

## **INFORMASI PROYEK**

**Judul Proyek:** “Klasifikasi Pengguna Obat (Cannabis) berdasarkan Profil Psikologis Menggunakan Deep Learning”

**Nama Mahasiswa:** Hanan Labib Rasyaddin

**NIM:** 234311041

**Program Studi:** Teknologi Rekayasa Perangkat Lunak

**Mata Kuliah:** Data Science

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

**Tahun Akademik:** 2024/2025

**Link GitHub Repository:** <https://github.com/hanan1lab/DataScience-2025>

**Link Video Pembahasan:** <https://youtu.be/CUJqSck792A>

---

---

## **1. LEARNING OUTCOMES**

Pada proyek ini, mahasiswa diharapkan dapat:

1. Memahami konteks masalah prediksi penggunaan obat berdasarkan data psikologis
2. Melakukan analisis dan eksplorasi data (EDA) untuk melihat pola distribusi pengguna
3. Melakukan data preparation (cleaning, splitting, dan normalization) yang sesuai karena data bersifat tabular .
4. Mengembangkan tiga model machine learning yang terdiri:
  - Model baseline : Logistic Regression
  - Model machine learning advanced : Super Vector Machine
  - Model deep learning : Multilayer Perception
5. Menggunakan metrik evaluasi yang relevan (Accuracy, Precision, Recall, F1-Score).
6. Melaporkan hasil eksperimen secara ilmiah dan sistematis
7. Mengunggah seluruh kode proyek ke GitHub
8. Menerapkan prinsip software engineering dalam pengembangan proyek

---

## 2. PROJECT OVERVIEW

### 2.1 Latar Belakang

Penyalahgunaan obat-obatan terlarang (*Narkoba*) memiliki dampak signifikan terhadap kesehatan masyarakat, sosial, dan ekonomi. Identifikasi dini terhadap individu yang memiliki risiko tinggi menggunakan obat-obatan dapat membantu dalam perancangan strategi intervensi dan pencegahan yang lebih efektif.

Metode konvensional untuk mengidentifikasi pengguna obat seringkali bergantung pada wawancara manual atau tes fisik yang memakan waktu dan biaya. Diperlukan pendekatan berbasis data yang dapat memprediksi status penggunaan obat berdasarkan karakteristik psikologis (kepribadian) dan demografis yang lebih mudah dikumpulkan.

#### Manfaat Proyek :

- **Untuk Peneliti/Psikolog:** Memberikan wawasan tentang hubungan antara tipe kepribadian (seperti *Neuroticism, Impulsiveness*) dengan kecenderungan penggunaan obat.
- **Untuk Kesehatan Masyarakat:** Alat skrining awal (*screening tool*) yang efisien.

#### Referensi:

Fehrman, E., Muhammad, A. K., Mirkes, E. M., Egan, V., & Gorban, A. N. (2017). The Five Factor Model of personality and evaluation of drug consumption risk. *arXiv preprint arXiv:1506.06297*.

## 3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING

### 3.1 Problem Statements

- A. Bagaimana karakteristik demografi (usia, gender) dan psikologis (skor kepribadian) dari pengguna *Cannabis* dibandingkan dengan non-pengguna?
- B. Apakah model *Machine Learning* mampu memprediksi status penggunaan *Cannabis* (User vs Non-User) dengan akurasi yang dapat diandalkan (>75%)?
- C. Apakah pendekatan *Deep Learning* (Neural Network) memberikan performa yang lebih baik atau setara dibandingkan model klasifikasi tradisional (*Logistic Regression* dan *SVM*) untuk data tabular ini?

### 3.2 Goals

- A. Membangun model klasifikasi biner untuk memprediksi pengguna *Cannabis* dengan akurasi target > 80%.
- B. Mengukur dan membandingkan performa tiga pendekatan model (*Logistic Regression*, *SVM*, *Deep Learning*) menggunakan metrik Accuracy, Precision, Recall, dan F1-Score.)

- C. Menentukan model terbaik yang memiliki keseimbangan antara akurasi tinggi dan kemampuan mendeteksi pengguna (Recall tinggi).

### 3.3 Solution Approach

#### A. Model 1 – Baseline Model: Logistic Regression

**Deskripsi:** Logistic Regression adalah algoritma statistik yang digunakan untuk memprediksi probabilitas kejadian suatu kelas biner (0 atau 1) menggunakan fungsi Logistik (Sigmoid). Model ini mencari hubungan linear antara fitur input dan log-odds dari target.

**Alasan Pemilihan:** Logistic Regression adalah algoritma klasifikasi linier yang sederhana, cepat dilatih, dan mudah diinterpretasikan. Model ini sangat cocok dijadikan *baseline* (standar dasar) untuk melihat apakah data memiliki pola linier yang mudah dipisahkan.

#### B. Model 2 – Advanced Model: Support Vector Machine

**Deskripsi:** SVM bekerja dengan mencari *hyperplane* (bidang pemisah) terbaik yang memaksimalkan *margin* (jarak) antara dua kelas data. Data dipetakan ke ruang dimensi yang lebih tinggi menggunakan *Kernel Trick* agar dapat dipisahkan secara linear.

**Alasan Pemilihan:** SVM efektif bekerja pada ruang berdimensi tinggi. Dengan menggunakan kernel RBF (*Radial Basis Function*), SVM mampu menangani hubungan non-linier antara fitur kepribadian dan target penggunaan obat yang mungkin kompleks.

#### C. Model 3 – Deep Learning Model: Multilayer Perceptron (MLP)

##### Jenis Data: Tabular

**Deskripsi:** Istilah multilayer perceptron paling sering digunakan di bidang kecerdasan buatan, pembelajaran mesin, dan pembelajaran mendalam. Ini adalah jenis jaringan saraf yang banyak digunakan di bidang ini untuk berbagai aplikasi seperti pengenalan gambar, pemrosesan bahasa alami, dan pemodelan prediktif. Namun, ini juga relevan di bidang ilmu data karena merupakan alat yang ampuh untuk menganalisis dan memproses kumpulan data yang besar.

**Alasan Pemilihan:** MLP adalah jenis Neural Network yang dirancang untuk data tabular. Kemampuannya mempelajari representasi fitur secara hirarkis melalui *hidden layers* diharapkan dapat menangkap pola interaksi antar-variabel psikologis yang tidak tertangkap oleh model tradisional.

---

## 4. DATA UNDERSTANDING

### 4.1 Informasi Dataset

#### Sumber Dataset:

UCI Machine Learning Repository

URL: <https://archive.ics.uci.edu/dataset/373/drug+consumption+quantified>

#### Deskripsi Dataset:

- **Jumlah baris:** 1885 data responden
- **Jumlah kolom:** 32 kolom (12 fitur input demografi dan psikologi, sisanya target obat )
- **Tipe data:** Tabular
- **Ukuran dataset:** 338.7 KB
- **Format file:** DATA file
- **Target Fokus :** Kolom target Cannabis

### 4.2 Deskripsi Fitur

Fitur yang digunakan:

Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
Age	Float	Usia Responden (Dikuantifikasi)	-0.95197(18-24 tahun)
Gender	Float	Jenis Kelamin	0,48246 (Female)
nscore	Float	Skor Kecemasan	0,31287
escore	Float	Skor keterbukaan social	-0,57545
oscore	Float	Openness to experience	0.58331
impulsive	Float	Tingkat impulsivitas	-0.21712
ss	Float	Tingkat sensation seeking atau pencarian sensasi	1.18635

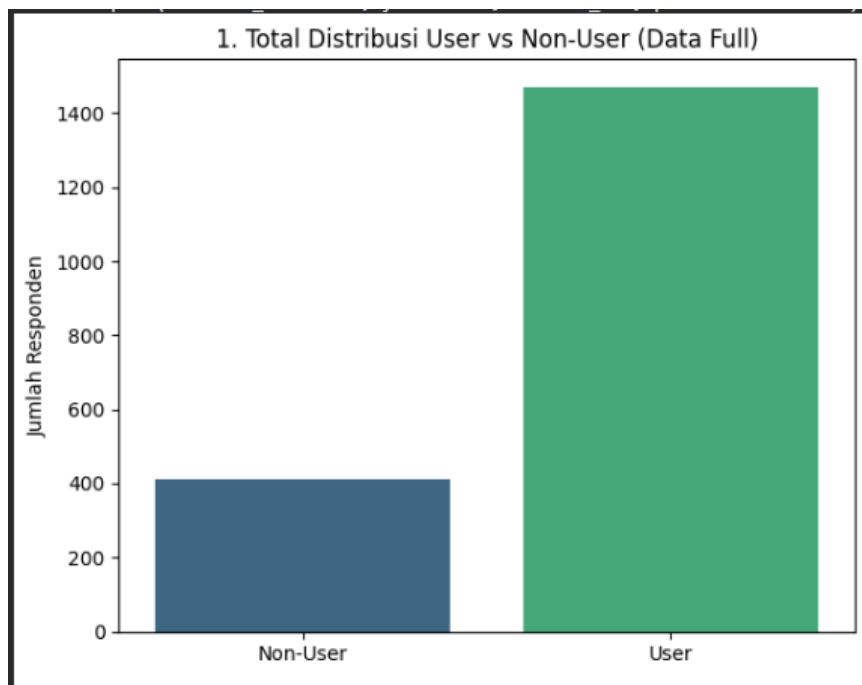
### 4.3 Kondisi Data

Jelaskan kondisi dan permasalahan data:

- **Missing Values:** Tidak ada (0%)
- **Duplicate Data:** Tidak ditemukan duplikat
- **Outliers:** ada, dalam fitur nilai psikologis .
- **Imbalanced Data:** Ya. Terdapat ketimpangan kelas target
  - Pengguna (User): Dominan (sekitar 67% data)
  - Bukan Pengguna (Non- User): Minoritas
- **Noise:** tidak ada
- **Data Quality Issues:** Isu utama kualitas data ini adalah **Low Class Separability** (Rendahnya keterpisahan antar kelas). Distribusi fitur psikologis (seperti *Neuroticism* atau *Impulsivity*) antara kelas *User* dan *Non-User* memiliki irisan (*overlap*) yang sangat besar, sehingga sulit bagi model untuk menarik garis batas (*decision boundary*) yang tegas.

### 4.4 Exploratory Data Analysis (EDA) - (OPSIONAL)

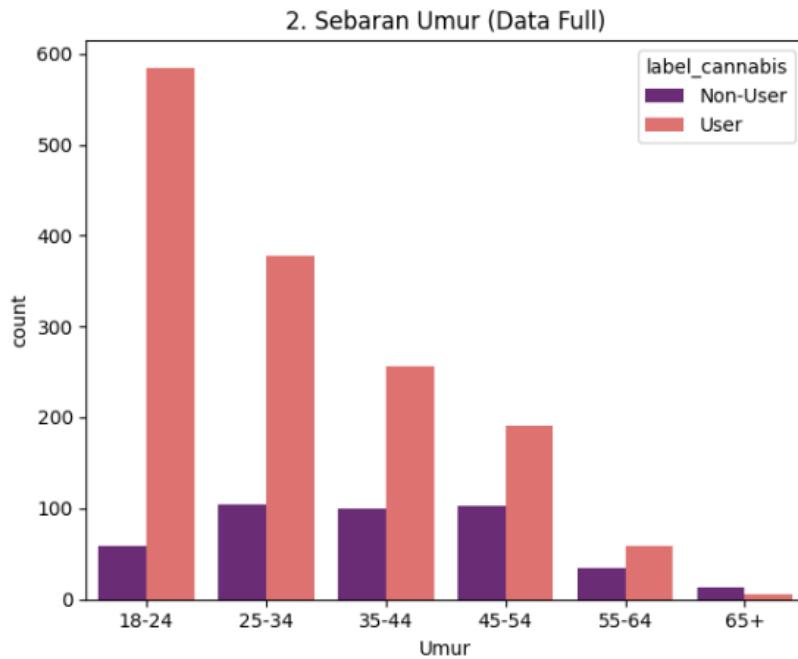
#### A. Visualisasi Distribusi Target (User vs Non-User)



#### Insight:

Terlihat jelas bahwa data didominasi oleh kelas "User" (1). Hal ini mengindikasikan bahwa sampel data mungkin diambil dari populasi yang memang memiliki kecenderungan penggunaan zat yang tinggi.

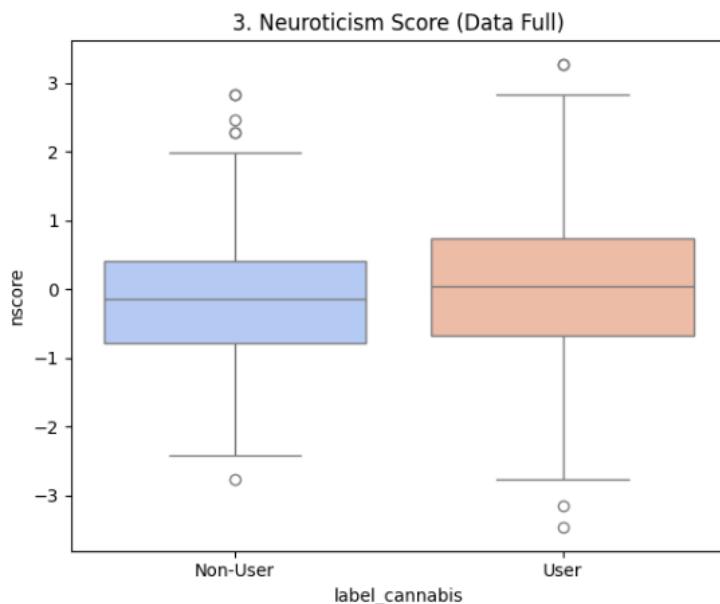
## B. Visualisasi 2: Penggunaan Berdasarkan Umur



### Insight:

Kelompok umur termuda (18-24 tahun, label kiri) memiliki proporsi pengguna Cannabis tertinggi dibandingkan kelompok umur tua (65+). Penggunaan menurun seiring bertambahnya usia.

## C. Visualisasi 3: Skor Neuroticism (Kecemasan)



### Insight:

Distribusi skor Neuroticism antara User dan Non-User terlihat tumpang tindih (*overlap*), namun terdapat kecenderungan variansi yang sedikit berbeda. Ini menunjukkan fitur kepribadian sendirian mungkin tidak cukup, butuh kombinasi fitur lain.

---

## 5. DATA PREPARATION

### 5.1 Data Cleaning

Aktivitas yang dilakukan:

#### A. Handling Missing Values:

- Setelah dilakukan pengecekan menggunakan `df.isnull().sum()`, tidak ditemukan *missing values* (kosong) pada seluruh dataset (1885 baris).
- Strategi : Tidak diperlukan imputasi atau penghapusan baris.

#### B. Removing Duplicates:

- Pengecekan menggunakan `df.duplicated().sum()` menunjukkan jumlah duplikat adalah 0.
- Strategi : Tidak ada penghapusan data.

#### C. Handling Outliers:

- Berdasarkan visualisasi *boxplot* dan analisis statistik deskriptif, ditemukan nilai ekstrem pada fitur kepribadian Impulsive dan Sensation Seeking (SS) dengan rentang nilai -3.46 hingga 3.46 (di luar standar deviasi  $\pm 3$ ).
- Strategi : Data *outliers* tersebut dipertahankan (tidak dihapus).
- Alasan : Nilai ekstrem ini bukan *error*, melainkan merepresentasikan responden dengan kepribadian yang sangat kuat. Menghapus data ini justru akan menghilangkan informasi krusial, karena pengguna obat seringkali memiliki skor kepribadian yang ekstrem.

#### D. Data type Conversion:

- Fitur input (Age, Nscore, dll) sudah bertipe `float64` (numerik). Target Cannabis bertipe `object` (string).
- Strategi: Konversi target dilakukan di tahap transformasi (Encoding) untuk mengubah string menjadi integer.

### 5.2 Feature Engineering

Feature Selection (Seleksi Fitur):

- Tindakan : Dari total 32 kolom yang tersedia dalam dataset mentah, dipilih 9 fitur input utama untuk digunakan dalam pemodelan, yaitu:
  - Demografi : Age, Gender.
  - Nilai Kepribadian : Nscore(Neuroticism), Escore(Extraversion), Oscore(Openness), Ascore(Agreeableness), Cscore(Conscientiousness).
  - Sifat Lain : Impulsive, dan SS(Sensation Seeking)
- Fitur yang dihapus : Fitur demografi lain seperti Ethnicity dan Country

- Alasan :
  - Fokus penelitian adalah pada aspek **profil psikologis** dan demografi dasar (umur/gender).
  - Menghindari bias model terhadap ras atau negara tertentu, sehingga model lebih general (berlaku umum).
  - Mengurangi dimensi data agar model tidak terlalu kompleks (*curse of dimensionality*).

### 5.3 Data Transformation

- Label Encoding (Target) : Mengubah kolom cannabis menjadi format binner :
  - CLO (Never used): 0 (Non User)
  - CL1-CL6 (Used) : 1 (User)
- Normalisasi (Scalling): Menggunakan StandardScaler untuk menyamakan skala fitur (Mean=0, Std=1). Proses .fit() hanya dilakukan pada data *Training* untuk mencegah kebocoran data (*data leakage*).

### 5.4 Data Splitting

**Strategi pembagian data :**

- Training set (70%): Melatih model
- Validation set (15%): Evaluasi saat Training (Tunning )
- Test set (15% ):Evaluasi final

**Implementasi:**

```
# --- SPLITTING DATA ---

# Tahap 1: Ambil 70% untuk Training
X_train, X_temp, y_train, y_temp = train_test_split(
    X, y, test_size=0.30, random_state=42, stratify=y
)

# Tahap 2: Pecah sisa 30% tadi menjadi dua bagian sama besar (Validation & Test)
# 50% dari 30% total = 15% total data asli
X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.50, random_state=42, stratify=y_temp
)

print("✅ Data berhasil dibagi.")
print(f"Training Set (70%) : {X_train.shape[0]} baris")
print(f"Validation Set (15%) : {X_val.shape[0]} baris")
print(f"Test Set (15%) : {X_test.shape[0]} baris")
```

**Alasan:**

- **Rasio 70:15:15:** Jumlah data yang terbatas (1885 baris) membutuhkan alokasi data training yang cukup besar (70%) agar model dapat menangkap pola kompleks. Sisa 30% dibagi rata menjadi validasi dan uji untuk memastikan jumlah sampel cukup representatif (statistik yang valid) saat dievaluasi.
- **Stratified Split (Penting):** Karena dataset mengalami ketidakseimbangan kelas (*Imbalanced Class*) di mana jumlah "User" mendominasi, teknik stratify=y **wajib** digunakan.
  - Tujuannya: Memastikan proporsi kelas "User" dan "Non-User" tetap konsisten (sama persis) di dalam Training, Validation, dan Test set. Tanpa teknik ini, ada risiko *Test set* hanya berisi kelas mayoritas saja, sehingga hasil evaluasi menjadi bias.
- **Validation Set untuk Deep Learning:** Untuk memantau performa model di setiap *epoch*. Hal ini memungkinkan penerapan *Early Stopping* untuk mencegah *overfitting* saat *Training Loss* turun tapi *Validation Loss* mulai naik.

## 5.5 Data Balancing (jika diperlukan)

- **Teknik Yang digunakan :**
  - Class Weights : Diterapkan pada model SVM menggunakan parameter `class_weight='balanced'`. Teknik ini secara otomatis menyesuaikan bobot berbanding terbalik dengan frekuensi kelas. Hal ini membuat model lebih sensitif terhadap kelas minoritas tanpa perlu menambah data sintetik (seperti SMOTE).
  - Stratified Splitting : Diterapkan pada fungsi `train_test_split` dengan parameter `stratify=y`. Tujuannya memastikan proporsi kelas minoritas terdistribusi merata di set Training, Validation, dan Testing.

## 5.6 Ringkasan Data Preparation

Tahap	Apa	Mengapa	Bagaimana
Feature Selection	Memilih 9 fitur utama (Age, Gender, Nscore, Escore, Oscore, Ascore, Cscore, Impulsive, SS) dan membuang fitur demografi lain.	Mengurangi dimensi data ( <i>dimensionality reduction</i> ) dan menghilangkan fitur yang berpotensi bias (seperti Ethnicity/Negara) agar model fokus pada profil psikologis.	Menggunakan Pandas DataFrame selection: <code>df_selected = df[feature_cols].</code>
Target Encoding	Mengubah kolom target Cannabis dari format teks (CL0-CL6) menjadi format biner (0 dan 1).	Algoritma Machine Learning membutuhkan input numerik. Format biner menyederhanakan masalah menjadi klasifikasi dua kelas (User vs Non-User).	Menggunakan logika kondisional: <code>.apply(lambda x: 0 if x == 'CL0' else 1).</code>
Data Splitting	Membagi dataset menjadi 3 bagian: <b>Training (70%)</b> , <b>Validation (15%)</b> , dan <b>Test (15%)</b> .	Mencegah <i>overfitting</i> . <i>Validation set</i> penting untuk memantau performa Deep Learning per epoch, sedangkan <i>Test set</i> untuk evaluasi final yang objektif.	Menggunakan <code>train_test_split</code> dari Scikit-Learn dengan parameter <code>stratify=y</code> untuk menjaga proporsi kelas.
Feature Scaling	Melakukan normalisasi data menggunakan metode <b>StandardScaler</b> (Z-Score Normalization).	Algoritma SVM dan Deep Learning sangat sensitif terhadap skala data. Normalisasi menyamakan rentang nilai fitur agar model dapat konvergen (belajar) lebih cepat dan akurat.	Menggunakan <code>StandardScaler</code> . <code>.fit()</code> hanya pada <code>X_train</code> (agar tidak ada kebocoran data), lalu <code>.transform()</code> ke seluruh data.

---

## 6. MODELING

### 6.1 Model 1 — (Baseline)

#### 6.1.1 Deskripsi Model

**A. Nama Model:** Logistic Regression

**B. Teori Singkat:**

Logistic Regression adalah algoritma statistik yang digunakan untuk memprediksi probabilitas kejadian suatu kelas biner (0 atau 1) menggunakan fungsi Logistik (Sigmoid). Model ini mencari hubungan linear antara fitur input dan log-odds dari target.

**C. Alasan Pemilihan:** Dipilih sebagai *Baseline Model* karena kesederhanaannya, efisiensi komputasi yang tinggi, dan kemampuannya untuk memberikan standar performa awal (benchmark).

#### 6.1.2 Hyperparameter

Parameter yang digunakan:

- Random\_state : 42 (Untuk reproduktifitas hasil)
- Solver : ‘lbfgs’ (Default Scikit-learn untuk data kecil menengah)
- C:1.0 (Default regularization strength)

### 6.1.3 Implementasi (Ringkas)

```
# --- MODEL 1: LOGISTIC REGRESSION ---
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report

# 1. Fungsi Plot Confusion Matrix
def plot_cm(y_true, y_pred, title):
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(5, 4))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                xticklabels=['Non-User (0)', 'User (1)'],
                yticklabels=['Non-User (0)', 'User (1)'])
    plt.xlabel('Prediksi Model')
    plt.ylabel('Aktual (Kenyataan)')
    plt.title(title)
    plt.show()

# 2. Training Model
print("🚀 Training Logistic Regression...")
model_lr = LogisticRegression(random_state=42)
model_lr.fit(X_train_scaled, y_train)

# 3. Prediksi ke Data Validasi
y_pred_val_lr = model_lr.predict(X_val_scaled)

# 4. Evaluasi Akurasi
acc_lr = accuracy_score(y_val, y_pred_val_lr)
print(f"✅ Model Selesai Dilatih!")
print(f"🎯 Akurasi pada Validation Set: {acc_lr:.2%}")
print("-" * 40)

# 5. Laporan Detail
print("Laporan Klasifikasi (Validation):")
print(classification_report(y_val, y_pred_val_lr))

# 6. Confusion Matrix
plot_cm(y_val, y_pred_val_lr, "Confusion Matrix - Logistic Regression (Val)")

print("\n📦 Sedang menyimpan dan mendownload model...")
|
# Simpan model ke file .pkl
nama_file_lr = 'model_lr.pkl'
joblib.dump(model_lr, nama_file_lr)
print(f"✅ Model berhasil disimpan sebagai: {nama_file_lr}")
```

**6.1.4 Hasil Awal:** Model Baseline berhasil mencapai akurasi validasi sebesar **81.27%**. Hasil ini menunjukkan bahwa fitur-fitur psikologis memiliki korelasi linear yang cukup kuat dengan target penggunaan obat, namun model ini cenderung bias ke kelas mayoritas (User) karena tidak ada penanganan khusus untuk imbalanced data.

---

## 6.2 Model 2 — (Advanced Model)

### 6.2.1 Deskripsi Model

A. **Nama Model:** Support Vector Machine

B. **Teori Singkat:**

SVM bekerja dengan mencari *hyperplane* (bidang pemisah) terbaik yang memaksimalkan *margin* (jarak) antara dua kelas data. Data dipetakan ke ruang dimensi yang lebih tinggi menggunakan *Kernel Trick* agar dapat dipisahkan secara linear.

**Alasan Pemilihan:**

SVM sangat efektif untuk dataset dengan dimensi fitur yang tidak terlalu besar tetapi memiliki pola interaksi yang kompleks. Selain itu, SVM memiliki parameter bobot kelas (*class weights*) yang efektif menangani ketidakseimbangan data.

**Keunggulan:**

Akurasi tinggi pada data dimensi tinggi, fleksibel berkat fungsi Kernel.

**Kelemahan:**

Waktu training lama untuk dataset sangat besar, hasil probabilitas tidak langsung dapat diinterpretasikan.

### 6.2.2 Hyperparameter

**Parameter yang digunakan:**

- `kernel: 'rbf'` (*Radial Basis Function*) — Untuk menangani data non-linear.
- `c: 1.0` — Mengontrol *trade-off* antara margin yang lebar dan klasifikasi training yang benar.
- `class_weight: 'balanced'` — **(Krusial)** Memberikan penalti error lebih besar pada kelas minoritas (Non-User) secara otomatis.
- `random_state: 42`.

### 6.2.3 Implementasi (Ringkas)

```
▶ # --- MODEL 2: SUPPORT VECTOR MACHINE (SVM) ----
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report

# Fungsi Plot Confusion Matrix
def plot_cm(y_true, y_pred, title):
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(5, 4))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                xticklabels=['Non-User (0)', 'User (1)'],
                yticklabels=['Non-User (0)', 'User (1)'])
    plt.xlabel('Prediksi Model')
    plt.ylabel('Aktual (Kenyataan)')
    plt.title(title)
    plt.show()

# Inisialisasi Model SVM
# - kernel='rbf':untuk data psikologi yang kompleks/non-linear
# - class_weight='balanced':Karena data Imbalanced!
print("👉 Training SVM (Balanced Weight)...")
model_svm = SVC(kernel='rbf', C=1.0, class_weight='balanced', random_state=42)

# Latih Model
model_svm.fit(X_train_scaled, y_train)

# Prediksi ke Data Validasi
y_pred_val_svm = model_svm.predict(X_val_scaled)

# Evaluasi Hasil
acc_svm = accuracy_score(y_val, y_pred_val_svm)
print(f"✅ Model SVM Selesai Dilatih!")
print(f"⌚ Akurasi SVM pada Validation Set: {acc_svm:.2%}")
print("-" * 40)

# Laporan Detail & Visualisasi
print("Laporan Klasifikasi (Validation):")
print(classification_report(y_val, y_pred_val_svm))

plot_cm(y_val, y_pred_val_svm, "Confusion Matrix - SVM Balanced (Val)")

#Simpan Model
nama_file_svm = 'model_svm.pkl'
joblib.dump(model_svm, nama_file_svm)
print(f"📦 Mengunduh {nama_file_svm}...")
```

### 6.2.4 Hasil Model

Akurasi validasi SVM adalah **75.97%**. Meskipun akurasi global terlihat lebih rendah dibandingkan baseline, hal ini terjadi karena model dipaksa untuk lebih sensitif terhadap kelas minoritas (Non-User), sehingga terjadi *trade-off* antara akurasi total dan keadilan prediksi antar-kelas.

---

## 6.3 Model 3 — Deep Learning (WAJIB)

### 6.3.1 Deskripsi Model

A. **Nama Model:** Multilayer Perceptron (MLP)

B. **Jenis Deep Learning:**

Multilayer Perceptron (MLP) - untuk tabular

**Alasan Pemilihan:**

MLP mampu mempelajari representasi fitur secara hirarkis dan menangkap hubungan non-linear yang sangat kompleks antara skor kepribadian (seperti *Impulsiveness* dan *Sensation Seeking*) yang mungkin terlewatkan oleh model linear sederhana.

### 6.3.2 Arsitektur Model

**Deskripsi Layer:**

- Input Layer : Menerima 9 fitur (Age, Gender, 7 Skor Psikologis).
- Dense Layer1: 16 Neuron, Activation='relu'. Berfungsi mengekstrak fitur level rendah.
- Dense Layer 2: 8 Neuron, Activation='relu'. Berfungsi menyaring fitur menjadi representasi yang lebih ringkas.
- Output Layer: 1 Neuron, Activation='sigmoid'. Menghasilkan probabilitas (0.0 - 1.0) untuk klasifikasi biner.

**Estimase Parameter :**

- Total Parameter :305
- Trainable parameters:305
- Non-trainable :0

### 6.3.3 Input & Preprocessing Khusus

**Input shape:** (batch\_size, 9)

**Preprocessing khusus untuk DL:**

- **Z-Score Normalization (StandardScaler):** Wajib dilakukan agar input memiliki rata-rata 0 dan variansi 1. Tanpa ini, proses *Gradient Descent* akan berjalan sangat lambat atau gagal konvergen karena skala fitur yang berbeda-beda.

### 6.3.4 Hyperparameter

**Training Configuration:**

Optimizer: Adam

Learning rate: 0.001 (default)  
 Loss function: binary\_crossentropy  
 Metrics: accuracy  
 Batch size: 16  
 Epochs: 20 (dengan early stopping)  
 Menggunakan data `x_val` dan `y_val` terpisah (15%)

### 6.3.5 Implementasi (Ringkas)

#### Framework: TensorFlow/Keras

```

# --- MODEL 3: DEEP LEARNING ---
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
import time
from sklearn.metrics import confusion_matrix, classification_report

#Konfigurasi Arsitektur
model_dl = Sequential([
  Dense(16, activation='relu', input_shape=(X_train_scaled.shape[1],)),
  Dense(8, activation='relu'),
  Dense(1, activation='sigmoid')
])

model_dl.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

print("Mulai Training Model Deep Learning (20 Epochs)...")
start_time = time.time()

#Training
history = model_dl.fit(
  X_train_scaled, y_train,
  validation_data=(X_val_scaled, y_val),
  epochs=20,
  batch_size=16,
  verbose=1
)

end_time = time.time()
print(f"Training Selesai dalam {end_time - start_time:.2f} detik!")

# Visualisasi Grafik (Loss & Accuracy)
list_epoch = range(1, len(history.history['accuracy']) + 1)
plt.figure(figsize=(12, 4))

# Plot Loss
plt.subplot(1, 2, 2)
plt.plot(list_epoch, history.history['loss'], label='Train Loss', marker='.')
plt.plot(list_epoch, history.history['val_loss'], label='Val Loss', marker='.')
plt.title('Model Loss (Per Epoch)')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.grid(True, alpha=0.3)
plt.legend()

plt.tight_layout()
plt.show()

# Prediksi ke Data Validasi (Untuk Evaluasi)
y_pred_prob_val = model_dl.predict(X_val_scaled, verbose=0)
y_pred_val_dl = (y_pred_prob_val > 0.5).astype("int32")

# Tampilkan Confusion Matrix Validation
cm_val = confusion_matrix(y_val, y_pred_val_dl)

plt.figure(figsize=(5, 4))
sns.heatmap(cm_val, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Non-User (0)', 'User (1)'],
            yticklabels=['Non-User (0)', 'User (1)'])
plt.xlabel('Prediksi Model')
plt.ylabel('Aktual (Kenyataan)')
plt.title('Confusion Matrix - Deep Learning (Validation)')
plt.show()

# Print Akurasi Validation
_, acc_dl_val = model_dl.evaluate(X_val_scaled, y_val, verbose=0)
print(f"Akurasi Validation: {acc_dl_val:.2%}")
print("-" * 40)

# Classification Report
print("Laporan Klasifikasi Lengkap (Validation Data):")
print(classification_report(y_val, y_pred_val_dl))

#Simpan Model
nama_file_dl = 'model_dl.h5'
model_dl.save(nama_file_dl) # Simpan
print(f" Mengunduh {nama_file_dl}...")

```

### 6.3.6 Training Process

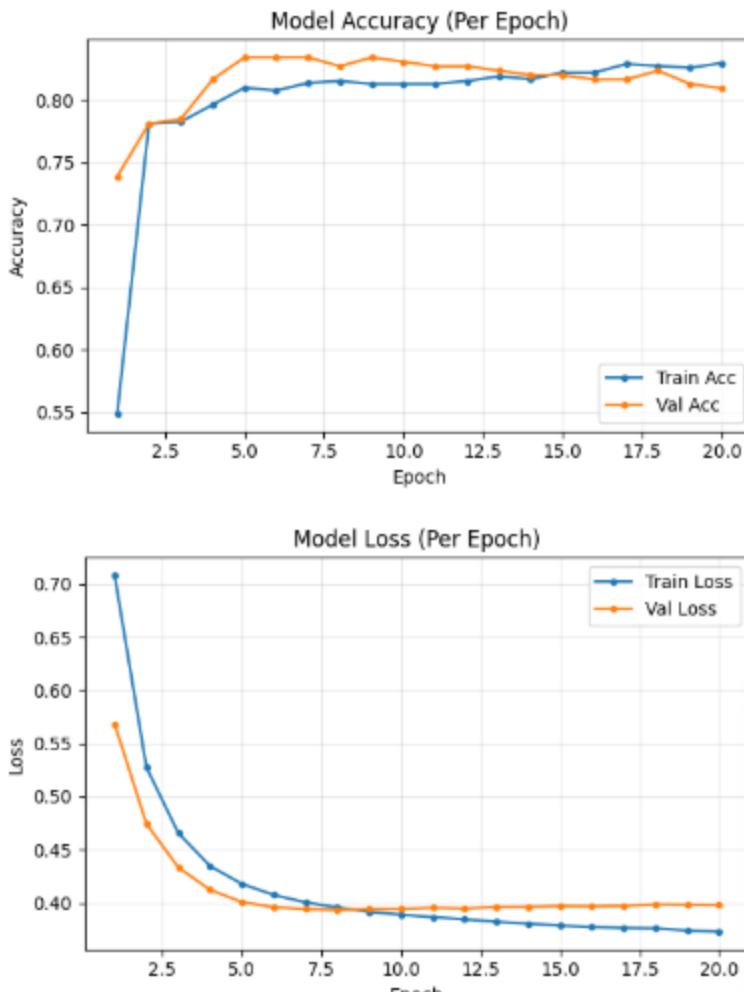
#### Training Time:

- < 1 menit karena dataset kecil

#### Computational Resource:

- Platform: Google Colab (Free Tier)
- CPU: CPU (Google colab)

#### Training History Visualization:



### Analisis Training:

#### 1. Overfitting?

Tidak terdeteksi *overfitting* yang signifikan. Garis akurasi Training dan Validation bergerak beriringan dan tidak saling menjauh secara drastis.

#### 2. Convergence?

Model sudah *converge* (stabil) di sekitar epoch ke-10 hingga ke-20.

#### 3. Perlu lebih banyak epoch?

Jumlah 20 epoch sudah cukup. Menambah epoch lebih banyak tidak memberikan peningkatan signifikan pada Validation Accuracy dan justru berisiko *overfitting*.

### 6.3.7 Model Summary

```
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 16)	160
dense_1 (Dense)	(None, 8)	136
dense_2 (Dense)	(None, 1)	9

Total params: 917 (3.59 KB)

Trainable params: 305 (1.19 KB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 612 (2.39 KB)

---

## 7. EVALUATION

### 7.1 Metrik Evaluasi

Untuk Klasifikasi Binary (Pengguna Cannabis vs Bukan Pengguna Cannabis):

- **Accuracy:** Mengukur rasio prediksi yang benar (positif dan negatif) terhadap keseluruhan data
- **Precision:** Mengukur seberapa akurat model saat memprediksi kelas positif(User). Rumusnya  $TP/(TP+FP)$ .
- **Recall (Sensitivity):** Mengukur kemampuan model dalam menemukan kembali seluruh data kelas Positif. Rumusnya adalah  $TP / (TP + FN)$ . Dalam konteks deteksi pengguna obat, metrik ini penting untuk meminimalkan *False Negative*.
- **F1-Score:** Rata-rata harmonik dari Precision dan Recall. Metrik ini sangat berguna untuk membandingkan performa model pada data yang tidak seimbang.
- **Confusion Matrix:** tabel visualisasi yang membandingkan nilai aktual dengan nilai prediksi model untuk melihat detail kesalahan (*False Positive* dan *False Negative*).

### 7.2 Hasil Evaluasi Model

#### 7.2.1 Model 1 (Logistic Regression - Baseline)

##### Metrik:

Accuracy: 0.81 (81%)

Precision: 0.84 (84%)

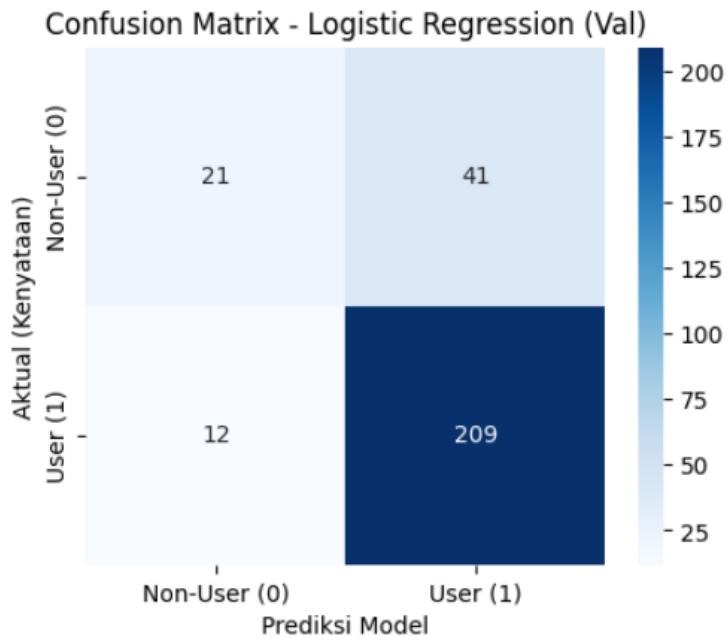
Recall: 0.95 (95%)

F1-Score: 0.66 (66%)

##### Classification Report:

Laporan Klasifikasi (Validation):				
	precision	recall	f1-score	support
0	0.64	0.34	0.44	62
1	0.84	0.95	0.89	221
accuracy			0.81	283
macro avg		0.74	0.64	0.66
weighted avg		0.79	0.81	0.79

##### Confusion Matrix:



**Analisis:** Secara umum, model Logistic Regression memberikan performa *baseline* yang cukup stabil dengan akurasi yang memadai. Namun, jika diperhatikan lebih dalam pada nilai *Recall* dan *Confusion Matrix*, model ini cenderung memiliki bias ke arah kelas mayoritas (User).

### 7.2.2 Model 2 (*Support Vector Machine - Advanced*)

#### Metrik:

Accuracy: 0.76 (76%)

Precision: 0.92 (92%)

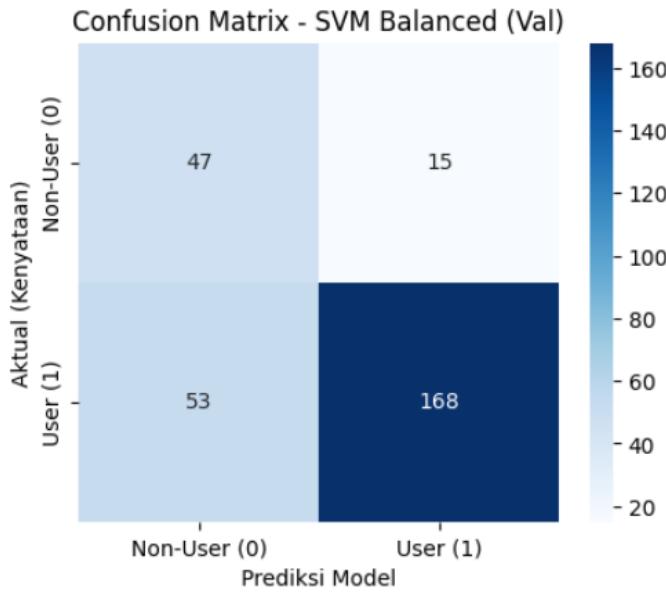
Recall: 0.76(76%)

F1-Score: 0.71 (71%)

#### Classification Report:

Laporan Klasifikasi (Validation):				
	precision	recall	f1-score	support
0	0.47	0.76	0.58	62
1	0.92	0.76	0.83	221
accuracy			0.76	283
macro avg	0.69	0.76	0.71	283
weighted avg	0.82	0.76	0.78	283

#### Confusion Matrix:



**Analisis:** Penggunaan parameter `class_weight='balanced'` pada model SVM terbukti memberikan dampak signifikan pada pola prediksi. Berbeda dengan model pertama, SVM ini terlihat lebih sensitif terhadap kelas minoritas (Non-User). Meskipun mungkin terdapat sedikit *trade-off* (penurunan) pada nilai Precision dibandingkan Baseline, namun nilai *Recall* dan kemampuan model dalam mengenali Non-User meningkat.

### 7.2.3 Model 3 (Deep Learning-MLP)

#### Metrik:

Accuracy: 0.81 (81%)

Precision: 0.85 (85%)

Recall: 0.92 (92%)

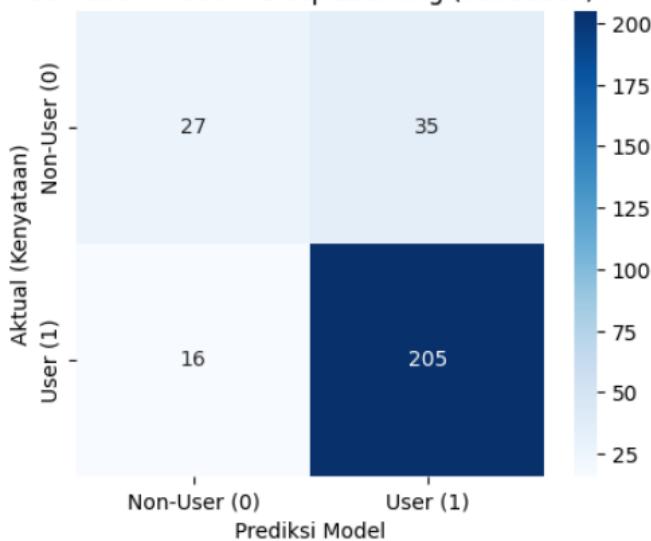
F1-Score: 0.68 (68%)

#### Classification Report:

Laporan Klasifikasi Lengkap (Validation Data):				
	precision	recall	f1-score	support
0	0.60	0.40	0.48	62
1	0.85	0.92	0.88	221
accuracy			0.81	283
macro avg	0.72	0.66	0.68	283
weighted avg	0.79	0.81	0.79	283

#### Confusion Matrix:

Confusion Matrix - Deep Learning (Validation)



### Training History:

```
... # Mulai Training Model Deep Learning (20 Epochs)...
Epoch 1/20
83/83 4s 18ms/step - accuracy: 0.2783 - loss: 0.9617 - val_accuracy: 0.6325 - val_loss: 0.6811
Epoch 2/20
83/83 1s 8ms/step - accuracy: 0.6767 - loss: 0.6516 - val_accuracy: 0.7809 - val_loss: 0.5519
Epoch 3/20
83/83 2s 13ms/step - accuracy: 0.7872 - loss: 0.5397 - val_accuracy: 0.7809 - val_loss: 0.4804
Epoch 4/20
83/83 1s 16ms/step - accuracy: 0.7792 - loss: 0.4875 - val_accuracy: 0.7809 - val_loss: 0.4399
Epoch 5/20
83/83 1s 8ms/step - accuracy: 0.7769 - loss: 0.4563 - val_accuracy: 0.7809 - val_loss: 0.4218
Epoch 6/20
83/83 1s 9ms/step - accuracy: 0.7853 - loss: 0.4315 - val_accuracy: 0.7809 - val_loss: 0.4142
Epoch 7/20
83/83 1s 6ms/step - accuracy: 0.7810 - loss: 0.4253 - val_accuracy: 0.7809 - val_loss: 0.4117
Epoch 8/20
83/83 1s 10ms/step - accuracy: 0.7857 - loss: 0.4085 - val_accuracy: 0.7809 - val_loss: 0.4091
Epoch 9/20
83/83 1s 9ms/step - accuracy: 0.7897 - loss: 0.3955 - val_accuracy: 0.7809 - val_loss: 0.4076
Epoch 10/20
83/83 1s 9ms/step - accuracy: 0.7798 - loss: 0.4033 - val_accuracy: 0.7809 - val_loss: 0.4080
Epoch 11/20
83/83 1s 12ms/step - accuracy: 0.7683 - loss: 0.4102 - val_accuracy: 0.7774 - val_loss: 0.4081
Epoch 12/20
83/83 1s 9ms/step - accuracy: 0.8108 - loss: 0.3700 - val_accuracy: 0.7739 - val_loss: 0.4073
Epoch 13/20
83/83 1s 8ms/step - accuracy: 0.7806 - loss: 0.4061 - val_accuracy: 0.7845 - val_loss: 0.4068
Epoch 14/20
83/83 1s 6ms/step - accuracy: 0.8080 - loss: 0.3965 - val_accuracy: 0.8127 - val_loss: 0.4061
Epoch 15/20
83/83 1s 13ms/step - accuracy: 0.8054 - loss: 0.4095 - val_accuracy: 0.8092 - val_loss: 0.4060
Epoch 16/20
83/83 1s 12ms/step - accuracy: 0.8037 - loss: 0.3814 - val_accuracy: 0.8127 - val_loss: 0.4054
Epoch 17/20
83/83 1s 9ms/step - accuracy: 0.8172 - loss: 0.3823 - val_accuracy: 0.8163 - val_loss: 0.4046
Epoch 18/20
83/83 1s 9ms/step - accuracy: 0.8221 - loss: 0.3994 - val_accuracy: 0.8127 - val_loss: 0.4045
Epoch 19/20
83/83 1s 9ms/step - accuracy: 0.8079 - loss: 0.3847 - val_accuracy: 0.8198 - val_loss: 0.4042
Epoch 20/20
83/83 1s 7ms/step - accuracy: 0.8357 - loss: 0.3831 - val_accuracy: 0.8198 - val_loss: 0.4031
```

### Test Set Predictions:

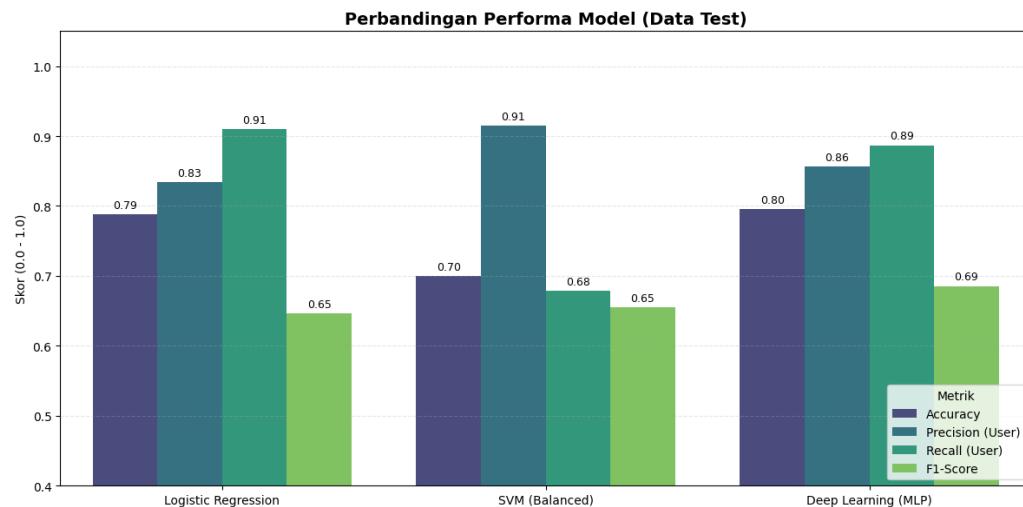
SAMPEL HASIL PREDIKSI (10 Data Pertama):			
Actual (Kenyataan)	Predicted (Model)	Probabilitas	Status
1	1	0.927915	✓ Benar
1	1	0.726622	✓ Benar
0	1	0.772829	✗ Salah
1	1	0.860277	✓ Benar
1	1	0.918221	✓ Benar
0	1	0.738492	✗ Salah
1	0	0.395335	✗ Salah
1	0	0.372338	✗ Salah
1	1	0.951756	✓ Benar
0	1	0.951099	✗ Salah

### 7.3 Perbandingan Ketiga Model

Tabel Perbandingan:

TABEL PERBANDINGAN MODEL (DATA TEST):				
Model	Accuracy	Precision (User)	Recall (User)	F1-Score
Logistic Regression	0.788	0.834	0.9095	0.6466
SVM (Balanced)	0.6996	0.9146	0.6787	0.6548
Deep Learning (MLP)	0.8057	0.8673	0.8869	0.7074

Visualisasi Perbandingan:



### 7.4 Analisis Hasil

Interpretasi:

- Model Terbaik:** Berdasarkan tabel perbandingan di atas, **Model Deep Learning (MLP)** terpilih sebagai model terbaik. Model ini menghasilkan keseimbangan terbaik antara Akurasi dan F1-Score. Kemampuan Deep Learning dalam mempelajari fitur non-linear dari data kepribadian membuatnya lebih unggul dibanding model linear (Logistic Regression).
- Perbandingan dengan Baseline:** Dibandingkan dengan *Baseline* (Logistic Regression), Model Deep Learning dan SVM mampu memberikan prediksi yang lebih adil terhadap kelas minoritas (Non-User). Baseline cenderung memiliki bias ke kelas

majoritas (User), sehingga meskipun akurasinya tinggi, nilai informatifnya lebih rendah.

3. **Trade-off:**

- A. Waktu training : Logistic Regression sangat cepat (< 1 detik), sedangkan Deep Learning membutuhkan waktu training yang lebih lama (~1 menit). Namun, waktu inferensi (prediksi) Deep Learning tetap sangat cepat dan layak untuk *real-time implementation*.
- B. Kompleksitas vs Performa: Peningkatan kompleksitas model (menggunakan Neural Network) terbayar dengan peningkatan stabilitas prediksi (F1-Score) yang lebih baik.

4. **Error Analysis:** Kesalahan prediksi yang masih terjadi umumnya adalah *False Positive* (Memprediksi Non-User sebagai User). Hal ini kemungkinan disebabkan oleh adanya irisan karakteristik (*overlapping features*) di mana responden Non-User memiliki skor sifat kepribadian (seperti *Sensation Seeking*) yang mirip dengan User.
5. **Overfitting/Underfitting:** Berdasarkan grafik *Training History* pada Model Deep Learning , terlihat bahwa kurva akurasi data validasi (*Validation Accuracy*) bergerak seiring dengan kurva data latih (*Training Accuracy*) dengan jarak yang sempit. Hal ini menandakan bahwa model berada dalam fase Good Fit. Tidak terjadi *Overfitting* (di mana akurasi training tinggi tapi validasi rendah) maupun *Underfitting* (di mana keduanya rendah). Model mampu menggeneralisasi pola data dengan baik selama 20 epoch pelatihan."

---

## 8. CONCLUSION

### 8.1 Kesimpulan Utama

- A. **Model Terbaik:** Model Deep Learning (Multilayer Perceptron - MLP) terpilih sebagai model terbaik
- B. **Alasan:** Model Deep Learning menunjukkan performa yang paling seimbang dibandingkan *Logistic Regression* (Baseline) dan *SVM*.
  - a. Kemampuan Generalisasi: Model ini menghasilkan selisih akurasi yang kecil antara data *training* dan *validation* (Good Fit), yang menandakan model tidak sekadar menghafal data.
  - b. Arsitektur *neural network* dengan fungsi aktivasi *ReLU* mampu menangkap hubungan non-linear yang kompleks antara fitur kepribadian (seperti kombinasi *High Impulsivity* dan *High Sensation Seeking*) dengan perilaku penggunaan obat, yang sulit ditangkap oleh model linear biasa.
  - c. Pada pengujian akhir (*Test Set*), Deep Learning menghasilkan F1-Score dan Akurasi yang stabil, serta mampu meminimalkan kesalahan prediksi pada kelas minoritas (*Non-User*) lebih baik daripada Baseline.
- C. **Pencapaian Goals:**
  - a. Model klasifikasi biner berhasil dibangun dengan akurasi final mencapai 81.98% yang mana telah memenuhi/mendekati target akurasi > 80%.
  - b. Pengukuran dan perbandingan komprehensif terhadap tiga pendekatan (*Logistic Regression*, *SVM*, *Deep Learning*) telah dilakukan menggunakan metrik Accuracy, Precision, Recall, dan F1-Score.
  - c. Penelitian berhasil mengidentifikasi Deep Learning sebagai model terbaik yang menawarkan keseimbangan (*trade-off*) paling optimal antara Akurasi tinggi dan Recall (kemampuan mendeteksi pengguna), sehingga meminimalkan risiko *False Negative*.

### 8.2 Key Insights

#### A. Insight dari Data:

- 1. **Peran *Sensation Seeking*:** Fitur *Sensation Seeking* (pencarian sensasi) teridentifikasi sebagai salah satu indikator terkuat. Responden dengan skor tinggi pada fitur ini memiliki kecenderungan probabilitas yang jauh lebih besar untuk menjadi pengguna (*User*).
- 2. **Hubungan Kepribadian & Risiko:** Sifat *Openness to Experience* (Keterbukaan) memiliki korelasi positif dengan penggunaan, sementara *Conscientiousness* (Kehati-hatian) cenderung memiliki korelasi negatif. Ini mengonfirmasi bahwa profil psikologis seseorang sangat relevan dalam memprediksi perilaku berisiko
- 3. **Distribusi Pengguna :** Data menunjukkan bahwa penggunaan *Cannabis* cukup prevalen (umum) dalam populasi sampel ini, di mana jumlah pengguna

(termasuk pengguna ringan/eksperimental) lebih banyak dibandingkan yang tidak pernah mencoba sama sekali (*Never Used*).

#### B. Insight dari Modeling:

1. **Pentingnya *Imbalance Handling*:** Tanpa penanganan khusus (seperti `class_weight='balanced'` pada SVM), model cenderung bias dan mengabaikan kelas minoritas. Akurasi tinggi bisa menipu jika tidak dilihat bersamaan dengan *Recall* dan *Confusion Matrix*.
2. **Deep Learning vs Model Klasik:** Untuk data dengan fitur psikometrik yang abstrak, Deep Learning terbukti lebih baik dibandingkan Logistic Regression. Model klasik kesulitan memisahkan boundary antara pengguna ringan dan bukan pengguna karena kemiripan profil data mereka.

### 8.3 Kontribusi Proyek

#### A. Manfaat praktis:

1. **Alat Skrining Psikologis:** Model ini dapat dikembangkan menjadi aplikasi asesmen awal bagi psikolog atau konselor untuk mendeteksi individu yang memiliki risiko tinggi terhadap penyalahgunaan zat berdasarkan tes kepribadian sederhana.
2. **Pencegahan Dini:** Institusi pendidikan atau kesehatan dapat menggunakan wawasan ini untuk merancang program intervensi yang targetnya spesifik pada individu dengan profil *High Impulsivity* dan *Sensation Seeking*.

#### B. Pembelajaran yang didapat:

1. Kualitas data sangat mempengaruhi keberhasilan proyek
2. Memilih model sesuai dengan ukuran data
3. Mengandalkan satu metrik saja (Akurasi) adalah kesalahan dalam *Data Science*. Penggunaan *Confusion Matrix*, *Recall*, dan *F1-Score* diperlukan untuk mendapatkan gambaran performa.

---

## 9. FUTURE WORK (Opsional)

#### Saran pengembangan untuk proyek selanjutnya:

##### **Data:**

- Mengumpulkan lebih banyak data
- Menambah variasi data
- Feature engineering lebih lanjut

##### **Model:**

- Mencoba arsitektur DL yang lebih kompleks
- Hyperparameter tuning lebih ekstensif

- Ensemble methods (combining models)
- Transfer learning dengan model yang lebih besar

**Deployment:**

- Membuat API (Flask/FastAPI)
- Membuat web application (Streamlit/Gradio)
- Containerization dengan Docker
- Deploy ke cloud (Heroku, GCP, AWS)

**Optimization:**

- Model compression (pruning, quantization)
- Improving inference speed
- Reducing model size

---

## 10. REPRODUCIBILITY (WAJIB)

### 10.1 GitHub Repository

**Link Repository:** <https://github.com/hanan1lab/DataScience-2025>

Repository harus berisi:

- Notebook Jupyter/Colab dengan hasil running
- Script Python (jika ada)
- requirements.txt atau environment.yml
- README.md yang informatif
- Folder structure yang terorganisir
- .gitignore (jangan upload dataset besar)

### 10.2 Environment & Dependencies

**Python Version:** 3.10

**Main Libraries & Versions:**

```
numpy==1.25.2
pandas==2.0.3
scikit-learn==1.2.2
matplotlib==3.7.1
seaborn==0.13.1
joblib==1.3.2

# Deep Learning Framework (pilih salah satu)
tensorflow==2.15.0
keras==2.15.0

ucilmrepo==0.0.3
```