IT3041 - Information Retrieval and Web Analytics

Lab Sheet 04

1). Let assume the following corpus

- D1 : I am Sam.
- D2 : Sam I am.
- D3 : I do not like green eggs and ham.
- D4 : I do not like them, Sam I am.

a) Write the code to create k-grams for the above corpus when k=1,2,3 (Just a guidance is given in the answer)
b) Write code to find out the Jaccard coefficient between the given documents based on different k (Just a guidance is given in the answer)

2) Apply levenshtein algorithm to “python” and “pythonly” and get the minimum edit distance

3) Implement the SOUNDEX algorithm

Q1 (a)

In [ ]:
```
from nltk.util import ngrams
```

In [ ]:
```
text="April is the best month"
text_new="$"+"April is the best month".replace(" ","$")+"$"
print(text_new)
```

In [ ]:
```
g2=ngrams(text_new,2)
g2list=["".join(i) for i in g2]
print(g2list)
```

Q1 (b)

In [ ]:
```
def jaccard(x,y):
    a=x.intersection(y)
    b=x.union(y)
    return len(a)/len(b)
```

Q2

In [ ]:
```
from nltk.metrics.distance import edit_distance
```

In [ ]:
```
print(edit_distance('python','pythonly'))
```

In [ ]:
```
# import pip
# pip.main(["install","Levenshtein"])
```

In [ ]:
```
from Levenshtein import distance
```

In [ ]:
```
print(distance('python','pythonly'))
```

Q3

In [1]:
```
def soundex(word: str) -> str:
    """
    Soundex implementation (slide version):
      1) Retain first letter (uppercase).
      2) Change A,E,I,O,U,H,W,Y -> '0'
      3) Map letters to digits:
           B,F,P,V -> 1
           C,G,J,K,Q,S,X,Z -> 2
           D,T -> 3
           L -> 4
           M,N -> 5
           R -> 6
      4) Remove pairs of consecutive duplicate digits.
      5) Remove all zeros.
      6) Pad with trailing zeros and return first four characters: LDDD
    """
    if not word:
        return "0000"

    w = word.strip()
```

```
    if not w:
        return "0000"

    first = w[0].upper()

    groups = {
        'B': '1', 'F': '1', 'P': '1', 'V': '1',
        'C': '2', 'G': '2', 'J': '2', 'K': '2',
        'Q': '2', 'S': '2', 'X': '2', 'Z': '2',
        'D': '3', 'T': '3',
        'L': '4',
        'M': '5', 'N': '5',
        'R': '6'
    }
    zeros = set("AEIOUHWY")

    encoded = []
    for ch in w[1:].upper():
        if ch in zeros:
            encoded.append('0')
        else:
            encoded.append(groups.get(ch, ''))

    dedup = []
    last = None
    for d in encoded:
        if d == '' :
            continue
        if d != last:
            dedup.append(d)
            last = d

    dedup_no_zeros = [d for d in dedup if d != '0']

    code = first + "".join(dedup_no_zeros)
    code = (code + "0000")[:4]
    return code
```

In [ ]:

```
print(soundex("Herman"))
print(soundex("Hermann"))
```