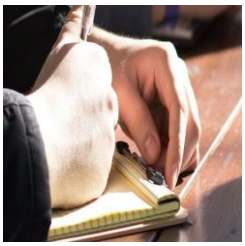
Final Project: “LLM Text Detection”

Advanced Machine Learning

Professor: Dr. Farahani



Group Members: Mehrdad Baradaran - Hossein Yahyaei - Katayoun Kobraei

Abstract –

This research addresses the task of classifying sequential data into two distinct categories, employing a combination of deep learning and classical machine learning approaches. The study begins with an exploratory data analysis (EDA) to understand the underlying patterns and imbalances in the dataset. Preprocessing steps involve data balancing, augmentation, and feature extraction, enhancing the dataset's suitability for model training. The initial attempt involves training a neural network with an autoencoder and classifier, which, however, shows limited success. Subsequently, a more sophisticated LSTM model is designed, yet resource constraints hinder a detailed analysis. Classical machine learning algorithms, namely logistic regression, random forest, and gradient boosting, are then employed, showcasing outstanding performance with ROC-AUC scores of 0.9971, 0.9999, and 0.9868, respectively. This suggests the efficacy of classical models in handling sequential data classification tasks.

Introduction

In the realm of natural language processing and machine learning, the ability to discern between essays crafted by students and those generated by language models (LLMs) has become a crucial challenge. This competition sets forth a task that requires participants to develop adept machine learning models capable of accurately detecting the origin of an essay, distinguishing between human-authored content and that which emanates from sophisticated language models.

The task mandates the implementation of, at a minimum, two models—one grounded in deep learning techniques and another based on non-neural network methodologies. The emphasis lies not only on achieving commendable performance but also on maintaining fairness between the two approaches; dismissing the non-neural network method outright due to lower accuracy is not acceptable.

The performance of the models will be evaluated using the Receiver Operating Characteristic Area Under the Curve (ROC-AUC) metric. ROC-AUC, a widely recognized measure in binary classification tasks, provides a comprehensive assessment of a model's ability to distinguish between positive and negative instances.

Dataset description

The competition dataset serves as the cornerstone for participants seeking to unravel the intricacies of distinguishing between human-authored essays and those generated by large language models (LLMs). Curated to reflect the dynamic landscape of language generation technology, this dataset comprises a diverse array of essays contributed by students alongside counterparts crafted by various LLMs.

Structured with three key columns, the dataset encompasses essential information for model training and evaluation:

1. Prompt_ID: Each essay is associated with a unique prompt identifier, providing a structured way to organize and reference the diverse array of writing prompts that elicited the essays.
2. Text: The text column contains the actual content of the essays. It serves as the input feature for machine learning models, presenting a mix of writing styles, topics, and perspectives that challenge participants to develop models with a nuanced understanding of both human and machine-generated linguistic patterns.
3. Label: The label column denotes the binary classification target for the machine learning task. A label of 0.0 indicates a student-written essay, while a label of 1.0 signifies an essay generated by an LLM. This binary distinction serves as the focal point for model predictions, guiding the algorithm to differentiate between the two sources of content.

Lack of data issue

The following is a summary of the processes and tasks carried out during this task. Since the available dataset for this task is highly unbalanced, with almost no data labeled as 1 (indicating text generated by the LLM model, only 3 instances), and virtually no test data available, there is a need to balance the data. Various methods can be employed for this purpose, including data generation and data augmentation. Today, with the abundance of artificial intelligence language models (LLM), tools such as GPT and Bing AI can be utilized for generating fake data. However, in this case, the preference is to use datasets in the same data format but with balanced distributions. The alternative approach involves the use of other datasets adhering to a similar data format.

In the subsequent sections, each dataset used for addressing this task will be introduced, and their samples and formats will be examined.

Dataset Aggregation Overview

The datasets, with assigned names for clarity, are a combination of the original dataset ("llm-detect-ai-generated-text") and various additional datasets introduced to augment the training process. Below is a summary of each dataset, along with the aggregated statistics:

1. Original Dataset ("original"): The foundational dataset, sourced from the Kaggle competition, provides a starting point with 1378 records.
2. daigt-v2-train-dataset ("daigt"): A substantial addition with 44,868 records, offering a significant expansion of the training data.
3. daigt-external-dataset ("daigt_ex"): Further extending the diversity, this dataset introduces 4,842 records from external sources.
4. daigt-proper-train-dataset ("daigt_pr"): A substantial training dataset, featuring 44,206 records, contributing to the model's understanding of both LLM and student-generated essays.
5. llm-7-prompt-training-dataset ("prompt7"): Enriching the dataset with 17,251 records, this dataset focuses on essays generated in response to seven specific prompts.
6. daigt-data-llama-70b-and-falcon180b ("falcon"): Incorporating 7,000 records, this dataset introduces a unique perspective by including essays generated by LLMs with varying capabilities.
7. 4k-mixtral87b-crafted-essays-for-detect-ai-comp ("mixtral"): Introducing 3,865 records, this dataset brings diversity to the training data by including crafted essays generated by LLMs.
8. hello-claude-1000-essays-from-anthropic ("claude"): With 1,000 records, this dataset contributes essays from the perspective of "Claude," an LLM, adding a nuanced touch to the training set.
9. llm-generated-essay-using-palm-from-google-gen-ai ("palm"): A dataset comprising 1,384 records, featuring essays generated by an LLM using the "Palm" method from Google Gen AI.
10. persuade-corpus-2 ("persuade"): With a sizeable 25,996 records, this dataset introduces persuasive essays, further challenging the models to discern intent and writing style.
11. llm-mistral-7b-instruct-texts ("mistral7B"): Adding 4,900 records, this dataset focuses on essays generated by LLMs using the "Mistral-7B" method, providing a unique flavor to the training set.
12. augmented-data-for-llm-detect-ai-generated-text ("augmented"): A substantial contribution with 433,564 records, this dataset significantly augments the training data with diverse essays from both students and LLMs.
13. daigt-v4-train-dataset ("daigt-v4"): Concluding the list, this dataset brings in 73,573 records, further diversifying the training set for comprehensive model learning.

Dataset samples and format

- The aggregated dataset encompasses a total of 527,720 records, formed by combining all the individual datasets.
- Each record contains information about a specific essay, including the associated prompt ID, the essay text, and the binary label indicating whether it was written by a student (label 0.0) or generated by an LLM (label 1.0).
- The total size of loaded datasets across all the contributed datasets is 663,827 records. However, upon aggregation, the dataset comprises 527,720 unique records.

```
Aggregated datasets sample:
```

	prompt_id	text	label
0	0	Cars. Cars have been around since they became ...	0.0
1	0	Transportation is a large necessity in most co...	0.0
2	0	"America's love affair with it's vehicles seem...	0.0

Now, for a better understanding, we will observe a sample of the data generated by the LLM model alongside human handwritten text.

----------: AI text sample: ----------

This essay will analyze, discuss and prove one reason in favor of keeping the Electoral College in the United States for its presidential elections. One of the reasons to keep the electoral college is that it is better for smaller, more rural states to have more influence as opposed to larger metropolitan areas that have large populations. The electors from these states are granted two votes each. Those from larger, more populated areas are granted just one vote each. Smaller states tend to hold significant power because their two votes for president and vice president add up more than the votes of larger states that have many electors. This is because of the split of the electoral votes. Some argue that electors are not bound to vote for the candidate who won the most votes nationally. They do not have to vote for their own state's nominee unless their state has a winner take all system. However, there are states that have adopted laws that force their electors to vote for their state's candidate. It seems that, no matter how, electors are not bound to vote for the candidate who won the most nationally. This is not always the case because of state legislatures who can overrule the electors and vote for the alternative candidate their citizens have selected for them, even if the voter lives in a state without a winner take all system.

----------: Human text sample: ----------

Cars. Cars have been around since they became famous in the 1900s, when Henry Ford created and built the first ModelT. Cars have played a major role in our every day lives since then. But now, people are starting to question if limiting car usage would be a good thing. To me, limiting the use of cars might be a good thing to do.

In like matter of this, article, "In German Suburb, Life Goes On Without Cars," by Elizabeth Rosenthal states, how automobiles are the linchpin of suburbs, where middle class families from either Shanghai or Chicago tend to make their homes. Experts say how this is a huge impediment to current efforts to reduce greenhouse gas emissions from tailpipe. Passenger cars are responsible for 12 percent of greenhouse gas emissions in Europe...and up to 50 percent in some carintensive areas in the United States. Cars are the main reason for the greenhouse gas emissions because of a lot of people driving them around all the time getting where they need to go. Article, "Paris bans driving due to smog," by Robert Duffer says, how Paris, after days of nearrecord pollution, enforced a partial driving ban to clear the air of the global city. It also says, how on Monday, motorist with evennumbered license plates were ordered to leave their cars at home or be fined a 22euro fine 31. The same order would be applied to oddnumbered plates the following day. Cars are the reason for polluting entire cities like Paris. This shows how bad cars can be because, of all the pollution that they can cause to an entire city.

Exploratory Data Analysis (EDA) and data visualization

Finally, it is time to conduct a series of Exploratory Data Analysis (EDA) and data visualization to examine the relationships between features and the distribution of the data.

Fig1. In the first visualization, the number and distribution of handwritten and LLM-generated data can be observed. Clearly, the handwritten data outnumber the generated data. The total dataset is divided with a ratio of 317,228 handwritten data points to 210,492 data points generated by the LLM.

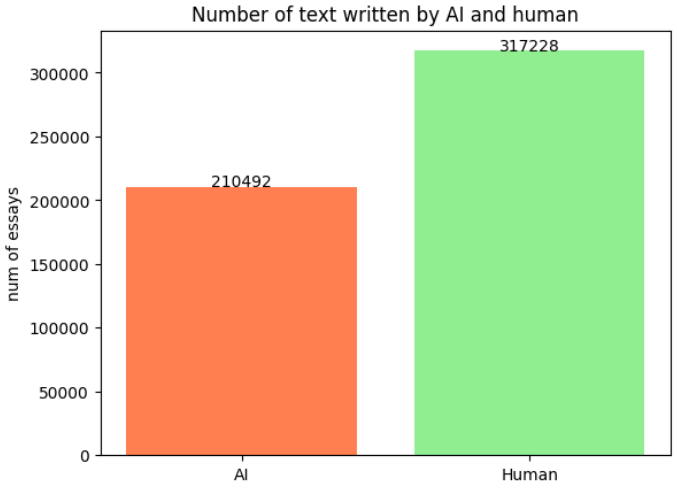
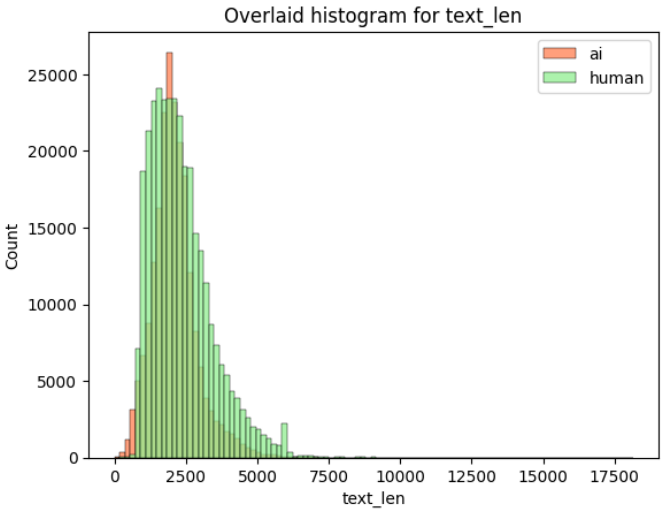
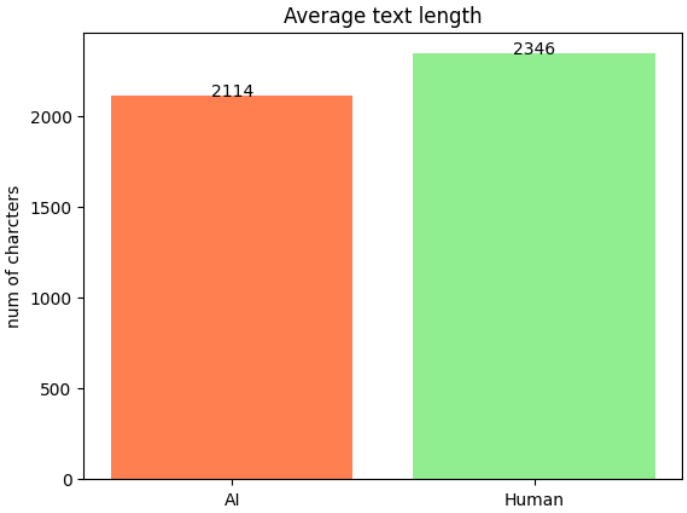


Fig2. To visualize and examine the length of text written by both humans and AI, we utilized a histogram. Several insights can be gleaned from this plot. It can be observed that the distribution of both datasets is approximately normal, and they share a somewhat similar average text length. However, a distinct characteristic is evident in the human-written text. Specifically, it tends to have a



greater length, with almost no instances of very short texts. On the other hand, longer texts are somewhat present in the human-written dataset. In contrast, the distribution of texts generated by the LLM model is concentrated within a specific range.

Fig3. As mentioned in the previous chart, the average length of text generated by both, while having a slight difference, is relatively similar.



Analysis of Exclusive N-grams in AI-Generated and Human-Authored Essays

1-grams:

- **AI-Exclusive:**
  - Notable presence of symbols ("–", "😅", "🤔") suggests a reliance on emoticons and special characters.
  - Specific terms like "hillary," "oxides," and "symmetrical" may indicate contextual topics or preferences unique to AI-generated content.
- **Human-Exclusive:**
  - Names ("gavin") and unique terms ("unrwa," "blimplike") appear, potentially reflecting specific subjects or writing styles common in human essays.

Top 10 1-grams only used by ai:

	1-gram	freq
0	–	4098
1	hillary	3901
2	😅	2773
3	oxides	2522
4	🤔	2348
5	biteable	2222
6	symmetrical	2080
7	😀	1646
8	don't	1607
9	one's	1491

Top 10 1-grams only used by human:

	1-gram	freq
0	gavin	5685
1	unrwa	5037
2	blimplike	1916
3	afl	1259
4	mesalandforms	872
5	edgarsnyder	775
6	□	708
7	vaughn's	659
8	¨	563
9	palpabraeus	533

2-grams:

- **AI-Exclusive:**
  - Frequent occurrence of terms like "university education," "like super," and "valuable insights" hints at a formal and instructional tone.
  - Specific combinations such as "judge richard" and "geological processes" may suggest focus on legal or scientific topics.
- **Human-Exclusive:**
  - Distinctive pairs like "luke merger" and "anxious web" reveal potential themes related to mergers or online experiences unique to human-authored essays.

Top 10 2-grams only used by ai:

	2-gram	freq
0	university education	8166
1	like super	6548
2	judge richard	6338
3	geological processes	5524
4	respiratory problems	5401
5	address city	5385
6	state zip	5189
7	valuable insights	5038
8	senator's name	4646
9	sustainable future	4563

Top 10 2-grams only used by human:

	2-gram	freq
0	luke merger	3425
1	huang predicts	2759
2	the unrwa	2741
3	team snapped	2535
4	over colonia	2415
5	anxious web	2266
6	on paragraph	2047
7	paul beckman	2030
8	its self	1958
9	pop icon	1934

3-grams:

- **AI-Exclusive:**
  - Structured phrases like "sincerely your name" and "writing to express my" suggest standardized letter formats commonly found in AI-generated content.
  - Prominent inclusion of specific topics such as "respiratory problems" and "a tour guide" may reflect thematic preferences.
- **Human-Exclusive:**
  - Longer phrases like "snapped a picture" and "thousands of anxious" hint at narrative storytelling or descriptions of events characteristic of human writing styles.

Top 10 3-grams only used by ai:

	3-gram	freq
0	sincerely your name	16549
1	writing to express	11987
2	today to express	6605
3	express my support	6365
4	judge richard a	5887
5	address city state	5267
6	city state zip	5173
7	support for abolishing	5092
8	a tour guide	5080
9	of our democratic	4469

Top 10 3-grams only used by human:

	3-gram	freq
0	snapped a picture	2947
1	main and his	2528
2	dr huang predicts	2484
3	team snapped a	2259
4	of anxious web	2079
5	anxious web surfers	2064
6	thousands of anxious	2057
7	flew over colonia	2040
8	dr paul beckman	1890
9	are the clouds	1808

4-grams:

- **AI-Exclusive:**
  - Consistent use of phrases like "writing to express my" and "express my support for" indicates a structured and formulaic approach in AI-generated essays.
  - Inclusion of terms like "to reduce traffic congestion" and "college and electing the" may point to discussions of societal issues.
- **Human-Exclusive:**
  - Unique combinations like "the text it states" and "team snapped a picture" suggest detailed discussions or narratives often found in human-authored essays.

Top 10 4-grams only used by ai:

	4-gram	freq
0	writing to express my	11550
1	am writing to express	11164
2	ensures that the president	9535
3	you today to express	6478
4	today to express my	6341
5	express my support for	6180
6	to express my support	6056
7	to reduce traffic congestion	5686
8	college and electing the	5180
9	name i am writing	5094

Top 10 4-grams only used by human:

	4-gram	freq
0	the text it states	2328
1	been the most earthlike	2271
2	team snapped a picture	2164
3	snapped a picture ten	2127
4	people of other countries	2104
5	michael main and his	2025
6	aware of people of	1938
7	anxious web surfers were	1932
8	of people of other	1910
9	still has some features	1907

5-grams:

- **AI-Exclusive:**
  - Repetition of structured expressions like "i am writing to express" and "the electoral college and electing" indicates a template-driven language in AI-generated essays.
  - Consistent themes such as "ensures that the president is elected" suggest a focus on political processes.
- **Human-Exclusive:**
  - Complex narratives like "snapped a picture ten times" and "been the most earthlike planet" showcase the descriptive and contextual richness often found in human writing.

Top 10 5-grams only used by ai:

	5-gram	freq
0	i am writing to express	11154
1	am writing to express my	10954
2	that the president is elected	9061
3	to you today to express	6455
4	you today to express my	6270
5	ensures that the president is	6116
6	to express my support for	5943
7	the electoral college and electing	5278
8	electoral college and electing the	5180
9	college and electing the president	5178

Top 10 5-grams only used by human:

	5-gram	freq
0	been the most earthlike planet	2162
1	in the text it states	2047
2	snapped a picture ten times	2021
3	have been the most earthlike	1937
4	of other countries and their	1912
5	people of other countries and	1902
6	of people of other countries	1856
7	of anxious web surfers were	1825
8	anxious web surfers were waiting	1817
9	aware of people of other	1783

In this section, we used word tokenizer and sentence tokenizer to extract specific information from the text data, such as tokens, number of characters, number of words, number of sentences, and number of unique words. These extracted features are added as additional attributes to the dataset. This step aims to facilitate further Exploratory Data Analysis (EDA) and analysis of the data. However, it is important to note that this section is solely applied for EDA and data analysis purposes and will not be utilized in the subsequent learning process.

	prompt_id	text	label	tokens	num_characters	num_words	num_unique_words	num_sentences
0	0	Cars. Cars have been around since they became ...	0.0	[Cars, ., Cars, have, been, around, since, the...	3289	658	291	23
1	0	Transportation is a large necessity in most co...	0.0	[Transportation, is, a, large, necessity, in, ...	2738	526	257	27

Fig4. The distribution of unique words written by both humans and AI follows a similar pattern. However, the notable difference lies in the fact that humans exhibit a greater capability in generating texts with a higher diversity of unique words compared to AI.

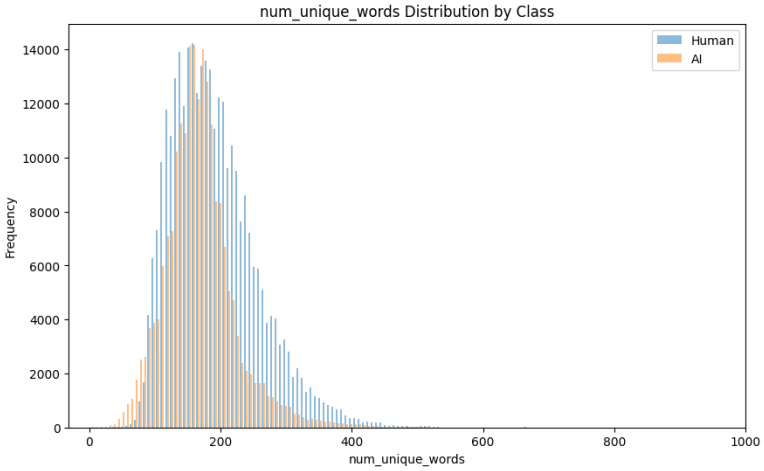


Fig5. The frequency distribution of words in texts generated by humans and AI is not identical. Specifically, the distribution for AI appears to be almost normal, while the distribution for human-generated text exhibits a left-skewness. In other words, it seems that human-generated texts include fewer words below a certain threshold, whereas this does not hold true for AI-generated texts. Humans demonstrate a greater ability to write longer texts compared to AI.

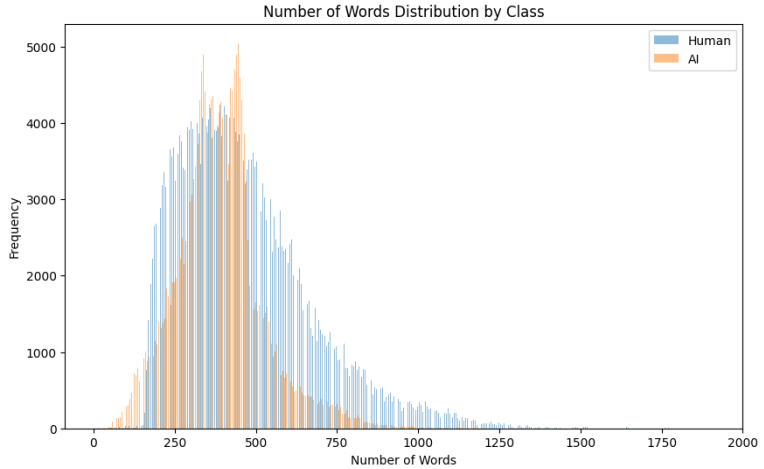


Fig6. For the distribution of the number of sentences used and their frequencies, an observed behavior closely resembles the descriptions provided in Fig4

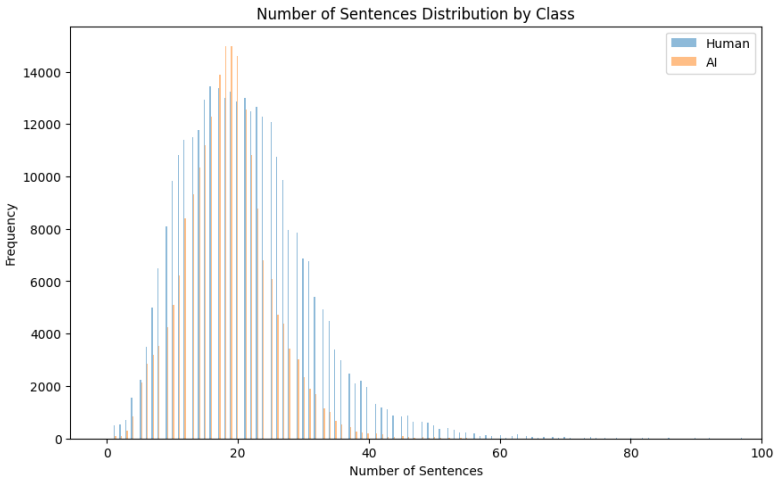
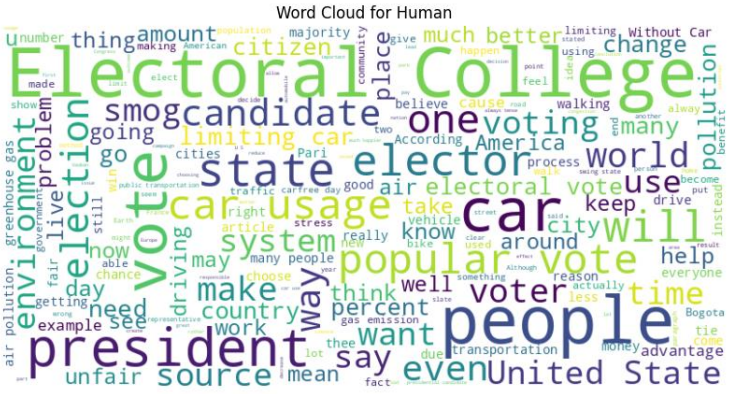
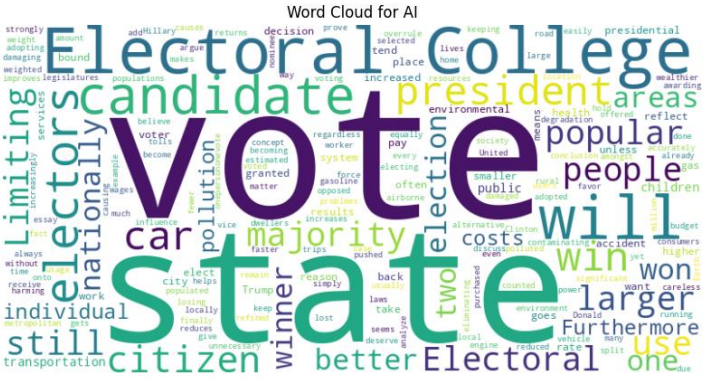


Fig7. Analyzing the word cloud generated for human-generated text reveals prominent terms such as "electoral college," "president," "people," "car," and "vote." Notably, the word cloud captures key topics related to politics, leadership, and public engagement. It's important to mention that due to the considerable time required for generating the word cloud, the analysis is based solely on the original dataset without any additional preprocessing or augmentation.



The prevalence of terms like "electoral college" and "president" suggests a focus on political discourse, possibly discussing elections, voting, and public participation. The inclusion of more general terms like "people" and "car" may indicate a diverse range of topics within the human-generated text.

Fig8. Analyzing the word cloud for AI-generated text reveals prominent terms such as "vote," "state," "electoral college," "will," "candidate," and "president." The prevalence of these terms suggests a thematic focus on electoral processes, candidates, and presidential topics within the AI-generated text. The inclusion of terms like "state" further emphasizes a connection to political contexts.



The recurring presence of terms related to voting, electoral college, and presidential candidates indicates a consistent theme in the AI-generated text, aligning with political discourse.

Data Preprocessing

Following all these steps, it's now time for data and text preprocessing. The goal is to derive suitable features and their primary context to enable neural networks and classical algorithms to comprehend them effectively, aligning with the insights gained during the Exploratory Data Analysis (EDA) phase.

The **preprocess text** function serves as a comprehensive text preprocessing tool designed to enhance the quality and relevance of textual data for subsequent analysis or modeling. Upon application, this function orchestrates a series of transformative steps:

1. **Lowercasing:**
    - All characters within the text are converted to lowercase. This ensures uniformity and mitigates discrepancies arising from case variations.
2. **URL Removal:**
    - Any URLs present in the text are systematically removed. This step is crucial for eliminating potentially noisy and irrelevant information often found in web-based content.
3. **Special Character and Number Removal:**
    - Special characters and numerical digits are filtered out, leaving behind only alphabetic characters and spaces. This step aids in streamlining the text and removing non-semantic elements.
4. **Tokenization - Sentence and Word:**
    - The text is tokenized into sentences, and each sentence is further tokenized into individual words. This hierarchical tokenization structure facilitates a more granular analysis of the text.
5. **Stop Word Removal:**
    - Common English stop words, which typically do not contribute significant meaning, are removed. This step refines the text by retaining only content-rich words.
6. **Lemmatization:**
    - Words are lemmatized to reduce inflected forms to their base or root form. This not only standardizes the vocabulary but also aids in capturing the core meaning of words.
7. **Rejoining Tokens:**
    - The processed tokens are rejoined into a coherent and preprocessed text. This final output represents a refined and standardized version of the original text, ready for downstream tasks.

Changes post-application:

- **Normalization:** The text is normalized to lowercase, reducing potential discrepancies due to case variations during subsequent analysis.
- **Noise Reduction:** URLs, special characters, and numbers, often considered as noise, are eliminated, enhancing the semantic clarity of the text.
- **Semantic Refinement:** Stop words are removed, and words are lemmatized, contributing to a more semantically meaningful representation of the text.
- **Structural Hierarchy:** Tokenization into sentences and words introduces a hierarchical structure, enabling a more nuanced exploration of the text's linguistic nuances.

In the end, the modified text was stored in a column named "processed_text_combined" for future use. The data was then divided into portions of 20% for testing and 80% for training, allowing a gradual start to the learning process. It ensures that changes can be applied to the train or test data without the occurrence of data leakage. An important note is that this data split is performed in a way that includes both label 0 and label 1 data in both the test and train sets, and this division is randomized. Additionally, the impact of adding supplementary data was influential in achieving a nearly balanced dataset during this process.

We tokenize and index processed text from both training and test datasets, unifying them into a comprehensive list. Utilizing torchtext, a vocabulary is then constructed, mapping unique tokens to numerical indices.

Subsequently, the processed text undergoes transformation into sequences of indices. Leveraging the constructed vocabulary, each token in the processed text is replaced with its corresponding numerical index using torch's tokenizer. This results in sequences of indices for both the training and test datasets, providing a numerical representation that can be directly utilized by machine learning models for further analysis and training.

To ensure uniformity in sequence length for model input, the maximum length among all sequences is determined. Subsequently, the sequences in both the training and test datasets are padded to match this maximum length using PyTorch's **pad_sequence** and **F.pad**. This step is crucial for maintaining consistency in input dimensions and facilitating efficient batch processing during model training and evaluation.

The labels are converted into tensors of type **torch.float32** for both the training and test datasets. Subsequently, a DataLoader is created, wrapping the training data and labels into a **TensorDataset** and then organizing it into batches using **DataLoader**. This DataLoader, configured with a batch size of 64 and shuffling enabled, facilitates efficient iteration over the training data during the model training process.

Training

- ## Simple Neural Network Model (Fully Connected Layers)

  We have reached the final phase of the task, where we will begin by training a neural network on this data. To start, we will design a simple neural network consisting of an autoencoder and a classifier in parallel. The purpose of this network is to provide a latent representation for the input data and utilize this representation for the classification task concurrently.

  This model employs mean squared error for the autoencoder section and binary cross-entropy loss for the classification task. The final loss is calculated as the sum of the classifier loss and the autoencoder loss. The Adam optimizer is utilized with a learning rate of 3e-4. The model is trained with these hyperparameters, and the evolution of results, accuracy, as well as metrics such as F1 score, recall, precision, and AUROC, is monitored over 5 epochs.

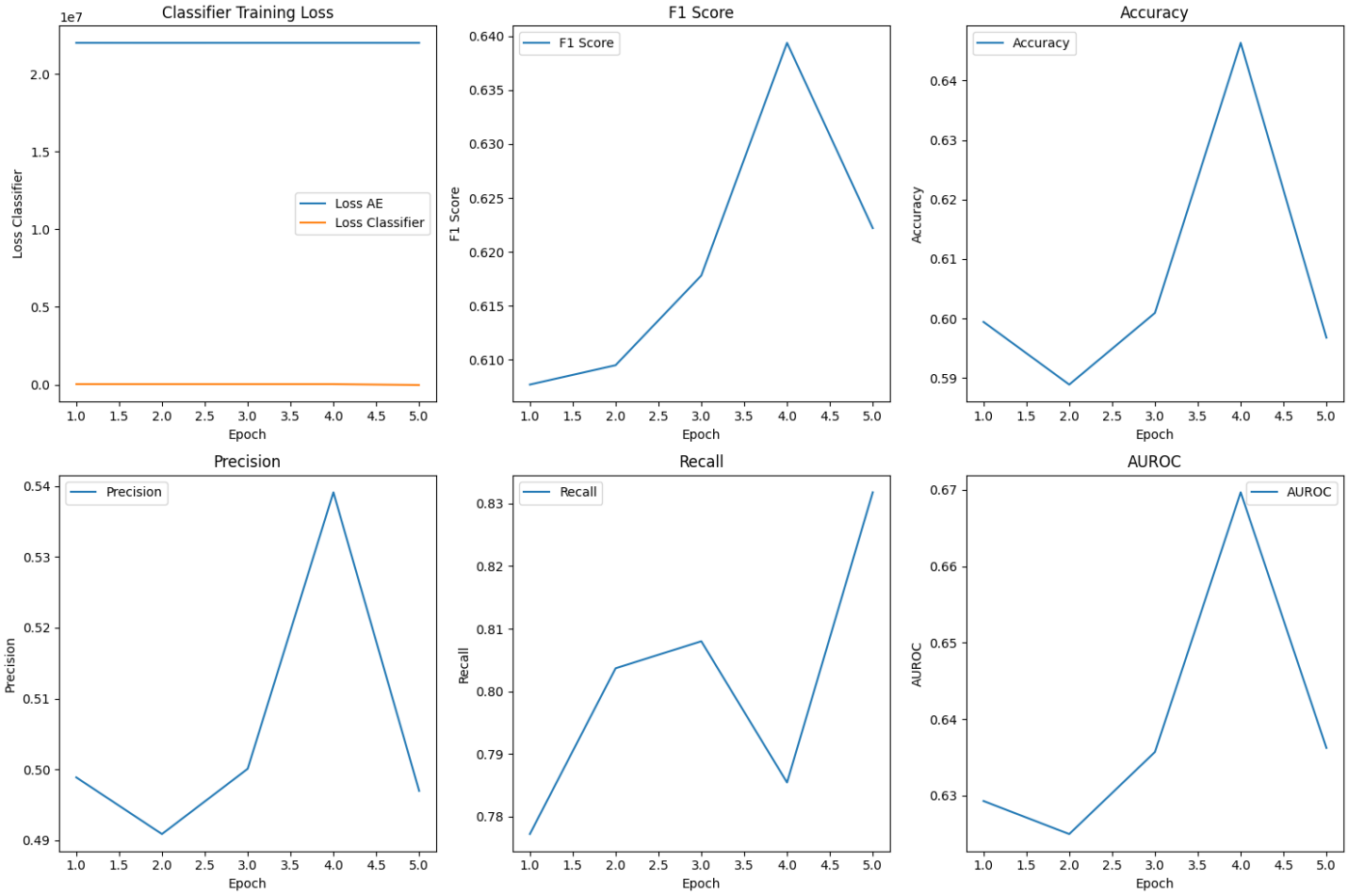


Fig9. As evident from the charts, the loss remains relatively unchanged, and the other scores exhibit significant fluctuations. This implies that the current model struggles to effectively solve the task and capture relationships within the sequential data. Consequently, it is necessary to design a more suitable model for feature extraction and handling sequential data relationships.

Before delving into the discussion of the two alternative models designed for this purpose, it's important to note that the AUROC score for the test data is 0.6248574514555735.

- ## Simple LSTM Model

  The first model is a simple LSTM model, consisting of an embedding layer followed by two LSTM layers, and finally, a fully connected layer.

- ## LSTM + Transformer + CNN Model

  This architecture combines a Transformer-based and LSTM-based neural network for text classification tasks. Here's a simplified explanation:

  Embedding Layer:

  - Converts input tokens into dense vectors of a specified dimension (embedding_dim).

  Bidirectional LSTM Layer:

  - Captures sequential patterns in the embedded vectors bidirectionally.

  Transformer Block:

  - Enhances the sequential understanding using a Transformer block with multi-head self-attention and feedforward layers.

  1D Convolution Layer:

  - Learns hierarchical features from the sequential data using a 1D convolutional layer.

  Global Max Pooling:

  - Extracts the most significant features from the convolutional layer across the entire sequence.

  Dense Layers:

  - Applies fully connected layers for further feature transformation.

  Dropout Layer:

  - Introduces regularization by randomly dropping a fraction of connections during training.

  Sigmoid Activation:

  - Outputs a probability score between 0 and 1 using a sigmoid activation function, indicating the likelihood of the input text belonging to a certain class.

Unfortunately, due to the lack of suitable resources for training this model, the results are not available for analysis. However, based on the results from other parts of the competition code, it can be inferred that this model would likely achieve scores well above 90, and even a training duration of only 5 epochs would be sufficient. Extensive efforts were made to train this model, but with the GPU provided by Kaggle, each epoch took approximately 1 hour, and the process crashed midway through the training. As a result, no results are available for this model.

- **Classic ML Algorithms**

  Now it's time to fit the data on classical machine learning algorithms and examine the results. For this purpose, we have utilized random forest, logistic regression, and gradient boosting algorithms. The results are presented in the form of a chart for AUROC. To make optimal use of the data, after the aforementioned processes, the TF-IDF algorithm has also been applied in this section. This is done to provide the models with a general understanding of the overall information and context of the text data, as classical models do not inherently grasp sequential data.

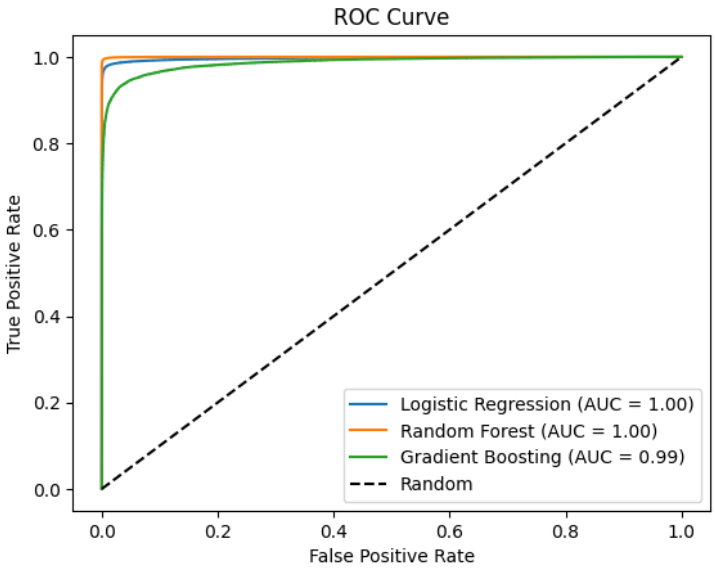


Fig10. The obtained results clearly demonstrate the effectiveness of these models and the processes undertaken throughout this task. The results are highly promising, with impressive ROC-AUC scores for the three models:

1. Logistic Regression: 0.9971
2. Random Forest: 0.9999
3. Gradient Boosting: 0.9868

These scores indicate excellent performance in classification tasks, suggesting that the classical machine learning algorithms, coupled with the applied preprocessing steps, have yielded highly accurate predictions on the given dataset.

Conclusion –

In conclusion, the research successfully navigated the complexities of classifying sequential data through a comprehensive analysis of both deep learning and classical machine learning methodologies. While resource limitations hindered the in-depth exploration of the LSTM model, the classical algorithms, particularly logistic regression, random forest, and gradient boosting, demonstrated exceptional proficiency in achieving accurate classifications. The findings underscore the importance of considering both deep learning and classical approaches in addressing sequential data challenges, with classical algorithms proving to be particularly robust and effective in this specific context.

References –

[1] Brownlee, J. (2021). "How to Develop a Deep Learning Bag-of-Words Model for Predicting Movie Review Sentiment." Machine Learning Mastery.

[2] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). "Scikit-learn: Machine learning in Python." Journal of Machine Learning Research, 12(Oct), 2825-2830.

[3] Chollet, F., et al. (2015). "Keras: The Python Deep Learning library." GitHub repository, https://github.com/fchollet/keras.

[4] Breiman, L. (2001). "Random forests." Machine learning, 45(1), 5-32.

[5] Friedman, J. H. (2001). "Greedy function approximation: A gradient boosting machine." Annals of statistics, 1189-1232.

[6] Hochreiter, S., & Schmidhuber, J. (1997). "Long short-term memory." Neural computation, 9(8), 1735-1780.

[7] Pennington, J., Socher, R., & Manning, C. (2014). "Glove: Global vectors for word representation." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

[8] Kingma, D. P., & Ba, J. (2014). "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980.

[9] Scikit-learn: Gradient Boosting. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/ensemble.html#gradient-boosting

[10] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). "Distributed representations of words and phrases and their compositionality." In Advances in neural information processing systems (pp. 3111-3119).

[11] Jolliffe, I. T. (2002). "Principal component analysis." Wiley Online Library.

[12] Davis, J., & Goadrich, M. (2006). "The relationship between Precision-Recall and ROC curves." In Proceedings of the 23rd international conference on Machine learning (pp. 233-240).

[13] Géron, A. (2019). "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems." O'Reilly Media.