UNIVERSITÉ DE LORRAINE

Traitement de corpus

TD 6 : Analyse de texte et graphiques

(TD inspiré des supports de Gaël Lejeune)

1 Manipulation préalable

Dans cet exercice, nous allons utiliser la libraire `matplotlib` qui nous permettra de faire des graphiques. Il faut l'installer dans PyCharm en suivant les instructions suivantes :

1. Ouvrir File → Settings → Project.
2. Dans Python Interpreter, cliquer sur le "+" et taper : "matplotlib" dans la barre de recherche.
3. Cliquer ensuite sur Install Package.

2 Distribution des mots selon la taille en caractères

2.1 Lire le fichier

1. Lire le fichier `13846-0.txt`.
2. Récupérer la liste des mots dans une liste `liste_mots`.
   Note : Vous pouvez utiliser un simple `split()` sans préciser le séparateur. Par défaut, la fonction coupe selon les espaces.
3. Afficher le nombre de mots du fichier.

2.2 Compter le nombre de mots selon la longueur

Dans cette partie, nous allons compter pour chaque longueur le nombre de mots de cette longueur. Par exemple, si on a 100 mots de taille 5 (de 5 caractères) et 50 mots de taille 4, on aurait les entrées suivantes :

```
dic_longueurs = {5 : 100, 4 : 50}
# La clé est la longueur et la valeur est le nombre de mots de cette longueur
```

1. Initialiser un dictionnaire vide appelé `dic_longueurs`.
2. Parcourir la liste des mots obtenue en 2.1.2 avec une boucle.
3. Stocker la longueur du mot dans une variable appelée `longueur`.
4. Vérifier que la longueur n'est pas déjà présente dans le dictionnaire avec une condition `if ... not in ...`.
   (a) Si elle n'est pas présente, on ajoute la longueur dans le dictionnaire et on met sa valeur à 1.
   (b) Si elle est déjà présente, on incrémente la valeur existante de 1.

   Pour vous aider, voici un aperçu du code que vous obtenez. Complétez les parties "...".

```
dic_longueurs = {}                          #1

for ... in liste_mots:                      #2
    longueur = ...                          #3
    if ... not in dic_longueurs:            #4
        dic_longueur[longueur] = ...        #4.a
    else:
        ...                                 #4.b
```

5. Afficher le dictionnaire obtenu. Vous devez obtenir le résultat suivant :

```
{4: 15694, 7: 9158, 9: 6051, 5: 13169, 2: 26679, 8: 6888, 3: 19000, 6: 9970, 12: 1476,
    11: 2294, 17: 18, 10: 3986, 13: 861, 1: 2798, 14: 501, 15: 160, 16: 67, 18: 4, 20: 2,
     38: 1, 24: 2, 22: 2, 40: 1, 31: 1, 21: 1, 30: 1, 19: 1, 25: 1}
```

2.3 Afficher les résultats proprement

Cette étape permet de préparer les données que nous utiliserons pour le graphique.

1. Utiliser une boucle `for ... in range(1, 31)` pour afficher les longueurs et leurs valeurs dans l'ordre croissant de la manière suivante :

```
1 : 2798
2 : 26679
3 : 19000
4 : 15694
5 : 13169
...
```

Attention : Avec seulement la boucle `for ... in range(1, 31)`, vous remarquerez que le code affiche une erreur car certaines longueurs ne sont pas présentes dans le dictionnaire. Ajouter une condition `if` pour pallier cette erreur. Voici le code que vous devez compléter :

```
for ... in range (1, 31):       #31 car la longueur maximale d'un mot est de 30
    if ... in ... :
        nb_occurrences = ...
        print (f" ... : {nb_occurrences}")
```

2. Avec le code précédent, les longueurs qui ne sont pas dans le dictionnaire ne sont pas affichées. Ajouter une clause `else` pour afficher toutes les longueurs de 1 à 30 en donnant la valeur 0 aux longueurs qui ne sont pas dans le dictionnaire.

   Exemple : La longueur 23 n'est pas dans le dictionnaire.

```
...
22 : 2
23 : 0
24 : 2
...
```

2.4 Représenter les résultats dans un graphique

1. Importer la bibliothèque `matplotlib` au début de votre script en ajoutant cette ligne :

```
import matplotlib.pyplot as pyplot
```

Note : `as pyplot` est un alias qui permet d'utiliser la librairie avec un nom plus court. Au lieu d'écrire `matplotlib.pyplot` à chaque utilisation, nous utiliserons simplement `pyplot`.

Pour créer notre graphique, il faut stocker les valeurs des longueurs dans une liste.

2. Initialiser une liste vide appelée `liste_valeurs`.
3. Adapter le code de la partie 2.3 pour ajouter la valeur de chaque longueur à la liste.

```
for ... in range (1, 31):
    if ... in ... :
        nb_occurrences = ...
        liste_valeurs.append(...)
    else:
        nb_occurrences = ...
        liste_valeurs.append(...)
```

4. Ajouter les deux lignes suivantes pour dessiner et afficher le graphique :

```
pyplot.plot(liste_valeurs)
pyplot.show()
```

Vous devez obtenir un graphique comme celui en Figure 1.

En Figure 2, on a ajouté des légendes et un titre au graphique. Il est aussi possible de changer le style de la courbe : sa couleur, son type de tracé etc. grâce aux lignes suivantes :

```
pyplot.plot(liste_valeurs, "r:o")                    # "r" pour red et "o" pour les points
pyplot.xlabel("Longueur des mots en caractères")     # titre de l'axe des abscisses
pyplot.ylabel("Nombre d'occurrences")                # titre de l'axe des ordonnées
pyplot.title("Distribution des mots selon la taille en caractères") # titre du graphique
pyplot.show()
```

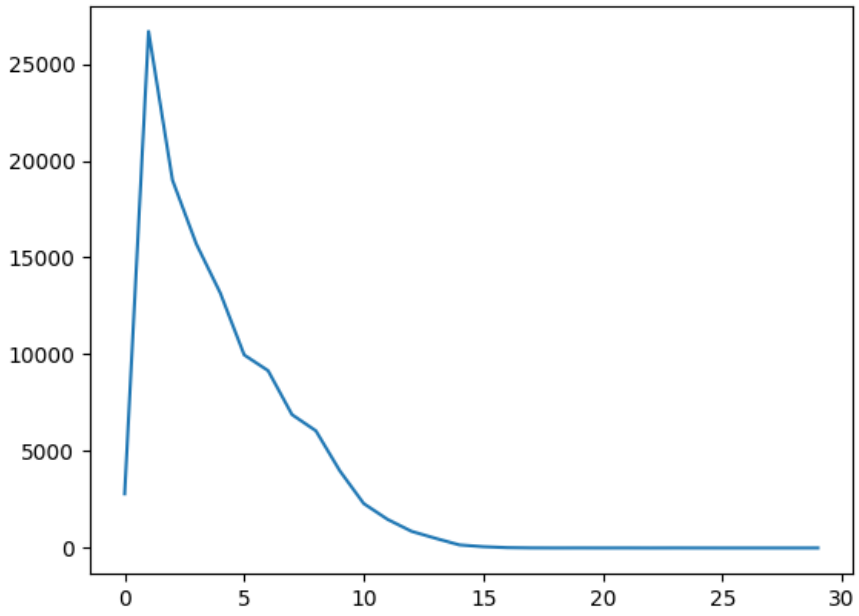

FIGURE 1 – Graphique simple par défaut.

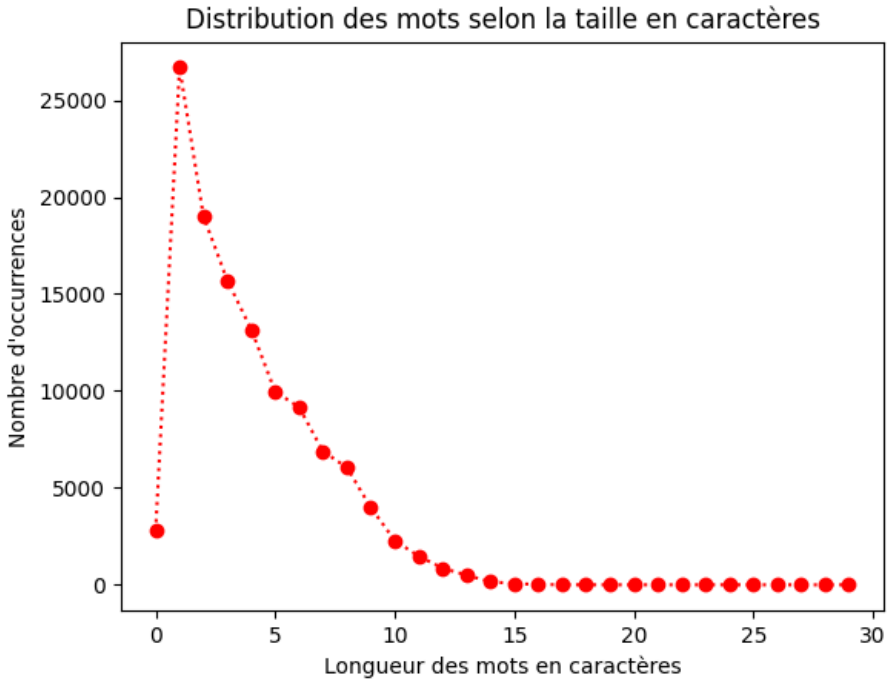

FIGURE 2 – Graphique amélioré.