Double-click (or enter) to edit

## Sample Spaces and Probability

A probability experiment is a chance process that leads to well-defined results called **outcomes**.

An outcome is the result of a single trial of a probability experiment.

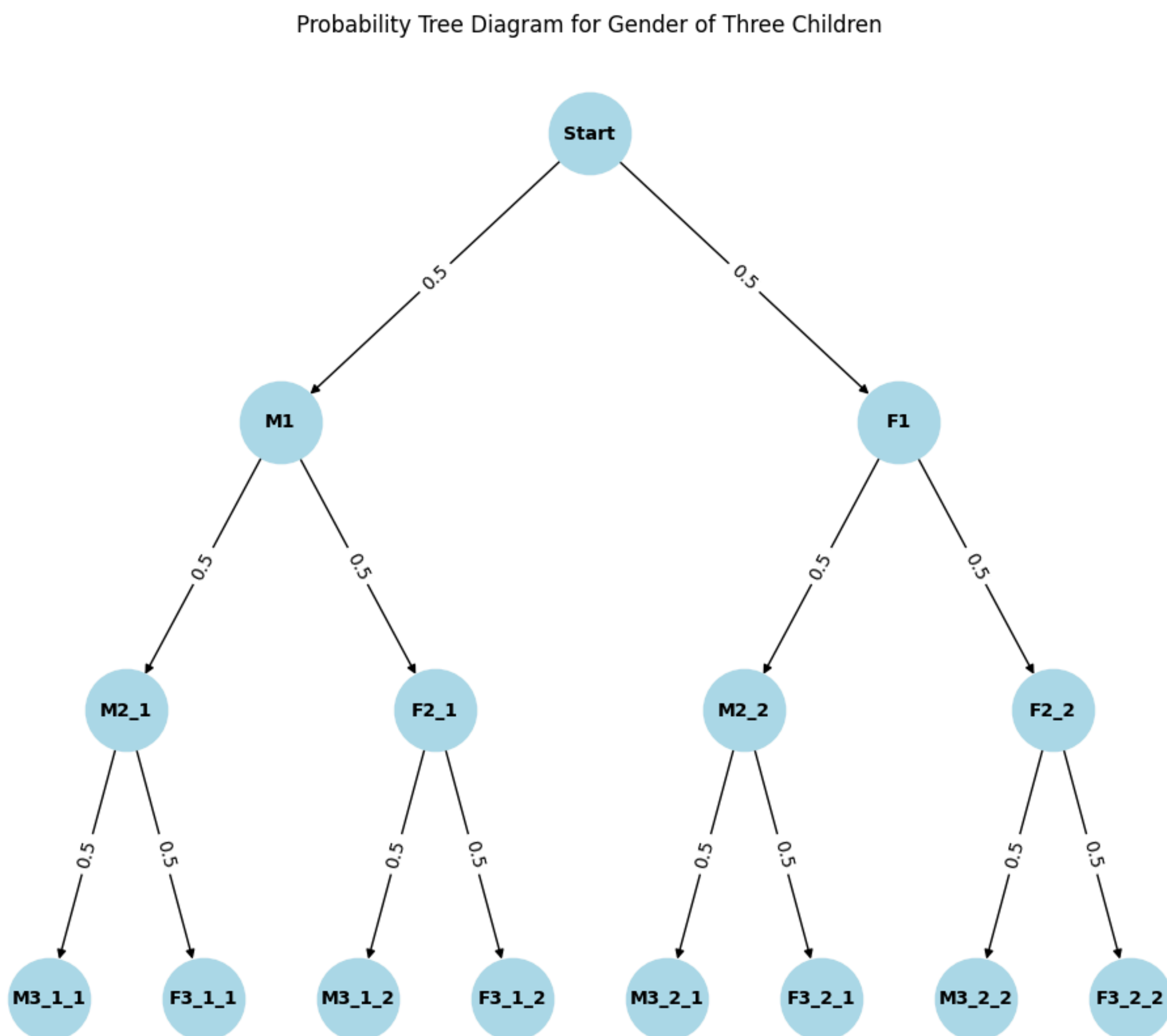
### Tree Diagram

A **tree diagram** is a device consisting of line segments emanating from a starting point and also from the outcome point. It is used to determine all possible outcomes of a probability experiment.

For example: Use a tree diagram to fund the sample space for the gender of three children in a family.

```
import networkx as nx
import matplotlib.pyplot as plt

# Define the probability tree structure
G = nx.DiGraph()

# Add nodes for events (representing the gender of children)
G.add_node("Start", label="Start")

# Add second-level nodes (gender of 1st child)
G.add_node("M1", label="M (Child 1)")
G.add_node("F1", label="F (Child 1)")

# Add third-level nodes (gender of 2nd child, branching from each 1st child)
G.add_node("M2_1", label="M (Child 2)")
G.add_node("F2_1", label="F (Child 2)")
G.add_node("M2_2", label="M (Child 2)")
G.add_node("F2_2", label="F (Child 2)")

# Add fourth-level nodes (gender of 3rd child, branching from each 2nd child)
G.add_node("M3_1_1", label="M (Child 3)")
G.add_node("F3_1_1", label="F (Child 3)")
G.add_node("M3_1_2", label="M (Child 3)")
G.add_node("F3_1_2", label="F (Child 3)")
G.add_node("M3_2_1", label="M (Child 3)")
G.add_node("F3_2_1", label="F (Child 3)")
G.add_node("M3_2_2", label="M (Child 3)")
G.add_node("F3_2_2", label="F (Child 3)")

# Add edges between nodes (with probabilities)
G.add_edge("Start", "M1", label=0.5)
G.add_edge("Start", "F1", label=0.5)

G.add_edge("M1", "M2_1", label=0.5)
G.add_edge("M1", "F2_1", label=0.5)
G.add_edge("F1", "M2_2", label=0.5)
G.add_edge("F1", "F2_2", label=0.5)

G.add_edge("M2_1", "M3_1_1", label=0.5)
G.add_edge("M2_1", "F3_1_1", label=0.5)
G.add_edge("F2_1", "M3_1_2", label=0.5)
G.add_edge("F2_1", "F3_1_2", label=0.5)
G.add_edge("M2_2", "M3_2_1", label=0.5)
G.add_edge("M2_2", "F3_2_1", label=0.5)
G.add_edge("F2_2", "M3_2_2", label=0.5)
G.add_edge("F2_2", "F3_2_2", label=0.5)

# Positioning for tree layout (creating a clean tree structure)
pos = {
    "Start": (0, 0),
    "M1": (-1, -1), "F1": (1, -1),
    "M2_1": (-1.5, -2), "F2_1": (-0.5, -2),
    "M2_2": (0.5, -2), "F2_2": (1.5, -2),
    "M3_1_1": (-1.75, -3), "F3_1_1": (-1.25, -3),
    "M3_1_2": (-0.75, -3), "F3_1_2": (-0.25, -3),
    "M3_2_1": (0.25, -3), "F3_2_1": (0.75, -3),
    "M3_2_2": (1.25, -3), "F3_2_2": (1.75, -3),
```

```python
    "M3_2_2": (1.25, -3), "F3_2_2": (1.75, -3),
}

# Draw the tree
plt.figure(figsize=(10, 8))
nx.draw(G, pos, with_labels=True, node_size=2000, node_color="lightblue",
font_size=10, font_weight="bold", arrows=True)

# Draw edge labels (probabilities)
edge_labels = nx.get_edge_attributes(G, 'label')
nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels)

plt.title("Probability Tree Diagram for Gender of Three Children")
plt.show()
```




There are cases to distinguish between an outcome and an event.

An **event** consistis of a set of outcomes of probalility experiment.
There are three basic interoretations of probability:

1. Classical probability
2. Emperical or relative frequency probability
3. Subjective probability

## Classical Probability

**Classical probability** uses sample spaces to determine the numerical probability tgat an event will happen. It assumes that all the outcomes in the sample space are equally likely to occur.

**Equally likely events** are events that have the same probability of occuring.

Mathematically,
The probability of any event E is $\frac{Number\ of\ outcomes\ in\ E}{Total\ number\ of\ outcomes\ in\ the\ sample\ space}$

It can be denoted by
$P(E) = \frac{n(E)}{n(S)}$

For example:
Find the probability of getting red face card (jack, queen, king) when randomly drawing a card from an ordinary deck.

```
total_card = 52
red_face_card = 6 # 3 heart and 3 diamond
probability = red_face_card/total_card

print(f"The probability of getting a red face card is {round(probability,3)}")
```

```
The probability of getting a red face card is 0.115
```

The above answer can also be represented in fraction or in percentage.

```
percentage = probability * 100
print(f"The probability in percentage is {percentage:.2f}%")
```

```
The probability in percentage is 11.54%
```

```
from fractions import Fraction
fraction = Fraction(red_face_card, total_card)
print(f"The probability in fraction is {fraction}")
```

```
The probability in fraction is 3/26
```

## Question 1

A card is drawn from an ordinary deck. Find the probability of getting a red card.

```
## Write Your Code Here ##
```

## Probability Rules

1. The probability of any event E is a number (either a fraction or decimal) between and including 0 and 1. This is denoted by $0 <= P(E) <= 1$.
2. The sum of the probabilities of all the outcomes in a sample space is 1.
3. If an event E cannot occur (i.e. the event contains no members in the sample space), its probability is 0.
4. If an event E is certain, then the probability of E is 1.

## Complementary Events

The complement of an event E is the set of outcomes in the sample space that are not included in the outcomes of event E. The complement of E is denoted by $\bar{E}$. Mathematically,
$P(\bar{E}) = 1 - P(E)$

For example:
Find the complement of selecting a month that has 31 days.

```
total_months = 12 # total months in a year
months_with_31_days = 7 # Jan, Mar, May, Jun, Aug, Oct, Dec
probability = months_with_31_days/total_months # calculating probability

print(f"The probability of selecting a month with 31 days is {round(probability,3)}")

complementary_events = 1 - probability # calculating complementary events
```

```

print(f"The complementary events of selecting a month that has 31 days is {round(complementary_events,3)}")
```

```
The probability of selecting a month with 31 days is 0.583
The complementary events of selecting a month that has 31 days is 0.417
```

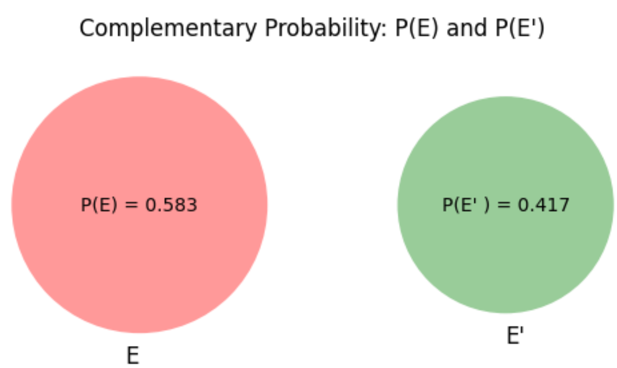
## Question 2

Find the complement of selecting a day of the week that begins with the letter 'T'.

```
##Write your code here. ##\
```

Showing the complementary event in venn-diagram.

```

import matplotlib.pyplot as plt
from matplotlib_venn import venn2

# Plotting the Venn diagram
venn = venn2(subsets=(probability, complementary_events, 0), set_labels=('E', "E'"))
venn.get_label_by_id('10').set_text(f'P(E) = {probability:.3f}')
venn.get_label_by_id('01').set_text(f'P(E\' ) = {complementary_events:.3f}')
plt.title("Complementary Probability: P(E) and P(E')")
plt.show()

```




## Emperical Probability

Emperical prbability calculates the actual experience of the likelihood of an event unlike the classical probability which assumes all the outcomes are equally likely to come.
Mathematically,
$P(E) = \frac{frequency\ for\ the\ class}{total\ frequencies\ in\ the\ distribution} = \frac{f}{n}$
For example:
In the travel survey, find the prbability that a person will travel by airplane over the Thanksgiving holiday.

| Method | Frequency |
|---|---|
| Drive | 41 |
| Fly | 6 |
| Train or bus | 3 |
| Total | 50 |

```
# Given from the table
frequency_airplane = 6
total_frequencies = 50
probability = frequency_airplane/total_frequencies

# displaying the result
print(f"The probability that a person will travel by airplance is {probability:.3f}")
```

```
The probability that a person will travel by airplance is 0.120
```

## Question 3

Find the probability of a person travelling by driving over the Thanksgiving.

```
## Write your code here ##
```

## Adding of the events

For example:

Hospital records indicated the knee replacement patients stayes in the hospital for the number of days shown in the distribution.

| Number of days stayed | Frequency |
|---|---|
| 3 | 15 |
| 4 | 32 |
| 5 | 56 |
| 6 | 19 |
| 7 | 5 |
| Total | 127 |

Question: A patient stayed fewer than 6 days.

```
# fewer than 6 days in the table are 3, or 4, or 5 days
# given from the table
days_3 = 15
days_4 = 32
days_5 = 56
total_patients = 127
probability_3_days = days_3/total_patients # probability of 3 days
probability_4_days = days_4/total_patients # probability of 4 days
probability_5_days = days_5/total_patients # probability of 5 days

# calculating the probability of a patient staying fewer than 6 days.
# 3, or 4, or 5 days. So tha probabilities are added.
probability_fewer_6_days = probability_3_days + probability_4_days + probability_5_days

# displaying the result
print(f"The probability of a patient stayed fewer tha 6 days is {probability_fewer_6_days:.3f}")
```

```
The probability of a patient stayed fewer tha 6 days is 0.811
```

## Question 4

From the above data of the hospital. Calculate the probability for a patient stayed at least 5 days.

```
## Write your code here ##
```