Taller_PosgradoUABC_s2

September 26, 2024

1 Descripción estadística del corpus

2 Instalación de librerías

Se usarán las siguientes librerías de la sesión pasada:

- **PANDAS**: Análisis de datos.
- **NUMPY**: Creación de vectores y matrices. Procesamiento matemático.
- **SEABORN**: Librería para visualización de datos.
- **MATPLOTLIB** Librería para visualización de datos.

[ ]:
```
!pip install pandas
!pip install numpy
!pip install seaborn
!pip install matplotlib
```

3 Importar librerías

Comando de la sesión anterior.

- **import** LIBRARY.

Recordar que, si no es necesaria toda la librería, se puede importar solo una función de ella. Esto hace que el coste computacional y temporal sean más eficientes. La forma de hacerlo es la siguiente:

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

[ ]:
```
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
```

4 Cargar e instanciar corpus

[ ]:
```
corpus = pd.read_csv('CBspectral-measures.csv',  encoding='utf-8')
#corpus = pd.read_csv('Cuestionario.csv',  encoding='utf-8')
```

5 Exploración y visualización del corpus

Se usarán los comandos de la sesión anterior para tener información sobre el corpus.

Recordar que la mayoría de estos comandos son funciones que se activan al crear variables y que ello hace que se activen herramientas específicas que leen la información de las variables, por ejemplo, .head, .info, .CABECERA.value_counts, etc.

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

[ ]:
```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['asp'].value_counts()
    for x_label, grp in corpus.groupby('coda')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('coda')
_ = plt.ylabel('asp')
```

[ ]:
```
from matplotlib import pyplot as plt
corpus['sF0'].plot(kind='hist', bins=20, title='sF0')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

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

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

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

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

[ ]:
```
corpus
```

[ ]:
```
corpus.rmul(2)    #método para multiplicar valores
```

5.1 Otras funciones para visualizar los datos: histogramas

[ ]:
```
corpus.hist() #genera histogramas a partir de los datos del corpus
```

[ ]:
```
corpus.hist('duration') #para hacer el histograma de una columna específica
plt.savefig('duration.png') #guardar el histograma en FILE.png
```

[ ]:
```
corpus.hist('duration', 'window') #comparación de datos
plt.savefig('comparisson.pdf')
```

[ ]:
```
#ejemplo para guardar una muestra aleatoria del corpus
corpus_150 = corpus.sample(150) #guarda 150 líneas en la variable

#si se quiere guardar los datos que están en la variable, se puede usar el␣
 ↪siguiente código
file1 = open ('corpus_150.txt', 'w')
file1.write(str((corpus_150)))
file1.close()
```

[ ]:
```
corpus_150
```

5.2 Estadística general con la función .describe()

[ ]:
```
corpus.shape #columnas por filas
```

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

[ ]:
```
file2 = open ('corpus_describe.txt', 'w')
file2.write(str((corpus.describe())))
file2.close()
```

6 Estadística descriptiva

Métricas descriptivas sobre los datos:

- Media
- Mediana
- Moda
- Desviación estándar
- Varianza

[ ]:
```
corpus.sF0.mean()
```

[ ]:
```
corpus.sF0.median()
```

[ ]:
```
corpus.sF0.mode()
```

```
[ ]: corpus.sF0.std()
```

```
[ ]: corpus.sF0.var()
```

```
[ ]: corpus.sF0.min()
```

```
[ ]:  corpus.sF0.max()
```

```
[ ]: corpus.sF0.quantile()
```

```
[ ]: corpus.sF0.describe()
```

```
[ ]: corpus.groupby('duration').agg({'Energy': ['mean', 'std', 'var']}).reset_index()
```

```
[ ]: corpus.groupby('duration').agg({'sF0': ['mean']}).reset_index()
```

```
[ ]: distN_Energy = sns.distplot(corpus.Energy)
```

```
[ ]: distN_tone = sns.distplot(corpus.tone)
```

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

7 Anotación POS

En esta sección veremos cómo se puede anotar el corpus con etiquetas de categoría gramatical de forma automática e iterativa. Para ello, usaremos el software TreeTagger.

```
[ ]: lexicon = corpus['manner'].astype(str)
file2 = open ('lexicon.txt', 'w')
file2.write(str((lexicon)))
file2.close()
```

```
[ ]: !sh install-tagger.sh
```

```
[ ]: !echo 'this is a test' | cmd/tree-tagger-english-bnc
```

```
[ ]: !cat lexicon.txt | cmd/tree-tagger-english-bnc > lexicon_annotated.txt
```

8 Refuerzo

Replica los mismos procesos pero con los datos de un nuevo dataset. Usa el archivo *experiencia.xlsx* para probar con la lectura de archivos de Excel.

```
[ ]: datos = pd.read_excel('experiencia.xlsx')
```

```
[ ]: datos
```

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

[ ]:
```
datos.dtypes
```

[ ]:
```
vocabulario = datos['Preferencia']
file3 = open ('vocabulario.txt', 'w')
file3.write(str((vocabulario)))
file2.close()
```

[ ]:
```
!sh install-tagger.sh
```

[ ]:
```
!echo 'esta es la prueba de instalación' | cmd/tree-tagger-spanish
```

[ ]:
```
!cat vocabulario.txt | cmd/tree-tagger-spanish > vocabulario_annotated.txt
```