Probability

Definition : Probability is a measure of the likelihood of an event to occur. Many events cannot be predicted with total certainty. We can predict only the chance of an event to occur i.e. how likely they are to happen, using it. Probability can range in from 0 to 1, where 0 means the event to be an impossible one and 1 indicates a certain event.The probability of all the events in a sample space adds up to 1.

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 Number of outcomes

Eg 1

In [3]:

```python
# probability of getting 3 when a die is rolled
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

Eg 2

In [1]:

```python
# 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 # P(A)
print("probability of getting atleast one head is:",pa)
```

probability of getting atleast one head is: 0.875

In [2]:

```python
# A glass jar contains 5 red, 3 blue and 2 green jelly beans. If a jelly bean is chosen at random from the jar,
#  mwhat is the 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]:

```python
# If the probability that person A will be alive in 20 years
#is 0.7 and the probability that person B will be alive in
# 20 years is 0.5, what is the probability that they will
#both be alive in 20 years?

#These are independent 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 [6]:

```python
def event_probability(n,s):
    return n/s
```

In [7]:

```
#A fair die is tossed twice. Find the probability of getting a 4 or 5 on the first toss and a 1,2, or 3 in the se
cond toss.
pa = event_probability(2,6) # probability of getting a 4 or 5 on the first toss
pb = event_probability(3,6) # probability of getting 1,2,3 in second toss
P = pa*pb
print("probability of getting a 4 or 5 on the first toss and a 1,2, or 3 in the second toss is:",P)
```

probability of getting a 4 or 5 on the first toss and a 1,2, or 3 in the second toss is: 0.166666666
66666666

In [8]:

```
# A bag contains 5 white marbles, 3 black marbles and 2 green marbles. In each draw, a marble is drawn from the b
ag
# and not replaced. In three draws, find the probability of obtaining white, black and green in that order.
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*pg))
```

the probability of obtaining white, black and green in that order is 0.041666666666666664

In [9]:

```
# Sample Space
cards = 52

# Calculate the probability of drawing a heart or a club
hearts = 13
clubs = 13
heart_or_club = event_probability(hearts, cards) + event_probability(clubs, cards)
print(heart_or_club )
```

0.5

In [10]:

```
# Calculate the probability of drawing an ace, king, or a queen
aces = 4
kings = 4
queens = 4
ace_king_or_queen = event_probability(aces, cards) + event_probability(kings, cards) + event_probability(queens,
cards)

print(heart_or_club)
print(ace_king_or_queen)
```

0.5
0.23076923076923078

In [12]:

```
# 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_probability(ace_of_heart
s, cards)
print(round(heart_or_ace, 1))
```

0.3

In [13]:

```
red_cards = 26
face_cards = 12
red_face_cards = 6
red_or_face_cards = event_probability(red_cards, cards) + event_probability(face_cards, cards) - event_probabilit
y(red_face_cards, cards)

print(round(heart_or_ace, 1))
print(round(red_or_face_cards, 1))
```

0.3
0.6

Complementary Events

For any event E1 there exists another event E1' which represents 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 represents 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.

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

eg.1

In [15]:

```python
#probabiltiy 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("probabilty of not getting 5 is:",1-pa)
```

probabilty 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 occurring, given B occurs P(A â© 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 [2]:

```python
import pandas as pd
import numpy as np
df = pd.read_csv('D:/AnitaRJ/DATA SCIENCE/MScI_DataSci_Practicals/Practical2/student-mat.csv')
df.head(3)
```

Out[2]:

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

3 rows × 33 columns

check the number of records.

In [3]:

```python
len(df)
```

Out[3]:

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 noting if a student achieved 80% or higher as a final score. Original values are on a 0 to 20 scale so we multiply by 5.

In [4]:

```python
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.

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

Add one more column to make building a pivot table easier

In [6]:

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

And drop all columns we dont care about.

In [7]:

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

Out[7]:

	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

Now we will create a pivot table from this.

In [8]:

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

In [9]:

```
print(final)
```

```
high_absenses     0    1
grade_A
0               277   78
1                35    5
```

We now have all the data we need to do our calculation. Lets start by calculating each individual part in the formula.

In our case: P(A) is the probability of a grade of 80% or greater. P(B) is the probability of missing 10 or more classes. P(A|B) is the probability of a 80%+ grade, given missing 10 or more classes.

Calculations of parts: P(A) = (35 + 5) / (35 + 5 + 277 + 78) = 0.10126582278481013 P(B) = (78 + 5) / (35 + 5 + 277 + 78) = 0.21012658227848102 P(A OR B) = 5 / (35 + 5 + 277 + 78) = 0.012658227848101266

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

P(A|B) = 0.012658227848101266/ 0.21012658227848102= 0.06

There we have it. 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 [ ]: