RECUPERACIÓN DE LA INFORMACIÓN

Adrián Mínguez Graña
UO272447

Contenido

RECUPERACIÓN DE INFORMACIÓN

La realización de los ejercicios ha sido utilizando la tecnología Python.

La entrega contiene esta documentación y todo el código y archivos utilizados divididos en carpetas por cada ejercicio.

EJERCICIO 1

El trabajo realizado en este ejercicio ha consistido en:

- Obtención de la información a través del api: https://api.pushshift.io/reddit/search/submission/?subreddit=
- Obtención de la lista de norvig facilitada en el enunciado.
- Almacenar dichos datos en diversos diccionarios para calcular las apariciones de cada palabra y el total del conjunto.
- Transformar el método rootLogLikelihoodRatio de PHP a Python.
- Aplicar el método y ordenar el resultado.

El resultado ha sido satisfactorio. Si se analizan las palabras con las puntuaciones más elevadas, todas se pueden relacionar o entran dentro del contexto de depresión (my, me, feel, sad…). Además, dentro de las palabras con menos puntuación, no hay ninguna que pudiera parecer errónea, es decir, que encajara más en las puntuaciones positivas.

No se han encontrado grandes dificultades, probablemente lo peor ha sido el manejo de tanta información, lo que provoca que el programa va un poco lento.

Lista de términos

Se encuentra en la documentación del ejercicio 1, en el fichero lista.txt.

EJERCICIO 3

Como no se ha realizado el ejercicio 2, solo se han utilizado palabras obtenidas al realizar el ejercicio 1. De las 100 primeras con puntuación más alta se han cogido todas saltando las que tenían como semilla depress.

Se han cogido 5000 muestras del subreddit offmychest.

La aplicación ha consistido en almacenar en un diccionario los textos de offmychest con la suma total de puntos obtenidos en base a los 100 términos seleccionados.

Se ha obtenido como resultado:

- Verdaderos positivos: 30
- Falsos verdaderos: 70
- Verdaderos negativos: 99
- Falsos negativos: 1
- Total textos positivos: 503

El rendimiento de la aplicación ha sido muy bajo para obtener los resultados positivos. Había 503 textos que incluían palabras con semilla depress, de los cuales solo ha puntuado dentro de los 100 primeros a 30.

Respecto a los negativos, aunque solo ha errado en uno, es un resultado algo negativo, ya que solo había un 10% de textos con semilla depress y aun así ha incorporado uno dentro de este conjunto.

Listas de términos

Se encuentra en la documentación del ejercicio 3:

- Puntuación más alta: muestraPositivo.txt.
- Puntuación más baja: muestraNegativo.txt

EJERCICIO 4

Dado que no dispongo de conocimientos de aprendizaje automático no he conseguido completar correctamente este ejercicio.

Para seleccionar la muestra aleatoria se ha utilizado el api empleado en el ejercicio 1 del Reddit de ProRevenge.

Para la selección de los elementos pertenecientes a la clase Depress se han cogido los 30 verdaderos positivos.

Como biblioteca de aprendizaje se ha utilizado Scikit-learn.

No se han utilizado datasets, se han utilizado dos listas, la primera con los textos y la segunda con las clases (Depress, NoDepress). Con esos dos arrays se ha realizado el entrenamiento utilizando las funciones propias de la librería y después se han pasado los valores de los diferentes Reddit para su predicción.

El resultado ha sido erróneo, ya que en todas las predicciones muestra NoDepress, probablemente porque este mal hecho y es la clase que más veces aparece en el entrenamiento (100 por 30 de Depress).

EJERCICIO 4.1

Como no he conseguido que funcione bien el ejercicio anterior, en este los resultados iban a ser los mismos, NoDepress. Se probó a cambiar las veces que aparecían las clases y se obtenían resultados diferentes. Si se igualaban las ocurrencias de Depress y NoDepress salían resultados dispares, como se puede ver en la imagen siguiente.

```
C:\Users\Adrian\Documents\GitHub\Master\Master_WebSemantica\Recuperacion de informacion\Ejercicio 4>py main.py
['Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress' 'Depress'
 'Depress' 'Depress' 'Depress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'Depress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'Depress'
 'NoDepress' 'NoDepress' 'Depress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress' 'NoDepress'
 'NoDepress']
```