Tutorial Lengkap: Klasifikasi E1 dengan AI

1. Tujuan

Panduan ini bertujuan membantu pengguna dalam membuat sistem otomatis untuk mengklasifikasikan isi kolom 'Development Description' menjadi kategori Leadership Program, Technical Program, atau nama training menggunakan AI.

2. Persiapan Awal

Sebelum memulai coding, pastikan kamu telah menyiapkan hal berikut:

a. Instal Python 3.9 atau versi lebih baru.

b. Install editor seperti VS Code, Jupyter Notebook, atau Google Colab.

c. Pastikan sudah menginstall library berikut:

pip install pandas openpyxl transformers torch rapidfuzz **(running codingan di terminal bawah kiri/ di add code jupyter)**

3. Struktur Folder

Pastikan file tersimpan dengan struktur berikut:

```
D:\IDP FINAL\
│
├── IDP NEW.xlsx
├── IDP_Predicted_E1_fix.xlsx
└── E1_fixx.xlsx
```

4. Langkah-Langkah Coding

🧩 Bagian 1: Membaca File Excel

Kotak kode di bawah bisa diisi dengan script untuk membaca file Excel.

```
import pandas as pd

# Ganti path ke lokasi file Excel kamu jika perlu
file_path = "D:\\IDP FINAL\\IDP NEW.xlsx"

# Baca file Excel
df = pd.read_excel(file_path)

# Tampilkan 5 baris pertama
print(df.head())
```

file_path = "D:\\IDP FINAL\\IDP NEW.xlsx" **(noted: copy path ada di sebelah kiri bagian dokumen)**

🧠 Bagian 2: Klasifikasi E1 dengan AI

kotak ini dengan script utama klasifikasi menggunakan AI.

```
import os
import pandas as pd
from transformers import pipeline
from openpyxl import load_workbook
from rapidfuzz import fuzz  # untuk fuzzy matching

# ==================== KONFIGURASI ====================
file_path = "D:\\IDP FINAL\\IDP NEW.xlsx"
output_file = "IDP_Predicted_E1_fix.xlsx"
e1_column_name = "Development Description"

# ==================== CEK FILE ====================
if not os.path.exists(file_path):
    raise FileNotFoundError(f"❌ File tidak ditemukan: {file_path}")

# ==================== BACA EXCEL ====================
try:
    df = pd.read_excel(file_path)
except Exception as e:
    raise RuntimeError(f"❌ Gagal membaca Excel: {e}")

if e1_column_name not in df.columns:
    raise ValueError(f"❌ Kolom '{e1_column_name}' tidak ditemukan dalam file.")

descriptions = df[e1_column_name].astype(str).fillna("")

# ==================== CEK WARNA MERAH DI EXCEL ====================
try:
    wb = load_workbook(file_path)
    ws = wb.active
except Exception as e:
    raise RuntimeError(f"❌ Gagal membuka Excel untuk warna: {e}")

red_rows = []
col_idx = df.columns.get_loc(e1_column_name) + 1  # Excel 1-based indexing

for row in range(2, ws.max_row + 1):  # mulai baris kedua (skip header)
    cell = ws.cell(row=row, column=col_idx)
    if cell.fill and cell.fill.start_color and cell.fill.start_color.rgb:
        if cell.fill.start_color.rgb.upper() in ["FFFF0000", "00FF0000"]:
            red_rows.append(row - 2)  # simpan index untuk skip prediksi
```

```
# ==================== LABEL PROGRAM ====================
program_labels = [
    # Leadership
    "Essential Professional Program (EPP)",
    "Supervisory Development Program (SDP)",
    "Management Development Program (MDP)",
    "People Manager 101",
    "Leader as Coach",
    "Young Professional Program",
    "Advanced Development Program (ADP)",

    # Technical
    "Project Management Excellence (PMX)",
    "Maintenance Inspector Program",
    "Preventive Maintenance Engineers (PME)",
    "Cement Manufacturing Course",
    "Analyst Excellence (AX)",
    "Basic Maintenance",
    "Finance for non Finance",
    "Packer Excellence",
    "Patroller Excellence",
    "Kiln Operator Excellence (KOX)",
    "Mill Operation Excellence (MOX)"
]

# Normalisasi label untuk case-insensitive check
program_labels_lower = {label.lower(): label for label in program_labels}

# ==================== LOAD MODEL ====================
print("⏳ Memuat model multilingual zero-shot...")
try:
    classifier = pipeline(
        "zero-shot-classification",
        model="MoritzLaurer/multilingual-MiniLMv2-L6-mnli-xnli"
    )
except Exception as e:
    raise RuntimeError(f"❌ Gagal memuat model zero-shot: {e}")

print("⏳ Memuat model text2text untuk ekstraksi training...")
try:
    extractor = pipeline(
        "text2text-generation",
        model="google/flan-t5-small"   # lebih ringan tapi patuh
    )
except Exception as e:
    raise RuntimeError(f"❌ Gagal memuat model extractor: {e}")

# ==================== PREDIKSI ====================
```

```
e1_predictions = []

print("🔍 Memulai prediksi...")
for idx, desc in enumerate(descriptions):
    # ⛔ Skip baris dengan warna merah
    if idx in red_rows:
        e1_predictions.append("")
        continue

    desc_lower = desc.lower()

    # ---------- Step 1: Cek Program Labels (case-insensitive + fuzzy) ----------
    matched_label = ""
    best_score = 0
    for label in program_labels:
        score = fuzz.partial_ratio(label.lower(), desc_lower)
        if score > best_score:
            best_score = score
            matched_label = label

    if best_score >= 85:  # threshold fuzzy lebih ketat
        e1_predictions.append(matched_label)
        continue

    # ---------- Step 2: Deteksi apakah TRAINING ----------
    try:
        training_check = classifier(
            desc,
            candidate_labels=["training", "pelatihan", "program"],
            multi_label=False
        )
        top_label = training_check["labels"][0]
        top_score = training_check["scores"][0]
    except Exception:
        top_label, top_score = "program", 0

    # Kalau teks mengandung kata "training" langsung dianggap training
    if "training" in desc_lower or "pelatihan" in desc_lower or (top_label in ["training",
"pelatihan"] and top_score >= 0.45):
        # ---------- Step 3: Ekstraksi Nama Training ----------
        try:
            prompt = f"{desc}\n\nJawab hanya nama training:"
            extraction = extractor(prompt, max_new_tokens=50,
clean_up_tokenization_spaces=True)
            training_name = extraction[0]["generated_text"].strip()

            #  🔹 Post-processing → selalu format "training <nama>"
            training_name = training_name.replace("training", "", 1).replace("Training", "",
1).strip()
```

```
            if training_name.lower().startswith("jawab"):
                training_name = training_name.split(":")[-1].strip()
            training_name = f"training {training_name}"
        except Exception:
            training_name = "training umum"

        e1_predictions.append(training_name)
        continue

    # ------------------ Step 4: Jika semua gagal ------------------
    e1_predictions.append("")

# ==================== SIMPAN HASIL ====================
df["E1"] = e1_predictions

final_output = output_file
if os.path.exists(output_file):
    try:
        os.remove(output_file)  # hapus dulu biar gak PermissionError
    except:
        base, ext = os.path.splitext(output_file)
        final_output = f"{base}_new{ext}"

try:
    df.to_excel(final_output, index=False)
    print(f"✅ Selesai! Hasil disimpan di: {final_output}")
except Exception as e:
    raise RuntimeError(f"❌ Gagal menyimpan hasil: {e}")
```

Di copy saja ini tetapi di Ganti di bagian **(file_path = "D:\\IDP FINAL\\IDP NEW.xlsx")** seusaikan di mana letaknya atau bisa saja copy path di bagian kiri.

🧹 Bagian 3: Membersihkan Hasil Akhir

Script ini digunakan untuk membersihkan hasil prediksi dan menyimpan file hasil akhir.

```
import pandas as pd
import os

# ================== KONFIGURASI ==================
file_path = "D:\\IDP FINAL\\IDP_Predicted_E1_fix.xlsx"      # ubah sesuai lokasi file kamu
output_path = "D:\\IDP FINAL\\E1_fixx.xlsx"  # hasil setelah dibersihkan
kolom_target = "E1"                    # nama kolom target
teks_dihapus = "Jawab hanya nama :"        # teks yang ingin dihapus
# ================================================

# Pastikan folder output ada
```

```
os.makedirs(os.path.dirname(output_path), exist_ok=True)

# Baca file Excel
df = pd.read_excel(file_path)

# Periksa apakah kolom E1 ada
if kolom_target in df.columns:
    # Hapus teks target dari kolom E1
    df[kolom_target] = (
        df[kolom_target]
        .astype(str)
        .str.replace(r"\s*Jawab\s+hanya\s+nama\s*:\s*", "", regex=True, case=False)
        .str.strip()
    )

    # Ubah "nan" atau "NaN" hasil konversi string menjadi kosong
    df[kolom_target] = df[kolom_target].replace(["nan", "NaN"], "")

    print(f" ✅ Teks 'Jawab hanya nama :' berhasil dihapus dari kolom '{kolom_target}'.")
else:
    print(f" ⚠️ Kolom '{kolom_target}' tidak ditemukan di file Excel.")

# Simpan hasil ke file baru tanpa NaN
df.to_excel(output_path, index=False, na_rep="")
print(" 💾 File hasil disimpan di:", output_path)
```

Ganti **(file_path = ...)** dengan excel hasil dari codingan sebelumnya.

5. Menjalankan Script

Setelah semua bagian diisi, pastikan file Excel tidak sedang dibuka. Kemudian jalankan script di terminal menggunakan perintah berikut:

```
python nama_script.py
```

6. Contoh Hasil Output

Contoh Sebelum Proses:

```
Development Description | E1
-------------------------|----
Peserta mengikuti pelatihan dasar kepemimpinan untuk supervisor |
Mengikuti kursus cement manufacturing tingkat lanjut |
Training penggunaan alat ukur vibrasi |
Workshop internal untuk tim produksi |
Mengikuti program Leader as Coach |
```

Contoh Setelah Proses AI:

Development Description	E1
Peserta mengikuti pelatihan dasar kepemimpinan untuk supervisor	Supervisory Development Program (SDP)
Mengikuti kursus cement manufacturing tingkat lanjut	Cement Manufacturing Course
Training penggunaan alat ukur vibrasi	training penggunaan alat ukur vibrasi
Workshop internal untuk tim produksi
Mengikuti program Leader as Coach	Leader as Coach

7. Tips Penting

- Gunakan koneksi internet stabil.
- Tutup file Excel sebelum menjalankan script.
- Hindari nama folder dengan spasi.
- Jika proses terlalu lama, jalankan sebagian kecil data dulu.

8. Kesimpulan

Dengan mengikuti panduan ini, pengguna dapat membuat sistem klasifikasi E1 otomatis berbasis AI yang mampu membaca file Excel, mengelompokkan deskripsi ke kategori Leadership, Technical, atau Training, dan menyimpan hasil analisis dalam file baru.