Assignment No. 1

Aim : To Learn to Calculate bigrams from the given Corpus and Calculate the probability Of Sentence.

Basic Knowledge:

- An N-gram language model predicts the probability of a given N-gram within any sequence of words in the language. If we have a good N-gram model, we can predict p(w | h) – what is the probability of seeing the word *w* given a history of previous words *h* – where the history contains n-1 words.
- The **bigram model** approximates the probability of word given all the previous words, by using only the conditional probability of the last preceding word:

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

- The assumption that the probability of a word depends only on the previous word is called a *Markov assumption.*
- We have to train the machine to understand the meaning of the Sentence using the probablitic Approach.
- It generally tries to understand the context of the given word according to the surrounding of the word(the word that comes after or before).
- We can calculate the Probablity Using Chain Rule Using Prabablity.

$$P(w1,w2.....wn) = \Pi(P(wi|w1,w2....wi-1)$$

- To calculate the probability of sentence from the above formula is not Practical so Markov Came with an Assumption.
- Markov Assumption:

$$P(w1,w2.....wn) = \Pi(P(wi|wi-k....wi)$$

- Probablity is Found Using :

$$P(A/B) = P(A,B)/P(B)$$

1. Calculating Bigrams :

- A 2-gram (or bigram) is a two-word sequence of words, like “I love”, “love reading”, or “Analytics Vidhya”. And a 3-gram (or trigram) is a three-word sequence of words like “I love reading”, “about data science” or “on Analytics Vidhya”.
- Examples of Bigrams : This Book, My Book , Our Book , Your Book ,His Book , Her Book etc.

Example:

- Sentence : We need to book our tickets soon.
  Bigrams From Above Sentence are :
  We need , need to , to book , book our , our tickets , tickets soon .

- Sentence : They are going to play together.

  Bigrams From Above Sentence are :

  They are , are going , going to , to play , play together.

2. Calculating The Probablity Of a Sentence:

- Estimating Bigram Probablity:

  $P(wi|wi-1) = Count(wi-1 , wi)/Count(wi-1)$

Example:

<s> I am Sam <\s>

<s> Sam I am <\s>

<s> I do not like to study<\s>

Bigrams : <s>I,I am, am Sam , Sam <\s> , <s> Sam , Sam I , am <\s> , I do , do not, not like , like to , to study , study <\s>.

- Probablities Of Bigrams :

  P(I|<s>) = c(<s>,I)/c(<s>) = 2/3 = 0.67

  P(Sam/<s>) = c(<s>,sam)/c(<s>) = 1/3 = 0.33

  P(am|I) = c(I,am)/c(I) = 2/3 = 0.67

  P(</s>|Sam) = ½ = 0.5

  P(Sam|am) = ½ =0.5

  P(do|I) = 1/3 = 0.33.

jupyter Assignment_1 Last Checkpoint: 4 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [1]:
```
def ReadData():
    in1 = input("Enter the Paragraph\n")
    ls = in1.split('.')
    ls1 = []
    ls2 = []
    for i in ls:
        if i == '':
            pass
        else:
            ls2.append(i)
    for i in range(len(ls2)):
        if i==' ':
            pass
        else:
            ls1.append(ls2[i].split())
    return ls1
```

In [2]:
```
def createBigram(data):
    listOfBigrams = []
    listOfUnigrams = []
    bigramCounts = {}
    unigramCounts = {}
    for i in range(len(data)):
        for j in range(len(data[i])-1):
            listOfBigrams.append((data[i][j],data[i][j+1]))
            if bigramCounts.get((data[i][j],data[i][j+1])) == None:
                bigramCounts[(data[i][j],data[i][j+1])] = 1
            else:
                bigramCounts[(data[i][j],data[i][j+1])] = bigramCounts[(data[i][j],data[i][j+1])] + 1
    for i in range(len(data)):
        for j in range(len(data[i])):
            listOfUnigrams.append(data[i][j])
            if unigramCounts.get(data[i][j]) == None:
                unigramCounts[data[i][j]] = 1
            else:
                unigramCounts[data[i][j]] = unigramCounts[data[i][j]] + 1
    return listOfUnigrams,unigramCounts,listOfBigrams,bigramCounts
```

In [3]:
```
def PrintUnigram(data):
    listOfUnigrams = []
    for i in range(len(data)):
        for j in range(len(data[i])):
            listOfUnigrams.append(data[i][j])
    if listOfUnigrams == []:
        print("No Input Given")
        return
    print(listOfUnigrams)
```

In [4]:
```
def PrintBigram(data):
    listOfBigrams = []
    for i in range(len(data)):
        for j in range(len(data[i])-1):
            listOfBigrams.append((data[i][j],data[i][j+1]))
    if listOfBigrams == []:
        print("No Input Given")
        return
    print(listOfBigrams)
```

In [5]:
```
def calcBigramProb(listOfBigrams,bigramCounts,unigramCounts):
    listOfProb = {}
    for bigram in listOfBigrams:
        word1 = bigram[0]
        word2 = bigram[1]
        listOfProb[bigram] = (bigramCounts.get(bigram)) / (unigramCounts.get(word1))
    return listOfProb
```

In [6]:
```
def Trained_model(listOfBigrams, bigramCounts, unigramCounts):
    BigramProbablity = calcBigramProb(listOfBigrams, bigramCounts, unigramCounts)
    inList = input("Write a Sentence to check the Probablity\n")
    inList = inList.split()
    outputProb1 = 1
    bilist = []
    bigrm = []
    for i in range(len(inList) - 1):
        if i < len(inList) - 1:
            bilist.append((inList[i], inList[i + 1]))
    print("The bigrams in given sentence are")
    print(bilist)
    for i in range(len(bilist)):
        if bilist[i] in BigramProbablity:
            outputProb1 *= BigramProbablity[bilist[i]]
        else:

            outputProb1 *= 0
    print("The Probablity Of the Sentence is ",outputProb1)
```

In [7]:
```
def PrintAll(ls):
    listOfUnigrams, unigramCounts, listOfBigrams, bigramCounts = createBigram(ls)
    BigramProbablity = calcBigramProb(listOfBigrams,bigramCounts,unigramCounts)
    if listOfUnigrams == []:
        print("No Input Given")
        return
    print("----List Of Unigrams----")
    print(listOfUnigrams)
    print()
    print("----List Of Bigrams----")
    print(listOfBigrams)
    print()
    print("----Frequency Of Unigrams----")
    print(unigramCounts)
```

```
    print(unigramCounts)
    print()
    print("----Frequency of Bigrams----")
    print(bigramCounts)
    print()
    print("----Probablity of Bigrams----")
    print(BigramProbablity)
val = 1
ex = 0
ls1 = [[]]
while(val==1 or ex <6):
    print('MENU : ')
    val = 0
    print(" 1: Writing the Paragraph \n 2: Print unigram \n 3: Print Bigram \n 4: Print EveryThing \n 5: Check The Trained Model
    ex = int(input("Enter Your Choice: "))
    if ex == 1:
        ls1 = ReadData()
        continue
    elif ex == 2:
        PrintUnigram(ls1)
        continue
    elif ex == 3:
        PrintBigram(ls1)
        continue
    elif ex == 4:
        PrintAll(ls1)
        continue
    elif ex == 5:
        listOfUnigrams, unigramCounts, listOfBigrams, bigramCounts = createBigram(ls1)
        Trained_model(listOfBigrams, bigramCounts, unigramCounts)
        continue
    elif ex>=6 or ex<=0:
        print("Program is Terminated")
        break
```

```
MENU :
 1: Writing the Paragraph
 2: Print unigram
 3: Print Bigram
 4: Print EveryThing
 5: Check The Trained Model
 6: Exit
Enter Your Choice: 1
Enter the Paragraph
Aakanksha Bhondve from Comp B
MENU :
 1: Writing the Paragraph
 2: Print unigram
 3: Print Bigram
 4: Print EveryThing
 5: Check The Trained Model
 6: Exit
Enter Your Choice: 2
['Aakanksha', 'Bhondve', 'from', 'Comp', 'B']
MENU :
 1: Writing the Paragraph
 2: Print unigram
 3: Print Bigram
 4: Print EveryThing
 5: Check The Trained Model
 6: Exit
Enter Your Choice: 3
[('Aakanksha', 'Bhondve'), ('Bhondve', 'from'), ('from', 'Comp'), ('Comp', 'B')]
MENU :
 1: Writing the Paragraph
 2: Print unigram
 3: Print Bigram
 4: Print EveryThing
 5: Check The Trained Model
 6: Exit
Enter Your Choice: 4
```

```
Enter Your Choice: 4
----List Of Unigrams----
['Aakanksha', 'Bhondve', 'from', 'Comp', 'B']

----List Of Bigrams----
[('Aakanksha', 'Bhondve'), ('Bhondve', 'from'), ('from', 'Comp'), ('Comp', 'B')]

----Frequency Of Unigrams----
{'Aakanksha': 1, 'Bhondve': 1, 'from': 1, 'Comp': 1, 'B': 1}

----Frequency of Bigrams----
{('Aakanksha', 'Bhondve'): 1, ('Bhondve', 'from'): 1, ('from', 'Comp'): 1, ('Comp', 'B'): 1}

----Probablity of Bigrams----
{('Aakanksha', 'Bhondve'): 1.0, ('Bhondve', 'from'): 1.0, ('from', 'Comp'): 1.0, ('Comp', 'B'): 1.0}
MENU :
 1: Writing the Paragraph
 2: Print unigram
 3: Print Bigram
 4: Print EveryThing
 5: Check The Trained Model
 6: Exit
Enter Your Choice: 5
Write a Sentence to check the Probablity
Comp B
The bigrams in given sentence are
[('Comp', 'B')]
The Probablity Of the Sentence is  1.0
MENU :
 1: Writing the Paragraph
 2: Print unigram
 3: Print Bigram
 4: Print EveryThing
 5: Check The Trained Model
 6: Exit
Enter Your Choice: 6
```

```
 6: Exit
Enter Your Choice: 6
Program is Terminated
```