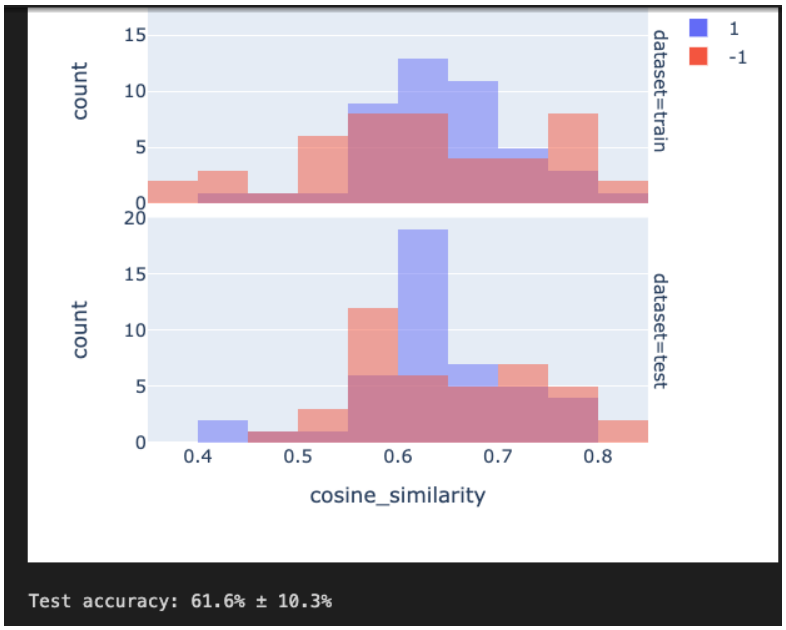
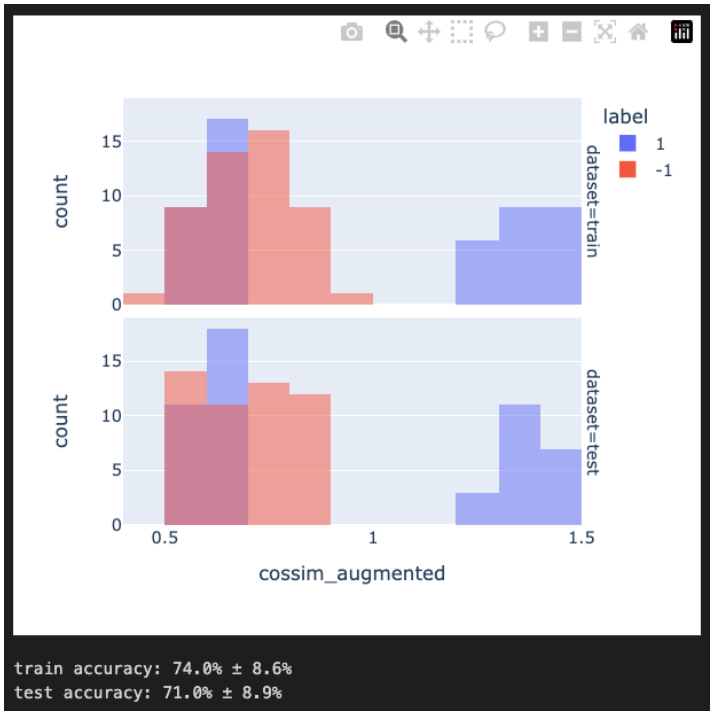
Following the linked Notebook, I generate synthetic negatives, or unrelated query-code pairs, and embed 200 datapoints in total. The metric is the difference between the cosine similarity distribution of the positives and negatives.

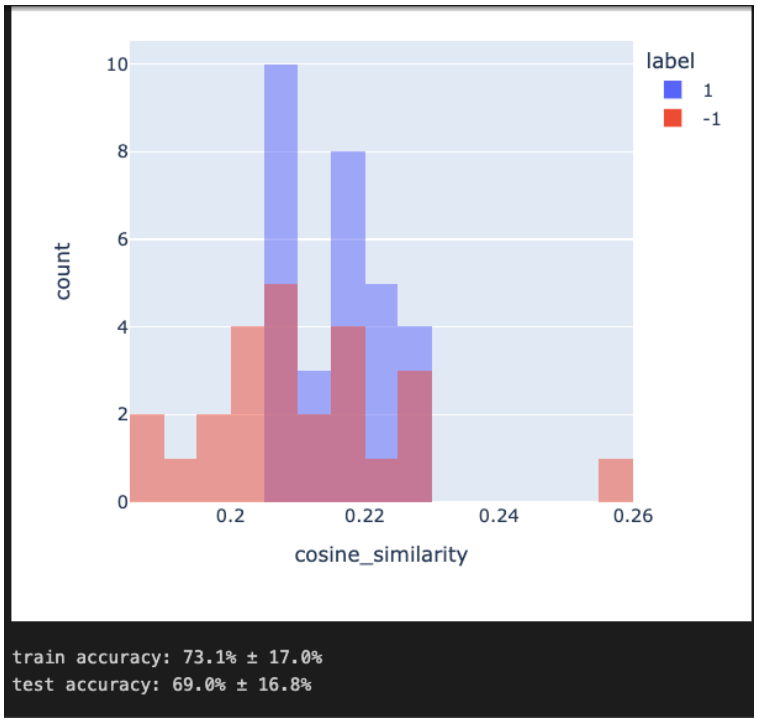
Attempt 0: Naive cosine similarity using the babbage-similarity engine from the OpenAI API:



Improvement attempt 1: I wanted to explore using information from the README: if the query and code both appear in the README, heighten their relevance score by adding their average cosine similarity to the naive cosine similarity. This gives better separation than the vanilla encoding, but not as good as the optimized separation.



Improvement Attempt 2: This Colab Notebook: [Customizing_embeddings.ipynb]



Limitations: I had to comment out the README embeddings because the API somehow timed out on them. The results still fall short of just using the Cookbook notebook out of the box (shown below) with the matrix optimization. I didn't have time to tune the temperature or normalization factor, which should yield better results; more importantly, the idea of local-global mixing can allow for fine-grained searches, and should show significant advantages given a more locally labeled dataset.