Taller_PosgradoUABC_s1

September 26, 2024

1 Procesamiento de corpus y visualización de características

2 Instalación de librerías

En esta sección se detalla qué librerías se deben instalar para hacer el procesamiento de un corpus:

- **PANDAS**: Análisis de datos.
- **NUMPY**: Creación de vectores y matrices. Procesamiento matemático.
- **SEABORN**: Librería para visualización de datos.
- **MATPLOTLIB** Librería para visualización de datos.
- **WORDCLOUD** Librería para crear nubes de palabras.
- **PILLOW**: Librería para edición de imágenes.

Nota: en este entorno se pueden insertar comentarios, los cuales no afectan la ejecución del código. Para insertarlos, es necesario introducirlos con la tecla #.

```
!pip install LIBRARY
# este comando se usa (!pip) para instalar librerías de Python
```

[ ]:
```
!pip install pandas
```

[ ]:
```
!pip install numpy
```

[ ]:
```
!pip install seaborn
```

[ ]:
```
!pip install matplotlib
```

[ ]:
```
!pip install wordcloud
```

[ ]:
```
!pip install pillow
```

3 Importar librerías

Las librerías que se acaban de instalar ya están dispobibles para hacer uso de ellas, solo que antes de emplearlas, hay que indicar en el código que se utilizarán. Para ello, se usa el comando:

- **import** LIBRARY.

```
[ ]: import pandas as pd #es común que se usen abreviaturas para hacer más ligero el␣
     ↪código. En este caso, se especifica con **as** ABREVIATURA
     import numpy as np
     import seaborn as sns
     import matplotlib.pyplot as plt
     import os
```

El proceso para importar puede especificarse para no cargar toda la librería, sino solo una parte. Por ejemplo, asumimos que la librería es *Office*, pero de ella solo queremos usar un programa como *Word* y no el resto de programas y plugins de Office. La forma de hacerlo es la siguiente:

- **from** LIBRARY **import** PACKAGE
- **from** OFFICE **import** WORD

```
[ ]: from wordcloud import WordCloud
     from PIL import Image
```

4 Cargar corpus

Es importante saber la ruta del archivo que tiene el corpus.

Lo recomendable es que se suba a una ruta del entorno virtual para acelerar el procesamiento.

Para este taller usaremos algunos de los corpus que están disponibles en mi página de Github.

```
[ ]: ls #comando para listar la información que está en una carpeta
```

4.1 Indexar el corpus en un DataFrame

Para poder explorar las características del corpus, primero hay que indexarlo en una variable. Esta variable en PANDAS tiene la estructura de un dataframe, en el cual la información del corpus está organizada en filas y columnas.

```
[ ]: corpus = pd.read_csv('hateSpeech.csv',  encoding='utf-8')
```

5 Exploración del corpus

Se usarán algunos comandos para tener información sobre el corpus. La mayoría de estos comandos son funciones que se activan al crear variables.

```
[ ]: corpus #primeras y últimas líneas del corpus
```

```
[ ]: corpus.head(10)
```

```
[ ]: corpus.tail(8)
```

```
[ ]: corpus.Class.value_counts()
```

[ ]:
```
corpus.Language.value_counts()
```

[ ]:
```
corpus.info()
```

[ ]:
```
corpus.dtypes
```

[ ]:
```
corpus.Text.size
```

[ ]:
```
corpus.Text.unique()
```

[ ]:
```
corpus.sample(30)
```

5.1 Referencias

Si quieres explorar más **atributos** y **métodos**, puedes consultar la documentación de PANDAS en este link.

6 Graficación de los datos

[ ]:
```
#Se graficará la información de Class con base en su frecuencia

labels = ['Violencia', 'Acoso'] #etiquetas en eje X
counts = [520, 257] #valores en eje Y
ypos = np.arange(len(labels)) #vectorización
ypos


plt.xticks(ypos, labels)
plt.xlabel("Categorías")
plt.ylabel("Frecuencia")
plt.title("Distribución por clases")
plt.bar(ypos, counts)
```

[ ]:
```
corpus.Language.value_counts()
```

[ ]:
```
#Se graficará la información de Language con base en su frecuencia

labels = ['E', 'I']
counts = [551, 225]
ypos = np.arange(len(labels))
ypos


plt.xticks(ypos, labels)
plt.xlabel("Tipo de lenguaje")
plt.ylabel("Frecuencia")
```

```
plt.title("Distribución por tipo de lenguaje")
plt.bar(ypos, counts)
plt.savefig('language.png')          #con este comando guardamos el gráfico en␣
 ↪el path
```

7 Visualización por nube de palabras

[ ]:
```
#Preparación de los datos por categoría. Se dividen con base en la categoría.

cloud1 = corpus[corpus.Class=='Violencia']
cloud2 = corpus[corpus.Class=='Acoso']

#Generación de listas por categoría. Se toman los datos de la columna Text.
lists_cloud1 = cloud1['Text'].tolist()
lists_cloud2 = cloud2['Text'].tolist()

#Genaración de elementos en listas
words_cloud1 = ("").join(lists_cloud1)


words_cloud2 = ("").join(lists_cloud2)
#filtered_career = filtered_career.lower()
```

[ ]:
```
#Se crea la plantilla de la nube de palabras para las palabras de la categoría␣
 ↪Violencia.
template = np.array(Image.open("nube.jpg"))

wordcloud_1 = WordCloud(max_font_size = 50,
                        margin=3,
                        stopwords=['que', 'la', 'el', 'los','las', 'con', 'y',␣
 ↪'pero', 'de', 'en', 'del',
                                   'con', 'a', 'sin', 'le', 'por', 'para',␣
 ↪'su', 'me', 'le', 'al', 'porque'
                                   'te', 'o', 'una', 'eso', 'ni', 'ya'],
                        background_color = "Black", mask = template,
                        colormap="Reds").generate(words_cloud1)
```

[ ]:
```
#Graficación de la nube de palabras para categoría Violencia

plt.figure(figsize=[8,8])
plt.imshow(wordcloud_1,interpolation='bicubic')
plt.axis("off")
plt.margins(x=1, y=1)

plt.savefig("wordcloud_Violencia.png", bbox_inches='tight') #comando para␣
 ↪guardar la nube.
```

```
plt.show()
```

[ ]:
```
#Se crea la plantilla de la nube de palabras para las palabras de la categoría␣
↪Acoso.

template = np.array(Image.open("nube.jpg"))

wordcloud_2 = WordCloud(max_font_size = 100,
                        margin=3,
                        stopwords=['que', 'la', 'el', 'los','las', 'con', 'y',␣
↪'pero', 'de', 'en', 'del',
                                   'con', 'a', 'sin', 'le', 'por', 'para',␣
↪'su', 'me', 'le', 'al', 'porque'
                                   'te', 'o', 'una', 'eso', 'ni', 'ya'],
                        background_color = "Grey",
                        mask = template,
                        colormap="Greens").generate(words_cloud2)
```

[ ]:
```
plt.figure(figsize=[8,8])
plt.imshow(wordcloud_2,interpolation='bilinear')
plt.axis("off")
plt.margins(x=1, y=1)

plt.savefig("wordcloud_Acoso.png", bbox_inches='tight')
plt.show()
```

8 Prueba con un corpus diferente

Nota

Para evitar ambigüedad y duplicidad de información, es conveniente que en este ejercicio definan variables diferentes. Por ejemplo, en lugar de que el dataframe se llame *corpus*, ahora le pondremos *datos*.

[ ]:
```
datos = pd.read_csv('data.csv', encoding='utf-8')
```

[ ]:
```
datos.head(10)
```

[ ]:
```
datos.info()
```

[ ]:
```
datos.UserLanguage.value_counts()
```

[ ]:
```
datos.worries_covid_group.value_counts()
```

[ ]:
```
datos.age.value_counts()
```

[ ]:
```
labels = ['18-29', '30-39', '50-59', '40-49', '60-69'] #Etiquetas en eje X
counts = [37, 17, 11, 7, 2] #Valores en eje Y
ypos = np.arange(len(labels)) #converting text labels to numberic value, 0 and 1
ypos


plt.xticks(ypos, labels)
plt.xlabel("Rango etario")
plt.ylabel("Frecuencia")
plt.title("Frecuencia por grupo etario")
plt.bar(ypos, counts)
plt.savefig('etario.png')
```

[ ]:
```
#Preparación de los datos por categoría. Se dividen con base en la categoría.

cloud1 = datos[datos.employment=='Self-employed']
#cloud2 = corpus[corpus.Class=='Acoso']

#Generación de listas por categoría. Se toman los datos de la columna Text.
lists_cloud1 = cloud1['association_covid'].tolist()
#lists_cloud2 = cloud2['Text'].tolist()

#Genaración de elementos en listas
words_cloud1 = ("").join(lists_cloud1)


#words_cloud2 = ("").join(lists_cloud2)

#Se crea la plantilla de la nube de palabras para las palabras de la categoría␣
 ↪Violencia.
template = np.array(Image.open("nube.jpg"))

wordcloud_1 = WordCloud(max_font_size = 50,
                        margin=3,
                        stopwords=['que', 'la', 'el', 'los','las', 'con', 'y',␣
 ↪'pero', 'de', 'en', 'del',
                                   'con', 'a', 'sin', 'le', 'por', 'para',␣
 ↪'su', 'me', 'le', 'al', 'porque'
                                   'te', 'o', 'una', 'eso', 'ni', 'ya'],
                        background_color = "Black", mask = template,
                        colormap="Reds").generate(words_cloud1)

#Graficación de la nube de palabras para categoría Violencia

plt.figure(figsize=[8,8])
plt.imshow(wordcloud_1,interpolation='bicubic')
plt.axis("off")
```

```
plt.margins(x=1, y=1)

plt.savefig("wordcloud_Violencia.png", bbox_inches='tight') #comando para␣
 ↪guardar la nube.
plt.show()
```