Question: Is it always necessary to vectorize text data?
Answer: No. Some NLP libraries do require a numeric format but others such as spaCy, do not.
Explanation: While it's often beneficial to vectorize text data for many machine learning models, some NLP libraries like spaCy can work directly with text data. spaCy creates word embeddings for each word, which are vector representations, but this is done internally.

Question: Which one of the following methods could not be used to represent the values in a document-term matrix (DTM)?
Answer: label encoding
Explanation: Label encoding assigns a unique numerical value to each unique categorical value. In the context of a document-term matrix, this wouldn't make sense because we're interested in the frequency (count vectorizer) or importance (TF-IDF) of each word in the document, not just whether it's present (one-hot encoding).

Question: Which of the following is a distance metric that is used to measure the distance between two word vectors?
Answer: cosine similarity
Explanation: Cosine similarity is a measure that calculates the cosine of the angle between two vectors. This can be used to understand how similar two word vectors are in a high dimensional space.

Question: One of the following cosine similarities between the set of documents doc0, doc1, and doc2 is a typo:
doc0 and doc1 is 2.2 doc0 and doc2 is 0.50 doc1 and doc2 is -0.5
Which two documents have an impossible value for cosine similarity?
Answer: doc0 and doc1
Explanation: Cosine similarity ranges from -1 to 1. A value of 2.2 is outside this range, so it is not a valid cosine similarity score.

Question: Which of the following choices is the best description of a word embedding?
Answer: Where each word is represented as a dense, fixed-length vector of floats.
Explanation: Word embeddings represent words in a high-dimensional space where the location and distance between words indicate how similar they are.

Question: If two word embeddings have very similar values, what can we say about those two words? Answer: The words have related meanings.
Explanation: Word embeddings are designed such that words with similar meanings have similar vectors.

Question: If given a list of strings named text, which of the following code choices will print the vocabulary learned by an instantiated CountVectorizer() class? Assume that vectorizer = CountVectorizer(). Answer: print(vectorizer.vocabulary_)

Explanation: The vocabulary_ attribute of a fitted CountVectorizer instance holds a dictionary where keys are terms and values are indices in the feature matrix.

Question: What does the following code return? Assume that text is a list of strings and vectorizer is an instance of the CountVectorizer().
vectors = vectorizer.transform(text) vectors_dense = pd.DataFrame(vectors.todense(), columns=vectorizer.get_feature_names())
Answer: This code returns a DataFrame, where the columns represent each word in the vocabulary and the rows represent the word counts for each document.
Explanation: The to_dense() function converts the sparse matrix output of transform() to a dense matrix. The resulting DataFrame has one row per document and one column per word in the vocabulary.

Question: When you apply the cosine_similarity method from the Guided Project to a document-term matrix and convert the result to a DataFrame, what does each row represent? Answer: Each row represents the similarity of one document to all other documents (including itself).
Explanation: The cosine_similarity function returns a matrix where each row corresponds to a document, and each column corresponds to a document. The value in each cell is the cosine similarity between the two documents.

Question: Which of the following code choices would return the word vector for the token "snowstorm"? Assume the spaCy language model available is nlp = spacy.load('en_core_web_lg'). Answer: doc = nlp("snowstorm") snowstorm_vector = doc.vector
Explanation: This code creates a spaCy document from the word "snowstorm" and then retrieves its vector representation.