# Report
# Detect AI Generated Text

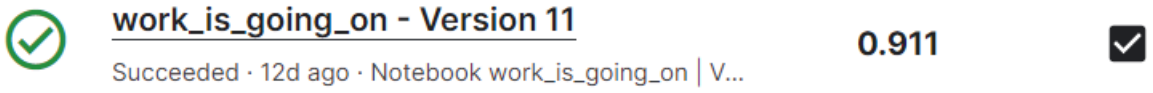
Garv Sethi
22115057

January 2024

GitHub Repository

## 1 Progress

So far i have tested different models and vectorizers/tokenizers to see how accurate my model performs, i have used the BERT Tokenizer and the fast tokenizer provided by HuggingFace.
The best score i have got so far is of 91.1%.




## 2 Approach

### 2.1 Imbalanced Data

Upon initial inspection, it became evident that the supplied data exhibited a significant skew, rendering it unsuitable for effective model training. Recognizing this limitation, I opted to integrate an external dataset, specifically Large Language Models, to ensure a more balanced representation. The external dataset featured a balanced ratio of AI-generated content to human-written text. This strategic augmentation not only addressed the imbalance issue but also enriched the training data, fostering a more robust and comprehensive learning experience for the model.

## 2.2 Tokenizer

To process the words we need to convert them to tokens so that we can convert them to vectors to predict distances between the words and hence use that to check for LLM detection. I used the BytePair Encoding strategy which involves merging words with close frequencies together until we reach a desired vocab size( which i had set to 30,000 in my case) . After the tokenizer was trained i applied them to the train and the test datasets.

## 2.3 TF-IDF Vectorization

Now we convert the tokens to vectors so now we can effectively calculate the distances between them for this I use the technique of TF-IDF Vectorization.

$$\text{TF(t,d)} = \frac{\text{Number of times term t appears in document d}}{\text{Total number of terms in document d}} \tag{1}$$

$$\text{IDF(t,d)} = \log\left(1 + \frac{\text{Total number of documents in the collection D}}{\text{Number of documents containing term t}}\right) \tag{2}$$

and using this TF-IDF score a vector is assigned to teach token in the document. We are using Bag of N-Grams for vectorization.

## 2.4 Model Selections

In the process of model experimentation, I discovered that Naive Bayes consistently outperformed other models, yielding an impressive accuracy of 91.1%. Stochastic Gradient Descent, while still commendable, achieved a slightly lower score of 88.6%. To leverage the strengths of both models, I decided to implement a Voting Classifier. This ensemble approach involved combining the predictive abilities of the Gradient Descent Algorithm and the Naive Bayes Algorithm with varying weights. By fine-tuning these weights, I aimed to harness the complementary strengths of each model, ultimately enhancing the overall predictive performance of the system. This strategic combination not only showcased the versatility of ensemble methods but also allowed me to achieve a more robust and reliable model for the given task.

### 2.4.1 Naive Bayes

Naive Bayes' is a classification algorithm(often dubbed as Idiot's Bayes') which is based on the Bayes' Theorem. The algorithm makes the bold assumption that the probability of all posteriors are independent given the class priors. This obviously is a false assumption but it tends to work good enough and speeds up the pipeline (source: Andrew Ng Course CSN229).

#### 2.4.2 Gradient Descent

As the name suggest we use the $\nabla$ operator here to find the direction with the maximum slope and move in the negative direction along it , until we reach the local optima. We Try to minimize the MSE(Mean Squared Error) function .

## 3 Learnings

In this project, I dived into the world of natural language processing (NLP) and machine learning. I learned how to process text through the NLP pipeline, breaking it down into smaller parts using tokenization. Decision trees became my go-to for understanding patterns in data, while text preprocessing taught me how to clean and prepare messy text for analysis. Vectorization, turning words into numbers, became a key step for machine learning. I got hands-on with various classification algorithms like Support Vector Machines, Naive Bayes, Gradient Descent, and Logistic Regression. Each algorithm had its strengths, and applying them in real-world scenarios showed me the practical side of these concepts. Overall, this project was a hands-on journey into the practical world of NLP and machine learning.