Lab – 3

Subject : NIS

Aim: Write a Program to do Encryption and Decryption using Vigenere Cipher.

1. Do the Cryptanalysis of Vigenere Cipher (Use sufficiently large CipherText). Use Index of Coincidence to verify the guessed Key Length. Use Mutual Index of Coincidence to guess the Key.

Description :-

- The vigenere cipher is an algorithm that is used to encrypt and decrypt the text.

- The vigenere cipher is an algorithm of encrypting alphabetic text that uses a series of Caesar ciphers. It is based on keyword's letter.

- It is an example of polyalphabetic substitution cipher. This algorithm is easy to understand and implement.

- There are two methods for encryption and decryption.

=>Method 1 is vigenere table or tabula recta.

- When the vigenere table given the encryption and decryption are done using the vigenere table (26 * 26 matrix).

=> Method 2 is using vigenere algebraically formula.

- In this method first thing to do is convert all alphabet a-z into numbers 0-25.

- The encryption formula is :-

$E(i) = [P(i) + K(i)] \bmod 26$

- The decryption formula is :-

$D(i) = [E(i) – K(i)] \bmod 26$

- In the case of decryption whenever d(i) value becomes negative then we will add 26 into negative value.

-> Cryptanalysis of vigenere cipher :- - Cryptanalysis of the Vigenere cipher has 2 main steps: identify the period of the cipher (the length of the key), then find the specific key. To identify the period we use a test based on the index of coinsidence, to find the specific key we use the chi-squared test. - Chi-squared test is very useful in machine learning it is used for compare distribution of probabilities.

- Method to find length of key :-

For example,

Cipher text is :- hfgtaskjiopqmncjfghs

- Assume some length of key and divide this cipher text for example if we assume length of key is 3 then

Y31 = htkomjh

Y32 = fajpnfs

Y33 = gsiqcg

- Then we need to find index of coincidence and sum them for this three and then take average = total_ic / 3. And saved it in another list.

- We have to run this procedure upto some length of key and take the index of the max and second max value from result as I did in above code.

- This two index is become our guess of key length then try to find the actual key by using this length.

- Method to find key :-

- Assume our guess for length is 3.

- Take frequency of actual English letter in one list.

- And take frequency of cipher text letter in another list

- Perform dot product of them then do left shift in cipher text frequency and then do again dot product and save this product value in another list.

- Perform above operation upto 26 times and take out the max value from the result. This character will become out first char of key.

- Repeat above 2 points upto 3 times because our length of key is 3.

- Get the key.

Program: -

```
from KasaskiTest import findLength

alphabets = "abcdefghijklmnopqrstuvwxyz"
english_probability = [0.08167, 0.01492, 0.02782, 0.04253, 0.12702,
0.02228, 0.02015,
                       0.06094, 0.06966, 0.00153, 0.00772, 0.04025,
0.02406, 0.06749,
                       0.07507, 0.01929, 0.00095, 0.05987, 0.06327,
0.09056, 0.02758,
                       0.00978, 0.02360, 0.00150, 0.01974, 0.00074]
base=ord('a')
icThreshold = 0.01
def getProb(LangYList):
    alphabetsList = [c for c in alphabets]
    n = len(LangYList)
        # print(LangYList, n)
    ProbList = [LangYList.count(c)/n for c in alphabetsList]
    return ProbList

def getKeyFromCipher(LangX_Probability, LangY, maxChar=26):
        print("___________\nExtracting Key from cipher")
        LangY = [c for c in LangY]
        LangY_Probability = getProb(LangY)
        MI_buffer = []
        for _ in range(len(LangX_Probability)):
            MI =
sum([LangX_Probability[countJ]*LangY_Probability[countJ]
                     for countJ in range(len(LangX_Probability))])
            MI_buffer.append(MI)
            LangY_Probability.append(LangY_Probability.pop(0))
        max_MI = max(MI_buffer)
        key = MI_buffer.index(max_MI)
        print("MIC : ", max_MI, "\nKey found :", chr(base + key))
        return key
def encrypt(msg,k):
    l=len(msg)
    l1=len(k)
```

```
    msg=list(msg)
    k=[ord(ch)-97 for ch in k]
    for _ in range(l):
        msg[_]=chr(((ord(msg[_]) - 97 + k[_ % l1]) % 26 ) + 97)
    return "".join(msg)

def decrypt(msg,k):
    l=len(msg)
    l1=len(k)
    k=[ord(ch)-97 for ch in k]
    msg=list(msg)
    for _ in range(l):
        msg[_]=chr(((ord(msg[_]) - 97 - k[_ % l1]) % 26 ) + 97)
    return "".join(msg)

def getKeysFromCipher(cipherText, length):
        secretKeysRetrived = []
        Y_parts = []
        for i in range(length):
            part = cipherText[i::length]
            Y_parts.append(part)
            tempKey = getKeyFromCipher(english_probability, Y_parts[i],
26)
            secretKeysRetrived.append(chr(base + tempKey))
        secretKeysRetrived = "".join(secretKeysRetrived)
        return secretKeysRetrived
def cryptanalysis(cipherText):
    m = findLength(cipherText, 25)
    print("M := ", m)
    keysRetrived = getKeysFromCipher(cipherText, m)
    print("Keys : ", keysRetrived)
if __name__ == "__main__":
    msg,k=input().split()
    enc_msg=encrypt(msg,k)
    dec_msg=decrypt(enc_msg,k)
    cryptanalysis(enc_msg)
```

Output: -

```
PS D:\DDIT CE\Sem 6\NIS\Lab 3> python .\VigenereCipher.py
thehitchhikersguidetothegalaxyisthefirstofsixbooksinthehitchhikersguidetothegalaxycomedysciencefictiontrilogybydouglasadamsthenovelisanadaptationo
fthefirstfourpartsofadamssradioseriesofthesamenamethenovelwasfirstpublishedinlondononoctoberitsoldcopiesinthefirstthreemonths pascal
M :=  6
____________
Extracting Key from cipher
MIC :  0.06266695652173912
Key found : p
____________
Extracting Key from cipher
MIC :  0.06734777777777777
Key found : a
____________
Extracting Key from cipher
MIC :  0.06235244444444444
Key found : s
____________
Extracting Key from cipher
MIC :  0.06390155555555557
Key found : c
____________
Extracting Key from cipher
MIC :  0.06446155555555555
Key found : a
____________
Extracting Key from cipher
MIC :  0.06774644444444444
Key found : l
Keys :  pascal
PS D:\DDIT CE\Sem 6\NIS\Lab 3>
```