TP : probabilits 2
Informatique pour tous

I Problme de Monty Hall

Le problme de Monty Hall (version CPGE) provient d'un jeu tlvis, o un candidat est face 3 portes. Derrire l'une d'elles se cache une admission l'cole Polytechnique [1], et rien [2] derrire les deux autres.
Le candidat choisit une porte P (uniformment au hasard). Puis le prsentateur choisit une porte, diffrente de P, o il n'y a rien. Si le prsentateur a le choix entre 2 portes, on suppose qu'il en choisit une avec probabilit $\frac{1}{2}$.
Le prsentateur demande alors au candidat s'il souhaite changer de porte, puis le candidat gagne ce qu'il y a derrire la porte.

1. votre avis, le candidat a t-il intrt changer de porte ?
2. Quelle est la probabilit de gagner en ne changeant jamais de porte ?
3. On veut conjecturer exprimentalement la probabilit de gagner en changeant toujours de porte, en simulant 1000 parties.
   On numrote les portes de 0 2 et, pour chaque partie : on choisit au hasard o mettre le cadeau, on simule au hasard le choix du candidat puis du prsentateur, et on regarde si cette partie est gagnante. On calcule enfin le nombre de parties gagnantes divis par le nombre de parties totales.
   On crira `import numpy.random as rd` puis on utilisera `rd.randint(a, b)` pour gnrer un nombre entier a et b (exclu) uniformment au hasard.
4. Prouver la conjecture prcdente.

II Probabilits sur les ensembles

Soit $n \in \mathbb{N}$ et $E_n = \{0, 1, ..., n-1\}$. Il existe 2^n sous-ensembles de E_n. On en choisit deux, A et B, indpendamment et uniformment au hasard (chaque sous-ensemble a probabilit $\frac{1}{2^n}$ d'tre choisi). On va calculer exprimentalement la probabilit $\mathbb{P}(A \subseteq B)$ que A soit inclus dans B. Pour cela, on va numrer tous les sous-ensembles de E_n.
Une faon simple d'numrer ces sous-ensembles consiste coder un sous-ensemble S de E_n par un nombre entier dont la reprsentation binaire possde un 1 en position i si et seulement si $i \in S$.
Par exemple, $S = \{0, 2, 3\}$ est reprsent par :

$$< \underbrace{0}_{4 \notin S} \underbrace{1}_{3 \in S} \underbrace{1}_{2 \in S} \underbrace{0}_{1 \notin S} \underbrace{1}_{0 \in S} >_2 = 2^0 + 2^2 + 2^3 = 13$$

1. crire une fonction `set_to_int` convertissant un ensemble (sous forme de liste) en un entier le reprsentant. Par exemple, `set_to_int([0, 2, 3])` doit renvoyer `13`.
2. crire une fonction `int_to_set` convertissant un entier en un ensemble le reprsentant, sous forme de liste. Par exemple, `int_to_set(13)` peut renvoyer `[0, 2, 3]`.

Dans la suite, on utilisera les fonctions prcdentes uniquement pour vrifier les rsultats.
On utilisera la fonction suivante qui permet de calculer l'intersection de deux ensembles reprsent par des entiers. Le rsultat est aussi un entier.

```
def inter(A, B):
    return A & B
```

3. En dduire une fonction `inclus` telle que `inclus(A, B)` renvoie `True` si et seulement si l'ensemble reprsent par l'entier A est inclus dans celui reprsent par l'entier B.
   On pourra utiliser le fait qu'on peut tester si deux ensembles sont les mmes avec `==`.

1. Une voiture dans le jeu initial
2. Une chvre dans le jeu initial

4. Quels sont les entiers reprsentant les sous-ensembles de E_n ? On peut donc tous les parcourir avec une boucle `for`.
5. numrer alors tous les sous-ensembles A et B de E_n (pour des valeurs pas trop grandes de n) et compter la proportion de ceux vrifiant $A \subseteq B$, avec la fonction `inclus`.
6. Quelle conjecture faites-vous sur $\mathbb{P}(A \subseteq B)$? La dmontrer mathmatiquement.