Universidad Rey Juan Carlos

TEMA 3 – HOJA DE EJERCICIOS II

Se plantean ejercicios relacionados con matrices de coocurrencia. En PLN es crucial poder entender las relaciones entre las palabras, por ello, las matrices de coocurrencia resultan una herramienta fundamental para capturar esas relaciones.

Una matriz de coocurrencia es una representación matemática que captura la frecuencia en la que pares de palabras aparecen juntas en un contexto dado (puede ser una ventana de *n* palabras, una frase, un párrafo o un documento).

Ejercicio 1. Crear matriz de coocurrencias.

Se quiere crear una matriz de coocurrencias para el siguiente párrafo:

"Mysterious tunnels sketched by Leonardo da Vinci in the late 1400s may have been found at the Castle. Secret tunnels at the Sforza Castle."

Los pasos que hay que seguir para hacer esto podrían ser los siguientes:

1) Preprocesar el texto eliminando stopwords, convirtiendo las palabras a minúsculas, tokenizando y eliminando los tokens no alfanuméricos.
2) Generar el vocabulario (palabras únicas en el texto, sin repeticiones).
3) Decidir el tamaño de ventana del contexto y buscar todos los pares de palabras que coocurren dentro de ese tamaño de ventana, contando las veces que coaparecen.
   Para este ejercicio poner un tamaño de ventana para el contexto = 2.
4) Crear la matriz de coocurrencias utilizando las frecuencias calculadas antes.
5) Visualizar la matriz utilizando DataFrame.

Ejercicio 2. Similitud coseno entre pares de palabras.

A partir de la matriz de coocurrencias del ejercicio anterior, se pide obtener la similitud entre diferentes pares de palabras, utilizando la similitud coseno.

Obtener la similitud coseno para los pares:

- "da" – "vinci"
- "may" – "tunnels"
- "tunnels" – "castle"
- "secret" – "tunnels"

Nota: se puede utilizar la librería sklearn para calcular la similitud coseno (`from sklearn.metrics.pairwise import cosine_similarity`). Para indicar las palabras se le pueden decir los índices de la posición que ocupa la palabra correspondiente en el vocabulario.

Ejercicio 3. Palabras más similares a una dada.

Utilizando la matriz de coocurrencias y la similitud coseno, obtener las 5 palabras más similares a las siguientes palabras:

- "leonardo"
- "da"

Analizar la salida. Es importante tener en cuenta cuál era el tamaño de la ventana del contexto.

Si se cambia el tamaño de ventana, ¿cambian las palabras más similares?

Ejercicio 4. Palabras más similares a una dada utilizando como pesado la función PPMI.

Se proporciona el siguiente código, que convierte una matriz de coocurrencias normal, basada en frecuencia de aparición, en una matriz con pesado utilizando la función PPMI:

```
import numpy as np
def create_ppmi_matrix(cooccurrence_matrix):
    # Calcular la suma total de la matriz de coocurrencias
    total_sum = np.sum(cooccurrence_matrix)

    # Calcular la suma de cada fila y cada columna
    row_sums = np.sum(cooccurrence_matrix, axis=1)
    col_sums = np.sum(cooccurrence_matrix, axis=0)

    # Inicializar la matriz PPMI con ceros
    ppmi_matrix = np.zeros_like(cooccurrence_matrix, dtype=float)

    # Calcular los valores PPMI para cada celda en la matriz
    for i in range(cooccurrence_matrix.shape[0]):
```

```
        for j in range(cooccurrence_matrix.shape[1]):
            p_ij = cooccurrence_matrix[i, j] / total_sum
            p_i = row_sums[i] / total_sum
            p_j = col_sums[j] / total_sum

            if p_ij > 0:
                ppmi_value = max(0, np.log2(p_ij / (p_i * p_j)))
                ppmi_matrix[i, j] = ppmi_value
    return ppmi_matrix
```

A partir de la matriz anterior, se pide lo mismo que en el ejercicio 3. ¿Cambian las palabras más similares?