

# Cloud Based Data Engineering

TF-IDF

# Solve Problems With PySpark

**TF-IDF**

# Term Frequency – Inverse Document Frequency

**TF-IDF (term frequency-inverse document frequency)** is a statistical measure that evaluates how relevant a word is to a document in a collection of documents.

Done by multiplying two metrics:

1. how many times a word appears in a document
2. The inverse document frequency of the word across a set of documents.

# Applications of TF-IDF

## Information retrieval

TF-IDF was invented for document search and can be used to deliver results that are most relevant to what you're searching for.

## Keyword Extraction

- Useful for extracting keywords from text
- The highest scoring words of a document are the most relevant to that document

# Terminology

1. t – term / word
2. d – document (set of words)
3. N – count of corpus
4. Corpus – the total document set

**TF-IDF = Term Frequency (TF) \* Inverse Document Frequency (IDF)**

# Terminology

**TF-IDF = Term Frequency (TF) \* Inverse Document Frequency (IDF)**

$$TF(\textcolor{red}{t}, \textcolor{blue}{d}) = \frac{\text{count of } \textcolor{red}{t} \text{ in } \textcolor{blue}{d}}{\text{number of words in } \textcolor{blue}{d}}$$

$DF(\textcolor{red}{t}) = \text{occurrence of } \textcolor{red}{t} \text{ in } N \text{ documents}$

$$IDF(\textcolor{red}{t}) = \log\left(\frac{N}{DF(\textcolor{red}{t})}\right)$$

$$TF - IDF(\textcolor{red}{t}, \textcolor{blue}{d}) = TF(\textcolor{red}{t}, \textcolor{blue}{d}) \cdot \log\left(\frac{N}{DF(\textcolor{red}{t})}\right)$$

# Example

**Document 1** It is going to rain today.

**Document 2** Today I am not going outside.

**Document 3** I am going to watch the season premiere.

## Step 1: Counts

| Word  | Count |
|-------|-------|
| going | 3     |
| to    | 2     |
| today | 2     |
| i     | 2     |
| am    | 2     |
| it    | 1     |
| is    | 1     |
| rain  | 1     |

# Example

## Step 2: Find TF

$$TF(t, d) = \frac{\text{count of } t \text{ in } d}{\text{number of words in } d}$$

| Word  | Document 1           |
|-------|----------------------|
| going | $\frac{1}{6} = 0.16$ |
| to    | 0.16                 |
| today | 0.16                 |
| i     | 0                    |
| am    | 0                    |
| it    | 0.16                 |
| is    | 0.16                 |
| rain  | 0.16                 |

# Example

## Step 2: Find TF

$$TF(t, d) = \frac{\text{count of } t \text{ in } d}{\text{number of words in } d}$$

| Word  | Document 1           | Document 2 |
|-------|----------------------|------------|
| going | $\frac{1}{6} = 0.16$ | 0.16       |
| to    | 0.16                 | 0          |
| today | 0.16                 | 0.16       |
| i     | 0                    | 0.16       |
| am    | 0                    | 0.16       |
| it    | 0.16                 | 0          |
| is    | 0.16                 | 0          |
| rain  | 0.16                 | 0          |

# Example

## Step 2: Find TF

$$TF(t, d) = \frac{\text{count of } t \text{ in } d}{\text{number of words in } d}$$

| Word  | Document 1           | Document 2 | Document 3 |
|-------|----------------------|------------|------------|
| going | $\frac{1}{6} = 0.16$ | 0.16       | 0.12       |
| to    | 0.16                 | 0          | 0.12       |
| today | 0.16                 | 0.16       | 0          |
| i     | 0                    | 0.16       | 0.12       |
| am    | 0                    | 0.16       | 0.12       |
| it    | 0.16                 | 0          | 0          |
| is    | 0.16                 | 0          | 0          |
| rain  | 0.16                 | 0          | 0          |

# Example

$DF(t) = \text{occurrence of } t \text{ in } N \text{ documents}$

## Step 3: Find IDF

$$IDF(t) = \log\left(\frac{N}{DF(t)}\right)$$

| Word  | Count | IDF                                |
|-------|-------|------------------------------------|
| going | 3     | $\log\left(\frac{3}{3}\right) = 0$ |
| to    | 2     | $\log(1.5) = 0.58$                 |
| today | 2     | $\log(1.5) = 0.58$                 |
| i     | 2     | $\log(1.5) = 0.58$                 |
| am    | 2     | $\log(1.5) = 0.58$                 |
| it    | 1     | $\log(3) = 1.58$                   |
| is    | 1     | $\log(3) = 1.58$                   |
| rain  | 1     | $\log(3) = 1.58$                   |

# Example

## Step 4: Find TF-IDF

| Word  | IDF  | Document 1 | Document 2 | Document 3 |
|-------|------|------------|------------|------------|
| going | 0    | 0.16       | 0.16       | 0.12       |
| to    | 0.58 | 0.16       | 0          | 0.12       |
| today | 0.58 | 0.16       | 0.16       | 0          |
| i     | 0.58 | 0          | 0.16       | 0.12       |
| am    | 0.58 | 0          | 0.16       | 0.12       |
| it    | 1.58 | 0.16       | 0          | 0          |
| is    | 1.58 | 0.16       | 0          | 0          |
| rain  | 1.58 | 0.16       | 0          | 0          |

# Example

## Step 4: Find TF-IDF

|       | going | to                            | today | i    | am   | it   | is   | rain |
|-------|-------|-------------------------------|-------|------|------|------|------|------|
| Doc 1 | 0     | $0.58 \cdot 0.16$<br>$= 0.09$ | 0.09  | 0    | 0    | 0.25 | 0.25 | 0.25 |
| Doc 2 | 0     | 0                             | 0.09  | 0.09 | 0.09 | 0    | 0    | 0    |
| Doc 3 | 0     | $0.58 \cdot 0.12$<br>$= 0.06$ | 0     | 0.06 | 0.06 | 0    | 0    | 0    |

- “it”, “is”, “rain” important for doc1 but not for docs 2 and 3.
- Docs 1 and 2 talks about today

# TF-IDF Distributed Approach

## Task 1: Word Frequency in Doc

Mapper

Input: (docname, contents)

Output: ((word, docname), 1)

Reducer

Sums counts for word in document

Outputs ((word, docname),  $n$ )

# TF-IDF Distributed Approach

## Task 2: Word Counts For Docs

Mapper

Input: ((word, docname),  $n$ )

Output: (docname, (word,  $n$ ))

Reducer

Sums frequency of individual  $n$ 's in same doc

Feeds original data through

Outputs ((word, docname), ( $n$ ,  $N$ ))

# TF-IDF Distributed Approach

## Task 3: Word Frequency In Corpus

Mapper

Input: ((word, docname), (n, N))

Output: (word, (docname, n, N, 1))

Reducer

Sums counts for word in corpus

Outputs ((word, docname), (n, N, m))

# TF-IDF Distributed Approach

## Task 4: Calculate TF-IDF

Mapper

Input: ((word, docname), (n, N, m))

Output ((word, docname), TF\*IDF)

Reducer

Just the identity function