Session d'exercices – Entropie

Dans la partie théorique du cours, nous avons étudié la définition de l'entropie. Nous proposons de généraliser ce concept en concevant un programme en Python permettant de calculer l'entropie d'un fichier. En préparation, vous avez exploré les dictionnaires, les n-uplets (*tuples*) et les ensembles (*sets*). Vous avez maintenant l'opportunité d'appliquer ces compétences nouvellement acquises à un exercice plus large et utile.

En théorie de l'information, l'entropie d'une source de valeurs mesure la quantité d'information qu'elle transmet. Pour une séquence de valeurs s, l'entropie est donnée par :

$$H(s) = -\sum_i p(i)\log_2(p(i)), \tag{1}$$

où $p(i)$ est la probabilité d'apparition de la valeur i, calculée comme le rapport entre le nombre d'occurrences de cette valeur dans la séquence et la longueur totale de la séquence.

Par convention, si une valeur n'apparaît jamais dans la séquence, c'est-à-dire si $p(i) = 0$, le terme $p(i)\log_2(p(i))$ est remplacé par 0 dans la somme ci-dessus. Par exemple, une séquence contenant toujours la même valeur a une entropie nulle, car $p(c) = 1$ et donc $\log_2(p(c)) = 0$.

Un exemple plus intéressant est la séquence de sept éléments : 4, 5, 6, 5, 6, 5, 6. On a alors $p(4) = \frac{1}{7}$, $p(5) = p(6) = \frac{3}{7}$, et $p(i) = 0$ pour les autres octets, c'est-à-dire $i \in \{0, \ldots, 3\} \cup \{7, \ldots, 255\}$. L'entropie vaut alors :

$$\begin{aligned} H &= -\left(p(4)\log_2\left(p(4)\right) + p(5)\log_2\left(p(5)\right) + p(6)\log_2\left(p(6)\right)\right) \\ &= -\left(\frac{1}{7}\log_2\left(\frac{1}{7}\right) + \frac{3}{7}\log_2\left(\frac{3}{7}\right) + \frac{3}{7}\log_2\left(\frac{3}{7}\right)\right) \\ &\approx 1.449. \end{aligned} \tag{2}$$

Le but de cet exercice est de calculer l'entropie d'un fichier texte. Trois fichiers vous sont fournis : `ibiza.txt`, `faust.txt` et `wealth_of_nations.txt`. L'exemple ci-dessous illustre le calcul de l'entropie du fichier `ibiza.txt`.

Exemple :

La chaîne de caractères “IL FAIT BEAU A IBIZA”, de longueur 20, contient les caractères suivants {I, L, , F, A, T, B, E, U, Z} (le troisième caractère est l’espace), ce qui donne les fréquences de répétition décrites dans le tableau ci-dessous :

Caractère	Nombre d’occurences	Fréquence de répétition
I	4	$4/20 = 0.20$
L	1	$1/20 = 0.05$
	4	$4/20 = 0.20$
F	1	$1/20 = 0.05$
A	4	$4/20 = 0.20$
T	1	$1/20 = 0.05$
B	2	$2/20 = 0.10$
E	1	$1/20 = 0.05$
U	1	$1/20 = 0.05$
Z	1	$1/20 = 0.05$

En appliquant l’équation (1) à ces fréquences de répétition, on peut calculer l’entropie de la chaîne de caractères comme suit :

$$S = -\Big(0.20\log_2(0.20) + 0.05\log_2(0.05) + 0.20\log_2(0.20) + 0.05\log_2(0.05) + 0.20\log_2(0.20)$$
$$+0.05\log_2(0.05) + 0.10\log_2(0.10) + 0.05\log_2(0.05) + 0.05\log_2(0.05) + 0.05\log_2(0.05)\Big) \approx 3.022$$

1. [Difficulté: ***] **Calcul des fréquences de caractères**

   Implémentez la fonction `frequency_list()` qui :

   - Prend une chaîne de caractères comme entrée ;
   - Calcule la fréquence d’apparition de chaque caractère ;
   - Retourne un dictionnaire avec les caractères comme clés et leurs fréquences comme valeurs, ainsi que le nombre total de caractères.

   Ignorez les caractères de saut de ligne : `\n`.

   Pensez à utiliser la méthode `get()`. Celle-ci permet d’obtenir la valeur associée à une clé donnée. Si la clé n’existe pas, elle retourne une valeur par défaut (par défaut : `None`), évitant ainsi une erreur.

   Syntaxe : `dictionnaire.get(clé, valeur_par_défaut)`

   - `clé` : clé recherchée.
   - `valeur_par_défaut` (optionnelle) : valeur retournée si la clé est absente.

   Exemple :

```python
mon_dict = {"nom": "Alice", "age": 25}
print(mon_dict.get("nom"))       # Résultat : Alice
print(mon_dict.get("adresse"))   # Résultat : None
print(mon_dict.get("adresse", "Inconnue")) # Résultat : Inconnue
```

2. [Difficulté: **] **Calcul de l'entropie**

   Implémentez la fonction `entropy()` qui calcule l'entropie selon la formule suivante :

   $$H = -\sum_i p(i) \cdot \log_2(p(i))$$

   où $p(i)$ est la probabilité d'apparition de chaque caractère.

3. [Difficulté: **] **Analyse des fichiers texte**

   Appliquez vos fonctions pour calculer et afficher l'entropie des fichiers `faust.txt` et `wealth_of_nations.txt`. Ces fichiers peuvent être téléchargés depuis Moodle. Assurez-vous qu'ils se trouvent dans le même dossier que votre script Python, afin de pouvoir simplement spécifier leur nom lors de l'ouverture, sans avoir à indiquer le chemin complet.

   Implémentez une fonction `analyze_file()` qui prend comme argument une chaîne contenant le nom d'un fichier et affiche l'entropie correspondante.

   Pour lire le contenu d'un fichier dans une chaîne de caractères, utilisez le code suivant :

```
with open(filename, "r", encoding="utf-8") as file:
    text = file.read()
```

   La commande `open(filename, "r", encoding="utf-8")` ouvre le fichier indiqué par la variable `filename` en mode lecture (`"r"`) et avec l'encodage UTF-8: https://fr.wikipedia.org/wiki/UTF-8. Le fichier est ensuite associé à l'objet file, qui est utilisé pour interagir avec le contenu du fichier. La méthode `file.read()` lit l'intégralité du contenu du fichier et renvoie ce contenu sous forme d'une chaîne de caractères. Cette chaîne est ensuite stockée dans la variable `text`, que vous pouvez utiliser pour manipuler ou analyser le contenu du fichier.

   Dans cet exemple, `filename` est une chaîne contenant le nom du fichier avec son extension: `faust.txt` ou `wealth_of_nations.txt`. Pour que ce code fonctionne, les fichiers doivent être placés dans le même dossier que votre script. Sinon, la chaîne doit inclure le chemin complet vers le fichier ainsi que son nom.

   Les resultats entendus :

```
File: faust.txt
Entropy: 4.67
File: wealth_of_nations.txt
Entropy: 4.34
```