**introduction**

**With the introduction of commercially available AI tools like ChatGPT, new challenges arise in the realm of digital communication. Now, the need to develop tools capable of discerning between human-authored text and AI-generated content is evermore crucial. From job applications to opinion surveys, ascertaining that the author is human is essential for the integrity and effectiveness of the operation. In this project, I utilize a simple machine learning model to classify text based on whether the author is human or an AI.**

**goals**

* develop a model capable of discerning between human text and AI generated content.
* Explore various text preprocessing methods for machine learning like vectorization and document embeddings.
* Optimize model to use the best hyperparameters.
* Showcase model accuracy and precision.

**Method and findings**

**Data overview.**

Dataset is 487235 entries , two columns,

text: text paragraphs of 500 -600 words on average.

generated: 0 or 1 value to signify whether the content is AI generated or not , Generated by AI (1) , by Human (0).

**Splitting the data.**

splitting the data twice because it's too big to test on

-first split(on the original dataset): 96% validation 4%subset.

- second split(on the 4% subset): 80% train 20% test.

All the tests will be performed on the second split.

**Word vectorization.**

* Word vectorization is a technique that converts words in a text into numerical vectors , where semantically similar words are mapped to nearby points, here we are using the nltk vectorization.
* A scikit transformer was built that applies tokenizes the text into words then removes words that are present in list, the list contains punctuation , stop words such as “the – or -and” and tags such as “ \n - \’ ”.
* A pipeline was built to utilize the scikit transformer and apply tf-idf tokenization, which is a method that converts words into a numerical statistic that reflects the importance of a word in a document relative to a collection of documents.
* The pipeline ends with a linear support vector machine model(SVC).
* The resulting accuracy of the tf-idf vectorization pipeline was 99.49%.

**Random search.**

* Random search was used on the tf-idf pipeline to look for the optimal hyperparameters for the svc.
* The best hyper parameters were:
  + C: 15805.16646233074,
  + gamma: 0.15813763277311954
  + kernel: rbf.
* The resulting accuracy using the best parameters was 99.56%.

**Document embeddings.**

* Embeddings are dense, fixed-length numerical representations of entities such as words, phrases, or documents , unlike tf-idf vectorization they capture the semantic relationships between words.
* Document embeddings capture the overall semantic meaning and context of an entire document as a whole.
* A pipeline was built using doc embeddings and a linear svc.
* The resulting accuracy of the doc embeddings pipeline was 98.53%.

**Comparing embeddings vs vectorization.**

When comparing both pipelines, we find that the vectorization pipeline exhibits superior precision and recall. Additionally, the area under the roc curve is larger for the vectorization pipeline.

When comparing the training curves, the vectorization pipeline showcases a more favorable learning curve as accuracy increases with sample size as opposed to the embeddings curve which initially increases in accuracy then declines which implies possible overfitting.

**Training vectorization on 80% of the whole data.**

Running 80% of the whole dataset as training data through the vectorization pipeline , It results in a roc-auc score of 99.67%

**Libraries and dataset**

Dataset link: https://www.kaggle.com/datasets/shanegerami/ai-vs-human-text

Python version: 3.11.7

NumPy version: 1.26.4

Pandas version: 2.1.4

Matplotlib version: 3.8.0

Scikit-learn version: 1.2.2

gensim version: 4.3.0

SciPy version: 1.11.4

nltk version: 3.8.1