Importing Libraries

library\_add

content\_copy

import seaborn as sns  
import matplotlib.pyplot as plt

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

* import seaborn as sns: This line imports the seaborn library, which is a powerful tool for creating statistical visualizations in Python. We use the alias sns for convenience.
* import matplotlib.pyplot as plt: This line imports the pyplot module from the matplotlib library. matplotlib is a widely used library for creating static, interactive, and animated visualizations in Python, and pyplot provides a simple interface for creating plots. We use the alias plt for convenience.

Creating the Scatter Plot

library\_add

content\_copy

plt.figure(figsize=(10, 6))  
sns.scatterplot(data=merged\_df, x='personal data', y='Fine [€]')  
plt.title('Fine vs. Personal Data Mentions')  
plt.show()

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

* plt.figure(figsize=(10, 6)): This line creates a new figure (the drawing area for the plot) with a specific size of 10 inches by 6 inches. This ensures the scatter plot has enough space to be displayed clearly.
* sns.scatterplot(data=merged\_df, x='personal data', y='Fine [€]'): This line is the core of the visualization. It uses the scatterplot function from seaborn to create a scatter plot.
  + data=merged\_df: This argument specifies the data source for the plot, which is the merged\_df DataFrame. This DataFrame likely contains information about GDPR fines and mentions of personal data.
  + x='personal data': This argument specifies that the 'personal data' column from merged\_df should be used for the x-axis of the scatter plot.
  + y='Fine [€]': This argument specifies that the 'Fine [€]' column from merged\_df should be used for the y-axis of the scatter plot.
* plt.title('Fine vs. Personal Data Mentions'): This line sets the title of the plot, clearly indicating what it represents.
* plt.show(): This line displays the created scatter plot to the user.

In Summary

This code snippet visualizes the relationship between GDPR fines and the number of times "personal data" is mentioned in related documents using a scatter plot. Each point on the plot represents a document, with its position determined by the fine amount and the frequency of "personal data" mentions. This visualization helps to explore whether there is a correlation between these two factors.

A válasz értékelése

thumb\_up

thumb\_down

The fine amount seems to be in scientific format and it is unlabeled, please fix these

spark

Okay, let's adjust the code to address the scientific notation and unlabeled fine amounts in the scatter plot:

library\_add

content\_copy

import seaborn as sns  
import matplotlib.pyplot as plt  
  
plt.figure(figsize=(10, 6))  
# Create the scatter plot with Seaborn  
scatter\_plot = sns.scatterplot(data=merged\_df, x='personal data', y='Fine [€]')  
  
# Format y-axis labels to display full numbers instead of scientific notation  
scatter\_plot.yaxis.set\_major\_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))  
  
# Add a y-axis label  
plt.ylabel('Fine (€)')   
  
plt.title('Fine vs. Personal Data Mentions')  
plt.show()

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

Here's how the changes address the issues:

Fixing Scientific Notation:

library\_add

content\_copy

scatter\_plot.yaxis.set\_major\_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

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

* scatter\_plot.yaxis.set\_major\_formatter(): This line targets the y-axis of the scatter plot to modify its tick labels.
* plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))): This part uses a FuncFormatter to create a custom formatting function. Let's break it down:
  + lambda x, loc:: This creates an anonymous function that takes two arguments: x (the tick value) and loc (the tick location).
  + "{:,}".format(int(x)): This is the formatting string. It converts the tick value (x) to an integer using int(x), and then formats it with commas as thousands separators using "{:,}". This prevents scientific notation.

Adding Y-Axis Label:

library\_add

content\_copy

plt.ylabel('Fine (€)')

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

This line simply adds a descriptive label to the y-axis, indicating that the values represent "Fine (€)".

With these modifications, the scatter plot will have more readable fine amounts without scientific notation and a clear label for the y-axis.

A válasz értékelése

thumb\_up

thumb\_down

thank you, i need another visualisation with the highest fines word distribution

spark

Okay, let's create a visualization to show the word distribution for documents with the highest fines:

library\_add

content\_copy

from wordcloud import WordCloud  
import matplotlib.pyplot as plt  
  
# 1. Filter for highest fines   
# Assuming 'merged\_df' contains your fine and text data  
highest\_fines\_df = merged\_df.nlargest(10, 'Fine [€]') # Select top 10 based on Fine column  
  
# 2. Combine text from highest fine documents  
text\_string = " ".join(highest\_fines\_df['Text'].astype(str).tolist()) # Assuming 'Text' column holds the text  
  
# 3. Preprocess text (clean, tokenize, etc.) if not already done  
# This step is important to remove irrelevant words (stop words, punctuation) and improve visualization  
# You might have functions like 'clean\_text' from earlier parts of the code  
  
cleaned\_text = clean\_text(text\_string)  # Example assuming 'clean\_text' function is available  
text\_string = " ".join(cleaned\_text)  
  
# 4. Create and display the word cloud  
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 Highest Fine GDPR Documents")  
plt.show()

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

**Explanation:**

1. **Filter for Highest Fines:**
   * highest\_fines\_df = merged\_df.nlargest(10, 'Fine [€]'): This line selects the top 10 rows (documents) from your merged\_df DataFrame based on the values in the 'Fine [€]' column. You can adjust the number 10 if you want a different number of highest fine documents.
2. **Combine Text:**
   * text\_string = " ".join(highest\_fines\_df['Text'].astype(str).tolist()): This line extracts the text data from the selected documents (assuming a column named 'Text' exists) and combines it into a single string called text\_string. It converts the text to strings if necessary and uses join to combine them with spaces.
3. **Preprocess Text:**
   * This step is crucial for a meaningful word cloud. It's where you clean up the text by removing things like stop words (common words like "the", "a", "is"), punctuation, and potentially performing other tasks like stemming or lemmatization (reducing words to their base form).
   * *Replace the comment with your actual cleaning function and modifications*
4. **Create Word Cloud:**
   * wordcloud = WordCloud(width=800, height=400, background\_color='white').generate(text\_string): This line creates a WordCloud object with the desired dimensions and background color and then generates the word cloud from the preprocessed text\_string.
   * plt.imshow(wordcloud, interpolation='bilinear'): This displays the word cloud as an image.
   * plt.axis('off'): This removes the axis lines.
   * plt.title("Most Frequent Terms in Highest Fine GDPR Documents"): This sets the title of the plot.
   * plt.show(): This shows the plot.

**Important Considerations:**

* **Preprocessing:** The quality of the word cloud heavily depends on how well you preprocess the text. Make sure you remove unnecessary words and symbols.
* **Adjustments:** You can customize the appearance of the word cloud by changing parameters like width, height, background\_color, and other options available in the WordCloud object.
* **Data:** Ensure that your 'Text' column in merged\_df actually contains the text data you want to visualize. Adjust the code if your text data is stored in a different column.

A válasz értékelése