Real or GPT-2?

Fake News Detection

December 2020

By: Huong Doan and Minh-Tuan Nguyen

Table of contents

Summary

Social media scrolling is a real addiction, especially during the Covid-19 pandemic. According to a Statista piece, 3.6 billion people are using social media in 2020 worldwide. The scary part is that bad actors are exploiting social media prominence as a channel for spreading disinformation (deliberately deceptive). With foreigh interference seeking to undermine our system, disinformation is considered a national security concern. If that was not bad enough, unaware users are also contributing to the problem by unknowingly spreading misinformation (false or inaccurate information regardless of the intention).

With the release of GPT-2 in 2018 and the recently announced GPT-3, bad actors now have complex tools at their disposal to deliberately deceive readers. GPT-2 uses 1.5B parameters and a short thematic input to generate convincingly human-like text while GPT-3 uses 175B.

Our goal for this project is to solve a subset of disinformation by attempting to detect whether a text was generated by GPT-2. Using the real and GPT-2 generated texts from OpenAI's GPT-2 Output GitHub, we created a dataset of 100,000 texts for our detector model.

While there are several detector models out there like the one created by Hugging Face (https://huggingface.co/openai-detector/) based on its Transformers implementation of RoBERTa (a modified BERT's language model) and Giant Language model Test Room (GLTR - http://gltr.io/dist/index.html) which is a statistical analysis of language models. A probabilistic approach was actually what we originally hypothesized before finding out that GLTR already existed. The idea is that words generated from a language model come from that model probability distribution so what we need to do is check if our texts follow GPT-2's distribution.

However, we called an audible and went in a different direction. Our selected methodology for detecting fake news generated by GPT-2 has 3 phases: feature input, prediction models and performance metrics. In our experiment, the neural network model performanced better than our decision tree model for both methods (PV-DM and PV-DBOW). Our PV-DBOW neural network has an accuracy rate of 0.78307 with F1-score, Precision and Recall slightly off from 0.78.

Though not above 90%, the result is acceptable considering GPT-2 was created to emulate human-like texts. This makes us wonder about the challenges of predicting texts generated by GPT-3 which, in its full version, has a capacity of 175 billion machine learning parameters. GPT-3 is currently not open source but can be requested from OpenAI.

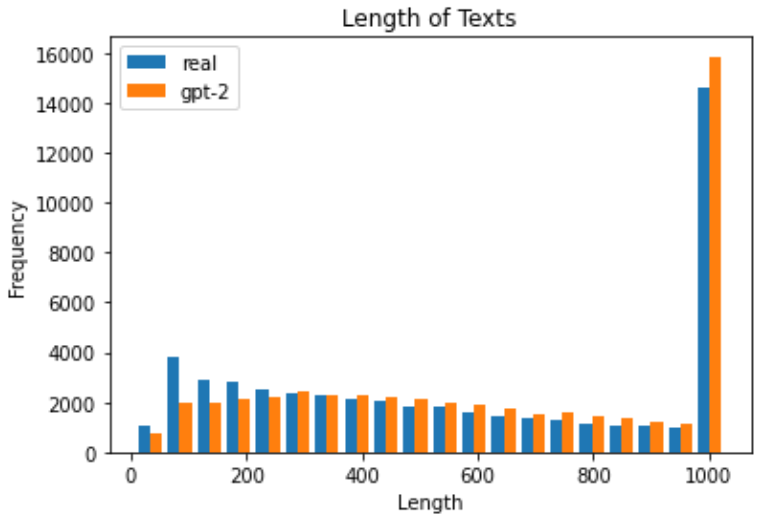
Data Collection and Processing

Extracting the datasets from OpenAI GitHub:
https://colab.research.google.com/drive/1Z_drJZXJPub18qoAVLe7Keno4a8M_HyF?usp=sharing
Data Exploration and Preprocessing:
https://colab.research.google.com/drive/15X9bnbG9Kn1yuQbuzJ7ClufNeIOI7_Z-?usp=sharing

One of the most tedious parts of our project was data collection and preprocessing. Working with text data is always time consuming as there are many issues, like slangs, acronyms, special characters, numbers, languages, etc.

Our dataset consisted of data from two json files on OpenAI's GPT-2 Output GitHub: webtext (real texts) and xl-1542M-K40 (GPT-2 texts) datasets. The xl-1542M-K40 texts were generated by GPT-2 using the webtext as a training set. Each json file had 3 datasets (train, valid and test sets). We combined the 3 datasets from each json file before randomly extracting 50,000 out of 260,000 observations. The final data collection step in our process was combining the two datasets of 50,000 observations into one full dataset and mapping the correct label (real or GPT-2) to their respective observations.

Exploring our newly created raw dataset, we saw that the majority of our texts (over 30,000 observations) had length around 1000. When comparing the length of texts of real and GPT-2 observations, we noticed that the frequency of GPT-2 texts for length above 400 was higher than that of real texts.



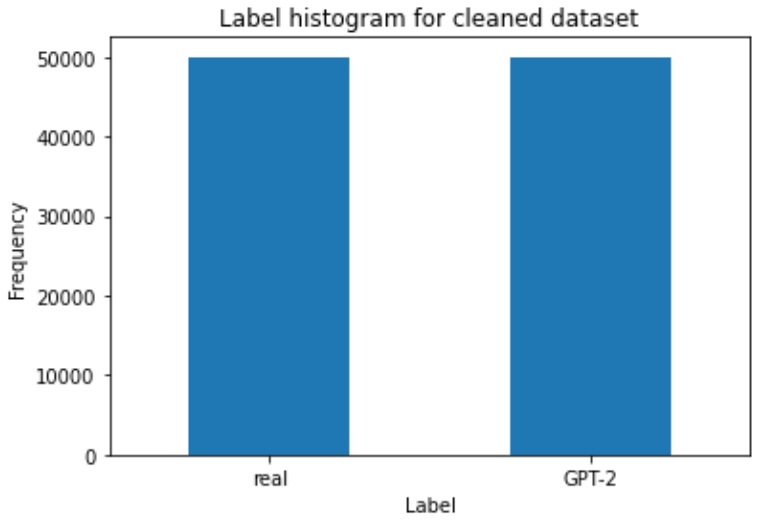
Below is the table of the top 10 words used in real text versus GPT-2 text:

Real Word	Value	GPT-2 Word	Value
say	92511	say	135489
one	58671	make	80508
make	51548	one	74148
year	49847	get	68136
user	46603	go	66408
get	43994	use	64686
time	42669	people	63756
go	39914	also	63169
also	39289	year	57455
like	38799	time	54050

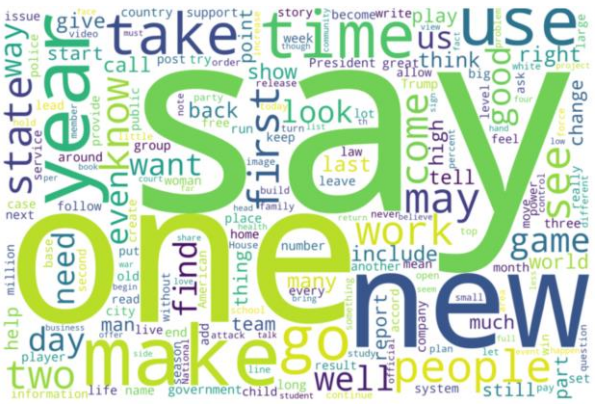
Our raw dataset was far from clean suffering from a lot of common text data issues: url links, special characters, numbers, html code, etc. Below are the steps we took during preprocessing:

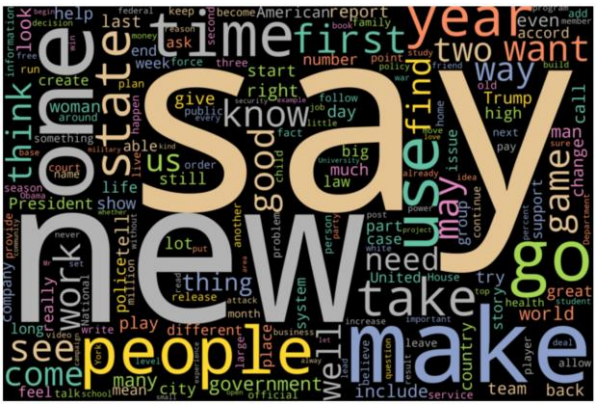
- Remove linebreaks, wrapping and other text formats
    - Replace “\n” with a blank space
    - Add space before and after square brackets
    - Replace anything that start with “[“ and end with “]” with a blank space
- Remove url links
    - Use `'(http\S+|\.gl\S+|www\S+)'` to replace url links with a blank space
- Text Normalization via Lemmatization
    - Group together the inflected forms of a word so that we can analyze them as a single item/word
- Remove brackets, extra spaces, special characters and digits
    - Use `"[^a-zA-Z]"` and `'\s+'` to remove special characters and digits
- Remove stopwords
    - Use “would” in conjunction with stopwords from nltk to remove all stopwords
- Remove words with length of 1
- Remove missing data

Our final dataset after preprocessing has 49,997 real texts and 49,986 GPT-2 texts.



Looking at the below word cloud representation of our real (left) and GPT-2 (right) texts, we can see the similarities. This was expected since the real texts were used to generate GPT-2 texts.



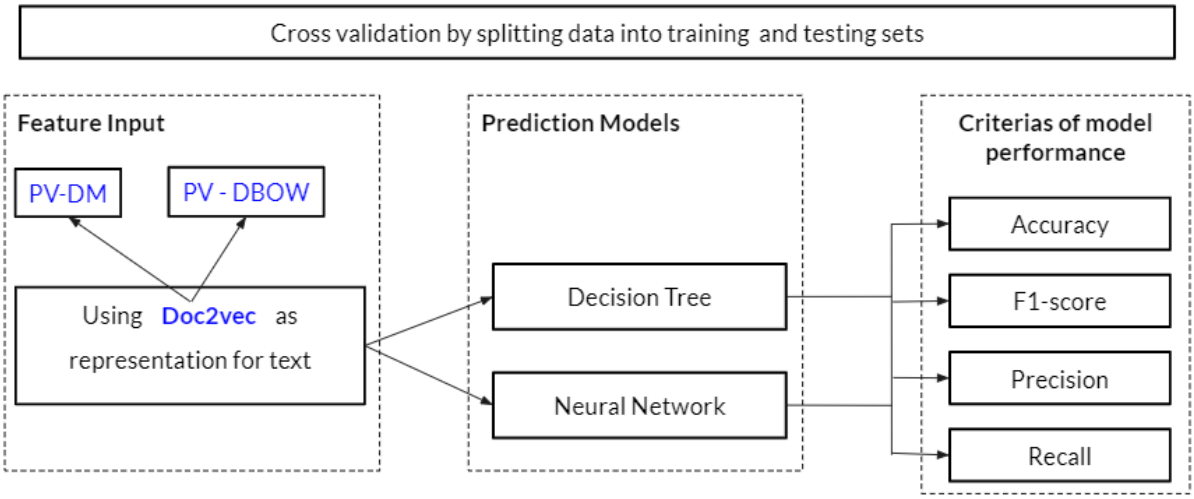


Below is a snapshot of our dataset:

	id	ended	length	text	type	label	tidy_text
0	47018	True	102	The overarching quality of the Bloomberg era w...	train	real	overarching quality Bloomberg era change marry...
1	94338	True	142	This is about as bad as my morning commute get...	train	real	bad morning commute get winter long minute wai...
2	44507	True	631	More on Saskatchewan Liquor Privatization\n\nM...	train	real	Saskatchewan Liquor Privatization late planet ...
3	163493	False	1024	Killen was here\n\nPatrick Killen helped defin...	train	real	Killen Patrick Killen help define modern archi...
4	91925	True	51	Mailbox Rental & Mail Forwarding\n\nServices i...	train	real	Mailbox Rental Mail Forwarding Services Delawa...
...	...	...	...	...	...	...	...
99978	88658	True	548	The official website for Donten ni Warau TV 's...	train	GPT-2	official website Donten ni Warau TV Maken ki t...
99979	7575	True	803	I have a long, slow, and somewhat unproductive...	train	GPT-2	long slow somewhat unproductive relationship G...
99980	38428	False	1024	"Flexibility Is Not an Option"\n\nAt The Nouri...	train	GPT-2	flexibility option Nourish School use plant ba...
99981	129005	True	137	\nDuck Dynasty's Phil Robertson will have to m...	train	GPT-2	Duck Dynasty Phil Robertson make due reality t...
99982	84470	False	1024	The last time I wrote at length about the poli...	train	GPT-2	last time write length political difference le...

Methodology

Doc2Vec and Prediction models: https://colab.research.google.com/drive/1p-oowNZmzwruGpGYJ9q2lpWqPQwcpQNv?usp=sharing

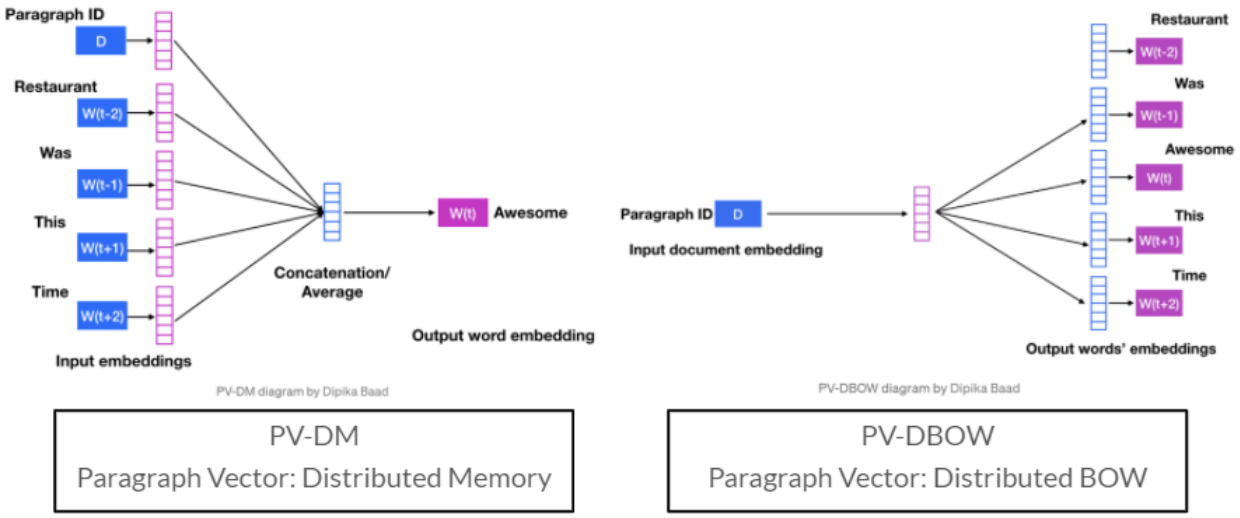


Our methodology for detecting fake news generated by GPT-2 had three phases: feature input, prediction models and performance metrics. However, before we dove into the approach, we implemented cross-validation by splitting data into training and test sets, 80% and 20% respectively. This resulted in 79,986 observations for the training set and 19,997 observations for the test set.

For our feature input, we used Doc2Vec as representation for our texts. Doc2Vec has two methods: PV-DM (paragraph vector: distributed memory) and PV-DBOW (paragraph vector: distributed bag of words). In this project, we compared both methods using a decision tree and neural network. PV-DM is similar to the continuous bag-of-words model in Word2Vec, consecutive words from a paragraph are randomly sampled then the model tries to predict a center word from the randomly sampled set of words by taking the context words and paragraph_id into account. While PV-DBOW is the Skip-Gram version of Word2Vec with paragraph_id as an additional parameter. Essentially, it utilizes a paragraph vector and the paragraph_id to classify all the words in our texts. In both methods, we first tagged the documents with the correct label (real or GPT-2) before building the Doc2Vec model and the final vector feature for the classifier.

Below are some of the differences between the two methods:

- PV-DM predicts 1 word from 4 inputs; PV-DBOW predicts 4 words from 1 input.
- PV-DM draws words from the surrounding words of the target word; PV-DBOW draws words from the paragraph.
- PV-DBOW stores less data – only the softmax weights are stored, as opposed to both softmax weights and word vectors in PV-DM.



Upon completing Doc2Vec representation, we ran two models: Decision Tree and Neural Network. We selected these models due to their differences. Decision trees are typically easier to interpret and implement while neural networks are complex but generally more stable.

Our decision tree classifier used entropy as a criterion with a minimum samples split of 50. Entropy controls how our decision tree splits the data and draws its boundaries. Once our decision tree classifier was trained, we predicted the response of our test dataset.

Our neural network model used “uniform” as a kernel initializer to generate tensors with a uniform distribution. The input and hidden layers used ReLU (Rectified Linear Unit) as the activation function while our output layer used Sigmoid.

Conceptually, ReLU is simple. It gives an output of x if x is positive and 0 if x is not.

$$A(x) = max(0, x)$$

We went with ReLU because it was computationally efficient while being nonlinear in nature meaning that any function can be approximated with combinations of ReLU.

Conceptually, Sigmoid has a S-shape curve because it exists between 0 and 1. This makes it applicable to our binary goal of distinguishing between real and GPT-2 texts.

$$\phi(z) = \frac{1}{1 + e^{-z}}$$

There are a myriad of hyperparameters but the gradient based optimization technique we used to optimize our MLP model was Root Mean Square Propagation (RMSProp). RMSProp uses a moving average of squared gradients to normalize the gradient of our model. Instead of treating the learning rate as a hyperparameter, our model learning rate adapts and changes over time.

Below is the mathematical explanation for RMSProp:

$$v_{dw} = \beta \cdot v_{dw} + (1 - \beta) \cdot dw^2$$

$$v_{db} = \beta \cdot v_{dw} + (1 - \beta) \cdot db^2$$

$$W = W - \alpha \cdot \frac{dw}{\sqrt{v_{dw}} + \epsilon}$$

$$b = b - \alpha \cdot \frac{db}{\sqrt{v_{db}} + \epsilon}$$

To avoid over training our neural network model, we added an early stop. With neural networks, too many epochs leads to overfitting (over training) and too few epochs leads to underfitting (under training). We implemented an early stop to halt training upon reaching a point where our model performance plateaued. Our early stop was set to monitor validation accuracy and stop when it reaches the max. Below is the code we used:

```
early_stopping_monitor = EarlyStopping(monitor='val_accuracy', mode='max',
                                       patience=3)
  MLP_model_history = MLP_model.fit(X_train_r, Y_train, batch_size=100,
                                    epochs=30, verbose=2,
                                           validation_data=(X_test_r, Y_test),
                              callbacks=[early_stopping_monitor])
```

To measure performance, we calculated accuracy, f1 score, precision and recall for our decision tree and neural network models using both methods (PV-DM and PV-DBOW). We used a weighted average for f1 score, precision and recall to calculate the results for each label (real and GPT-2). Their average was weighted by support because there was imbalance in our preprocessed dataset with real observations being slightly greater than GPT-2. Weighted average calculates the metrics of both real and GPT-2 texts and finds their average weighted by the number of their true instances.

		Predicted/Classified	
		Negative	**Positive**
Actual	**Negative**	998	0
	Positive	1	1

- Accuracy is how often our model predicts correctly.
  - $\frac{True\ Positive + True\ Negative}{Total\ Positive + False\ Positive + False\ Negative + True\ Negative}$
- F1 Score is the weighted average of Precision and Recall which is useful when we have unbalanced samples.
  - 2 x $\frac{Precision * Recakk}{Precision + Recall}$
- Precision is how our model predicts correctly when it predicts yes/positive.
  - $\frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{True\ Positive}{Total\ Predicted\ Positive}$
- Recall is how often our model predicts correctly when the actual label is yes/positive.
  - $\frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{True\ Positive}{Total\ ActualPositive}$

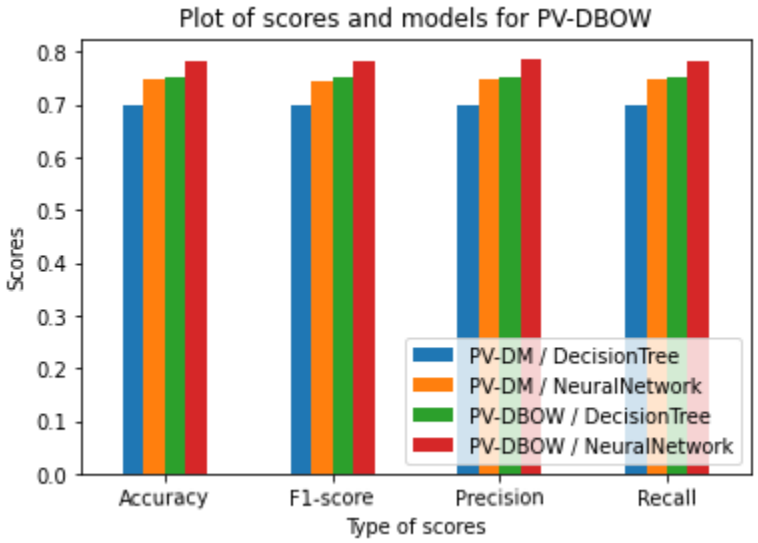
Looking at the performance table below, you can see that our neural network models performed better than our decision trees and PV-DM performed better than PV-DBOW.

	Doc2Vec Type	Model	Accuracy	F1-score	Precision	Recall
0	DV-DM	Decision Tree	0.696705	0.696599	0.696802	0.696705
1	DV-DM	Neural Network	0.745862	0.745702	0.746214	0.745862
2	DV-DBOW	Decision Tree	0.750913	0.750853	0.751002	0.750913
3	DV-DBOW	Neural Network	0.783067	0.782388	0.785953	0.783067

Conclusion

In 2020, there are 3.6 billion people using social media worldwide. That is roughly half of the world population. However, the issue is not the amount of users or accounts but rather the growing number of disinformation exploiting our scrolling habits to undermine our system and intelligence. The more advanced the AI algorithm, the greater the need for a mean to detect it.

In our effort to combat disinformation, we trained two models (decision tree and neural network) to distinguish real texts from GPT-2 texts. Our neural network model performanced better than our decision tree model, 78% and 75% respectively. The below graph provides a summary of the results of our models using both methods (PV-DM and PV-DBOW).



Future Work

Upon completion, we came up with a few next steps for future work if our interest is still pique. Below is a list of other preprocessing and parameter tuning steps we can take:

- Check languages and remove any non-English observation - We noticed after we ran our models that we had a multilingo issue. Albeit not overwhelming but removing non-English observations should improve our model performance.

	en	ca	fr	it	ja	sv	pl	no	es	fi	nl	et	de	pt	so	sw	hu	cy	ko	af	ro
language	99924	13	8	5	5	5	3	3	3	2	2	1	1	1	1	1	1	1	1	1	1

	length	text	tidy_text	label	language
18473	1024	즐거움이 가득한 보물창고, 티빙!\n\n\n1. 가장 트렌디한 예능과 기대작을 만나...	tvN Mnet ocn JTBC MBN CJ ENM pick Clips TV LTE...	real	ko
18474	1024	Het is altijd makkelijk als je even geen tijd ...	Het altijd makkelijk als je even geen tijd heb...	real	nl
27765	119	Sociology 621, Class, State and Ideology Fall ...	sociology Class State Ideology Fall Jacques Bi...	real	fr
32013	93	Tiêu đề\n\n"The mask of command": Bernard L. M...	Ti mask command Bernard Montgomery George Patt...	real	de
39727	1024	You have searched for packages that names cont...	search package name contain viper suite sectio...	GPT-2	ca
35591	432	New at SubtleTV! Close\n\nVideo: Video: NWA - ...	new subtletv close video video NWA like girl W...	GPT-2	no
55021	216	Régine Café\n\nRégine c'est la « matante » idé...	gine Caf gine est la matante id ale que nous a...	real	fr

- Tinker around with parameters of Doc2vec like window, vector-size, etc. - After further research, we realized that there were other options for optimizing our Doc2Vec model.
    - Increasing vector size adds more dimensions to our created vectors.
    - Window size adjusts the distance between the current and predicted word within a sentence. Higher window size means predicting with a broader range of words.

Parameters ->	Vector Size	Window Size	Avg. Accuracy
1	500	7	0.45
2	1000	3	0.46
3	2000	3	0.45
4	2000	5	0.46
5	7000	7	0.47
6	10000	3	0.47
7	20000	5	0.47

- Apply others machine learning models (Support Vector Machines, Bagging, Boosting, Naive Bayes, k-Nearest Neighbors, etc.) or include current detector models to our own.

Appendices

Dataset:

1. https://github.com/openai/gpt-2-output-dataset/tree/master/detector

GLTR model that we reproduced for research and testing purposes:
https://colab.research.google.com/drive/12IvnO46cBVbeTXAhxSPzRGNau3iIcjkQ?usp=sharing

Reference Link:

1. Cross validation https://towardsdatascience.com/cross-validation-explained-evaluating-estimator-performance-e51e5430ff85
2. Detecting Fake News With and Without Code https://towardsdatascience.com/detecting-fake-news-with-and-without-code-dd330ed449d9
3. Full Pipeline Project: Python AI for detecting fake news https://towardsdatascience.com/full-pipeline-project-python-ai-for-detecting-fake-news-with-nlp-bbb1eec4936d
4. An Exhaustive Guide to Detecting and Fighting Neural Fake News using NLP https://www.analyticsvidhya.com/blog/2019/12/detect-fight-neural-fake-news-nlp/
5. OpenAI's GPT-2: A Simple Guide to Build the World's Most Advanced Text Generator in Python https://www.analyticsvidhya.com/blog/2019/07/openai-gpt2-text-generator-python/
6. RoBERTa https://huggingface.co/transformers/model_doc/roberta.html
7. Combine dataframe from different files https://realpython.com/python-keras-text-classification/
8. Cleaning text https://medium.com/analytics-vidhya/twitter-sentiment-analysis-b9a12dbb2043
9. Sentiment Classification https://medium.com/swlh/sentiment-classification-for-reviews-using-doc2vec-660ba594c336
10. Differences between PV-DM and PV-DBOW https://joshuakyh.wordpress.com/2017/12/07/distributed-representation-of-anything/#:~:text=Differences%3A,word%20vectors%20in%20PV%2DDM.
11. Number of social network users worldwide from 2017 to 2025 https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/#:~:text=Social%20media%20usage%20is%20one,almost%204.41%20billion%20in%202025.

Related researches:

1. Fake news detection within online social media using supervised artificial intelligence algorithms https://www.sciencedirect.com/science/article/abs/pii/S0378437119317546
2. Detecting Fake News in Social Media Networks https://www.sciencedirect.com/science/article/pii/S1877050918318210
3. Release Strategies and the Social Impacts of Language Models https://arxiv.org/ftp/arxiv/papers/1908/1908.09203.pdf