
Gambar 19 Data murni <i>passing tonnage</i> DIVRE IV 2019	V-4
Gambar 20 Data murni gerbong DIVRE III dan IV 2017	V-4
Gambar 21 Data murni gerbong DIVRE III dan IV 2019	V-4
Gambar 22 Data final pemodelan	V-5
Gambar 23 Program pemodelan bagian inisiasi	V-7
Gambar 24 Program pemodelan bagian data analysis and <i>pre processing</i> (analisis data)	V-8
Gambar 25 Program pemodelan bagian data analysis and <i>pre processing</i> (penanganan missing value)	V-9
Gambar 26 Program pemodelan bagian data analysis and <i>pre processing</i> (penilaian fitur menggunakan <i>univariate statistical model</i>)	V-9
Gambar 27 Program pemodelan bagian data analysis and <i>pre processing</i> (hasil penilaian fitur menggunakan <i>univariate statistical model</i>)	V-10
Gambar 28 Program pemodelan bagian data analysis and <i>pre processing</i> (<i>heatmap</i>)	V-10
Gambar 29 Program pemodelan bagian data analysis and <i>pre processing</i> (hasil penggunaan <i>heatmap</i>)	V-10
Gambar 30 Program pemodelan bagian data analysis and <i>pre processing</i> (penyusunan data final untuk pembuatan model)	V-11
Gambar 31 Program pemodelan bagian data analysis and <i>pre processing</i> (eliminasi baris data yang berisi <i>outlier</i>)	V-12
Gambar 32 Program pemodelan bagian data analysis and <i>pre processing</i> (hasil eliminasi baris data yang berisi <i>outlier</i>)	V-12
Gambar 33 Program pemodelan bagian data analysis and <i>pre processing</i> (hasil penilaian fitur menggunakan <i>univariate statistical model</i> tanpa <i>outlier</i>)	V-13
Gambar 34 Program pemodelan bagian data analysis and <i>pre processing</i> (hasil penggunaan <i>heatmap</i> tanpa <i>outlier</i>)	V-13
Gambar 35 Program pemodelan bagian data analysis and <i>pre processing</i> (penyusunan data final untuk pembuatan model tanpa <i>outliers</i>)	V-14
Gambar 36 Pemodelan awal (regresi logistik)	V-14
Gambar 37 Pemodelan awal (<i>support vector machine</i>)	V-15
Gambar 38 Pemodelan awal (<i>decision tree</i>)	V-15
Gambar 39 Pemodelan awal (<i>random forest</i>)	V-15
Gambar 40 Pemodelan awal (<i>k-nearest neighbors</i>)	V-15
Gambar 41 Pemodelan akhir (<i>support vector machine</i> kernel rbf)	V-16
Gambar 42 Pemodelan akhir (<i>support vector machine</i> kernel sigmoid)	V-16
Gambar 43 Pemodelan akhir (<i>support vector machine</i> kernel linear)	V-16

Gambar 44 Pemodelan akhir (evaluasi model <i>support vector machine</i> dengan <i>outlier</i> kernel linear)	V-17
Gambar 45 Pemodelan akhir (evaluasi model <i>support vector machine</i> dengan <i>outlier</i> kernel rbf) ...	V-17
Gambar 46 Pemodelan akhir (evaluasi model <i>support vector machine</i> dengan <i>outlier</i> kernel sigmoid)	V-17
Gambar 47 Pemodelan akhir (evaluasi model <i>support vector machine</i> tanpa <i>outlier</i> kernel linear) ..	V-17
Gambar 48 Pemodelan akhir (evaluasi model <i>support vector machine</i> tanpa <i>outlier</i> kernel rbf)	V-17
Gambar 49 Pemodelan akhir (evaluasi model <i>support vector machine</i> tanpa <i>outlier</i> kernel sigmoid) ..	V-18
Gambar 50 Pemodelan akhir (program menentukan <i>n_estimator</i> dalam teknik <i>random forest</i>).....	V-18
Gambar 51 Pemodelan akhir (<i>n_estimator</i> untuk model <i>random forest</i> dengan <i>outlier</i>)	V-19
Gambar 52 Pemodelan akhir (<i>n_estimator</i> untuk model <i>random forest</i> tanpa <i>outlier</i>)	V-19
Gambar 53 Pemodelan akhir (program menentukan <i>random_state</i> dalam teknik <i>random forest</i> untuk model dengan <i>outlier</i>).....	V-19
Gambar 54 Pemodelan akhir (<i>random_state</i> untuk model <i>random forest</i> dengan <i>outlier</i>).....	V-20
Gambar 55 Pemodelan akhir (program menentukan <i>random_state</i> dalam teknik <i>random forest</i> untuk model tanpa <i>outlier</i>).....	V-20
Gambar 56 Pemodelan akhir (<i>random_state</i> untuk model <i>random forest</i> tanpa <i>outlier</i>).....	V-20
Gambar 57 Pemodelan akhir (program pemodelan <i>random forest</i> dengan <i>outlier</i>)	V-21
Gambar 58 Pemodelan akhir (program pemodelan <i>random forest</i> tanpa <i>outlier</i>).....	V-21
Gambar 59 Pemodelan akhir (program menentukan <i>n_neighbors</i> dalam teknik <i>k-nearest neighbor</i>)	V-21
Gambar 60 Pemodelan akhir (<i>n_neighbors</i> untuk model <i>k-nearest neighbor</i> dengan <i>outlier</i>)	V-22
Gambar 61 Pemodelan akhir (<i>n_neighbors</i> untuk model <i>k-nearest neighbor</i> tanpa <i>outlier</i>)	V-22
Gambar 62 Pemodelan akhir (program pemodelan <i>k-nearest neighbor</i>).....	V-22
Gambar 63 Evaluasi (model regresi logistik awal dengan <i>outlier</i>)	V-23
Gambar 64 Evaluasi (model <i>support vector machine</i> awal dengan <i>outlier</i>).....	V-23
Gambar 65 Evaluasi (model <i>decision tree</i> awal dengan <i>outlier</i>)	V-23
Gambar 66 Evaluasi (model <i>random forest</i> awal dengan <i>outlier</i>)	V-23
Gambar 67 Evaluasi (model <i>k-nearest neighbor</i> awal dengan <i>outlier</i>).....	V-23
Gambar 68 Evaluasi (model regresi logistik awal tanpa <i>outlier</i>).....	V-24
Gambar 69 Evaluasi (model <i>support vector machine</i> awal tanpa <i>outlier</i>).....	V-24
Gambar 70 Evaluasi (model <i>decision tree</i> awal tanpa <i>outlier</i>).....	V-24
Gambar 71 Evaluasi (model regresi <i>random forest</i> tanpa <i>outlier</i>).....	V-24
Gambar 72 Evaluasi (model <i>k-nearest neighbor</i> awal tanpa <i>outlier</i>).....	V-24
Gambar 73 Evaluasi (model regresi logistik akhir dengan <i>outlier</i>)	V-25
Gambar 74 Evaluasi (model <i>support vector machine</i> akhir dengan <i>outlier</i>)	V-25
Gambar 75 Evaluasi (model <i>decision tree</i> akhir dengan <i>outlier</i>).....	V-25
Gambar 76 Evaluasi (model <i>random forest</i> akhir dengan <i>outlier</i>).....	V-25
Gambar 77 Evaluasi (model <i>k-nearest neighbor</i> akhir dengan <i>outlier</i>).....	V-25
Gambar 78 Evaluasi (model regresi logistik akhir tanpa <i>outlier</i>)	V-26
Gambar 79 Evaluasi (model <i>support vector machine</i> akhir tanpa <i>outlier</i>)	V-26
Gambar 80 Evaluasi (model <i>decision tree</i> akhir tanpa <i>outlier</i>)	V-26
Gambar 81 Evaluasi (model <i>random forest</i> akhir tanpa <i>outlier</i>)	V-26
Gambar 82 Evaluasi (model <i>k-nearest neighbor</i> akhir tanpa <i>outlier</i>)	V-26
Gambar 83 <i>Deployment</i> (tampilan utama aplikasi).....	V-27
Gambar 84 <i>Deployment</i> (tampilan hasil jika ada kemungkinan rel patah).....	V-28
Gambar 85 <i>Deployment</i> (tampilan hasil jika tidak ada kemungkinan rel patah)	V-28

DAFTAR TABEL

Tabel 1 Pengelompokkan algoritma <i>machine learning</i> berdasarkan keterlibatan proses pembelajaran dengan supervisi manusia	II-5
Tabel 2 Kebutuhan fungsional sistem prediksi kejadian dan lokasi patah rel kereta api.....	IV-7
Tabel 3 Transformasi data.....	V-5
Tabel 4 Hasil evaluasi pengujian model awal dengan <i>outlier</i>	V-23
Tabel 5 Hasil evaluasi pengujian model awal tanpa <i>outlier</i>	V-24
Tabel 6 Hasil evaluasi pengujian model akhir dengan <i>outlier</i>	V-25
Tabel 7 Hasil evaluasi pengujian model awal tanpa <i>outlier</i>	V-26

BAB I

PENDAHULUAN

I.1 Latar Belakang

Kereta api merupakan salah satu moda transportasi massal yang masih cukup diminati masyarakat hingga saat ini (VIVA, 2017). Salah satu penyebabnya adalah penggunaan kereta api yang membuat perjalanan menghabiskan waktu secara lebih efektif dibandingkan dengan penggunaan mode transportasi lain. Selain itu, kereta api termasuk mode transportasi publik yang hingga saat ini biayanya cukup terjangkau oleh semua level masyarakat dengan berbagai jumlah penghasilan (Aziz, 2017).

Besarnya minat masyarakat terhadap kereta api dapat dibuktikan dengan meningkatnya jumlah penumpang kereta dari tahun ke tahun. Pada 2016, penumpang kereta api tercatat sebanyak 352,31 juta penumpang (Purnamasari, 2018). Jumlah ini selanjutnya meningkat hingga 394 juta penumpang pada tahun 2017. Peningkatan jumlah penumpang terus berlanjut ke tahun 2018 yang ditunjukkan dengan naiknya jumlah penumpang hingga 425 juta. Hasil observasi terakhir dari jumlah penumpang kereta api tahunan pada 2019 masih menampakkan peningkatan jumlah penumpang dengan nilai akhir sebesar 429 juta (Pebrianto, 2020).

Banyaknya masyarakat yang memilih kereta api sebagai moda transportasi meningkatkan urgensi penjaminan keselamatan dalam penggunaan kereta api dari pihak para penyedia layanan yang terlibat. Namun, berdasarkan data investigasi kecelakaan perkeretaapian tahun 2010-2016 yang dikeluarkan oleh Komite Nasional Keselamatan Transportasi (KNKT), sebagian besar penyebab kecelakaan masih didominasi oleh kesalahan dalam faktor prasarana, termasuk patahnya rel di lokasi-lokasi terkait. Patah rel yang dimaksud adalah berupa patah total atau patah gompal dengan penyebab sebagai berikut :

1. Pelubangan baut plat sambung yang tidak sesuai sehingga terbentuk awal retakan (crack initiation) pada tepi lubang kasar pada web rail, penjalaran retakan (crack propagation) hingga patah akhir (total disintegration)

2. Benturan yang berulang antara plat sambung dan kepala rel selama dilewati kereta api. Benturan disebabkan oleh penggunaan jumlah baut yang tidak sesuai dengan persyaratan teknis.

KNKT telah memberikan detail penyebab yang menghasilkan kondisi tersebut, namun beberapa kondisi yang menjadi fokus dalam penelitian ini adalah berupa hasil temuan inspeksi KNKT dengan studi kasus pada jalur KA di Divre IV tanjungkarang sebagai berikut.

1. Tidak dilaksanakannya Keputusan Direksi PT.KAI (Persero) tahun 2013 berupa pelaporan risiko keselamatan dalam bentuk profil risiko dan Keputusan Direksi PT.KAI (Persero) tahun 2015 berupa pelaporan Level of Safety secara konsisten sehingga tidak terdeteksinya kondisi prasarana yang berisiko tinggi terhadap keselamatan dan menjadi prioritas utama dalam tindakan perawatan.
2. Tidak dijelaskannya standar keandalan dari perawatan berdasarkan kelas jalur kereta api sehingga tidak ada acuan/target dalam mempertahankan konsistensi hasil perawatan.
3. Belum dilaksanakannya uji berkala dari jalur kereta api di wilayah III.2.10 Resort Peninjauan, Sub Divre III.2 (Divre IV) Tanjungkarang.

Model prediksi analisis patah rel yang didesain dalam penelitian ini diharapkan mampu membantu penyelesaian permasalahan dalam industri perkereta apian dalam bidang prasarana khususnya mengenai ketiga hal tersebut dengan memanfaatkan teknologi berupa *machine learning*.

I.2 Rumusan Masalah

Rumusan masalah dalam penelitian ini adalah sebagai berikut.

1. Fitur data apa saja yang yang memengaruhi model *machine learning* untuk analisis prediksi kemungkinan kejadian patah rel kereta api?
2. Apa teknik pemodelan *machine learning* yang paling tepat untuk mengembangkan model *machine learning* untuk prediksi kemungkinan kejadian patah rel kereta api?

3. Bagaimana spesifikasi konfigurasi yang tepat pada model *machine learning* prediksi kemungkinan kejadian patah rel kereta api sehingga menghasilkan nilai akurasi paling besar?
4. Berapa nilai akurasi model paling besar yang dapat dicapai dari hasil pemodelan *machine learning* untuk analisis prediksi kemungkinan kejadian patah rel kereta api?

I.3 Tujuan

Tujuan penelitian ini adalah mengembangkan sistem untuk mengidentifikasi area rel yang memiliki kemungkinan besar menyebabkan kejadian rel patah berdasarkan rekaman data kejadian terkait yang sudah terjadi sebelumnya dan faktor-faktor lain yang memiliki kemungkinan memengaruhi hal tersebut.

I.4 Batasan Masalah

Batasan masalah dalam penelitian tugas akhir ini adalah sebagai berikut.

1. Hasil akhir dari penelitian tugas akhir ini adalah berupa aplikasi berbasis *machine learning* sederhana dengan fungsi untuk melakukan prediksi kejadian dan lokasi rel patah
2. Pengembangan model *machine learning* untuk analisis prediksi kemungkinan kejadian patah rel kereta api dilakukan menggunakan data kejadian patah rel pada tahun 2017, 2018, dan 2019
3. Prediksi di kejadian dan lokasi rel patah ditunjukkan dalam bentuk kemungkinan terjadinya patah rel pada lokasi dengan atribut-atribut yang menjadi *input* pada saat penggunaan fungsi prediksi. Kemungkinan terjadinya rel patah ditunjukkan dalam bentuk keterangan hasil klasifikasi antara terjadinya rel patah atau tidak terjadinya rel patah
4. Lokasi yang ditinjau dalam penelitian dibatasi pada daerah operasi kereta api di Sumatera Selatan, khususnya DIVRE III dan DIVRE IV

I.5 Sistematika Pembahasan

Penelitian ini dibahas dalam laporan yang dibagi menjadi lima bagian yang meliputi :

Bab satu berisi pendahuluan yang membahas isi penelitian secara singkat. Bagian ini didahului oleh latar belakang yang mendeskripsikan kondisi penyebab dibutuhkanannya

penelitian ini. Selanjutnya bagian ini mendeskripsikan masalah kunci yang menjadi inti bahasan dalam penelitian ini dan dilanjutkan oleh deskripsi tujuan penelitian ini. Bagian ini ditutup oleh deskripsi sistematika pembahasan penelitian dalam laporan yang menjadi salah satu hasil akhir penelitian.

Bab dua berisi studi literatur yang membahas landasan teknik, teori, dan metode yang digunakan dalam penelitian ini. Bagian pertama pada bab ini mendeskripsikan teori-teori ilmiah yang menjadi dasar melakukan penelitian dan penarikan kesimpulan. Selanjutnya, bagian kedua mendeskripsikan penelitian-penelitian yang berkaitan dengan penelitian ini dan menjadi bahan penting dalam pelaksanaannya.

Bab tiga berisi pembahasan metode penelitian dalam pembuatan model prediksi patah rel kereta api di Indonesia. Bagian ini didahului oleh analisis dan hasil analisis dari permasalahan yang dikaji dalam penelitian ini yang kemudian dilanjutkan oleh deskripsi solusi dalam level konseptual. Deskripsi ini disertai juga dengan penjelasan dari pendekatan yang digunakan dalam penelitian dalam solusi terkait.

Bab empat berisi pembahasan hasil penelitian yang telah dilakukan berdasarkan metode pada bab tiga. Bagian ini meliputi pembahasan parameter yang paling efektif untuk digunakan dalam pembuatan model prediksi patah rel kereta api di Indonesia dan model yang dihasilkan. Bagian ini ditutup dengan evaluasi dari pelaksanaan penelitian sebagai referensi untuk penelitian terkait di masa depan.

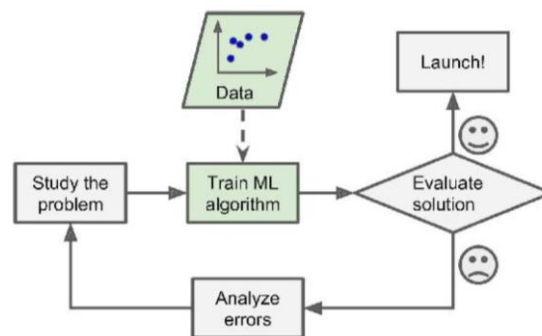
Bab lima berisi penutup laporan penelitian pembuatan model prediksi patah rel kereta api di Indonesia. Bagian ini meliputi kesimpulan yang didapatkan dari penelitian yang telah dilakukan berdasarkan rumusan masalah yang telah dideskripsikan dan saran untuk pengembangan penelitian terkait di masa depan.

BAB II

STUDI LITERATUR

II.1 Machine Learning

Machine learning adalah sebuah konsep pemrograman yang mampu memberikan komputer kemampuan untuk melakukan pembelajaran berdasarkan data yang telah tersedia. Menurut Arthur Samuel, salah seorang peneliti yang menjadi pelopor dalam penelitian terkait kecerdasan buatan, *machine learning* secara umum adalah bidang studi yang memungkinkan pemberian kemampuan kepada komputer untuk belajar tanpa diprogram secara eksplisit (Geron, 2017). *Machine learning* adalah sebuah bagian dari *artificial intelligence* yang menggunakan berbagai macam teknik optimisasi, probabilitas, dan statistika yang memungkinkan computer untuk belajar dari kasus-kasus yang telah terjadi dan mendeteksi pola yang sulit diketahui dari *dataset* yang besar serta kompleks (Nithya, 2016).



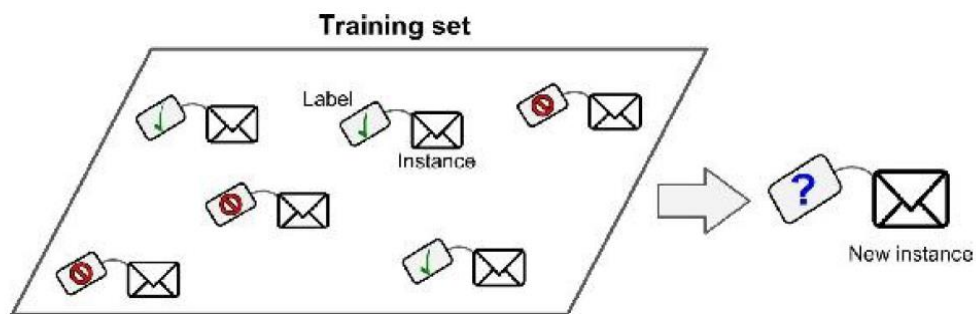
Gambar 1 *Machine Learning Approach* (Geron, 2017)

Sistem *machine learning* dapat diklasifikasi berdasarkan beberapa kriteria. Kriteria tersebut meliputi keterlibatan proses pembelajaran dengan supervisi manusia, kemampuan sistem dalam mempelajari aliran data yang masuk secara bertahap, dan proses prediksi hasil oleh sistem berbasis *machine learning*. Penggunaan kriteria-kriteria tersebut tidak bersifat eksklusif dan sebuah metode *machine learning* dapat memiliki lebih dari satu karakter yang meliputi dua atau lebih kriteria berbeda (Geron, 2017).

Berdasarkan keterlibatan proses pembelajaran dengan supervisi manusia, sistem *machine learning* dibagi menjadi empat kategori sebagai berikut (Geron, 2017):

1. *Supervised Learning*

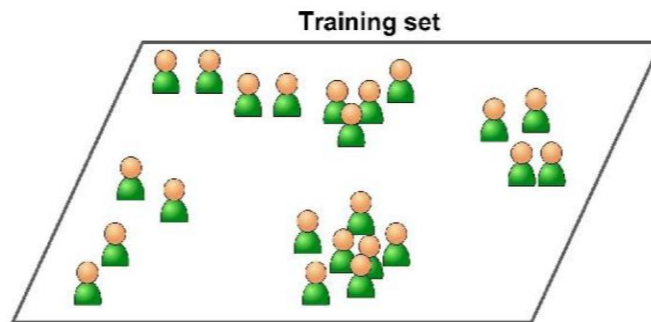
Sistem yang memiliki metode pembelajaran *supervised learning* memerlukan pendampingan entitas eksternal sistem dalam proses pembelajarannya, khususnya pada proses pengolahan data awal. Pada sistem ini, dataset yang ada dibagi menjadi dua untuk proses pembelajaran mesin dan pengujian. Dataset pembelajaran mesin memiliki variable berupa hasil prediksi atau klasifikasi. Sistem yang menerapkan *supervised learning* melakukan proses pembelajaran dengan menganalisis pola dari dataset pembelajaran dan menerapkannya pada proses prediksi atau klasifikasi menggunakan dataset pengujian (Dey, 2016). Variabel hasil yang ada di dalam dataset pembelajaran disebut label.



Gambar 2 Dataset yang sudah diberi label untuk *supervised learning* (Geron, 2017)

2. *Unsupervised Learning*

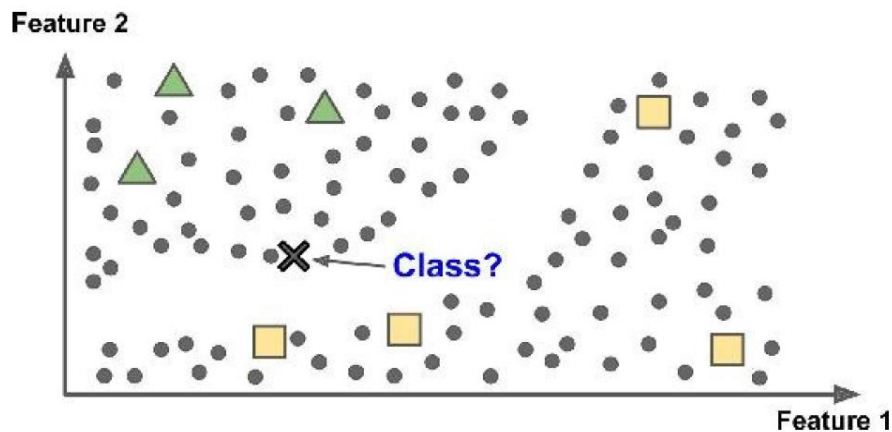
Sistem yang memiliki metode pembelajaran *unsupervised machine learning* mempelajari beberapa fitur langsung dari data (Dey, 2016). Algoritma yang digunakan dalam sistem ini akan langsung menganalisis hubungan antar data yang sudah ada dan melakukan klasifikasi hasil tanpa menggunakan label ketika dataset baru dimasukan. Ketika digunakan untuk melakukan prediksi atau klasifikasi data baru, sistem menentukan hasilnya berdasarkan kedekatan karakteristik data tersebut dengan karakteristik setiap kelas yang telah dianalisis oleh sistem dalam proses pembelajaran.



Gambar 3 Dataset pembelajaran tanpa label untuk *unsupervised learning* (Geron, 2017)

3. *Semi Supervised Learning*

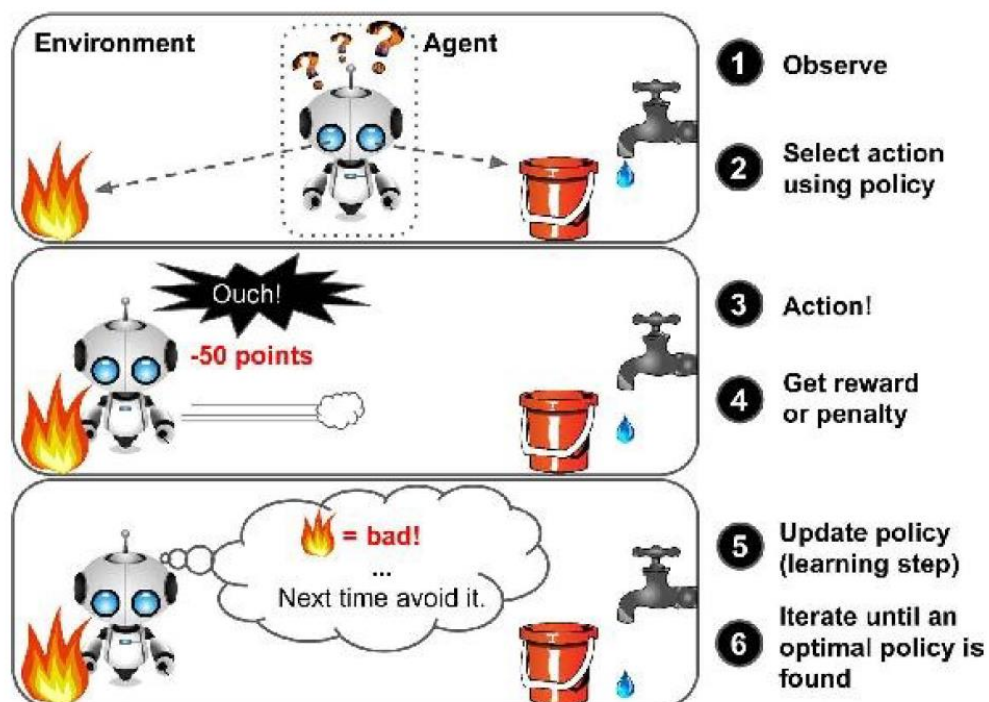
Sistem dengan metode pembelajaran *semi supervised machine learning* dapat mengolah data tidak seluruhnya memiliki label. Terkait hal tersebut, biasanya data yang memiliki label sangat sedikit dibandingkan dengan data yang tidak memiliki label (Geron, 2017). Metode ini dapat menjadi sangat berguna ketika data tanpa label sudah tersedia dan memperoleh data berlabel membutuhkan proses yang sulit (Dey, 2016).



Gambar 4 *Semi supervised Learning* (Geron, 2017)

4. Reinforcement Learning

Proses pembelajaran dalam metode *reinforcement learning* dilakukan oleh sistem dengan membiarkan sistem dalam sebuah lingkungan dan memilih serta melakukan *action* tertentu. Sistem akan diberikan *reward* berdasarkan pilihan *action* yang dilakukan (*reward* positif untuk pilihan yang benar dan *reward* negatif untuk pilihan yang salah). Seiring waktu berjalan sistem akan mempelajari strategi terbaik, yang disebut *policy*, untuk mendapatkan *reward* positif paling banyak dalam lingkungan pembelajaran. *Policy* nantinya digunakan untuk mendefinisikan *action* yang harus dipilih oleh sistem di situasi yang dihadapi di dunia nyata (Geron, 2017). Hal ini menunjukkan bahwa fitur penting dan unik dari metode pembelajaran *reinforcement learning* adalah dua karakter yang dimiliki metode tersebut, yakni *trial and error search* dan *delayed reward* (Sutton, 1992).



Gambar 5 Reinforcement learning (Geron, 2017)

Berikut adalah daftar algoritma *machine learning* yang sering digunakan yang dikelompokkan berdasarkan keterlibatan proses pembelajaran dengan supervisi manusia (Nithya, 2016).

Tabel 1 Pengelompokkan algoritma *machine learning* berdasarkan keterlibatan proses pembelajaran dengan supervisi manusia

No.	<i>Type of Learning</i>	<i>Mode/ Method</i>	<i>Extensively Used Algorithm</i>
1	<i>Supervised Learning</i>	<i>Decision Tree Technoque</i>	<ul style="list-style-type: none"> • Classification and Regression Tree (CART) • Iterative Dichotomiser 3 (ID3) • C4.5 and C5.0 • Chi-squared Automatic Interaction Detection (CHAID) • Decision Stump • M5 • Conditional Decision Trees
		<i>Bayesian Methods</i>	<ul style="list-style-type: none"> • Naive Bayes (NB) • Gaussian Naive Bayes • Multinomial Naive Bayes • Averaged One-dependence Estimators (AODE) • Bayesian Belief Network (BBN) • Bayesian Network (BN)
		<i>Artificial Neural Network</i>	<ul style="list-style-type: none"> • Perceptron • Back-Propagation • Hopfield Network

<i>No.</i>	<i>Type of Learning</i>	<i>Mode/ Method</i>	<i>Extensively Used Algorithm</i>
			<ul style="list-style-type: none"> • Radial Basis Function Network (RBFN)
		<i>Instance Based Learning</i>	<ul style="list-style-type: none"> • K - Nearest Neighbour (kNN) • Learning Vector Quantization (LVQ) • Self-Organizing Map (SOM) • Locally Weighted Learning (LWL)
		<i>Ensemble Methods</i>	<ul style="list-style-type: none"> • Boosting • Bootstrapped Aggregation (Bagging) • AdaBoost • Stacked Generalization (blending) • Gradient Boosting Machines (GBM) • Gradient Boosted Regression Trees (GBRT) • Random Forest
2	<i>Unsupervised Learning</i>	<i>Clustering Methods</i>	<ul style="list-style-type: none"> • k-Means • k-Medians • Expectation Maximization (EM) • Hierarchical Clustering
3	<i>Supervised/Unsupervised Learning</i>	<i>Regression Algorithms</i>	<ul style="list-style-type: none"> • Ordinary Least Squares Regression (OLSR) • Linear Regression

<i>No.</i>	<i>Type of Learning</i>	<i>Mode/ Method</i>	<i>Extensively Used Algorithm</i>
			<ul style="list-style-type: none"> • Logistic Regression • Stepwise Regression

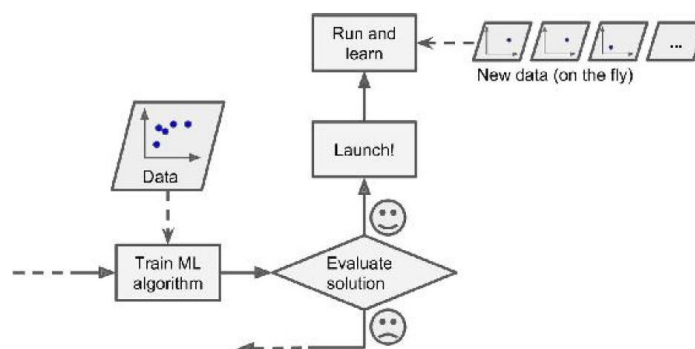
Berdasarkan kemampuan sistem dalam mempelajari aliran data yang masuk secara bertahap, sistem *machine learning* dibagi menjadi dua kategori sebagai berikut (Geron, 2017) :

1. *Batch Learning*

Dalam metode pembelajaran *batch learning*, sistem tidak bisa melakukan pembelajaran secara bertahap. Proses belajar sistem harus langsung dilakukan oleh seluruh data yang tersedia. Penggunaan metode ini secara umum menggunakan waktu dan sumber daya computer yang cukup banyak sehingga biasanya dilakukan secara *offline*. Setelah sistem dirilis dalam tahap produksi, sistem akan berjalan tanpa proses belajar kembali. Jika sistem perlu belajar kembali, maka sistem lama harus dihentikan dan diganti sepenuhnya oleh sistem baru yang telah dilatih dengan menggunakan data yang telah disempurnakan.

2. *Online Learning*

Dalam metode pembelajaran *online learning*, sistem dapat melakukan pembelajaran secara bertahap dengan pemberian data secara sekuensial. Setiap proses pembelajaran dapat dilakukan dengan cepat dan murah sehingga sistem dapat terus menerus belajar tentang data terbaru dalam waktu yang tidak lama sejak kemunculan data tersebut.

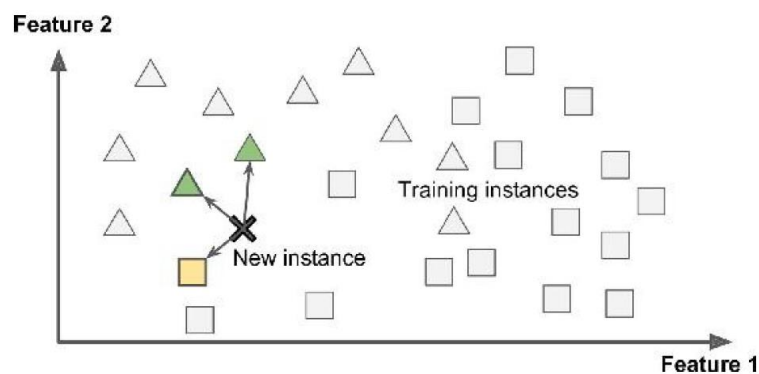


Gambar 6 *Online learning* (Geron, 2017)

Berdasarkan metode yang digunakan dalam proses prediksi hasil, sistem berbasis *machine learning* dibagi menjadi dua kategori sebagai berikut (Geron, 2017):

1. *Instance-Based Learning*

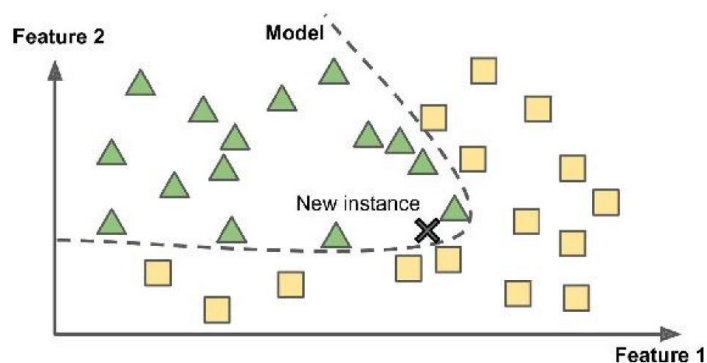
Proses penarikan kesimpulan dalam metode *instance-based learning* dilakukan dengan membandingkan tingkat kesamaan karakteristik data baru dengan data yang sudah ada. Kesimpulan yang diambil dari penggunaan metode ini adalah data baru dianggap sama dengan *instance* yang memiliki karakteristik paling mendekati data baru tersebut.



Gambar 7 *Instance-based learning* (Geron, 2017)

2. *Model-Based Learning*

Sistem dengan *model-based learning* melakukan penarikan kesimpulan dengan menggunakan model hasil pembelajaran menggunakan dataset *training*. Pada proses pembelajaran, sistem akan terlebih dahulu mengolah dataset training untuk menemukan pola atau hubungan antar fitur dan menyimpulkan model yang paling tepat merepresentasikan data tersebut. Pada proses penerapan sistem, prediksi hasil dari masukan yang diberikan ke dalam sistem dilakukan dengan penerapan data yang masuk dalam model tersebut.



Gambar 8 *Model-based learning* (Geron, 2017)

II.2 Penelitian Terkait

Penelitian ini dilakukan berdasarkan beberapa referensi utama berupa penelitian terkait model prediksi lokasi dan patah rel kereta serta penerapan *machine learning* yang telah dilakukan sebelumnya. Berikut adalah penelitian-penelitian yang dimaksud.

1. *Multivariate Statistical Model for Predicting Occurrence and Location of Broken Rails*

Penelitian ini dilakukan oleh C. Tyler Dick, Christopher P.L. Barkan, Edward R. Chapman dan Mark P. Stenly pada tahun 2003. Penelitian ini dilakukan untuk meningkatkan kualitas pelayanan pengadaan jasa transportasi kereta di Amerika Serikat dalam bidang keamanan. Model prediksi yang dikembangkan dalam penelitian ini diharapkan bisa membuat proses deteksi patah rel dapat dilakukan secara lebih efektif dengan menggunakan seluruh sumber daya yang ada.

Pengembangan model prediksi dilakukan dengan menggunakan data milik Burlington Northern Santa Fe terkait *service failure* dengan detail informasi berupa tanggal, lokasi, tipe *service failure*, serta data teknis dan operasional dari lokasi setiap kejadian selama dua tahun sebelumnya. Penggunaan data *service failure* dalam pengembangan model menjadikan penelitian ini tepat untuk dijadikan referensi dalam pengerjaan tugas akhir ini. Pengembangan model dalam penelitian ini dilakukan menggunakan *Statistical Analysis System* (SAS) dan prosedur LOGISTIC.

Berikut adalah model prediksi yang dihasilkan dalam penelitian ini.

$$P_{SF2} = \frac{e^U}{(1 + e^U)}$$

where

P_{SF2} = probability that a service failure occurred at a particular point during the study period;

$U = Z + Y$;

$Z = -4.569$, model-specific constant;

$Y = 0.059A + 0.025AC - 0.000084^2C^2 + 5.1017/S + 217.9W/S - 3861.6W^2/S^2 + 0.897(2N - 1) - 1.108P/S$;

A = rail age (years);

C = degree of curvature (= 0 for tangent);

T = annual traffic [million gross tons (MGT)];

S = rail weight (pounds);

$W = 4T/L$ = annual number of wheel passes (millions);

$P = L(1 + V/100)$ = estimated average dynamic wheel load;

$N = 1$ if at turnout, 0 if not at turnout;

L = tons per car; and

V = track speed.

Gambar 9 Hasil penelitian berjudul *Multivariate Statistical Model for Predicting Occurrence and Location of Broken Rails* (Dick, Barkan, Chapman, & Stehly, 2003)

Perbedaan antara penelitian ini dan penelitian tugas akhir yang sedang dilakukan terletak pada keluaran dari kedua penelitian yang berbeda. Penelitian ini memiliki keluaran berupa model statistika untuk memprediksi kejadian dan lokasi patah rel. Penelitian tugas akhir memiliki keluaran berupa sistem berbasis *machine learning* yang dapat melakukan pemodelan data dan dapat digunakan secara langsung dalam melakukan prediksi.

2. *A Prediction Model for Broken Rails and an Analysis of Their Economic Impact*

Penelitian ini dilakukan oleh Darwin H. Schafer II dan Christopher P. L. Barkan pada tahun 2008. Penelitian ini merupakan penelitian lanjutan dari penelitian sebelumnya yang berjudul *Multivariate Statistical Model for Predicting Occurrence and Location of Broken Rails*. *Objective* penelitian ini adalah untuk mengembangkan model yang bisa digunakan untuk mengidentifikasi lokasi di jaringan rel yang memiliki kemungkinan terjadinya patah rel paling tinggi berdasarkan data *service failure* dan faktor-faktor lain yang kemungkinan berpengaruh. Penggunaan data *service failure* dalam pengembangan model menjadikan penelitian ini tepat untuk dijadikan referensi dalam pengerjaan tugas akhir ini. Pemodelan dalam penelitian ini dilakukan dengan menggunakan Statistical Analysis Software (SAS) dan prosedur LOGISTIC.

Sebelum penelitian dilakukan, model yang telah dihasilkan sebelumnya diuji kembali untuk memastikan tingkat akurasi model saat menggunakan dataset yang berbeda dan lebih baru (data tahun 2005 dan 2006). Hasil pengujian tersebut menunjukkan bahwa tingkat akurasi menurun secara drastis dari sebelumnya. Tingkat akurasi yang sebelumnya melebihi 87,4%, menurun menjadi 54,8% dalam pengujian tersebut. Hasil pengujian ditindaklanjuti dengan pemeriksaan lebih lanjut terhadap dataset *service failure* yang digunakan dalam penelitian sebelumnya. Pemeriksaan ini memberikan hasil bahwa ada kemungkinan dataset sebelumnya belum mencakup semua *service failure* yang terjadi dalam jaringan rel pada periode dataset yang digunakan. Penemuan ini menghasilkan kesimpulan terkait dibutuhkan pengembangan model statistik yang baru dan lebih *update* untuk memprediksi lokasi terjadinya *service failure*.

Berikut adalah model prediksi yang dihasilkan dalam penelitian ini.

$$Z^* = Z + \ln\left(\frac{P_{SF2}}{1 - P_{SF2}}\right) = 4.94 + \ln\left(\frac{.00543}{1 - .00543}\right) = -0.270 \quad (\text{Eq. 1})$$

$$P_{SF2} = \frac{e^u}{(1 + e^u)} \quad (\text{Eq. 2})$$

$$U = Z^* - 0.0454S - 1.35R - 0.0106A + 0.00899T + 0.0232L + 1.61I + 0.823G + 1.63B \quad (\text{Eq. 3})$$

where,

$Z^* = -0.270$, adjusted model constant

$Z = 4.94$, model specific constant

P_{SF2} = probability that a service failure occurred during a four-year period

S = rail weight (in pounds per yard)

R = rail type (1 if welded, 0 if bolted)

A = rail age (in years)

T = annual traffic (in million gross tons)

L = weight of car (in tons)

I = presence of an ultrasonic defect in the last three years (1 if present, 0 otherwise)

G = presence of a geometric defect in the last three years (1 if present, 0 otherwise)

B = presence of a bridge within 200 feet of segment (1 if present, 0 otherwise)

Gambar 10 Hasil penelitian berjudul *A Prediction Model for Broken Rails and An Analysis of Their Economic Impact* (Schafer & Barkan, 2008)

Sebagaimana dengan penelitian sebelumnya, perbedaan antara penelitian ini dan penelitian tugas akhir yang sedang dilakukan terletak pada keluaran dari kedua penelitian yang berbeda. Penelitian ini memiliki keluaran berupa model statistika untuk memprediksi kejadian dan lokasi patah rel. Penelitian tugas akhir memiliki keluaran berupa sistem berbasis *machine learning* yang dapat melakukan pemodelan data dan dapat digunakan secara langsung dalam melakukan prediksi.

3. *A Comparative Study of Machine Learning Techniques for Aviation Applications*

Penelitian ini dilakukan oleh Apoorv Maheshwari, Navindran Davendralingam, dan Daniel A. DeLaurentis pada tahun 2018. Penelitian dilakukan untuk keperluan berupa *Aviation Technology, Integration, and Operations Conference* yang diselenggarakan pada tahun 2018 oleh AIAA Aviation Forum di Atlanta, Georgia. *Objective* penelitian ini adalah untuk membandingkan penggunaan bermacam-macam teknik *machine learning* untuk penyelesaian masalah dalam bidang penerbangan dengan menggunakan permasalahan berupa pemodelan *air travel demand* sebagai contoh. Keluaran dari penelitian ini adalah berupa evaluasi berbagai teknik *machine learning* dalam pemodelan *air travel demand* berdasarkan tolak ukur sebagai berikut.

- *Accuracy in general*
- *Speed of learning*
- *Speed of classification/prediction*
- *Tolerance to irrelevant attributes*
- *Overfitting Issues*
- *Readability/Transparency of knowledge*
- *Ease of use – Tuning hyperparameters*

Penggunaan *machine learning* secara umum sebagai metode penyelesaian masalah dalam penelitian ini menjadikan penelitian ini tepat untuk dijadikan referensi dalam pengerjaan tugas akhir ini. Penelitian ini memberikan informasi berupa hal-hal dasar terkait penerapan *machine learning* dalam penerapan masalah. Salah satunya adalah berupa prinsip berupa tidak adanya

satu algoritma yang lebih unggul dari algoritma lain dalam menyelesaikan seluruh dataset.

BAB III

METODOLOGI

Metodologi yang dipilih dalam penelitian ini adalah CRISP-DM (Cross-Industry Standard Processing for Data Mining). CRISP-DM adalah metodologi standar untuk melakukan *data mining* dalam industri terkait hal tersebut (IBM, 2012). Metodologi ini dikembangkan pada tahun 1996 oleh Daimler Chrysler, SpSS Inc. (saat itu bernama sebagai perusahaan bernama Intergral Solusion Limited), dan NCR Corporation selalu praktisi dan perusahaan yang sudah berpengalaman dalam bidang *data mining* pada masa tersebut (Chapman, et al., 2000).

Metodologi *data mining* dipilih sebagai metodologi penelitian dengan pertimbangan utama berupa keluaran penelitian yang fokus utamanya adalah fitur aplikasi sederhana yang mampu memprediksi kejadian dan lokasi patah rel di Indonesia menggunakan pemodelan algoritma *machine learning*. Pengertian *data mining* secara umum adalah proses penemuan model untuk data tertentu. Pemodelan dalam *data mining* sering dilakukan dengan menggunakan algoritma *machine learning* (Leskocev, Rajaraman, & Ullman, 2014). Berdasarkan khususnya cakupan metodologi *data mining* terhadap pemodelan data, memungkinkannya penerapan algoritma *machine learning* dalam metodologi *data mining*, dan fokus utama serta batasan keluaran penelitian, *data mining* dianggap tepat untuk menjadi metodologi dalam penelitian ini.

Pemilihan CRISP-DM sebagai metodologi *data mining* yang digunakan dalam penelitian ini dilakukan berdasarkan penelitian berjudul KDD, SEMMA, and CRISP-DM : *A parallel overview* yang ditulis oleh Ana Azevedo dan Manuel Filipe Santos. Penelitian ini dipublikasikan di IADIS *European Conference in Data Mining* pada tahun 2008 di Kota Amsterdam, Belanda. Penelitian ini membahas mengenai ulasan singkat dan perbandingan antara KDD, SEMMA, dan CRISP-DM selalu metodologi terkenal yang paling sering digunakan oleh praktisi *data mining*.

Pada penelitian tersebut disimpulkan bahwa SEMMA dan CRISP-DM dapat dilihat sebagai sebuah proses dalam KDD setelah ditemukan bahwa *data mining* hanya sebuah proses dalam KDD. Di antara dua metodologi *data mining* yang tersisa, CRISP-DM disimpulkan sebagai metodologi yang lebih sempurna dibandingkan

dengan SEMMA. Hal ini disebabkan CRISP-DM memiliki fase *business understanding* dan *deployment* yang sangat penting dilakukan dalam penelitian jika ditinjau dari sudut pandang bisnis yang tidak ada padanan prosesnya pada SEMMA. Hal ini menjadi pertimbangan utama dipilihnya CRISP-DM sebagai metodologi penelitian tugas akhir ini.

KDD	SEMMA	CRISP-DM
Pre KDD	-----	Business understanding
Selection	Sample	Data Understanding
Pre processing	Explore	
Transformation	Modify	Data preparation
Data mining	Model	Modeling
Interpretation/Evaluation	Assessment	Evaluation
Post KDD	-----	Deployment

Gambar 11 Hasil perbandingan metodologi *data mining* dalam penelitian KDD, SEMMA, and CRISP-DM: A Parallel Overview (Azevedo & Santos, 2008)

Metodologi CRISP-DM memiliki rangkaian fase sebagai berikut (Chapman, et al., 2000).

1) *Business understanding*

Fase pertama ini berfokus pada kegiatan memahami *objective* pelaksanaan *data mining* dan seluruh kebutuhan terkait yang ditinjau dari sudut pandang bisnis. Informasi ini kemudian diproses menjadi definisi masalah dalam kegiatan *data mining* dan rencana awal yang dibuat untuk mencapai tujuan yang telah diidentifikasi.

2) *Data understanding*

Fase ini dimulai dengan proses pengumpulan data awal dan melanjutkannya dengan memahami data yang tersedia, mengidentifikasi masalah-masalah terkait kualitas data, dan mendeteksi subset yang menarik untuk dijadikan hipotesis terkait informasi yang tersembunyi.

3) *Data preparation*

Fase ini mencakup seluruh aktivitas yang dibutuhkan untuk membangun dataset final yang akan digunakan dalam proses pemodelan. Kegiatan yang dilakukan dalam fase ini biasanya dilakukan secara berulang kali tanpa urutan yang kaku. Kegiatan tersebut meliputi pembuatan tabel dan rekaman,

pemilihan atribut, serta transformasi dan pembersihan data untuk alat pemodelan.

4) *Modeling*

Pada fase ini dilakukan pemilihan dan penerapan berbagai teknik pemodelan data dengan parameter yang sudah dikalibrasi dengan konfigurasi yang dapat membuat hasil pemodelan berada pada tingkat akurasi paling optimal. Biasanya, setiap tipe permasalahan *data mining* dapat diselesaikan oleh beberapa teknik pemodelan. Penerapan beberapa teknik pemodelan juga memiliki syarat-syarat terkait kondisi data yang perlu dipenuhi. Hal ini menyebabkan perlunya kembali ke tahap pemodelan ketika fase-fase berikutnya sedang berjalan.

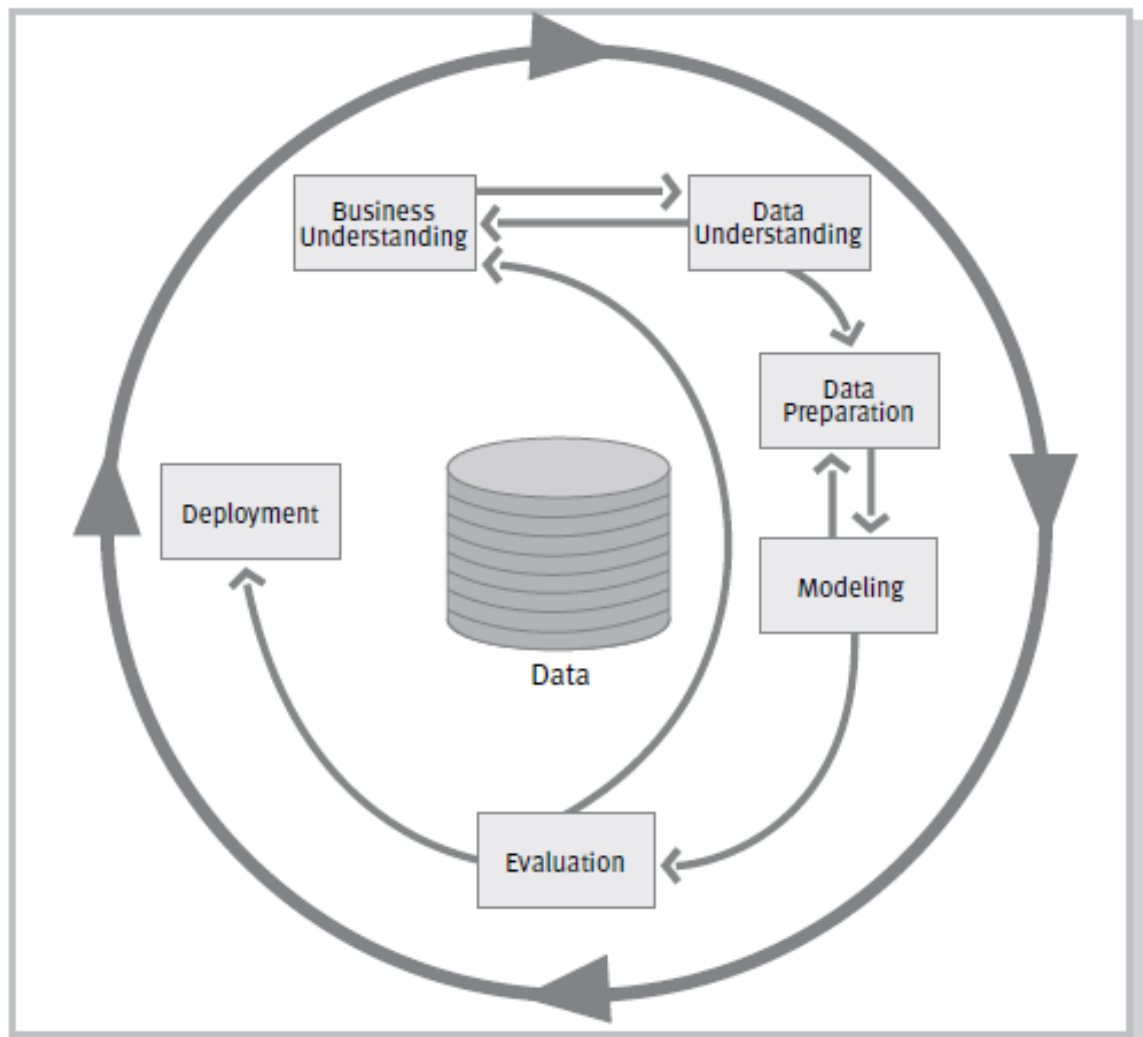
5) *Evaluation*

Pada fase ini dilakukan evaluasi atas model-model yang telah dibuat. Evaluasi dilakukan untuk memastikan bahwa ada model yang telah memenuhi *business objective* yang telah ditentukan di awal. Pada akhir fase ini, model final yang akan digunakan dalam *data mining* ditentukan berdasarkan hasil evaluasi tersebut.

6) *Deployment*

Pada fase ini dilakukan penerapan model terhadap media yang dapat memungkinkan pengguna memahami cara menggunakan model yang telah dibuat. Bentuk kegiatan *deployment* didasarkan kembali pada daftar kebutuhan yang telah ditentukan pada fase awal. Bentuk kegiatan *deployment* bisa dilakukan dalam berbagai bentuk mulai dari sebatas laporan, hingga penerapan sistem untuk interaksi langsung antara sistem dan pengguna.

Berikut adalah alur pelaksanaan fase CRISP-DM dalam bentuk visualisasi berupa gambar.



Gambar 12 Rangkaian proses metodologi CRISP-DM (Chapman, et al., 2000)

BAB IV

ANALISIS DAN PERANCANGAN

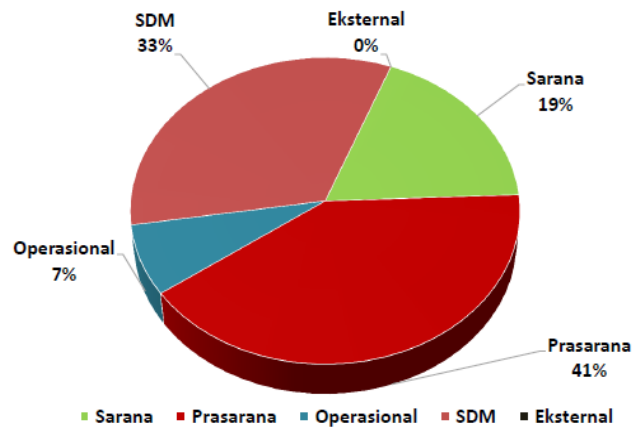
Berikut ini adalah hasil kegiatan analisis dan perancangan penelitian berdasarkan metodologi yang sudah dideskripsikan di bab 3.

IV.1 *Business Understanding*

Pada tahapan ini penelitian dilakukan dengan identifikasi permasalahan dan kebutuhan terkait perkereta apian Indonesia, khususnya dalam bidang keamanan. melalui studi pustaka. Sumber utama yang digunakan dalam tahap ini berupa laporan media *release* dari Komite Nasional Keselamatan Transportasi (KNKT) Indonesia pada tahun 2016 yang berjudul Data Investigasi Kecelakaan Perkeretaapian tahun 2010 – 2016. Laporan ini meliputi *overview* hasil investigasi kecelakaan dalam kereta api di Jawa dan Sumatera serta deskripsi secara lebih rinci mengenai *highlight* kecelakaan pada periode tersebut sebagai contoh. Sumber-sumber lainnya yang mendukung dalam kegiatan ini adalah berita-berita dari portal berita *online*.

Pada tahun 2010-2016, terjadi 35 kecelakaan perkeretaapian di Indonesia yang diinvestigasi oleh KNKT. Jenis kecelakaan yang paling banyak terjadi adalah berjenis anjlok/ terguling dengan jumlah sebanyak 24 dari 35 kejadian. Sembilan kejadian lainnya merupakan kecelakaan dengan jenis tumbukan antar kereta api, sementara dua kecelakaan lainnya dikategorikan sebagai jenis lain-lain.

Berdasarkan hasil investigasi KNKT secara umum, faktor penyebab utama kecelakaan tersebut sebagian besar didominasi oleh faktor prasarana yang mengambil persentase sebanyak 41 % dari seluruh kejadian. Faktor penyebab utama kecelakaan terbesar selanjutnya ditempati oleh sumber daya manusia sebanyak 33 %. Sementara itu, faktor – faktor lainnya seperti sarana, operasional, dan eksternal tidak memiliki persentase yang cukup besar dengan detail berupa 19 %, 7 %, dan 0 % untuk masing-masing faktor. Dengan mempertimbangan bahwa faktor prasarana memiliki persentase terbesar sebagai penyebab utama kecelakaan kereta api, faktor prasarana dipilih sebagai lingkup dari masalah yang akan diselesaikan dalam penelitian.



Gambar 13 Faktor penyebab kecelakaan kereta api (KNKT, 2016)

Dalam laporan secara lebih medetail terkait *highlight* investigasi kecelakaan perkeretaapian tahun 2016, terdapat rincian mengenai faktor penyebab kecelakaan secara lebih deskriptif dan mendetail. Kecelakaan yang menjadi *highlight* pada periode tersebut adalah kecelakaan berjenis anjlokan yang terjadi pada KA 3008 di KM 262+100/200 dalam perjalanan dari St. Lubukrukam menuju St. Peninjawan, Tanjung karang pada 1 Maret 2016. Hasil investigasi menunjukkan bahwa penyebab utama dari kecelakaan tersebut adalah faktor prasarana berupa terjadinya patah (patah total dan patah gompal) dengan penyebab sebagai berikut (KNKT, 2016):

1. Pelubangan baut pelat sambung yang tidak sesuai sehingga terbentuk awal retakan (*crack inititation*) pada tepi lubang kasar pada *web rail*, penjaralan retakan (*crack propagation*) hingga patah akhir (*total disintegration*)
2. Benturan yang berulang antara plat sambung dan kepala rel selama dilewati KA. Benturan ini disebabkan oleh penggunaan jumlah baut yang tidak sesuai dengan persyaratan teknis.

Penemuan yang didapatkan dari investigasi kecelakaan KA 3008 dalam bidang prasarana secara lengkap adalah sebagai berikut (KNKT,2016) :

1. Ditemukan adanya kepala rel gompal.
2. Adanya rel patah pada sambungan rel di lokasi kejadian.
3. Adanya porositas di sambungan rel pada daerah lasan (pengelasan tidak sempurna). Hal ini disebabkan oleh metode pengelasan yang tidak sesuai prosedur dan penggunaan material yang disyaratkan.

4. Permukaan lubang baut tidak sempurna (terdapat sudut tajam pada bentuk lubang).
5. Jumlah baut di pelat sambung tidak lengkap (3 baut yang terpasang, berdasarkan spek seharusnya 6 baut).
6. Kondisi jalan rel disekitar lokasi kejadian kurang batu ballast.
7. Batalan beton pecah dan masih digunakan.
8. Secara umum kerusakan jalur KA di Divre IV tanjungkarang pada tahun 2016 adalah sebagai berikut:
 - a. Rel yang mengalami kerusakan sebanyak 54% dari total keseluruhan panjang jalur KA.
 - b. Program perawatan yang berjalan hanya dapat mengcover 22% dari total keseluruhan jalur KA yang rusak.
 - c. Masih terdapat backlog yang tersisa sebanyak 42% dari total keseluruhan panjang jalur KA.

Hasil investigasi ini menunjukkan bahwa patah rel merupakan salah satu faktor prasarana yang mampu menyebabkan terjadinya kecelakaan kereta api yang cukup berbahaya hingga menarik perhatian untuk menjadi *highlight* kecelakaan pada tahun 2016. Dengan mempertimbangkan hal tersebut, dapat disimpulkan bahwa perawatan rel dengan kualitas yang baik dan benar dapat berperan dalam mengurangi kecelakaan kereta api yang berbahaya. Penemuan dalam bidang prasarana kemudian membawa investigasi ke beberapa penemuan dalam bidang organisasi dan manajemen yang terkait dengan manajemen perawatan rel sebagai berikut (KNKT, 2016) :

1. Belum disusunnya standar kerusakan dan prosedur inspeksi jalan rel sebagai dasar untuk menentukan kondisi dan klasifikasi kerusakan jalan rel yang menjadi acuan dalam menentukan risiko keselamatan dan prioritas perawatan.
2. Belum dilaksanakannya uji berkala dari jalur kereta api di wilayah III.2.10 Resort Peninjawan, Sub Divre III.2 (Divre IV) Tanjungkarang.

Inspeksi rel di Indonesia saat ini masih dilakukan secara manual oleh petugas dengan berjalan kaki dari satu stasiun ke stasiun lain. Penemuan patah rel selanjutnya dilaporkan kepada pihak berwenang untuk ditindaklanjuti (Nugroho, 2019). Tidak adanya standar kerusakan dan prosedur inspeksi membuat analisis kerusakan sangat

bergantung kepada pemahaman pelaku inspeksi sendiri. Dengan mempertimbangkan mekanisme inspeksi yang digunakan, hal ini menimbulkan kemungkinan besarnya pengaruh *human error* yang tidak bisa diidentifikasi dalam proses inspeksi. Mekanisme inspeksi yang sangat bergantung kepada sumber daya manusia juga memiliki celah untuk tidak terlaksananya uji berkala pada jalur kereta yang ditimbulkan dari *human error*.

Penelitian ini mengajukan solusi berupa otomasi proses penilai patah rel untuk meminimalisir keterlibatan manusia dalam proses tersebut. Hal ini diperlukan agar proses inspeksi yang sangat memengaruhi kondisi rel dapat dipastikan untuk selalu dilakukan dengan berdasarkan standar yang seragam.

Sistem otomasi yang diajukan dalam penelitian ini adalah berupa sistem teknologi informasi dalam bentuk aplikasi berbasis *machine learning*. Penggunaan algoritma *machine learning* dapat digunakan untuk membangun model dalam menentukan penilaian kondisi rel berdasarkan data hasil perawatan rel yang telah dilakukan di masa lampau. Penerapan algoritma *machine learning* dalam aplikasi juga memungkinkan dilakukannya pembaharuan model yang digunakan dalam prediksi seiring bertambahnya data kejadian patah rel sejak sistem digunakan. Agar perawatan preventif dapat dilakukan sehingga mengurangi kemungkinan terjadinya kecelakaan, keluaran dari sistem ini akan dibuat dalam bentuk prediksi kejadian dan lokasi patah rel. Hasil prediksi ini nantinya bisa digunakan untuk melakukan rencana perawatan dengan menggunakan sumber daya yang ada secara lebih efektif dan efisien.

Berdasarkan analisis yang telah disimpulkan bahwa tujuan dari penelitian atau proses *data mining* yang dilakukan adalah berupa pengadaan sistem untuk mengidentifikasi area rel yang memiliki kemungkinan besar menyebabkan kejadian rel patah berdasarkan rekaman data kejadian terkait yang sudah terjadi sebelumnya dan faktor-faktor lain yang memiliki kemungkinan memengaruhi hal tersebut.

IV.2 Data Understanding

Dengan mempertimbangkan keluaran dari hasil akhir penelitian yang berupa nilai prediksi kejadian dan lokasi patah rel, maka data yang akan digunakan adalah berupa data kerusakan yang disertai dengan data teknis dan operasional terkait. Hal ini dilakukan juga berdasarkan pertimbangan terkait penelitian sebelumnya yang berjudul

Multivariate Statistical Model for Predicting Occurrence and Location of Broken Rails yang menggunakan data tersebut untuk membuat model statistic prediksi kejadian dan lokasi patah rel dengan tingkat akurasi cukup tinggi pada masanya, yakni 87,4%. Penyusunan dataset juga dilakukan berdasarkan hasil wawancara dengan perwakilan bagian pengelolaan prasarana PT. Kereta Api Indonesia yang memberikan hasil pengamatan terkait kejadian patah rel di Indonesia. Dataset penelitian yang akan digunakan dalam penelitian ini adalah data kejadian rel patah/gompal dan tidak patah/gompal di daerah operasi kereta api DIVRE III dan DIVRE IV pada tahun 2017 – 2019 dengan atribut berupa data kejadian dan teknis operasional sebagai berikut :

- Tanggal (hari, bulan, dan tahun)
- Lokasi (titik kilometer dihitung dari titik nol yang ditentukan)
- Kecepatan maksimal (km/jam)
- *Average tons per car* (satuan ton)
- *Average dynamic tons per car* (satuan ton)
- *Annual gross tonnage* (satuan juta ton)
- *Annual wheel passes* (satuan juta)

IV.3 Data Preparation

Pada penelitian ini terdapat beberapa proses yang perlu dilakukan untuk mempersiapkan data. Proses pertama yang perlu dilakukan adalah pengubahan seluruh data bertipe string yang akan digunakan dalam pemodelan menjadi data yang bisa diproses seperti integer atau real. Proses selanjutnya adalah pembuatan basis data untuk menampung data yang akan digunakan dalam proses pemodelan. Dengan mempertimbangkan bahwa data diterima dari PT. Kereta Api Indonesia dalam bentuk yang terpisah-pisah antara atributnya, maka basis data yang dibuat harus menyediakan tabel yang menggabungkan seluruh atribut yang dibutuhkan untuk proses pemodelan. Proses selanjutnya adalah penanganan data yang kosong dan diperlukan dalam pemodelan. Hal ini dilakukan dengan mengisi sel kosong tersebut dengan nilai rata-rata kolom sel terkait.

IV.4 Modeling

Pemodelan dalam penelitian ini akan dilakukan menggunakan berbagai macam teknik. Selanjutnya, seluruh hasil pemodelan akan dievaluasi menggunakan metode

yang telah ditetapkan. Hasil evaluasi tersebut kemudian akan dibandingkan antara satu sama lain untuk menentukan model terbaik yang digunakan di sistem yang akan digunakan di kehidupan nyata. Berikut adalah teknik-teknik yang akan digunakan dalam pemodelan :

1. Logistic regression
2. *Support Vector Machine*
3. *Decision Tree*
4. Random Forest
5. K-Nearest Neighbor

IV.5 Evaluation

Sebelum pemodelan dilakukan, dataset yang ada akan dibagi dua dengan persentase sebanyak 80% untuk *training* dan 20% untuk pengujian. Setelah pemodelan dilakukan, setiap model yang dihasilkan diuji akurasi menggunakan nilai *accuracy* dan data *training* yang telah dipisahkan. Model untuk *deployment* kemudian ditentukan dengan memilih model yang memiliki nilai *accuracy* terbesar dalam hasil pengujian.

IV.6 Deployment

Deployment dalam penelitian ini akan dilakukan dalam bentuk aplikasi sederhana dengan *user interface* untuk menggunakan model prediksi kejadian dan lokasi patah rel yang akan dibuat. Berikut adalah rancangan sistem yang akan digunakan dalam fase *deployment* penelitian ini.

IV.6.1 Rancangan Aplikasi Prediksi Kejadian dan Lokasi Patah rel Kereta Api

IV.6.1.1 Deskripsi Umum Aplikasi

Aplikasi prediksi kejadian dan lokasi patah rel kereta api adalah sistem yang dapat digunakan untuk menilai kemungkinan terjadinya patah rel di titik rel tertentu pada waktu tertentu. Hasil prediksi yang ditampilkan pada pengguna diberikan dalam bentuk informasi positif atau negative adanya kemungkinan kecelakaan.



Gambar 14 Gambaran umum sistem prediksi kejadian dan lokasi patah rel kereta api

Sistem mulai bekerja saat pengguna memberikan masukan berupa titik lokasi rel kereta api dalam bentuk kilometer dan daerah operasi atau divre, serta tahun ingin diprediksinya kondisi rel. Selanjutnya sistem mengumpulkan data-data terkait dengan masukan yang diberikan pengguna yang diperlukan untuk melakukan perhitungan dengan model hasil pelatihan. Sistem kemudian melakukan perhitungan menggunakan model *machine learning* dan data yang telah dikumpulkan dan menampilkan hasil prediksi pada *user interface*.

IV.6.1.2 Kebutuhan Fungsional

Berikut adalah kebutuhan fungsional yang perlu diperhatikan dalam pengembangan aplikasi prediksi kejadian dan lokasi patah rel kereta.

Tabel 2 Kebutuhan fungsional sistem prediksi kejadian dan lokasi patah rel kereta api

ID	Kebutuhan	Penjelasan
FR-01	Sistem dapat melakukan pelatihan model prediksi	Penanaman algoritma pelatihan model prediksi dalam sistem
FR-02	Sistem dapat digunakan pengguna untuk mengetahui kemungkinan terjadinya patah rel pada lokasi dan waktu tertentu	Pengguna dapat memberikan masukan kepada sistem berupa titik rel (kilometre dan daerah operasi atau divre) dan tahun, kemudian menerima hasil berupa kemungkinan terjadinya patah rel berdasarkan masukan yang diberikan

BAB V

IMPLEMENTASI DAN EVALUASI

V.1 Data Preparation

Pada fase ini dilakukan persiapan data yang digunakan dalam penelitian. Fase ini dimulai dari pengambilan data dari pihak penyedia data hingga transformasi data ke bentuk yang diperlukan. Penyedia data yang digunakan dalam penelitian ini adalah bagian organisasi yang dipimpin oleh direktur pengelolaan prasarana di PT. Kereta Api Indonesia.

Data yang digunakan dalam penelitian, tidak seluruhnya diminta dari pihak penyedia data. Hal ini disebabkan ada beberapa data yang bisa diturunkan dari data lainnya. Data yang diminta dari pihak penyedia data adalah data kejadian rel patah/gompal di daerah operasi kereta api DIVRE III dan DIVRE 1V pada tahun 2017 – 2019 dengan atribut berupa data kejadian dan teknis operasional sebagai berikut :

- Tanggal (hari, bulan, dan tahun)
- Lokasi (titik kilometer dihitung dari titik nol yang ditentukan)
- Kecepatan maksimal kereta(km/jam)
- *Average tons per car* (satuan ton)
- *Annual gross tonnage* (satuan juta ton)

V.1.1 Kondisi data awal

Kondisi data yang diberikan oleh pihak penyedia data pertama kali adalah saling terpisah-pisah antara satu sama lain. Berikut adalah betuk data murni yang didapatkan secara langsung dari pihak penyedia data :

- Satu dokumen berisi daftar kejadian rel patah di divre III dan divre IV pada periode tahun 2017 hingga 2019

	B	C	D	E	F	G	H	I	J	K	O
1	GANGGUAN JALAN REL DAN JEMBATAN BERDASARKAN LAPORAN OC PUSAT										
2											
3	Tanggal	Daop	QC	RESORT	Antara	Mulai	Selesai	Lama	Kategori Gangguan	Gangguan	ANDIL (menit)
14	8 Februari 2017	DIVRE IV	IVA TNK	IV.4 TGI	di KM 46+8/9	20:54	21:22	00:28	Rel patah & somplak	Jam 19:25 wib terima info dari ppka Rengas meneruskan laporan dari ppi, perhal rel putus di KM 46+8/9 petak jalan Rengas-Tegineseng, info dari ppi bisa dilewati 5 km/jam dan di	
15	8 Februari 2017	DIVRE IV	IVA TNK	IV.5 BKI	di KM 59+0/1	22:40	23:00	00:20	Rel patah & somplak	Jam 21:20 terima info dari ppka Hajipemangilan meneruskan laporan ppi, perhal rel patah di KM 59+0/1 petak jalan Hajipemangilan-Bekri, info ppi dipasang semb. 2B (20	
16	9 Februari 2017	DIVRE IV	IVA TNK	IV.5 BKI	di km 59+0/1	04:02	04:23	00:21	Rel patah & somplak	Hari : Rabu—Tgl : 08-02-2017 —jam : 01:58 wib jam 01:52 terima info dari ppka hip ardi meneruskan laporan ppi dedi w perhal rel gompel sepanjang 10 cm di km 59+0/1 pj hip-bki jam 03:00 terima info dari kares j khusus semb.3 dicabut bisa dilewati 5 kpi, diaga.	
17	10 Februari 2017	DIVRE IV	IVF TIR	IV.22 PNW	di km 271+1/2 petak jalan Pnw-Tlb	20:20	21:20	01:00	Rel patah & somplak	Terima info dari crew ka 3034A mass: Wahyu, Assmas: Dori Rel patah di km 271+1/2 petak jalan Pnw-Tlb. Jam 19:40 wib, laporan Kaur JJ : Andika,bisa dilewati 5 km/jam dan diaga	
18	10 Februari 2017	DIVRE IV	IVA TNK	IV.4 TGI	di km 47+3/4 jalur 2 rgs	22:20	23:20	01:00	Rel patah & somplak	Terima info dari ppka rengas sarwono meneruskan laporan ppi misrak perhal rel patah di km 47+3/4 jalur 2 rgs,untuk ka dilewatkan jalur 1(belok) jam 22:08 info dari mandor paryoto	
19	11 Februari 2017	DIVRE IV	IVB KB	IV.8 KB	di km 104+2/3 antara sta Cempaka Kotabumi (cep-kb)	03:30	03:45	00:15	Rel patah & somplak	Jam 02:47 perhal rel putus di km 104+2/3 antara sta Cempaka Kotabumi (cep-kb), dapat dilahi 5 kpi, diaga mulai pengerjaan 03:30 selesai jam 03:45, taspas dicabut ka dapat	
	12 Februari 2017	DIVRE IV	IVB KB	IV.9 CEP	di km 107+4/5 PJ Ktp-Cep	18:01	18:15	00:14	Rel patah & somplak	Hari : Sabtu—Tgl : 11-02-2017—jam : 15:50 wib Terima info dari crew ka 3037 (60) Masinis : Rudi Assmas : Apriono, di km 107+4/5 PJ Ktp-Cep indikasi rel gompel. Ka 3037 ber Ktp 15:25 wib dat Cep 15:50 wib. Jam 16:12 terima info ppka Cep, Ridwan meneruskan laporan Mandor Haryanto jam 16:10 wib di km 107+4/5 JL hilir PJ Ktp-Cep terdapat rel gompel 15 cm. Untuk sementara semb.3. jam 16:48 info kaur JJ Pepen bisa di lewati 5 kpi. Mulai pengerjaan : 18:01 wib (est 15")	
	<div> <div>...</div> <div>gangguan jj Februari 2017</div> <div>gangguan jj Maret 2017</div> <div>gangguan jj April 2017</div> <div>gangguan jj Mei 2017</div> <div>gangguan jj Juni 2017</div> </div>										

Gambar 15 Data murni kejadian rel patah

- Satu tabel berisi *annual tonnage* dan kecepatan maksimal di titik-titik pada divre III pada tahun 2017

	A	B	C	D	E	F	G	H	I	J	K	L
1	PASSING TONAGE											
2	WILAYAH DIVRE III PALEMBANG											
3												
4	NO.	KORIDOR LINTAS	JUMLAH KA/HR	PASSING TONAGE (JT TON/THN)	V. MAX (KM/JAM)	GOLONGAN UIC	KELAS JALN (PD.10)	REL		BANTALAN	PENAMBAT	KETERANGAN
5								Type	PANJANG/ PENDEK			
6												
7												
8	1	Kpt - Pbm	50	27,702	80	3	II	R.42	Panjang	Beton	Elastis	
9												
10	2	Pbm - X6	38	20,301	70	3	II	R.42	Panjang	Beton	Elastis	
11												
12	3	X.6 - NRU (Hilir)	38	44,677	60	4	II	R.54	Panjang	Beton	Elastis	
13												
14	4	X.6 - NRU (Hulu)	39	129,083	60	1	I	R.54	Panjang	Beton	Elastis	
15												
16	5	NRU - ME (Hilir)	39	46,598	60	1	I	R.54	Panjang	Beton	Elastis	
17												

Gambar 16 Data murni *passing tonnage* DIVRE III 2017

- | | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | | | | | | | | |
|-----|-----------------------------------|---|---|---|---|---|---|---|---|---|---|---|----|---------|---------|---|---|---|---|---|----------------|---|------|---|---------|---|------|---|-------|
| 281 | PASSING TONAGE SUMSEL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 282 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 283 | 1. PASSING TONAGE LINTAS Tjh - X5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 284 | Rumus : | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 285 | TE | = | $Tp + (Kb \times Tb) + (K1 \times T1)$ | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 286 | T | = | $360 \times S \times TE$ | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 287 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 288 | T | = | Daya angkut lintas (ton/hr) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 289 | TE | = | Tonage ekvalen (ton/hari) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 290 | Tp | = | Tonage penumpang dan kereta harian. | | | | | | | | | | Tp | = | 1.436 | | | | | | | | | | | | | | |
| 291 | Tb | = | Tonage barang dan gerbong per hari. | | | | | | | | | | Kb | = | 1.50 | | | | | | | | | | | | | | |
| 292 | S | = | Koefesien yang besarnya tergantung kualitas lintas: | | | | | | | | | | Tb | = | 104.924 | | | | | | | | | | | | | | |
| 293 | | | 1,10 untuk lintas dgn ka prnp, V maks = 120 Km/Jam. | | | | | | | | | | K1 | = | 1.40 | | | | | | | | | | | | | | |
| 294 | | | 1,00 untuk lintas tanpa ka prnp. | | | | | | | | | | T1 | = | 8.832 | | | | | | | | | | | | | | |
| 295 | Kb | = | Koefesien yang besarnya tergantung beban gandar. | | | | | | | | | | S | = | 1,10 | | | | | | | | | | | | | | |
| 296 | | | 1,5 untuk beban gandar ≤ 20 ton. | | | | | | | | | | V | = | 70 | | | | | | | | | | | | | | |
| 297 | | | 1,30 untuk beban gandar > 20 ton. | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 298 | T1 | = | Beralokomotif. | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 299 | K1 | = | Koefisien Lok 1.40 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 300 | V | = | Kecepatan Ka. (Km/Jam). | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 301 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 302 | | | | | | | | | | | | | TE | = | 1.436 | | | | | | | | | | | | | | |
| 303 | | | | | | | | | | | | | = | 171.187 | | | | | | | ton/hari | + | 1,50 | x | 104.924 | + | 1,40 | x | 8.832 |
| 304 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 305 | | | | | | | | | | | | | T | = | 360 | | | | | | | x | 1,10 | x | 171.187 | | | | |
| 306 | | | | | | | | | | | | | = | 67.790 | | | | | | | juta ton / thn | | | | | | | | |

- Satu tabel berisi *annual tonnage* dan kecepatan maksimal di titik-titik pada divre III pada tahun 2019

	A	B	C	D	E	F	G	H	I	J	K	L
1	PASSING TONAGE											
2	WILAYAH DIVRE III PALEMBANG											
3												
4												
5												
6												
	NO.	KORIDOR LINTAS	JUMLAH KA/HR	PASSING TONAGE (JT TON/THN)	V. MAX (KM/JAM)	GOLONGAN UIC	KELAS JALN (PD.10)	REL		BANTALAN	PENAMBAT	KETERANGAN
								Type	PANJANG/ PENDEK			
12	3	X.6 - NRU (Hilir)	38	44.677	60	4	II	R.54	Panjang	Beton	Elastis	
13	4	X.6 - NRU (Hulu)	39	129.083	60	1	I	R.54	Panjang	Beton	Elastis	
14	5	NRU - ME (Hilir)	39	46.598	60	1	I	R.54	Panjang	Beton	Elastis	
15	6	NRU - ME (Hulu)	38	128.158	60	4	II	R.54	Panjang	Beton	Elastis	
16	10	X5- X6	40	38.116	70	2	I	R.54	Panjang	Beton	Elastis	

V-3

- Satu tabel berisi perhitungan *annual tonnage* di titik- titik pada divre IV pada tahun 2019 yang mencakup kecepatan maksimal di titik terkait

1 Perhitungan Passing Tonnage Kooridor : PID-THN, GR-TNK Kecepatan 45 km/jam

$$TE = Tp + (Kb \times Tb) + (K1 \times T1)$$

$$T = 360 \times S \times TE$$

T = Daya angkut lintas / Passing tonnage (ton/thn)

TE = Tonase Ekuivalen (ton/hari)

Tp = Tonase penumpang dan kereta harian

Tb = Tonase barang dan gerbong per hari

S = Koefisien yang besarnya tergantung kualitas lintas :

1,10 untuk lintas dengan ka penumpang, Kecepatan (V) maksimum = 120 Km/Jam

1,00 untuk lintas tanpa ka penumpang

Kb. = Koefisien yang besarnya tergantung beban gandar

1,50 untuk lintas dengan beban gandar < 18 ton

1,30 untuk lintas dengan beban gandar > 18 ton

T1 =

Berat Lokomotif

V =

Koefisien Lok. 1,40

Kecepatan Ka. (Km/Jam)

Tp.	=	0	S	=	1	UIC 1	=	> 42		ton
Kb	=	1,5	V	=	45	UIC 2	=	29,75	s.d.	42
Tb.	=	150372				UIC 3	=	17,5	s.d.	29,75
K1	=	1,4				UIC 4	=	9,8	s.d.	17,5
T1	=	12264				UIC 5	=	4,9	s.d.	9,8
						UIC 6	=	2,45	s.d.	4,9
						UIC 7	=	1,225	s.d.	2,45
						UIC 8	=	0,525	s.d.	21,5
						UIC 9	=	<	0,525	ton

$$TE = 0 + 1,5 \times 150372 + 1,4 \times 12264 = 242.727,60 \text{ ton/hr}$$

$$T = 360 \times 1 \times 242.727,60 = 87,382 \text{ juta ton/thn}$$

$$\text{GOL UIC} = 1 \quad (T > 42 \text{ juta ton/thn})$$

$$\text{KLS JALAN PD 10} = 1 \quad (T > 20 \text{ juta ton/thn})$$

Frekuensi pemecokan (F) km/tahun :

$$F = \frac{0,023 \times T^{*03} \times V_{maks}^{*0.5} \times (1 + Fp)}{0,023 \times 3,823 \times 6,7082 \times 2,0625} = 1,22 \text{ kali / tahun} = 10 \text{ bulan sekali}$$

Gambar 19 Data murni *passing tonnage* DIVRE IV 2019

- Satu tabel berisi data gerbong yang melewati titik-titik pada divre III dan IV pada tahun 2017

No	No KA	Nama KA	lintas	Rangkaian										Berat (Ton)										Jumlah					
				K1	K2	KMP2	MP2	K3	KP3	KMP3	B	GB	GT	GK	GD	Lok	K1	K2	KMP2	MP2	K3	KP3	KMP3		B	GB	GT	GK	GD
807																													
808	2	S.2	Limak Srawaja 2	Kgr-Trk	2			5	1							8	84	84,0	-	-	-	190,0	-	36	-				384,00
811	3	S.14	Express Rajabasa 2	Trk-Kgr	3			5	-	1						10	84	85,0	-	-	-	216,0	-	36	-				462,00
812	9	3994	PULP	Thn-Niu							22					22	84	-	-	-	-	-	-	-	330				414,00
813	10	3990	PULP	Thn-Niu							22					22	84	-	-	-	-	-	-	-	330				414,00
814	11	3996F	PULP	Thn-Niu							22					22	84	-	-	-	-	-	-	-	330				414,00
815	12	3995F	PULP	Thn-Niu							22					22	84	-	-	-	-	-	-	-	330				414,00
816	13	3880	Semen	Kgr-Tjh								20				20	84	-	-	-	-	-	-	-				640	524,00
817	14	3887/3995	Semen	Tjh-Lig								20				20	84	-	-	-	-	-	-	-				600	684,00
818	38	3002	BER 24 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00
819	39	3004	BER 25 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00
820	40	3006	BER 26 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00
821	41	3008	BER 27 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00
822	42	3010	BER 28 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00
823	43	3012	BER 29 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00
824	44	3014	BER 30 kosong	Thn-Pbr Xb-Tmb							60					60	252	-	-	-	-	-	-	-	1620				1872,00

Gambar 20 Data murni gerbong DIVRE III dan IV 2017

- Satu tabel berisi data gerbong yang melewati titik-titik pada divre III dan IV pada tahun 2019

STAMPORMASI KA TAHUN 2019																											
DIVRE IV TANJUNG KARANG																											
(1.1) KORIDOR : PID-THN, GR-TNK KECEPATAN 45 KM/JAM																											
No	No KA	Nama KA	lintas	Rangkaian										Berat (Ton)										Jumlah			
				K1	K2	KMP2	MP2	K3	KP3	KMP3	B	GB	GT	GR	GD	Loak	K1	K2	KMP2	MP2	K3	KP3	KMP3		B	GB	GT
7	1	3994-3991	PULP	Thn-Niu							22				22	84	-	-	-	-	-	-	-	-	1.393	1.437,00	
8	2	3990-3995	PULP	Thn-Niu							22				22	84	-	-	-	-	-	-	-	-	1.393	1.437,00	
9	3	3996F-3993F	PULP	Thn-Niu							22				22	84	-	-	-	-	-	-	-	-	1.393	1.437,00	
10	4	3995F-3990F	PULP	Thn-Niu							22				22	84	-	-	-	-	-	-	-	-	1.393	1.437,00	
11	5	3888	Semen	Tjh-Sin								20			20	84	-	-	-	-	-	-	-	-	300	384,00	
12	6	3889	Semen	Sin-Tjh								20			20	84	-	-	-	-	-	-	-	-	1.160	1.224,00	
13	7	L3888	Dinas Langer	Thn-Sin											84	-	-	-	-	-	-	-	-	-	84,00		
14	8	L3889	Dinas Langer	Sin-Thn											84	-	-	-	-	-	-	-	-	-	84,00		
15	9	3002-3001	BER 1 is	Tmb-Pbr Xb-Thn							60				60	216	-	-	-	-	-	-	-	-	4.620	4.636,00	
16	10	3004-3003	BER 2 is	Tmb-Pbr Xb-Thn							60				60	216	-	-	-	-	-	-	-	-	4.620	4.636,00	

Gambar 21 Data murni gerbong DIVRE III dan IV 2019

V.1.2 Kondisi data akhir dan transformasi data

Kondisi data akhir atau data yang digunakan dalam pemodelan di penelitian ini berbentuk sebuah tabel dengan setiap barisnya mendeskripsikan satu kejadian rel patah atau gompal. Berikut adalah bentuk data yang sudah siap dipakai dalam proses pemodelan.

tahun	bulan	tangga	lokasi	ann_ton	vmax	br_cart	br_dinam	wheel	acc
2017	12	1	25,7669424	40,40513928	70	54,84245221	93,23216875	2,946997273	0
2017	12	2	137,45	63,96424848	60	54,823219	87,7171504	4,666945842	1
2017	12	2	237,1647578	67,79000448	70	55,8328877	94,91590909	4,856636099	0
2017	12	2	365,7348472	128,158	60	61,92265268	99,07624429	8,27858591	0
2017	12	3	294,7	67,79000448	70	55,8328877	94,91590909	4,856636099	1
2017	12	3	108,65	63,96424848	60	54,823219	87,7171504	4,666945842	1
2018	12	3	368,25	46,698	60	21,58761733	34,54018773	8,652738149	1
2017	12	3	320,4656606	38,116	70	45,387	77,1579	3,359199771	0
2019	12	3	294,65	98,4680928	75	55,8328877	97,70755348	7,054486834	1
2019	12	3	138,0384686	71,2532888	60	77,15762604	123,4532017	3,603017065	0

Gambar 22 Data final pemodelan

Kondisi data akhir tersebut diperoleh dengan melakukan proses transformasi dari data awal. Dalam proses transformasi tersebut data kejadian rel tidak patah dan gompal dari tahun 2017 hingga 2019 di divre III dan IV diperoleh dengan melakukan ekstraksi titik-titik lokasi antara titik awal daerah observasi (0 km) dan titik akhir daerah observasi dengan nilai yang tidak sama dengan nilai lokasi yang sudah ada pada data kejadian. Proses transformasi juga meliputi penentuan nilai-nilai atribut yang mendeskripsikan setiap data kejadian. Berikut adalah proses transformasi yang dilakukan untuk menentukan nilai-nilai atribut dari data setiap kejadian.

Tabel 3 Transformasi data

Atribut	Tipe data	Simbol di tabel	Proses transformasi
Tahun	Integer	Tahun	Mengambil dari data tanggal pada dokumen kejadian
Bulan	Integer	Bulan	Mengambil dari data tanggal pada dokumen kejadian
Hari	Integer	Tanggal	Mengambil dari data tanggal pada dokumen kejadian
Lokasi	Float	Lokasi	Mengambil data kilometer dan mengubahnya menjadi data berjenis float dari kolom “Antara” di tabel pada dokumen kejadian yang jenisnya masih berupa string dan tercampur dengan karakter bukan

Atribut	Tipe data	Simbol di tabel	Proses transformasi
			angka
<i>Annual Gross Tonnage</i>	Float	ann_ton	Mengaitkan data tahun dan lokasi kejadian dengan data lokasi di tabel <i>passing tonnage</i> pada tahun terkait untuk memperoleh <i>passing tonnage</i> di lokasi terkait
Kecepatan maksimal	Float	vmax	Mengaitkan data tahun dan lokasi kejadian dengan data lokasi di tabel <i>passing tonnage</i> pada tahun terkait untuk memperoleh kecepatan maksimal di lokasi terkait
<i>Average tons per car</i> (br_cart)	Float	br_cart	Mengaitkan data tahun dan lokasi kejadian dengan data lokasi di tabel <i>passing tonnage</i> pada tahun terkait untuk memperoleh kecepatan maksimal di lokasi terkait.
<i>Average dynamic tons per car</i>	Float	br_dinamis	Melakukan perhitungan berdasarkan rumus : $A = T(1+V/100)$ dengan legenda sebagai berikut A : <i>Average dynamic tons per car</i> T : <i>Average tons per car</i> V : kecepatan maksimal kereta
<i>Annual wheel passes</i>	Float	Wheel	Melakukan perhitungan berdasarkan rumus : $W = 4T/L$ dengan legenda sebagai berikut W : <i>Annual number of wheel passes</i> T : <i>Annual gross tonnage</i> L : <i>Average tons per car</i>
Status rel patah dalam titik yang dideskripsikan (0	Integer	Acc	Membuat kolom sendiri dan memberikan nilai 1 untuk data kejadian rel patah dan nilai 0 untuk

Atribut	Tipe data	Simbol di tabel	Proses transformasi
= tidak terjadi; 1 = ada kejadian rel patah)			kejadian rel tidak patah

V.2 Modeling

Pemodelan dilakukan menggunakan google colab dengan Bahasa pemrograman python. Modul utama yang digunakan dalam pemrograman di fase pemodelan ini adalah scikit-learn. Scikit-learn adalah modul yang memungkinkan pemrograman *machine learning* berbasis python dan didistribusikan di bawah lisensi 3-Clause BSD (Mueller, 2020). Modul ini dimasukkan dalam program di bagian awal atau bagian inisiasi. Pada bagian inisiasi ini, *file* yang akan digunakan dalam pemodelan juga dimuat ke dalam memori yang digunakan saat eksekusi aplikasi.



```

[ ] import warnings
import numpy as np
warnings.simplefilter(action='ignore', category=FutureWarning)

# Pemrosesan Awal
import pandas as pd
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation

# Import data yang digunakan
df = pd.read_excel("02_kombinasi.xlsx")

```

Gambar 23 Program pemodelan bagian inisiasi

V.2.1 Data analysis and pre processing

V.2.1.1 Data analysis and pre processing tanpa penindaklanjutan terhadap data outlier

Proses selanjutnya adalah analisis dan *pre-processing* data secara lebih lanjut. Kegiatan yang dilakukan dalam proses ini meliputi peninjauan data yang telah dimuat dengan mengecek *header* data, mengecek jenis data setiap atribut, mengecek *missing value* dan menanganinya, serta *feature selection*.

□ ▾ Data analysis and pre processing

```
[ ] import math
import matplotlib.pyplot as plt
import seaborn as sns

[ ] # Cek data yang digunakan
df.head(3)

[ ] # Melihat kondisi data yang digunakan
df.describe()

[ ] # Jumlah data yang digunakan
df.shape

[ ] df.info()
```

Gambar 24 Program pemodelan bagian data analysis and *pre processing* (analisis data)

Penanganan *missing value* dalam penelitian ini dilakukan dengan mengisi sel kosong menggunakan rata-rata dari kolom sel kosong terkait.

```

<>
#identifikasi missing value

total = df.isnull().sum().sort_values(ascending=False)
percent = (df.isnull().sum()/df.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
f, ax = plt.subplots(figsize=(15, 6))
plt.xticks(rotation='90')
sns.barplot(x=missing_data.index, y=missing_data['Percent'])
plt.xlabel('Features', fontsize=15)
plt.ylabel('Percent of missing values', fontsize=15)
plt.title('Percent missing data by feature', fontsize=15)
missing_data.head()

[ ] ##mengisi data menggunakan rata-rata kolom
df.wheel.fillna(df.wheel.mean(),inplace=True)
df.br_dinamis.fillna(df.br_dinamis.mean(),inplace=True)
df.br_cart.fillna(df.br_cart.mean(),inplace=True)
df.vmax.fillna(df.vmax.mean(),inplace=True)
df.ann_ton.fillna(df.ann_ton.mean(),inplace=True)
df.lokasi.fillna(df.lokasi.mean(),inplace=True)

```

Gambar 25 Program pemodelan bagian *data analysis and pre processing* (penanganan *missing value*)

Feature selection dilakukan dengan menggunakan *univariate feature selection* dan *correlation matrix*. *Univariate feature selection* bekerja dengan memilih fitur-fitur terbaik berdasarkan *univariate statistical test* (Scikit-learn, n.d.). Dalam penelitian ini, metode tersebut digunakan untuk melihat skor masing-masing fitur berdasarkan *univariate statistical model*.

```

[49] from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

X = df.iloc[:,0:9] #independent columns
y = df.iloc[:, -1] #target column i.e price range
#apply SelectKBest class to extract top 10 best features
bestfeatures = SelectKBest(score_func=chi2, k=9)
fit = bestfeatures.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs', 'Score'] #naming the dataframe columns
print(featureScores.nlargest(10,'Score')) #print 10 best features

```

Gambar 26 Program pemodelan bagian *data analysis and pre processing* (penilaian fitur menggunakan *univariate statistical model*)

Hasil penilaian fitur data menggunakan *univariate statistical model* menunjukkan bahwa lokasi merupakan fitur yang paling berpengaruh terhadap terjadinya patah rel dengan dengan skor bernilai 806,309. Fitur yang berpengaruh selanjutnya adalah 'br_dinamis' dengan skor yang menurun drastis dari fitur sebelumnya, yakni 139,47. Fitur 'br_cart' berada di urutan selanjutnya dengan skor bernilai kurang lebih setengah dari fitur sebelumnya, yakni 76,18. Skor pada fitur selanjutnya kembali menurun dengan drastis yang ditunjukkan pada fitur wheel dengan skor 9,96. Setelah penurunan drastis tersebut, fitur-fitur selanjutnya memiliki skor yang memiliki selisih tidak jauh antara satu sama lain.

	Specs	Score
3	lokasi	806.390478
7	br_dinamis	139.475476
6	br_cart	76.180302
8	wheel	9.968470
1	bulan	5.875225
5	vmax	3.545456
4	ann_ton	2.713369
2	tanggal	0.873816
0	tahun	0.001662

Gambar 27 Program pemodelan bagian data analysis and pre processing (hasil penilaian fitur menggunakan *univariate statistical model*)

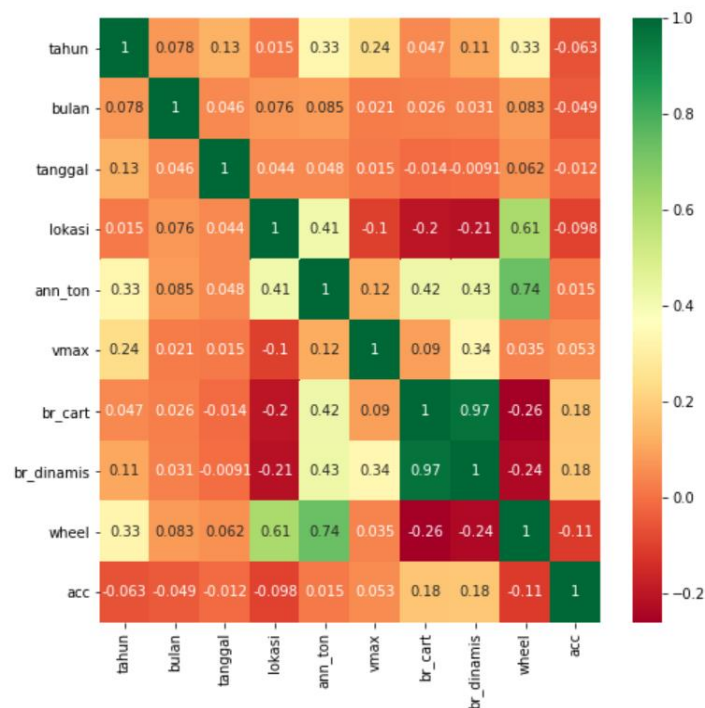
Correlation matrix digunakan untuk melihat keterkaitan antara fitur dalam data.

Matriks yang digunakan dalam penelitian ini adalah *heatmap*.

```
[51] # Melihat hubungan antar data
X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,1] #target column i.e price range
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(8,8))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

Gambar 28 Program pemodelan bagian data analysis and pre processing (*heatmap*)

Hasil penggunaan *heatmap* menunjukkan bahwa hanya beberapa fitur yang berpengaruh terhadap kemungkinan terjadinya patah rel di suatu titik lokasi. Hal ini ditunjukkan dengan adanya sebagian fitur yang memiliki nilai korelasi negatif dengan fitur 'acc' yang melambangkan kemungkinan rel patah. Fitur-fitur yang memiliki nilai korelasi positif dengan fitur 'acc' meliputi *annual tonnage* ('ann_ton'), kecepatan maksimal ('vmax'), berat rata-rata gerbong ('br_cart'), dan berat dinamis rata-rata gerbong ('br_dinamis').



Gambar 29 Program pemodelan bagian data analysis and pre processing (hasil penggunaan *heatmap*)

Hasil penilaian fitur data menggunakan *univariate statistical model* dan *heatmap* selanjutnya digunakan untuk menentukan kombinasi fitur yang akan digunakan dalam pemodelan. Hal ini dilakukan dengan mengetes beberapa kombinasi fitur pada model pemodelan dasar. Berdasarkan percobaan tersebut dihasilkan kesimpulan bahwa fitur-fitur yang paling berpengaruh terhadap kemungkinan patah rel berdasarkan data set yang ada tanpa penindaklanjutan *outlier* meliputi lokasi, *annual tonnage* ('ann_ton'), kecepatan maksimal ('vmax'), berat rata-rata gerbong ('br_cart'), dan berat dinamis rata-rata gerbong ('br_dinamis').

```
[52] #Loading data yang telah di pre-process ke variabel untuk diolah berdasarkan heatmap
#X=df[['tahun','bulan','tanggal','lokasi','ann_ton','vmax','br_cart','br_dinamis','wheel']].values
X=df[['lokasi','ann_ton','vmax','br_cart','br_dinamis']].values
Y=df['acc'].values

# Split dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=1)
```

Gambar 30 Program pemodelan bagian *data analysis and pre processing* (penyusunan data final untuk pembuatan model)

V.2.1.2 *Data analysis and pre processing* dengan penindaklanjutan terhadap data *outlier*

Penindaklanjutan data *outlier* dapat memberikan hasil yang berbeda pada hasil pemodelan data dibandingkan dengan model tanpa penindaklanjutan *outlier*. *Outlier* adalah hasil observasi yang karakternya berbeda jauh dari hasil observasi lain dalam *sample* sebuah populasi (U.S Department of Commerce, n.d.). Identifikasi *outlier* dalam penelitian ini dilakukan berdasarkan mekanisme dalam teknik *box plot*.

Box plot adalah teknik yang menghasilkan penampakan grafis untuk mendeskripsikan karakter data di tengah serta akhir distribusi. *Box plot* dibuat menggunakan median, quartil bawah (*25th percentile*), dan quartil atas (*75th percentile*). Jika quartil bawah adalah Q1 dan quartil atas adalah Q3, maka selisih antara keduanya (Q3-Q1) disebut *interquartile range* atau IQ (U.S Department of Commerce, n.d.).

Box plot menggunakan quartil atas dan quartil bawah untuk mendeteksi *outlier*. Batasan nilai untuk menentukan *outlier* dalam *box plot* disebut *fence*. Berikut adalah formula yang digunakan dalam teknik *box plot* untuk menentukan nilai *fence* (U.S Department of Commerce, n.d.) :

1. *lower inner fence*: $Q1 - 1.5 \cdot IQ$
2. *upper inner fence*: $Q3 + 1.5 \cdot IQ$
3. *lower outer fence*: $Q1 - 3 \cdot IQ$
4. *upper outer fence*: $Q3 + 3 \cdot IQ$

Nilai yang berada di luar *range* antara *lower inner fence* dan *upper inner fence* dikategorikan sebagai *mild outlier*. Nilai yang berada di luar *range* antara *lower outer fence* dan *upper outer fence* dikategorikan sebagai *extreme outlier* (U.S Department of Commerce, n.d.). Batas yang digunakan dalam penelitian ini adalah *mild outlier*.

Berikut adalah program yang digunakan untuk menentukan quartil atas dan quartil bawah data serta eliminasi data yang dikategorikan sebagai *outlier*.

```
[ ] Q1 = df.quantile(0.25)
    Q3 = df.quantile(0.75)
    IQR = Q3 - Q1
    print(IQR)
```

```
tahun      2.000000
bulan      6.000000
tanggal    15.000000
lokasi     190.229564
ann_ton    24.555389
vmax       10.000000
br_cart    1.009669
br_dinamis 7.198759
wheel      3.353114
acc        1.000000
dtype: float64
```

```
[ ] df = df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))),any(axis=1)]
```

Gambar 31 Program pemodelan bagian *data analysis and pre processing* (eliminasi baris data yang berisi *outlier*)

Jumlah data yang tersisa setelah eliminasi *outlier* adalah sebesar 987 baris dan 10 kolom. Hal ini menunjukkan pengurangan yang cukup besar dari data semula yang memiliki ukuran sebesar 1269 baris dan 10 kolom. Jumlah data yang tereliminasi dari data semula adalah sebanyak 282 baris. Hal ini menunjukkan bahwa jumlah baris data yang memiliki *outlier* berjumlah sebanyak 282 baris.

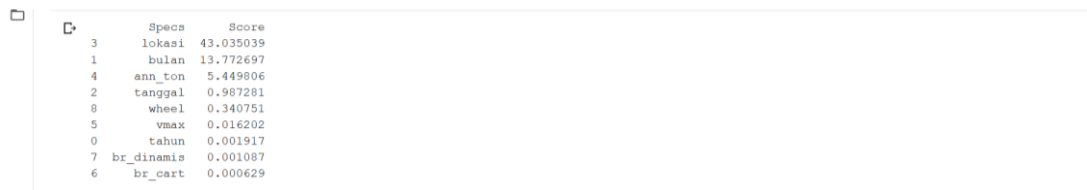
```
[ ] df.shape
```

```
(987, 10)
```

Gambar 32 Program pemodelan bagian *data analysis and pre processing* (hasil eliminasi baris data yang berisi *outlier*)

Perubahan susunan data akibat eliminasi *outlier* menyebabkan adanya perbedaan hasil penilaian fitur menggunakan *univariate feature selection* dan *correlation matrix* dari penilaian yang telah dilakukan sebelumnya. Hasil penilaian fitur menggunakan *univariate feature selection* setelah eliminasi *outlier* menunjukkan perbedaan skor antar fitur yang tidak sebesar pada penilaian sebelumnya. Susunan urutan fitur dengan skor tertinggi juga berbeda dari hasil penilaian sebelumnya. Penilaian fitur pada *heatmap* juga menunjukkan komposisi fitur dengan nilai korelasi positif yang berbeda dari hasil penilaian sebelumnya.

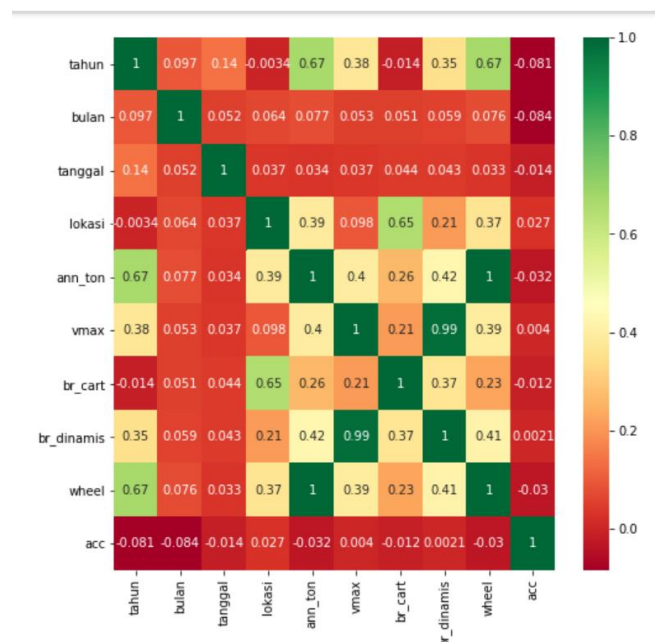
Hasil penilaian fitur data menggunakan *univariate statistical model* menunjukkan bahwa lokasi merupakan fitur yang paling berpengaruh terhadap terjadinya patah rel dengan skor bernilai 43,035. Fitur yang berpengaruh selanjutnya adalah ‘bulan’ dengan skor 13,77. Fitur *annual tonnage* (ann_ton) berada di urutan selanjutnya dengan skor 5,45. Fitur di urutan selanjutnya adalah tanggal dengan skor 0,98. Fitur-fitur selanjutnya memiliki skor yang memiliki selisih tidak jauh antara satu sama lain, yakni kurang dari 1.



	Specs	Score
3	lokasi	43.035039
1	bulan	13.772697
4	ann_ton	5.449806
2	tanggal	0.987281
8	wheel	0.340751
5	vmax	0.016202
0	tahun	0.001917
7	br_dinamis	0.001087
6	br_cart	0.000629

Gambar 33 Program pemodelan bagian *data analysis and pre processing* (hasil penilaian fitur menggunakan *univariate statistical model* tanpa outlier)

Hasil penggunaan *heatmap* menunjukkan bahwa hanya beberapa fitur yang berpengaruh terhadap kemungkinan terjadinya patah rel di suatu titik lokasi. Hal ini ditunjukkan dengan adanya sebagian fitur yang memiliki nilai korelasi negatif dengan fitur ‘acc’ yang melambangkan kemungkinan rel patah. Fitur-fitur yang memiliki nilai korelasi positif dengan fitur ‘acc’ meliputi lokasi (‘lokasi’), kecepatan maksimal (‘vmax’), dan berat dinamis rata-rata gerbong (‘br_dinamis’).



Gambar 34 Program pemodelan bagian *data analysis and pre processing* (hasil penggunaan *heatmap* tanpa outlier)

Hasil penilaian fitur data menggunakan *univariate statistical model* dan *heatmap* selanjutnya digunakan untuk menentukan kombinasi fitur yang akan digunakan dalam pemodelan. Hal ini dilakukan dengan mengetes beberapa kombinasi fitur pada model pemodelan dasar. Berdasarkan percobaan tersebut dihasilkan kesimpulan bahwa fitur-fitur yang paling berpengaruh terhadap kemungkinan patah rel berdasarkan data set yang ada tanpa *outlier* meliputi lokasi ('lokasi'), bulan ('bulan'), *annual tonnage* ('ann_ton'), kecepatan maksimal ('vmax'), dan berat dinamis rata-rata gerbong ('br_dinamis').

```
[ ] #Loading data yang telah di pre-process ke variabel untuk diolah berdasarkan heatmap
#X=df[['tahun','bulan','tanggal','lokasi','ann_ton','vmax','br_cart','br_dinamis','wheel']].values

X=df[['lokasi','bulan','vmax','ann_ton','br_dinamis']].values
Y=df['acc'].values

# Split dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=1)
```

Gambar 35 Program pemodelan bagian *data analysis and pre processing* (penyusunan data final untuk pembuatan model tanpa *outliers*)

V.2.2 Pemodelan awal

Proses ini berisi kegiatan pemodelan menggunakan teknik-teknik yang telah ditentukan tanpa mengatur nilai-nilai yang bisa dikonfigurasi dalam pembuatan model. Berikut adalah program yang digunakan dalam proses ini.

V.2.2.1 Pemodelan awal menggunakan regresi logistik

Berikut adalah program untuk pemodelan data awal menggunakan teknik regresi logistik beserta evaluasinya.

```
[53] from sklearn.linear_model import LogisticRegression
from sklearn import metrics

logreg = LogisticRegression()
logreg.fit(X_train, Y_train)

y_pred = logreg.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

Gambar 36 Pemodelan awal (regresi logistik)

V.2.2.2 Pemodelan awal menggunakan *support vector machine*

Berikut adalah program untuk pemodelan data awal menggunakan teknik *support vector machine* beserta evaluasinya.



```

#Import svm model
from sklearn import svm

#Create a svm Classifier
clf = svm.SVC()

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

Gambar 37 Pemodelan awal (*support vector machine*)

V.2.2.3 Pemodelan awal menggunakan *decision tree*

Berikut adalah program untuk pemodelan data awal menggunakan teknik *decision tree* beserta evaluasinya.



```

from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier

clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

Gambar 38 Pemodelan awal (*decision tree*)

V.2.2.4 Pemodelan awal menggunakan *random forest*

Berikut adalah program untuk pemodelan data awal menggunakan teknik *random forest* beserta evaluasinya.



```

[57] from sklearn.ensemble import RandomForestClassifier

# create regressor object
classifier = RandomForestClassifier()

# fit the regressor with x and y data
classifier.fit(X_train, Y_train)
y_pred = classifier.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

Gambar 39 Pemodelan awal (*random forest*)

V.2.2.5 Pemodelan awal menggunakan *k-nearest neighbors*

Berikut adalah program untuk pemodelan data awal menggunakan teknik *k-nearest neighbors* beserta evaluasinya.



```

[59] from sklearn.neighbors import KNeighborsClassifier

# fit the regressor with x and y data
knn=KNeighborsClassifier()
knn.fit(X_train,Y_train)

y_pred = knn.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

Gambar 40 Pemodelan awal (*k-nearest neighbors*)

V.2.3 Pemodelan akhir

Proses ini berisi kegiatan pemodelan menggunakan teknik-teknik yang telah ditentukan dengan mengatur nilai-nilai yang bisa dikonfigurasi dalam pembuatan model. Hal ini dilakukan untuk meningkatkan akurasi model dari model dasar yang telah dibuat di proses pemodelan awal. Model yang dikonfigurasi dalam proses ini hanya meliputi model *support vector machine*, *random forest*, dan *k-nearest neighbor*. Hal ini dilakukan karena tidak adanya peningkatan performa pada model regresi logistik dan *decision tree* setelah beberapa percobaan penerapan beberapa konfigurasi. Berikut adalah modifikasi program yang diterapkan dalam proses ini.

V.2.3.1 Pemodelan akhir menggunakan *support vector machine*

Nilai yang dikonfigurasi dalam pemodelan data menggunakan teknik *support vector machine* di penelitian ini adalah jenis kernel. Tipe kernel yang diuji dalam model *support vector machine* di penelitian ini meliputi gamma, sigmoid, dan linear.

```
#Import svm model
from sklearn import svm
#Create a svm Classifier
clf = svm.SVC(kernel='rbf', gamma="auto")

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

Gambar 41 Pemodelan akhir (*support vector machine* kernel rbf)

```
[ ] #Create a svm Classifier
clf = svm.SVC(kernel='sigmoid', gamma="auto") # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

Gambar 42 Pemodelan akhir (*support vector machine* kernel sigmoid)

```
[ ] #Create a svm Classifier
clf = svm.SVC(kernel='linear', gamma="auto") # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

Gambar 43 Pemodelan akhir (*support vector machine* kernel linear)

Hasil percobaan pada model dengan *outlier* menunjukkan bahwa tipe kernel yang dapat memberikan model terbaik adalah tipe 'sigmoid'. Hal ini ditunjukkan dengan

nilai akurasi model dengan tipe kernel ‘sigmoid’ yang lebih besar jika dibandingkan dengan akurasi model bertipe kernel lain, yakni sebesar 0,55. Sementara itu, nilai akurasi masing-masing model bertipe kernel ‘rbf’ dan ‘linear’ adalah sebesar 0,547 dan 0,50.

```

➡ Accuracy Score : 0.5039370078740157
Precision Score : 0.45774647887323944
Recall Score : 0.5701754385964912
F1 Score : 0.5078125

```

Gambar 44 Pemodelan akhir (evaluasi model *support vector machine* dengan *outlier* kernel linear)

```

➡ Accuracy Score : 0.547244094488189
Precision Score : 0.4956521739130435
Recall Score : 0.5
F1 Score : 0.4978165938864629

```

Gambar 45 Pemodelan akhir (evaluasi model *support vector machine* dengan *outlier* kernel rbf)

```

➡ Accuracy Score : 0.5511811023622047
Precision Score : 0.0
Recall Score : 0.0
F1 Score : 0.0
/usr/local/lib/python3.6/dist-packages/
_warn_prf(average, modifier, msg_star

```

Gambar 46 Pemodelan akhir (evaluasi model *support vector machine* dengan *outlier* kernel sigmoid)

Hasil percobaan pada model tanpa *outlier* menunjukkan bahwa tipe kernel yang dapat memberikan model terbaik adalah tipe ‘rbf’. Hal ini ditunjukkan dengan nilai akurasi model dengan tipe kernel ‘rbf’ yang lebih besar jika dibandingkan dengan akurasi model bertipe kernel lain, yakni sebesar 0,6. Sementara itu, nilai akurasi masing-masing model bertipe kernel sigmoid dan linear adalah sebesar 0,48 dan 0,47.

```

➡ Accuracy Score : 0.4696969696969697
Precision Score : 0.46835443037974683
Recall Score : 0.7789473684210526
F1 Score : 0.5849802371541502

```

Gambar 47 Pemodelan akhir (evaluasi model *support vector machine* tanpa *outlier* kernel linear)

```

➡ Accuracy Score : 0.6060606060606061
Precision Score : 0.5794392523364486
Recall Score : 0.6526315789473685
F1 Score : 0.6138613861386139

```

Gambar 48 Pemodelan akhir (evaluasi model *support vector machine* tanpa *outlier* kernel rbf)

```
➤ Accuracy Score : 0.4797979797979798  
Precision Score : 0.4797979797979798  
Recall Score : 1.0  
F1 Score : 0.6484641638225256
```

Gambar 49 Pemodelan akhir (evaluasi model *support vector machine* tanpa *outlier* kernel sigmoid)

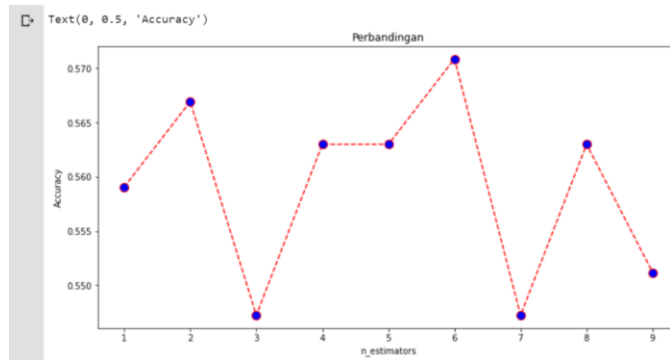
V.2.3.2 Pemodelan menggunakan *random forest*

Nilai yang dikonfigurasi dalam pemodelan data menggunakan teknik *random forest* di penelitian ini adalah nilai variabel `n_estimator` dan `random_state`. Variabel `n_estimator` digunakan untuk menentukan jumlah *decision tree* yang digunakan dalam model *random forest* yang akan dibuat. Variabel `random_state` digunakan untuk mengontrol random number generator yang digunakan ketika *randomization* merupakan bagian dari algoritma scikit-learn.

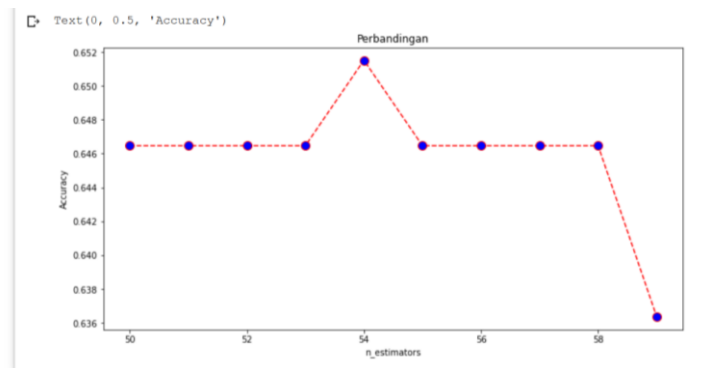
Pada pemodelan akhir menggunakan random forest di penelitian ini, hal yang pertama dilakukan adalah mencari nilai variabel `n_estimator` yang dapat menghasilkan model dengan performa terbaik. Hal ini dilakukan dengan mengecek akurasi pada model dengan nilai `n_estimator` yang berbeda-beda dari nilai `n_estimator` sama dengan satu hingga nilai `n_estimator` yang menghasilkan nilai akurasi konstan. Hasil dari percobaan ini menunjukkan bahwa `n_estimator` dengan nilai 6 memberikan model random forest dengan *outlier* performa terbaik dalam penelitian ini. Hasil percobaan juga menunjukkan bahwa `n_estimator` dengan nilai 54 memberikan model random forest tanpa *outlier* performa terbaik dalam penelitian ini.

```
from sklearn.ensemble import RandomForestClassifier  
acc = []  
  
# Melihat n_estimator yang dapat memberikan accuracy terbesar  
for i in range(1, 100):  
    classifier = RandomForestClassifier(n_estimators = i, random_state = 1)  
    classifier.fit(X_train, Y_train)  
    y_pred = classifier.predict(X_test)  
    acc.append(metrics.accuracy_score(Y_test, y_pred))  
  
plt.figure(figsize=(12, 6))  
plt.plot(range(1, 100), acc, color='red', linestyle='dashed', marker='o',  
        markerfacecolor='blue', markersize=10)  
plt.title('Perbandingan')  
plt.xlabel('n_estimators')  
plt.ylabel('Accuracy')
```

Gambar 50 Pemodelan akhir (program menentukan `n_estimator` dalam teknik *random forest*)



Gambar 51 Pemodelan akhir (n_estimator untuk model *random forest* dengan *outlier*)



Gambar 52 Pemodelan akhir (n_estimator untuk model *random forest* tanpa *outlier*)

Selanjutnya, penelitian dilakukan dengan mencari nilai variabel `random_state` yang dapat menghasilkan model dengan performa terbaik jika variabel `n_estimator` bernilai 6 untuk model dengan *outlier*. Hal ini dilakukan dengan mengecek akurasi pada model dengan nilai `random_state` yang berbeda-beda dari `random_state` sama dengan satu hingga nilai `random_state` yang menghasilkan nilai akurasi konstan. Hasil dari percobaan ini menunjukkan bahwa `random_state` dengan nilai 211 memberikan model *random forest* dengan *outlier* performa terbaik dalam penelitian ini.

```

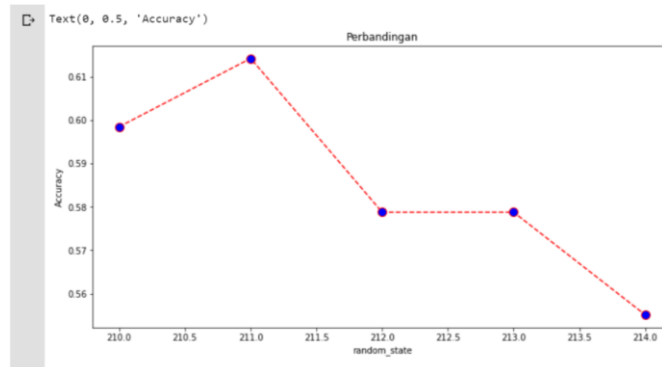
accuracy = []

# Melihat random_state yang dapat memberikan accuracy terbesar
for i in range(210, 215):
    classifier = RandomForestClassifier(n_estimators = 6, random_state = i)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    accuracy.append(metrics.accuracy_score(Y_test, y_pred))

plt.figure(figsize=(12, 6))
plt.plot(range(210, 215), accuracy, color='red', linestyle='dashed', marker='o',
         markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('random_state')
plt.ylabel('Accuracy')

```

Gambar 53 Pemodelan akhir (program menentukan `random_state` dalam teknik *random forest* untuk model dengan *outlier*)



Gambar 54 Pemodelan akhir (random_state untuk model *random forest* dengan *outlier*)

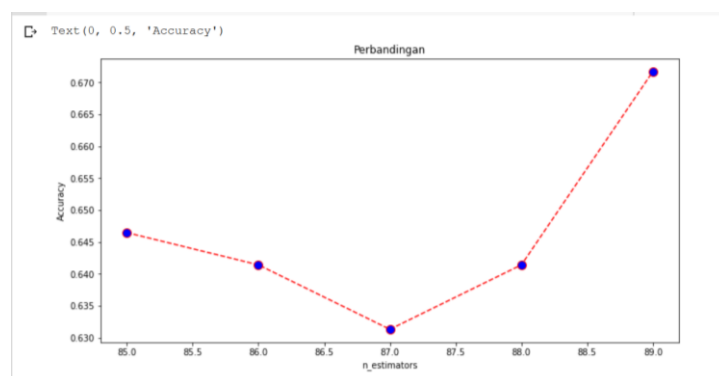
Selanjutnya, penelitian dilakukan dengan mencari nilai variabel *random_state* yang dapat menghasilkan model dengan performa terbaik jika variabel *n_estimator* bernilai 54 untuk model tanpa *outlier*. Hasil dari percobaan ini menunjukkan bahwa *random_state* dengan nilai 89 memberikan model *random forest* tanpa *outlier* dengan performa terbaik dalam penelitian ini.

```
from sklearn.ensemble import RandomForestClassifier
acc = []

# Melihat n_estimator yang dapat memberikan accuracy terbesar
for i in range(1, 300):
    classifier = RandomForestClassifier(n_estimators = 54, random_state = i)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))

plt.figure(figsize=(12, 6))
plt.plot(range(1, 300), acc, color='red', linestyle='dashed', marker='o',
         markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('n_estimators')
plt.ylabel('Accuracy')
```

Gambar 55 Pemodelan akhir (program menentukan *random_state* dalam teknik *random forest* untuk model tanpa *outlier*)



Gambar 56 Pemodelan akhir (random_state untuk model *random forest* tanpa *outlier*)

Setelah nilai *n_estimator* dan *random_state* yang dapat menghasilkan model dengan performa terbaik ditemukan, selanjutnya dilakukan fitting model final kembali. Hal ini dilakukan agar model dapat dievaluasi secara lebih detail menggunakan matriks

penilaian yang telah ditentukan. Berikut adalah program untuk pemodelan data akhir menggunakan teknik *random forest* beserta evaluasinya.

```
[64] # create regressor object
      classifier = RandomForestClassifier(n_estimators = 6, random_state = 211)

      # fit the regressor with x and y data
      classifier.fit(X_train, Y_train)
      y_pred = classifier.predict(X_test)

      print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
      print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
      print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
      print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

Gambar 57 Pemodelan akhir (program pemodelan *random forest* dengan *outlier*)

```
[ ]
      # create regressor object
      classifier = RandomForestClassifier(n_estimators = 54, random_state = 89)

      # fit the regressor with x and y data
      classifier.fit(X_train, Y_train)
      y_pred = classifier.predict(X_test)

      print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
      print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
      print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
      print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

Gambar 58 Pemodelan akhir (program pemodelan *random forest* tanpa *outlier*)

V.2.3.3 Pemodelan menggunakan k-nearest neighbor

Nilai yang dikonfigurasi dalam pemodelan data menggunakan teknik *k-nearest neighbor* di penelitian ini adalah nilai variabel `n_neighbors`. Variabel `n_neighbors` digunakan untuk menentukan jumlah *neighbors* untuk digunakan dalam *query* program *k-neighbors*. Konfigurasi dilakukan dengan mengecek akurasi pada model dengan nilai `n_neighbors` yang berbeda-beda dari nilai `n_neighbors` sama dengan satu hingga nilai `n_neighbors` yang menghasilkan nilai akurasi konstan.

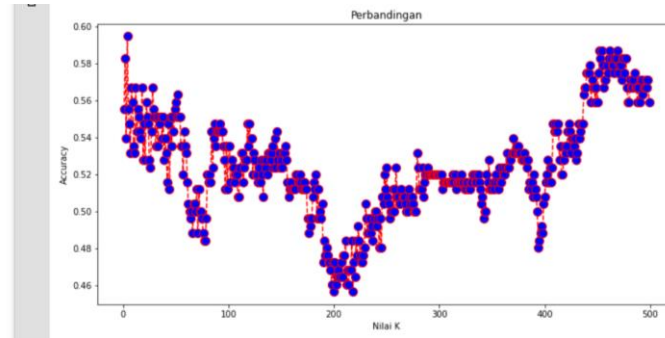
```
from sklearn.neighbors import KNeighborsClassifier
acc = []

# Calculating error for K values between 1 and 500
for i in range(1, 500):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, Y_train)
    y_pred = knn.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))

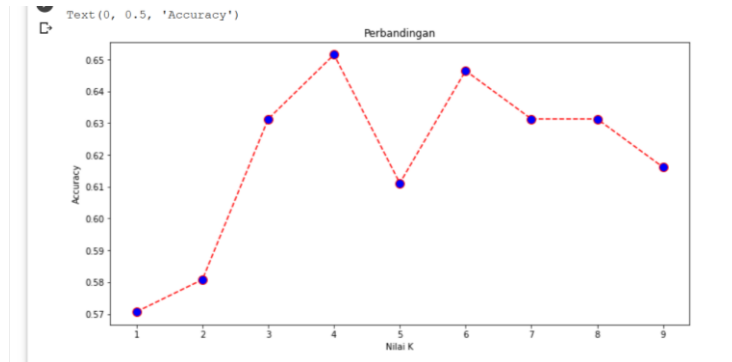
plt.figure(figsize=(12, 6))
plt.plot(range(1, 500), acc, color='red', linestyle='dashed', marker='o',
         markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('Nilai K')
plt.ylabel('Accuracy')
```

Gambar 59 Pemodelan akhir (program menentukan `n_neighbors` dalam teknik *k-nearest neighbor*)

Hasil dari percobaan ini menunjukkan bahwa `n_neighbors` dengan nilai 4 memberikan model *k-nearest neighbor* dengan *outlier* performa terbaik dalam penelitian ini. Hasil dari percobaan juga menunjukkan bahwa `n_neighbors` dengan nilai 4 memberikan model *k-nearest neighbor* tanpa *outlier* performa terbaik dalam penelitian ini.



Gambar 60 Pemodelan akhir ($n_neighbors$ untuk model k -nearest neighbor dengan outlier)



Gambar 61 Pemodelan akhir ($n_neighbors$ untuk model k -nearest neighbor tanpa outlier)

Setelah nilai $n_neighbor$ yang dapat menghasilkan model dengan performa terbaik ditemukan, selanjutnya dilakukan fitting model final kembali. Hal ini dilakukan agar model dapat dievaluasi secara lebih detail menggunakan matriks penilaian yang telah ditentukan. Berikut adalah program untuk pemodelan data akhir menggunakan teknik k -nearest neighbor beserta evaluasinya.

```
[ ]
# fit the regressor with x and y data
knn=KNeighborsClassifier(n_neighbors=4)
knn.fit(X_train,Y_train)

y_pred = knn.predict(X_test)

print('Accuracy Score : ' + str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' + str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' + str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))
```

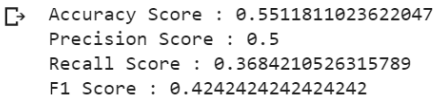
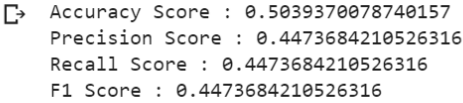
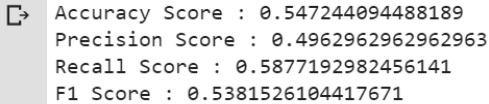
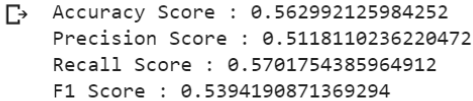
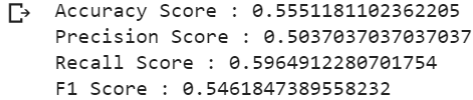
Gambar 62 Pemodelan akhir (program pemodelan k -nearest neighbor)

V.3 Evaluation

V.3.1.1 Hasil pemodelan awal dengan outlier

Eksekusi program yang telah dibuat untuk melakukan pemodelan awal dan evaluasinya pada data dengan outlier memberikan nilai dalam matriks evaluasi sebagai berikut.

Tabel 4 Hasil evaluasi pengujian model awal dengan *outlier*






Metode	Nilai Accuracy	Keluaran Program Evaluasi
Regresi Logistik	0,55	 <p>Accuracy Score : 0.5511811023622047 Precision Score : 0.5 Recall Score : 0.3684210526315789 F1 Score : 0.4242424242424242</p> <p>Gambar 63 Evaluasi (model regresi logistik awal dengan <i>outlier</i>)</p>
Support Vector Machine	0,504	 <p>Accuracy Score : 0.5039370078740157 Precision Score : 0.4473684210526316 Recall Score : 0.4473684210526316 F1 Score : 0.4473684210526316</p> <p>Gambar 64 Evaluasi (model <i>support vector machine</i> awal dengan <i>outlier</i>)</p>
Decision Tree	0,547	 <p>Accuracy Score : 0.547244094488189 Precision Score : 0.4962962962962963 Recall Score : 0.5877192982456141 F1 Score : 0.5381526104417671</p> <p>Gambar 65 Evaluasi (model <i>decision tree</i> awal dengan <i>outlier</i>)</p>
Random Forest	0,56	 <p>Accuracy Score : 0.562992125984252 Precision Score : 0.5118110236220472 Recall Score : 0.5701754385964912 F1 Score : 0.5394190871369294</p> <p>Gambar 66 Evaluasi (model <i>random forest</i> awal dengan <i>outlier</i>)</p>
K-Nearest Neighbour	0,55	 <p>Accuracy Score : 0.5551181102362205 Precision Score : 0.5037037037037037 Recall Score : 0.5964912280701754 F1 Score : 0.5461847389558232</p> <p>Gambar 67 Evaluasi (model <i>k-nearest neighbor</i> awal dengan <i>outlier</i>)</p>

Hasil pengujian awal seluruh model dengan *outlier* menunjukkan bahwa penggunaan metodologi *random forest* dalam pemodelan data di penelitian ini menghasilkan *accuracy* paling besar, yakni senilai 0,56. Hal tersebut menunjukkan bahwa hasil percobaan prediksi menggunakan model *random forrest* memiliki perbedaan nilai paling kecil dengan data actual (pengujian).

V.3.1.2 Hasil pemodelan awal tanpa *outlier*

Eksekusi program yang telah dibuat untuk melakukan pemodelan awal dan evaluasinya pada data tanpa *outlier* memberikan nilai dalam matriks evaluasi sebagai berikut.

Tabel 5 Hasil evaluasi pengujian model awal tanpa *outlier*

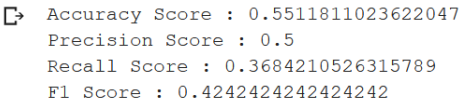
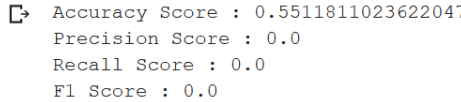
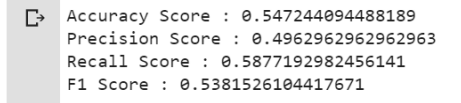
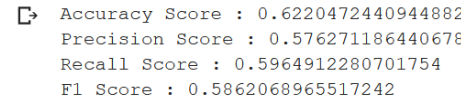
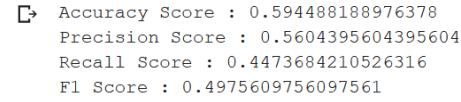
Metode	Nilai Accuracy	Keluaran Program Evaluasi
Regresi Logistik	0,5	 Accuracy Score : 0.5 Precision Score : 0.4852941176470588 Recall Score : 0.6947368421052632 F1 Score : 0.5714285714285715 Gambar 68 Evaluasi (model regresi logistik awal tanpa <i>outlier</i>)
Support Vector Machine	0,479	 Accuracy Score : 0.4797979797979798 Precision Score : 0.46825396825396826 Recall Score : 0.6210526315789474 F1 Score : 0.5339366515837105 Gambar 69 Evaluasi (model <i>support vector machine</i> awal tanpa <i>outlier</i>)
Decision Tree	0,6	 Accuracy Score : 0.6060606060606061 Precision Score : 0.5913978494623656 Recall Score : 0.5789473684210527 F1 Score : 0.5851063829787234 Gambar 70 Evaluasi (model <i>decision tree</i> awal tanpa <i>outlier</i>)
Random Forest	0,63	 Accuracy Score : 0.6313131313131313 Precision Score : 0.6078431372549019 Recall Score : 0.6526315789473685 F1 Score : 0.6294416243654823 Gambar 71 Evaluasi (model regresi <i>random forest</i> tanpa <i>outlier</i>)
K-Nearest Neighbour	0,61	 Accuracy Score : 0.6111111111111112 Precision Score : 0.5849056603773585 Recall Score : 0.6526315789473685 F1 Score : 0.6169154228855722 Gambar 72 Evaluasi (model <i>k-nearest neighbor</i> awal tanpa <i>outlier</i>)

Hasil pengujian awal seluruh model tanpa *outlier* menunjukkan bahwa penggunaan metodologi *random forest* dalam pemodelan data di penelitian ini menghasilkan *accuracy* paling besar, yakni senilai 0,63. Hal tersebut menunjukkan bahwa hasil percobaan prediksi menggunakan model *random forrest* memiliki perbedaan nilai paling kecil dengan data aktual (pengujian).

V.3.1.3 Hasil pemodelan akhir dengan *outlier*

Eksekusi program yang telah dibuat untuk melakukan pemodelan akhir dan evaluasinya pada data dengan *outlier* memberikan nilai dalam matriks evaluasi sebagai berikut.

Tabel 6 Hasil evaluasi pengujian model akhir dengan *outlier*






Metode	Nilai <i>Accuracy</i>	Keluaran Program Evaluasi
Regresi Logistik	0,55	 <p>Accuracy Score : 0.5511811023622047 Precision Score : 0.5 Recall Score : 0.3684210526315789 F1 Score : 0.4242424242424242</p> <p>Gambar 73 Evaluasi (model regresi logistik akhir dengan <i>outlier</i>)</p>
<i>Support Vector Machine</i>	0,55	 <p>Accuracy Score : 0.5511811023622047 Precision Score : 0.0 Recall Score : 0.0 F1 Score : 0.0</p> <p>Gambar 74 Evaluasi (model <i>support vector machine</i> akhir dengan <i>outlier</i>)</p>
<i>Decision Tree</i>	0,547	 <p>Accuracy Score : 0.547244094488189 Precision Score : 0.4962962962962963 Recall Score : 0.5877192982456141 F1 Score : 0.5381526104417671</p> <p>Gambar 75 Evaluasi (model <i>decision tree</i> akhir dengan <i>outlier</i>)</p>
<i>Random Forest</i>	0,622	 <p>Accuracy Score : 0.6220472440944882 Precision Score : 0.576271186440678 Recall Score : 0.5964912280701754 F1 Score : 0.5862068965517242</p> <p>Gambar 76 Evaluasi (model <i>random forest</i> akhir dengan <i>outlier</i>)</p>
<i>K-Nearest Neighbour</i>	0,594	 <p>Accuracy Score : 0.594488188976378 Precision Score : 0.5604395604395604 Recall Score : 0.4473684210526316 F1 Score : 0.4975609756097561</p> <p>Gambar 77 Evaluasi (model k-nearest neighbor akhir dengan <i>outlier</i>)</p>

Hasil pengujian akhir seluruh model menunjukkan bahwa penggunaan metodologi *random forest* dalam pemodelan data di penelitian ini menghasilkan *accuracy* paling besar, yakni 0,622. Hal tersebut menunjukkan bahwa hasil percobaan prediksi menggunakan model *random forest* memiliki perbedaan nilai paling kecil dengan data actual (pengujian).

V.3.1.4 Hasil pemodelan akhir tanpa *outlier*

Eksekusi program yang telah dibuat untuk melakukan pemodelan akhir dan evaluasinya pada data dengan *outlier* memberikan nilai dalam matriks evaluasi sebagai berikut.

Tabel 7 Hasil evaluasi pengujian model awal tanpa *outlier*

Metode	Nilai <i>Accuracy</i>	Keluaran Program Evaluasi
Regresi Logistik	0,5	 Accuracy Score : 0.5 Precision Score : 0.4852941176470588 Recall Score : 0.6947368421052632 F1 Score : 0.5714285714285715 Gambar 78 Evaluasi (model regresi logistik akhir tanpa <i>outlier</i>)
<i>Support Vector Machine</i>	0,6	 Accuracy Score : 0.6060606060606061 Precision Score : 0.5794392523364486 Recall Score : 0.6526315789473685 F1 Score : 0.6138613861386139 Gambar 79 Evaluasi (model <i>support vector machine</i> akhir tanpa <i>outlier</i>)
<i>Decision Tree</i>	0,6	 Accuracy Score : 0.6060606060606061 Precision Score : 0.5913978494623656 Recall Score : 0.5789473684210527 F1 Score : 0.5851063829787234 Gambar 80 Evaluasi (model <i>decision tree</i> akhir tanpa <i>outlier</i>)
<i>Random Forest</i>	0,67	 Accuracy Score : 0.6717171717171717 Precision Score : 0.65 Recall Score : 0.6842105263157895 F1 Score : 0.6666666666666667 Gambar 81 Evaluasi (model <i>random forest</i> akhir tanpa <i>outlier</i>)
<i>K-Nearest Neighbour</i>	0,65	 Accuracy Score : 0.6515151515151515 Precision Score : 0.6805555555555556 Recall Score : 0.5157894736842106 F1 Score : 0.5868263473053893 Gambar 82 Evaluasi (model k-nearest neighbor akhir tanpa <i>outlier</i>)

Hasil pengujian akhir seluruh model menunjukkan bahwa penggunaan metodologi *random forest* dalam pemodelan data di penelitian ini menghasilkan *accuracy* paling besar, yakni 0,67. Hal tersebut menunjukkan bahwa hasil percobaan prediksi menggunakan model *random forest* memiliki perbedaan nilai paling kecil dengan data actual (pengujian).

Berdasarkan seluruh hasil pengujian yang telah dilakukan, disimpulkan bahwa teknik *random forest* dengan data yang telah dibersihkan dari *outlier* beserta konfigurasinya adalah metode pengembangan model paling tepat untuk data kerusakan rel kereta yang telah tersedia. Konfigurasi yang dimaksud adalah *n_estimator* dengan nilai 54 dan *random_state* dengan nilai 89. Hal ini ditunjukkan dengan paling besarnya nilai *accuracy* model dengan spesifikasi tersebut dibandingkan dengan spesifikasi lainnya,

yakni 0,67. Berdasarkan hal tersebut, disimpulkan bahwa metode pemodelan *random forest* dengan spesifikasi tersebut dipilih untuk menjadi metode *machine learning* pada sistem prediksi yang akan dikembangkan.

V.4 Deployment

Deployment model *machine learning* dalam penelitian ini dibuat dalam bentuk aplikasi *website* khusus untuk menguji fungsi prediksi. Pengembangan aplikasi dilakukan menggunakan bahasa pemrograman python untuk *back-end* dan bahasa pemrograman html untuk *front-end*.

Tampilan utama aplikasi adalah sebuah form isian bagi pengguna untuk memberikan masukan yang menjadi variabel dalam menentukan kerusakan rel kereta api di suatu titik. Variabel-variabel yang dimaksud meliputi titik lokasi dihitung dari titik 0 dalam peta yang menjadi lingkup penelitian dalam satuan km, bulan kejadian kerusakan rel diperkirakan, *annual tonnage* dalam satuan juta ton, kecepatan maksimal kereta dalam satuan km/jam, dan berat dinamis dalam satuan ton. Form juga meliputi satu tombol *submit* untuk ditekan pengguna saat isian yang dimasukkan ke dalam form sudah dipastikan benar dan akan memulai perhitungan kemungkinan terjadinya kerusakan rel berdasarkan masukan terkait. Berikut adalah tampilan utama aplikasi dalam gambar.

Gambar 83 *Deployment* (tampilan utama aplikasi)

Keluaran aplikasi setelah penekanan tombol submit ada dua kemungkinan, yakni informasi mungkin terjadinya patah rel, dan informasi tidak mungkin terjadinya patah rel. Informasi mungkin terjadinya patah rel ditunjukkan dengan keluaran berupa kalimat “Ada kemungkinan terjadi patah rel”, Informasi tidak mungkin terjadinya patah rel ditunjukkan dengan keluaran berupa kalimat “tidak ada kemungkinan terjadi patah rel”. Berikut adalah contoh masing-masing keluaran yang dimaksud.

The screenshot shows a web browser window titled 'Prediksi Kerusakan Rel Kereta Api - Mozilla Firefox'. The address bar shows 'localhost:8080'. The page has a title 'Prediksi Kerusakan Rel Kereta Api' and a section 'Spesifikasi rel :'. Below this, there are input fields for 'Lokasi (km)' (252.95), 'Bulan' (12), 'Annual tonnage (juta ton)' (98.4680928), 'Kecepatan maksimal kereta (km/jam)' (75), and '> Berat dinamis (ton)' (97.7075534759358). A 'Submit' button is present. Below the inputs, the section 'Perkiraan terjadinya kerusakan rel :' shows the output 'Ada kemungkinan terjadi patah rel'.

Gambar 84 *Deployment* (tampilan hasil jika ada kemungkina rel patah)

The screenshot shows the same web application as Gambar 84, but with different input values. The 'Lokasi (km)' field now contains '304.275336995496'. The 'Submit' button is still present. Below the inputs, the section 'Perkiraan terjadinya kerusakan rel :' shows the output 'Tidak ada kemungkinan terjadi patah rel'.

Gambar 85 *Deployment* (tampilan hasil jika tidak ada kemungkinan rel patah)

BAB VI PENUTUP

VI.1 Kesimpulan

Berikut ini adalah kesimpulan yang didapatkan dari penelitian yang telah dilakukan dalam pengerjaan tugas akhir.

1. Penelitian ini menghasilkan model *machine learning* untuk mendeteksi kemungkinan patah rel di lokasi dan waktu tertentu. Hal tersebut meliputi metode menentukan teknik paling tepat hingga penerapan hasil *machine learning* dalam aplikasi sederhana yang dapat digunakan secara langsung oleh pengguna.
2. Fitur-fitur data kejadian patah rel kereta api yang memengaruhi pembentukan model *machine learning* untuk analisis prediksi kemungkinan kejadian patah rel kereta api berdasarkan data penelitian meliputi lokasi ('lokasi'), bulan ('bulan'), *annual tonnage* ('ann_ton'), kecepatan maksimal ('vmax'), dan berat dinamis rata-rata gerbong ('br_dinamis').
3. Teknik pemodelan data yang memberikan hasil paling baik jika ditinjau dari nilai akurasi dan penerapan pada data penelitian adalah *random forest* beserta konfigurasinya dengan eliminasi *outlier*. Konfigurasi yang dimaksud adalah *n_estimator* dengan nilai 54 dan *random_state* dengan nilai 89. Data penelitian yang dimaksud adalah data kejadian kerusakan rel kereta api dari PT. Kereta Api Indonesia di DIVRE III dan DIVRE IV pada tahun 2017-2019. Hasil pemodelan yang dimaksud adalah model untuk menentukan kemungkinan terjadinya rel patah pada lokasi dan waktu tertentu.
4. Nilai akurasi hasil pemodelan tertinggi yang dapat dicapai dalam penelitian adalah 0,67. Nilai tersebut menunjukkan besarnya potensi peningkatan kualitas model dalam penelitian pada masa yang akan datang agar dapat diterapkan penggunaannya dengan baik dalam dunia nyata. Nilai tersebut lebih kecil dibandingkan dengan nilai akurasi model pada penelitian sebelumnya di Amerika Utara yang mencapai nilai 87,4 persen. Namun perbedaan lokasi penelitian dan status ketersediaan data membuat penelitian ini memiliki

keunggulan dibandingkan penelitian sebelumnya. Data kejadian patah rel di Indonesia yang kelengkapan fiturnya tidak sama dengan data di Amerika Utara menyebabkan model dalam penelitian ini lebih memungkinkan untuk diterapkan dalam sistem perkereta apian di Indonesia. Selain itu, tidak semua fitur data yang berpengaruh dalam penelitian sebelumnya memberikan pengaruh besar atau positif dalam pembentukan model di penelitian ini. Hal ini ditunjukkan dalam hasil *feature selection* yang menunjukkan pengaruh masing-masing fitur data terhadap keputusan model. Penelitian ini juga menunjukkan adanya fitur data lain yang berpengaruh dalam prediksi kejadian patah rel di Indonesia yang tidak menjadi faktor pembentukan model di penelitian sebelumnya, yakni bulan terjadinya patah rel.

VI.2 Saran

Berikut ini adalah saran yang diberikan untuk pengembangan lanjutan dari penelitian ini.

1. Akurasi maksimal dalam penelitian ini yang bernilai 0,67 menunjukkan diperlukannya penelitian lebih lanjut untuk peningkatan kualitas model sebelum penelitian diterapkan di dunia nyata. Dengan mempertimbangkan bahwa nilai akurasi pada model dasar yang sudah rendah, peningkatan kualitas model dapat dilakukan dengan masa observasi pengumpulan data yang lebih lama atau penambahan lingkup lokasi pengamatan. Selain itu, penambahan fitur pada dataset juga bisa dilakukan dengan mengacu pada penelitian yang sebelumnya telah dilakukan di Amerika Utara dan observasi lanjutan.
2. Pengembangan aplikasi dalam penelitian ini harus dilakukan secara lebih lanjut untuk dapat memberikan manfaat nyata yang telah dideskripsikan dalam proses *business understanding*, yakni mengurangi tingkat kecelakaan kereta api akibat rel patah. Pengembangan aplikasi lebih lanjut bisa dilakukan untuk memastikan bahwa data yang digunakan dalam model terus di-*update* sehingga akurasi model tidak menurun dari masa ke masa.

DAFTAR PUSTAKA

- [1] Azevedo, A., & Santos, M. F. (2008). KDD, SEMMA, And CRISP-DM: A Parallel Overview. *IADIS European Conference on Data Mining* (pp. 182-185). Amsterdam: IADIS.
- [2] Aziz, A. (2017, January 16). *Laba KRL Tinggi, Tapi Penumpang Masih Keluhkan Layanan Dasar*. Retrieved from Tirto.id: <https://tirto.id/laba-krl-tinggi-tapi-penumpang-masih-keluhkan-layanan-dasar-cgWv>
- [3] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0 Step-by-Step Data Mining Guide*. SpSS Inc.
- [4] Dey, A. (2016). Machine Learning Algorithms: A Review. *International Journal of Computer Science and Information Technologies*, Vol. 7, 1174-1179.
- [5] Dick, C. T., Barkan, C. P., Chapman, E. R., & Stehly, M. P. (2003). Multivariate Statistical Model for Predicting Occurence and Location of Broken Rails. *Transportation Research Record* 1825, 48-55.
- [6] Geron, A. (2017). *Hands-On Machine Learning with Scikit-Learn & TensorFlow*. (N. Tache, Ed.) Sebastopol, California, United States of America: O'Reilly Media, Inc.
- [7] IBM. (2012). *CRISP-DM Help Overview*. Retrieved from IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/SS3RA7_15.0.0/com.ibm.spss.crispdm.help/crisp_overview.htm
- [8] Kaur, M., Dhalaria, M., Sharma, P. K., & Park, J. H. (2019). Supervised Machine-Learning Predictive Analytics for National Quality of Life Scoring. *Applied Sciences* 9(8):1613.
- [9] KNKT. (2016). *Data Investigasi Kecelakaan Perkereta apian tahun 2010 – 2016*. Jakarta: KNKT.
- [10] Leskocev, J., Rajaraman, A., & Ullman, J. D. (2014). *Mining of Massive Datasets*. Cambridge: Cambridge University Press.
- [11] Maheshwari, A., Davendralingam, N., & DeLaurentis, D. A. (2018). A Comparative Study of Machine Learning Techniques for Aviation Applications. *Aviation Technology, Integration, and Operations Conference*. Atlanta: The American Institute of Aeronautics and Astronautics, Inc.
- [12] Mueller, A. (2020). *scikit-learn 0.22.2.post1*. Retrieved from PYPI: <https://pypi.org/project/scikit-learn/#description>

- [13] Muschelli, J., Betz, J., & Varadhan, R. (2014). Chapter 7 - Binomial Regression in R. In *Handbook of Statistics 32* (pp. 257-308). Elsevier.
- [14] NCSS, L. (n.d.). *NCSS Statistical Software*. Kaysville: NCSS.
- [15] Nithya. (2016). An Analysis on Applications of Machine Learning Tools, Techniques and Practices in Health Care System. *International Journal of Advanced Research in Computer Science and Software Engineering*, 1-6.
- [16] Nugroho, L. A. (2019, January 3). *Begini Cara Petugas Kereta Api Awasi Jalur KA Pangrango, Susur Rel Sejauh 7 KM*. Retrieved from Tribunnews bogor: <https://bogor.tribunnews.com/2019/01/03/begini-cara-petugas-kereta-api-awasi-jalur-ka-pangrango-susur-rel-sejauh-7-km>
- [17] Pebrianto, F. (2020, February 10). *KAI: 2019, Jumlah Penumpang Kereta Naik 4 Juta*. Retrieved from Tempo.co: <https://bisnis.tempo.co/read/1305758/kai-2019-jumlah-penumpang-kereta-naik-4-juta/full&view=ok>
- [18] Purnamasari, D. (2018, January 16). *90% Pengguna Kereta Api adalah Penumpang Jarak Dekat*. Retrieved from Tirto.id: <https://tirto.id/90-pengguna-kereta-api-adalah-penumpang-jarak-dekat-cDiY>
- [19] Putra, A. F. (2012, September 6). *In Picture: Kegiatan Perawatan Jalur Rel Kereta Api*. Retrieved from Republika: <https://www.republika.co.id/berita/nasional/jabodetabek-nasional/12/09/06/m9xb9g-kegiatan-perawatan-jalur-rel-kereta-api>
- [20] Ray, S. (2015, August 14). *7 Regression Techniques you should know!* Retrieved from Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/>
- [21] Schafer, D. H., & Barkan, C. P. (2008). A Prediction Model for Broken Rails and an Analysis of Their Economic Impact. *Proceedings of the American Railway Engineering and Maintenance-of-Way Association Annual Conference*. Salt Lake City.
- [22] Scikit-learn. (n.d.). *Feature selection*. Retrieved from Scikit-learn: https://scikit-learn.org/stable/modules/feature_selection.html
- [23] Sutton, R. S. (1992). Introduction: The Challenge of Reinforcement. *Machine Learning*, 8, 225-227.
- [24] U.S Department of Commerce. (n.d.). *Engineering Statistics Handbook*. Retrieved from National Institute of Standards and Technology: <https://www.itl.nist.gov/div898/handbook/prc/section1/prc16.htm>

- [25] VIVA. (2017, July 30). *Kereta Api Masih Jadi Moda Transportasi Populer*. Retrieved from VIVA.co.id: <https://www.viva.co.id/gaya-hidup/travel/941094-kereta-api-masih-jadi-moda-transportasi-populer>

LAMPIRAN

1. Program pemodelan awal dengan *outlier*

```
## Inisiasi
"""

import warnings
import numpy as np
warnings.simplefilter(action='ignore', category=FutureWarning)

# Pemrosesan Awal
import pandas as pd
from sklearn.model_selection import train_test_split # Import
train_test_split function
from sklearn import metrics #Import scikit-learn metrics module
for accuracy calculation

# Import data yang digunakan
df = pd.read_excel("02_kombinasi.xlsx")

"""## Data analysis and pre processing"""

import math
import matplotlib.pyplot as plt
import seaborn as sns

# Cek data yang digunakan
df.head(3)

# Melihat kondisi data yang digunakan
df.describe()

# Jumlah data yang digunakan
df.shape

##mengisi data menggunakan rata-rata kolom
df.wheel.fillna(df.wheel.mean(),inplace=True)
df.br_dinamis.fillna(df.br_dinamis.mean(),inplace=True)
df.br_cart.fillna(df.br_cart.mean(),inplace=True)
df.vmax.fillna(df.vmax.mean(),inplace=True)
df.ann_ton.fillna(df.ann_ton.mean(),inplace=True)
df.lokasi.fillna(df.lokasi.mean(),inplace=True)

from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,-1] #target column i.e price range
#apply SelectKBest class to extract top 10 best features
bestfeatures = SelectKBest(score_func=chi2, k=9)
fit = bestfeatures.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

```

#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score'] #naming the
dataframe columns
print(featureScores.nlargest(10,'Score')) #print 10 best
features

# Melihat hubungan antar data
X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,-1] #target column i.e price range
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(8,8))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")

#Loading data yang telah di pre-process ke variabel untuk
diolah berdasarkan heatmap

X=df[['lokasi','ann_ton','vmax','br_cart','br_dinamis']].valu
es
Y=df['acc'].values

# Split dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.2, random_state=1)

"""## Membentuk Model Regresi Logistik"""

from sklearn.linear_model import LogisticRegression
from sklearn import metrics

logreg = LogisticRegression()
logreg.fit(X_train, Y_train)

y_pred = logreg.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model SVR"""

#Import svm model
from sklearn import svm

#Create a svm Classifier
clf = svm.SVC()

#Train the model using the training sets
clf.fit(X_train, Y_train)

```



```

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Decision tree"""

from sklearn.tree import DecisionTreeClassifier # Import
Decision Tree Classifier

clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Random forest"""

from sklearn.ensemble import RandomForestClassifier
# create regressor object
classifier = RandomForestClassifier()

# fit the regressor with x and y data
classifier.fit(X_train, Y_train)
y_pred = classifier.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model KNN"""

from sklearn.neighbors import KNeighborsClassifier
# fit the regressor with x and y data
knn=KNeighborsClassifier()
knn.fit(X_train,Y_train)

y_pred = knn.predict(X_test)

```

```

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

2. Program pemodelan awal tanpa *outlier*

```

## Inisiasi
"""

import warnings
import numpy as np
warnings.simplefilter(action='ignore', category=FutureWarning)

# Pemrosesan Awal
import pandas as pd
from sklearn.model_selection import train_test_split # Import
train_test_split function
from sklearn import metrics #Import scikit-learn metrics module
for accuracy calculation

# Import data yang digunakan
df = pd.read_excel("02_kombinasi.xlsx")

"""## Data analysis and pre processing"""

import math
import matplotlib.pyplot as plt
import seaborn as sns

# Cek data yang digunakan
df.head(3)

# Melihat kondisi data yang digunakan
df.describe()

# Jumlah data yang digunakan
df.shape

##mengisi data menggunakan rata-rata kolom
df.wheel.fillna(df.wheel.mean(),inplace=True)
df.br_dinamis.fillna(df.br_dinamis.mean(),inplace=True)
df.br_cart.fillna(df.br_cart.mean(),inplace=True)
df.vmax.fillna(df.vmax.mean(),inplace=True)
df.ann_ton.fillna(df.ann_ton.mean(),inplace=True)
df.lokasi.fillna(df.lokasi.mean(),inplace=True)

Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
print(IQR)

df = df[~((df < (Q1 - 1.5 * IQR)) |(df > (Q3 + 1.5 *
IQR))).any(axis=1)]

```

```

df.shape

from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

X = df.iloc[:,0:9] #independent columns
y = df.iloc[:, -1] #target column i.e price range
#apply SelectKBest class to extract top 10 best features
bestfeatures = SelectKBest(score_func=chi2, k=9)
fit = bestfeatures.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs', 'Score'] #naming the dataframe
columns
print(featureScores.nlargest(10,'Score')) #print 10 best
features

# Melihat hubungan antar data
X = df.iloc[:,0:9] #independent columns
y = df.iloc[:, -1] #target column i.e price range
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(8,8))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYl
Gn")

#Loading data yang telah di pre-process ke variabel untuk diolah
berdasarkan heatmap
#X=df[['tahun','bulan','tanggal','lokasi','ann_ton','vmax','br_c
art','br_dinamis','wheel']].values

X=df[['lokasi','bulan','vmax','ann_ton','br_dinamis']].values
Y=df['acc'].values

# Split dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.2, random_state=1)

"""## Membentuk Model Regresi Logistik"""

from sklearn.linear_model import LogisticRegression
from sklearn import metrics

logreg = LogisticRegression()
logreg.fit(X_train, Y_train)

y_pred = logreg.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

```

"""## Membentuk Model SVR"""

#Import svm model
from sklearn import svm

#Create a svm Classifier
clf = svm.SVC() # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Decision tree"""

from sklearn.tree import DecisionTreeClassifier # Import
Decision Tree Classifier

clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Random forest"""

from sklearn.ensemble import RandomForestClassifier
# create regressor object
classifier = RandomForestClassifier()

# fit the regressor with x and y data
classifier.fit(X_train, Y_train)
y_pred = classifier.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

```

"""## Membentuk Model KNN"""

from sklearn.neighbors import KNeighborsClassifier
# fit the regressor with x and y data
knn=KNeighborsClassifier()
knn.fit(X_train,Y_train)

y_pred = knn.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

3. Program pemodelan akhir dengan *outlier*

```

import warnings
import numpy as np
warnings.simplefilter(action='ignore', category=FutureWarning)

# Pemrosesan Awal
import pandas as pd
from sklearn.model_selection import train_test_split # Import
train_test_split function
from sklearn import metrics #Import scikit-learn metrics module
for accuracy calculation

# Import data yang digunakan
df = pd.read_excel("02_kombinasi.xlsx")

"""## Data analysis and pre processing"""

import math
import matplotlib.pyplot as plt
import seaborn as sns

# Cek data yang digunakan
df.head(3)

# Melihat kondisi data yang digunakan
df.describe()

# Jumlah data yang digunakan
df.shape

df.info()

##identifikasi missing value

total = df.isnull().sum().sort_values(ascending=False)
percent =
(df.isnull().sum()/df.isnull().count()).sort_values(ascending=Fa
lse)
missing_data = pd.concat([total, percent], axis=1,
keys=['Total', 'Percent'])
f, ax = plt.subplots(figsize=(15, 6))

```

```

plt.xticks(rotation='90')
sns.barplot(x=missing_data.index, y=missing_data['Percent'])
plt.xlabel('Features', fontsize=15)
plt.ylabel('Percent of missing values', fontsize=15)
plt.title('Percent missing data by feature', fontsize=15)
missing_data.head()

##mengisi data menggunakan rata-rata kolom
df.wheel.fillna(df.wheel.mean(),inplace=True)
df.br_dinamis.fillna(df.br_dinamis.mean(),inplace=True)
df.br_cart.fillna(df.br_cart.mean(),inplace=True)
df.vmax.fillna(df.vmax.mean(),inplace=True)
df.ann_ton.fillna(df.ann_ton.mean(),inplace=True)
df.lokasi.fillna(df.lokasi.mean(),inplace=True)

from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,-1] #target column i.e price range
#apply SelectKBest class to extract top 10 best features
bestfeatures = SelectKBest(score_func=chi2, k=9)
fit = bestfeatures.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score'] #naming the
dataframe columns
print(featureScores.nlargest(10,'Score')) #print 10 best
features

# Melihat hubungan antar data
X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,-1] #target column i.e price range
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(8,8))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")

#Loading data yang telah di pre-process ke variabel untuk
diolah berdasarkan heatmap
#X=df[['tahun','bulan','tanggal','lokasi','ann_ton','vmax','br_
cart','br_dinamis','wheel']].values

X=df[['lokasi','ann_ton','vmax','br_cart','br_dinamis']].values
Y=df['acc'].values

# Split dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.2, random_state=1)

"""## Membentuk Model Regresi Logistik"""

from sklearn.linear_model import LogisticRegression
from sklearn import metrics

```

```

logreg = LogisticRegression()
logreg.fit(X_train, Y_train)

y_pred = logreg.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model SVR"""

#Import svm model
from sklearn import svm

#Create a svm Classifier
clf = svm.SVC(kernel='linear') # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

#Create a svm Classifier
clf = svm.SVC(kernel='rbf', gamma="auto")

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

#Create a svm Classifier
clf = svm.SVC(kernel='sigmoid', gamma="auto") # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

```

```

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Decision tree"""

from sklearn.tree import DecisionTreeClassifier # Import
Decision Tree Classifier

clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Random forest"""

from sklearn.ensemble import RandomForestClassifier
acc = []

# Melihat n_estimator yang dapat memberikan accuracy terbesar
for i in range(1, 500):
    classifier = RandomForestClassifier(n_estimators = i,
random_state = 1)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))

plt.figure(figsize=(12, 6))
plt.plot(range(1, 500), acc, color='red', linestyle='dashed',
marker='o',
        markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('n_estimators')
plt.ylabel('Accuracy')

accuracy = []

# Melihat random_state yang dapat memberikan accuracy terbesar
for i in range(210, 215):
    classifier = RandomForestClassifier(n_estimators = 6,
random_state = i)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    accuracy.append(metrics.accuracy_score(Y_test, y_pred))

```



```

plt.figure(figsize=(12, 6))
plt.plot(range(210, 215), accuracy, color='red',
         linestyle='dashed', marker='o',
         markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('random_state')
plt.ylabel('Accuracy')

# create regressor object
classifier = RandomForestClassifier(n_estimators = 6,
random_state = 211)

# fit the regressor with x and y data
classifier.fit(X_train, Y_train)
y_pred = classifier.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model KNN"""

from sklearn.neighbors import KNeighborsClassifier
acc = []

# Calculating error for K values between 1 and 500
for i in range(1, 500):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, Y_train)
    y_pred = knn.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))
plt.figure(figsize=(12, 6))
plt.plot(range(1, 500), acc, color='red', linestyle='dashed',
         marker='o',
         markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('Nilai K')
plt.ylabel('Accuracy')

# fit the regressor with x and y data
knn=KNeighborsClassifier(n_neighbors=4)
knn.fit(X_train,Y_train)

y_pred = knn.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

4. Program pemodelan akhir tanpa outlier

```
import warnings
import numpy as np
warnings.simplefilter(action='ignore', category=FutureWarning)
from sklearn.model_selection import GridSearchCV

# Pemrosesan Awal
import pandas as pd
from sklearn.model_selection import train_test_split # Import
train_test_split function
from sklearn import metrics #Import scikit-learn metrics module
for accuracy calculation

# Import data yang digunakan
df = pd.read_excel("02_kombinasi.xlsx")

"""## Data analysis and pre processing"""

import math
import matplotlib.pyplot as plt
import seaborn as sns

# Cek data yang digunakan
df.head(3)

# Melihat kondisi data yang digunakan
df.describe()

# Jumlah data yang digunakan
df.shape

##mengisi data menggunakan rata-rata kolom
df.wheel.fillna(df.wheel.mean(),inplace=True)
df.br_dinamis.fillna(df.br_dinamis.mean(),inplace=True)
df.br_cart.fillna(df.br_cart.mean(),inplace=True)
df.vmax.fillna(df.vmax.mean(),inplace=True)
df.ann_ton.fillna(df.ann_ton.mean(),inplace=True)
df.lokasi.fillna(df.lokasi.mean(),inplace=True)

Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
print(IQR)

df = df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 *
IQR))).any(axis=1)]

df.shape

# Melihat kondisi data yang digunakan
df.describe()

from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,-1] #target column i.e price range
```

```

#apply SelectKBest class to extract top 10 best features
bestfeatures = SelectKBest(score_func=chi2, k=9)
fit = bestfeatures.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score'] #naming the dataframe
columns
print(featureScores.nlargest(10,'Score')) #print 10 best
features

# Melihat hubungan antar data
X = df.iloc[:,0:9] #independent columns
y = df.iloc[:,-1] #target column i.e price range
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(8,8))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYl
Gn")

#Loading data yang telah di pre-process ke variabel untuk diolah
berdasarkan heatmap
#X=df[['tahun','bulan','tanggal','lokasi','ann_ton','vmax','br_c
art','br_dinamis','wheel']].values

X=df[['lokasi','bulan','vmax','ann_ton','br_dinamis']].values
Y=df['acc'].values

# Split dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.2, random_state=1)

"""## Membentuk Model Regresi Logistik"""

from sklearn.linear_model import LogisticRegression
from sklearn import metrics

logreg = LogisticRegression()
logreg.fit(X_train, Y_train)

y_pred = logreg.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model SVR"""

#Import svm model
from sklearn import svm
#Create a svm Classifier
clf = svm.SVC(kernel='rbf', gamma="auto")

```

```

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

#Create a svm Classifier
clf = svm.SVC(kernel='sigmoid', gamma="auto") # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

#Create a svm Classifier
clf = svm.SVC(kernel='linear', gamma="auto") # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
      str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
      str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
      str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Decision tree"""

from sklearn.tree import DecisionTreeClassifier # Import
Decision Tree Classifier
acc = []

# Melihat n_estimator yang dapat memberikan accuracy terbesar
for i in range(960, 1000):
    classifier = DecisionTreeClassifier(max_depth=i)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))

```

```

plt.figure(figsize=(12, 6))
plt.plot(range(960, 1000), acc, color='red', linestyle='dashed',
marker='o',
        markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('n_estimators')
plt.ylabel('Accuracy')

clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model Random forest"""

from sklearn.ensemble import RandomForestClassifier
acc = []

# Melihat n_estimator yang dapat memberikan accuracy terbesar
for i in range(50, 60):
    classifier = RandomForestClassifier(n_estimators = i,
random_state = 1)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))

plt.figure(figsize=(12, 6))
plt.plot(range(50, 60), acc, color='red', linestyle='dashed',
marker='o',
        markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('n_estimators')
plt.ylabel('Accuracy')

from sklearn.ensemble import RandomForestClassifier
acc = []

# Melihat n_estimator yang dapat memberikan accuracy terbesar
for i in range(85, 90):
    classifier = RandomForestClassifier(n_estimators = 54,
random_state = i)
    classifier.fit(X_train, Y_train)
    y_pred = classifier.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))

plt.figure(figsize=(12, 6))
plt.plot(range(85, 90), acc, color='red', linestyle='dashed',
marker='o', markerfacecolor='blue', markersize=10)

```

```

plt.title('Perbandingan')
plt.xlabel('n_estimators')
plt.ylabel('Accuracy')

# create regressor object
classifier = RandomForestClassifier(n_estimators = 54,
random_state = 89)

# fit the regressor with x and y data
classifier.fit(X_train, Y_train)
y_pred = classifier.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

"""## Membentuk Model KNN"""

from sklearn.neighbors import KNeighborsClassifier
acc = []

# Calculating error for K values between 1 and 500
for i in range(1, 10):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, Y_train)
    y_pred = knn.predict(X_test)
    acc.append(metrics.accuracy_score(Y_test, y_pred))
plt.figure(figsize=(12, 6))
plt.plot(range(1, 10), acc, color='red', linestyle='dashed',
marker='o',
        markerfacecolor='blue', markersize=10)
plt.title('Perbandingan')
plt.xlabel('Nilai K')
plt.ylabel('Accuracy')

# fit the regressor with x and y data
knn=KNeighborsClassifier(n_neighbors=4)
knn.fit(X_train,Y_train)

y_pred = knn.predict(X_test)

print('Accuracy Score : ' +
str(metrics.accuracy_score(Y_test,y_pred)))
print('Precision Score : ' +
str(metrics.precision_score(Y_test,y_pred)))
print('Recall Score : ' +
str(metrics.recall_score(Y_test,y_pred)))
print('F1 Score : ' + str(metrics.f1_score(Y_test,y_pred)))

```

5. Back-end aplikasi

```
import os
import mysql.connector
import pandas as pd
import numpy as np

#Import modul Random forest
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import make_pipeline

from flask import Flask, render_template, request

app = Flask(__name__, template_folder="templates")
app.jinja_env.cache = None

db = mysql.connector.connect(
    host="localhost",
    user=" ",
    passwd=" ",
    database=" "
)
cursor = db.cursor()

df = pd.read_excel("02_kombinasi.xlsx")
test = pd.read_excel("02_kombinasi.xlsx")

##mengisi data menggunakan rata-rata kolom
df.wheel.fillna(df.wheel.mean(),inplace=True)
df.br_dinamis.fillna(df.br_dinamis.mean(),inplace=True)
df.br_cart.fillna(df.br_cart.mean(),inplace=True)
df.vmax.fillna(df.vmax.mean(),inplace=True)
df.ann_ton.fillna(df.ann_ton.mean(),inplace=True)
df.lokasi.fillna(df.lokasi.mean(),inplace=True)

@app.route("/", methods=['GET', 'POST'])
def home():
    test_y1 = None
    res=None
    if request.method == 'POST':

        X=df[['lokasi','ann_ton','vmax','br_cart','br_dinamis']].values
        Y=df['acc'].values

        # Split dataset into training set and test set
        X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
        test_size=0.2, random_state=1)

        # create regressor object
        classifier = RandomForestClassifier(n_estimators = 54,
        random_state = 89)

        # fit the regressor with x and y data
        classifier.fit(X_train, Y_train)

        test_x0 = request.form['lokasi']
        test_x1 = request.form['ann_ton']
```

```

        test_x2 = request.form['vmax']
        test_x3 = request.form['bulan']
        test_x4 = request.form['br_dinamis']

test_x=np.vstack((test_x0,test_x1,test_x2,test_x3,test_x4)).T
test_y1=classifier.predict(test_x)

print(test_y1)

if test_y1==0:
    res="Tidak ada kemungkinan terjadi patah rel"

else:
    res="Ada kemungkinan terjadi patah rel"

print(res)

else:
    print("haha")

return render_template(
    "index.html",
    result=res
)

def main():

    port = int(os.environ.get('PORT', 8080))

    app.run(debug=True, port=port, host='0.0.0.0')

if __name__ == "__main__":
    main()

```

6. *Front-end* aplikasi

```

<html>
  <head>
    <!--<link rel="stylesheet" href="{{ url_for('static',
filename='css/styling.css') }}"-->
    <title>Prediksi Kerusakan Rel Kereta Api</title>
  </head>
  <body class="center">
    <div><h1>Prediksi Kerusakan Rel Kereta Api</h1></div>
    <div class="box">
      <div>
        <h2>Spesifikasi rel : </h2>
        <div class="container">
          <form class="request" method="post"
enctype="multipart/form-data">
            <label for="lokasi">Lokasi (km)</label>
            <br><input type="text" id="lokasi"
name="lokasi"><br><br>

            <label for="bulan">Bulan</label>

```



```

        <br><input type="text" id="bulan"
name="bulan"><br><br>

        <label for="ann_ton">Annual tonnage (juta
ton)</label>
        <br><input type="text" id="ann_ton"
name="ann_ton"><br><br>

        <label for="vmax">Kecepatan maksimal kereta
(km/jam)</label>
        <br><input type="text" id="vmax"
name="vmax"><br><br>

        <label for="vmax">Berat dinamis (ton)</label>
        <br><input type="text" id="br_dinamis"
name="br_dinamis"><br><br>

        <button type="submit" style="padd"
class="button button2">Submit</button>
    </form>
</div>
</div>
<div>
    <h2 style="margin-top: 10px">Perkiraan
terjadinya kerusakan rel :</h2>
    <!-- {% if result is not none %} -->
        <div><label style="font-size: 16px; margin-
left: 10px">{{result}}</label></div>
        <!-- {% endif %} -->
    </div>
</div>
</div>

<script>

</script>
</body>
</html>

```

