물론이죠! TF-IDF(Term Frequency-Inverse Document Frequency)는 문서 내 단어의 중요도를 정량화하는 대표적인 방법입니다. 아래에 개념, 수식, 그리고 간단한 예제를 같이 설명할게요.

✅ 1. 개념 정리 TF (Term Frequency) 단어가 문서에서 얼마나 자주 나오는지 나타냅니다.

$$TF(t,d)$$

단어 t 의 빈도 문서 d 의 총 단어 수 TF(t,d)= 문서 d 의 총 단어 수 단어 t 의 빈도

IDF (Inverse Document Frequency) 단어가 전체 문서 집합에서 얼마나 드문지를 나타냅니다.

$$IDF(t)$$

log $\left(\frac{N}{1+df(t)}\right)$ IDF(t)=log(1+df(t) N) N N: 전체 문서 수

$df(t)$ df(t): 단어 t가 나타난 문서 수

TF-IDF

$$\text{TF-IDF}(t,d)$$

$TF(t,d) \times IDF(t)$ TF-IDF(t,d)=TF(t,d)×IDF(t) ✅ 2. 간단한 예 🔶 문서 3개: vbnet Copy Edit D1: "I like machine learning" D2: "I like deep learning" D3: "I like pizza" 🔶 단어 후보: ["I", "like", "machine", "deep", "learning", "pizza"] 🔶 Step 1: TF 계산 (문서 D1 기준) "I": 1 / 4 = 0.25

"like": 1 / 4 = 0.25

"machine": 1 / 4 = 0.25

"learning": 1 / 4 = 0.25

🔶 Step 2: IDF 계산 (전체 문서 기준) "I": 나타난 문서 수 = 3 → log $\left(\frac{3}{1+3}\right)$

log（0.75）

log(1+3 3)=log(0.75)= 음수 (보통 stopword라 제거함)

"machine": 1개 문서 → log（3 1 + 1）

log（1.5）≈ 0.405 log(1+1 3)=log(1.5)≈0.405

"learning": 2개 문서 → log（3 / 3）

log（1）

0 log(3/3)=log(1)=0

"pizza": 1개 문서 → log（3 / 2）≈ 0.405 log(3/2)≈0.405

🔶 Step 3: TF-IDF 계산 (D1, 단어: "machine") TF = 0.25, IDF ≈ 0.405

TF-IDF = 0.25 × 0.405 ≈ 0.101

✅ 결론 문서에서 자주 등장하고, 전체 문서에서 드물게 나오는 단어일수록 TF-IDF가 높습니다.

TF는 문서 내 중요도, IDF는 전체 문서 내 희소성을 반영합니다.

In [1]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

docs = ["I like machine learning", "I like deep learning", "I like pizza"]
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(docs)

print(vectorizer.get_feature_names_out())
print(tfidf_matrix.toarray())
```

```
['deep' 'learning' 'like' 'machine' 'pizza']
[[0.         0.54783215 0.42544054 0.72033345 0.        ]
 [0.72033345 0.54783215 0.42544054 0.         0.        ]
 [0.         0.         0.50854232 0.         0.861037  ]]
```