AI-Powered Resume Screening System

Machine Learning & NLP Pipeline

1. Overview

The pipeline uses a hybrid approach: Unsupervised Cosine Similarity for immediate ranking and a Supervised Logistic Regression model for classification.

2. Text Extraction Layer

• **PDF**: pdfminer.six (extracts text from stream)

• **DOCX**: python-docx (iterates over paragraphs)

• **TXT**: Native Python read (UTF-8)

3. Preprocessing Pipeline

```
Input: "Senior Python Developer - 5 Years Exp!"
   |
   v
Lowercase: "senior python developer - 5 years exp!"
   |
   v
Remove Special Chars & Stopwords: "senior python developer years exp"
   |
   v
Lemmatize: ["senior", "python", "developer", "year", "exp"]
```

4. Feature Engineering (TF-IDF)

Term Frequency - Inverse Document Frequency converts text to vectors.

Why TF-IDF? It penalizes common words (like 'experience') that appear in every resume, focusing on unique skills.

5. Supervised Classification (Logistic Regression)

Training Data Structure:

Resume Segment	Job Req	Label
"Flask, Django..."	"Need Python Web..."	1
"Sales, Marketing"	"Need Python Web..."	0

Model Evaluation Metrics:

• **Precision**: Avoid false positives (irrelevant candidates).

• **Recall**: Avoid false negatives (missed talent).

• **F1-Score**: Balance between the two.

6. Scoring & Ranking Logic

Final Score = Cosine Similarity (Vector_Job, Vector_Resume)

Range: 0.0 to 1.0

```
from sklearn.metrics.pairwise import cosine_similarity

# vectors = numeric representation of text
similarity = cosine_similarity(job_vec, resume_vec)
# Result: [[0.85]] -> 85% Match
```