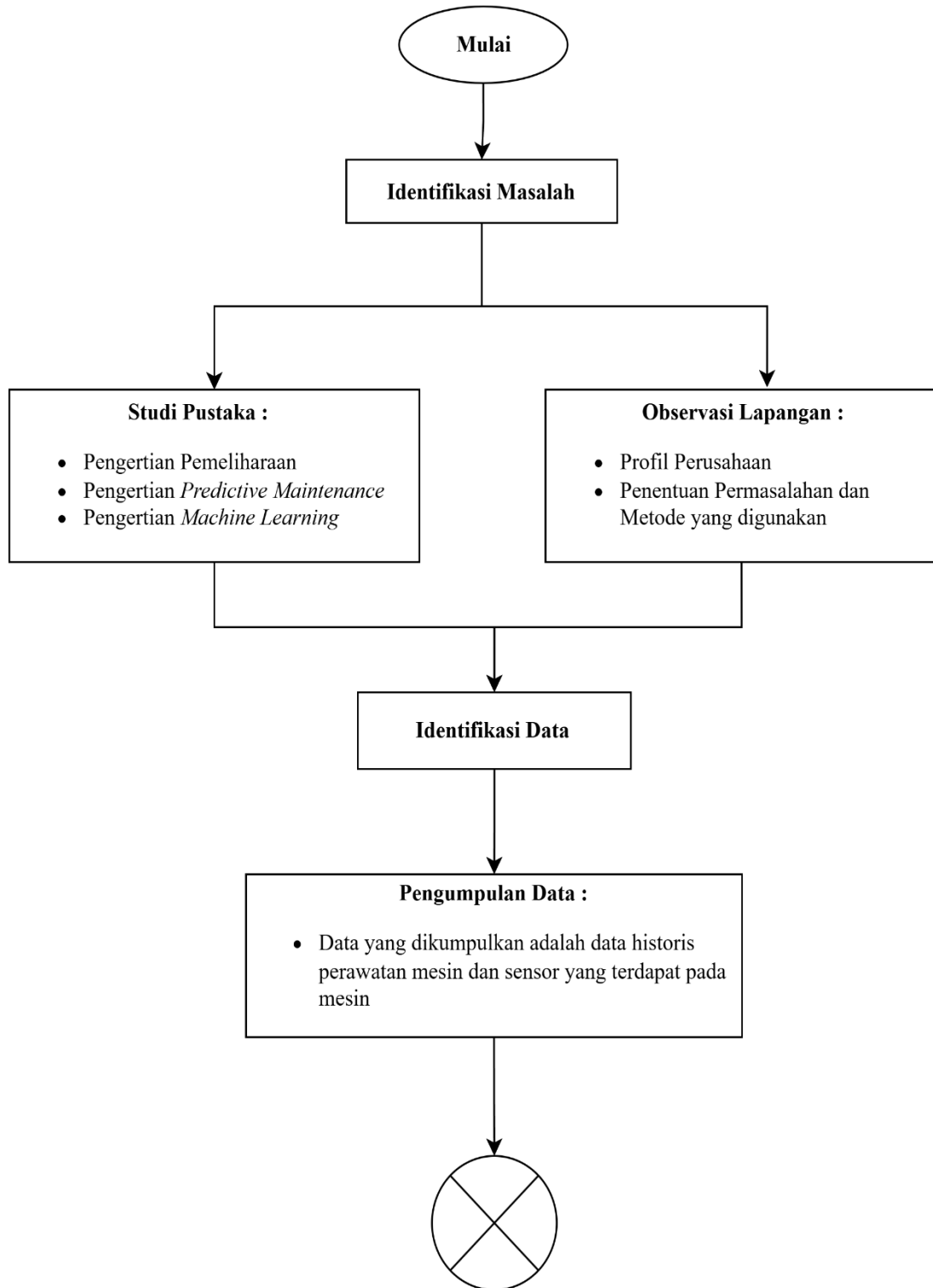
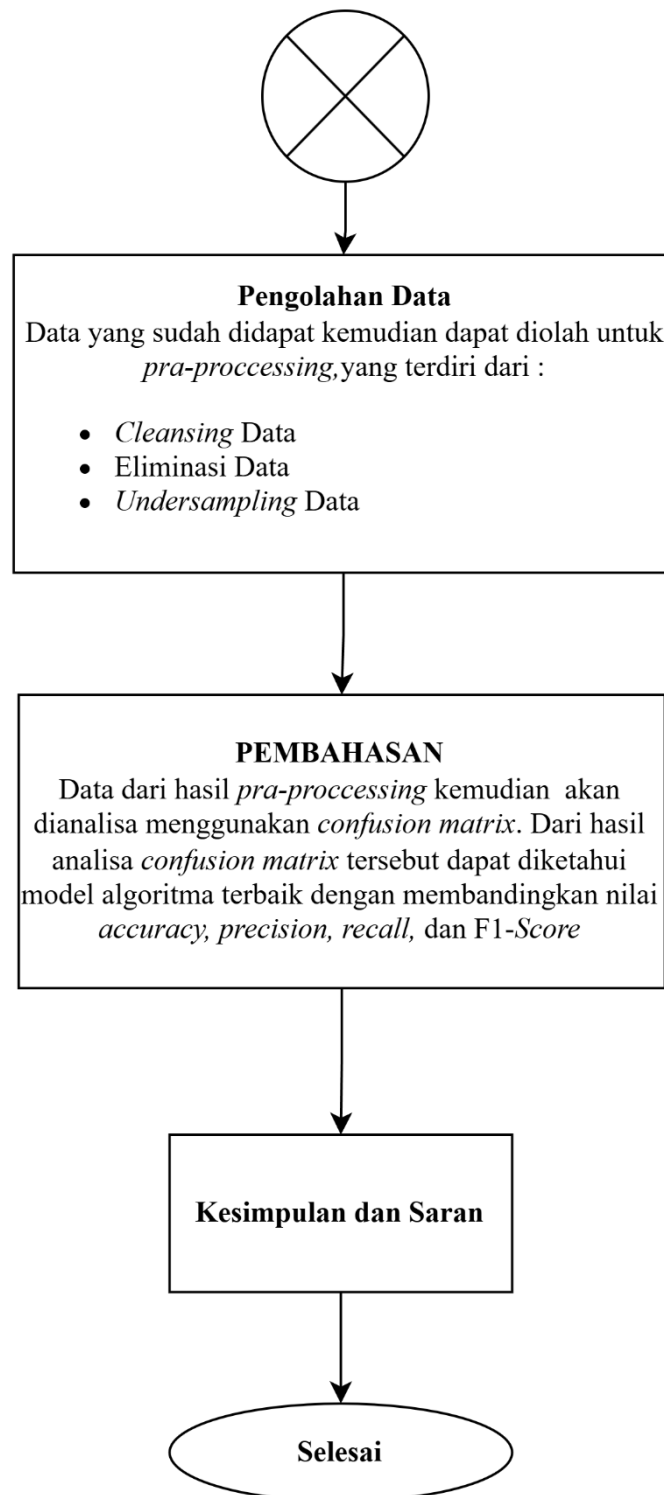


# BAB III METODOLOGI PENELITIAN

## 3.1 Diagram Alir (*Flowchart*) Penelitian





### **3.2 Keterangan Diagram Alir**

#### **1. Mulai**

Merupakan langkah awal dalam melakukan penelitian

#### **2. Identifikasi Masalah**

Merupakan langkah dimana dilakukannya penentuan masalah yang akan menjadi fokus utama.

#### **3. Observasi Lapangan**

Merupakan langkah untuk melakukan pengumpulan data aktual lapangan, yaitu PT. Jinwoo Engineering Indonesia. Pengumpulan data observasi ini dilakukan dengan observasi, diskusi dengan pihak terkait, dan dokumentasi untuk mengetahui permasalahan utama yang ada.

#### **4. Studi Pustaka**

Merupakan langkah kegiatan pencarian pustaka untuk memahami teori yang berhubungan dengan permasalahan yang terjadi. Studi ini berguna untuk memberikan informasi kepada pembaca tentang berbagai hasil penelitian lain yang berkaitan dengan penelitian yang sedang dilakukan, menghubungkan penelitian dengan literatur yang ada agar hasil penelitian dapat dipertanggung jawabkan secara ilmiah.

#### **5. Identifikasi Data**

Merupakan langkah kegiatan dimana data yang sudah didapat akan diproses lebih lanjut.

#### **6. Pengumpulan Data**

Merupakan langkah lanjutan identifikasi data dimana proses ini adalah tahap pengumpulan data apa saja yang dibutuhkan dalam penelitian Beberapa data yang telah dikumpulkan sebagai berikut:

- Observasi
- Data Perawatan Mesin
- Data Sensor yang digunakan Mesin

## **7. Pengolahan Data**

Merupakan langkah proses pengolahan data yang sudah didapat dari pengumpulan data untuk kemudian dilakukan *pre-processing*. Berikut langkah pengolahan data yang dilakukan dengan *pre-processing*:

- *Cleansing Data* : Menghapus data duplikat, *outlier*, atau data kosong
- *Eliminasi Data* : Menghapus fitur (kolom) yang kurang relevan atau redundan (*feature selection*)
- *Undersampling Data* : Mengurangi jumlah data pada kelas mayoritas untuk menangani ketidakseimbangan (*imbalance*)

## **8. Pembahasan**

Merupakan langkah dimana data yang telah diolah kemudian dianalisis lebih lanjut menggunakan *confusion matrix* untuk mengidentifikasi model algoritma terbaik.

## **9. Kesimpulan dan Saran**

Merupakan langkah untuk memberikan kesimpulan dan saran dari permasalahan yang terjadi berdasarkan hasil penelitian ini. Pada tahap ini, kesimpulan dari analisis data dirumuskan, disertai saran untuk perbaikan.

## **10. Selesai**

Penelitian telah selesai dilakukan.

## BAB IV HASIL DAN PEMBAHASAN

### 4.1 Gambaran Umum Perusahaan

#### 4.1.1 Profil Perusahaan



**Gambar 4. 1** Logo PT. Jinwoo Engineering Indonesia

PT. Jinwoo Engineering Indonesia (JEI) merupakan perusahaan yang bergerak di bidang *Plastic Sheet and Metal Fabrication Product* untuk kebutuhan *part* elektronik. JEI berdiri pada tahun 2000 di Tangerang yang dirintis dibawah pimpinan Mr. Jung Jin Chul seorang pengusaha dari korea.

Pada awalnya JEI bergerak pada proses pembuatan alat musik khususnya gitar, namun pada tahun 2009 JEI mulai bergerak pada manufaktur pembuatan *part* kulkas (*refrigerator*) dengan bekerja sama dengan LG Indonesia. JEI memproduksi *part* kulkas (*refrigerator*) diantaranya yaitu *backplate*, *frontplate*, dan *inner refrigerator* untuk memenuhi kebutuhan LG Indonesia.



**Gambar 4. 2** Produk PT. JEI

#### 4.1.2 Visi dan Misi Perusahaan

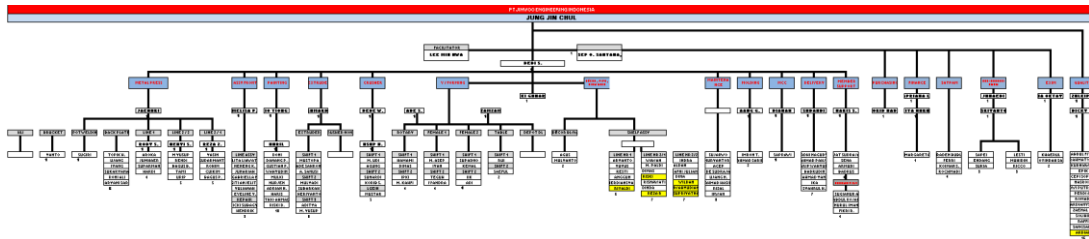
Visi :

Menjadi perusahaan nasional terkemuka di industri konstruksi serta memberikan kesejahteraan kepada karyawan dan administrator melalui komitmen terhadap tata kelola perusahaan yang baik.

Misi :

Selalu menjaga komitmen untuk menyediakan kepuasan pelanggan dengan menjaga harga yang sesuai dan waktu pemrosesan tepat pada setiap produk yang dibuat dengan tidak mengurangi kualitas pada tiap produk serta memberikan kesempatan kerja kepada sumber daya manusia potensial.

#### 4.1.3 Struktur Organisasi Perusahaan



**Gambar 4. 3** Struktur Organisasi PT. JEI

Berikut merupakan struktur organisasi PT. Jinwoo Engineering Indonesia :

#### 4.2 Pengumpulan Data

Data yang digunakan pada penelitian ini merupakan data historis perawatan mesin dan getaran dalam bentuk frekuensi yang dihasilkan pada mesin yang diteliti. Pengambilan data getaran yang digunakan menggunakan sensor yang ada pada mesin yang berfungsi untuk mendeteksi keadaan yang ada pada mesin tersebut. Sensor tersebut terletak pada beberapa titik yang terhubung langsung ke pusat pengendalian proses pada mesin.

Data yang digunakan adalah sebanyak 1200 data yang masih mentah dalam format JSON file.

```

{
  "data": {
    "getSensorReports": {
      "reports": [
        {
          "sensor_uid": "840706-7715",
          "type": 0,
          "summary": "{\"peak\": [8.34, 8.573, 2.917, 9.04, 4.045, 7.991, 3.856, 5.547, 1.0,
9.494, 1.065, 4.676, 9.152, 2.26, 7.5, 2.68, 8.163, 0.714, 4.974, 8.928, 8.133, 3.475, 3.751, 9.365,
5.04, 7.913, 4.322, 6.506, 5.141, 0.992, 3.411, 5.727, 7.71, 9.641, 5.526, 7.018, 7.491, 5.725, 4.519,
4.792, 6.202, 0.48, 4.493, 7.241, 5.599, 4.156, 0.89, 1.052, 2.824, 4.939, 1.205, 7.484, 7.291, 7.782,
9.327, 1.857, 6.286, 5.955, 4.616, 3.035, 5.976, 1.758, 2.01, 6.586, 6.312, 6.367, 7.519, 5.155, 6.805,
1.912, 7.124, 7.181, 0.521, 3.71, 6.233, 0.477, 3.035, 0.328, 7.706, 1.166, 2.673, 6.043, 4.855, 0.883,
6.185, 6.366, 2.546, 3.349, 6.883, 4.919, 1.577, 1.341, 7.719, 4.207, 2.265, 1.642, 9.824, 1.346, 4.469,
4.14, 5.108, 6.48, 0.887, 8.861, 5.056, 8.001, 1.117, 9.731, 1.922, 0.795, 8.481, 6.707, 9.284, 9.171,
9.688, 2.635, 8.321, 9.386, 9.8, 3.558, 7.588, 9.904, 2.95, 9.133, 6.191, 2.65, 7.249, 3.567, 4.724,
6.795, 0.262, 5.655, 9.313, 6.754, 0.427, 6.618, 1.278, 1.415, 2.692, 4.22, 8.575, 2.891, 9.607, 4.89,
3.261, 4.301, 7.377, 8.3, 1.563, 6.505, 4.242, 6.231, 2.276, 6.61, 4.487, 9.858, 5.574, 1.341, 3.602,
2.661, 8.441, 9.623, 3.382, 6.839, 4.596, 6.722, 5.241, 3.693, 5.423, 0.371, 4.908, 0.535, 6.493, 9.117,
9.27, 6.743, 6.961, 4.108, 5.377, 2.441, 4.107, 0.593, 6.863, 7.116, 5.952, 2.022, 5.838, 3.885, 7.649,
8.906, 5.797, 9.384, 7.432, 5.335, 4.404, 1.512, 7.986, 3.37, 6.506, 7.888, 6.936, 6.695, 1.251, 0.376,
3.783, 6.415, 1.566, 2.304, 8.014, 9.557, 6.046, 3.963, 4.932, 8.518, 6.339, 0.53, 0.491, 2.573, 3.128,
2.23, 6.937, 1.324, 0.311, 9.726, 5.104, 4.92, 6.187, 8.531, 5.769, 4.229, 0.838, 1.837, 4.218, 9.69,
4.6, 5.371, 5.743, 9.0, 1.356, 7.952, 7.146, 5.517, 2.75, 9.992, 7.592, 3.251, 1.661, 6.387, 0.624,
7.36, 5.894, 8.784, 1.172, 9.747, 8.583, 0.671, 0.475, 0.911, 5.058, 2.932, 8.684, 6.047, 3.75, 8.138,
0.51, 0.154, 6.733, 8.415, 4.867, 8.623, 5.692, 5.796, 3.319, 2.145, 1.075, 5.892, 4.017, 5.267, 9.858,

```

**Gambar 4. 4** Data File Mentah File JSON

#### 4.2.1 Pre-Processing Data

*Pre-Processing Data* merupakan tahapan persiapan dan transformasi data mentah sebelum dilakukan analisis lebih lanjut. Penelitian ini menggunakan *cleansing data*, identifikasi data NA, dan eliminasi data yang tidak diperlukan.

##### 4.2.1.1 Cleansing Data

*Cleansing Data* adalah proses membersihkan data dari kesalahan atau ketidaksesuaian yang bisa mengganggu hasil analisis. *Cleansing Data* yang dilakukan adalah dengan merubah data mentah yang belum bisa dibaca oleh manusia menjadi *readable* dengan membuat baris dan kolom sesuai dengan jenis data yang didapatkan. Data yang didapat meliputi UID, Tanggal,Waktu, *Vacuum Pressure*, *Air Pressure*, *Vibration*, *Temperature Heater*, *Temperatur Process*, *Rate* dan Kondisi Mesin. *Cleansing Data* dilakukan menggunakan Bahasa pemograman *python* menggunakan *software jupyterlab*. Berikut merupakan proses *Cleansing Data* yang dilakukan :

```

import pandas as pd
import numpy as np
import json

#Buat json file dapat dibaca oleh manusia
import pprint

#Load Data JSON Maintenance
with open ("./DATA SENSOR - NEW.json") as f:
    data = json.load(f)

#Bentuk Dataframe yang dibutuhkan
df = pd.DataFrame(data)

df.rename(columns={
    'Vacuum_Pressure_kPa': 'Vacuum.Pressure',
    'Air_Pressure_bar': 'Air.Pressure',
    'Vibration_g': 'Vibration',
    'Temperatur_Heater_C': 'Temperatur.Heater',
    'Temperatur_Proses_C': 'Temperatur.Proses',
    'Rate': 'Rate',
    'Kondisi_Mesin': 'Kondisi Mesin'
}, inplace=True)

expected_columns = ['UID', 'Tanggal', 'Waktu', 'Vacuum.Pressure', 'Air.Pressure', 'Vibration', 'Temperatur.Heater',
                    'Temperatur.Proses', 'Rate', 'Kondisi Mesin']
df = df[expected_columns]

print(df.head(25))

```

**Gambar 4.5** *Cleansing Data*

```

# Identifikasi nilai NA per kolom
print("\nJumlah nilai NA per kolom:")
print(df.isna().sum())

# Tampilkan baris yang mengandung NA (jika ada)
na_rows = df[df.isna().any(axis=1)]
print("\nBaris dengan nilai NA:")
print(na_rows)

```

**Gambar 4.6** Identifikasi Data NA

	UID	Tanggal	Waktu	Vacuum.Pressure	Air.Pressure	Vibration
0	UID_0001	2024-11-21	10:12	-83.013143	6.025045	0.016391
1	UID_0002	2024-09-15	11:47	-85.553057	5.914119	0.020884
2	UID_0003	2024-09-04	15:17	-82.409246	6.024460	0.017267
3	UID_0004	2024-12-04	19:37	-78.907881	6.108660	0.018642
4	UID_0005	2024-10-06	20:33	-85.936613	6.009772	0.028367
5	UID_0006	2024-10-02	17:18	-85.936548	6.008118	0.026702
6	UID_0007	2024-09-29	09:53	-78.683149	5.859602	0.013502
7	UID_0008	2024-09-18	11:18	-81.930261	5.867420	0.024149
8	UID_0009	2024-12-04	11:23	-86.877898	5.719479	0.024057
9	UID_0010	2024-09-14	10:19	-82.829760	6.349915	0.014259
10	UID_0011	2024-11-26	08:45	-86.853671	5.751227	0.024094
11	UID_0012	2024-12-04	16:08	-86.862919	5.861419	0.027690
12	UID_0013	2024-12-24	12:02	-84.032151	5.856319	0.014387
13	UID_0014	2024-11-09	08:35	-92.653121	6.178985	0.015412
14	UID_0015	2024-09-12	12:44	-91.899671	5.941010	0.025088
15	UID_0016	2024-11-15	10:40	-87.249150	6.249548	0.021357
16	UID_0017	2024-10-25	21:48	-89.051324	5.865302	0.022757
17	UID_0018	2024-09-05	15:06	-83.743011	6.055799	0.021703
18	UID_0019	2024-09-04	21:00	-88.632096	5.832931	0.021953
19	UID_0020	2024-09-12	17:18	-90.649215	6.429030	0.013368
20	UID_0021	2024-09-28	15:30	-79.137405	5.762480	0.025237
21	UID_0022	2024-09-30	15:21	-85.903105	6.061964	0.025848
22	UID_0023	2024-11-04	10:03	-84.729887	6.126755	0.018853
23	UID_0024	2024-11-17	12:55	-90.698993	6.082760	0.019783
24	UID_0025	2024-09-04	15:07	-87.177531	5.962942	0.012344

**Gambar 4.7** Hasil *Cleansing Data*



```

Jumlah nilai NA per kolom:
UID                0
Tanggal            0
Waktu              0
Vacuum.Pressure    0
Air.Pressure       0
Vibration          0
Temperatur.Heater  0
Temperatur.Proses  0
Rate               0
Kondisi Mesin     0
dtype: int64

Baris dengan nilai NA:
Empty DataFrame
Columns: [UID, Tanggal, Waktu, Vacuum.Pressure, Air.Pressure, Vibration, Temperatur.Heater, Temperatur.Proses, Rate, Kondisi Me
sin]
Index: []

```

**Gambar 4. 8** Hasil Identifikasi Data NA

Setelah melakukan proses pembersihan data (*data cleansing*), data tersebut menjadi lebih terstruktur dan mudah dipahami sehingga siap digunakan dalam tahap pengolahan selanjutnya. Proses *data cleansing* bertujuan untuk memastikan bahwa data yang digunakan bebas dari kesalahan, ketidakkonsistestan, atau nilai-nilai yang tidak valid sehingga dapat meminimalkan terjadinya *error* selama analisis atau pemodelan yang akan dilakukan. Hasil dari proses pembersihan data ini menghasilkan dataset akhir yang terdiri dari 1200 baris dan 9 kolom yang terdiri dari UID, Tanggal, Waktu, *Vacuum Pressure*, *Air Pressure*, *Vibration*, *Temperatur Heater*, *Temperatur Proses*, *Rate*, dan *Kondisi Mesin*. Struktur data yang sudah bersih ini menjadi dasar penting dalam tahapan analisis dan pembangunan model prediktif yang lebih akurat.

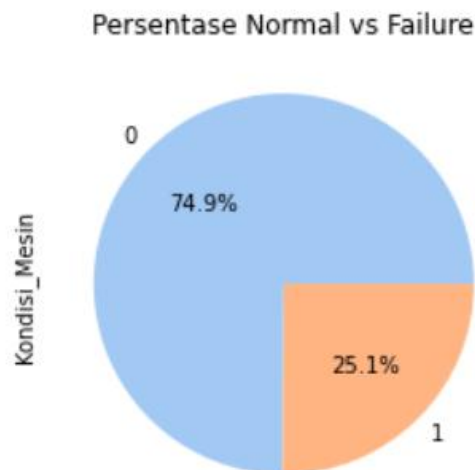
#### 4.2.1.2 Eliminasi Data

Tahap selanjutnya dalam proses *pre-processing* adalah melakukan eliminasi terhadap atribut-atribut data yang dianggap tidak relevan atau tidak diperlukan dalam proses pengolahan lebih lanjut. Tujuan dari eliminasi data ini adalah untuk menyederhanakan struktur data, mengurangi dimensi dataset, dan meningkatkan kinerja algoritma pembelajaran mesin yang akan digunakan, baik dari segi waktu komputasi maupun efisiensi pemrosesan. Atribut yang dieliminasi dipilih berdasarkan ketidakterkaitannya dengan tujuan analisis atau rendahnya kontribusi terhadap prediksi model, sehingga hanya fitur-fitur yang memiliki nilai informasi tinggi yang dipertahankan dalam dataset. Adapun data yang dieliminasi adalah sebagai berikut :

UID : Kode urutan penerimaan sensor yang ada pada mesin *vacuum forming*  
Tanggal : Tanggal penerimaan data sensor yang ada pada mesin *vacuum forming*  
Waktu : Waktu penerimaan data sensor yang ada pada mesin *vacuum forming*

#### 4.2.1.3 *Undersampling Data*

Tahap selanjutnya dalam proses *pre-processing* adalah melakukan *undersampling*. *Undersampling Data* adalah teknik penyeimbangan data yang dilakukan dengan cara mengurangi jumlah sampel pada kelas mayoritas. Tujuan dari proses *undersampling* adalah untuk mengatasi ketidakseimbangan distribusi kelas pada data yang digunakan yang akan berdampak pada model *machine learning* menjadi bias pada kelas yang jumlahnya dominan. Dengan menerapkan *undersampling*, distribusi data antar kelas dapat menjadi lebih seimbang, sehingga algoritma *machine learning* dapat menghasilkan model yang lebih akurat dan mampu mengenali pola dari kedua kelas secara optimal.



**Gambar 4. 9** Distribusi Kelas Pada Data

Gambar 4.9 menunjukkan distribusi kelas pada variabel *Kondisi\_Mesin* yang terbagi menjadi dua kategori, yaitu 0 untuk kondisi mesin normal dan 1 untuk kondisi mesin *failure* (gagal). Berdasarkan data tersebut terlihat bahwa sebanyak 74,9% dari data merupakan kondisi mesin normal sedangkan 25,1% sisanya merupakan kondisi mesin dalam keadaan *failure* (gagal). Ketimpangan distribusi ini mengindikasikan bahwa dataset tergolong *imbalanced*, di mana salah satu kelas (dalam hal ini kelas normal) memiliki

proporsi yang jauh lebih besar dibandingkan kelas failure sehingga diperlukan pendekatan *undersampling* untuk menyeimbangkan data sebelum model *machine learning* dilatih agar hasil prediksi lebih akurat.

```
from imblearn.under_sampling import RandomUnderSampler
from collections import Counter
import pandas as pd

# Misalnya x dan y sudah ada, dan x adalah DataFrame
print("Distribusi sebelum undersampling:", Counter(y))

# Buat objek undersampler
undersample = RandomUnderSampler(random_state=42)

# Lakukan undersampling
x_under, y_under = undersample.fit_resample(x, y)

print("Distribusi setelah undersampling:", Counter(y_under))

# Gabungkan kembali ke DataFrame
resampled_data = pd.concat([
    pd.DataFrame(x_under, columns=x.columns), # pastikan x masih DataFrame
    pd.DataFrame(y_under, columns=["Kondisi_Mesin"])
], axis=1)

# Simpan ke file CSV
resampled_data.to_csv("data_undersample_skripsi.csv", index=False)
```

**Gambar 4. 10** Undersampling Data

```
Distribusi sebelum undersampling: Counter({0: 899, 1: 301})
Distribusi setelah undersampling: Counter({0: 301, 1: 301})
```

**Gambar 4. 11** Hasil *Undersampling Data*

Setelah dilakukan undersampling, jumlah data pada kelas mayoritas 0 dikurangi menjadi 301, sehingga distribusi antar kelas menjadi seimbang, yaitu masing-masing sebanyak 301 data. Penyeimbangan ini bertujuan untuk menghindari bias model terhadap kelas mayoritas dan meningkatkan kemampuan model dalam mengenali pola dari kelas minoritas.

#### 4.2.2 Hasil *Pre-Processing Data*

*Pre-Processing Data* yang telah dilakukan membuat data sudah dapat digunakan untuk melakukan pengolahan data pembuatan model *machine learning*. Hasil dari *pre-processing data* yang telah dibuat akan disimpan dalam bentuk *file .csv* agar dapat mudah dilihat dan digunakan pada saat melakukan input data di *software Jupyterlab*. Berikut merupakan hasil dari *processing data*

**Tabel 4. 1** Hasil *Pre-Processing Data* yang telah dilakukan :

Vacuum Pressure	Air Pressure	Vibration	Temperatur Heater	Temperatur Proses	Rate	Kondisi Mesin
-89.733	6.097	0.011	186.516	91.375	61.938	0
-83.092	5.903	0.020	242.868	139.858	134.035	0
-87.724	5.985	0.014	150.575	120.968	137.713	0
-82.272	6.303	0.023	206.602	89.835	130.094	0
-83.704	6.066	0.015	168.057	96.894	130.883	0
-84.654	5.459	0.026	258.530	99.967	95.565	0
-88.594	5.786	0.023	261.503	148.945	85.698	0
-87.303	5.989	0.017	153.723	86.729	58.257	0
-80.347	5.973	0.021	169.020	126.333	53.981	0
-82.497	6.136	0.022	200.776	132.767	68.988	0
-88.429	5.902	0.022	151.064	95.334	53.896	0
-81.100	5.907	0.030	202.879	89.033	50.605	0
-82.984	5.929	0.021	172.712	129.988	135.793	0
-84.144	5.878	0.023	208.536	126.605	62.130	0
-87.055	6.237	0.017	265.926	113.968	69.649	0
-83.076	5.958	0.018	218.942	117.419	125.142	0
-86.877	5.798	0.016	195.176	111.769	70.406	0
-88.329	5.905	0.016	222.761	108.830	147.490	0
-90.336	5.814	0.023	251.595	106.947	93.014	0
-91.430	6.087	0.019	183.071	141.704	132.025	0

#### 4.3 Pengolahan Data

##### 4.3.1 Model *Machine Learning* Klasifikasi

Metode pada pengolahan data pada kali ini dilakukan dengan menggunakan model *machine learning* dengan menggunakan algoritma *Support Vector Machine*, *Random Forest*, dan *XGBoost*. Data yang digunakan pada pengolahan data adalah data yang sudah melalui *Pra-processing Data* sebelumnya. Data yang diperoleh akan dikelompokkan menjadi 2 variabel, yaitu variabel x independen dan variabel y dependen. Variabel x independen adalah *Vacuum\_Pressure*, *Air\_Pressure*, *Vibration*, *Temperatur\_Heater*, *Temperatur\_Proces*, dan *Rate*. Sedangkan untuk variabel y dependen adalah *Kondisi\_Mesin*.

### 4.3.1.1 Pembuatan Model *Machine Learning Support Vector Machine*

Berikut adalah langkah-langkah pembuatan model *machine learning Support Vector Machine* :

1. Data yang sudah diperoleh diolah menjadi dua, yaitu data *Training Set* dan *Test Set* dengan perbandingan 70% *Training Set* dan 30% *Test Set*
2. Membuat Model Klasifikasi *Support Vector Machine* menggunakan *library* yang sudah ada di *Jupyterlab*. Pada tahap ini variabel Y yaitu *Kondisi\_Mesin* melakukan prediksi terhadap variabel X yaitu *Vacuum\_Pressure*, *Air\_Pressure*, *Vibration*, *Temperatur\_Heater*, *Temperatur\_Proses*, dan *Rate*
3. Melakukan pembuatan model prediksi menggunakan kernel linear dan dilatih menggunakan *Test Set*

Berikut adalah hasil pembuatan model *machine learning* klasifikasi menggunakan algoritma *support vector machine* diaplikasi *Jupyterlab* :

**Tabel 4. 2** Hasil Model *Machine Learning* Klasifikasi *Support Vector Machine*

Vacuum Pressure	Air Pressure	Vibration	Temperatur Heater	Temperatur Proses	Rate	Kondisi Mesin	Prediksi
-89.733	6.097	0.011	186.516	91.375	61.938	0	1
-83.092	5.903	0.020	242.868	139.858	134.035	0	1
-79.186	6.208	0.025	281.909	98.537	149.433	1	1
-92.469	6.215	0.024	289.840	106.595	140.530	1	0
-85.359	6.316	0.016	218.945	137.526	104.543	0	0
-84.130	6.359	0.016	178.666	120.964	66.650	0	0
-84.601	5.758	0.030	219.590	104.971	94.066	1	0
-86.078	5.871	0.028	181.396	128.022	131.575	0	0
-89.479	6.239	0.013	163.351	124.955	108.244	0	0
-84.759	6.278	0.023	217.213	118.970	117.916	0	0
.....							
-77.362	6.098	0.018	164.607	125.555	83.632	1	1
-86.949	5.952	0.024	289.256	82.685	124.066	1	1
-91.678	6.103	0.016	280.158	109.463	126.875	1	0
-87.650	6.148	0.019	294.989	121.949	107.681	1	1
-90.313	6.114	0.018	206.747	104.545	125.128	0	0
-87.580	6.247	0.019	239.312	138.528	145.239	0	0
-87.123	6.120	0.020	284.439	114.581	112.023	1	1
-87.800	5.947	0.021	160.910	108.011	141.510	0	1
-79.212	5.846	0.028	205.574	142.811	130.998	1	1
-83.238	5.710	0.026	209.958	88.866	143.618	0	0

Berdasarkan Tabel Hasil *Model Machine Learning Klasifikasi Support Vector Machine* diatas dapat disimpulkan bahwa model tersebut berhasil melakukan prediksi dan dapat digunakan untuk melihat evaluasi *confusion matrix* untuk mengetahui akurasi model tersebut.

### 4.3.1.2 Pembuatan Model *Machine Learning Random Forest*

Berikut adalah langkah-langkah pembuatan model *machine learning Random Forest* :

1. Data yang sudah diperoleh diolah menjadi dua, yaitu data *Training Set* dan *Test Set* dengan perbandingan 70% *Training Set* dan 30% *Test Set*
2. Membuat Model Klasifikasi *Random Forest* menggunakan *library* yang sudah ada di *Jupyterlab*. Pada tahap ini variabel Y yaitu Kondisi\_Mesin melakukan prediksi terhadap variabel X yaitu *Vacuum\_Pressure*, *Air\_Pressure*, *Vibration*, *Temperatur\_Heater*, *Temperatur\_Proses*, dan *Rate*
3. Melakukan pembuatan model prediksi menggunakan parameter *default* dan dilatih menggunakan *Test Set*

Berikut adalah hasil pembuatan model *machine learning* klasifikasi menggunakan algoritma *random forest* diaplikasi *Jupyterlab* :

**Tabel 4. 3** Hasil Model *Machine Learning* Klasifikasi *Random Forest*

Vacuum Pressure	Air Pressure	Vibration	Temperatur Heater	Temperatur Proses	Rate	Kondisi Mesin	Prediksi
-89,733	6,097	0,114	186,516	91,375	61,938	0	1
-83,092	5,903	0,195	242,868	139,858	134,035	0	1
-79,186	6,208	0,025	281,909	98,537	149,433	1	1
-92,469	6,215	0,024	289,840	106,595	140,530	1	1
-85,359	6,316	0,016	218,945	137,526	104,543	0	0
-84,130	6,359	0,016	178,666	120,964	66,650	0	0
-84,601	5,758	0,030	219,590	104,971	94,066	1	1
-86,078	5,871	0,028	181,396	128,022	131,575	0	0
-89,479	6,239	0,013	163,351	124,955	108,244	0	0
-84,759	6,278	0,023	217,213	118,970	117,916	0	0
.....							
-77,362	6,098	0,018	164,607	125,555	83,632	1	1
-86,949	5,952	0,024	289,256	82,685	124,066	1	1
-91,678	6,103	0,016	280,158	109,463	126,875	1	1
-87,650	6,148	0,019	294,989	121,949	107,681	1	1
-90,313	6,114	0,018	206,747	104,545	125,128	0	0
-87,580	6,247	0,019	239,312	138,528	145,239	0	0
-87,123	6,120	0,020	284,439	114,581	112,023	1	1
-87,800	5,947	0,021	160,910	108,011	141,510	0	1
-79,212	5,846	0,028	205,574	142,811	130,998	1	1
-83,238	5,710	0,026	209,958	88,866	143,618	0	0

Berdasarkan Tabel Hasil *Model Machine Learning Klasifikasi Random Forest* diatas dapat disimpulkan bahwa model tersebut berhasil melakukan prediksi dan dapat digunakan untuk melihat evaluasi *confusion matrix* untuk mengetahui akurasi model tersebut.

### 4.3.1.3 Pembuatan Model *Machine Learning* XGBoost

Berikut adalah langkah-langkah pembuatan model *machine learning* XGBoost:

1. Data yang sudah diperoleh diolah menjadi dua, yaitu data *Training Set* dan *Test Set* dengan perbandingan 70% *Training Set* dan 30% *Test Set*
2. Membuat Model Klasifikasi XGBoost menggunakan *library* yang sudah ada di *Jupyterlab*. Pada tahap ini variabel Y yaitu Kondisi\_Mesin melakukan prediksi terhadap variabel X yaitu *Vacuum\_Pressure*, *Air\_Pressure*, *Vibration*, *Temperatur\_Heater*, *Temperatur\_Proses*, dan *Rate*
3. Melakukan pembuatan model prediksi menggunakan *boosting gradient* dan dilatih menggunakan *Test Set*

Berikut adalah hasil pembuatan model *machine learning* klasifikasi menggunakan algoritma xgboost diaplikasi *Jupyterlab* :

**Tabel 4. 4** Hasil *Model Machine Learning* Klasifikasi XGBoost

Vacuum Pressure	Air Pressure	Vibration	Temperatur Heater	Temperatur Proses	Rate	Kondisi Mesin	Prediksi
-89,733	6,097	0,114	186,516	91,375	61,938	0	1
-83,092	5,903	0,195	242,868	139,858	134,035	0	1
-79,186	6,208	0,025	281,909	98,537	149,433	1	1
-92,469	6,215	0,024	289,840	106,595	140,530	1	1
-85,359	6,316	0,016	218,945	137,526	104,543	0	0
-84,130	6,359	0,016	178,666	120,964	66,650	0	0
-84,601	5,758	0,030	219,590	104,971	94,066	1	0
-86,078	5,871	0,028	181,396	128,022	131,575	0	0
-89,479	6,239	0,013	163,351	124,955	108,244	0	0
-84,759	6,278	0,023	217,213	118,970	117,916	0	0
.....							
-77,362	6,098	0,018	164,607	125,555	83,632	1	1
-86,949	5,952	0,024	289,256	82,685	124,066	1	1
-91,678	6,103	0,016	280,158	109,463	126,875	1	1
-87,650	6,148	0,019	294,989	121,949	107,681	1	1
-90,313	6,114	0,018	206,747	104,545	125,128	0	0
-87,580	6,247	0,019	239,312	138,528	145,239	0	0
-87,123	6,120	0,020	284,439	114,581	112,023	1	1
-87,800	5,947	0,021	160,910	108,011	141,510	0	1
-79,212	5,846	0,028	205,574	142,811	130,998	1	1
-83,238	5,710	0,026	209,958	88,866	143,618	0	0

Berdasarkan Tabel Hasil *Model Machine Learning* XGBoost diatas dapat disimpulkan bahwa model tersebut berhasil melakukan prediksi dan dapat digunakan untuk melihat evaluasi *confusion matrix* untuk mengetahui akurasi model tersebut.

#### 4.4 Evaluasi Model *Machine Learning*

##### 4.4.1 *Confusion Matrix Support Vector Machine*

Berikut adalah hasil *confusion matrix Support Vector Machine* yang telah didapatkan :

	precision	recall	f1-score	support
0	0.83	0.65	0.73	100
1	0.66	0.84	0.74	81
accuracy			0.73	181
macro avg	0.75	0.74	0.73	181
weighted avg	0.76	0.73	0.73	181

Akurasi SVM : 73.48%

**Gambar 4. 12** Perhitungan *Confusion Matrix Support Vector Machine*

Berdasarkan gambar Perhitungan *Confusion Matrix Support Vector Machine* diatas dapat dilihat bahwa hasil nilai *precision*, *recall*, dan *f1-score*. Berdasarkan gambar diatas dapat diketahui bahwa nilai akurasi sebesar 73,48%, nilai *precision* pada kelas 0 sebesar 83% dan kelas 1 sebesar 66%, nilai *recall* pada kelas kelas 0 sebesar 65% dan kelas 1 sebesar 84%, dan nilai *f1-score* pada kelas 0 sebesar 73% dan kelas 1 sebesar 74%

##### 4.4.2 *Confusion Matrix Support Random Forest*

Berikut adalah hasil *confusion matrix Support Vector Machine* yang telah didapatkan :

	precision	recall	f1-score	support
0	0.99	0.98	0.98	100
1	0.98	0.99	0.98	81
accuracy			0.98	181
macro avg	0.98	0.98	0.98	181
weighted avg	0.98	0.98	0.98	181

Akurasi Random Forest : 98.34%

**Gambar 4. 13** Perhitungan *Confusion Matrix Random Forest*



Berdasarkan gambar Perhitungan *Confusion Matrix Random Forest* diatas dapat dilihat bahwa hasil nilai *precision*, *recall*, dan *f1-score*. Berdasarkan gambar diatas dapat diketahui bahwa nilai akurasi sebesar 98,34%, nilai *precision* pada kelas 0 dan kelas 1 sebesar 98%, nilai *recall* pada kelas 0 dan kelas 1 sebesar 98%, dan nilai *f1-score* pada kelas 0 dan kelas 1 sebesar 98%

#### 4.4.3 *Confusion Matrix XGBoost*

Berikut adalah hasil *confusion matrix* XGBoost yang telah didapatkan :

Akurasi XGBoost : 97.22%

Classification Report:

	precision	recall	f1-score	support
0	0.97	0.97	0.97	35
1	0.97	0.97	0.97	37
accuracy			0.97	72
macro avg	0.97	0.97	0.97	72
weighted avg	0.97	0.97	0.97	72

**Gambar 4. 14** Perhitungan *Confusion Matrix XGBoost*

Berdasarkan gambar Perhitungan *Confusion Matrix XGBoost* diatas dapat dilihat bahwa hasil nilai *precision*, *recall*, dan *f1-score*. Berdasarkan gambar diatas dapat diketahui bahwa nilai akurasi sebesar 97,22%, nilai *precision* pada kelas 0 dan kelas 1 sebesar 97%, nilai *recall* kelas 0 dan kelas 1 sebesar 97%, dan nilai *f1-score* kelas 0 dan kelas 1 sebesar 97%.

#### 4.4.4 Rekapitulasi Perhitungan *Confusion Matrix*

Berikut adalah hasil rekapitulasi perhitungan *confusion matrix* yang telah dibuat :

**Tabel 4. 5** Rekapitulasi Perhitungan *Confusion Matrix*

<b>Model Machine Learning</b>	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>F1-Score</b>
<i>Support Vector Machine</i>	73%	75%	74%	73%
<i>Random Forest</i>	98%	98%	98%	98%
XGBoost	97%	97%	97%	97%

Berdasarkan tabel rekapitulasi perhitungan *confusion matrix* diatas, dapat disimpulkan bahwa model *machine learning* klasifikasi *support vector machine*

memiliki nilai akurasi sebesar 73%, nilai *precision* sebesar 75%, nilai *recall* sebesar 74% dan nilai *f1-score* sebesar 73%. Model *machine learning* klasifikasi *random forest* memiliki nilai akurasi sebesar 98%, nilai *precision* sebesar 98%, nilai *recall* sebesar 98% dan nilai *f1-score* sebesar 98%. Model *machine learning* klasifikasi XGBoost memiliki nilai akurasi sebesar 97%, nilai *precision* sebesar 97%, nilai *recall* sebesar 97% dan nilai *f1-score* sebesar 97%.

Berdasarkan rekapitulasi yang sudah dilakukan dapat dilihat bahwa performa SVM yang lebih rendah daripada model *machine learning* lainnya. Hal tersebut terjadi karena algoritma ini memiliki keterbatasan dalam mengenali pola data yang bersifat non-linear. Pada mesin *vacuum forming*, data sensor yang digunakan memiliki variasi yang cukup kompleks sehingga penggunaan kernel linear pada SVM kurang mampu memetakan hubungan antar fitur dengan baik. Selain itu, SVM juga sensitif terhadap distribusi data yang tidak seimbang. Walaupun sudah dilakukan teknik *undersampling* maupun *oversampling*, hasil prediksi SVM masih cenderung bias ke kelas mayoritas. Hal ini berbeda dengan *Random Forest* dan XGBoost yang berbasis *ensemble learning* dengan *decision tree*, sehingga lebih adaptif terhadap pola data non-linear serta lebih tahan terhadap ketidakseimbangan data. Oleh karena itu, kedua algoritma tersebut menunjukkan kinerja yang lebih unggul dibandingkan SVM

Oleh karena itu, dapat disimpulkan bahwa model *machine learning* klasifikasi *Random Forest* adalah model terbaik karena memiliki nilai *accuracy*, *precision*, *recall*, dan *f1-score* paling tinggi diantara model *machine learning* klasifikasi lainnya.

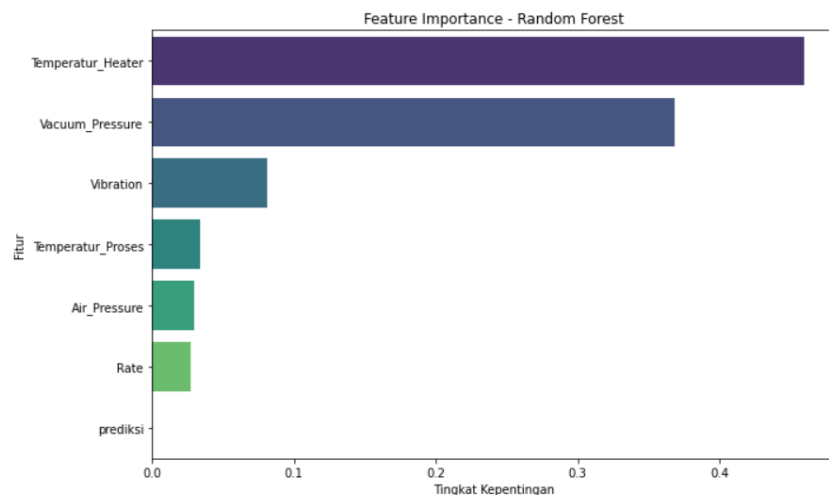
#### **4.5 Analisis Data *Feature Importance***

*Feature Importance* digunakan untuk mengetahui seberapa besar pengaruh masing-masing fitur yang ada pada sebuah *dataset* yang digunakan terhadap hasil prediksi model *machine learning*. Model *machine learning* yang terbaik berdasarkan hasil yang sudah didapatkan yaitu model *Random Forest*.

Berikut merupakan visualisasi yang telah didapatkan menggunakan *feature importance* yang ada pada *library Jupyterlab* :

Feature	Importance
Temperatur_Heater	0.459344
Vacuum_Pressure	0.368506
Vibration	0.081127
Temperatur_Proses	0.033871
Air_Pressure	0.029603
Rate	0.027549
prediksi	0.000000

**Gambar 4. 15** Hasil *Feature Importance Random Forest*



**Gambar 4. 16** Visualisasi *Feature Importance Random Forest*

Berdasarkan hasil visualisasi *feature importance* yang ditampilkan dapat diketahui bahwa 3 variabel yang paling penting dan memengaruhi hasil prediksi terdiri dari *Temperatur\_Heater* sebesar 0,45, *Vacuum\_Pressure* sebesar 0,36, dan *Vibration* sebesar 0,08. Hal tersebut dapat disimpulkan bahwa model mengandalkan data dari pemanasan mesin dan tekanan vakum sebagai indikator utama dalam membedakan antara kondisi mesin yang normal dan yang berpotensi mengalami kegagalan. Tingginya pengaruh dari *Temperatur\_Heater* bisa mengindikasikan bahwa fluktuasi temperatur selama proses produksi memiliki hubungan kuat terhadap performa atau keausan mesin.

Dengan fokus pada fitur yang paling berpengaruh, perusahaan dapat mengembangkan strategi perawatan yang lebih efisien, misalnya dengan menambahkan sensor pemantauan temperatur *heater* atau mengatur ulang parameter tekanan agar tetap optimal. Pendekatan ini dapat meningkatkan efektivitas *predictive maintenance* dan mengurangi risiko *downtime* yang tidak diinginkan.

#### 4.6 Analisis Biaya Penggunaan *Predictive Maintenance*

Analisis biaya ini digunakan untuk menghitung jumlah estimasi pengeluaran dan efisiensi biaya yang diperoleh PT. Jinwoo Engineering Indonesia dalam menerapkan sistem *predictive maintenance*. Data didapat dari hasil wawancara operator dan kepala *maintenance* mesin *vacuum forming*.

##### 4.6.1 Biaya *Maintenance* Konvensional PT. Jinwoo Engineering Indonesia

Berikut adalah komponen biaya *maintenance* konvensional yang sekarang digunakan oleh PT. Jinwoo Engineering Indonesia untuk mesin *vacuum forming*:

**Tabel 4. 6** Biaya *Maintenance* Konvensional Mesin *Vacuum Forming*

Komponen Biaya	Jumlah Unit/Frekuensi	Biaya per Unit (Rp)	Total per Tahun
Tenaga teknisi perbaikan (kerusakan)	12 kali / tahun	1.000.000	12.000.000
Penggantian suku cadang	12 kali	2.000.000	24.000.000
Downtime produksi (8 jam x 12 kejadian)	96 jam	500.000 / jam	48.000.000
Inspeksi berkala preventif	1x per bulan	500.000	6.000.000
<b>Total Biaya <i>Maintenance</i> Konvensional</b>			90.000.000

Biaya yang dikeluarkan untuk *maintenance* konvensional dibagi menjadi 4 komponen biaya, yaitu tenaga teknisi perbaikan, penggantian suku cadang, *downtime* produksi, dan inspeksi preventif secara berkala.

Biaya perbaikan oleh teknisi dan penggantian suku cadang, masing-masing dengan frekuensi 12 kali per tahun, menghasilkan total biaya sebesar Rp12.000.000,00 dan Rp24.000.000,00. Biaya *downtime* produksi selama 96 jam total (8 jam x 12 kejadian) mencapai Rp48.000.000,00. Terakhir, inspeksi berkala preventif sebanyak 12 kali per tahun memakan biaya Rp6.000.000,00. Dengan demikian, total biaya *maintenance* konvensional yang tercatat adalah sebesar Rp90.000.000,00.

#### 4.6.2 Biaya Implementasi Sistem *Predictive Maintenance*

Berikut adalah komponen biaya implementasi *predictive maintenance* PT. Jinwoo Engineering Indonesia untuk mesin *vacuum forming*:

**Tabel 4. 7** Biaya Implementasi Awal Sistem *Predictive Maintenance*

<b>Komponen Biaya</b>	<b>Biaya (Rp)</b>	<b>Keterangan</b>
Pengadaan sensor tambahan integrasi <i>machine learning</i>	15.000.000	Pengadaan sensor integrasi <i>machine learning</i> untuk sensor suhu, tekanan, dan getaran
Biaya pengembangan model <i>machine learning</i>	20.000.000	Data <i>scientist</i> dan pengujian model
Perangkat integrasi <i>machine learning</i> (komputer)	10.000.000	Proses lokal prediksi
Pelatihan teknisi dan operator	5.000.000	1 kali pelatihan internal
Biaya <i>maintenance</i> sistem dan <i>software</i> tahunan	7.000.000	Untuk <i>upgrade</i> dan <i>support</i> teknis
<b>Total Biaya Implementasi</b>		57.000.000

Biaya yang dikeluarkan untuk implementasi awal system *predictive maintenance* dibagi menjadi 5 komponen biaya, yaitu pengadaan sensor tambahan integrasi *machine learning*, pengembangan model *machine learning*, perangkat integrasi *machine learning*, pelatihan teknisi dan operator, dan *maintenance* untuk system dan *software* tahunan.

Pengadaan sensor tambahan integrasi *machine learning* membutuhkan biaya awal sebesar Rp15.000.000 untuk menyediakan sensor yang terintegrasi pada sensor suhu, tekanan, dan getaran. Pengembangan model *machine learning* memakan biaya sebesar Rp20.000.000 untuk melakukan pengolahan data dan pengujian model *machine learning*. Perangkat integrasi *machine learning* membutuhkan biaya sebesar Rp10.000.000 untuk melakukan proses lokal prediksi. Pelatihan teknisi dan operator membutuhkan biaya sebesar Rp5.000.000 untuk melakukan pelatihan internal sistem *predictive maintenance* bagi teknisi dan operator. *Maintenance* sistem dan *software* membutuhkan biaya sebesar Rp7.000.000 untuk *upgrade* dan *support* teknis. Dengan demikian, total implementasi awal sistem *predictive maintenance* adalah sebesar Rp57.000.000

#### 4.6.3 Efisiensi Penerapan Sistem *Predictive Maintenance*

Berikut adalah komponen yang dihemat dengan menerapkan sistem *predictive maintenance* yang dilakukan oleh PT. Jinwoo Engineering Indonesia untuk mesin *vacuum forming*:

**Tabel 4. 8** Efisiensi Setelah Memakai *Predictive Maintenance*

Komponen yang Dihemat	Biaya Sebelumnya (Rp)	Estimasi Hemat (%)	Biaya Baru (Rp)	Penghematan (Rp)
Biaya <i>downtime</i> produksi	48.000.000	70%	14.400.000	33.600.000
Biaya penggantian suku cadang	24.000.000	50%	12.000.000	12.000.000
Biaya teknisi perbaikan	12.000.000	50%	6.000.000	6.000.000
Biaya inspeksi berkala	6.000.000	50%	3.000.000	3.000.000
<b>Total Penghematan per Tahun</b>				<b>54.600.000</b>

Biaya yang dihemat dalam implementasi sistem *predictive maintenance* dibagi menjadi 4 komponen biaya, yaitu biaya *downtime* produksi, biaya penggantian suku cadang, biaya teknisi perbaikan, dan biaya inspeksi berkala.

Biaya *downtime* produksi yang sebelumnya sebesar Rp48.000.000 dengan menggunakan estimasi efisien sistem *predictive* dihemat menjadi hanya sebesar Rp14.400.000. Biaya penggantian suku cadang yang sebelumnya sebesar Rp24.000.000 dengan menggunakan estimasi efisien sistem *predictive* dihemat menjadi hanya sebesar Rp12.000.000. Biaya teknisi perbaikan yang sebelumnya sebesar Rp12.000.000 dengan menggunakan estimasi efisien sistem *predictive* dihemat menjadi hanya sebesar Rp6.000.000. Biaya inspeksi berkala yang sebelumnya sebesar Rp6.000.000 dengan menggunakan estimasi efisien sistem *predictive* dihemat menjadi hanya sebesar Rp3.000.000. Dengan demikian, total penghematan setelah implementasi sistem *predictive maintenance* adalah sebesar Rp54.600.000

#### 4.6.4 Ringkasan Analisis Biaya

Berikut adalah ringkasan analisis biaya penerapan sistem *predictive maintenance* yang dilakukan oleh PT. Jinwoo Engineering Indonesia untuk mesin *vacuum forming*:

**Tabel 4. 9** Ringkasan Analisis Biaya Penerapan *Predictive Maintenance*

<b>Item</b>	<b>Nilai (Rp)</b>
Biaya Konvensional (per tahun)	90.000.000
Biaya Implementasi (tahun pertama)	57.000.000
Penghematan per tahun setelah implementasi	54.600.000
<b><i>Payback Period</i></b>	~1 tahun

Biaya *maintenance* aktual yang dikeluarkan per tahun adalah sebesar Rp90.000.000,00. Sedangkan biaya implementasi awal sistem *predictive maintenance* adalah sebesar Rp57.000.000. Dengan perhitungan yang ada, penghematan atau efisiensi yang didapat adalah sebesar Rp54.600.000

Dengan demikian, berdasarkan perhitungan yang ada, periode pengembalian modal (*payback period*) dari investasi implementasi tersebut diperkirakan selama satu tahun. Hal ini menunjukkan bahwa investasi tersebut memiliki tingkat pengembalian yang cepat, sehingga dapat menjadi pilihan yang menguntungkan bagi perusahaan.