**C1**

import pandas as pd

from gensim.models import Word2Vec

import re

import string

# Assuming 'ten\_k\_df' contains the full 10-K text data with a 'text' column

# Preprocess text: remove punctuation, lowercase, and split into words

# Step 1: Preprocess the text

def preprocess\_text(text):

text = text.lower() # Lowercase

text = re.sub(f'[{string.punctuation}]', '', text) # Remove punctuation

return text.split() # Tokenize text into words

# Apply preprocessing to each document in the 10-K dataset

ten\_k\_df['clean\_text'] = ten\_k\_df['text'].apply(preprocess\_text)

# Step 2: Train Word2Vec Model

# Collect all tokenized sentences into a list of lists

sentences = ten\_k\_df['clean\_text'].tolist()

# Define and train the Word2Vec model

# min\_count: Ignores words with frequency lower than this

# vector\_size: Size of word vectors

# window: Context window size

word2vec\_model = Word2Vec(sentences, vector\_size=100, window=5, min\_count=2, workers=4)

# Step 3: Save or explore the model

# Save the trained model

word2vec\_model.save("10k\_word2vec.model")

# Example exploration: finding similar words to "growth"

# Check if the word "growth" is in the vocabulary

if "growth" in word2vec\_model.wv:

print(word2vec\_model.wv.most\_similar("growth"))

else:

print("The word 'growth' is not in the vocabulary.")

**C2**

# Assuming word clouds were generated and inspected in Step B.3, and three representative keywords were chosen

# Let's define these keywords for further analysis

# Example chosen keywords relevant to the industry sector of interest

keywords = ["innovation", "sustainability", "growth"]

# Display the chosen keywords

Keywords

**C3**

# Loading the necessary libraries and assuming the Word2Vec model has been trained and saved as "10k\_word2vec.model"

from gensim.models import Word2Vec

# Load the trained Word2Vec model

word2vec\_model = Word2Vec.load("10k\_word2vec.model")

# Define the selected keywords (based on industry relevance, as discussed)

selected\_keywords = ["growth", "innovation", "sustainability"]

# Check if the keywords are in the model's vocabulary

valid\_keywords = [keyword for keyword in selected\_keywords if keyword in word2vec\_model.wv]

# Find the most similar five words for each valid keyword

similar\_words = {keyword: word2vec\_model.wv.most\_similar(keyword, topn=5) for keyword in valid\_keywords}

# Display the similar words

similar\_words

valid\_keywords