Expt 9

Implementation of uncertain method (Dempster Shafer theory)

Aim:- Implementation of uncertain method (Dempster Shafer theory)

Problem Formulation:-

To solve inference problem representing uncertain method to obtain a belief function.

Using the mass function which has built in combination rules obtain combination the dempster rule of combination

Initial State:-

n1 = {'a': 0.4, 'b': 0.2, 'ab': 0.1, 'bc': 0.3}

n2 = {'b': 0.6, 'c': 0.2, 'ac': 0.3, 'a': 0.2}

Final State:-

{'ac': 0.107894, 'c': 0.10526, 'b': 0.5623 ... }

Problem Solving:-

The combination is calculated from the two sets of masses m_1 and m_2 in the following manner:

$\therefore m_{1,2}(\phi) = 0$

$\therefore m_{1,2}(A) = (m_1 \oplus m_2)(A) = \frac{1}{1-K} \sum_{B \cap C = A} n_1(B), m($

AI LAB
EXPERIMENT NO: 9

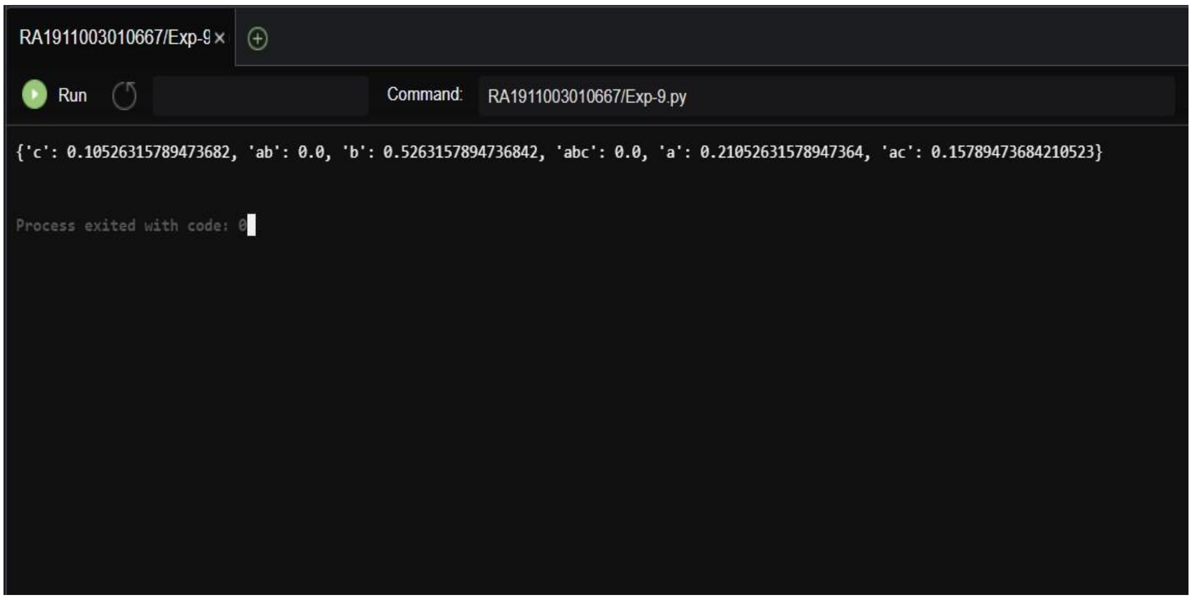
ALGORITHM:-

Step 1: Start
Step 2: Each piece of evidence is represented by a separate belief function
Step 3: Combination rules are then used to successively fuse all these belief
functions in order to obtain a belief function representing all available evidence.
Step 4: Specifically, the combination (called the joint mass) is calculated from
the two sets of masses m1 and m2 in the following manner:
• m1,2(∅) =0
• m1,2(A)=(m1⊕m2)(A)=(1/1−K) ∑B∩C=A≠∅ m1(B) m2(C)
where,
• K=∑B∩C=∅ m1(B) m2(C) K
K is a measure of the amount of conflict between the two mass sets.
Step 5: In python Mass-Function has the built-in combination rules.
Step 6: Stop

CODE:-

```
from numpy import *
def DempsterRule(m1, m2):
## extract the frame of discernment
sets=set(m1.keys()).union(set(m2.keys()))
result=dict.fromkeys(sets,0)
## Combination process
for i in m1.keys():
for j in m2.keys():
if set(str(i)).intersection(set(str(j))) == set(str(i)):
result[i]+=m1[i]*m2[j]
elif set(str(i)).intersection(set(str(j))) == set(str(j)):
result[j]+=m1[i]*m2[j]
## normalize the results
f= sum(list(result.values()))
for i in result.keys():
 result[i] /=f
 return result

m1 = {'a':0.4, 'b':0.2, 'ab':0.1, 'abc':0.3}
 m2 = {'b':0.5, 'c':0.2, 'ac':0.3, 'a':0.0}
 print(DempsterRule(m1, m2))
```

<u>OUTPUT:-</u>



RA1911003010667/Exp-9 ×

Run

Command: RA1911003010667/Exp-9.py

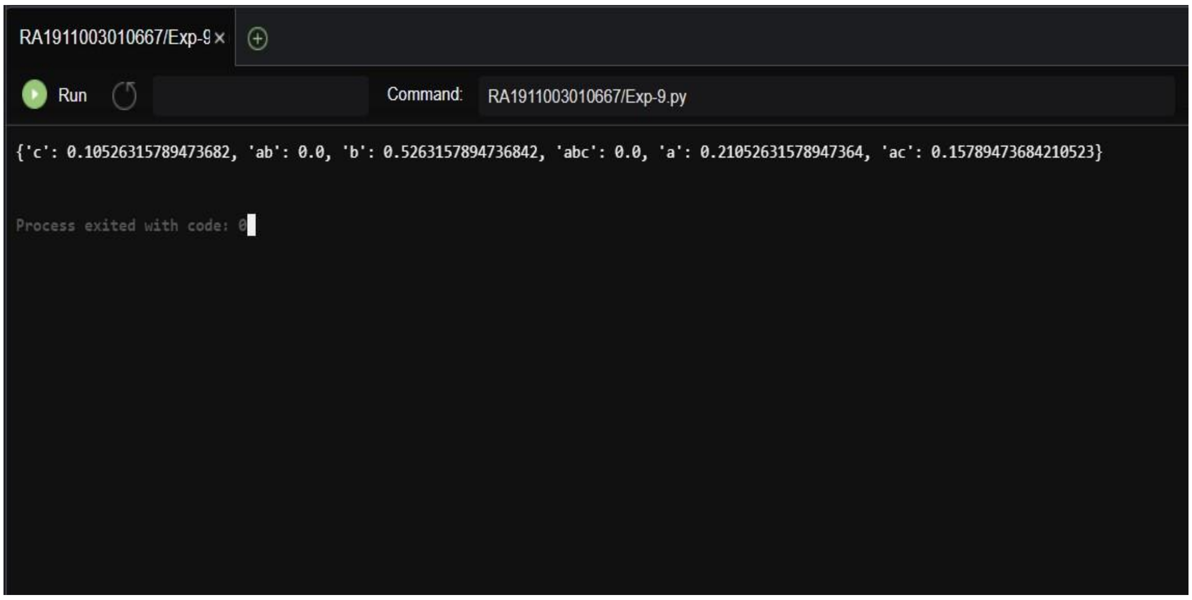
```
{'c': 0.10526315789473682, 'ab': 0.0, 'b': 0.5263157894736842, 'abc': 0.0, 'a': 0.21052631578947364, 'ac': 0.15789473684210523}


Process exited with code: 0
```

<u>RESULT:-</u>

Hence, the Implementation of Dempster Shafer Theory is done successfully.

```
m1 = {'a':0.4, 'b':0.2, 'ab':0.1, 'abc':0.3}
  m2 = {'b':0.5, 'c':0.2, 'ac':0.3, 'a':0.0}
  print(DempsterRule(m1, m2))
```

OUTPUT:-



RA1911003010667/Exp-9

Command: RA1911003010667/Exp-9.py

```
{'c': 0.10526315789473682, 'ab': 0.0, 'b': 0.5263157894736842, 'abc': 0.0, 'a': 0.21052631578947364, 'ac': 0.15789473684210523}


Process exited with code: 0
```

RESULT:-

Hence, the Implementation of Dempster Shafer Theory is done successfully.