Word Cloud Generation

This code snippet is responsible for generating and displaying a word cloud visualization from the text extracted from GDPR documents.

Here's a breakdown of what each part does:

1. **Import necessary libraries:**

library\_add

content\_copy

from wordcloud import WordCloud  
import matplotlib.pyplot as plt

[Körültekintően használja a kódot](https://g.co/legal/generative-code)

* WordCloud: This library is used to create the word cloud visualization.
* matplotlib.pyplot: This library is used for general plotting and visualization in Python, in this case, for displaying the generated word cloud.

1. **Prepare the text:**

library\_add

content\_copy

text\_string = " ".join(clean\_text(texts))

[Körültekintően használja a kódot](https://g.co/legal/generative-code)

* This line takes the texts which contains the content of the GDPR documents and transforms this data into a single, large string using the clean\_text function to remove noise such as punctuation, extra spaces, converts the text into lower case, and removing common english words.
* clean\_text: This function preprocesses the text and likely removes stop words, converts to lowercase, and potentially performs lemmatization. We can assume these basic text cleaning steps are performed here, because of the comment in the code: Assuming \cleaned\_text` contains your preprocessed text (no stopwords, lowercased, lemmatized)`

1. **Generate the word cloud:**

library\_add

content\_copy

wordcloud = WordCloud(width=800, height=400, background\_color='white').generate(text\_string)

[Körültekintően használja a kódot](https://g.co/legal/generative-code)

* This line creates a WordCloud object with specified dimensions (width, height) and background color (background\_color).
* The generate() method is called with the prepared text\_string to create the word cloud, where the size of each word in the cloud is determined by its frequency in the text. More frequent words appear larger.

1. **Display the word cloud:**

library\_add

content\_copy

plt.figure(figsize=(10, 5))  
plt.imshow(wordcloud, interpolation='bilinear')  
plt.axis('off')  
plt.title("Most Frequent Terms in GDPR Documents")  
plt.show()

[Körültekintően használja a kódot](https://g.co/legal/generative-code)

* plt.figure(): Creates a new figure for the plot with a specified size (figsize).
* plt.imshow(): Displays the generated wordcloud image on the plot. interpolation='bilinear' is used for smoother rendering of the word cloud.
* plt.axis('off'): Hides the plot axes.
* plt.title(): Sets the title of the plot.
* plt.show(): Shows the plot containing the word cloud.

In summary, this code takes the processed text from GDPR documents, generates a word cloud that visually represents the most frequent terms in the text, and then displays the word cloud as an image. This visualization can give insights into topics, key concepts, and frequently occurring themes within the GDPR documents.

Források

A válasz értékelése

thumb\_up

thumb\_down

it runs into an error because the variable for the preprocessed text is incorrect can you help me find the correct value within this document?

spark

Okay, let's identify the issue and find the correct value for the preprocessed text variable.

The error likely arises from this line:

library\_add

content\_copy

text\_string = " ".join(clean\_text(texts))

[Körültekintően használja a kódot](https://g.co/legal/generative-code)

The issue is that clean\_text() expects a single string as input, but texts is likely a dictionary where keys are filenames and values are the corresponding text content.

To resolve this, we need to apply the clean\_text() function to each text content within the texts dictionary and then combine the results into a single string.

Here's the modified code snippet:

library\_add

content\_copy

from wordcloud import WordCloud  
import matplotlib.pyplot as plt  
  
# Assuming `cleaned\_text` contains your preprocessed text (no stopwords, lowercased, lemmatized)  
# Correctly apply clean\_text to each text in the texts dictionary  
cleaned\_texts = [clean\_text(text) for text in texts.values()]    
text\_string = " ".join([" ".join(cleaned\_text) for cleaned\_text in cleaned\_texts ])  
  
wordcloud = WordCloud(width=800, height=400, background\_color='white').generate(text\_string)  
  
plt.figure(figsize=(10, 5))  
plt.imshow(wordcloud, interpolation='bilinear')  
plt.axis('off')  
plt.title("Most Frequent Terms in GDPR Documents")  
plt.show()

[Körültekintően használja a kódot](https://g.co/legal/generative-code)

**Explanation of changes:**

1. **Preprocess Each Text:** cleaned\_texts = [clean\_text(text) for text in texts.values()]
   * This line iterates through the values (text content) of the texts dictionary.
   * For each text, it applies the clean\_text() function to preprocess it.
   * The results (cleaned texts) are stored in a new list called cleaned\_texts.
2. **Combine Cleaned Texts:** text\_string = " ".join([" ".join(cleaned\_text) for cleaned\_text in cleaned\_texts ])
   * This line now correctly joins all the cleaned text snippets from the cleaned\_texts list using spaces to create a single large string called text\_string.
   * First, it iterate over all the cleaned text in the cleaned\_text list and joins the text with " ".
   * Then, the text strings from the first iteration are joined together to a single string with text\_string

With these modifications, the text\_string variable should now contain the preprocessed text as intended, and the word cloud should generate without the previous error.