

1. Serah Terima Program Kepada Laboran Program Studi Informatika  
( Melina Atikawati., S.Kom )



2. Tampilan Program

**Instalasi Modul dan  
Library**

```
# Import Libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
```

**Import Data**

```
# Meminta pengguna untuk memasukkan path atau nama file
file_path = input("Masukkan path atau nama file dataset (contoh: data/informatika.xlsm): ")

# Load file Excel
try:
    data = pd.ExcelFile(file_path)
    print("File berhasil dimuat!")
except FileNotFoundError:
    print("File tidak ditemukan! Pastikan path yang dimasukkan benar.")
    exit()

# Load data dari Sheet1 (atau Anda bisa meminta pengguna memilih sheet)
df = data.parse('Sheet1')
print("Dataset berhasil dimuat.")
```

```
Masukkan path atau nama file dataset (contoh: data/informatika.xlsm): informatika.xlsm
File berhasil dimuat!
Dataset berhasil dimuat.
```

Data yang digunakan dalam penelitian ini adalah data akademik mahasiswa dari Program Studi Informatika Universitas Muhammadiyah Sidoarjo angkatan 2020-2021, yang terdiri dari informasi Indeks Prestasi Kumulatif (IPK) dan Jumlah SKS yang diambil dari semester 1 hingga semester 6. Data ini digunakan untuk membangun model prediksi kelulusan

## Data Cleaning

```
# Step 2: Data Cleaning
df_cleaned = df.dropna() # Hapus baris dengan nilai kosong
df_cleaned['Avg_IPS'] = df_cleaned[['IPS_1', 'IPS_2', 'IPS_3', 'IPS_4', 'IPS_5', 'IPS_6']].mean(axis=1)
df_cleaned['Status'] = np.where(df_cleaned['Avg_IPS'] >= 2.70, 'Lulus', 'Tidak Lulus')
```

Loading...

```
/tmp/ipython-input-2922536085.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#
df_cleaned['Avg_IPS'] = df_cleaned[['IPS_1', 'IPS_2', 'IPS_3', 'IPS_4', 'IPS_5', 'IPS_6']].mean(axis=1)
/tmp/ipython-input-2922536085.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#
df_cleaned['Status'] = np.where(df_cleaned['Avg_IPS'] >= 2.70, 'Lulus', 'Tidak Lulus')
```

Dataset:

|   | NIM          | NAMA                       | IPS_1 | SKS_1 | IPS_2 | SKS_2 | \ |
|---|--------------|----------------------------|-------|-------|-------|-------|---|
| 0 | 2.010002e+11 | MOKHAMAD DIKI ARMANDA      | 3.83  | 20.0  | 3.80  | 20.0  |   |
| 1 | 2.010002e+11 | MUHAMMAD ALAIKA ASYROF     | 3.78  | 20.0  | 3.30  | 20.0  |   |
| 2 | 2.010002e+11 | ARI TOPAN IQBAL MADAGASKAR | 3.80  | 20.0  | 3.57  | 20.0  |   |
| 3 | 2.010002e+11 | FAHRI FAUZI HIDAYAT        | 3.72  | 20.0  | 3.78  | 18.0  |   |
| 4 | 2.010002e+11 | RENO FAIZAL FAHMI          | 2.53  | 20.0  | 3.13  | 18.0  |   |

|   | IPS_3 | SKS_3 | IPS_4 | SKS_4 | IPS_5 | SKS_5 | IPS_6 | SKS_6 |
|---|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 | 3.84  | 20.0  | 3.60  | 21.0  | 3.80  | 23.0  | 3.91  | 21.0  |
| 1 | 3.35  | 23.0  | 3.55  | 21.0  | 2.67  | 23.0  | 3.60  | 15.0  |
| 2 | 3.26  | 23.0  | 3.62  | 21.0  | 3.55  | 23.0  | 3.72  | 18.0  |
| 3 | 3.47  | 23.0  | 3.65  | 18.0  | 3.60  | 23.0  | 3.70  | 23.0  |
| 4 | 3.47  | 20.0  | 3.52  | 20.0  | 3.60  | 23.0  | 3.76  | 21.0  |

Gambar 1. Contoh Hasil Pembersihan Data Akademik

Gambar diatas menunjukkan hasil pembersihan data akademik mahasiswa, di mana pada tahap ini data masih dalam bentuk awal setelah diekstraksi dari sumber. Data tersebut terdiri atas kolom NIM, Nama, nilai IPS dan jumlah SKS dari semester 1 hingga semester 5. Proses pembersihan data dilakukan untuk memastikan bahwa seluruh nilai memiliki format yang seragam, seperti penggunaan titik desimal pada data numerik, serta menghilangkan duplikasi dan nilai kosong yang dapat mengganggu proses analisis. Pada tahap ini, belum terdapat atribut target seperti status kelulusan mahasiswa.

|   | NIM          | NAMA                       | IPS_1 | SKS_1 | IPS_2 | SKS_2 | \ |
|---|--------------|----------------------------|-------|-------|-------|-------|---|
| 0 | 2.010002e+11 | MOKHAMAD DIKI ARMANDA      | 3.83  | 20.0  | 3.80  | 20.0  |   |
| 1 | 2.010002e+11 | MUHAMMAD ALAIKA ASYROF     | 3.78  | 20.0  | 3.30  | 20.0  |   |
| 2 | 2.010002e+11 | ARI TOPAN IQBAL MADAGASKAR | 3.80  | 20.0  | 3.57  | 20.0  |   |
| 3 | 2.010002e+11 | FAHRI FAUZI HIDAYAT        | 3.72  | 20.0  | 3.78  | 18.0  |   |
| 4 | 2.010002e+11 | RENO FAIZAL FAHMI          | 2.53  | 20.0  | 3.13  | 18.0  |   |

|   | IPS_3 | SKS_3 | IPS_4 | SKS_4 | IPS_5 | SKS_5 | IPS_6 | SKS_6 | Avg_IPS  | Status |
|---|-------|-------|-------|-------|-------|-------|-------|-------|----------|--------|
| 0 | 3.84  | 20.0  | 3.60  | 21.0  | 3.80  | 23.0  | 3.91  | 21.0  | 3.796667 | Lulus  |
| 1 | 3.35  | 23.0  | 3.55  | 21.0  | 2.67  | 23.0  | 3.60  | 15.0  | 3.375000 | Lulus  |
| 2 | 3.26  | 23.0  | 3.62  | 21.0  | 3.55  | 23.0  | 3.72  | 18.0  | 3.586667 | Lulus  |
| 3 | 3.47  | 23.0  | 3.65  | 18.0  | 3.60  | 23.0  | 3.70  | 23.0  | 3.653333 | Lulus  |
| 4 | 3.47  | 20.0  | 3.52  | 20.0  | 3.60  | 23.0  | 3.76  | 21.0  | 3.335000 | Lulus  |

Gambar 2. Contoh Hasil Hasil Transformasi Data Akademik

Gambar diatas menampilkan hasil transformasi data akademik setelah dilakukan proses feature engineering. Perubahan utama yang terlihat adalah penambahan dua kolom baru, yaitu avg\_IPS yang merupakan rata-rata dari nilai IPS semester 1 hingga 5, dan kolom Status yang menunjukkan label kelulusan mahasiswa (Lulus atau Tidak Lulus).

### Data Testing Dan Data Training

```
[ ] # Step 3: Features dan Target
features = df_cleaned[['IPS_1', 'IPS_2', 'IPS_3', 'IPS_4', 'IPS_5', 'IPS_6',
                        'SKS_1', 'SKS_2', 'SKS_3', 'SKS_4', 'SKS_5', 'SKS_6']]
target = df_cleaned['Status']

# Split data menjadi training dan testing
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.3, random_state=42)
```

Persentase data training dan data testing yang digunakan dapat dilihat pada Tabel

Tabel 2.1. Kelompok Data Training dan Data Testing

| Kelompok | Data Training | Jumlah Data | Data Testing | Jumlah Data |
|----------|---------------|-------------|--------------|-------------|
| 1        | 80%           | 480         | 20%          | 120         |
| 2        | 60%           | 360         | 40%          | 240         |

Evaluasi dilakukan dengan dua skenario pembagian data, yaitu: 80% data training dan 20% data testing. Juga 60% data training dan 40% data testing.

### Naïve Bayes

```
# Step 4: Naive Bayes
nb_model = GaussianNB()
nb_model.fit(X_train, y_train)
y_pred_nb = nb_model.predict(X_test)
accuracy_nb = accuracy_score(y_test, y_pred_nb)
print("Model: Naive Bayes")
print(f"Akurasi: {accuracy_nb:.2f}")
print("Classification Report:")
print(classification_report(y_test, y_pred_nb))
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred_nb))
print("-" * 60)
```

```

Model: Naive Bayes
Akurasi: 0.86
Classification Report:
              precision    recall  f1-score   support

    Lulus           1.00      0.84      0.91         70
   Tidak Lulus      0.48      1.00      0.65         10

   accuracy              0.86         80
  macro avg           0.74      0.92      0.78         80
 weighted avg           0.93      0.86      0.88         80

Confusion Matrix:
[[59 11]
 [ 0 10]]
-----
```

Perhitungan dilakukan menggunakan fungsi `accuracy_score()` dari pustaka *scikit-learn*, yang menghitung rasio jumlah prediksi benar terhadap seluruh data. Nilai akurasi dari masing-masing skenario ditampilkan pada Tabel

Tabel 2.2 Hasil Akurasi Model Naïve Bayes pada Skenario 1 dan 2

| Skenario Pembagian Data    | Akurasi |
|----------------------------|---------|
| 80% Training : 20% Testing | 86.25%  |
| 60% Training : 40% Testing | 86.25%  |

Tabel 2.3 Hasil Akurasi Model Naïve akurasi terbaik

| Classification Report: |           |        |          |         |
|------------------------|-----------|--------|----------|---------|
|                        | precision | recall | f1-score | support |
| Lulus                  | 1.00      | 0.84   | 0.91     | 70      |
| Tidak Lulus            | 0.48      | 1.00   | 0.65     | 10      |
| accuracy               |           |        | 0.86     | 80      |
| macro avg              | 0.74      | 0.92   | 0.78     | 80      |
| weighted avg           | 0.93      | 0.86   | 0.88     | 80      |
| Akurasi NB : 86.25%    |           |        |          |         |
| Confusion Matrix:      |           |        |          |         |
| [[59 11]               |           |        |          |         |
| [ 0 10 ]]              |           |        |          |         |

Pada algoritma Naïve Bayes, hasil akurasi yang diperoleh pada skenario pertama maupun skenario kedua adalah 86.25%. Tidak adanya perubahan nilai akurasi ini

## Decision Tree

```
# Step 5: Decision Tree
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
y_pred_dt = dt_model.predict(X_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print("Model: Decision Tree")
print(f"Akurasi: {accuracy_dt:.2f}")
print("Classification Report:")
print(classification_report(y_test, y_pred_dt))
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred_dt))
print("-" * 60)
```

Model: Decision Tree  
Akurasi: 0.96  
Classification Report:

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| Lulus        | 0.97      | 0.99   | 0.98     | 70      |
| Tidak Lulus  | 0.89      | 0.80   | 0.84     | 10      |
| accuracy     |           |        | 0.96     | 80      |
| macro avg    | 0.93      | 0.89   | 0.91     | 80      |
| weighted avg | 0.96      | 0.96   | 0.96     | 80      |

Confusion Matrix:  
[[69 1]  
[ 2 8]]

Evaluasi model Decision Tree dilakukan dengan pendekatan serupa. Model dibangun menggunakan fungsi Decision Tree Classifier () dari pustaka *scikit-learn*. Data dibagi menjadi dua skenario: 80% training dan 20% testing, serta 60% training dan 40% testing Model dilatih menggunakan data training, lalu diuji pada data testing untuk menghasilkan prediksi. Confusion matrix kemudian dibentuk dari hasil prediksi, dan akurasi dihitung berdasarkan jumlah prediksi benar menggunakan rumus:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Proses evaluasi ini menghasilkan akurasi yang dihitung secara otomatis melalui fungsi `accuracy_score()`, dengan hasil yang ditampilkan pada Tabel 3.3.

Tabel 2.4 Hasil Akurasi Model *Decion Tree* pada Skenario 1 dan 2

| Skenario Pembagian Data    | Akurasi |
|----------------------------|---------|
| 80% Training : 20% Testing | 96.25%  |
| 60% Training : 40% Testing | 96.25%  |

Tabel 2.5 Hasil Akurasi Model *Decision Tree* akurasi terbaik

#### Classification Report:

|                         | precision | recall | f1-score | support |
|-------------------------|-----------|--------|----------|---------|
| Lulus                   | 0.97      | 0.99   | 0.98     | 70      |
| Tidak Lulus             | 0.89      | 0.80   | 0.84     | 10      |
| accuracy                |           |        | 0.96     | 80      |
| macro avg               | 0.93      | 0.89   | 0.91     | 80      |
| weighted avg            | 0.96      | 0.96   | 0.96     | 80      |
| Akurasi NB: 96.25%      |           |        |          |         |
| Confusion Matrix:       |           |        |          |         |
| [[ <u>69</u> <u>1</u> ] |           |        |          |         |
| [ <u>2</u> <u>8</u> ]   |           |        |          |         |

Pada algoritma Decision Tree, akurasi yang dihasilkan juga tetap sama pada kedua skenario, yaitu sebesar 96.25%. Konsistensi nilai akurasi ini menandakan bahwa Decision Tree mampu membentuk aturan klasifikasi yang solid dan efektif, meskipun proporsi data latih dikurangi.

## Random Forest

```
[10] # Step 6: Random Forest
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print("Model: Random Forest")
print(f"Akurasi: {accuracy_rf:.2f}")
print("Classification Report:")
print(classification_report(y_test, y_pred_rf))
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred_rf))
print("-" * 60)
```

Model: Random Forest  
Akurasi: 0.97

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| Lulus        | 0.97      | 1.00   | 0.99     | 70      |
| Tidak Lulus  | 1.00      | 0.80   | 0.89     | 10      |
| accuracy     |           |        | 0.97     | 80      |
| macro avg    | 0.99      | 0.90   | 0.94     | 80      |
| weighted avg | 0.98      | 0.97   | 0.97     | 80      |

Confusion Matrix:  
[[70 0]  
[ 2 8]]

-----

Algoritma Random Forest, terjadi perbedaan akurasi antara kedua skenario. Pada skenario 80:20, Random Forest mencatat akurasi tertinggi yaitu 97.50%, sedangkan pada skenario 60:40, akurasinya sedikit menurun menjadi 96.88%. Meskipun terjadi penurunan, selisih akurasi ini relatif kecil dan tidak berdampak signifikan terhadap keandalan model.

Tabel 2.6 Hasil Akurasi Model *Random Forest* pada Skenario 1 dan 2

| Skenario Pembagian Data    | Akurasi |
|----------------------------|---------|
| 80% Training : 20% Testing | 97.50%  |
| 60% Training : 40% Testing | 96.88%  |

Tabel 32.7 Hasil Akurasi Model *Random Forest* akurasi terbaik

| Classification Report:        |           |        |          |         |
|-------------------------------|-----------|--------|----------|---------|
|                               | precision | recall | f1-score | support |
| Lulus                         | 0.97      | 1.00   | 0.99     | 70      |
| Tidak Lulus                   | 1.00      | 0.80   | 0.89     | 10      |
| accuracy                      |           |        | 0.97     | 80      |
| macro avg                     | 0.99      | 0.90   | 0.94     | 80      |
| weighted avg                  | 0.98      | 0.97   | 0.97     | 80      |
| Akurasi Random Forest: 97.50% |           |        |          |         |
| Confusion Matrix:             |           |        |          |         |
| [[70 0]                       |           |        |          |         |
| [ 2 8]]                       |           |        |          |         |

## Visualisasi Confusion Matrix

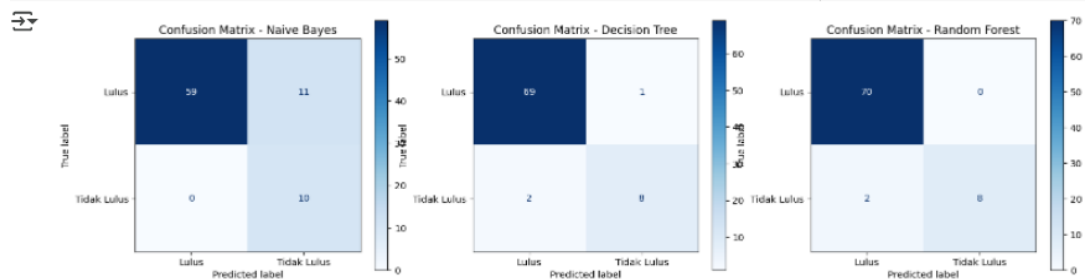
```
# Visualisasi Confusion Matrix
fig, axes = plt.subplots(1, 3, figsize=(18, 5))

# Naive Bayes
cm_nb = confusion_matrix(y_test, y_pred_nb)
disp_nb = ConfusionMatrixDisplay(confusion_matrix=cm_nb, display_labels=nb_model.classes_)
disp_nb.plot(ax=axes[0], cmap='Blues')
axes[0].set_title('Confusion Matrix - Naive Bayes')

# Decision Tree
cm_dt = confusion_matrix(y_test, y_pred_dt)
disp_dt = ConfusionMatrixDisplay(confusion_matrix=cm_dt, display_labels=dt_model.classes_)
disp_dt.plot(ax=axes[1], cmap='Blues')
axes[1].set_title('Confusion Matrix - Decision Tree')

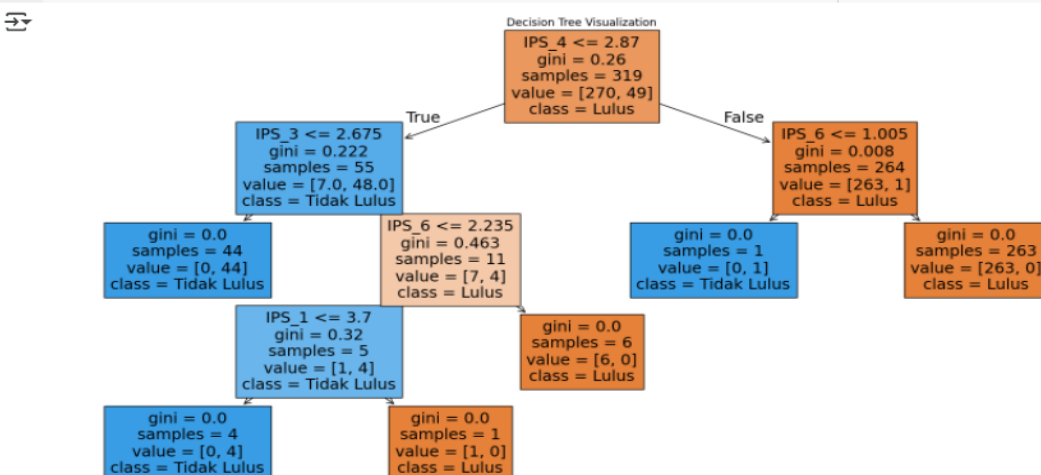
# Random Forest
cm_rf = confusion_matrix(y_test, y_pred_rf)
disp_rf = ConfusionMatrixDisplay(confusion_matrix=cm_rf, display_labels=rf_model.classes_)
disp_rf.plot(ax=axes[2], cmap='Blues')
axes[2].set_title('Confusion Matrix - Random Forest')

plt.show()
```



## Visualisasi Pohon Keputusan

```
# Step 8: Visualisasi Pohon Keputusan
plt.figure(figsize=(20, 10))
plot_tree(dt_model, feature_names=features.columns, class_names=dt_model.classes_, filled=True)
plt.title("Decision Tree Visualization")
plt.show()
```



Dalam pohon keputusan ini, algoritma hanya memilih fitur-fitur yang paling membantu memisahkan data dengan baik. Karena IPS\_2 tidak memberikan pemisahan yang signifikan dibandingkan fitur lain seperti IPS\_3, IPS\_4, dan IPS\_6, maka IPS\_2 tidak digunakan dalam pembentukan pohon. Jadi, IPS\_2 diabaikan karena atribut lain lebih efektif dalam membedakan mahasiswa yang lulus dan tidak lulus.

Analisis Hasil Evaluasi Akurasi Tiga Algoritma

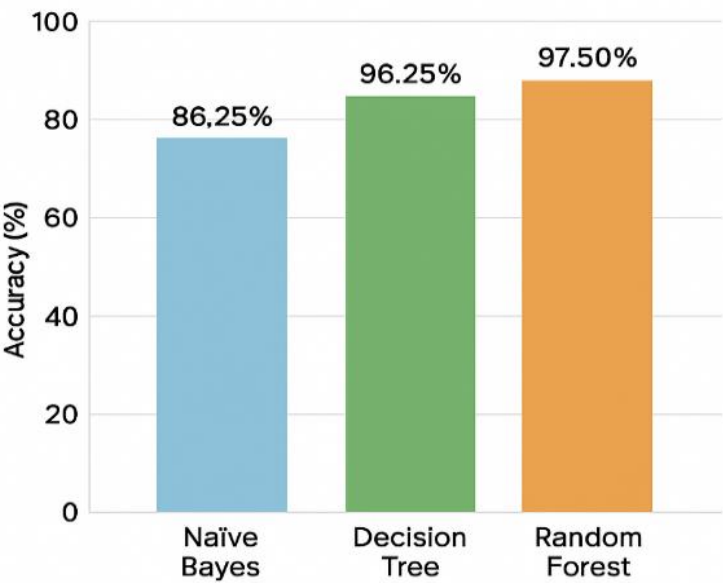
```
# Output akurasi
print(f"Akurasi Naive Bayes: {accuracy_nb * 100:.2f}%")
print(f"Akurasi Decision Tree: {accuracy_dt * 100:.2f}%")
print(f"Akurasi Random Forest: {accuracy_rf * 100:.2f}%")
```

Akurasi Naive Bayes: 86.25%  
Akurasi Decision Tree: 96.25%  
Akurasi Random Forest: 97.50%

Berikut merupakan grafik dan tabel dari algoritma *naïve bayes*, *decision tree* dan *random forest* setelah dilakukan evaluasi pada setiap algoritma dan juga skenario :

Tabel 3.6 Hasil Akurasi tiga Model Algoritma dengan akurasi terbaik

| Algoritma        | Akurasi | Presisi | Recall | F1-Score |
|------------------|---------|---------|--------|----------|
| 1. Naïve Bayes   | 86.25%  | 93.00%  | 86.00% | 88.00%   |
| 2. Decision Tree | 96.25%  | 96.00%  | 96.00% | 96.00%   |
| 3. Random Forest | 97.50%  | 98.00%  | 97.00% | 97.00%   |



Gambar 5. Grafik Akurasi tiga Algoritma

Berdasarkan tabel evaluasi, Random Forest mencatat performa tertinggi di semua metrik, dengan akurasi 97.50%, presisi 98%, recall 97%, dan F1-score 97%, menjadikannya algoritma terbaik untuk prediksi kelulusan. Decision Tree berada di posisi kedua dengan nilai yang stabil pada semua metrik, yaitu 96.25% akurasi serta presisi, recall, dan F1-score masing-masing 96%. Ini menunjukkan bahwa model bekerja konsisten dan efektif. Sementara itu, Naïve Bayes memiliki akurasi paling rendah (86.25%), meskipun presisinya cukup tinggi (93%). Recall dan F1-score yang lebih rendah menunjukkan bahwa model kurang mampu menangkap keseluruhan pola kelulusan secara optimal.

### 3. Berita Acara Serah Terima Kepada Laboran Program Studi Informatika

  
UNIVERSITAS MUHAMMADIYAH SIDOARJO  
FAKULTAS SAINS DAN TEKNOLOGI  
PROGRAM STUDI : INFORMATIKA (SI) • TEKNIK INDUSTRI (IT) • TEKNIK NEPRAKERJA • TEKNIK SPASIAL  
• TEKNIK ELEKTRO (SE) • TEKNOLOGI HASIL PERTANIAN (ST) • AGROTEKNOLOGI (ST)

**BERITA ACARA SERAH TERIMA**  
**PERBANDINGAN ALGORITMA *MACHINE LEARNING* DALAM MEMPREDIKSI**  
**KELULUSAN MAHASISWA**  
**2025**  
**Prodi Informatika**  
**Fakultas Sains dan Teknologi**  
**Universitas Muhammadiyah Sidoarjo**

Pada hari ini : Selasa  
Tanggal / Bulan / Tahun : 22 Juli 2025  
Bertempat di : Jl. Raya Gelam RT 009 / RW 003, Desa Gelam, Koc. Candi, Kab. Sidoarjo, Jawa Timur

Yang bertanda tangan dibawah ini :

1. Nama : M Cholis Afandi
2. NIM : 211080200166

Dalam hal ini bertindak sebagai Dosen / Mahasiswa, pemberi sumbangan atas nama Prodi Informatika yang selanjutnya disebut **PIHAK PERTAMA**

3. Nama : Melina Atikawati, S.Kom
- Tempat : Universitas Muhammadiyah Sidoarjo
- Alamat : Jl. Raya Gelam No.250, Pagerwaja, Gelam, Kec. Candi, Kabupaten Sidoarjo, Jawa Timur 61271

Dalam hal ini bertindak atas **Penerima** dari Perbandingan Algoritma *Machine Learning* Dalam Memprediksi Kelulusan Mahasiswa Dengan *Google Colaboratory* dari Prodi Informatika – Saintek UMSIDA, yang selanjutnya disebut **PIHAK KEDUA**. Dengan ini **PIHAK KEDUA** menyatakan Bersedia Mengimplementasikan dan Menggunakan *Google Colaboratory* bagi tempat **PIHAK KEDUA**, serta **PIHAK PERTAMA** tidak memungut biaya (*free*). **PIHAK PERTAMA** akan membantu proses perawatan selama 12 bulan mulai tertanggal serah terima.

Demikian Berita Acara ini dibuat untuk dipergunakan sebagaimana mestinya.

Sidoarjo, 22 Juli 2025

  
(Melina Atikawati, S.Kom)

  
(M Cholis Afandi)

Mengetahui  
Kaprosdi Informatika

(Ade Eviyanti, S.Kom., M.Kom)  
NIK. 204252

Jl. Raya Gelam 250, Candi, Telp: 081220183773 Sidoarjo – 61271  
Email : info@umsida.ac.id | www.umsida.ac.id