RESUME FILTERING SYSTEM USING TF-IDF AND SENTENCE TRANSFORMERS

📌 Disclaimer

The Job Description (JD) used in this project was generated using AI to simulate a realistic hiring scenario. The CV/resume samples were publicly sourced from an open resume repository online [1], intended solely for demonstration and educational purposes. No real or private data has been used in this project.

📌 Background

In Sri Lanka, it's common for recruiters to receive 100+ resumes per job post. Since many companies still manually screen CVs, this process is time-consuming and inefficient. Often, applicants submit irrelevant resumes, adding to recruiters' burden.

The goal of this project is to automate the elimination of completely irrelevant resumes, so that recruiters can focus on high-potential candidates.

🔧 Methodology

To evaluate how well a resume fits a Job Description (JD), I used **two techniques**:

1. TF-IDF (Term Frequency – Inverse Document Frequency)

- **What it is**: A traditional NLP method that converts text into numeric vectors by analyzing word frequency.
- **Domain**: Information Retrieval / Natural Language Processing
- **How it works**:
    - Common words (like "the", "and") are given low weight.
    - Unique and meaningful words in context are given higher importance.
- **Similarity**: Cosine similarity is used to measure how similar a resume vector is to the JD vector.
- **Limitation**: Requires **preprocessing**, is **sensitive to exact word matches**, and lacks semantic understanding (e.g., "analyst" ≠ "analyzing").

2. Sentence Transformers (Semantic Similarity)

- **What it is**: A transformer-based deep learning model (e.g., all-MiniLM-L6-v2) from the Sentence-Transformers library.
- **Domain**: Machine Learning / Deep NLP
- **Developed by**: UKP Lab @ TU Darmstadt, built on top of BERT and HuggingFace Transformers.
- **How it works**:
    - Transforms sentences into dense **semantic embeddings**.
    - Captures **meaning**, not just keywords.
- **Advantage**: Doesn't require heavy preprocessing; understands synonyms and context better.

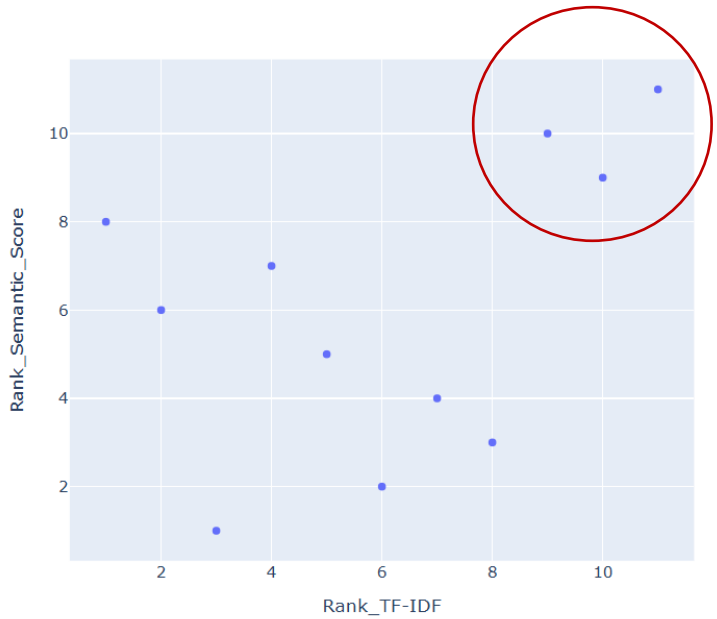
🧪 **Experiment Setup**

- Created a synthetic **Data Analyst Job Description**.
- Tested against **11 resumes**:
    - 8 were Data Analyst resumes (intern, junior, senior, domain-specialized).
    - 3 were completely unrelated resumes.

📊 **Results**

Resume	TF-IDF Score	Semantic Score	Rank_TF-IDF	Rank_Semantic_Score
data-analyst-intern-resume-example.pdf	0.139	0.631	8	3
data-analyst-resume-example.pdf	0.153	0.622	7	4
entry-level-risk-adjustment-data-analyst.pdf	0.223	0.609	2	6

experienced-data-analyst-resume.pdf	0.176	0.619	5	5
junior-data-analyst-resume.pdf	0.195	0.684	3	1
Non_DA_Sample_1.pdf	0.016	0.281	11	11
Non_DA_Sample_2.pdf	0.068	0.349	9	10
Non_DA_Sample_3.pdf	0.017	0.475	10	9
revenue-reporting-data-analyst.pdf	0.157	0.660	6	2
senior-data-analyst-resume.pdf	0.187	0.608	4	7
senior-insurance-data-analyst.pdf	0.288	0.598	1	8



✅ **Observations**

- **Both methods ranked unrelated resumes at the bottom**, proving the system can filter them out efficiently.
- **TF-IDF** focused on **keyword match** and gave low scores to resumes with different terminology (even if relevant).

- **Semantic Similarity** handled variations better and identified contextually relevant resumes more accurately.
- **Key Insight**: Even if TF-IDF fails, semantic scoring catches the intent, ensuring a **balanced filtering mechanism**.

🔍 Use Case Impact

If a recruiter receives 200 resumes, and 50 are off-target submissions:

- This system can **instantly flag and eliminate** those 50, saving hours of manual effort.
- Remaining resumes can be ranked and prioritized for detailed review.

🧠 Which Model Performs Best?

While **TF-IDF** is useful for simple keyword-based matching, **Sentence Transformers** clearly outperform it by capturing **semantic meaning**—understanding the context and intent behind words. This makes them more reliable for real-world resume screening where terminology varies. For even more advanced matching, techniques like **fine-tuned BERT models**, **cross-encoders**, or **retrieval-augmented generation (RAG)** can be explored. These allow for deeper context understanding, relevance scoring, and even Q&A-style filtering—pushing resume ranking into the next level of intelligence.