FDS Practical: 03

Formula for Probability

The probability formula is defined as the possibility of an event to happen is equal to the ratio of the number of favourable outcomes and the total number of outcomes.

Probability of event to happen P(E) = Number of favourable outcomes / Total Numbers of outcomes

In [1]:
```
# probability of getting 3 when a die is rolied
ns = 6 #n(S) = {1,2,3,4,5,6}
na = 1 #n(A) ={3}
pa = na / ns # P(A)
print("probability of getting 3 is :", pa)
```

```
probability of getting 3 is : 0.16666666666666666
```

In [2]:
```
# probability of atleast getting one head when a coin is tossed thrice
ns = 8 #n(S) = {HHH,HHT,HTH,THH,TTH,THT,HTT,TTT}
na = 7 #n(A) = {HHH,HHT,HTH,THH,TTH,THT,HTT}
pa = na / ns
print("probability of getting atleat one head is : ",pa)
```

```
probability of getting atleat one head is :  0.875
```

In [3]:
```
# A glass jar contains 5 red, 3 blue and 2 green jelly beans. If a jelly bean is chosen
# probability that it is not blue ?

ns = 10 #n(S) = {5red , 3blue , 2green}
na = 7 #n(A) = {5red , 2green}
pa = na / ns # P(A)
print("probability of getting not blue jellybean is: " ,pa)
```

```
probability of getting not blue jellybean is:  0.7
```

Independent and Dependent Events

If the occurrence of any event is completely unaffected by the occurrence of any other event , such events are known as an independent event in probability and the events which are affected by other events are known as dependent events.

eg. 1

In [4]:
```
# if the probability that person A will be alive in 20 years is 0.7 and the probability
# is 0.5, what is the probability that they will both be alive in 20 years ?
# These are indepent events, so
p = 0.7 * 0.5
print("probability that they will be alive after 20 years is : " , p)
```

```
probability that they will be alive after 20 years is :  0.35
```

In [5]:
```
def event_probability(n,s):
```

```python
    return n/s
```

In [6]:

```python
# A fair die is tossed twice. Find the probability of getting a 4 or 5 on the first tos

pa = event_probability(2,6) # probability of getting a 4 or 5
pb = event_probability(3,6) # probability of getting a 1,2 or 3
P = pa*pb
print("probability of getting a 4 or 5 on the first toss and a 1 ,2 or 3 in the second
```

```
probability of getting a 4 or 5 on the first toss and a 1 ,2 or 3 in the second toss is
:  0.16666666666666666
```

In [7]:

```python
# A bag contains 5 white marbles, 3 black marbles and 2 green marbles. In each draw , a
# and not replaced. In three draws , find the probability of obtaining white, black and
pw = event_probability(5,10)
pb = event_probability(3,9)
pg = event_probability(2,8)

print("The probability of obtaining white , black and green in that order is ",(pw*pb*p
```

```
The probability of obtaining white , black and green in that order is  0.041666666666666
664
```

In [8]:

```python
# Calculate the probability of drawing a heart or a club

# Sample Space
cards = 52
hearts = 13
clubs = 13
heart_Or_club = event_probability(hearts , cards) + event_probability(clubs , cards)
print(heart_Or_club)
```

```
0.5
```

In [9]:

```python
# Calculate the probability of drawing on ace , king , or a queen

aces = 4
kings = 4
queens = 4
ace_king_Or_queen = event_probability(aces,cards) + event_probability(kings,cards) + ev

print(ace_king_Or_queen)
```

```
0.23076923076923078
```

In [10]:

```python
# Calculate the probability of drawing a heart or an ace

hearts = 13
aces = 4
ace_of_hearts = 1
heart_Or_ace = event_probability(hearts, cards) + event_probability(aces,cards) - event_
print(round(heart_Or_ace , 1))
```

```
0.3
```

Complementary Events

For any event E1 there another event E1' which represent the remaining elements of the sample space S.

E1 =S - E1'

If a dice is rolled then the sample space S is given as S = {1,2,3,4,5,6}. If event E1 represent all the outcomes which is greater than 4, then E1 ={5,6} and E1' ={1,2,3,4}

Thus E1' is the complement of the event E1

Smiliarly, the complement of E1,E2,E3....En will be represented as E1',E2',E3'......En'

eg. 1

In [11]:
```
# probability of not getting 5 when a fair die is rolled
ns = 6 #n(S) = {1,2,3,4,5,6}
na = 1 #n(A) = {5}
pa = na / ns #P(A)
print("probability of not getting 5 is : ",1-pa )
```

```
probability of not getting 5 is :  0.8333333333333334
```

Conditional Probability

The formula for conditional probability is

```
P(A|B) = P(A OR B) / P(B).
```

The parts P(A|B) = probability of A occuring given B occurs P(A Or B) = probability of both A and B occurring P(B) = probability of B occurring

Calculate the probability a student gets an A(80% +) in math, given they miss 10 or more classes.

In [12]:
```
import pandas as pd
import numpy as np
df = pd.read_csv('C:/Users/Akshay/Downloads/student-mat.csv')
df.head(3)
```

Out[12]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	g
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	

3 rows × 33 columns

In [13]:
```
len(df)
```

Out[13]: 395

We are only concerned with the columns , absences (number of absences), and G3 (final grade from 0 to 20).

Let us create a couple new boolean columns based on these columns to make our lives easier.

Add a boolean column called grade_A nothing if a student achieved 80% or higher as final score. Original values are on a 0 to 20 scale so we multiply by 5.

In [14]:

```
df['grade_A'] = np.where(df['G3']*5 >= 80,1,0)
```

Make another boolean column called high_absenses with a value of 1 if a student missed 10 or more classes.

In [15]:

```
df['high_absenses'] = np.where(df['absences'] >= 10,1,0)
```

Add one more column to make building a pivot table easier

In [16]:

```
df['count'] =1
```

drop all columns we dont care about.

In [17]:

```
df = df[['grade_A','high_absenses','count']]
df.head()
```

Out[17]:

	grade_A	high_absenses	count
0	0	0	1
1	0	0	1
2	0	1	1
3	0	0	1
4	0	0	1

In [19]:

```
final = pd.pivot_table(
    df,
    values = 'count',
    index = ['grade_A'],
    columns =['high_absenses'],
    aggfunc = np.size,
    fill_value =0
)
```

In [20]:

```
print(final)
```

```
high_absenses    0    1
grade_A
```

```
0                277   78
1                 35    5
```

P(A) is the probability of a grade of 80% or greater.

P(A) = (35 + 5) / (35 + 5 + 277 + 78)=

P(B) is the probability of missing 10 or more classes. P(B) = (78 + 5) / (35 + 5 + 277 + 78)

P(A OR B) = 5 / (35 + 5 + 277 + 78)

And per the formula, P(A|B) = P(A OR B) / P(B) , put it together.

The probability of getting at least an 80% final grade , given missing 10 or more classes is 6%
Conclusion

While the learning from our specific example is clear - go to class if you want good grades.

In [40]:

```
na =(35 + 5 ) / (35 + 5 + 277 + 78) #P(A)
print("P(A) is the probability of a grade of 80% or greater : ",na)
```

```
P(A) is the probability of a grade of 80% or greater :  0.10126582278481013
```

In [41]:

```
nb = (78 + 5) / (35 + 5 + 277 + 78) #P(B)
print("P(B) is the probability of missing 10 or more classes : ", nb)
```

```
P(B) is the probability of missing 10 or more classes :  0.21012658227848102
```

In [42]:

```
ns = 5 / (35 + 5 + 277 + 78) # P(A OR B)
print("P(A OR B) : " ,ns)
```

```
P(A OR B) :  0.012658227848101266
```

In [43]:

```
ans = ns / nb
print("P(A|B) = P(A OR B) / P(B) : ",ans)
```

```
P(A|B) = P(A OR B) / P(B) :  0.060240963855421686
```

The probability of getting at least an 80% final grade , given missing 10 or more classes is 6%
Conclusion

While the learning from our specific example is clear - go to class if you want good grades.