**Technical Information**

This code performs resume matching with job descriptions using PDF CVs. It involves several key components: PDF data extraction, job description data understanding, candidate-job matching, and results presentation. Here's a technical breakdown of each section:

**Section 1: PDF Data Extraction**

Function: extract\_cv\_details(pdf\_path)

Input: pdf\_path (Path to a PDF CV)

Output: cv\_details (Dictionary containing extracted details)

**Workflow:**

The code utilizes the PyPDF2 library to open and read the PDF file.

It iterates through the PDF pages and extracts text from each page.

Using regular expressions, it searches for specific details:

Category (Job role)

Skills

Extracted details are stored in the cv\_details dictionary.

The dictionary is returned as the output.

Function: process\_cvs\_in\_directory(directory\_path)

Input: directory\_path (Path to a directory containing PDF CVs)

Output: cv\_details\_list (List of dictionaries containing CV details)

Workflow:

The code iterates through files in the specified directory.

It filters files with the ".pdf" extension.

For each PDF file, it calls the extract\_cv\_details function to extract CV details.

Extracted details are appended to the cv\_details\_list.

The list of dictionaries is returned as the output.

**Section 2: Job Description Data Understanding**

Data Retrieval: Hugging Face Datasets

The code utilizes the Hugging Face datasets library to load job descriptions from the "jacob-hugging-face/job-descriptions" dataset.

It fetches job descriptions from the "train" split, selecting the first 15 descriptions for processing and matching.

**Section 3: Candidate-Job Matching**

Tokenization and Embedding

The code uses the DistilBERT tokenizer and model from the Transformers library (Hugging Face).

Function text\_to\_embedding(text) tokenizes and converts text into embeddings using DistilBERT.

Function calculate\_similarity(embedding1, embedding2) calculates cosine similarity between two embeddings using scikit-learn.

Matching Process

For each job description, it calculates an embedding for the description.

It iterates through the extracted CV details from Section 1.

For each CV, it combines the Category, Skills, and Education into a single text.

Calculates the similarity between the job description embedding and CV embedding.

Matches are stored as tuples containing CV details and similarity scores.

The matches are sorted by similarity in descending order, and the top 5 matches are selected.

**Section 4: Results Presentation**

Use of the tabulate Library

The code enhances the presentation of top matches using the tabulate library.

For each job description, it prints a header and creates a table to display similarity scores and CV details for the top 5 matches.

The results are printed in a well-structured and visually appealing tabular format, improving readability.

**Conclusion**

This code project successfully automates the process of matching candidates' CVs with job descriptions. It leverages PDF data extraction, NLP techniques, and machine learning to achieve this. The use of the tabulate library enhances the presentation of results, making it user-friendly and informative.

The code can be further extended and optimized for larger datasets and more complex matching requirements. Additionally, regular updates to job descriptions and CVs would ensure that the matching process remains relevant in a dynamic job market.