Prácticas de SAR

Práctica 2: Cuenta Palabras

Cuenta Palabras

Descripción del problema

Para hacer estudios sobre la autoría de unos documentos, se desea obtener estadísticas del estilo literario del autor (centrándonos en el uso del vocabulario).

¿Qué debo hacer?

Escribe un programa en python que analice ficheros de texto, calcule estadísticas sobre cada fichero y las escriba en disco.

- Recibirá uno o más nombres de fichero obligatoriamente.
- Por cada fichero de texto generará otro con las estadísticas.
- Aceptará los siguiente argumentos opcionales:
  - `-h`, `--help`: muestra el mensaje de ayuda.
  - `-l`, `--lower`: pasa todo el contenido a minúsculas antes de calcular las estadísticas.
  - `-s` fichero (también `--stop` fichero): para especificar un fichero con las *stopwords* que se deben eliminar.
  - `-b` (también `--bigram`) realiza también el cálculo a nivel de bigramas.
  - `-f`, `--full`: se deben mostrar las estadísticas completas. En caso contrarío solo se muestras 20 entradas de cada categoría.
- Para realizar el análisis se eliminarán todos los símbolos no alfanuméricos.

¿Qué debo hacer?

Salida con -h

```
prompt> python SAR_p2_cuenta_palabras.py --help
usage: SAR_p2_cuenta_palabras.py [-h] [-s STOPWORDS] [-l] [-b] [-f]
                                 file [file ...]

Compute some statistics from text files.

positional arguments:
  file                  text file.

optional arguments:
  -h, --help            show this help message and exit
  -s STOPWORDS, --stop STOPWORDS
                        filename with the stopwords.
  -l, --lower           lowercase all words before computing stats.
  -b, --bigram          compute bigram stats.
  -f, --full            show full stats.
```

Análisis por palabras

¿Qué debo hacer?

Escribe un programa en python que analice ficheros de texto, calcule estadísticas sobre cada fichero y las escriba en disco.

El programa en python mostrará la siguiente información:

- Nº líneas.
- Nº palabras.
- Nº de palabras, excluyendo stopwords (en el caso de elegir eliminarlas).
- Vocabulario: nº palabras distintas que aparecen en el texto.
- Símbolos: nº letras que aparecen en el texto.
- Símbolos distintos: nº letras distintas que aparecen en el texto.
- Nº veces que aparece cada palabra: ordenado alfabéticamente y por el nº veces que aparecen.
- Nº veces que aparece cada letra: ordenado alfabéticamente y por el nº veces que aparecen.
- Los prefijos de 2, 3 y 4 letras.
- Los sufijos de 2, 3 y 4 letras.

Análisis por palabras, ¿Qué debo hacer?

Ejemplo de funcionamiento

```
prompt> python SAR_p2_cuenta_palabras.py spam.txt --lower
prompt> cat spam_l_stats.txt
Lines: 11
Number words (including stopwords): 77
Vocabulary size: 22
Number of symbols: 324
Number of different symbols: 23
Words (alphabetical order):
    a: 2
    and: 12
    aux: 1
...
    spam: 27
    thermidor: 1
    top: 1
Words (by frequency):
    spam: 27
    and: 12
    egg: 9
...
    thermidor: 1
```

Análisis por palabras, ¿Qué debo hacer?

Ejemplo de funcionamiento (continuación)

```
Symbols (alphabetical order):
    a: 63
    b: 10
    c: 8
    d: 17
...
    s: 40
    t: 9
    u: 7
    v: 1
Symbols (by frequency):
    a: 63
    s: 40
    m: 29
    p: 29
...
    f: 3
    l: 2
    w: 2
    y: 2
```

Análisis por palabras, ¿Qué debo hacer?

Ejemplo de funcionamiento (continuación)

```
Prefixes (by frequency):
    sp-: 27
    spa-: 27
    an-: 12
    eg-: 9
...
    bea-: 1
    bean-: 1
    br-: 1
    bra-: 1
Suffixes (by frequency):
    -am: 27
    -pam: 27
    -nd: 12
    -gg: 9
...
    -ate: 1
    -auce: 1
    -ay: 1
    -ce: 1
```

Análisis por bigramas

Análisis por bigramas, ¿Qué debo hacer?

Además del análisis por palabras, se debe hacer:

1. Realizar un análisis de los pares de palabras consecutivas (bigramas) que aparecen en las frases.
   - se mostrarán los resultados ordenados por orden alfabético y por frecuencia.
   - se considerará cada línea del fichero como una frase.
   - se deberá añadir un símbolo (`'$'`) como primera y última palabra de cada frase.
2. Realizar un análisis de los pares de letras consecutivas que aparecen en cada palabra.
   - se mostrarán los resultados ordenados por orden alfabético y por frecuencia.

El análisis adicional de bigramas se activará mediante el argumento `-b`.

Nota

Si se ha seleccionado la opción de eliminación de stopwords, no se deben considerar bigramas que contengan stopwords.

Análisis por bigramas, ¿Qué debo hacer?

Ejemplo de funcionamiento

```
prompt> python SAR_p2_cuenta_palabras_2.py spam.txt --lower --bigram
prompt> cat spam_lb_stats.txt
Lines: 11
...
Words (alphabetical order):
...
Words (by frequency):
...
Symbols (alphabetical order):
...
Symbols (by frequency):
...
Word pairs (alphabetical order):
    $ egg: 5
    $ lobster: 1
    $ spam: 5
    a fried: 1
...
    egg and: 3
    egg bacon: 2
    egg on: 1
    egg sausage: 2
```

Análisis por bigramas, ¿Qué debo hacer?

Ejemplo de funcionamiento (continuación)

```
Word pairs (by frequency):
    spam spam: 11
    and spam: 9
    spam $: 9
...
    and a: 1
    aux crevettes: 1
Symbol pairs (alphabetical order):
    ac: 6
    ag: 4
...
    ea: 1
    ed: 3
Symbol pairs (by frequency):
    pa: 28
    am: 27
...
    er: 2
Prefixes (by frequency):
...
Suffixes (by frequency):
...
```

Nombre de los ficheros de estadísticas

Nombre de los ficheros de estadísticas

Por cada fichero que analice, el programa debe generar un fichero con las estadísticas. El nombre del fichero de estadísticas debe cumplir las siguientes normas:

- Debe tener la misma extensión que el fichero original.
- Al nombre del fichero original se le debe añadir (separado por `'_'`):
  - un código con las opciones elegidas, **l,s,b,f** en ese orden. **l**:*lower*, **s**:*stopwords*, **b**:*bigram*, **f**:*full*.
  - `'stats'` al final del nombre original.

Ejemplo de nombres de fichero:

- python SAR_p2_cuenta_palabras.py **spam.txt -l -b**
  Generaría el fichero ***spam_lb_stats.txt***
- python SAR_p2_cuenta_palabras.py **tirant -f -s** english.txt **-l**
  Generaría el fichero ***tirant_lsf_stats***
- python SAR_p2_cuenta_palabras.py **spam.txt**
  Generaría el fichero ***spam_stats.txt***

Plantilla proporcionada

Plantilla proporcionada

```
if __name__ == "__main__":
    parser = argparse.ArgumentParser(description='Compute some statistics
     from text files.')
    parser.add_argument('file', metavar='file', type=str, nargs='+',
                        help='text file.')
    parser.add_argument('-l', '--lower', dest='lower',
                        action='store_true', default=False,
                        help='lowercase all words before computing stats.')
    parser.add_argument('-s', '--stop', dest='stopwords', action='store',
                        help='filename with the stopwords.')
    parser.add_argument('-b', '--bigram', dest='bigram',
                        action='store_true', default=False,
                        help='compute bigram stats.')
    parser.add_argument('-f', '--full', dest='full',
                        action='store_true', default=False,
                        help='show full stats.')
    args = parser.parse_args()
    wc = WordCounter()
    wc.compute_files(args.file,
                     lower=args.lower,  stopwordsfile=args.stopwords,
                     bigrams=args.bigram,  full=args.full)
```

Plantilla proporcionada

```
class WordCounter:

    def __init__(self):
        """
           Constructor de la clase WordCounter
        """
        self.clean_re = re.compile('\W+')

    def write_stats(self, filename:str, stats:dict, use_stopwords:bool, full
     :bool):
        """
        Este método escribe en fichero las estadísticas de un texto

        :param
            filename: el nombre del fichero destino.
            stats: las estadísticas del texto.
            use_stopwords: booleano, si se han utilizado stopwords
            full: boolean, si se deben mostrar las stats completas
        """

        with open(filename, 'w', encoding='utf-8', newline='\n') as fh:
            ## completar
            pass
```

Plantilla proporcionada

```
def file_stats(self, fullfilename:str, lower:bool, stopwordsfile:
 Optional[str], bigrams:bool, full:bool):
    """
    Este método calcula las estadísticas de un fichero de texto
    :param
        fullfilename: el nombre del fichero, puede incluir ruta.
        lower: booleano, se debe pasar todo a minúsculas?
        stopwordsfile: nombre del fichero con las stopwords o None si no
  se aplican
        bigram: booleano, se deben calcular bigramas?
        full: booleano, se deben montrar la estadísticas completas?
    """
    stopwords = [] if stopwordsfile is None else open(stopwordsfile,
 encoding='utf-8').read().split()
    # variables for results
    sts = { 'nwords': 0, 'nlines': 0, 'word': {},
     'symbol': {}, 'prefix': {}, 'suffix': {} }
    if bigrams:
        sts['biword'] = {}
        sts['bisymbol'] = {}
    # COMPLETAR
    # AYUDA: line = self.clean_re.sub(' ', line)
    filename, ext0 = os.path.splitext(fullfilename)
    new_filename = "" # cambiar
    self.write_stats(new_filename, sts, stopwordsfile is not None, full)
```

Plantilla proporcionada

```
def compute_files(self, filenames:str, **args):
    """
    Este método calcula las estadísticas de una lista de ficheros de
 texto

    :param
        filenames: lista con los nombre de los ficheros.
        args: argumentos que se pasan a "file_stats".

    :return: None
    """

    for filename in filenames:
        self.file_stats(filename, **args)
```