

# Perbandingan Dampak Penggunaan *LASSO* dan *Hyperparameter Tuning* terhadap Performa Prediksi Kegagalan Mesin pada Model *Machine Learning*

1<sup>st</sup> Wahyu Setio Aji  
Universitas Buana Perjuangan Karawang  
Karawang, Indonesia  
If21.wahyuaji@mhs.ubpkarawang.ac.id

2<sup>nd</sup> Sutan Faisal  
Universitas Buana Perjuangan Karawang  
Karawang, Indonesia  
sutan.faisal@ubpkarawang.ac.id

3<sup>rd</sup> Santi Arum Puspita Lestari  
Universitas Buana Perjuangan Karawang  
Karawang, Indonesia  
santi.arum@ubpkarawang.ac.id

4<sup>th</sup> Yana Cahyana  
Universitas Buana Perjuangan Karawang  
Karawang, Indonesia  
yana.cahyana@ubpkarawang.ac.id

**Abstract**— Revolusi Industri 4.0 telah mendorong integrasi teknologi seperti *Big Data*, *Internet of Things (IoT)*, dan *machine learning* dalam industri untuk meningkatkan efisiensi dan mengurangi *downtime*. Salah satu tantangan utama di sektor manufaktur adalah kerusakan mesin mendadak yang dapat mengganggu produksi dan menambah biaya perawatan. *Predictive Maintenance* menjadi solusi penting dengan memanfaatkan data sensor dan model prediksi untuk mencegah kerusakan sejak dini. Penelitian ini membandingkan performa tiga algoritma *machine learning* *K-Nearest Neighbor (KNN)*, *Logistic Regression (LR)*, dan *Support Vector Machine (SVM)* dalam memprediksi kegagalan mesin berdasarkan 4 Skenario. Skenario pertama menguji tiga algoritma klasifikasi KNN, SVM, dan Logistic Regression baseline tanpa optimasi. Skenario kedua menerapkan seleksi fitur menggunakan *LASSO*, lalu menguji ketiga algoritma yang sama. Pada skenario ketiga, dilakukan *hyperparameter tuning* tanpa seleksi fitur. Sementara itu, skenario keempat menggabungkan seleksi fitur *LASSO* dengan *hyperparameter tuning* untuk memperoleh model yang lebih optimal. Hasil menunjukkan algoritma KNN terbukti sebagai model dengan kinerja paling unggul, pada Skenario 4, KNN mencapai akurasi maksimum sebesar 97.4% dengan parameter terbaik  $n\_neighbors = 4$  dan  $weights = 'distance'$ . Di bawahnya, algoritma SVM mencatat akurasi sebesar 97.2% menggunakan parameter  $C = 13.716$ ,  $gamma = 'scale'$ , dan  $kernel = 'rbf'$ . Sementara itu, *Logistic Regression* menunjukkan performa paling rendah, meskipun sudah dioptimalkan, akurasi tertingginya hanya mencapai 88.2% dengan konfigurasi  $C = 2.053$ ,  $penalty = 'l2'$ , dan  $solver = 'sag'$ .

**Kata kunci** — *Hyperparameter tuning*, *Kegagalan Mesin*, *LASSO*, *Machine learning*, *Predictive Maintenance*

## I. PENDAHULUAN

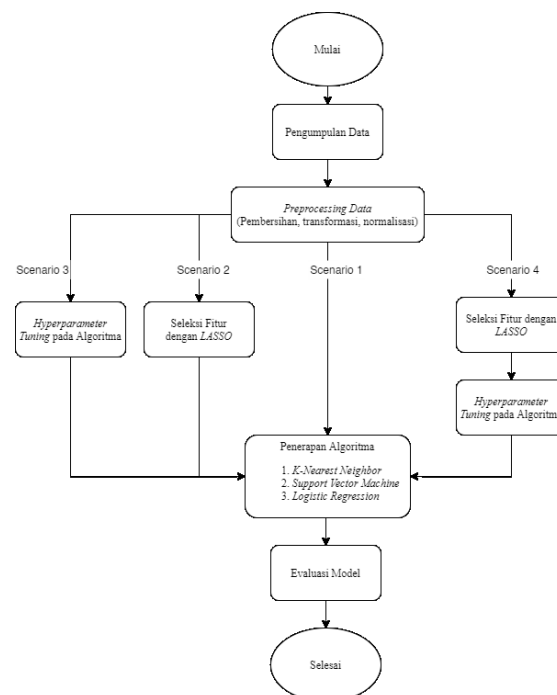
Revolusi Industri 4.0 telah mengubah dunia industri melalui integrasi otomasi dan internet [1]. Dunia industri kini memanfaatkan *Big Data*, *Artificial Intelligence (AI)*, dan *Internet of Things (IoT)* untuk efisiensi dan fleksibilitas operasional. Dalam konteks *Big Data* dan *Smart Manufacturing*, gagasan *Prognostic Health Management (PHM)* telah menjadi tren yang penting untuk memantau dan mengelola kondisi mesin [2], serta mengurangi risiko kegagalan serius guna meningkatkan keselamatan [3]. Ada tiga strategi utama dalam pemeliharaan mesin yaitu korektif, preventif, dan prediktif [4]. *Predictive Maintenance (PdM)* berbasis *Condition-Based Maintenance (CBM)* memanfaatkan data sensor *real-time* dan *machine learning* untuk mengoptimalkan jadwal perawatan dan mencegah kerusakan mesin. Pemantauan kondisi mesin melalui data sensor bertujuan mencegah kegagalan mesin yang tidak terduga. Kegagalan mesin dapat mengakibatkan produksi terhambat, dan meningkatkan biaya, baik untuk perawatan maupun akibat *Downtime* yang menyebabkan keterlambatan proses produksi [1]. Menurut Zonta et al [5] *maintenance* menyumbang 15-60% dari total biaya operasional semua manufaktur. Oleh karena itu membuat model *machine learning* yang dapat memprediksi kegagalan mesin secara akurat merupakan hal yang sangat penting [6]. Hal ini berpotensi mengurangi biaya perawatan dan penggantian secara signifikan, serta meminimalkan *Downtime*.

Namun demikian, performa model *machine learning* sangat bergantung pada kualitas data yang digunakan, serta proses *Preprocessing* yang diterapkan [7]. Penelitian ini menerapkan *Preprocessing* dengan seleksi fitur menggunakan *Least Absolute Shrinkage and Selection Operator (LASSO)* dan *Hyperparameter tuning* melalui *Random Search*. *LASSO* membantu menyederhanakan model dengan mengeliminasi fitur yang kurang relevan dan mencegah *overfitting* [8]. Efektivitas metode ini telah dibuktikan oleh Afrin et al [9] yang berhasil meningkatkan akurasi prediksi penyakit hati dari 64% menjadi 94,3% setelah penerapan *LASSO* dan *supervised learning* dengan algoritma *decision tree* dengan *score precision* 92%, *recall* 99%, *F1-score* 96%. Sementara itu, *Random Search* digunakan untuk menemukan kombinasi parameter terbaik pada *Dataset* tertentu [10]. Penelitian Rizky et al [11] *Random Search* sebagai teknik *Hyperparameter tuning* pada *Random Forest* menunjukkan kinerja baik dalam prediksi cacat perangkat lunak, dengan rata-rata AUC sebesar 0.7936 pada *Dataset ReLink* melalui penyesuaian jumlah *estimator* dan *maximum depth*. Penelitian oleh Fawaz et al [12] membandingkan metode *K-Nearest Neighbor (KNN)* dan *Naïve Bayes* dalam menentukan penghargaan untuk salesman sepeda motor. KNN dengan  $K=11$  menunjukkan performa lebih baik dengan akurasi 94.04%, presisi

83.78%, dan *recall* 96.87%, dibandingkan *Naïve Bayes* yang hanya mencapai akurasi 81.81%, presisi 72.00%, dan *recall* 81.25%. Penelitian selanjutnya yang dilakukan oleh Adnan et al [13] Model *Logistic Regression* digunakan untuk mengklasifikasikan data emisi gas rumah kaca, dengan hasil akurasi 87.60%, presisi 87.76%, sensitivitas 87.04%, dan spesifisitas 88.14%. Hasil ini menegaskan efektivitas *Logistic Regression* dalam tugas klasifikasi tersebut. Kemudian penelitian oleh Kusumaningrum et al [4] Penelitian ini membandingkan kinerja *Support Vector Machine (SVM)* dan *Random Forest (RF)* dalam memprediksi kegagalan mesin dan sisa umur pakai (RUL) menggunakan data *multi-sensor*. *Random Forest* unggul dengan akurasi 97% untuk diagnostik dan 89.6% untuk prognostik setelah validasi *10-fold cross-validation*.

Berdasarkan penelitian sebelumnya pendekatan teknik optimasi menggunakan LASSO maupun *hyperparameter tuning* telah terbukti dapat meningkatkan akurasi prediksi. Penelitian ini bertujuan untuk membandingkan performa beberapa model *machine learning*, yaitu KNN, *Logistic Regression*, dan SVM dalam memprediksi kegagalan mesin berdasarkan 4 Skenario. Skenario pertama bertujuan untuk mendapatkan gambaran awal kinerja masing-masing algoritma dengan menggunakan model *baseline* tanpa optimasi. Skenario kedua berfokus pada reduksi dimensi data melalui seleksi fitur menggunakan metode LASSO, kemudian ketiga algoritma tersebut diterapkan pada data hasil seleksi fitur untuk melihat dampak pengurangan fitur terhadap performa model. Skenario ketiga melakukan optimasi parameter melalui proses *hyperparameter tuning* pada masing-masing algoritma tanpa melakukan seleksi fitur, guna meningkatkan akurasi model berdasarkan data asli. Sedangkan skenario keempat merupakan pendekatan komprehensif, yaitu mengombinasikan seleksi fitur menggunakan LASSO dan *hyperparameter tuning*. Dalam skenario ini, fitur-fitur yang dipilih oleh LASSO digunakan untuk membangun model dari masing-masing algoritma dengan parameter terbaik yang diperoleh dari proses tuning. Keempat skenario ini dirancang untuk membandingkan pengaruh seleksi fitur dan tuning parameter terhadap performa model.

## II. METODE PENELITIAN



**Gambar 1.** Metode Penelitian

Penelitian ini diawali dengan proses pengumpulan data yang dilanjutkan dengan tahap *preprocessing*, mencakup pembersihan data dari *missing value*, *outlier*, dan duplikat, transformasi struktur data agar sesuai dengan algoritma, serta normalisasi untuk menyamakan skala antar fitur. Selanjutnya, dilakukan pemodelan dalam empat skenario. Skenario pertama menguji tiga algoritma klasifikasi KNN, SVM, dan *Logistic Regression baseline* tanpa optimasi. Skenario kedua menerapkan seleksi fitur menggunakan LASSO, lalu menguji ketiga algoritma yang sama. Pada skenario ketiga, dilakukan *hyperparameter tuning* tanpa seleksi fitur. Sementara itu, skenario keempat menggabungkan seleksi fitur LASSO dengan *hyperparameter tuning* untuk memperoleh model yang lebih optimal. Evaluasi dari semua skenario dilakukan menggunakan metrik akurasi, presisi, *recall*, dan *F1-score*.

### A. Pengumpulan Data

Penelitian ini menggunakan data kegagalan mesin yang di dapat dari laman kaggle.com dengan nama *Machine Downtime.csv* berisi kumpulan data sensor mesin yang meliputi *hydraulic pressure*, *coolant pressure*, *air system pressure*, *coolant temperature*, *hydraulic oil temperature*, *spindle bearing temperature*, *spindle vibration*, *tool vibration*, *spindle speed*, *voltage*, *torque*, *cutting* serta *Downtime* yang digunakan untuk targetnya.

|   | Date       | Machine_ID           | Assembly_Line_No | Hyd_Press  | Cool_Press | Air_Press | Cool_Temp | Oil_Temp | Spn_Bear_Temp | Spn_Vib | Tool_Vib | Speed   | Voltage | Torque    | Cutting | Downtime           |
|---|------------|----------------------|------------------|------------|------------|-----------|-----------|----------|---------------|---------|----------|---------|---------|-----------|---------|--------------------|
| 0 | 31-12-2021 | Makino-L1-Unit1-2013 | Shopfloor-L1     | 71.040000  | 6.933725   | 6.284965  | 25.6      | 46.0     | 33.4          | 1.291   | 26.492   | 25892.0 | 335.0   | 24.055326 | 3.58    | Machine_Failure    |
| 1 | 31-12-2021 | Makino-L1-Unit1-2013 | Shopfloor-L1     | 125.330000 | 4.936892   | 6.196733  | 35.3      | 47.4     | 34.6          | 1.382   | 25.274   | 19856.0 | 368.0   | 14.202890 | 2.68    | Machine_Failure    |
| 2 | 31-12-2021 | Makino-L3-Unit1-2015 | Shopfloor-L3     | 71.120000  | 6.839413   | 6.655448  | 13.1      | 40.7     | 33.0          | 1.319   | 30.608   | 19851.0 | 325.0   | 24.049267 | 3.55    | Machine_Failure    |
| 3 | 31-05-2022 | Makino-L2-Unit1-2015 | Shopfloor-L2     | 139.340000 | 4.574382   | 6.560394  | 24.4      | 44.2     | 40.6          | 0.618   | 30.791   | 18461.0 | 360.0   | 25.860029 | 3.55    | Machine_Failure    |
| 4 | 31-03-2022 | Makino-L1-Unit1-2013 | Shopfloor-L1     | 60.510000  | 6.893182   | 6.141238  | 4.1       | 47.3     | 31.4          | 0.983   | 25.516   | 26526.0 | 354.0   | 25.515874 | 3.55    | Machine_Failure    |
| 5 | 31-03-2022 | Makino-L2-Unit1-2015 | Shopfloor-L2     | 137.370000 | 5.918357   | 7.228066  | 5.4       | 48.0     | 32.7          | 0.903   | 25.597   | 27613.0 | 319.0   | 25.521330 | 3.55    | Machine_Failure    |
| 6 | 31-03-2022 | Makino-L1-Unit1-2013 | Shopfloor-L1     | 135.930000 | 6.560332   | 6.710999  | 19.3      | 48.8     | 37.4          | 1.240   | 32.138   | 26605.0 | 438.0   | 25.454652 | 3.58    | Machine_Failure    |
| 7 | 31-03-2022 | Makino-L3-Unit1-2015 | Shopfloor-L3     | 127.715164 | 5.060709   | 6.002229  | 20.8      | 45.8     | 37.5          | 1.125   | 19.823   | 14266.0 | 334.0   | 34.973004 | 2.02    | No_Machine_Failure |
| 8 | 31-03-2022 | Makino-L3-Unit1-2015 | Shopfloor-L3     | 123.618456 | 5.074380   | 6.039524  | 4.5       | 51.5     | 32.1          | 0.690   | 16.972   | 20413.0 | 278.0   | 32.519299 | 2.88    | No_Machine_Failure |
| 9 | 31-03-2022 | Makino-L3-Unit1-2015 | Shopfloor-L3     | 134.020000 | 5.567857   | 6.733096  | 14.0      | 47.9     | 35.2          | 0.748   | 36.601   | 20504.0 | 379.0   | 25.618567 | 3.93    | Machine_Failure    |

Gambar 2. Dataset Machine Downtime

## B. Preprocessing

Pada tahap ini dilakukan *preprocessing* data yang bertujuan untuk membersihkan, mengubah, dan menyiapkan data mentah agar siap digunakan dalam pemodelan *machine learning*. Langkah-langkah yang dilakukan meliputi pembersihan data, tranformasi data, penanganan *Outlier*, normalisasi data dan seleksi fitur.

### • Pembersihan Data

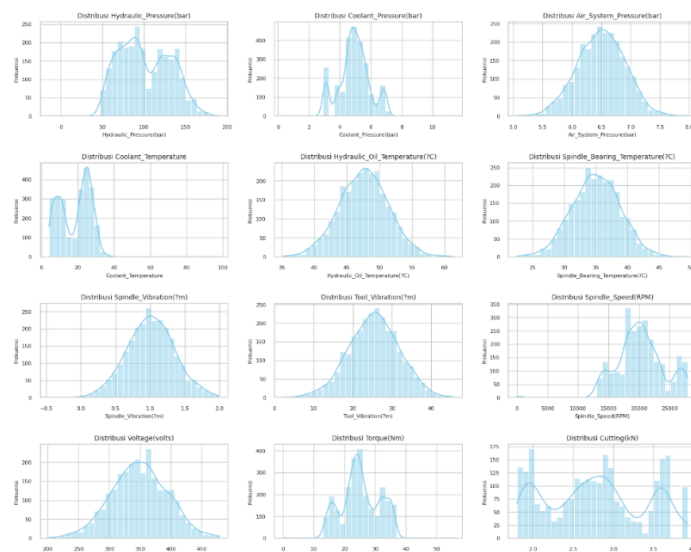
Tahap awal *Preprocessing* dimulai dengan pembersihan data, dengan tujuan untuk menangani data yang tidak relevan. Proses ini mencakup penanganan *missing value* dan penghapusan duplikat data. Sebelum dilakukan penanganan, terlebih dahulu dilakukan pengecekan terhadap *missing value* dan data duplikat. Proses pengecekan awal dapat dilihat pada Gambar 3.

data shape: (2500, 16)

|                                 | Data Type | Missing# | Missing% | Dups | Uniques | Count |
|---------------------------------|-----------|----------|----------|------|---------|-------|
| Date                            | object    | 0        | 0.0000   | 0    | 164     | 2500  |
| Machine_ID                      | object    | 0        | 0.0000   | 0    | 3       | 2500  |
| Assembly_Line_No                | object    | 0        | 0.0000   | 0    | 3       | 2500  |
| Hydraulic_Pressure(bar)         | float64   | 10       | 0.0040   | 0    | 1977    | 2490  |
| Coolant_Pressure(bar)           | float64   | 19       | 0.0076   | 0    | 1628    | 2481  |
| Air_System_Pressure(bar)        | float64   | 17       | 0.0068   | 0    | 2472    | 2483  |
| Coolant_Temperature             | float64   | 12       | 0.0048   | 0    | 275     | 2488  |
| Hydraulic_Oil_Temperature(?C)   | float64   | 16       | 0.0064   | 0    | 209     | 2484  |
| Spindle_Bearing_Temperature(?C) | float64   | 7        | 0.0028   | 0    | 204     | 2493  |
| Spindle_Vibration(?m)           | float64   | 11       | 0.0044   | 0    | 1144    | 2489  |
| Tool_Vibration(?m)              | float64   | 11       | 0.0044   | 0    | 2350    | 2489  |
| Spindle_Speed(RPM)              | float64   | 6        | 0.0024   | 0    | 1180    | 2494  |
| Voltage(volts)                  | float64   | 6        | 0.0024   | 0    | 238     | 2494  |
| Torque(Nm)                      | float64   | 21       | 0.0084   | 0    | 1327    | 2479  |
| Cutting(kN)                     | float64   | 7        | 0.0028   | 0    | 170     | 2493  |
| Downtime                        | object    | 0        | 0.0000   | 0    | 2       | 2500  |

Gambar 3. Hasil pengecekan *missing value* dan duplikat data

Dari hasil pengecekan pada Gambar 3 diketahui bahwa tidak ada duplikat data namun ada beberapa *missing value* pada kolom numerik, meskipun persentasenya relatif kecil. Sebelum melakukan imputasi terhadap *missing value* pada kolom numerik, terlebih dahulu dilakukan pengecekan terhadap distribusi data masing-masing fitur. Tujuannya adalah untuk menentukan metode imputasi yang sesuai.



Gambar 4. Hasil pengecekan distribusi data pada kolom numerik

Pengecekan terhadap distribusi data dilakukan dengan menghitung nilai skewness pada setiap fitur numerik. Berdasarkan hasil pada Gambar 4, dilakukan imputasi menggunakan nilai rata-rata (*mean*) untuk fitur-fitur yang memiliki distribusi mendekati normal, yaitu: *Air\_System\_Pressure*, *Tool\_Vibration*, *Spindle\_Vibration*, *Voltage*, *Spindle\_Bearing\_Temperature*, dan *Hydraulic\_Oil\_Temperature*. Sementara itu, fitur-fitur yang memiliki distribusi cenderung tidak simetris, seperti *Hydraulic\_Pressure*, *Coolant\_Pressure*, *Coolant\_Temperature*, *Spindle\_Speed*, *Torque*, dan *Cutting*, ditangani menggunakan metode *K-Nearest Neighbors (KNN) Imputer* agar lebih adaptif terhadap pola data. Setelah dilakukan penanganan *missing value* dengan proses imputer pada kolom-kolom yang terdapat *missing value*, hasilnya dapat dilihat pada Gambar 5 dimana sudah dipastikan *Dataset* tidak memiliki *missing value*.

|                                 |   |
|---------------------------------|---|
|                                 | 0 |
| Date                            | 0 |
| Machine_ID                      | 0 |
| Assembly_Line_No                | 0 |
| Hydraulic_Pressure(bar)         | 0 |
| Coolant_Pressure(bar)           | 0 |
| Air_System_Pressure(bar)        | 0 |
| Coolant_Temperature             | 0 |
| Hydraulic_Oil_Temperature(?C)   | 0 |
| Spindle_Bearing_Temperature(?C) | 0 |
| Spindle_Vibration(?m)           | 0 |
| Tool_Vibration(?m)              | 0 |
| Spindle_Speed(RPM)              | 0 |
| Voltage(volts)                  | 0 |
| Torque(Nm)                      | 0 |
| Cutting(kN)                     | 0 |
| Downtime                        | 0 |

Gambar 5. Hasil pengecekan setelah penanganan *missing value*

Pada tahap ini juga dilakukan penghapusan beberapa kolom yang tidak relevan terhadap analisis data sensor mesin, yaitu kolom *Date*, *Machine ID*, dan *Assembly Line No*. Penghapusan ini dilakukan karena fokus utama dari penelitian adalah pada data sensor mesin yang berperan langsung dalam proses prediksi. Hasilnya dapat dilihat pada Gambar 6.

```
Index(['Hydraulic_Pressure(bar)', 'Coolant_Pressure(bar)',
      'Air_System_Pressure(bar)', 'Coolant_Temperature',
      'Hydraulic_Oil_Temperature(?C)', 'Spindle_Bearing_Temperature(?C)',
      'Spindle_Vibration(?m)', 'Tool_Vibration(?m)', 'Spindle_Speed(RPM)',
      'Voltage(volts)', 'Torque(Nm)', 'Cutting(kN)', 'Downtime'],
      dtype='object')
```

Gambar 6. Hasil setelah kolom *Date*, *Machine ID*, dan *Assembly line* dihapus

- Transformasi Data

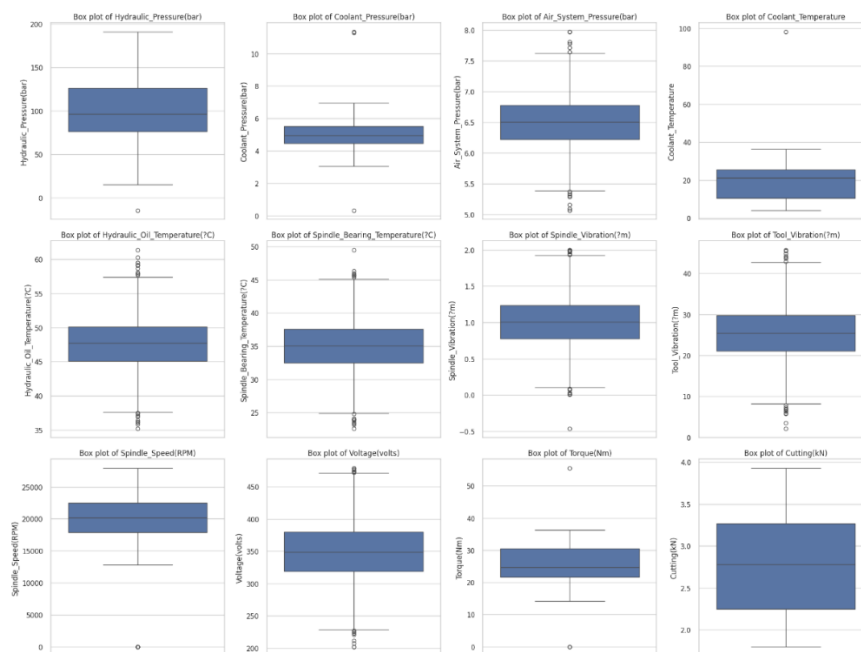
Pada tahap transformasi data, dilakukan proses encoding pada kolom *Downtime* yang berfungsi sebagai variabel target. Nilai kategori pada kolom tersebut diubah ke dalam format numerik, yaitu "*machine failure*" dikonversi menjadi 1, dan "*no\_machine\_failure*" dikonversi menjadi 0, agar dapat digunakan dalam pemodelan *machine learning*.

|  |          |
|--|----------|
|  | Downtime |
|  | 1        |
|  | 1        |
|  | 1        |
|  | 1        |
|  | 1        |
|  | 1        |
|  | 0        |
|  | 0        |

Gambar 7. Hasil *Encoding*

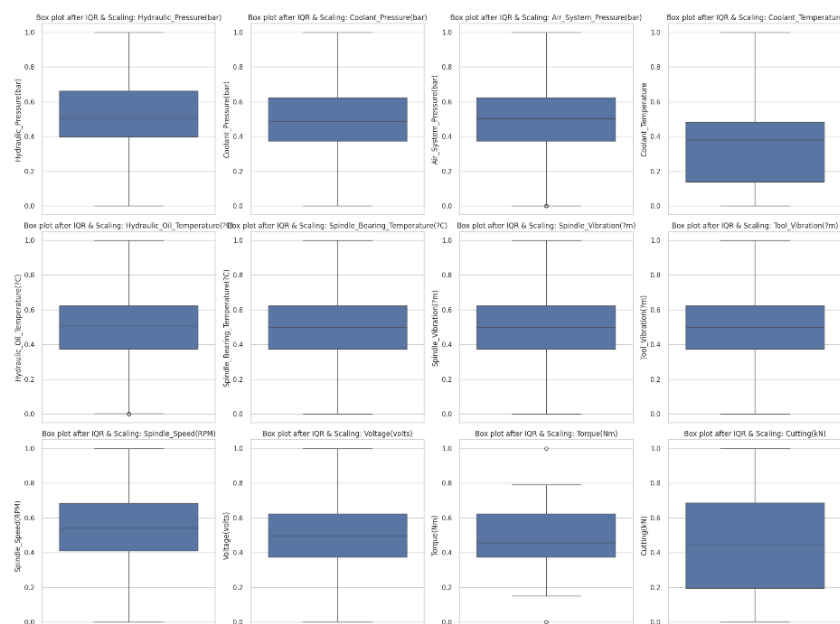
- Penanganan *Outlier* & Normalisasi Data

*Outlier* data merupakan data yang memiliki nilai yang jauh menyimpang dari sebagian besar nilai lainnya dalam kumpulan data. Pengecekan *Outlier* dilakukan menggunakan visualisasi *boxplot* untuk setiap variabel numerik. Hasil dari pengecekan ini dapat dilihat pada Gambar 8, yang menunjukkan adanya beberapa nilai ekstrem di sejumlah fitur.



Gambar 8. Pengecekan *Outlier* sebelum dilakukan penanganan

Namun, sebelum proses penanganan *Outlier* dilakukan, data terlebih dahulu dibagi (*split*) menjadi data latih dan data uji untuk menghindari terjadinya data *leakage* yang dapat memengaruhi keakuratan model. *Dataset* dibagi menjadi dua bagian yaitu 80% untuk data pelatihan dan 20% untuk data pengujian (*testing*). Setelah proses pembagian data selesai, tahap selanjutnya adalah melakukan penanganan *Outlier* menggunakan metode IQR pada data latih, kemudian dilanjutkan dengan normalisasi untuk menyamakan skala antar fitur menggunakan metode *Min-Max Normalization* pada data latih tersebut. Untuk memastikan proses *preprocessing* data berjalan secara terstruktur dan konsisten, terutama saat diterapkan pada data uji, seluruh tahapan ini dikemas ke dalam sebuah *pipeline*. Dengan menggunakan *pipeline*, setiap langkah praprosesan dapat dieksekusi secara berurutan dan terhindar dari kesalahan pengolahan data di luar urutan yang semestinya. Visualisasi melalui *boxplot* dapat dilihat pada Gambar 9 hasilnya menunjukkan bahwa nilai-nilai ekstrem pada fitur numerik telah berhasil diatasi. Fitur-fitur yang sebelumnya memiliki *Outlier* kini berada dalam rentang yang wajar, tanpa adanya nilai yang menyimpang secara signifikan dari distribusi data utama. Selain penanganan *Outlier*, hasil normalisasi juga dapat diamati pada visualisasi tersebut. Seluruh nilai pada fitur numerik telah berada dalam rentang 0 hingga 1 setelah dilakukan normalisasi menggunakan *Min-Max Normalization*, sehingga model dapat belajar secara lebih optimal tanpa dipengaruhi oleh perbedaan skala antar variabel.



Gambar 9. Hasil setelah dilakukan penanganan *Outlier* dan normalisasiC. Skenario 1 : *Baseline Model*

Pada skenario ini dilakukan pengujian terhadap 3 algoritma klasifikasi yaitu KNN, SVM, dan *Logistic Regression* terhadap data yang sudah dilakukan *preprocessing* sebelumnya. *Baseline model* tanpa optimasi LASSO maupun *hyperparameter tuning* bertujuan memperoleh gambaran awal kinerja algoritma.

## D. Skenario 2 : Model dengan LASSO

Dalam Skenario 2 ini, metode LASSO digunakan sebagai teknik seleksi fitur. LASSO memiliki kemampuan untuk melakukan regularisasi sekaligus mengeliminasi variabel yang koefisiennya mengecil hingga nol, sehingga hanya fitur-fitur penting yang dipertahankan dalam proses pemodelan [9]. Untuk menentukan nilai parameter *alpha* yang optimal dalam metode LASSO, digunakan pendekatan *LassoCV* dengan rentang *alpha* yang ditentukan menggunakan *np.logspace(-4, 0, 50)*. Rentang ini mencakup 50 nilai *alpha* secara logaritmik dari 0.0001 hingga 1, yang bertujuan untuk mengeksplorasi berbagai tingkat regularisasi. Proses validasi silang dilakukan sebanyak 5 *fold* (*cv=5*) untuk memastikan pemilihan *alpha* yang stabil dan akurat.

E. Skenario 3 : Model dengan *Hyperparameter tuning*

Di Skenario 3 ini mengarah pada penyempurnaan konfigurasi model, dengan cara dilakukan *hyperparameter tuning* untuk KNN, SVM, dan *Logistic Regression* menggunakan metode *Random Search*. Tujuannya adalah menemukan kombinasi parameter optimal yang mampu memaksimalkan performa model. Pada algoritma KNN, parameter yang dioptimasi meliputi *n\_neighbors*, yaitu jumlah tetangga terdekat yang dipertimbangkan dalam rentang 1 hingga 20, serta *weights*, yakni metode pembobotan dengan pilihan '*uniform*' atau '*distance*'. Untuk algoritma SVM, dilakukan tuning terhadap parameter C sebagai pengontrol regularisasi dalam rentang 0.1 hingga 20, *kernel* dengan pilihan '*linear*', '*sigmoid*', '*poly*', dan '*rbf*', serta *gamma*, yang diatur dengan nilai '*scale*', '*auto*', dan nilai numerik antara 0.05 hingga 0.2. Sementara itu, pada *Logistic Regression*, parameter yang dioptimasi mencakup C dalam rentang 0.1 hingga 20, *penalty* dengan pilihan '*l1*' dan '*l2*', serta *solver* yang terdiri dari '*liblinear*', '*lbfgs*', dan '*sag*', masing-masing disesuaikan dengan jenis penalti yang digunakan.

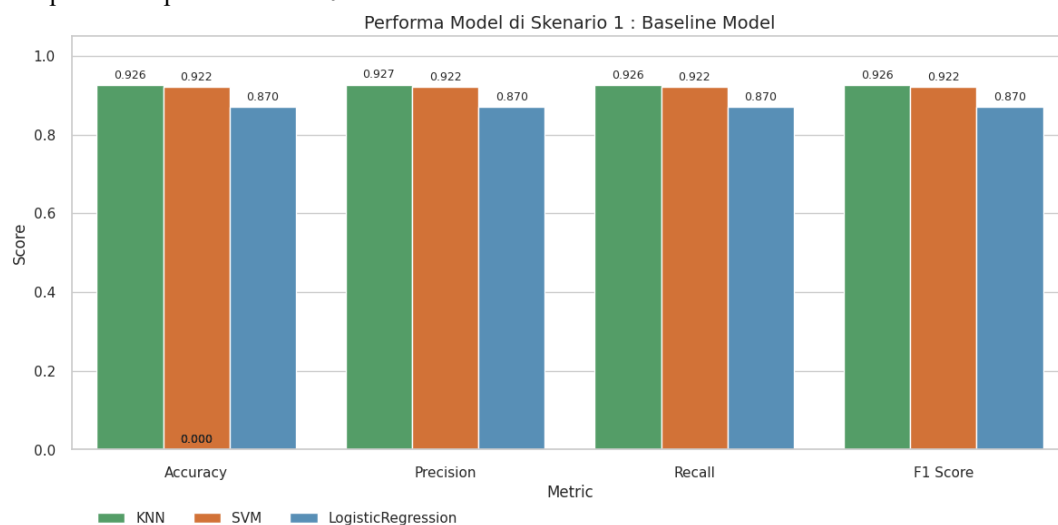
F. Skenario 4 : Model dengan LASSO & *Hyperparameter tuning*

Pada skenario 4 menggabungkan dua pendekatan bertahap: pertama, fitur diseleksi menggunakan LASSO untuk menentukan subset yang paling berpengaruh dengan penentuan nilai *alpha* sama dengan skenario 2. Kedua, dilakukan optimasi menggunakan *hyperparameter tuning* pada KNN, SVM, dan *Logistic Regression* melalui *Random Search* dengan rentang parameter sama seperti pada skenario 3.

## III. PEMBAHASAN DAN HASIL

A. Skenario 1 : *Baseline Mode*

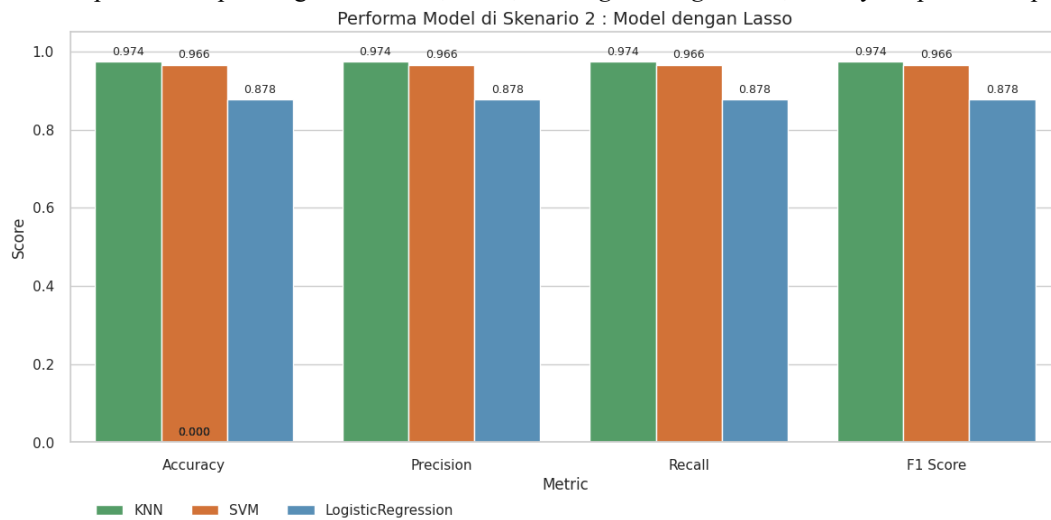
Pada pengujian awal tanpa optimasi, KNN mencatat akurasi tertinggi sebesar 92.6%, dengan presisi, *recall*, dan *F1-score* yang konsisten di kisaran 92.6–92.7%. Hal ini menunjukkan kemampuan KNN dalam mengenali pola secara stabil. SVM menyusul dengan akurasi 92.2%, dan seluruh metrik evaluasinya menunjukkan angka yang sama yaitu 92.2%, menandakan performa yang seimbang tanpa kelemahan mencolok. Sementara itu, *Logistic Regression* memperoleh akurasi 87%. Meskipun lebih rendah, metriknya tetap konsisten, menunjukkan bahwa model ini mampu menjaga keseimbangan antara kesalahan positif dan negatif. Visualisasi hasil dapat dilihat pada Gambar 10.



Gambar 10. Performa Model di Skenario 1

### B. Skenario 2 : Model dengan LASSO

Berdasarkan hasil dari *LassoCV*, diperoleh nilai  $\alpha$  terbaik sebesar 0.000146, yang menghasilkan model dengan performa optimal. Sebanyak 6 fitur terpilih dari total 12 fitur awal, keenam fitur tersebut adalah *Hydraulic\_Pressure(bar)*, *Coolant\_Pressure(bar)*, *Coolant\_Temperature*, *Spindle\_Speed(RPM)*, *Torque(Nm)*, dan *Cutting(kN)*. Kemudian fitur yang terpilih oleh LASSO dilakukan pemodelan pada algoritma KNN, SVM, dan *Logistic Regression*, hasilnya dapat dilihat pada Gambar 11.

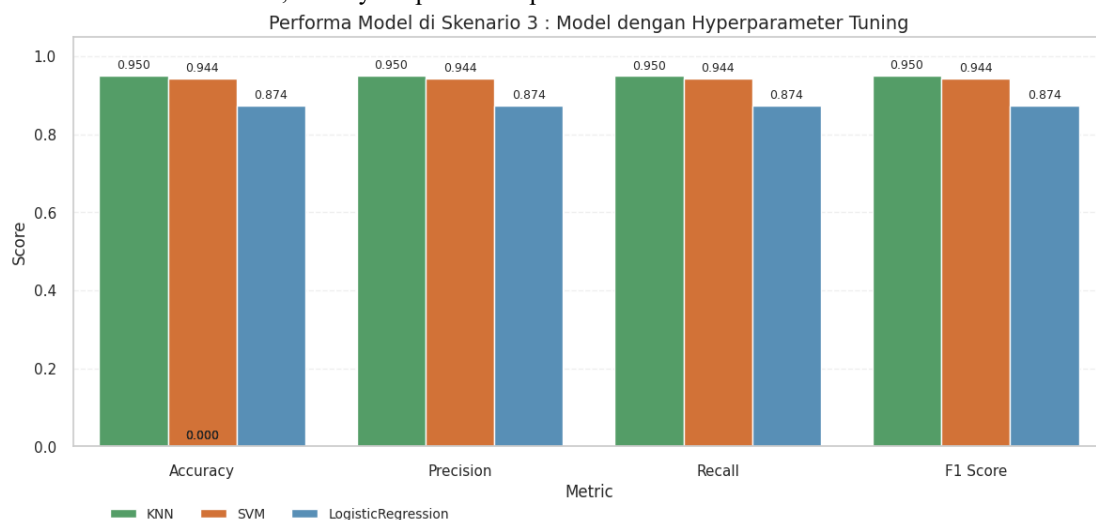


Gambar 11. Performa Model di Skenario 2

KNN mengalami kenaikan performa yang signifikan setelah seleksi fitur, dengan akurasi meningkat dari 92.6% menjadi 97.4%. Seluruh metrik evaluasi juga konsisten di angka tersebut, menunjukkan bahwa pengurangan fitur berhasil memperkuat kemampuan algoritma dalam mengenali pola lokal. Hal ini kemungkinan disebabkan oleh tersaringnya fitur-fitur redundan yang sebelumnya mengganggu perhitungan jarak. SVM pun menunjukkan peningkatan signifikan, dari 92.2% menjadi 96.6%, dengan presisi, *recall*, dan *F1-score* yang seluruhnya seimbang. Hasil ini mengisyaratkan bahwa subset fitur hasil LASSO sangat selaras dengan karakteristik *kernel* SVM, sehingga proses pemisahan kelas menjadi lebih optimal. Sebaliknya, *Logistic Regression* hanya mengalami perbaikan marginal. Akurasinya naik tipis dari 87.0% menjadi 87.8%, menandakan bahwa model ini kurang responsif terhadap reduksi fitur dalam skenario ini.

### C. Model dengan Hyperparameter tuning

Skenario ketiga difokuskan untuk menyempurnakan konfigurasi model melalui proses *hyperparameter tuning* dengan memanfaatkan metode Random Search, hasilnya dapat dilihat pada Gambar 12.



Gambar 12. Performa Model di Skenario 3

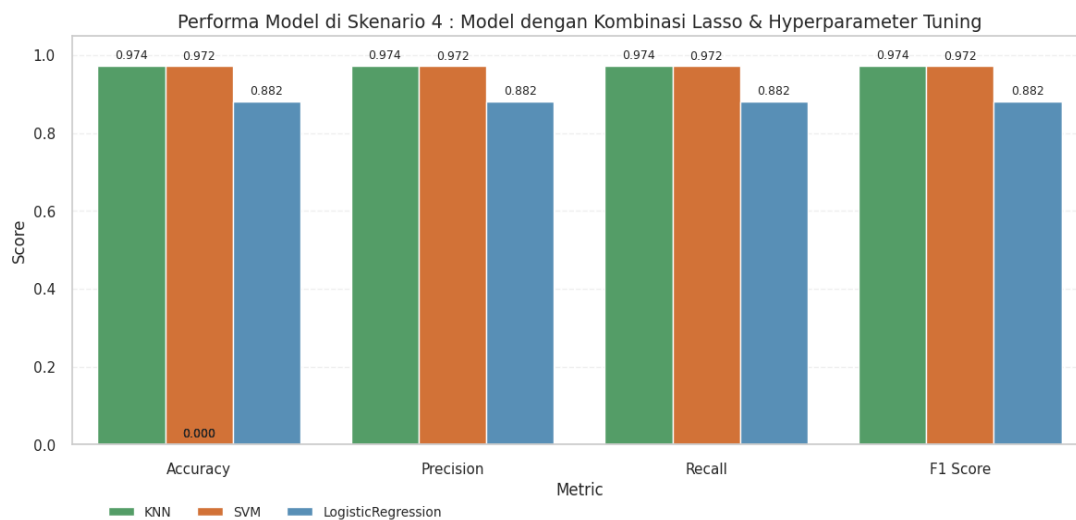
Hasilnya Setelah dilakukan tuning, KNN menjadi algoritma dengan performa terbaik, mencatat akurasi 95% dengan seluruh metrik evaluasi (presisi, *recall*, *F1-score*) konsisten di angka 95%, mencerminkan model yang seimbang dan stabil. Konfigurasi optimalnya adalah  $n\_neighbors = 10$  dan  $weights='distance'$ . Penggunaan *kernel* RBF dengan  $C=7.23$  dan  $gamma='scale'$  terbukti efektif dalam meningkatkan kinerja dan membuat SVM mencapai akurasi 94.4% dengan seluruh metrik yang seragam di 94.4% menegaskan kestabilan model ini. Sementara itu, *Logistic Regression* belum menunjukkan peningkatan signifikan. Meskipun



tuning  $C=2.05$ ,  $penalty='l2'$ ,  $solver='sag'$  sedikit meningkatkan nilai presisi, *recall*, dan *F1-score* menjadi 87.4%, akurasi tetap stagnan di 87%.

#### D. Skenario 4 : Model dengan LASSO & Hyperparameter tuning

Skenario 4 ini dilakukan pendekatan komprehensif dengan cara mengkombinasikan seleksi fitur LASSO dan *hyperparameter tuning*, dimana hasil seleksi fitur LASSO digunakan untuk membuat model dari ketiga algoritma dengan pemilihan parameter yang optimal menggunakan *hyperparameter tuning*. Hasilnya KNN berhasil mempertahankan performa tertinggi dengan akurasi 97.4%, meskipun  $n\_neighbor$  dikurangi dari 10 pada Skenario 3 menjadi 4. Konfigurasi  $n\_neighbors = 4$  dan  $weights = 'distance'$  tetap optimal berkat kualitas fitur hasil seleksi LASSO. Hal ini menegaskan bahwa relevansi fitur lebih penting daripada jumlah parameter dalam meningkatkan akurasi. SVM mencatatkan rekor baru dengan akurasi 97.2%, naik 0.6% dari Skenario 2 dan 2.8% dari Skenario 3. Penggunaan  $C=13.72$  dan *kernel* RBF dalam ruang fitur yang telah direduksi terbukti efektif. Nilai  $C$  yang lebih besar menunjukkan model lebih fleksibel terhadap kesalahan kecil, yang sesuai dengan fitur-fitur penting hasil seleksi LASSO. *Logistic Regression* juga mengalami peningkatan, meskipun lebih terbatas. Akurasi naik menjadi 88.2%, meningkat 0.4% dari Skenario 2 dan 1.2% dari *Baseline*. Meskipun konfigurasi parameter tetap sama ( $C=2.05$ ,  $penalty='l2'$ ), hasil ini menunjukkan bahwa eliminasi fitur yang tidak relevan melalui LASSO turut membantu meningkatkan kinerja model *linear*, Visualisasi hasil dapat dilihat pada Gambar 13.



Gambar 13. Performa Model di Skenario 4

Hasil performa model pada keempat skenario dapat dilihat dalam Tabel 1, yang memuat nilai rata-rata dari masing-masing metrik evaluasi.

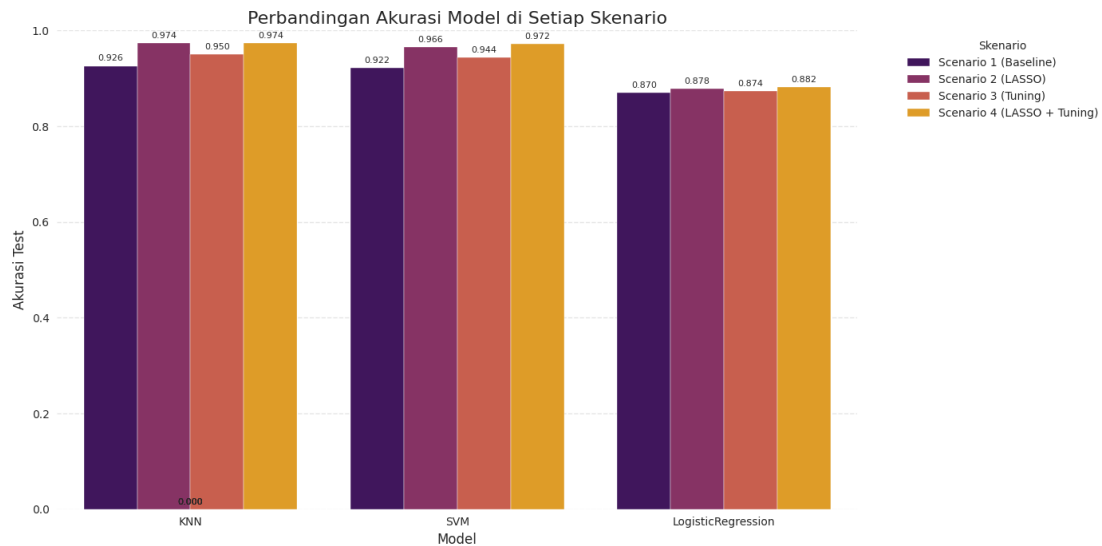
**Tabel 1.** Performa Setiap Model pada Keempat Skenario

| Model | Skenario       | Accuracy | Precision | Recall | F1-score |
|-------|----------------|----------|-----------|--------|----------|
| KNN   | Baseline       | 92.6 %   | 92.7 %    | 92.6 % | 92.6 %   |
|       | LASSO          | 97.4 %   | 97.4 %    | 97.4 % | 97.4 %   |
|       | Tuning         | 95.0 %   | 95.0 %    | 95.0 % | 95.0 %   |
|       | LASSO + Tuning | 97.4 %   | 97.4 %    | 97.4 % | 97.4 %   |
| SVM   | Baseline       | 92.2 %   | 92.2 %    | 92.2 % | 92.2 %   |
|       | LASSO          | 96.6 %   | 96.6 %    | 96.6 % | 96.6 %   |
|       | Tuning         | 94.4 %   | 94.4 %    | 94.4 % | 94.4 %   |
|       | LASSO + Tuning | 97.2 %   | 97.2 %    | 97.2 % | 97.2 %   |
| LR    | Baseline       | 87.0 %   | 87.0 %    | 87.0 % | 87.0 %   |
|       | LASSO          | 87.8 %   | 87.8 %    | 87.8 % | 87.8 %   |
|       | Tuning         | 87.4 %   | 87.4 %    | 87.4 % | 87.4 %   |
|       | LASSO + Tuning | 88.2 %   | 88.2 %    | 88.2 % | 88.2 %   |

Berdasarkan evaluasi terhadap algoritma KNN, SVM, dan *Logistic Regression* pada empat skenario yang diuji, KNN terbukti secara konsisten menjadi model dengan performa terbaik. Pada Skenario 1, KNN mencatat akurasi 92.6%, unggul tipis dari SVM



(92.2%) dan cukup jauh di atas *Logistic Regression* (87.0%). Lonjakan signifikan terjadi pada Skenario 2, di mana KNN mencapai akurasi 97.4%, disusul SVM 96.6%, sementara LR hanya naik tipis ke 87.8%. Pola serupa berlanjut pada Skenario 3 dan 4, dengan KNN masing-masing meraih 95.0% dan 97.4%, sedikit di atas SVM yang mencetak 94.4% dan 97.2%. Sementara itu, *Logistic Regression* secara konsisten menunjukkan kinerja paling rendah, tidak pernah melampaui akurasi 88.2% di seluruh skenario. Nilai presisi, *recall*, dan *F1-score* pada masing-masing model selaras dengan akurasinya, mencerminkan konsistensi performa dalam mengklasifikasikan prediksi positif dengan tepat. Visualisasi perbandingan akurasi dapat dilihat pada Gambar 14.



Gambar 14. Perbandingan Akurasi Model di Setiap Algoritma

#### IV. KESIMPULAN DAN SARAN

Hasil pengujian empat skenario menunjukkan bahwa penerapan seleksi fitur LASSO dan *hyperparameter tuning* secara signifikan meningkatkan akurasi model prediksi. Pada Skenario 1 (*Baseline*), KNN mencatat akurasi tertinggi sebesar 92,6% tanpa optimasi. Skenario 2 menggunakan LASSO dengan  $\alpha$  optimal 0.000146 yang berhasil menyaring enam fitur paling relevan, menghasilkan peningkatan akurasi secara konsisten. Skenario 3 menerapkan *hyperparameter tuning* tanpa seleksi fitur dan tetap menunjukkan hasil lebih baik dibanding baseline. Skenario 4 yang menggabungkan LASSO dan tuning memberikan hasil terbaik, membuktikan efek sinergis kedua teknik. Secara keseluruhan, KNN menjadi algoritma paling unggul dengan akurasi tertinggi sebesar 97,4% ( $n\_neighbors = 4$ ,  $weights = 'distance'$ ), diikuti oleh SVM dengan 97,2%, sementara *Logistic Regression* tetap memiliki performa terendah dengan akurasi 88,2% meskipun telah dioptimasi. Dengan demikian, kombinasi LASSO dan *hyperparameter tuning* terbukti efektif meningkatkan performa model, dan KNN menjadi metode terbaik dalam memprediksi potensi kegagalan mesin.

Pengembangan penelitian ke depan dapat diarahkan pada penerapan model prediktif ini ke dalam sistem nyata berbasis *Internet of Things (IoT)*, agar model tidak hanya dianalisis secara *offline*, tetapi juga dapat bekerja secara *real-time* dalam memantau kondisi mesin secara langsung. Selain itu, penggabungan pendekatan *machine learning* dengan visualisasi *dashboard* interaktif juga dapat meningkatkan kejelasan informasi bagi pengguna akhir, khususnya teknisi atau operator di lapangan.

#### PENGAKUAN

Makalah ini merupakan bagian dari penelitian Tugas Akhir penulis yang disusun sebagai kontribusi akademik dalam bidang visi komputer, dengan judul “Perbandingan Dampak Penggunaan LASSO dan *Hyperparameter tuning* terhadap Performa Prediksi Kegagalan Mesin pada Model *Machine learning*”. Penelitian ini dilakukan secara mandiri tanpa dukungan sponsor dari pihak mana pun, dan disusun sebagai bagian dari pemenuhan syarat akademik di Universitas Buana Perjuangan Karawang.

#### DAFTAR PUSTAKA

- [1] N. Hafidhoh, A. P. Atmaja, G. N. Syaifuddiin, I. B. Sumafta, S. M. Pratama, and H. N. Khasanah, “*Machine learning* untuk Prediksi Kegagalan Mesin dalam *Predictive Maintenance System*,” *J. Masy. Inform.*, vol. 15, no. 1, pp. 56–66, May 2024, doi: 10.14710/jmasif.15.1.63641.
- [2] Z. M. Çınar, A. A. Nuhu, Q. Zeeshan, O. Korhan, M. Asmael, and B. Safaei, “*Machine learning* in *Predictive Maintenance* towards sustainable smart manufacturing in industry 4.0,” *Sustain.*, vol. 12, no. 19, Oct. 2020, doi: 10.3390/su12198211.
- [3] L. Biggio and I. Kastanis, “Prognostics and Health Management of Industrial Assets: Current Progress and Road Ahead,” *Front. Artif. Intell.*, vol. 3, Nov. 2020, doi: 10.3389/frai.2020.578613.
- [4] D. Kusumaningrum, N. Kurniati, and B. Santosa, “*Machine learning* for *Predictive Maintenance*,” in *Proceedings of the International Conference on Industrial Engineering and Operations Management*, Michigan, USA: IEOM Society International, Apr. 2021, pp. 2348–2356. doi: 10.46254/SA02.20210717.
- [5] T. Zonta, C. A. da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, “*Predictive Maintenance* in the Industry 4.0: A systematic literature review,” *Comput. Ind. Eng.*, vol. 150, p. 106889, Dec. 2020, doi: 10.1016/j.cie.2020.106889.

- [6] A. Shaheen, M. Hammad, W. Elmedany, R. Ksantini, and S. Sharif, "Machine failure prediction using joint reserve intelligence with feature selection technique," *Int. J. Comput. Appl.*, vol. 45, no. 10, pp. 638–646, Oct. 2023, doi: 10.1080/1206212X.2023.2260619.
- [7] M. Raka Sujono, A. Bahtiar, and B. Irawan, "Analisis Model *Machine learning* Untuk Jenis Aspal Di Jawa Barat Menggunakan Algoritma Decision Tree Dan Random Forest," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 7, no. 6, pp. 3886–3891, 2024, doi: 10.36040/jati.v7i6.8275.
- [8] S. Abdumalikov, J. Kim, and Y. Yoon, "Performance Analysis and Improvement of *Machine learning* with Various Feature Selection Methods for EEG-Based Emotion Classification," *Appl. Sci.*, vol. 14, no. 22, 2024, doi: 10.3390/app142210511.
- [9] S. Afrin *et al.*, "Supervised *machine learning* based liver disease prediction approach with LASSO feature selection," *Bull. Electr. Eng. Informatics*, vol. 10, no. 6, pp. 3369–3376, Dec. 2021, doi: 10.11591/eei.v10i6.3242.
- [10] P. Putu, P. Pratistha, R. R. Huizen, and D. Hermawan, "Pengaruh *Hyperparameter tuning* pada DeepSpeech2," *Pros. Semin. Has. Penelit. Inform. dan Komput. 2024*, vol. 1, pp. 824–828, 2024, [Online]. Available: <https://www.openslr.org/12/>
- [11] M. H. Rizky, M. R. Faisal, I. Budiman, D. Kartini, and F. Abadi, "Effect of *Hyperparameter tuning* Using Random Search on Tree-Based Classification Algorithm for Software Defect Prediction," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 18, no. 1, p. 95, 2024, doi: 10.22146/ijccs.90437.
- [12] M. A. Fawaz, K. Khairul, and A. P. U. Siahaan, "Analysis of Performance Comparison between K-Nearest Neighbor (KNN) Method and Naïve Bayes Method in Reward for Honda Motorcycle Salesman Tour," *sinkron*, vol. 8, no. 3, pp. 1932–1944, 2024, doi: 10.33395/sinkron.v8i3.13935.
- [13] A. Adnan, A. M. Yolanda, G. Erda, N. E. Goldameir, and Z. Indra, "The Comparison of Accuracy on Classification Climate Change Data with *Logistic Regression*," *Sink. J. dan Penelit. Tek. Inform.*, vol. 7, no. 1, pp. 56–61, 2023, [Online]. Available: <https://jurnal.polgan.ac.id/index.php/sinkron/article/view/11914/1262>