lab_1_solutions

January 26, 2025

title: “LAB - 1 - Teoria de la Probabilidad”

author: “Adrian Arimany (211063) - Daniel Sarmiento (231105)”

date: “Enero 26, 2025”

[1]:
```
#USE for import any package or internal file.
import itertools as it
from collections import Counter
```

0.1 Problema 1

1. Una empresa tiene un grupo de 8 empleados Ana, Bruno, Carlos, Dina, Eli, Frank, Gaby y Hugo, de los cuales necesita formar equipos de 3 personas para un proyecto.
   a. ¿Cuántos equipos diferentes de 3 personas pueden formarse? ¿En cuántos de éstos está Ana?
   b. ¿Cuántos equipos diferentes de 4 personas pueden formarse? ¿En cuántos de éstos no está Frank?

[2]:
```
# Problema A1
print("Problema A1:")
S = {"A", "B", "C", "D", "E", "F", "G", "H"}

# Para simplificar, tomemos solo la primera letra de los nombres.
re1 = set(it.combinations(S, 3))
result = len(re1)

getA = 0
for i in re1:
     if "A" in i:
          getA += 1


print(f"El numero total de grupos de 3 es {result}")
print(f"La cantidad de grupos que contienen Ana es de {getA}")

# Resultado: 56, 21
```

```
# Problema B1
print("Problema B1:")

re2 = set(it.combinations(S, 4))
result = len(re2)

getF = 0
for i in re2:
     if "A" in i:
          getF += 1


print(f"El numero total de grupos de 4 es {result}")
print(f"La cantidad de grupos que contienen Frank es de {getF}")

#Resultado: 70, 35
```

```
Problema A1:
El numero total de grupos de 3 es 56
La cantidad de grupos que contienen Ana es de 21
Problema B1:
El numero total de grupos de 4 es 70
La cantidad de grupos que contienen Frank es de 35
```

0.2 Problema 2:

a. ¿Cuántas palabras diferentes se pueden formar usando todas las letras de la palabra SEERESS

b. ¿Cuántas de estas palabras comienzan con S y terminan con R?

c. ¿Cuántas palabras diferentes se pueden formar usando cinco o más letras de la palabra SEERES

[3]:
```
#Problema A2
S = "SEERESS"
result = set(it.permutations(S))
resultA = len(result)

print(f"El conjunto de posibles palabras usando la palabra SEERESS es S:␣
 ↪{resultA}")

#result 140
#Problema B2
ansB = sum(1 for k in result if k[0] == "S" and k[6] == "R")
print(f"La cantidad de palabras que empiezan en S y terminan en R: {ansB}")
#result 10
#Problema C2

total_unique_words = 0
```

```
letter_count = Counter(S)

for length in range(5, 8):
    perms = set()
    for perm in it.permutations(S, length):
        word_perm = ''.join(perm)
        if Counter(word_perm) <= letter_count:
            perms.add(word_perm)
    total_unique_words += len(perms)

print(f"Numero total de palabras con cinco o mas letras: {total_unique_words}")
```

```
El conjunto de posibles palabras usando la palabra SEERESS es S: 140
La cantidad de palabras que empiezan en S y terminan en R: 10
Numero total de palabras con cinco o mas letras: 370
```

0.3 Problema 3:

¿Cuántos números entre 1 y 1,000,000 son primos?

[4]:
```
#Problema A3

def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

# Contar números primos entre 1 y 1,000,000
primos = [n for n in range(1, 1000001) if is_prime(n)]
total_primos = len(primos)

total_primos
```

[4]: 78498

0.4 Problema 4

a. ¿Cuántos números decimales de 4 dígitos se pueden formar usando los dígitos del 0 al 9?

b. ¿Cuántos tienen uno o más dígitos repetidos? Ej.: 2324, 5255, 6767, etc.

c. ¿Cuántos tienen los 4 dígitos repetidos? Ej.: 3333.

d. ¿Cuántos tienen dos dígitos (diferentes) repetidos dos veces cada uno? Ej.: 2424.

e. ¿Cuántos de estos repiten un dígito dos veces y los otros no se repiten? Ej.: 3137.

f. ¿Cuántos de estos repiten un dígito tres veces y el otro no se repite? Ej.: 9199.

[5]:
```
#Problema A4

#A4

S = it.product(range(10), repeat=4)
valid_numbers = [k for k in S if k[0] != 0]
# Exluide todo los tuples que inicial en 0, ya que se buscan solo los␣
 ↪decimales, y un decimal no empieza con 0
ansA = len(valid_numbers)
print(f"El numero total de digitos de 4 que se pueden formar son: {ansA}")

#9,000 números decimales de 4 dígitos

#B4
unrepeated_values = [k for k in valid_numbers if len(set(k)) == len(k)]
ansBComplement = len(unrepeated_values)
#if we know that there are 9000 tuples in total,
# and we know that 4536 of them are unique,
# so by the complementary property we can determine the solution of B with the␣
 ↪following:
ansB = ansA - ansBComplement
print(f"El numero total de digitos de 4, que tienen 1 o mas numeros repetidos␣
 ↪son: {ansB}")
#4464 números con uno o más dígitos repetidos
#C4
repeated_digits = [int("".join(map(str, combo)))for combo in valid_numbers  if␣
 ↪len(set(combo)) == 1 ]
ansC = len(repeated_digits)
print(f"Numero total de digitos de 4 que se repiten los 4 digitos: {ansC},␣
 ↪excluyendo el evento {{0000}}")

#9 números con los 4 dígitos repetidos
#D4
digitsTwice = [r for r in valid_numbers if len(set(r)) == 2 and all(r.
 ↪count(digit) == 2 for digit in set(r))]
ansD = len(digitsTwice)
print(f"Numero total de digitos de 4 que se repiten dos veces son: {ansD}")
#soln: 243
#E4
repeatedOnce = [r for r in valid_numbers if len(set(r)) == 3 and any(r.
 ↪count(digit) == 2 for digit in set(r))]
ansE = len(repeatedOnce)
print(f"Numero total de digitos de 4 que se repinten escritamente 2 veces 1 par␣
 ↪son {ansE}")
```

```
#F4

repeatedThreeTimes = [r for r in valid_numbers if len(set(r)) == 2 and any(r.
 ↪count(digit) == 3 for digit in set(r))]
ansD = len(repeatedThreeTimes)
print(f"Numero total de digitos de 4 que se repinten tres veces y el otro no␣
 ↪son {ansD}")

#360 números que repiten un dígito tres veces y el otro no se repite
```

```
El numero total de digitos de 4 que se pueden formar son: 9000
El numero total de digitos de 4, que tienen 1 o mas numeros repetidos son: 4464
Numero total de digitos de 4 que se repiten los 4 digitos: 9, excluyendo el
evento {0000}
Numero total de digitos de 4 que se repiten dos veces son: 243
Numero total de digitos de 4 que se repinten escritamente 2 veces 1 par son 3888
Numero total de digitos de 4 que se repinten tres veces y el otro no son 324
```

0.5 MUY IMPORTANTE sobre el Problema 4

El problema 4 no define claramente cual es el espacio muestral en este ejercisio y por ende los incisos no se entiende cual resultado se deberian de considerar.

A lo que me refiero especificamente es lo siguente, el caso donde el valor "0" esta en el primer digito o primer elemento parte del espacio de eventos, i.e. {0143}, {0249} etc?

La confusion es, se deberia de considerar el caso anteriormente mencionado parte del sigma-algebra?

Nosotros lo tomamos como "NO", y la razon es considere el siguente ejemplo: el numero "0143" es equivalente a "143", pero "143" no tiene 4 digitos, y por ello no es parte del espacio muestral, dado que el espacio muestral son todos aquellos numeros que tiene 4 digitos.

En el codigo, esto fue programado como "valid_numbers = [k for k in S if k[0] != 0]", y por ello en todos los metodos/loops donde ejecutan "valid_numbers" no consideran los eventos donde el "0" esta posicionado como primer digito. Y por ello tengo los resultados que usted puede revisar en el chunk anterior.

Menciono esto porque en este grupo nos estaban dando resultados diferentes y sin mucha dificultad se logro observar que los resultados varian dependiendo de como uno interpreta el espacio muestra. Claramente, no se quiere perder puntos porque no se tuvo los "resultados adecuados", y por ello estoy escribiendo esto para justificar los resultados anteriormente mencionados.

Abajo copie el codigo, modificando 'valid_numbers' para que incluya los eventos que tienen el 0 al principio, y como puede observar los resultados son diferentes a los que fueron anteriormente mencioados.

[6]:
```
#Problema A4 -modificado
#Estos fueron los resultados que no coincidiean en este grupo.
S = it.product(range(10), repeat=4)
valid_numbers = [k for k in S]
```

```python
#D4
digitsTwice = [r for r in valid_numbers if len(set(r)) == 2 and all(r.
 ↪count(digit) == 2 for digit in set(r))]
ansD = len(digitsTwice)
print(f"Numero total de digitos de 4 que se repiten dos veces son: {ansD}")

#E4
repeatedOnce = [r for r in valid_numbers if len(set(r)) == 3 and any(r.
 ↪count(digit) == 2 for digit in set(r))]
ansE = len(repeatedOnce)
print(f"Numero total de digitos de 4 que se repinten escritamente 2 veces 1 par␣
 ↪son {ansE}")
#F4

repeatedThreeTimes = [r for r in valid_numbers if len(set(r)) == 2 and any(r.
 ↪count(digit) == 3 for digit in set(r))]
ansD = len(repeatedThreeTimes)
print(f"Numero total de digitos de 4 que se repinten tres veces y el otro no␣
 ↪son {ansD}")
```

```
Numero total de digitos de 4 que se repiten dos veces son: 270
Numero total de digitos de 4 que se repinten escritamente 2 veces 1 par son 4320
Numero total de digitos de 4 que se repinten tres veces y el otro no son 360
```