d1 = new york times

d2 = new york post

d3 = los angeles times

q = new new times

Compute the similarity using :-

a) Similarity Coefficient

b) Cosine Similarity

In [37]:
```
q = "new new times"
d1 = "new york times"
d2 = "new york post"
d3 = "los angeles times"
```

Using Similarity Coeffient(Jaccard Similarity)

In [54]:
```
def jaccard(Query, Statement):
    words_doc1 = set(Query.lower().split())
    words_doc2 = set(Statement.lower().split())
    intersection = words_doc1.intersection(words_doc2)
    union = words_doc1.union(words_doc2)
    return float(len(intersection)) / len(union)
```

In [55]:
```
jaccard(q, d1)
```

Out[55]:
```
0.6666666666666666
```

In [56]:
```
jaccard(q, d2)
```

Out[56]:
```
0.25
```

In [57]:
```
jaccard(q, d3)
```

Out[57]:
```
0.25
```

Using Cosine Similarity

In [33]:
```
import nltk
nltk.download('punkt')
from nltk.corpus import stopwords
```

```
from nltk.tokenize import word_tokenize
def cos_sim(Query, Statement):
    X_list = word_tokenize(Query)
    Y_list = word_tokenize(Statement)
    sw = stopwords.words('english')
    l1 =[];l2 =[]
    X_set = {w for w in X_list if not w in sw}
    Y_set = {w for w in Y_list if not w in sw}
    rvector = X_set.union(Y_set)
    for w in rvector:
        if w in X_set: l1.append(1)
        else: l1.append(0)
        if w in Y_set: l2.append(1)
        else: l2.append(0)
    c = 0
    for i in range(len(rvector)):
                c+= l1[i]*l2[i]
    cosine = c / float((sum(l1)*sum(l2))**0.5)
    return cosine
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\91874\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

In [34]:
```
cos_sim(q, d1)
```

Out[34]:
```
0.8164965809277261
```

In [35]:
```
cos_sim(q, d2)
```

Out[35]:
```
0.4082482904638631
```

In [36]:
```
cos_sim(q, d3)
```

Out[36]:
```
0.4082482904638631
```

In [ ]:
```

```