Name:P.Asha Belcilda

Roll no:225229104

Lab-3 Computing Document Similarity using VSM

Exercise-1:Print TFIDF values

In [2]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [3]:
```
import pandas as pd
```

In [4]:
```
docs=["good movie","not a good movie","did not like","i like it","good one"]
```

In [6]:
```
#using default tokenizer in TfidfVectorizer
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, 2)	0.5773502691896257
  (1, 0)	0.5773502691896257
  (1, 3)	0.5773502691896257
  (2, 3)	0.7071067811865476
  (2, 1)	0.7071067811865476
  (3, 1)	1.0
```

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

```
   good movie      like     movie       not
0    0.707107  0.000000  0.707107  0.000000
1    0.577350  0.000000  0.577350  0.577350
2    0.000000  0.707107  0.000000  0.707107
3    0.000000  1.000000  0.000000  0.000000
4    0.000000  0.000000  0.000000  0.000000
```

EXERCISE-2:

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

In [24]:
```
tfidf=TfidfVectorizer(min_df=1,max_df=0.5,ngram_range=(2,2))
features=tfidf.fit_transform(docs)
print(features)
```

```
  (0, 17)	0.3741047724501572
  (0, 8)	0.4636932227319092
  (0, 7)	0.4636932227319092
  (0, 18)	0.4636932227319092
  (0, 9)	0.4636932227319092
  (1, 15)	0.49552379079705033
  (1, 3)	0.6141889663426562
  (1, 14)	0.6141889663426562
  (2, 17)	0.3741047724501572
  (2, 10)	0.4636932227319092
  (2, 13)	0.4636932227319092
  (2, 1)	0.4636932227319092
  (2, 6)	0.4636932227319092
  (3, 15)	0.4222421409859579
  (3, 2)	0.5233582502695435
  (3, 5)	0.5233582502695435
  (3, 0)	0.5233582502695435
  (4, 16)	0.5
  (4, 4)	0.5
  (4, 12)	0.5
  (4, 11)	0.5
```

In [22]:
```
df=pd.DataFrame(features.todense(),columns=tfidf.get_feature_names())
print(df)
```

```
    ate the  ate the mouse  away from  away from the  away from the house  \
0  0.000000       0.000000   0.000000       0.000000             0.000000   
1  0.000000       0.000000   0.000000       0.000000             0.000000   
2  0.000000       0.000000   0.270657       0.270657             0.270657   
3  0.306413       0.306413   0.000000       0.000000             0.000000   
4  0.000000       0.000000   0.000000       0.000000             0.000000   

   cat finally  cat finally ate  cat finally ate the   cat saw  cat saw the  \
0     0.000000         0.000000             0.000000  0.000000     0.000000   
1     0.000000         0.000000             0.000000  0.361529     0.361529   
2     0.000000         0.000000             0.000000  0.000000     0.000000   
3     0.306413         0.306413             0.306413  0.000000     0.000000   
4     0.000000         0.000000             0.000000  0.000000     0.000000   

         ...         the end of  the end of the  the house  the house had  \
0        ...           0.000000        0.000000   0.236365       0.292968   
1        ...           0.000000        0.000000   0.000000       0.000000   
2        ...           0.000000        0.000000   0.218364       0.000000   
3        ...           0.000000        0.000000   0.000000       0.000000   
4        ...           0.301511        0.301511   0.000000       0.000000   

   the house had tiny  the mouse ran  the mouse ran away  the mouse story  \
0            0.292968       0.000000            0.000000         0.000000   
1            0.000000       0.000000            0.000000         0.000000   
2            0.000000       0.270657            0.270657         0.000000   
3            0.000000       0.000000            0.000000         0.000000   
4            0.000000       0.000000            0.000000         0.301511   

   tiny little  tiny little mouse  
0     0.292968           0.292968  
1     0.000000           0.000000  
2     0.000000           0.000000  
3     0.000000           0.000000  
4     0.000000           0.000000  

[5 rows x 54 columns]
```

EXERCISE-3:

In [8]:
```
from sklearn.metrics.pairwise import linear_kernel
```

In [10]:
```
#cosine score between 1st and 2nd doc
doc1=features[0:1]
doc2=features[1:2]
score=linear_kernel(doc1,doc2)
print(score)
```

```
[[0.81649658]]
```

In [13]:
```
#cosine score between 1st and all other docs
scores=linear_kernel(doc1,features)
print(scores)
```

```
[[1.         0.81649658 0.         0.         0.        ]]
```

In [15]:
```
#cosine similarity for a new doc
query="I like this good movie"
qfeature=tfidf.transform([query])
scores2=linear_kernel(doc1,features)
print(scores2)
```

```
[[1.         0.81649658 0.         0.         0.        ]]
```

EXERCISE-4:Find Top-N similar documents

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

In [17]:
```
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 with all other documents.Which is the most similar document?.

In [18]:
```
tfidf = TfidfVectorizer(min_df=2, max_df=0.5, ngram_range=(1, 2))
features = tfidf.fit_transform(docs)
print(features)
```

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

In [19]:
```
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-2 similar documents for the 3rd document based on Consine similarity values.

In [20]:
```
scores2 = linear_kernel(doc1, features)
print(scores2)
```

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