The skeleton of the project.

1. Tools used to achieve Resume matching.

- Pandas
- Numpy
- PyPDF2
- Pdfplumber
- CountVectorizer
- CosineSimilarity

2. Some changes

```
DeprecationError: PdfFileReader is deprecated and was removed in PyPDF2 3.0.0. Use PdfReader instead.
```

Why use these tools?

Why PyPDF2?(General info)

PyPDF2 is a Python library for working with PDF files. It provides many functionalities, such as reading, merging, splitting, cropping, and encrypting PDF files. It is based on the original PyPDF library, but with additional features and bug fixes.

Here are some of the key features of PyPDF2:

- **Reading and writing PDF files: PyPDF2 allows you to read and write PDF files in Python. You can extract text, metadata, and images from PDF files, as well as add new pages, and merge, split, or rotate PDF files.**

- Encryption and decryption: PyPDF2 supports encrypting and decrypting PDF files with passwords.
- Watermarking: You can use PyPDF2 to add watermarks or stamps to PDF files.
- Table of contents: PyPDF2 can generate a table of contents for a PDF file based on the headings and subheadings.
- Form filling: PyPDF2 can fill out form fields in a PDF file.
- Compression: PyPDF2 provides support for compressing and decompressing PDF files.

PyPDF2 is a popular library among developers who work with PDF files in Python, and it has been used in many projects across different industries. It is available under the MIT license, which makes it accessible and open-source.

It is important to note that PyPDF2 is no longer actively maintained, and the latest version of Python (3.10) is not compatible with PyPDF2. As of PyPDF2 version 1.26.0, the library is considered feature-complete, and any further development will be done in the forked libraries such as PyPDF4.

Why pdfplumber?(General info)

Pdfplumber is a Python library that enables the extraction of text and other data from PDF documents. It is a lightweight, open-source, and easy-to-use library that offers a simple interface to extract data from PDF files. Pdfplumber was created to address the limitations of other PDF parsing libraries and provides additional features such as extraction of tables, images, and metadata.

Pdfplumber can extract text from a PDF file, including Unicode characters and special symbols. It can also extract metadata from the PDF file such as the title, author, creation date, and modification date. Additionally, pdf plumber can extract tables from PDF files and convert them into Pandas data frames.

Here are some key features of pdf plumber:

- Extraction of text from PDF files, including Unicode characters and special symbols.
- Extraction of metadata from PDF files such as the title, author, creation date, and modification date.
- Extraction of tables from PDF files and conversion to Pandas data frames.
- Extraction of images from PDF files.
- Integration with other Python libraries such as NumPy, Pandas, and Matplotlib.
- Easy installation through pip.

Pdfplumber is an actively maintained library and has gained popularity among developers who need to extract data from PDF files. It is available under the MIT license, which makes it free and open-source.

Why CountVectorizer?(General info)

CountVectorizer is a feature extraction technique used in Natural Language Processing (NLP) for converting text into numerical features. It is a part of the Scikit-learn library, which is a widely-used library in Python for machine-learning tasks.

CountVectorizer converts a collection of text documents to a matrix of token counts. It works by tokenizing the text into words or n-grams and then counting the frequency of each token. The resulting matrix can be used as input to a machine-learning algorithm.

Here are some key features of CountVectorizer:

- Tokenization: CountVectorizer splits the text into individual tokens (words or n-grams) using a tokenizer function. By default, it uses a regular expression tokenizer that splits on whitespace and punctuation.
- Vocabulary building: CountVectorizer builds a vocabulary of all the tokens in the text corpus, assigning a unique integer index to each token.
- Counting: CountVectorizer counts the frequency of each token in each document and stores it in the corresponding position in the resulting matrix.

- Stop words removal: CountVectorizer can optionally remove stop words (common words that are not informative) from the vocabulary.
- n-grams: CountVectorizer can extract n-grams (consecutive sequences of words) instead of single words, which can capture more complex relationships between words.

CountVectorizer is a powerful tool for preprocessing text data and preparing it for machine-learning models. It is often used in combination with other NLP techniques such as TF-IDF and word embeddings to further improve the quality of the extracted features.

Why use CosineSimilarity?(General info)

Cosine similarity is a metric used to measure the similarity between two non-zero vectors of an inner product space. In Natural Language Processing (NLP), cosine similarity is commonly used to measure the similarity between two text documents represented as vectors of word counts or TF-IDF values.

The cosine similarity between two vectors, u and v, is defined as the dot product of the vectors divided by the product of their magnitudes:

cosine_similarity(u, v) = dot(u, v) / (||u|| * ||v||)

where dot(u, v) is the dot product of vectors u and v, and ||u|| and ||v|| are the magnitudes of vectors u and v, respectively.

Cosine similarity ranges from -1 to 1, with a value of 1 indicating that the two vectors are identical, a value of 0 indicating that the two vectors are orthogonal (i.e., have no correlation), and a value of -1 indicating that the two vectors are opposite in direction.

In NLP, cosine similarity is commonly used in applications such as document clustering, information retrieval, and recommendation systems. For example, cosine similarity can be used to find documents that are similar to a query document, or to recommend products or articles based on their similarity to a user's preferences.