In [ ]:
```
#SWETHA JENIFER S_18-01-23
```

# Lab3. Computing Document Similarity using VSM

### EXCERCISE-1:Print TFIDF values

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

In [2]:
```
import pandas as pd
```

In [3]:
```
docs=["good movie","not a god movie","did not like","i like it","good one"]
```

In [4]:
```
tfidf=TfidfVectorizer(min_df=2,max_df=0.5,ngram_range=(1,2))
features=tfidf.fit_transform(docs)
print(features)
```

```
  (0, 2)        0.7071067811865476
  (0, 0)        0.7071067811865476
  (1, 3)        0.7071067811865476
  (1, 2)        0.7071067811865476
  (2, 1)        0.7071067811865476
  (2, 3)        0.7071067811865476
  (3, 1)        1.0
  (4, 0)        1.0
```

In [5]:
```
df=pd.DataFrame(
    features.todense(),
    columns=tfidf.get_feature_names())
print(df)
```

```
       good      like     movie       not
0  0.707107  0.000000  0.707107  0.000000
1  0.000000  0.000000  0.707107  0.707107
2  0.000000  0.707107  0.000000  0.707107
3  0.000000  1.000000  0.000000  0.000000
4  1.000000  0.000000  0.000000  0.000000

C:\Users\sweth\anaconda3\lib\site-packages\sklearn\utils\deprecation.py:87: Fut
ureWarning: Function get_feature_names is deprecated; get_feature_names is depr
ecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out inst
ead.
  warnings.warn(msg, category=FutureWarning)
```

## EXCERCISE-2:

### 1 Change the values of min_df and ngram_range and observe various outputs

**Change the values of min_df and ngram_range and observe various outputs**

In [6]:
```
tfidf=TfidfVectorizer(min_df=1,max_df=.5,ngram_range=(2,4))
features=tfidf.fit_transform(docs)
print(features)
```

```
  (0, 3)        1.0
  (1, 7)        0.5773502691896258
  (1, 2)        0.5773502691896258
  (1, 6)        0.5773502691896258
  (2, 1)        0.5773502691896258
  (2, 8)        0.5773502691896258
  (2, 0)        0.5773502691896258
  (3, 5)        1.0
  (4, 4)        1.0
```

In [7]:
```
#pretty printing
df=pd.DataFrame(features.todense(),columns=tfidf.get_feature_names())
print(df)
```

```
   did not  did not like  god movie  good movie  good one  like it  not god  \
0  0.00000       0.00000    0.00000         1.0       0.0      0.0  0.00000
1  0.00000       0.00000    0.57735         0.0       0.0      0.0  0.57735
2  0.57735       0.57735    0.00000         0.0       0.0      0.0  0.00000
3  0.00000       0.00000    0.00000         0.0       0.0      1.0  0.00000
4  0.00000       0.00000    0.00000         0.0       1.0      0.0  0.00000

   not god movie  not like
0        0.00000   0.00000
1        0.57735   0.00000
2        0.00000   0.57735
3        0.00000   0.00000
4        0.00000   0.00000
```

```
C:\Users\sweth\anaconda3\lib\site-packages\sklearn\utils\deprecation.py:87: Fut
ureWarning: Function get_feature_names is deprecated; get_feature_names is depr
ecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out inst
ead.
  warnings.warn(msg, category=FutureWarning)
```

## EXCERCISE-3:Compute Cosine Similarity between 2 Documents

In [8]:
```
from sklearn.metrics.pairwise import linear_kernel
doc1=features[0:1]
doc2=features[1:2]
score=linear_kernel(doc1,doc2)
print(score)
```

```
[[0.]]
```

In [9]:
```
scores=linear_kernel(doc1,features)
print(scores)
```

```
[[1. 0. 0. 0. 0.]]
```

In [10]:
```
query="I like this good movie"
qfeature=tfidf.transform([query])
scores2=linear_kernel(doc1,features)
print(scores2)
```

```
[[1. 0. 0. 0. 0.]]
```

## EXCERCISE-4:Find Top-N similar documents

***Question-1.Consider the following documents and compute TFIDF values***

In [11]:
```
docs=["the house had a tiny little mouse",
      "the cat saw the mouse",
      "the mouse ran away from the house",
      "the cat finally ate the mouse",
      "the end of the mouse story"]
```

***Question-2.Compute Cosine similarity between 3rd document ("the mouse ran away from the house") with all other documents.Which is the most similar document?***

In [12]:
```
tfidf = TfidfVectorizer(min_df=2, max_df=0.5, ngram_range=(1, 2))
features = tfidf.fit_transform(docs)
print(features)
```

```
  (0, 3)        0.7071067811865476
  (0, 1)        0.7071067811865476
  (1, 2)        0.7071067811865476
  (1, 0)        0.7071067811865476
  (2, 3)        0.7071067811865476
  (2, 1)        0.7071067811865476
  (3, 2)        0.7071067811865476
  (3, 0)        0.7071067811865476
```

In [13]:
```
doc1=features[0:3]
sr=linear_kernel(doc1, features)
print(sr)
```

```
[[1. 0. 1. 0. 0.]
 [0. 1. 0. 1. 0.]
 [1. 0. 1. 0. 0.]]
```

***Question-3.Find Top-2similar documents for the 3rd document based on Cosine similarity values.***

In [14]:
```
scores2 = linear_kernel(doc1, features)
print(scores2)
```

```
[[1. 0. 1. 0. 0.]
 [0. 1. 0. 1. 0.]
 [1. 0. 1. 0. 0.]]
```