Probability

Probability is a branch of mathematics that deals with the likelihood of different outcomes in uncertain situations. It quantifies uncertainty and provides a measure of how likely an event is to occur.

Type *Markdown* and LaTeX: α^2

Basic Concepts in Probability

1. **Experiment**: A process or activity that generates observable results. For example, rolling a die, flipping a coin, or drawing a card from a deck.

2. **Outcome**: A single possible result of an experiment. For example, getting a 3 when rolling a die or drawing an Ace of Spades from a deck.

3. **Event**: A set of one or more outcomes. For example, rolling an even number on a die (2, 4, or 6) or drawing a red card from a deck.

4. **Sample Space (S)**: The set of all possible outcomes of an experiment. For example, for a single die roll, the sample space is {1, 2, 3, 4, 5, 6}.

5. **Probability of an Event (P)**: A measure of the likelihood that an event will occur, calculated as the ratio of the number of favorable outcomes to the total number of possible outcomes. It ranges from 0 (impossible event) to 1 (certain event).

$$P(A) = \frac{\text{Number of favorable outcomes}}{\text{Total number of possible outcomes}}$$

example : exp= "rolling a die"
x=[1,2,3,4,5,6]=6 nos #all possible outcomes-sample space

event="odd number" fav outcomes=[1,3,5]=3 nos probability(event)=p(even)=3/6

event="divisible by 3"

fav="3,6" p(event)=2/6

x=["head","tail"]=sample space p("head")=1/2

In [1]:
```
 # https://docs.google.com/spreadsheets/d/1oiviSQx20EECuNc_YuQ0Txo3begXFaTOialpKrgFNk4/edit?usp=sharing
import pandas as pd
df=pd.read_csv("C:/Users/HP/Downloads/dataset - Sheet1.csv")
df
```

Out[1]:

	Student_ID	Age	Gender	Grade_Level	Math_Score	Reading_Score
0	1	16	Male	11th	82	70
1	2	17	Female	12th	50	97
2	3	14	Female	11th	68	69
3	4	16	Female	9th	51	57
4	5	16	Female	12th	93	56
...	...	...	...	...	...	...
95	96	15	Female	12th	85	65
96	97	15	Female	10th	82	63
97	98	17	Female	11th	53	61
98	99	15	Male	9th	82	72
99	100	14	Male	12th	63	64

100 rows × 6 columns

In [2]:
```
df["Gender"].value_counts()
```

Out[2]:
```
Male      56
Female    44
Name: Gender, dtype: int64
```

In [3]:
```
df
```

Out[3]:

	Student_ID	Age	Gender	Grade_Level	Math_Score	Reading_Score
0	1	16	Male	11th	82	70
1	2	17	Female	12th	50	97
2	3	14	Female	11th	68	69
3	4	16	Female	9th	51	57
4	5	16	Female	12th	93	56
...	...	...	...	...	...	...
95	96	15	Female	12th	85	65
96	97	15	Female	10th	82	63
97	98	17	Female	11th	53	61
98	99	15	Male	9th	82	72
99	100	14	Male	12th	63	64

100 rows × 6 columns

In [4]:
```
df["Gender"].value_counts()
```

Out[4]:
```
Male      56
Female    44
Name: Gender, dtype: int64
```

In [5]:
```
df["Gender"][df["Gender"]=="Female"].value_counts()
```

Out[5]:
```
Female    44
Name: Gender, dtype: int64
```

In [6]:
```
df[df["Gender"]=="female"].value_counts()
```

Out[6]:
```
Series([], dtype: int64)
```

In [7]:
```
df["Gender"][df["Gender"]=="Female"].value_counts()
```

Out[7]:
```
Female    44
Name: Gender, dtype: int64
```

In [8]:
```
len(df[df["Gender"]=="Female"])/len(df)
```

Out[8]:
```
0.44
```

In [ ]:

In [ ]:

What is the probability of selecting a student who is a female?

44/100

In [9]:
```
df["Gender"].value_counts()
```

Out[9]:
```
Male      56
Female    44
Name: Gender, dtype: int64
```

In [10]:
```
df["Gender"][df["Gender"]=="Female"].value_counts()
```

Out[10]:
```
Female    44
Name: Gender, dtype: int64
```

In [11]:
```
len(df)
```

Out[11]:
```
100
```

In [12]:
```
total_students = len(df)

prob_female = len(df[df['Gender'] == 'Female']) / total_students
prob_female
```

Out[12]: 0.44

2. Probability of selecting a student in the 11th grade

In [13]:
```
elev_th=df[df["Grade_Level"]=="11th"]
```

In [14]:
```
len(elev_th)
```

Out[14]: 23

In [15]:
```
prob_11th_grade = len(df[df['Grade_Level'] == '11th']) / len(df)
prob_11th_grade
```

Out[15]: 0.23

3. Probability of selecting a student who scored more than 80 in math

In [16]:
```
prob_math_above_80 = len(df[df['Math_Score'] > 80]) / total_students
prob_math_above_80
```

Out[16]: 0.4

4. Probability of selecting a male student who scored more than 90 in reading

In [ ]:

In [17]:
```
df["Gender"]=="Male" # condition 1
```

Out[17]:
```
0      True
1     False
2     False
3     False
4     False
      ...
95    False
96    False
97    False
98     True
99     True
Name: Gender, Length: 100, dtype: bool
```

In [18]:
```
df["Reading_Score"]>90 # condition 2
```

Out[18]:
```
0     False
1      True
2     False
3     False
4     False
      ...
95    False
96    False
97    False
98    False
99    False
Name: Reading_Score, Length: 100, dtype: bool
```

In [19]:
```
(df["Gender"]=="Male") & (df["Reading_Score"]>90) #(cond-1) & (cond 2)
```

Out[19]:
```
0     False
1     False
2     False
3     False
4     False
      ...
95    False
96    False
97    False
98    False
99    False
Length: 100, dtype: bool
```

In [20]:
```
# selecting from df
df[(df["Gender"]=="Male") & (df["Reading_Score"]>90)] #(cond-1) & (cond 2)
```

Out[20]:

	Student_ID	Age	Gender	Grade_Level	Math_Score	Reading_Score
16	17	17	Male	12th	55	94
32	33	17	Male	12th	68	98
34	35	15	Male	12th	81	93
38	39	14	Male	11th	89	95
45	46	17	Male	10th	74	98
46	47	17	Male	9th	99	95
61	62	16	Male	10th	54	95
65	66	15	Male	10th	90	96
77	78	17	Male	10th	57	92
78	79	16	Male	9th	85	97
81	82	17	Male	12th	69	91
84	85	16	Male	10th	74	99
90	91	17	Male	9th	67	94
94	95	15	Male	10th	90	95

In [21]:
```
len(df[(df["Gender"]=="Male") & (df["Reading_Score"]>90)]) #(cond-1) & (cond 2)
```

Out[21]: 14

In [22]:
```
prob_male_reading_above_90 = len(df[(df['Gender'] == 'Male') & (df['Reading_Score'] > 90)]) / total_students
prob_male_reading_above_90
```

Out[22]: 0.14

In [23]:
```
len(df[(df["Reading_Score"]>90) & (df["Gender"]=="Male")])
```

Out[23]: 14

4. Probability of selecting a student who is 16 or 15?

In [24]:
```
len(df[(df["Age"]==16) | (df["Age"]==15)])
```

Out[24]: 50

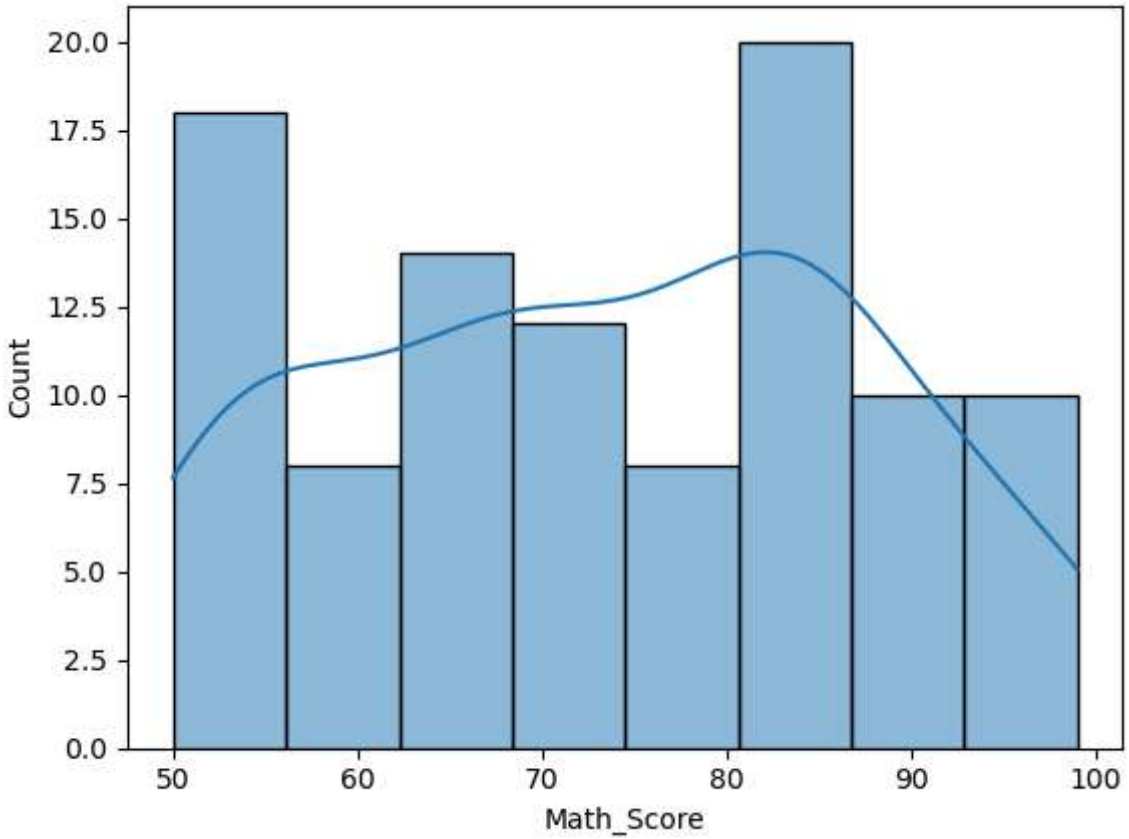
task ?

12 th grade student & math >80

probability distributions

In [26]:
```
#normal
```

In [27]:
```
age=25

x=df["Math_Score"]
import seaborn as sns
import matplotlib.pyplot as plt
sns.histplot(x,kde=True)
plt.show()
```



In [28]:
```python
import numpy as np

# Define the list x
x = [ 6, 15, 10 , 6 ,13]


# Calculate the mean
mean_x = np.mean(x)

# Calculate the standard deviation
std_x = np.std(x)

mean_x, std_x
```

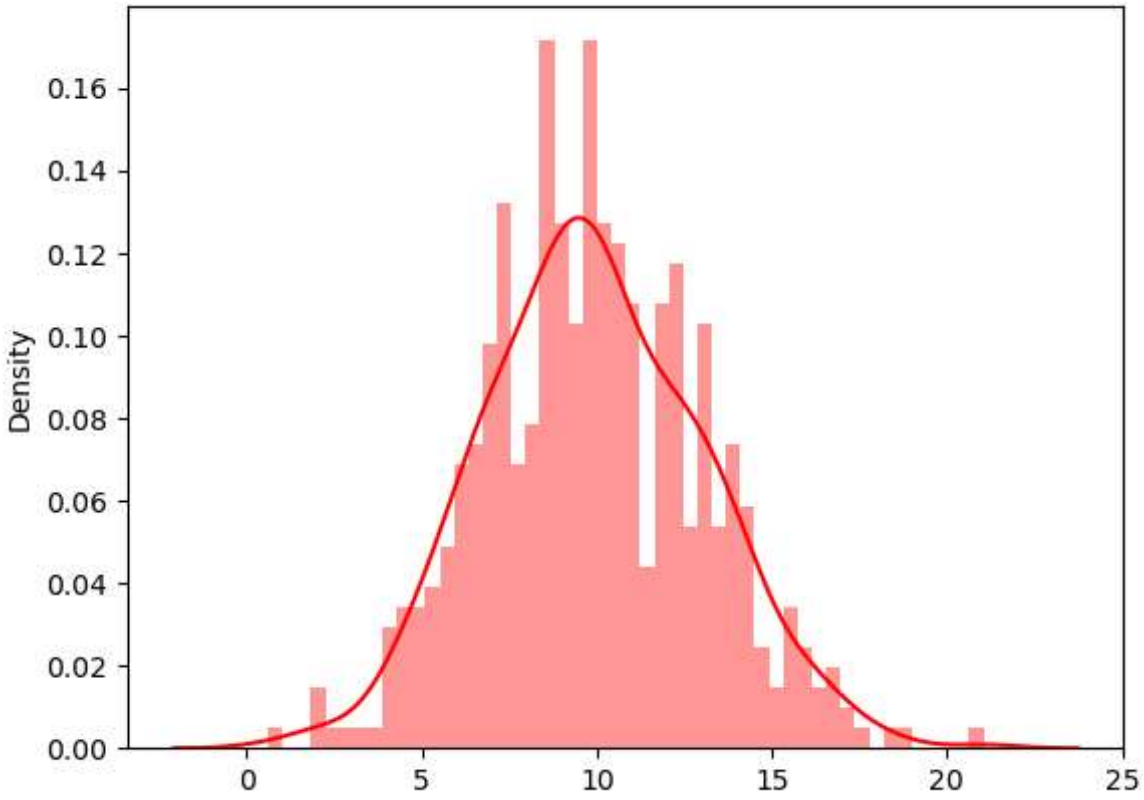
Out[28]:
```
(10.0, 3.63318042491699)
```

Normal Distribution

- **Shape**: Bell-shaped and symmetric around the mean.
- **Parameters**: Mean (μ) and standard deviation (σ).
- **Characteristics**:
  - Most data points cluster around the mean.
  - About 68% of data falls within 1 standard deviation from the mean, 95% within 2 standard deviations, and 99.7% within 3 standard deviations (Empirical Rule).
- **Usage**: Often used to model natural phenomena (e.g., heights, test scores).

In [29]:
```
# import packages
import scipy.stats as stats
import seaborn as sns
import matplotlib.pyplot as plt

import numpy as np

# generate data
data =np.random.normal(10,3,500)
# plotting a histogram
data

sns.distplot(data,bins=50,kde=True,color='red')

plt.show()
```

```
C:\Users\HP\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a fig
ure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

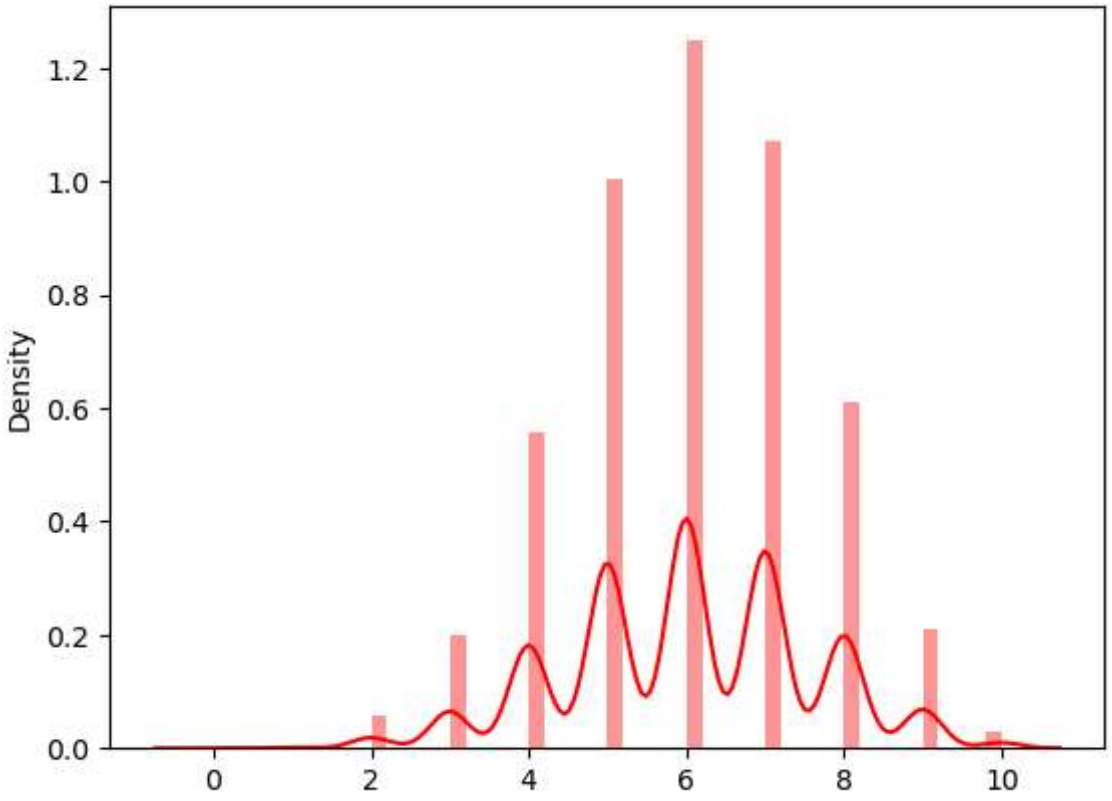

Density
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25

Binomial Distribution

- **Shape**: Varies depending on the parameters; can be symmetric or skewed.
- **Parameters**: Number of trials (n) and probability of success (p).
- **Characteristics**:
  - Models the number of successes in a fixed number of independent trials.
  - Each trial has two possible outcomes (success or failure).
  - The probability of success (p) is constant for each trial.
- **Usage**: Used for scenarios with a fixed number of trials and two outcomes, like flipping a coin or pass/fail exams.

In [30]:
```
# import packages
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# generate data
# n== number of trials,p== probability of each trial
n, p = 10, .6
data = np.random.binomial(n, p, 10000)

# plotting a histogram
sns.distplot(data,kde=True,color='red')

plt.show()
```

```
C:\Users\HP\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a fig
ure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

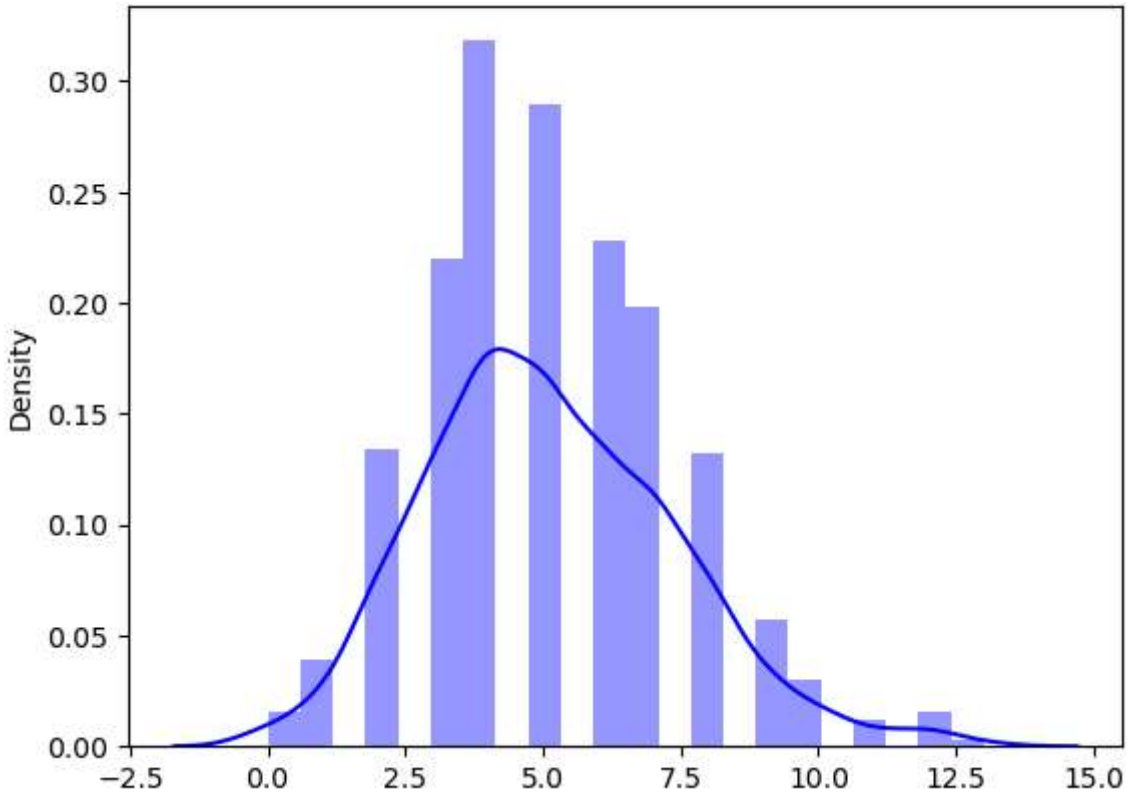

Density
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10

Poisson Distribution

- **Shape**: Skewed to the right, but becomes more symmetric as the mean increases.
- **Parameters**: Mean (λ), which is the average number of events in a given time period or space.
- **Characteristics**:
  - Models the number of events occurring within a fixed interval of time or space.
  - Events occur independently.
  - The average rate (λ) is constant.
- **Usage**: Often used for rare events over a continuous interval, like the number of emails received per hour or the number of accidents at a crossing.

In [31]:
```
# import packages
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# generate poisson data
poisson_data = np.random.poisson(lam=5, size=1000)

# plotting a histogram
sns.distplot(poisson_data,kde=True,color='blue')

plt.show()
```

```
C:\Users\HP\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a fig
ure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [32]:

Out[32]: 0

In [33]:

```
  File "C:\Users\HP\AppData\Local\Temp\ipykernel_9828\1732651236.py", line 1
    and
    ^
SyntaxError: invalid syntax
```

In [ ]: