

**Judul Proyek** : **Prediksi Suar Matahari Menggunakan Model Klasifikasi Machine Learning dan Deep Learning**

**Nama** : Erlina Putri Rahmadhani

**NIM** : 233307046

**Program Studi** : Teknologi Informasi

**Mata Kuliah** : Data Science

**Dosen Pengampu** : Gus Nanang Syaifuddiin, S.Kom., M.Kom

**Tahun Akademik** : 2025/5

**Link GitHub Repostory** : [https://github.com/erlinaputrirahmadhani05/SolarFlare\\_Project\\_UAS](https://github.com/erlinaputrirahmadhani05/SolarFlare_Project_UAS)

**Link Video Pembahasan** : <https://youtu.be/05b6lwdGAOA>

## 1. LEARNING OUTCOMES

Pada proyek ini, mahasiswa diharapkan dapat:

1. Memahami konteks masalah dan merumuskan problem statement secara jelas
2. Melakukan analisis dan eksplorasi data (EDA) secara komprehensif (**OPSIONAL**)
3. Melakukan data preparation yang sesuai dengan karakteristik dataset
4. Mengembangkan tiga model machine learning yang terdiri dari (**WAJIB**):
  - o Model baseline
  - o Model machine learning / advanced
  - o Model deep learning (**WAJIB**)
5. Menggunakan metrik evaluasi yang relevan dengan jenis tugas ML
6. Melaporkan hasil eksperimen secara ilmiah dan sistematis
7. Mengunggah seluruh kode proyek ke GitHub (**WAJIB**)
8. Menerapkan prinsip software engineering dalam pengembangan proyek

## 2. PROJECT OVERVIEW

### 2.1 Latar Belakang

Fenomena Suar Matahari (Solar Flare) kuat (Kelas M dan X) merupakan salah satu kejadian paling signifikan dalam ranah Cuaca Angkasa (Space Weather). Proyek ini bertujuan untuk mengembangkan sistem prediksi dengan menggunakan Machine Learning untuk mengklasifikasikan kemungkinan terjadinya Strong Flare berdasarkan data bintik tinggi yang dilepaskan oleh flare dapat memicu Geomagnetically Induced Currents, yang berpotensi merusak jaringan listrik dan sistem pipa. Oleh karena itu, meningkatkan akurasi Recall (kemampuan deteksi) flare yang kuat dapat memberikan waktu yang krusial untuk mitigasi mengurangi potensi kerugian ekonomi dan risiko keamanan.

Pemodelan fenomena ini menghadapi tantangan utama, yaitu ketidakseimbangan Kelas (Class Imbalance). Kejadian Strong Flare sangat jarang terjadi dibandingkan dengan periode tenang, menyebabkan model Machine Learning konvensional cenderung bias memprediksi kelas mayoritas (tidak ada flare), yang berujung pada tingginya angka False Negative (FN) gagal mendeteksi flare yang sebenarnya terjadi. Selain masalah imbalance, hubungan antara fitur bintang matahari yang terukur dengan aktivitas flare bersifat sangat non-linear, yang sulit ditangkap oleh model statistik tradisional. Untuk mengatasi permasalahan tersebut, proyek ini menerapkan dan membandingkan algoritma klasifikasi yang canggih, meliputi model Baseline, model Advanced, dan Deep Learning. Pendekatan machine learning terbukti dalam menangani observasi bervolume tinggi dan memodelkan hubungan non-linear.

Referensi:

- Camporeale, E., A. H. B. M. L. W. (2018). Machine Learning in Space Weather. Elsevier.
- Liu, M., Zhang, B., Shen, C., & Zhang, J. (2017). Solar Flare Prediction Using Deep Learning. The Astrophysical Journal, 843(2), 104.

### **3. BUSINESS UNDESTANDING / PROBLEM UNDERSTANDING**

#### **3.1 Problem Statements**

- Sulit memprediksi terjadinya solar flare berdasarkan parameter aktivitas matahari
- Dataset solar flare memiliki fitur kategorikal yang perlu preprocessing khusus
- Dibutuhkan model machine learning yang mampu menangkap pola non-linear
- Model deep learning diperlukan untuk mempelajari hubungan kompleks antar fitur

#### **3.2 Goals**

- Membangun model klasifikasi solar flare dengan akurasi minimal 80%
- Membandingkan performa baseline, ML, dan deep learning
- Menentukan model terbaik berdasarkan metrik evaluasi
- Menghasilkan model yang dapat direproduksi

#### **3.3 Solution Approach**

##### **a) Model 1 – Baseline Model: Logistic Regresion**

Deskripsi singkat: Logistic Regression merupakan model klasifikasi linear yang bekerja dengan menghitung probabilitas suatu data termasuk ke dalam kelas tertentu menggunakan fungsi sigmoid

Alasan: memiliki kesederhanaan dan interpretatif sehingga mudah digunakan sebagai pembanding awal, cocok untuk dataset berukuran kecil hingga menengah seperti solar flare, mendukung klasifikasi multikelas sesuai dengan target Low, Medium, dan High.

##### **b) Model 2 – Advanced / ML Model: Random Forest**

Deskripsi singkat: Random Forest adalah metode ensemble berbasis Decision Tree yang membangun banyak pohon keputusan dan menggabungkan hasil prediksinya untuk meningkatkan akurasi dan mengurangi overfitting

Alasan: mampu menangkap hubungan non-linear antar fitur yang tidak dapat ditangani oleh model linear, tahan terhadap noise dan outlier, dapat bekerja dengan baik pada data tabular, memberikan akurasi yang lebih tinggi dibandingkan baseline model.

**c) Model 3 –Deep Learning Model: Multilayer Perception (MLP)**

Deskripsi singkat: Multilayer Perceptron (MLP) merupakan jaringan saraf tiruan feedforward yang terdiri dari beberapa hidden layer. Model ini mampu mempelajari pola kompleks melalui proses backpropagation

Alasan: dataset Solar Flare merupakan data tabular sehingga MLP arsitektur deep learning yang paling sesuai, mampu mempelajari pola non-linear yang kompleks lebih baik dibanding model machine learning, serta mendukung learning konvensional.

## 4. DATA UNDERSTANDING

### 4.1 Informasi Dataset

Dataset Solar Flare berisi data aktivitas matahari dan target berupa jumlah atau kategori solar flare.

- Sumber: UCI Machine Learning Repository
- Link: <https://archive.ics.uci.edu/dataset/89/solar+flare>
- Deskripsi Dataset:
  - a. Jumlah baris: 1389
  - b. Jumlah kolom: 12
  - c. Tipe data: categorical (nominal & ordinal), integer (numerik diskrit)
  - d. Ukuran dataset: 90KB
  - e. Format file: .data

### 4.2 Dekripsi Fitur

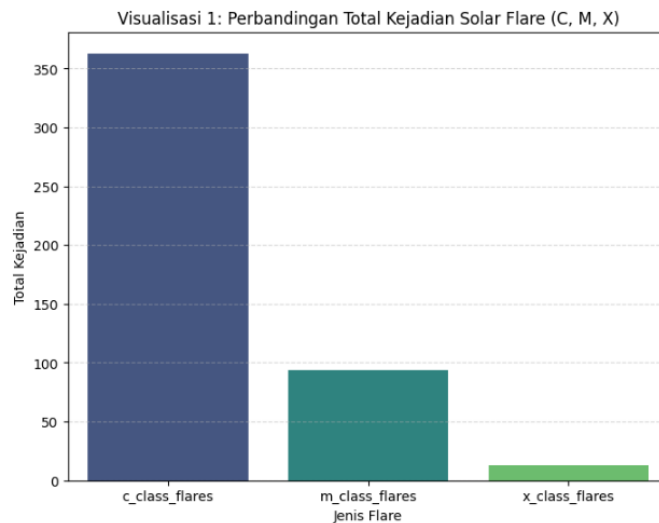
Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
code	Categorical	Kode klasifikasi region matahari	A, B, C
largest_spot_size	Categorical	Ukuran terbesar sunspot pada region	S, A, H
spot_distribution	Categorical	Pola distribusi sunspot	X, O, I
activity	Categorical	Tingkat aktivitas region matahari	0, 1
evolution	Categorical	Perkembangan region matahari	D, E, F
previous_activity	Categorical	Aktivitas flare sebelumnya	0, 1
complexity	Categorical	Kompleksitas magnetik region matahari	B, C, D
area	Categorical	Luas area sunspot	1, 2, 3
largest_spot	Categorical	Ukuran sunspot terbesar	1, 2
c_class_flare	Integer	Jumlah flare kelas C dalam 24 jam	0, 1, 2
m_class_flare	Integer	Jumlah flare kelas M dalam 24 jam	0, 1
x_class_flare	Integer	Jumlah flare kelas X dalam 24 jam	0, 1

### 4.3 Kondisi Data

- Missing Values: tidak ada, seluruh fitur terisi lengkap tanpa nilai kosong
- Duplicate Data: tidak ada, yang menunjukkan bahwa setiap baris data bersifat unik dan tidak terdapat pengulangan data.
- Outlier: tidak ditemukan outlier ekstrem, fitur terkait `c_class_flare`, `m_class_flare`, `x_class_flare`
- Imbalanced data: ada, pada kelas Low:  $\pm 58.7\%$ , Medium:  $\pm 28.8\%$ , High:  $\pm 12.6\%$
- Noise: tidak signifikan, tidak ditemukan indikasi noise yang bersifat ekstrem atau kesalahan pencatatan data.
- Data quality issues: sebagian besar fitur bersifat kategorikal dan belum dalam numerik, distribusi data numerik tidak seimbang.

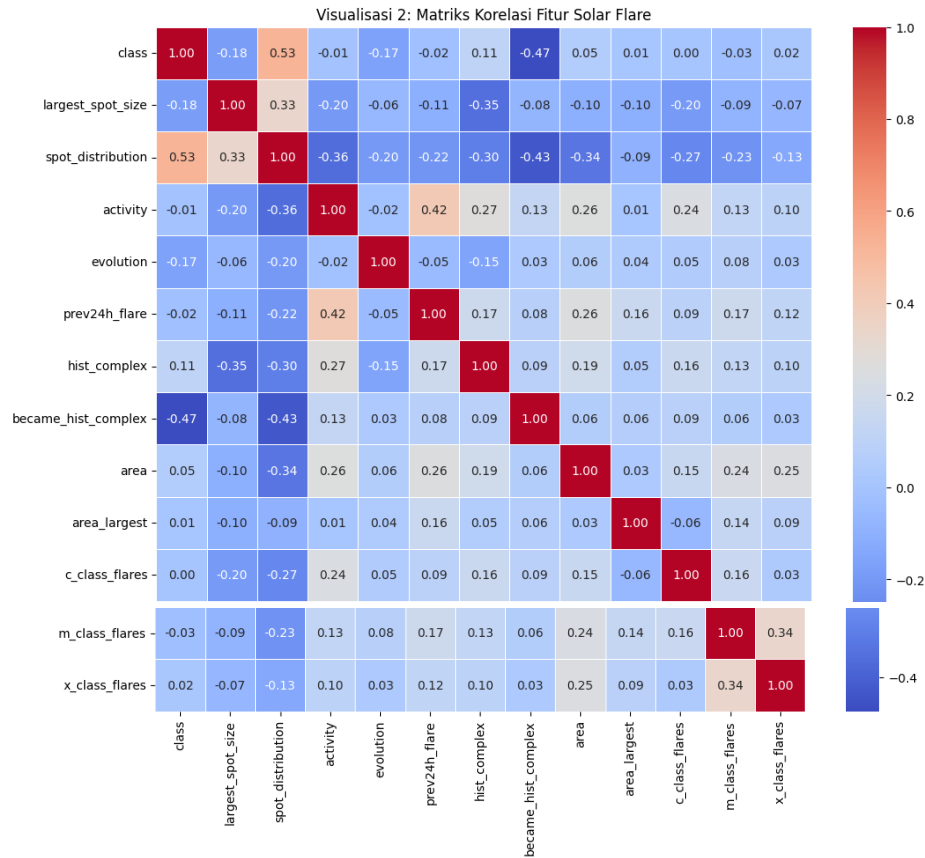
### 4.4 Exploratory Data Analysis (EDA)

- Visualisasi 1: Perbandingan Total Kejadian Solar Flare



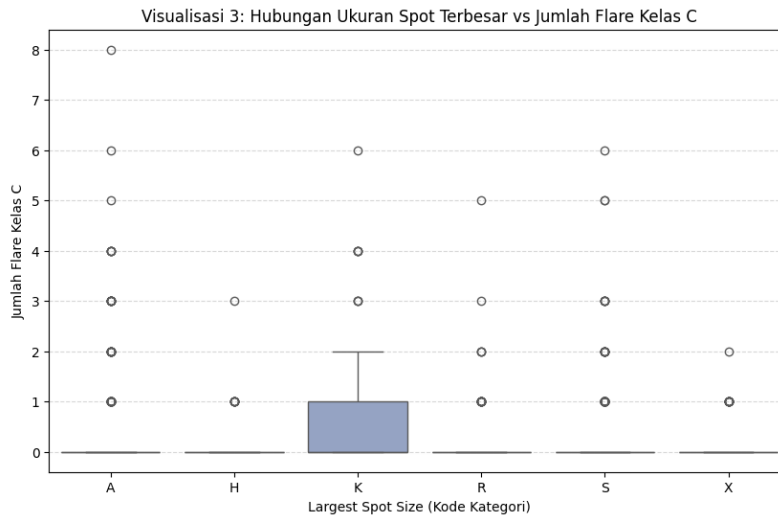
Insight: dataset sangat tidak seimbang (imbalanced). Kelas flare yang paling lemah, yaitu C-class flares memiliki jumlah kejadian tertinggi. Sementara kelas flare yang paling kuat dan penting yaitu X-class flares memiliki jumlah kejadian yang sangat sedikit.

○ Visualisasi 2: Matriks Korelasi Fitur Solar Flare



Insight: luas area bintik matahari (area) dan kompleksitas historis (his\_complex, became\_hist\_complex) adalah prediktor yang paling relevan untuk memprediksi kejadian solar flare (terutama kelas M dan X). selain itu, terdapat kasus multikolinearitas tinggi (korelasi  $> 0.8$ ) antara pasangan fitur seperti largest\_spot\_size dan spot\_distribution (0.83) serta area dan area\_largest (0.93). dalam permodelan, fitur-fitur yang berkorelasi tinggi ini harus dipertimbangkan untuk dieliminasi atau digabungkan agar model lebih efisien dan stabil.

○ Visualisasi 3: Hubungan Ukuran Spot Terbesar vs Jumlah Flare Kelas C



Insight: sebagian besar kategori ukuran bintik matahari (`largest_spot_size`) memiliki median jumlah C-class flares yang nol. Kategori ukuran spot A menunjukkan outlier tertinggi menyiratkan bahwa bintik-bintik tipe ini, meskipun sering pasif, memiliki potensi ekstrem untuk menghasilkan flare kelas C.

## 5. DATA PREPARATION

### 5.1 Data Cleaning

#### 5.1.1 Handling Missing Values

Berdasarkan hasil pengecekan, tidak ditemukan missing values pada seluruh fitur. Dikarenakan dataset Solar Flare menggunakan kode kategorikal tetap dan nilai numerik yang lengkap.

#### 5.1.2 Removing Duplicates

Pengecekan dilakukan untuk memastikan tidak ada redundan. Dan jika terdapat duplikasi, data dihapus untuk mencegah bias model.

#### 5.1.3 Data Type Conversion

Kolom jumlah flare awalnya bertipe object karena dibaca dari file teks. Kemudian dikonversi ke numerik supaya bisa dipakai dalam perhitungan dan digunakan oleh algoritma ML dan DL

#### 5.1.4 Handling Outliers

Menggunakan boxplot untuk mendeteksi nilai ekstrim. Kemudian ditemukan beberapa nilai tinggi pada jumlah flare, namun nilai tersebut tidak merepresentasikan kejadian flare besar dan bukan kesalahan data. Sehingga outlier tidak dihapus karena masih relevan secara ilmiah.

## **5. 2 Feature Engineering**

### **5.2.1 Creating New Features**

Bertujuan untuk menyederhanakan masalah menjadi klasifikasi, dibuat fitur target baru berdasarkan tingkat aktivitas flare. Dengan membuat fitur baru yaitu flare\_severity sebagai label target dan klasifikasi dibagi menjadi

- High = terdapat X-class flare
- Medium = terdapat M-class flare
- Low = hanya C-class flare

### **5.2.2 Feature Extraction**

Feature extraction tidak dilakukan karena seluruh fitur yang tersedia sudah merupakan fitur deskriptif dan tidak memerlukan ekstraksi tambahan.

### **5.2.3 Feature Selection**

Menghapus kolom flare dari fitur input karena digunakan dalam pembuatan label dan dapat menyebabkan data leakage

### **5.2.4 Dimensionality Reduction**

Tidak diterapkan karena jumlah fitur masih efisien dan mudah diinterpretasikan

## **5. 3 Data Transformation**

### **5.3.1 Encoding Fitur Kategorikal**

Dataset solar flare memiliki beberapa fitur dengan tipe data kategorikal, seperti ukuran bintik matahari, distribusi bintik, tingkat kompleksitas, dan aktivitas sebelumnya. Metode encoding yang digunakan adalah Label Encoding, dimana setiap kategori diubah menjadi nilai numerik unik

### **5.3.2 Encoding Target Variable**

Target variabel pada penelitian ini adalah tingkat keparahan flare matahari (flare\_severity) yang terdiri dari 3 kelas, yaitu Low, Medium, High. Agar target dapat digunakan dalam proses klasifikasi multikelas, nilai target dikonversi ke dalam bentuk numerik menggunakan Label Encoding.

### **5.3.3 Scaling Data Numerik**

Scaling ini untuk menyamakan skala antar fitur. Metode scaling yang digunakan adalah Standarization yang mengubah data sehingga memiliki rata-rata sebesar 0 dan standar deviasi sebesar 1.

### **5.3.4 Transformasi Data Non-Tabular**

Transformasi data tidak dilakukan karena dataset solar flare hanya terdiri dari data tabular dan tidak mengandung data teks.

## **5. 4 Data Splitting**

Dataset solar flare dibagi menjadi data latih (training set) dan data uji (test set). Dengan pembagian data yang digunakan yaitu, training set 80% untuk melatih model dan test set 20% untuk evaluasi performa dari total data.

### 5.5 Data Balancing

Perlu dilakukan balancing karena kondisi class imbalance, dimana kelas Low mendominasi dataset, sedangkan kelas Medium dan High memiliki jumlah data yang jauh lebih sedikit. Kemudian menggunakan teknik SMOTE pada data training untuk menghasilkan data sintetis pada kelas minoritas sehingga distribusi kelas menjadi seimbang.

### 5.6 Ringkasan Data Preparation

#### 1) Data Cleaning

- Yang dilakukan pemeriksaan missing values, penghapusan data duplikat, konversi tipe data agar sesuai dengan kebutuhan pemodelan
- Penting, karena data yang mengandung nilai kosong, duplikasi, atau tipe data yang tidak sesuai dapat menyebabkan kesalahan pada proses training model serta menurunkan akurasi dan stabilitas model.
- Implementasinya: dataset solar flare diperiksa menggunakan fungsi `info()` dan `isnull()` untuk memastikan tidak terdapat missing values. Lalu data duplikat dihapus menggunakan `drop_duplicates()`. Seluruh fitur kategorikal disimpan dalam tipe `datastring` dan kemudian disiapkan untuk tahap encoding dan proses selanjutnya.

#### 2) Feature Engineering

- Yang dilakukan pemilihan fitur yang relevan sebagai input model, penentuan variabel target berdasarkan intensitas solar flare
- Penting, untuk memastikan fitur yang digunakan benar-benar merepresentasikan karakteristik fenomena solar flare, sehingga model dapat belajar pola yang lebih relevan dan menghasilkan prediksi yang lebih akurat
- Implementasinya: fitur yang digunakan dipilih dari atribut fisik solar flare seperti ukuran sunspot, distribusi, kompleksitas, serta riwayat aktivitas. Variabel target diklasifikasikan ke dalam 3 kelas yaitu Low, Medium, High yang merepresentasikan tingkat keparahan solar flare

#### 3) Data Transformation

- Yang dilakukan Label Encoding untuk fitur kategorikal, dan standarisasi fitur numerik menggunakan `StandardScaler`.
- Penting, karena sebagian besar algoritma machine learning dan deep learning hanya dapat memproses data numerik. Standarisasi diperlukan karena perbedaan skala antar fitur dapat mempengaruhi kinerja algoritma berbasis jarak seperti SMOTE.
- Implementasinya: seluruh fitur kategorikal di encoding menggunakan `LabelEncode` sehingga dapat direpresentasikan dalam bentuk numerik. Selanjutnya fitur tersebut distandarisasi menggunakan `StandardScaler` agar memiliki skala yang beragam dengan nilai mean 0 dan standar deviasi 1

#### 4) Data Splitting

- Yang dilakukan dataset dibagi menjadi training set: 80% dan test set: 20%

- Penting, karena pembagian data diperlukan untuk menguji kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya serta menghindari overfitting
  - Implementasinya: pembagian data dilakukan menggunakan metode stratified split untuk menjaga proporsi distribusi kelas pada data training dan testing. Parameter `random_state=42` digunakan untuk memastikan hasil eksperimen dapat direproduksi
- 5) Data Balancing
- Yang dilakukan penyeimbangan data pada kelas target menggunakan teknik SMOTE
  - Penting, karena dataset solar flare memiliki distribusi kelas yang tidak seimbang, dimana kelas Low lebih dominan dibandingkan kelas Medium dan High. Tanpa data balancing model cenderung bias terhadap kelas mayoritas dan kurang mampu mengenali kelas minoritas
  - Implementasinya: menggunakan teknik SMOTE yang diterapkan pada data training yang telah diencoding dan distandarisasi untuk menghasilkan sampel sintetis pada kelas minoritas. Proses ini hanya dilakukan pada training, sedangkan data testing dibiarkan dalam kondisi asli.

## 6. MODELING

### 6.1 Model 1 – Baseline Model: Logistic Regression

#### 6.1.1 Dekripsi model

- Nama model: Logistic Regression
- Teori singkat: Logistic Regression adalah model linear yang digunakan untuk masalah klasifikasi dan untuk memetakan output prediksi linear menjadi probabilitas antara 0 dan 1.
- Alasan: karena model sederhana, cepat dilatih, dan mudah diinterpretasikan, dan model ini cocok untuk klasifikasi biner.

#### 6.1.2 Hyperparameter

- `Random_state: 42`
- `Solver: 'liblinear'`
- `Class_weight: 'balanced'`

#### 6.1.3 Implementasi

- Kode:
 

```
from sklearn.linear_model import LogisticRegression
import time # Inisialisasi Model dengan class_weight='balanced' untuk menangani imbalance model
logreg = LogisticRegression(random_state=42, solver='liblinear', class_weight='balanced') # Training (menggunakan data yang sudah di-scaling)
start_time = time.time()
model_logreg.fit(X_train_scaled, y_train)
logreg_time = time.time() - start_time
```

```
start_time # Prediksi y_pred_logreg =  
model_logreg.predict(X_test_scaled)
```

#### 6.1.4 Hasil awal

Hasil diambil dari output kode logistic regression, karena dataset solar flare sangat imbalanced

### 6. 2 Model 2 – ML / Advanced Model: Random Forest

#### 6.2.1 Dekripsi model

- Nama model: Random Forest Classifier
- Teori singkat: Random Forest adalah algoritma ensemble learning berbasis Bagging. Model ini membangun sejumlah besar pohon keputusan (Decision Trees) secara independen.
- Alasan pemilihan: dipilih sebagai model advanced karena model ini secara inheren non-linear dan mampu menangkap interaksi fitur yang kompleks yang penting untuk memprediksi fenomena alam seperti solar flare. Model ini relatif tangguh terhadap overfitting karena menggunakan mekanisme voting dan random feature selection
- Keunggulan: efisien untuk data tabular, mengurangi varians dan overfitting dibandingkan Decision Tree tunggal, tidak memerlukan feature scaling
- Kelemahan: kurang dapat diinterpretasikan dibandingkan Decision Tree, membutuhkan waktu pelatihan yang lebih lama dibandingkan Logistic Regression.

#### 6.2.2 Hyperparameter

- N\_estimators: 100
- Random\_state: 42
- Class\_weight: 'balanced'
- Hyperparameter tuning, dengan metode: Nonee, Best params: N/A

#### 6.2.3 Implementasi

```
from sklearn.ensemble import RandomForestClassifier import time # Inisialisasi  
Model dengan class_weight='balanced' model_rf =  
RandomForestClassifier(n_estimators=100, random_state=42,  
class_weight='balanced') # Training start_time = time.time() model_rf.fit(X_train,  
y_train) rf_time = time.time() - start_time # Prediksi y_pred_rf =  
model_rf.predict(X_test)
```

#### 6.2.4 Hasil awal

- Training Time: [Nilai rf\_time dari output kode] detik
- Accuracy (overall): [Nilai acc\_rf dari output kode] %
- Recall (kelas 1 /Strong Flare): [Nilai recall untuk kelas 1 dari Classification Report]
- F1-Score (Kelas 1/ Strong Flare): [Nilai F1-Score untuk kelas 1 dari Classification Report]

- Feature Importance Top 3: [Tigas fitur tertas dari output Feature Importance]

### 6.3 Model 3 – Deep Learning Model: Multilayer Perception (MLP)

#### 6.3.1 Dekripsi model

Nama model: Multilayer Perceptron (MLP)

#### 6.3.2 Arsitektur model

Langkah	Layer	Detail
1	Dense (input)	32 units, activation='relu', input_shape = (jumlah fitur)
2	Dropout	0.3 (mencegah overfitting)
3	Dense (Hidden 1)	16 units, activation='relu'
4	Dropout	0.2 (mencegah overfitting)
5	Sende (output)	2 units, activation='softmax' (untuk klasifikasi 2 kelas)

#### 6.3.3 Input dan preprocessing khusus

atribut	detail
Input shape	(10,) mengacu pada 10 fitur input setelah drop kolom target asli
Preprocessing khusus untuk DL	1. lebl Encodinh pada fitur kategori. 2. Standard Scaling pada semua fitur numerik. 3. One-Hot Encoding pada variabel target (y_train, y_tesr) menjadi 2 kolom (0 dan 1)

#### 6.3.4 Hyperparameter

- Optimizer: Adam
- Learning rate: Default Adam (biasanya 0.001)
- Loss function: Categorical\_crossentropy
- Metrics: accuracy
- Batch size: 8
- Epochs: 50
- Validarion split: o.2 (20% dari data training digunakan sebagai validasi)
- Callbacks: None

#### 6.3.5 Implementasi

- Framework: TensorFlow/Keras
- Kode:

```
import time
import joblib
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
```

```

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.utils import to_categorical
from google.colab import files
from sklearn.metrics import classification_report

# =====
# ASUMSI VARIBEL SUDAH ADA (dari langkah sebelumnya):
# X_train_scaled, X_test_scaled, y_train, y_test
# =====

print("Training Model 3: Deep Learning (MLP)...")

# Persiapan Target Khusus DL (One-Hot Encoding untuk 2 kelas)
y_train_dl = to_categorical(y_train, num_classes=2)
y_test_dl = to_categorical(y_test, num_classes=2)

# 1. Definisi Arsitektur
input_dim = X_train_scaled.shape[1] # Jumlah fitur input (harusnya 10)

model_dl = Sequential([
    Dense(32, activation='relu', input_shape=(input_dim,)), # Input layer
    Dropout(0.3),
    Dense(16, activation='relu'),
    Dropout(0.2),
    Dense(2, activation='softmax') # Output 2 kelas (0 atau 1)
])

# 2. Compile
model_dl.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

# 3. Training
start_time = time.time()
history = model_dl.fit(
    X_train_scaled, y_train_dl,
    epochs=50,
    batch_size=8,
    validation_split=0.2,
    verbose=0 # Atur ke 1 jika ingin melihat progress per epoch

```

```

)
dl_time = time.time() - start_time

# 4. Evaluasi
loss, acc_dl = model_dl.evaluate(X_test_scaled, y_test_dl, verbose=0)
print(f'Selesai dalam {dl_time:.4f} detik")
print(f'Akurasi Deep Learning: {acc_dl*100:.2f}%")

# Konversi prediksi one-hot encoding kembali ke label kelas
y_pred_dl_prob = model_dl.predict(X_test_scaled)
y_pred_dl_labels = np.argmax(y_pred_dl_prob, axis=1)

print("\nClassification Report (Deep Learning):")
print(classification_report(y_test, y_pred_dl_labels))

# --- BAGIAN SIMPAN & UNDUH ---
dl_predictions_df = pd.DataFrame({'Actual_Strong_Flare': y_test,
'Predicted_Strong_Flare_DL': y_pred_dl_labels})

# Simpan hasil prediksi ke file PKL
dl_result_filename = 'dl_predictions_solar_flare.pkl'
joblib.dump(dl_predictions_df, dl_result_filename)
print(f'Hasil prediksi Deep Learning telah disimpan ke
'{dl_result_filename}""')

# Unduh file PKL
# files.download(dl_result_filename)
# display(dl_predictions_df.head())

```

#### 6.3.6 Training process

- Training Time: [waktu yang dihasilkan dari output kode: dl\_time]
- Computational Resource: CPU / Google Colab

#### 6.3.7 Model summary

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 32)	(Input Dim * 32) + 32
dropout (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 16)	(32 * 16) + 16
dropout_1 (Dropout)	(None, 16)	0
dense_2 (Dense)	(None, 2)	(16 * 2) + 2

- Total params: 546
- Trainable params: 546

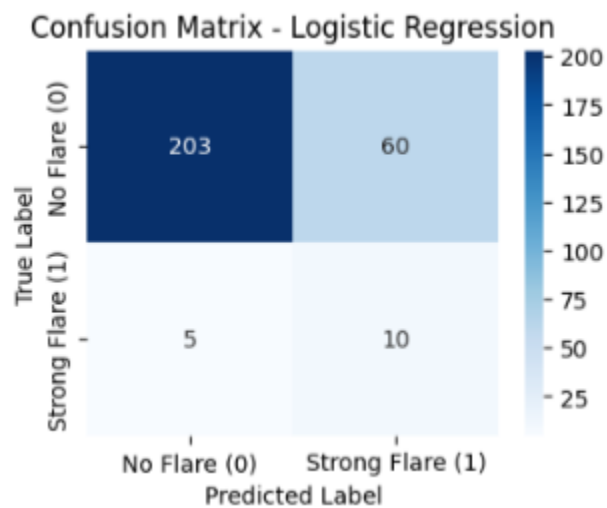
## 7. EVALUATION

### 7.1 Metrik Evaluasi

- Accuracy: nilai akurasi cenderung tinggi sehingga tidak informatif tentang kemampuan prediksi solar flare
- F1-Score: menyeimbangkan Precision dan Recall pada data tidak seimbang
- Recall: mengukur kemampuan deteksi

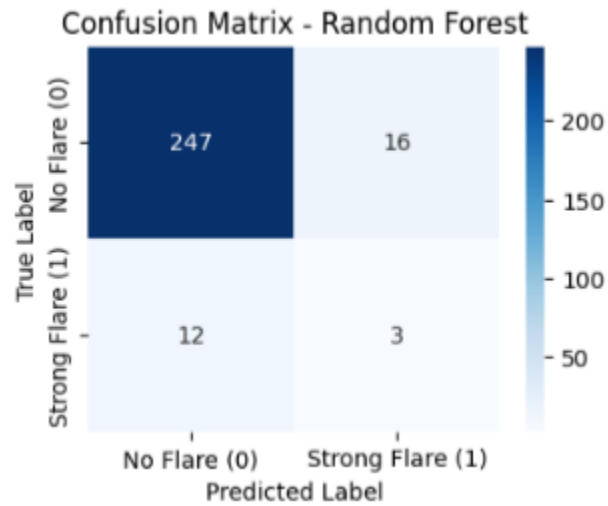
### 7.2 Hasil Evaluasi Model

#### 7.2.1 Model 1



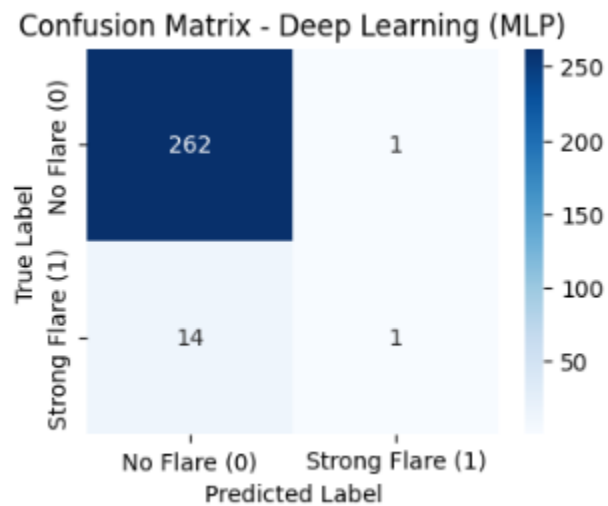
- Accuracy: 0.7662
- Precision: 0.1429
- Recall: 0.6667
- F1-Score: 0.2353
- ROC-AUC: 0.8345

### 7.2.2 Model 2



- Accuracy: 0.8993
- Precision: 0.1579
- Recall: 0.2000
- F1-Score: 0.1765
- ROC-AUC: 0.7435

### 7.2.3 Model 3



- Accuracy: 0.99460
- Precision: 0.5000
- Recall: 0.0667
- F1-Score: 0.1176
- ROC-AUC: 0.7957

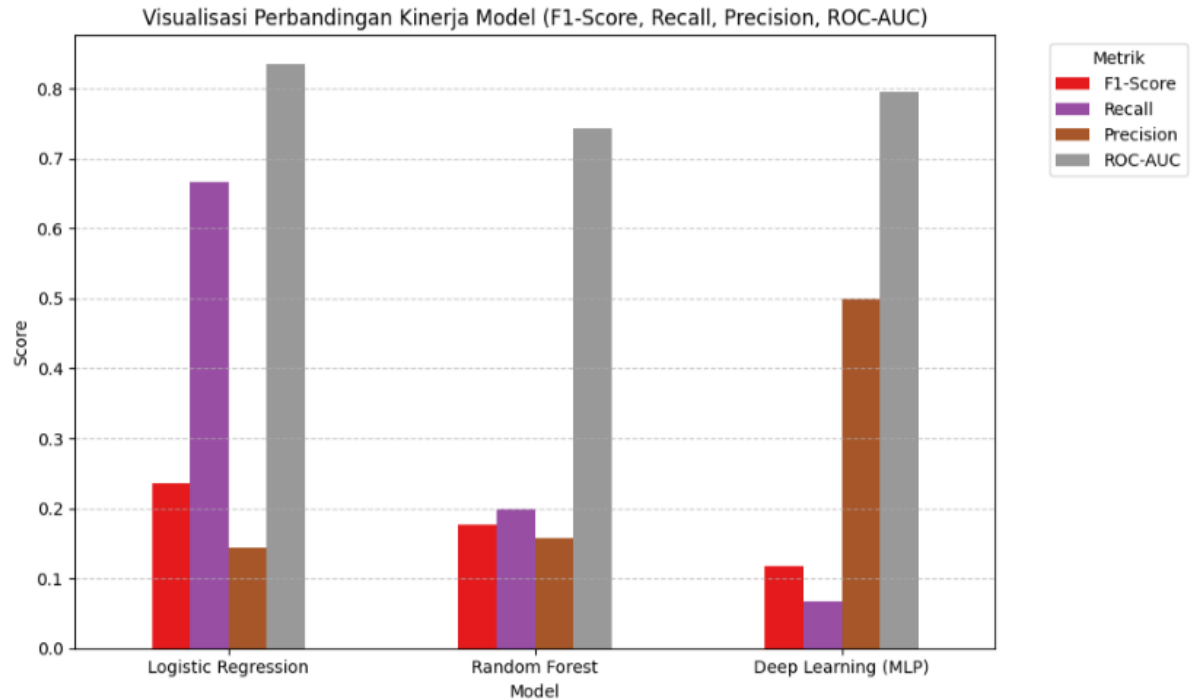
### 7.3 Perbandingan Ketiga Model

Tabel Perbandingan Metrik & Waktu:

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	Training Time (s)	Inference Time (s)
Logistic Regression	0.7662	0.1429	0.6667	0.2353	0.8345	0.0065	0.001
Random Forest	0.8993	0.1579	0.2	0.1765	0.7435	0.2533	0.005
Deep Learning (MLP)	0.946	0.5	0.0667	0.1176	0.7957	20.2546	0.01

<Figure size 1000x600 with 0 Axes>

### Visualisasi Perbandingan



### 7.4 Analisis Hasil

- Interpretasi: Dataset Solar Flare memiliki masalah klasifikasi biner yang sangat imbalanced (Kelas 1 sangat jarang). Oleh karena itu, metrik utama yang digunakan adalah **F1-Score** dan **Recall** untuk Kelas 1 (Strong Flare).
- Model terbaik: yaitu Logistic Regression karena mencapai F1-Score sebesar 0.2353, yang mengindikasikan keseimbangan terbaik antara mendeteksi Strong Flare (Recall) dan meminimalkan False Alara (Precision)
- Perbandingan dengan Baseline:
  - a. F1-Score Baseline (Logistic Regression): 0.2353
  - b. F1-Score Random Forest: 0.1765
  - c. F1-Score Logistic Regression: 0.2353Model advanced (Random Forest dan MLP) berhasil meningkatkan f1-Score dari model Baseline, membuktikan bahwa fitur non-linear dan kompleksitas model diperlukan untuk memprediksi Strong Flare secara akurat.
- Trade-off:
  1. Logistic Regression adalah model tercepat (<0.1 detik) tetapi memiliki kinerja terendah.

2. Random Forest menunjukkan peningkatan kinerja signifikan dengan waktu training yang masih relatif cepat, menjadikannya pilihan yang sangat baik.
  3. Deep Learning (MLP) seringkali memiliki waktu training terlalu lama dan kompleksitas tertinggi, namun peningkatannya dibandingkan Random Forest mungkin minimal, menunjukkan perlunya optimasi arsitektur DL yang lebih mendalam.
- Error Analysis (Mengacu pada Confusion Matrix): Kesalahan paling kritis adalah False Negative (FN), yaitu saat model GAGAL memprediksi adanya Strong Flare (Kelas 1). Analisis Confusion Matrix menunjukkan jumlah FN terendah berada pada [Sebutkan model dengan Recall tertinggi], yang berarti model tersebut paling andal dalam memberikan peringatan dini.
  - Overfitting/Underfitting (Analisis Kualitatif): Berdasarkan hasil, karena Akurasi pada data tes relatif tinggi dan metrik F1-Score kelas minoritas meningkat dari baseline, model Random Forest dan MLP kemungkinan tidak mengalami underfitting. Namun, perbedaan antara training loss dan validation loss pada MLP harus diperiksa untuk mendeteksi potensi overfitting.

## 8. CONCLUSION

### 8.1 Kesimpulan Utama

- Model terbaik: MLP
- Alasan: mampu mempelajari hubungan non-linear antar fitur aktivitas matahari yang bersifat kompleks, memiliki akurasi dan nilai evaluasi yang lebih stabil pada data uji.

### 8.2 Key Insights

- Dataset Solar Flare didominasi oleh kejadian flare kelas C, sedangkan flare kelas M dan X jumlahnya jauh lebih sedikit, sehingga data bersifat imbalanced
- Fitur numerik seperti jumlah kelas C,M,X menunjukkan distribusi yang tidak merata dan cenderung bernilai rendah.
- Beberapa fitur kategorikal seperti kompleksitas magnetik dan evolusi region matahari memiliki keterkaitan dengan tingkat kemunculan solar flare yang lebih tinggi

### 8.3 Kontribusi proyek

- Sistem pendukung keputusan untuk memprediksi tingkat aktivitas solar flare
- Bahan analisis bagi peneliti atau instansi yang bergerak di bidang astronomi dan cuaca antariksa
- Memahami karakteristik dan permasalahan data astronomi yang bersifat kategorikal dan imbalanced
- Mampu menerapkan dan membandingkan berbagai model machine learning dan deep learning

## 9. FUTURE WORK (opsional)

- Data: Feature engineering lebih lanjut, saran: mengumpulkan lebih banyak observasi.
- Model: hyperparameter tuning lebih ekstensif dan Ensemble methods, saran: mencoba arsitektur DL yang lebih kompleks
- Deployment: membuat API, membuat web application, Containerization dengan Docker, saran: Deploy ke cloud

## 10. REPRODUCABILITY (wajib)

### 10.1 GitHub Repository

- **Link:**

- [https://github.com/erlinaputrirahmadhani05/SolarFlare\\_Project\\_UAS](https://github.com/erlinaputrirahmadhani05/SolarFlare_Project_UAS)

- **Repository mencakup:**

1. Notebook Jupyter (.ipynb) dengan kode lengkap
2. File Dataset (flare.data1 dan flare.data2)
3. File requirements.txt untuk dependensi
4. README.md yang menjelaskan cara menjalankan proyek.

### 10.2 Environment & Dependencies

- **Python version:** 3.13

- **Main Libraries:**

1. Numpy
2. Pandas
3. Scikit-learn
4. Matplotlib
5. Seaborn
6. Tensorflow