Resume Matching System Using Machine Learning and Cosine Similarity

1. Objective

The objective of this project is to create a resume matching system that can automatically assess resumes against a given job description using machine learning and text similarity techniques. The system is designed to:

Classify resumes based on their relevance to specific job categories.

Rank resumes according to their similarity to the job description.

Provide an automated solution to help recruiters in matching resumes to job roles efficiently.

2. Methodology

The project is built using the Flask web framework for the backend, with a combination of machine learning and Natural Language Processing (NLP) techniques to process and match resumes. The methodology can be broken down into the following key steps:

2.1 Data Collection

The dataset consists of resumes in the form of text files, each associated with a specific job category. This dataset is used to train the machine learning model and for vectorizing the resumes to compute similarities. The data is structured in a CSV file with two columns:

- Resume Text: Contains the textual content of the resumes.
- Category: The job category corresponding to each resume.

2.2 Text Preprocessing

- Text Extraction: Resumes can be uploaded in multiple formats such as PDF, DOCX, and TXT. The relevant text is extracted using different methods:
  - PDF: Text is extracted using the PyPDF2 library.

- DOCX: Text is extracted using the python-docx library.
- TXT: Simple file reading is used to extract the text.

- Vectorization: The extracted text is then vectorized using the TF-IDF (Term Frequency-Inverse Document Frequency) method, which converts the textual data into numerical vectors that represent the importance of words in relation to the entire dataset.

2.3 Model Training

The Multinomial Naive Bayes classifier (MultinomialNB) is used to classify resumes into job categories. The model is trained on the vectorized resumes and their corresponding categories.

The TF-IDF Vectorizer is also saved as a separate model object, which is later used to convert new resume data into vectors during prediction.

2.4 Cosine Similarity Calculation

To match resumes to job descriptions, the cosine similarity between the job description and the resumes is calculated. The job description is vectorized using the TF-IDF method, and the similarity between the job description's vector and the vectors of all uploaded resumes is computed. The formula for cosine similarity is:

cosine_similarity(A, B) = (A * B) / (||A|| ||B||)

Where:

A and B are the vectorized representations of the job description and resume.

The result is a similarity score between 0 and 1, where 1 indicates perfect similarity.

2.5 Ranking and Result Display

The resumes are ranked based on their cosine similarity scores to the job description. The system displays:

- A list of resumes ranked by similarity to the job description.
- The predicted job category for each resume.
- The calculated similarity score between the job description and each resume.

3. System Architecture

The system is structured with the following components:

- Frontend (User Interface): A simple Flask-based web application where users can upload resumes and input the job description. The user interacts with the system through the / route to upload files and /matcher to view the results.
- Backend (Processing and Model Execution):
  - File Upload Handling: The system accepts PDF, DOCX, and TXT file formats for resumes and ensures only valid file types are processed.
  - Model Loading: The pre-trained machine learning model and vectorizer are loaded using joblib, allowing the system to use the saved models during resume matching.
  - Cosine Similarity Calculation: The system computes the cosine similarity between the job description and each resume to determine which resume most closely matches the job description.

4. Results

Once the user uploads resumes and enters a job description, the system outputs:

- A list of resumes ranked by their similarity to the job description.
- The predicted job category for each resume.
- The calculated cosine similarity between each resume and the job description.

This allows recruiters to quickly identify the most relevant resumes based on the job description,

streamlining the recruitment process.

5. Model Evaluation

While the primary metric for evaluation is cosine similarity, the performance of the model is also assessed based on how accurately it classifies resumes into the correct job categories. The Multinomial Naive Bayes classifier is an effective choice for text classification tasks as it handles large text datasets efficiently.

6. Challenges

- Data Quality: The accuracy of the model depends heavily on the quality of the dataset. Incorrect job categories or poorly formatted resumes can affect the predictions.
- Text Extraction: Extracting clean text from complex file formats (like PDFs) can be challenging. Some resumes may contain text embedded in images or scanned files, making extraction more difficult.
- Limited Training Data: A small dataset may lead to overfitting, where the model performs well on training data but fails to generalize to new, unseen resumes.

7. Future Work and Improvements

- Larger Dataset: Collecting a more diverse and extensive dataset would improve the model's ability to accurately classify resumes and match them to job descriptions.
- Advanced NLP Models: Implementing advanced NLP models such as BERT or GPT could enhance the system's understanding of context and improve matching accuracy.
- User Feedback: Incorporating a feedback mechanism where users can indicate the accuracy of the matches would help improve the system over time.

8. Conclusion

Resume Matching System Using Machine Learning and Cosine Similarity

This project presents an effective resume matching system using TF-IDF vectorization, Multinomial Naive Bayes classification, and cosine similarity. The system provides an efficient, automated method for matching resumes to job descriptions, streamlining the recruitment process. The approach used serves as a solid foundation for further enhancements in automated hiring tools.