In [1]:
```
# !pip install sklearn
# !pip install pandas
# !pip install nltk
# nltk.download()
```

Applied ML Series by Sanjana Sahayaraj

In [1]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import pandas as pd
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
```

In [2]:
```
from string import punctuation
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
stop_words = set(stopwords.words('english'))
```

In [3]:
```
corpus = [
    "I would like to learn machine learning",
    "Natural Language Processing is a sub field in machine learning",
    "NLP stands for Natural Language Processing",
    "Text embedding is an important step in NLP",
    "Text embedding produces numerical representation of texts",
    "TFIDF can produce dense numerical vector form of text"
]
```

In [4]:
```
boilerplate = []
```

In [5]:
```
cleanCorpus = []
for sentence in corpus:
    text_tokens = word_tokenize(sentence)
    cleaned_tokens = [w.lower() for w in text_tokens if (not w in stop_words) and (not w in punctuation) and (n
    cleanCorpus.append(' '.join(cleaned_tokens))
```

In [6]:
```
cleanCorpus
```

Out[6]:
```
['i would like learn machine learning',
 'natural language processing sub field machine learning',
 'nlp stands natural language processing',
 'text embedding important step nlp',
 'text embedding produces numerical representation texts',
 'tfidf produce dense numerical vector form text']
```

In [7]:
```
vectorizer = TfidfVectorizer()
vectors = vectorizer.fit_transform(cleanCorpus)
```

In [8]:
```
feature_names = vectorizer.get_feature_names_out()
dense = vectors.todense()
denselist = dense.tolist()
df = pd.DataFrame(denselist, columns=feature_names)
df
```

Out[8]:

	dense	embedding	field	form	important	language	learn	learning	like	machine	natural	nlp	nu
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.479748	0.393400	0.479748	0.393400	0.000000	0.000000	0.
1	0.000000	0.000000	0.431849	0.000000	0.000000	0.354122	0.000000	0.354122	0.000000	0.354122	0.354122	0.000000	0.
2	0.000000	0.000000	0.000000	0.000000	0.000000	0.426900	0.000000	0.000000	0.000000	0.000000	0.426900	0.426900	0.
3	0.000000	0.419328	0.000000	0.000000	0.511367	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.419328	0.
4	0.000000	0.373346	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
5	0.403183	0.000000	0.000000	0.403183	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.

In [9]:
```
df_vectors = df.to_numpy()
```

In [10]:
```
cosine_similarity([df_vectors[0]],[df_vectors[1]]).item(0)
```

Out[10]:
```
0.2786233168412409
```

In [11]:
```
cosine_similarity([df_vectors[1]],[df_vectors[2]]).item(0)
```

Out[11]:
```
0.4535245396642898
```

In [12]:
```
cosine_similarity([df_vectors[1]],[df_vectors[1]]).item(0)
```

Out[12]:
```
1.0
```