LAB 5

OBJECTIVE

To implement Markov chain and Chi Square Test

THEORY

A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules. The defining characteristic of a Markov chain is that no matter how the process arrived at its present state, the possible future states are fixed. In other words, the probability of transitioning to any particular state is dependent solely on the current state and time elapsed. The state space, or set of all possible states, can be anything: letters, numbers, weather conditions, baseball scores, or stock performances.

The testing for uniformity can be achieved through different frequency tests. These tests use the Kolmogorov-Smirnov or the chi- square test to compare the degree of agreement between the distribution of the set of random numbers generated to a uniform distribution.

CODE

```
# Python program to implement Markov Chain (Rain vs No Rain)

n = int(input("Enter the day for which the weather need to be
predicted (1,2,3,..) : "))

#    Rain   No rain
P = [[0.8,  0.2], #Rain
     [0.4,  0.6]] #No rain

#    Rain  No rain
Pi = [ 1,     0] #Current weather

for i in range(n):
    Pi = [Pi[0]*P[0][0] + Pi[1]*P[1][0],
          Pi[0]*P[0][1] + Pi[1]*P[1][1]]

print(f"Probability of rain at day {n} : {round(Pi[0],5)}")
print(f"Probability of no rain at day {n} : {round(Pi[1],5)}")
```

OUTPUT

```
Enter the day for which weather need to be predicted (1,2,3,..) : 2
Probability of rain at day 2 : 0.72
Probability of no rain at day 2 : 0.28
```

CODE

```
# Python program to implement Markov Chain (Coke vs Pepsi)

n = int(input("Enter no of purchase after which the probability
needs to be calculated : "))

#    Coke   Pepsi
P = [[0.9,  0.1], #Coke
     [0.2,  0.8]] #Pepsi

#    Coke Pepsi
Pi = [ 0,    1] #Current Purchase

for i in range(n):
    Pi = [Pi[0]*P[0][0] + Pi[1]*P[1][0],
          Pi[0]*P[0][1] + Pi[1]*P[1][1]]

print(f"Probability of purchasing coke after {n} purchases :
{round(Pi[0],5)}")
print(f"Probability of purchasing pepsi after {n} purchases :
{round(Pi[1],5)}")
```

OUTPUT

```
Enter no of purchase after which the probability needs to be
calculated : 3
Probability of purchasing coke after 3 purchases : 0.438
Probability of purchasing pepsi after 3 purchases : 0.562
```

CODE

```
# Python program to implement Chi square test

data = [0.34, 0.83, 0.96, 0.47, 0.79, 0.99, 0.37, 0.72, 0.06, 0.18,
        0.90, 0.76, 0.99, 0.30, 0.71, 0.17, 0.51, 0.43, 0.39, 0.26,
        0.25, 0.79, 0.77, 0.17, 0.23, 0.99, 0.54, 0.56, 0.84, 0.97,
        0.89, 0.64, 0.67, 0.82, 0.19, 0.46, 0.01, 0.97, 0.24, 0.88,
        0.87, 0.70, 0.56, 0.56, 0.82, 0.05, 0.81, 0.30, 0.40, 0.64,
        0.44, 0.81, 0.41, 0.05, 0.93, 0.66, 0.28, 0.94, 0.64, 0.47,
        0.12, 0.94, 0.52, 0.45, 0.65, 0.10, 0.69, 0.96, 0.40, 0.60,
        0.21, 0.74, 0.73, 0.31, 0.37, 0.42, 0.34, 0.58, 0.19, 0.11,
        0.46, 0.22, 0.99, 0.78, 0.39, 0.18, 0.75, 0.73, 0.79, 0.29,
        0.67, 0.74, 0.02, 0.05, 0.42, 0.49, 0.49, 0.05, 0.62, 0.78]
N = len(data) # total number of observations
n = 10 # total number of intervals
degree_of_freedom = n - 1
chi_square_tabulated = 16.9


class_interval = [i/10 for i in range(0,n+1,1)]
# print(class_interval)
items_in_ci = list()
for i in range(n):
    items_in_ci.append([])


i=0
j=1
for item in data:
    while True:
        if(class_interval[i]<=item)and(item<class_interval[j]):
            items_in_ci[i].append(item)
            i=0
            j=1
            break
        else:
            i+=1
            j+=1
# print(items_in_ci)
```

```
Observed = list()
for i in range(n):
    Observed.append(len(items_in_ci[i]))
# print(Observed)

Expected = list()
for i in range (n):
    Expected.append(10)
#   print(Expected)

E_O_diff=list()
E_O_diff_squared=list()
E_O_E=list()

for i in range(n):
    E_O_diff.append(Expected[i]-Observed[i])
    E_O_diff_squared.append(E_O_diff[i]**2)
    E_O_E.append(E_O_diff_squared[i]/Expected[i])
# print(E_O_diff)
# print(E_O_diff_squared)
# print(E_O_E)


chi_square=0
for i in range (n):
    chi_square+=E_O_E[i]
# print(chi_square)


# Print the table
print("Class Interval(i)\tOi\tEi\tEi-Oi\t(Ei-Oi)^2\t(Ei-Oi)^2/Ei\n")
for i in range(n):
    print(f"[{class_interval[i]} - {class_interval[i+1]})\t\t
                {Observed[i]}\t{Expected[i]}\t{E_O_diff[i]}\t
                {E_O_diff_squared[i]}\t\t {E_O_E[i]}")
print(f"\nTotal\t\t\tN={N}\t\t\t\t\t X^2={chi_square}")
```

```
# Print conclusion
print(f"\nCalculated chi square = {chi_square}")
print(f"Tabulated chi square = {chi_square_tabulated}")

if chi_square < chi_square_tabulated:
    print("The calculated value of chi square is less than tabulated
value.")
    print("The null hypothesis of uniform distribution is
accepted.")
else:
    print("The calculated value of chi square is more than tabulated
value.")
    print("The null hypothesis of uniform distribution is
rejected.")
```

OUTPUT

Class Interval(i)	Oi	Ei	Ei-Oi	(Ei-Oi)^2	(Ei-Oi)^2/Ei
[0.0 - 0.1)	7	10	3	9	0.9
[0.1 - 0.2)	9	10	1	1	0.1
[0.2 - 0.3)	8	10	2	4	0.4
[0.3 - 0.4)	9	10	1	1	0.1
[0.4 - 0.5)	14	10	-4	16	1.6
[0.5 - 0.6)	7	10	3	9	0.9
[0.6 - 0.7)	10	10	0	0	0.0
[0.7 - 0.8)	15	10	-5	25	2.5
[0.8 - 0.9)	9	10	1	1	0.1
[0.9 - 1.0)	12	10	-2	4	0.4
Total	N=100				X^2=7.0

```
Calculated chi square = 7.0
Tabulated chi square = 16.9
The calculated value of chi square is less than tabulated value.
The null hypothesis of uniform distribution is accepted.
```

CONCLUSION

In this lab, I implemented Markov chain to predict the weather and probable purchase of customers as well as the Chi Square test to test the uniformity of random numbers.