Lab 8: Cosine and Euclidean Distance

This is an INDIVIDUAL assignment. Due date is as indicated on BeachBoard. Follow ALL instructions otherwise you will lose points. In this lab, you will be classifying data using cosine distance and Euclidean distance.

Background:

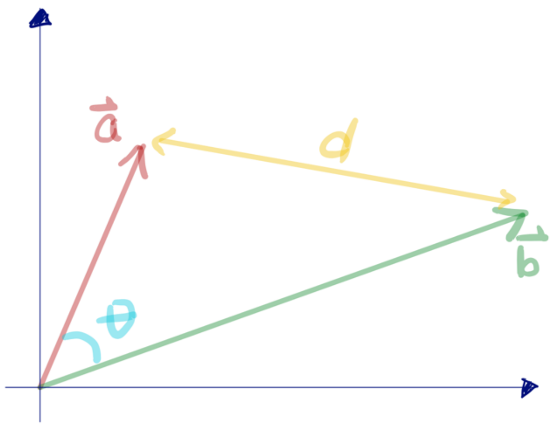
You have already learned about the cosine formula:

$$cos\theta = \frac{a \cdot b}{|a||b|}$$

You can find the angle created by two vectors by using this formula.

You have also learned how to find the distance between two vectors:

$$d = \sqrt{(a-b)\cdot(a-b)}$$



The idea is to classify data based on these distance metrics. For example, if you have two vectors that have a relatively small angle between them, they could be similar.

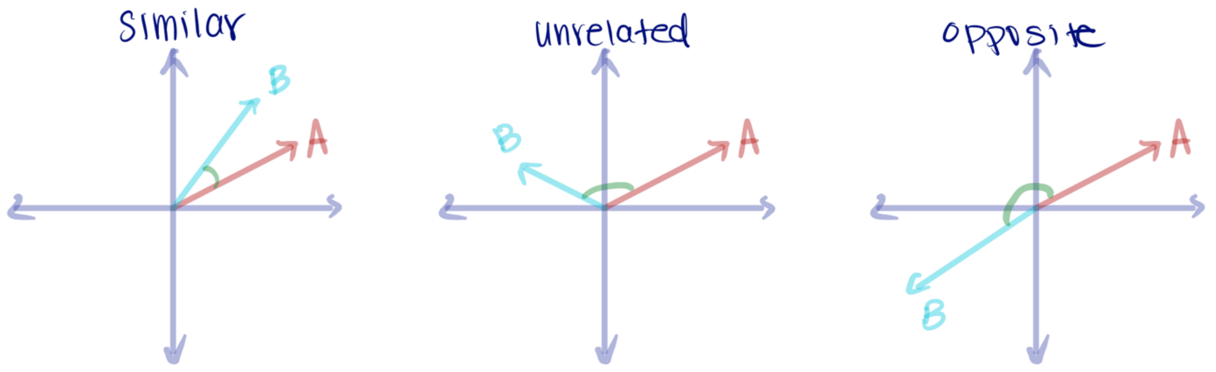


Figure 1: using cosine similarity

Another example: two vectors can be measured using traditional distance to see how far apart the two vectors are. If they are far apart (shown in green), then they could be unrelated. If the vectors are close, then they are more likely to be similar.

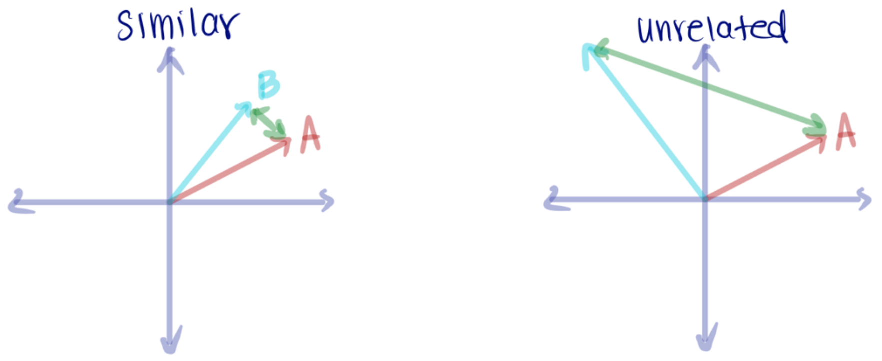


Figure 2: using distance metric

Task:

1. Take a close look at the `metric_similarity.py` file. There are some functions that have already been implemented. In addition, there are two functions that you need to fill in: `most_similar_cosine()` and `most_similar_euclid()`. Read through both of their descriptions carefully.
   - `most_similar_cosine()` : Find the class that best matches the test input using cosine similarity . Whichever vector in the class list has the smallest angle from the test vector is the class that you want to return.
   - `most_similar_euclid()` : Find the class that best matches the test input using Euclid's distance. Whichever vector in the class list has the smallest distance from the test vector is the class that you want to return.

   Remember, you will lose points if you do not follow the instructions. We are using a grading script.

2. Your job is to implement both `most_similar_cosine()` and `most_similar_euclid()` so that it passes any test case. There are four sample test cases provided for you, but these are not the only cases that we will test. We will be testing other test cases in the same way the test cases are presented.
3.
4. After completing these functions, comment out the test cases (or delete them) or else the grading script will pick it up and mark your program as incorrect.

5. Rename your `metric_similarity.py` file to
   `lastName_firstName_section#_idNumber.py`
   This step is very important!!!! Do not forget it! Please avoid spaces and additional symbols when renaming your file.

6. Convert your newly named `metric_similarity.py` file to a `.txt` file with the same naming scheme as above. Submit your newly named `Vector.py` file and your `.txt` file on BeachBoard. Do NOT submit it in compressed folder.

Grading rubric:

To achieve any points, your submission must have the following. Anything missing from this list will result in an automatic zero. NO EXCEPTIONS!

- Submit everything: py file, txt file
- Files named correctly
- Program has no errors (infinite loops, syntax errors, logical errors, etc.) that terminates the program

Please note that if you change the function headers or if you do not return the proper outputs according to the function requirements, you risk losing all points for those test cases.

Points	Requirement
10	Implemented `similar_cosine()` correctly
10	Implemented `most_similar_euclid()` correctly
TOTAL: 20	