**Ringkasan Proyek**

Secara garis besar, kode yang kita bangun adalah sebuah sistem **AI untuk klasifikasi teks multi-label**. Tujuannya adalah untuk secara otomatis memprediksi **Sektor** dan **Sub-sektor** bisnis berdasarkan deskripsi **Aktivitas** yang dimasukkan oleh pengguna. Untuk meningkatkan akurasi, sistem ini dilengkapi dengan modul **koreksi ejaan (spell correction)** yang cerdas sebelum melakukan prediksi.

**1. Library yang Digunakan (Perpustakaan Kode)**

Setiap library memiliki peran spesifik untuk membuat sistem ini bekerja:

* **pandas**: Ini adalah "Excel-nya Python". Library ini digunakan untuk memuat, membaca, dan memanipulasi data dari file CSV Anda (membuat kolom baru, membersihkan data, dll).
* **numpy**: Bekerja sama dengan pandas, library ini menyediakan dukungan untuk operasi numerik yang efisien.
* **scikit-learn (sklearn)**: Ini adalah library utama untuk Machine Learning di Python. Dari sini kita menggunakan beberapa komponen penting:
  + TfidfVectorizer: Untuk mengubah teks deskripsi aktivitas menjadi angka (vektor).
  + MultiLabelBinarizer: Untuk mengubah label Sektor dan Sub-sektor (yang bisa lebih dari satu) menjadi format yang dimengerti model.
  + LogisticRegression: Ini adalah model AI utama yang kita gunakan untuk belajar dan melakukan klasifikasi.
  + OneVsRestClassifier: Sebuah "pembungkus" yang memungkinkan model LogisticRegression menangani kasus di mana satu aktivitas bisa memiliki banyak label (misalnya, masuk ke Sektor Teknologi dan juga Jasa Profesional).
* **pyspellchecker**: Library khusus untuk mendeteksi dan mengoreksi kesalahan ketik dalam bahasa Inggris.
* **re (Regular Expressions)**: Digunakan untuk memecah kalimat menjadi kata-kata (membuat kamus domain).
* **ipywidgets & IPython.display**: Digunakan untuk membuat antarmuka interaktif (kotak input teks dan area output) di lingkungan seperti Google Colab atau Jupyter Notebook.
* **google.colab.files & io**: Digunakan khusus di Google Colab untuk mengunggah dan membaca file dari komputer Anda.

**2. Dataset yang Digunakan**

Anda menggunakan satu dataset utama:

* **50dataset.xlsx - Sheet1.csv**: Ini adalah **sumber pengetahuan (source of truth)** bagi AI kita. Dataset ini berisi 50 contoh data historis di mana setiap baris menghubungkan deskripsi Activities dengan Sector dan Sub-sector yang relevan. AI akan **belajar pola** dari data ini. Semakin berkualitas dan beragam data di sini, semakin pintar AI-nya.

**3. Metode dan Alur Kerja AI**

Berikut adalah langkah-langkah yang dilakukan oleh kode Anda dari awal hingga akhir:

**Langkah A: Preprocessing Data (Pembersihan Data)**

Ini adalah langkah pondasi. Sebelum AI bisa belajar, datanya harus bersih dan terstruktur.

1. **Memuat Data**: pandas memuat file CSV Anda ke dalam sebuah tabel (DataFrame).
2. **Membersihkan Nama Kolom**: Secara otomatis mengubah nama kolom menjadi huruf kecil dan mengganti spasi/hubung dengan garis bawah (contoh: 'Sub-sector\_1' menjadi 'sub\_sector\_1'). Ini mencegah error karena salah ketik nama kolom.
3. **Menggabungkan Label**: Karena satu aktivitas bisa punya banyak sektor/sub-sektor, kita menggabungkan kolom sector\_1, sector\_2, sector\_3 menjadi satu kolom baru bernama all\_sectors. Hal yang sama dilakukan untuk sub-sektor. Ini penting untuk klasifikasi multi-label.

**Langkah B: Koreksi Ejaan (Pendekatan Hibrida)**

Ini adalah salah satu bagian paling cerdas dari sistem Anda.

1. **Membuat Kamus Domain**: Pertama, kode membaca *semua* kata dari kolom activities di dataset Anda dan membuatnya menjadi kamus unik. Kamus ini berisi istilah-istilah spesifik bisnis Anda (seperti fintech, agritech, saas, dll).
2. **Melatih pyspellchecker**: Kamus domain ini kemudian "diajarkan" kepada pyspellchecker. Tujuannya agar pyspellchecker **tidak menganggap istilah domain sebagai salah ketik**.
3. **Proses Koreksi**: Saat Anda memasukkan teks, pyspellchecker akan memeriksa setiap kata. Jika ada kata yang salah ketik (misalnya, consultting), ia akan memperbaikinya menjadi consulting. Namun, jika ia menemukan kata seperti fintech, ia akan membiarkannya karena sudah diajarkan bahwa kata itu valid.

**Langkah C: Feature Engineering (Mengubah Teks menjadi Angka)**

Model AI tidak bisa membaca teks, jadi kita perlu mengubahnya menjadi angka. Metode yang digunakan adalah **TF-IDF (Term Frequency-Inverse Document Frequency)**.

* **Cara Kerja Sederhana**: TF-IDF memberikan "skor" pada setiap kata dalam sebuah kalimat. Kata-kata yang sering muncul dalam satu deskripsi aktivitas tetapi jarang muncul di deskripsi lain (kata-kata unik dan penting) akan mendapatkan skor tinggi. Contoh: kata "software" akan lebih penting daripada kata "dan".
* **Hasil**: Setiap deskripsi aktivitas diubah menjadi serangkaian angka (vektor) yang merepresentasikan pentingnya setiap kata di dalamnya.

**Langkah D: Pelatihan Model (Machine Learning)**

Di sinilah proses "belajar" terjadi.

1. **Model Inti: Logistic Regression**: Ini adalah model klasifikasi yang efisien dan populer. Model ini belajar memetakan pola dari vektor TF-IDF (dari Langkah C) ke label Sektor dan Sub-sektor yang benar.
2. **Pembungkus Multi-Label: OneVsRestClassifier**: Bayangkan Anda punya 5 sektor. Pembungkus ini akan melatih 5 model Logistic Regression secara terpisah: satu model untuk memprediksi "Apakah ini Sektor Teknologi atau bukan?", satu model untuk "Apakah ini Sektor Perdagangan atau bukan?", dan seterusnya. Ini memungkinkan output memiliki lebih dari satu label positif.
3. **Hasil Pelatihan**: Kita mendapatkan dua "otak" AI yang sudah terlatih: model\_sector dan model\_subsector.

**Langkah E: Prediksi (Menggunakan Model)**

Ini adalah tahap akhir saat Anda menggunakan sistemnya.

1. **Input Pengguna**: Anda mengetik deskripsi aktivitas (misalnya, devloping saas for fintech).
2. **Koreksi**: Teks Anda dilewatkan ke fungsi koreksi ejaan hibrida, mengubahnya menjadi developing saas for fintech.
3. **Transformasi TF-IDF**: Teks yang sudah bersih ini diubah menjadi vektor angka menggunakan tfidf yang sudah dilatih.
4. **Prediksi Probabilitas**: Vektor ini dimasukkan ke model\_sector dan model\_subsector. Model tidak langsung memberikan jawaban ya/tidak, tetapi **probabilitas** untuk setiap kelas. Contoh: {Teknologi: 95%, Jasa: 30%, Perdagangan: 5%}.
5. **Output**: Kode mengambil beberapa kelas dengan probabilitas tertinggi (di atas ambang batas tertentu) dan menampilkannya sebagai hasil prediksi Anda.

Semoga penjelasan ini memberikan gambaran yang jelas tentang bagaimana setiap bagian dari kode Anda bekerja sama untuk menciptakan sistem AI yang andal!