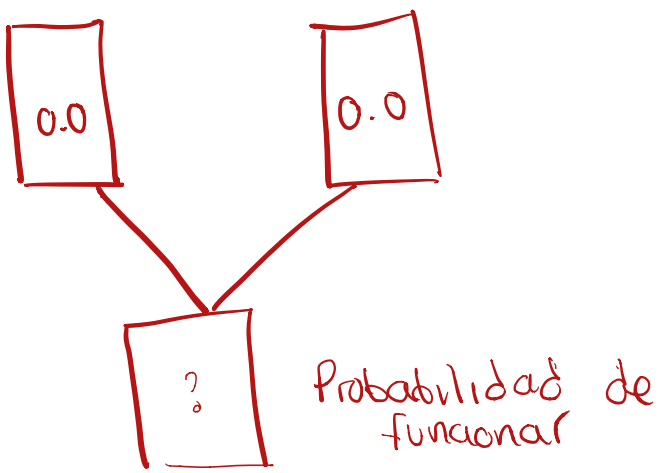
Breakdown:

El problema de training cores trata de repartir ciertos puntos de mejora entre los cores disponibles. Estos puntos se utilizan para aumentar la probabilidad de que funcionen los cores.

Cada caso cuenta con tres inputs. El primer input recibe dos números, el número de cores disponibles y el número de cores necesarios para que funcione el Dataset. El segundo input recibe el número de puntos que se pueden distribuir a lo largo de los cores disponibles. El tercer input cuenta con la probabilidad que tiene cada core disponible de funcionar

Ejemplo

2 2
1.0000
0.000 0.000

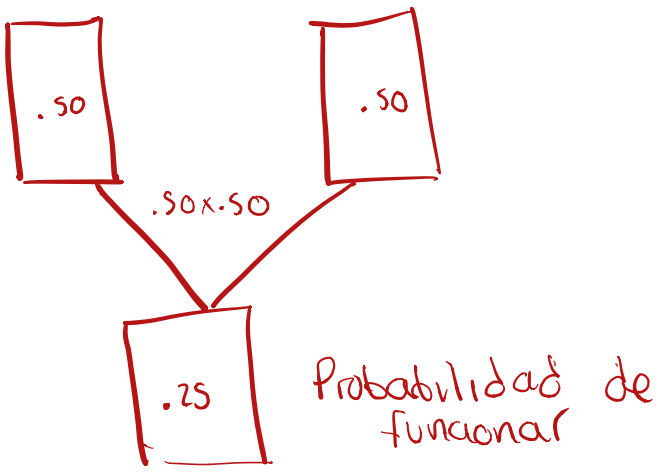


Existen dos tipos de casos, en donde el número de cores disponibles sean el mismo número de cores necesarios para que funcione el dataset, y en donde el número de cores disponibles sea mayor o igual al número de cores necesarios para que funcione el dataset.

En el ejemplo anterior, el número de cores disponibles es igual al número de cores necesarios, por lo que es necesario que la distribución sea lo más equitativa posible.

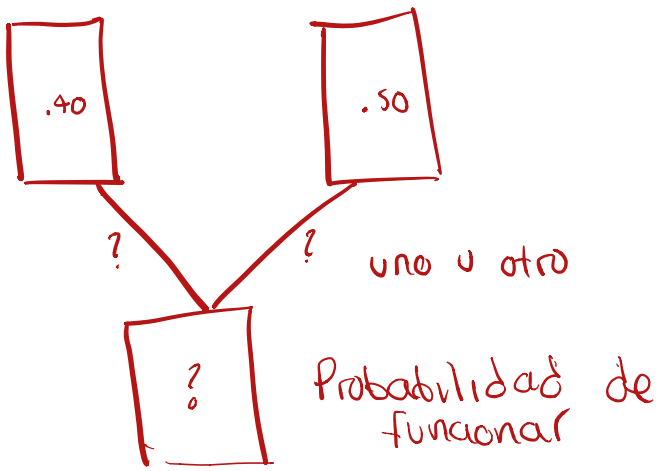
En este caso, ambos cores no tienen probabilidad de funcionar y contamos con 1.000, por lo que simplemente repartimos por mitad a cada core.

Una vez repartidos los puntos, hay .50% y .50% de probabilidad de que funcione el dataset. Para sacar la probabilidad final, se multiplican ambas.

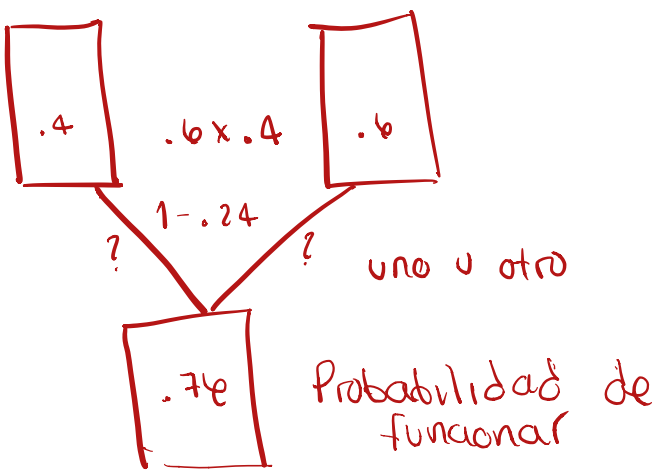


El segundo caso es cuando hay más cores de los necesarios. Para esto, es mejor que haya más puntos en un core del otro, por lo que se busca que los puntos se repartan al core con mayor probabilidad hasta llegar a 1.00 y luego cambia al siguiente mayor.

Ejemplo
2 1
.1000
.400 .500



En este caso, el segundo core cuenta con mayor probabilidad de funcionar, por lo que se le reparten los puntos. ya que solo uno es necesario, se busca la probabilidad que tiene cada uno de fallar y luego se le restan a 1.



Conclusión:

Existen dos posibles casos

- Cores disponibles = Cores necesarios ≥ 1
- Cores disponibles > cores necesarios ≥ 1

Para el primer case, los puntos se dividen de manera en que se busque que todos tengan la misma probabilidad y luego se multiplican las probabilidades para conseguir una final

Para el segundo caso, los puntos se reparten al core de mayor eficacia hasta llegar a 1.0 y luego busca al siguiente mayor. Se sacan las probabilidades de falla de cada uno, se multiplican entre sí y finalmente se le restan a 1.0 para sacar la probabilidad final