Vocabulary Challenge

Get the book “Los Miserables” by Victor Hugo, from Carlos Slim Foundation: https://aprende.org/pruebat?sectionId=6

1. Convert the book into text (csv)
2. Clean the csv file
3. Standardize (no upper chars, no whitespace, no punctuation, accents, if possible)
4. Create a vocabulary with the words in the book, following the ideas when we discussed how to create a vocabulary in the slides (7..14) of this presentation.
5. Store the vocabulary on disk in parquet format
6. Do statistics, including

- How many words in original text (book)
- How many different words in vocabulary
- Print the 100 most frequent words in vocabulary
- Print the 100 least frequent words in vocabulary

7. Produce a 2 page report to describe your experience, methods, etc.
8. Write code in the language of your choice from this set (Go, Julia, Python) Assigned: 03/08/2025 Deadline: 03/14/2025 @ 04:00 PM CDMX Time, using the Github page : https://github.com/camachojua/diplomado-ia/tree/main/python/src/student_submissions/Vocabulary

Se cargan las librerías

- **`import csv`**: Permite leer y escribir archivos CSV (Comma-Separated Values).
- **`import string`**: Proporciona herramientas para trabajar con cadenas de texto, como la manipulación de caracteres, letras, puntuación, etc.
- **`import unicodedata`**: Ofrece funciones para trabajar con caracteres Unicode (normalizar texto y eliminar acentos).
- **`import pandas as pd`**: Manipulación de datos que facilita el trabajo con estructuras de datos, como DataFrames.

```
import csv
import string
import unicodedata
import pandas as pd
```

Función para limpiar el texto

Esta función toma como argumento un string y ejecuta tres principales tareas:

1. Pasar las cadenas a minúsculas
2. Eliminar acentos (si así se desea)
3. Eliminar signos de puntuación en la cadena

```
# Función para limpiar símbolos no imprimibles y quitar índices
def estandarizar_texto(texto):
    # Pasar a minúsculas
    texto = texto.lower()

    # Eliminar acentos
    #texto = ''.join(c for c in unicodedata.normalize('NFD', texto) if unicodeda

    # Eliminar puntuación y signos especiales como ¡¿
    texto = ''.join(c for c in texto if c not in string.punctuation and c not in

    return texto
```

Abrir el archivo y crear la lista con el texto limpio

Este bloque abre y lee el archivo CVS para después iterar sobre las lineas, a cada linea se le aplica la función `estandarizar_texto`, posteriormente se divide el texto limpio en palabras para despues filtrarlas y elmiminar todos los valores numéricos y por último agregarlas a una lista.

```
# Abrir y leer el archivo CSV
with open('Los-miserables.csv', 'r', encoding='utf-8') as file:
    reader = csv.reader(file)
    lista = []
    todas_las_palabras = []

    for row in reader:
        # Limpiar cada celda del row de forma explícita
        row_limpio = []
        for celda in row:
            texto_limpio = estandarizar_texto(celda)
            if texto_limpio:  # Evitar agregar celdas vacías
                row_limpio.append(texto_limpio)  # Agrega a la lista row_limpio

                # Divide el texto limpio en palabras
                palabras = texto_limpio.split()

                # Filtrar palabras que contengan números
                palabras_sin_numeros = [palabra for palabra in palabras if not a

                # Agregar las palabras filtradas (sin números) a todas_las_palab
                todas_las_palabras.extend(palabras_sin_numeros)

        if row_limpio:  # Evitar agregar filas vacías
            lista.append(row_limpio)  # Agregar la fila limpia a la lista
```

Crear una clase Dict para guardar el vocabulario

En el bloque anterior se generó una lista llamada `todas_las_palabras` , esta lista continene todas las palabras del archivo limpias, por lo que se requiere hacer un conteo y generar un diccionario que guarde las palabras como llaves el número de veces que aparecen en la lista como valor

```
# Crear el diccionario de conteo de palabras
vocabulary = {}

for palabra in todas_las_palabras:
    palabra = palabra.lower()  # Normalizar a minúsculas
    if palabra:
        if palabra in vocabulary :
            vocabulary [palabra] += 1
        else:
            vocabulary [palabra] = 1

# Con esta línea se invierte el orden, palabras más comunes van primero
vocabulary=dict(sorted(vocabulary.items(), key=lambda item: item[1], reverse=Tr
```

Imprimir las estadísticas del vocabulario (palabras más comunes, menos comunes, palabras que aparecen una sola vez)

```
# Ordenar las palabras por frecuencia (de mayor a menor)
palabras_ordenadas = sorted(vocabulary.items(), key=lambda x: x[1], reverse=Tru

# Crear un DataFrame con las 100 palabras más comunes
top_100 = pd.DataFrame(palabras_ordenadas[:10], columns=['Palabra', 'Frecuencia
bottom_100 = pd.DataFrame(palabras_ordenadas[-10:], columns=['Palabra', 'Frecue

# Imprimir como tabla
print("\nTop 100 palabras más frecuentes:")
print(top_100.to_string(index=False))

print("\n----\n")

print("Bottom 100 palabras menos frecuentes:")
print(bottom_100.to_string(index=False))

print("\n----\n")

palabras_unicas=0
# Iterar sobre las palabras y sus conteos
for i, conteo in vocabulary.items():
    if conteo == 1:  # Si el conteo es 1, es una palabra única
        palabras_unicas += 1

print(f'Número de palabras que aparecen una sola vez: {palabras_unicas}')

# Palabras que aparecen más de n veces
print("\n----\n")
palabras_n=0
for i, conteo in vocabulary.items():
    if conteo >= 500:
        palabras_n += 1

print(f'Número de palabras que aparecen más de 500 veces: {palabras_n}')
```

```
Top 100 palabras más frecuentes:
Palabra  Frecuencia
     de        5324
     la        3918
    que        3504
      y        3123
     el        3081
     en        2836
      a        2489
     se        1633
     un        1601
     no        1498

----

Bottom 100 palabras menos frecuentes:
     Palabra  Frecuencia
posiblemente           1
    reservar           1
  simplificó           1
    estricto           1
   enterrada           1
    gratuito           1
  cementerio           1
 encontrados           1
      sufrió           1
promiscuidad           1

----

Número de palabras que aparecen una sola vez: 7313

----

Número de palabras que aparecen más de 500 veces: 26
```

DataFrame usando Pandas

Para guardar el vocabulario en formato Parquet

```
df = pd.DataFrame(vocabulary.items(), columns=["w.s.", "w.f"])

# Guardar el DataFrame en formato Parquet
df.to_parquet("vocabulary.parquet", index=True)
```

Una vez creado el archivo .parquet, se carga para verificar que funcione

```
df_cargado = pd.read_parquet("vocabulary.parquet")
print(df_cargado)
```

```
                w.s.   w.f
0                 de  5324
1                 la  3918
2                que  3504
3                  y  3123
4                 el  3081
...              ...   ...
13259      gratuito     1
13260    cementerio     1
13261   encontrados     1
13262         sufrió     1
13263  promiscuidad     1

[13264 rows x 2 columns]
```