JOSHUA E

225229117

Lab3. Computing Document Similarity using VSM

EXCERCISE-1:Print TFIDF values

In [13]:

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

In [14]:

```
import pandas as pd
```

In [15]:

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

In [32]:

```
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 [33]:

```
df=pd.DataFrame(
    features.todense(),
    columns=tfidf.get_feature_names())
print(df)
```

```
        cat     house   the cat  the house
0  0.000000  0.707107  0.000000   0.707107
1  0.707107  0.000000  0.707107   0.000000
2  0.000000  0.707107  0.000000   0.707107
3  0.707107  0.000000  0.707107   0.000000
4  0.000000  0.000000  0.000000   0.000000
```

EXCERCISE-2:

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

In [27]:

```
tfidf=TfidfVectorizer(min_df=1,max_df=.5,ngram_range=(2,4))
features=tfidf.fit_transform(docs)
print(features)
```

```
  (0, 46)	0.23636461617263152
  (0, 22)	0.29296784934087045
  (0, 19)	0.29296784934087045
  (0, 52)	0.29296784934087045
  (0, 25)	0.29296784934087045
  (0, 47)	0.29296784934087045
  (0, 23)	0.29296784934087045
  (0, 20)	0.29296784934087045
  (0, 53)	0.29296784934087045
  (0, 48)	0.29296784934087045
  (0, 24)	0.29296784934087045
  (0, 21)	0.29296784934087045
  (1, 38)	0.2916794154657719
  (1, 8)	0.36152911730069653
  (1, 36)	0.36152911730069653
  (1, 41)	0.36152911730069653
  (1, 9)	0.36152911730069653
  (1, 37)	0.36152911730069653
  (1, 42)	0.36152911730069653
  (1, 10)	0.36152911730069653
  (2, 46)	0.21836428188496418
  (2, 26)	0.27065689895808104
  (2, 33)	0.27065689895808104
  (2, 2)	0.27065689895808104
  (2, 17)	0.27065689895808104
  :	:
  (2, 28)	0.27065689895808104
  (2, 35)	0.27065689895808104
  (2, 4)	0.27065689895808104
  (3, 38)	0.24721169864215167
  (3, 5)	0.3064125284733739
  (3, 14)	0.3064125284733739
  (3, 0)	0.3064125284733739
  (3, 39)	0.3064125284733739
  (3, 6)	0.3064125284733739
  (3, 15)	0.3064125284733739
  (3, 1)	0.3064125284733739
  (3, 40)	0.3064125284733739
  (3, 7)	0.3064125284733739
  (3, 16)	0.3064125284733739
  (4, 43)	0.3015113445777636
  (4, 11)	0.3015113445777636
  (4, 30)	0.3015113445777636
  (4, 29)	0.3015113445777636
  (4, 44)	0.3015113445777636
  (4, 12)	0.3015113445777636
  (4, 31)	0.3015113445777636
  (4, 51)	0.3015113445777636
  (4, 45)	0.3015113445777636
  (4, 13)	0.3015113445777636
  (4, 32)	0.3015113445777636
```

In [34]:

```
#pretty printing
df=pd.DataFrame(features.todense(),columns=tfidf.get_feature_names())
print(df)
```

```
        cat     house   the cat  the house
0  0.000000  0.707107  0.000000   0.707107
1  0.707107  0.000000  0.707107   0.000000
2  0.000000  0.707107  0.000000   0.707107
3  0.707107  0.000000  0.707107   0.000000
4  0.000000  0.000000  0.000000   0.000000
```

EXCERCISE-3:Compute Cosine Similarity between 2 Documents

In [21]:

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

```
[[0.5]]
```

In [22]:

```
scores=linear_kernel(doc1,features)
print(scores)
```

```
[[1.         0.5        0.         0.         0.70710678]]
```

In [24]:

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

```
[[1.         0.5        0.         0.         0.70710678]]
```

EXCERCISE-4:Find Top-N similar documents

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

In [25]:

```
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 [35]:

```
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 [36]:

```
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 [37]:

```
scores2 = linear_kernel(doc1, features)
print(scores2)
```

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