In-Depth Explanation of the TF-IDF Algorithm

TF-IDF (Term Frequency - Inverse Document Frequency) is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents, often used in information retrieval, natural language processing (NLP), and text mining. In your job portal project, this can help rank resumes by how relevant they are to a job description.

Let’s break down every term involved in the TF-IDF calculation:

1. Term Frequency (TF)

Term Frequency (TF) measures the frequency of a term (word) in a single document (or resume). The assumption is that the more often a term appears in a document, the more important it is in that document. However, common words (e.g., "the," "and," "is") may appear frequently but aren't relevant in most cases.

Mathematical Formula for TF:

TF(t,d)=ftN\text{TF}(t, d) = \frac{f\_t}{N}TF(t,d)=Nft​​

Where:

* ftf\_tft​ = Frequency of the term ttt (how many times the term appears in the document ddd)
* NNN = Total number of terms (words) in the document ddd

For example, consider Resume 1: "JavaScript React Node.js MongoDB Express.js REST API"

If we're calculating the term frequency of the term "JavaScript":

* fJavaScript=1f\_{\text{JavaScript}} = 1fJavaScript​=1 (because "JavaScript" appears once in Resume 1)
* N=6N = 6N=6 (total number of words in Resume 1)

So, the TF for "JavaScript" in Resume 1 is:

TF(JavaScript,Resume 1)=16=0.1667\text{TF}(\text{JavaScript}, \text{Resume 1}) = \frac{1}{6} = 0.1667TF(JavaScript,Resume 1)=61​=0.1667

2. Inverse Document Frequency (IDF)

Inverse Document Frequency (IDF) measures the importance of a term across all documents in the corpus. The intuition behind IDF is that terms that appear in many documents are less useful for identifying relevant documents, and thus their importance is reduced. Conversely, terms that appear in fewer documents are considered more informative.

Mathematical Formula for IDF:

IDF(t)=log⁡(D1+dt)\text{IDF}(t) = \log \left( \frac{D}{1 + d\_t} \right)IDF(t)=log(1+dt​D​)

Where:

* DDD = Total number of documents in the corpus (e.g., the total number of resumes)
* dtd\_tdt​ = Number of documents containing the term ttt (i.e., how many resumes contain the term "JavaScript")
* 1 + d\_t: The "+1" ensures we don't divide by zero if no document contains the term.

The logarithmic function helps to scale down the impact of the term frequency across the documents, making sure that terms in every document don’t disproportionately affect the IDF.

For example, consider:

* D=3D = 3D=3 (Total documents: Resume 1, Resume 2, and Resume 3)
* "JavaScript" appears in 2 out of 3 documents (Resume 1 and Resume 3), so dJavaScript=2d\_{\text{JavaScript}} = 2dJavaScript​=2.

The IDF for "JavaScript" is:

IDF(JavaScript)=log⁡(31+2)=log⁡(1)=0\text{IDF}(\text{JavaScript}) = \log \left( \frac{3}{1 + 2} \right) = \log(1) = 0IDF(JavaScript)=log(1+23​)=log(1)=0

This means "JavaScript" is a very common term in the documents (appears in 2 out of 3), so its IDF score is low, indicating that it’s less useful for distinguishing between resumes.

3. TF-IDF Calculation

The TF-IDF score is the product of Term Frequency (TF) and Inverse Document Frequency (IDF). It provides a measure of how relevant a term is within a specific document while accounting for its importance across the entire collection of documents.

Mathematical Formula for TF-IDF:

TF-IDF(t,d)=TF(t,d)×IDF(t)\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)TF-IDF(t,d)=TF(t,d)×IDF(t)

Where:

* TF(t,d)\text{TF}(t, d)TF(t,d) is the frequency of term ttt in document ddd
* IDF(t)\text{IDF}(t)IDF(t) is the inverse document frequency of term ttt across the entire corpus of documents

For example:

* TF for "JavaScript" in Resume 1 = 0.1667
* IDF for "JavaScript" = 0 (since it appears in 2 out of 3 resumes)

Thus:

TF-IDF(JavaScript,Resume 1)=0.1667×0=0\text{TF-IDF}(\text{JavaScript}, \text{Resume 1}) = 0.1667 \times 0 = 0TF-IDF(JavaScript,Resume 1)=0.1667×0=0

4. Final Ranking

Once the TF-IDF score for each term is calculated, the scores are summed up for each document (resume). The higher the TF-IDF score, the more relevant that document is for the given term.

After calculating the TF-IDF score for all terms in each resume, you can sort the resumes in descending order of their total TF-IDF score to find which resume is most relevant to the job description.

For example: If you have several relevant terms in a resume (such as "JavaScript", "React", and "Node.js"), the sum of the TF-IDF scores will be high, making the resume more relevant.

Visualizing the Calculation:

Here’s a summary of the formula with an example table (if you need to show it visually in your presentation):

| Term | Resume 1 TF | Resume 2 TF | Resume 3 TF | Resume 1 IDF | Resume 2 IDF | Resume 3 IDF | TF-IDF Resume 1 | TF-IDF Resume 2 | TF-IDF Resume 3 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| JavaScript | 0.1667 | 0 | 0.1667 | 0 | 0 | 0 | 0 | 0 | 0 |
| React | 0.1667 | 0 | 0.1667 | 0 | 0 | 0 | 0 | 0 | 0 |
| Node.js | 0.1667 | 0 | 0.1667 | 0 | 0 | 0 | 0 | 0 | 0 |

Note: A real-world example would have more diverse terms, and the IDF would not be 0 unless the word appears in all documents.

Key Takeaways:

* TF measures how important a term is within a single document.
* IDF measures how important a term is across all documents.
* TF-IDF combines these two to highlight words that are both important in a specific document and rare across the entire corpus, making it useful for tasks like resume ranking.