DAFTAR PUSTAKA

[1] Enrico Dini, Patricia Ricard, Sophie Roux. "*Ai-Driven Predictive Maintenance For Industrial Machinery In Indonesian Manufacturing Sectors*". *The International Conferences on Engineering Sciences* Volume. 1, No. 1, 2024

[2] Krisman Yusuf Nazara. "Perancangan *Smart Predictive Maintenance* untuk Mesin Produksi" Seminar Nasional Official Statistics, 2022

[3] Mobley, R. K. "*An Introduction to Predictive Maintenance"*. Elsevier, 2002

[4] Marek Moleda et al., "*From Corrective to Predictive Maintenance—A Review of Maintenance Approaches for the Power Industry*". *Sensors, 23, 5970.* 2023

[5] Yassirullah Apta Saputra, Nurillah Jamil Achamawati Novel. "STRATEGI PEMELIHARAAN ASET PROPERTI DENGAN PENDEKATAN KOREKTIF DAN PREVENTIF PADA PT. JASWITA JABAR (PERSERODA)". Jurnal Manajemen Aset dan Penilaian Vol. 4 No. 1. 2024

[6] Hafiz Azhari, Jimmy Greei Ganap, Safitri Ristanti, Firda Ainun Nisah. "Analisis Perawatan Mesin Kapal dengan Metode Reliability Centered Maintenance (RCM) di PT Jasa Armada Indonesia Tbk". Industrika Jurnal Ilmiah Teknik Industri Volume 8, No. 2, April 2024

[7] Muhammad Suprapto. "*Preventive Maintenance Strategies to Increase Industrial Machine Life"*. *Journal of Renewable Engineering* Vol.2. No. 2 April 2025

[8] Tom Mithcell. "*Machine Learning"*. McGraw-Hill, 1997

[9] Agung Wijoyo, Asep Yudistira Saputra, Safitri Ristanti, Sultan Rafly Sya'Ban, Mila Amalia, Randi Febriansyah. "Pembelajaran *Machine Learning*". OKTAL : Jurnal Ilmu Komputer dan *Science* Volume 3, No. 2, Februari 2024

[10] Sindy Caesar Rusnanda Himatul Haliza, Anita Qoiriah. "*Predictive Maintenance* untuk Kendaraan Bermotor dengan Menggunakan *Support Vector Machine* (SVM)". JINACS *(Journal of Informatics and Computer Science):* Volume 02 Nomor 03, 2021

[11] Matthew Wiens et al., "*A Tutorial and Use Case Example of the eXtreme Gradient Boosting (XGBoost) Artificial Intelligence Algorithm for Drug Development Applications*". *Clinical and Translational Science*, 2025

[12] Narayanan, K., & Manimaran, A. "*Using Decision Risk and Decision Accuracy Metrics for Decision Making for Remote Sensing and GIS Applications"* (pp. 125–136). *Springer Nature*, 2024.

[13] Mulkan Azhari, Zakaria Situmorang, Rika Rosnelly. "Perbandingan Akurasi, Recall, dan Presisi Klasifikasi pada Algoritma C4.5, Random Forest, SVM dan Naive Bayes*"* Jurnal Media Informatika Budidarma Volume 5, Nomor 2, April 2021.

[14] Guntoro, Haryati. "Perbandingan Algoritma Naive Bayes Dan K-Nearest Neighbor (K-NN) Untuk Klasifikasi Penyakit Gagal Jantung *Journal of Information System Research (JOSH), 6(2), 997-1005.*, 2021.

[15] Hiromasa Kaneko. "*Interpretation of Machine Learning Models for Data Sets with Many Features Using Feature Importance*". ACS Omega, 2023

LAMPIRAN

```
import pandas as pd
import numpy as np
import json
import seaborn as sns
import matplotlib.pyplot as plt

#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)

# Konversi tipe data tanggal dan waktu
df['Tanggal'] = pd.to_datetime(df['Tanggal'], errors='coerce')
df['Waktu'] = pd.to_datetime(df['Waktu'], format='%H:%M', errors='coerce').dt.time

# Hapus baris dengan nilai kosong
df.dropna(inplace=True)

# Ambil 1200 baris secara acak jika data lebih banyak
if len(df) > 1200:
    df = df.sample(n=1200, random_state=42)

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))
```

Lampiran 1 Proses *Cleaning Data*

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

Lampiran 2 *Import Library Support Vector Machine*

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

Lampiran 3 *Import Library Random Forest*

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV, RandomizedSearchCV, KFold
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.ensemble import RandomForestClassifier
import xgboost as XGBClassifier
```

Lampiran 4 *Import Library XGBoost*

```
# 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)
```

Lampiran 5 Identifikasi Data NA

```
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)
```

Lampiran 6 Proses *Undersample Data*

```
scaler = StandardScaler()
scaler.fit(x)
x = scaler.transform(x)
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=42)
```

```
clf = SVC(kernel="linear")
clf.fit(x_train, y_train)

y_pred = clf.predict(x_test)
CLF_acc = accuracy_score(y_pred, y_test)

print(classification_report(y_test, y_pred))
print("Akurasi SVM : {:.2f}%".format(CLF_acc * 100))
```

Lampiran 7 Pembuatan Model *Support Vector Machines*

```
scaler = StandardScaler()
scaler.fit(x)
x = scaler.transform(x)
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=42)
```

```
rlf = RandomForestClassifier()
rlf.fit(x_train, y_train)

y_pred = rlf.predict(x_test)
RLF_acc = accuracy_score(y_pred, y_test)

print(classification_report(y_test, y_pred))
print("Akurasi Random Forest : {:.2f}%".format(RLF_acc * 100))
```

Lampiran 8 Pembuatan Model *Random Forest*

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=42)
x_train, x_val, y_train, y_val = train_test_split(x_train, y_train, test_size=0.3, random_stat
```

```
label_encoder = LabelEncoder()
y_train= label_encoder.fit_transform(y_train)
y_val= label_encoder.transform(y_val)
y_test= label_encoder.transform(y_test)
```

```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_val = scaler.transform(x_val)
x_test = scaler.transform(x_test)
```

```
from xgboost import XGBClassifier

xgb_model = XGBClassifier(random_state=42)

xgb_model.fit(x_train, y_train)
y_pred = xgb_model.predict(x_val)

from sklearn.metrics import accuracy_score
test_accuracy = accuracy_score(y_val, y_pred)

print("Akurasi XGBoost : {:.2f}%".format(test_accuracy * 100))

print("\nClassification Report:\n", classification_report(y_val, y_pred))
```

Lampiran 9 Pembuatan Model *XGBoost*

```python
# Import library yang dibutuhkan
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns


# Pisahkan fitur (X) dan target (y)
X = df.drop('Kondisi_Mesin', axis=1)  # Variabel independen
y = df['Kondisi_Mesin']               # Variabel dependen

# Skala fitur agar performa model optimal
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split data ke dalam data latih dan data uji
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Buat dan latih model Random Forest
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Ambil nilai feature importance
importances = model.feature_importances_
feature_names = X.columns
feature_importance_df = pd.DataFrame({'Feature': feature_names, 'Importance': importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

# Tampilkan tabel
print(feature_importance_df)

# Plot feature importance
plt.figure(figsize=(10,6))
sns.barplot(x='Importance', y='Feature', data=feature_importance_df, palette='viridis')
plt.title('Feature Importance - Random Forest')
plt.xlabel('Tingkat Kepentingan')
plt.ylabel('Fitur')
plt.tight_layout()
plt.show()
```

Lampiran 10 *Feature Importance Random Forest*