Term Frequency - Inverse Document Frequency

in NLP TF-IDF is used to find a set of documents that are similiar to a query document.

What is term frequency?

1. No of times a term appear in a particular document
2. It is specific to a document
3. Few ways to computer term frequency are:

    - tf(d) = # of term appear in a document
    - tf(d) = # of term appear in a document / # of terms in a document
    - tf(d) = # of term appear in a document / frequency of most frequently found word in a document

    NB: sklearn uses the first one, ie tf(d) = # of term appear in a document

What is IDF

1. it measures how common or rare is a term accross entire corpus
2. if a word, g: I, like appear in multiple documents, then the IDF value will be close to zero or else it will approach one.

```
idf(t) = ln[n/df(t)]

idf(t) = ln([1+n/1+df(t)]+1) => sklearn: smooth_idf=True

idf(t) = ln[(n/df(t))+1] => sklearn: smooth_idf=False

no of documents is n

if smooth_idf is True (default) a constant 1 is added to the neumerator and denominator as if an extra document was seen containing every term exactly once. this prevents division by zero
```

TF-IDF(t) = tf(t) * idf(t)

TF-IDF(t) = tf(t) * ln[n/df(t)]+1

log is a monotonically increasing

In [4]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [5]:

```
d1='petrol cars are cheaper than the diesel cars'

d2='diesel is cheaper than petrol'
```

In [6]:

```
doc_corpus =[d1 , d2]
doc_corpus
```

Out[6]:

```
['petrol cars are cheaper than the diesel cars',
 'diesel is cheaper than petrol']
```

In [7]:

```
#clean the document; remove stop words

vec = TfidfVectorizer(stop_words='english')
matrix = vec.fit_transform(doc_corpus) #pass always 1D list of strings
```

In [8]:

```
print(vec.get_feature_names_out()) #columns of tfidf matrix
```

```
['cars' 'cheaper' 'diesel' 'petrol']
```

In [9]:

```
print(vec.vocabulary_)
```

```
{'petrol': 3, 'cars': 0, 'cheaper': 1, 'diesel': 2}
```

In [10]:

```
print(matrix.shape)
```

```
(2, 4)
```

In [11]:

```
print(matrix.toarray())
```

```
[[0.85135433 0.30287281 0.30287281 0.30287281]
 [0.         0.57735027 0.57735027 0.57735027]]
```

In [ ]:

```

```