FAKE NEWS DETECTION USING NATURAL LANGUAGE PROCESSING

Batch Member

510521104026: V MOHANRAJ

Phase 4

submission document- ***Development Part 2***

Project Title:Fake News detection using NLP:-

Phase 4: Feature Engineering, Model Training A nd Evaluation

Overview:

- ✓ ***Feature engineering in NLP is understanding the context of the text.***
- ✓ ***In this blog, we will look at some of the common feature engineering in NLP.***
- ✓ ***We will compare the results of a classification task with and without doing feature engineering***

Introduction:

- ***Feature engineering is one of the most important steps in machine learning. It is the process of using domain knowledge of the data to create features that make machine learning algorithms work.***
- ***Think machine learning algorithm as a learning child the more accurate information you provide the more they will be able to interpret the information well.***
- ***Focusing first on our data will give us better results than focusing only on models. Feature engineering helps us to create better data which helps the model understand it well and provide reasonable results.***
- ***NLP is a subfield of artificial intelligence where we understand human interaction with machines using natural languages.***

- ***To understand a natural language, you need to understand how we write a sentence, how we express our thoughts using different words, signs, special characters, etc basically we should understand the context of the sentence to interpret its meaning.***
- ***If we can use these contexts as features and feed them to our model then the model will be able to understand the sentence better.***
- ***Some of the common features that we can extract from a sentence are the number of words, number of capital words, number of punctuation, number of unique words, number of stopwords, average sentence length, etc.***
- ***We can define these features based on our data set we are using. In this blog, we will use a Twitter data set so we can add some others features like the number of hashtags, number of mentions, etc. We will discuss them in detail in the coming sections.***

- ***Fake news detection using Natural Language Processing (NLP) is a critical application of machine learning and text analysis techniques.***
- ***In this context, feature engineering, model training, and evaluation play a vital role in building effective models to distinguish between genuine information and deceptive or misleading content***
- ***. This introductory overview provides insight into these three key components of the fake news detection process:***

1. ***Feature Engineering:***

In fake news detection using NLP, feature engineering involves extracting relevant linguistic and content-based features from text data to help the model discriminate between real news and fake news. Some common feature engineering techniques include:

- ***Text preprocessing: Cleaning and tokenizing the text data, handling capitalization, punctuation, and special characters.***
- ***Text representation: Converting the text into numerical features, commonly using methods like TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings (e.g., Word2Vec, GloVe).***
- ***Sentiment analysis: Incorporating sentiment scores to gauge the emotional tone of the text.***
- ***N-grams and bag-of-words: Considering word or character combinations to capture contextual information.***
- ***Source and author credibility: Incorporating information about the source or author of the news article.***

2. ***Model Training:***

Model training involves selecting an appropriate machine learning or deep learning model and training it using labeled datasets containing both real and fake news samples. Common steps in model training include:

- ***Model selection: Choosing the right model architecture for the task. Common choices include recurrent neural networks (RNNs), convolutional neural networks (CNNs), or transformer-based models like BERT.***
- ***Data splitting: Splitting the dataset into training and testing sets to assess the model's performance.***

- ***Data balancing: Handling class imbalances to ensure the model doesn't become biased towards the majority class (usually real news).***
- ***Hyperparameter tuning: Adjusting model hyperparameters to optimize performance.***
- ***Transfer learning: Leveraging pre-trained NLP models (e.g., BERT, GPT-3) and fine-tuning them on the specific task of fake news detection.***

3. ***Model Evaluation:***

Model evaluation is essential to determine how well the trained model can discriminate between real and fake news. Evaluation in fake news detection typically involves:

- ***Performance metrics: Selecting appropriate metrics such as accuracy, precision, recall, F1 score, or area under the ROC curve (AUC) to assess the model's effectiveness.***
- ***Confusion matrix: Analyzing true positives, true negatives, false positives, and false negatives to understand where the model excels and where it struggles.***
- ***Cross-validation: Employing techniques like k-fold cross-validation to ensure the model's robustness and generalization to unseen data.***
- ***Real-world testing: Evaluating the model's performance on real news articles and fake news samples to validate its practical applicability.***

✓ *In the context of fake news detection, the goal is to create a model that can identify deceptive content accurately, thereby helping to combat the spread of misinformation and protect the integrity of information sources.*

✓ *The effectiveness of the model hinges on the quality of feature*

NLP task overview:

- **To understand the feature engineering task in NLP, we will be implementing it on a Twitter dataset. We will be using COVID-19 Fake News Dataset.**
- **The task is to classify the tweet as Fake or Real. The dataset is divided into train, validation, and test set. Below is the distribution**

Split	Real	Total	Total
Train	3360	3060	6420
Validation	1120	1020	2140
Test	1120	1020	2140

List of features:

- ***I will be listing out a total of 15 features that we can use for the above dataset, number of features totally depends upon the type of dataset you are using.***
- **In fake news detection using Natural Language Processing (NLP), feature engineering is a critical step.**
- **You need to extract and engineer relevant features from the text data to help the model identify patterns that distinguish between real and fake news.**
- **Here's a list of common features and techniques that can be useful in fake news detection:**

1. *Number of Characters:*

- ✓ **Count the number of characters present in a tweet.**
- **def count_chars(text):**
- **return len(text)**

2. *Number of words:*

- ✓ **Count the number of words present in a tweet.**
- **def count_words(text):**
- **return len(text.split())**

3. Number of capital characters:

- ✓ **Count the number of capital characters present in a tweet.**
- ✓ **Python Code:**

4. *Number of capital words:*

- ✓ **Count the number of capital words present in a tweet.**
- **def count_capital_words(text):**
- **return sum(map(str.isupper,text.split()))**

5. *Count the number of punctuations:*

- ➢ **In this function, we return a dictionary of 32 punctuation with the counts, which can be used as separate features, which I will discuss in the next section.**
- **def count_punctuations(text):**
- **punctuations='!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~'**
- **d=dict()**
- **for i in punctuations:**
- **d[str(i)+' count']=text.count(i)**
- **return d**

6. ***Number of words in quotes:***

- ✓ ***The number of words in the single quotation and double quotation.***
- ***def count_words_in_quotes(text):***
- ***x = re.findall("'.'|"."", text)***
- ***count=0 if x is None:***
- ***return 0***
- ***else:***
- ***for i in x:***
- ***t=i[1:-1] count+=count_words(t)***
- ***return count***

7. ***Number of sentences:***

- ✓ ***Count the number of sentences in a tweet.***
- def count_sent(text):
- return len(nltk.sent_tokenize(text))

8. Count the number of unique words:

- ✓ *Count the number of unique words in a tweet.*
- **def count_unique_words(text):**
- **return len(set(text.split()))**

9. Count of hashtags:

- ✓ **Since we are using the Twitter dataset we can count the number of times users used the hashtag.**
- **def count_htags(text):**
- **x = re.findall(r'(#w[A-Za-z0-9]*)', text)**
- **return len(x)**

10. Count of mentions:

- ➢ **On Twitter, most of the time people reply or mention someone in their tweet, counting the number of mentions can also be treated as a feature.**
  - **def count_mentions(text):**

- **x = re.findall(r'(@w[A-Za-z0-9]*)', text)**
- **return len(x)**

11. Count of stopwords:

✓ **Here we will count the number of stopwords used in a tweet.**

- **def count_stopwords(text):**
- **stop_words = set(stopwords.words('english'))**
- **word_tokens = word_tokenize(text)**
- **stopwords_x = [w for w in word_tokens if w in stop_words]**
- **return len(stopwords_x)**

12. Calculating average word length:

✓ **This can be calculated by dividing the counts of characters by counts of words.**

- **df['avg_wordlength'] = df['char_count']/df['word_count']**

13. Calculating average sentence length:

✓ **This can be calculated by dividing the counts of words by the counts of sentences.**

- **df['avg_sentlength'] = df['word_count']/df['sent_count']**

14. unique words vs word count feature:

✓ **This feature is basically the ratio of unique words to a total number of words.**

- **df['unique_vs_words'] =**
- **df['unique_word_count']/df['word_count']**

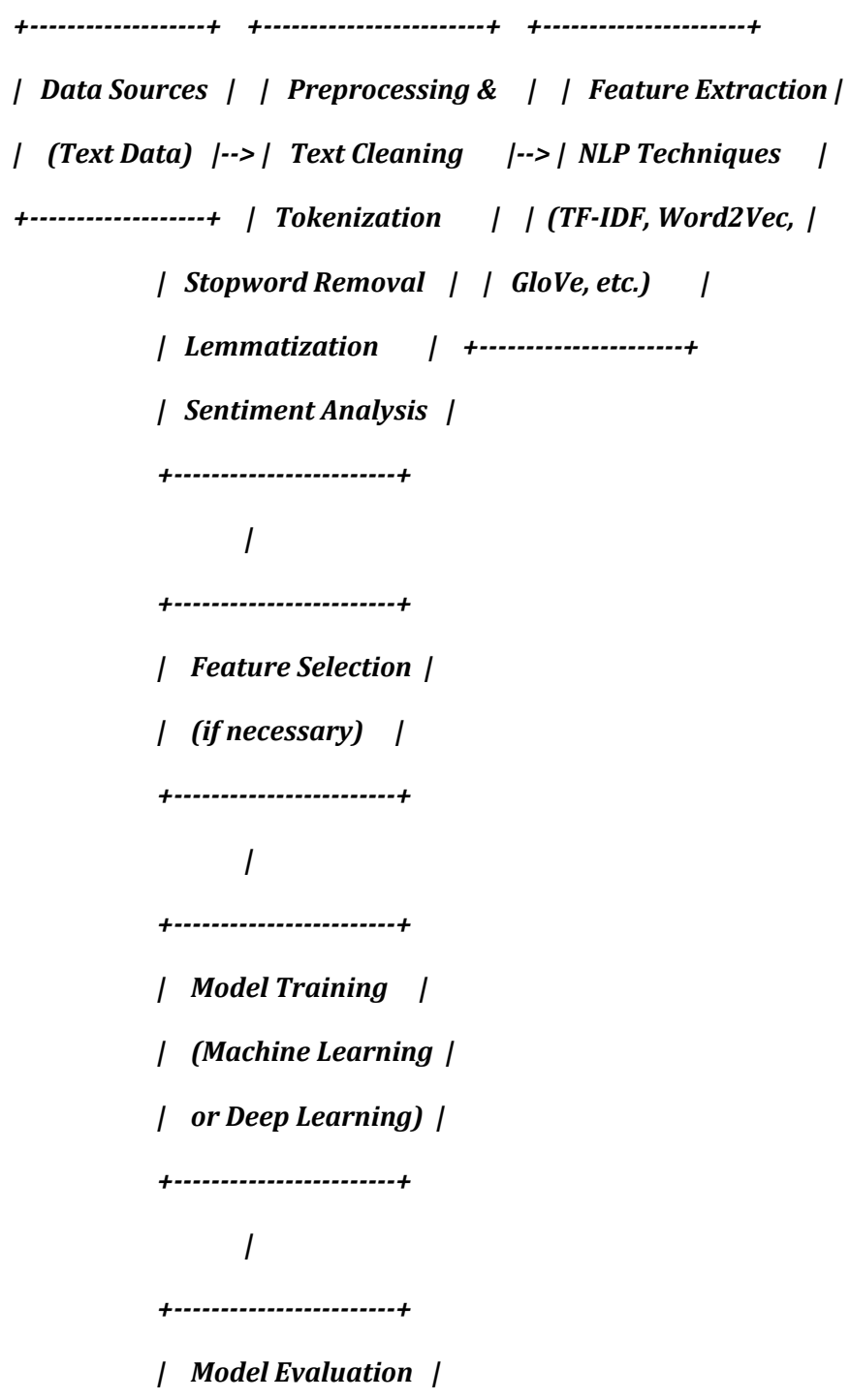
15. Stopwords count vs words counts feature:

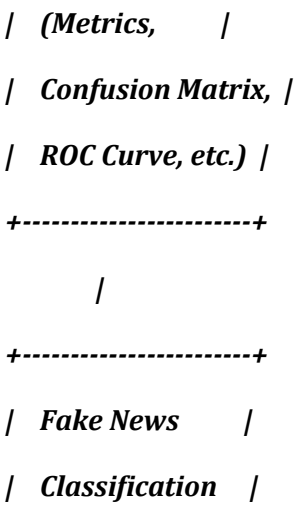
✓ **This feature is also the ratio of counts of stopwords to the total number of words.**

- **df['stopwords_vs_words'] =**
- **df['stopword_count']/df['word_count']**

Block Diagram:

- ***Creating a block diagram for fake news detection using Natural Language Processing (NLP) involves visualizing the key steps involved in the process. Below is a simplified block diagram illustrating the various components and steps in fake news detection using NLP:***





✓ **Here's a breakdown of each component in the block diagram:**

- ***Data Sources:***

  This is where you gather the text data that you'll use for fake news detection. Data sources can include social media, news articles, blogs, or any other text-based content.

- ***Preprocessing & Text Cleaning:***

  In this step, you clean and preprocess the text data. This includes tasks like removing special characters, lowercasing, tokenization, stopword removal, and lemmatization. Additionally, you might perform sentiment analysis to extract sentiment features.

- ***Feature Extraction:***

  NLP techniques are applied to convert the preprocessed text data into numerical features that can be used for machine learning. Common techniques include TF-IDF (Term Frequency-Inverse Document Frequency), Word2Vec, GloVe, and others.

- ***Feature Selection:***

  If the feature space is too large, feature selection methods can be applied to choose the most relevant features to feed into the model.

- ***Model Training:***

  Machine learning or deep learning models are trained on the extracted features. Common models for text classification include Naive Bayes, Support Vector Machines, Recurrent Neural Networks (RNNs), or Transformers like BERT.

- ***Model Evaluation:***

The performance of the trained model is assessed using evaluation metrics such as accuracy, precision, recall, F1-score, ROC curve, and confusion matrix. Cross-validation may be employed for robust evaluation.

- ***Fake News Classification:***

  The final step is using the trained model to classify news articles or text as either "fake" or "real" based on the features and patterns it has learned.

  - ***This block diagram provides a high-level overview of the fake news detection process using NLP. It's worth noting that the actual implementation may involve further refinements, multiple iterations, and specific NLP techniques and model architectures depending on the complexity of the task.***

Implementation:

- ***You can download the dataset from here. After downloading we can start implementing all features we defined above. We will focus more on feature engineering, for this we will keep the approach simple, by using TF-IDF and simple pre-processing. All the code will be available on my GitHub repository https://github.com/ahmadkhan242/Feature-Engineering-in-NLP.***

- Detecting fake news using Natural Language Processing (NLP) involves a combination of feature engineering, model training, and evaluation. Below, I'll outline a step-by-step implementation guide for this task:

1. Data Collection and Preprocessing:

- Gather a labeled dataset of news articles with labels indicating whether each article is real or fake.
- Preprocess the text data, including tasks like lowercasing, tokenization, stop-word removal, and stemming or lemmatization.

2. Feature Engineering:

- Convert the text data into numerical features that can be used for machine learning. Common techniques include:
  - TF-IDF (Term Frequency-Inverse Document Frequency) vectorization: It assigns weights to words based on their importance in the document.
  - Word embeddings (e.g., Word2Vec, GloVe): These techniques map words to dense vector representations that capture semantic information.
  - N-grams: Including combinations of words (bigrams, trigrams) as features.
- You can also extract other relevant features like the length of the article, sentiment analysis, and readability scores.

3. Model Selection:

- Choose an NLP-based machine learning model suitable for text classification. Common choices include:
    - **Multinomial Naive Bayes**: A simple yet effective choice for text classification.
    - **Logistic Regression**: A linear model that works well with text data.
    - **Random Forest, Gradient Boosting**: Ensemble methods that can capture complex relationships in the data.
    - **Recurrent Neural Networks (RNNs)**: If you want to explore deep learning, RNNs can be used to capture sequential information in the text.
    - **Transformer-based models**: State-of-the-art models like BERT, GPT-2, or RoBERTa can provide excellent performance but require substantial computational resources.

4. Data Splitting:

- Split your dataset into training, validation, and test sets. A common split might be 70% for training, 15% for validation, and 15% for testing.

5. Model Training:

- Train your chosen model on the training data using the numerical features obtained from feature engineering.
- Fine-tune hyperparameters if necessary. For deep learning models, this could include learning rate, batch size, and the number of layers.
- Monitor model performance on the validation set to avoid overfitting.

6. Model Evaluation:

- Evaluate the model's performance on the test set using appropriate metrics for binary classification, such as accuracy, precision, recall, F1-score, and AUC-ROC.
- Consider other metrics, such as confusion matrices, to understand the model's behavior more comprehensively.
- Conduct an error analysis to identify patterns in misclassifications and areas for improvement.

7. Model Tuning and Iteration:

- If the initial model does not meet your performance goals, consider model tuning, including:
    - Adjusting hyperparameters.
    - Trying different feature engineering techniques.
    - Experimenting with different NLP models.
- Iterate through model training and evaluation until you achieve satisfactory results.

8. Deployment:

- Once you have a model with acceptable performance, deploy it for real-world use, such as in a web application or API.

    - Remember that the choice of features, models, and hyperparameters may require experimentation and fine-tuning to achieve the best results in fake news detection using NLP. Additionally, keeping the model updated with new data is important to maintain its effectiveness over time.

Result comparison:

- **For comparison, we first trained our model on the above dataset by using features engineering techniques and then without using feature engineering techniques.**

- **In both approaches, we pre-processed the dataset using the same method as described above and TF-IDF was used in both approaches for encoding the text data. You can use whatever encoding techniques you want to use like word2vec, glove, etc.**

1. Feature Engineering for Fake News Detection:

    - Text Representation: Convert the raw text data into numerical features that can be used as input for machine learning models. Common techniques include TF-IDF (Term Frequency-Inverse Document Frequency) vectorization or word embeddings (e.g., Word2Vec, GloVe, or fastText).

    - Feature Extraction: Extract relevant features from the text, such as n-grams (unigrams, bigrams, trigrams), sentiment analysis scores, or linguistic features like parts of speech and named entities.

    - Preprocessing: Clean the text data by removing stop words, punctuation, and special characters. Apply lemmatization or stemming to reduce words to their base forms.

    - Additional Contextual Features: Incorporate external features or metadata, such as the source of the news, publication date, and social media engagement metrics, which can provide valuable context for fake news detection.

2. Model Training for Fake News Detection:

    - Model Selection: Choose NLP models suitable for text classification tasks, such as Naive Bayes, Support Vector Machines, or more advanced models like Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), or Transformer-based models like BERT.

- Data Splitting: Split your labeled dataset into training, validation, and test sets to ensure you can evaluate the model's performance accurately.

- Model Training: Train your selected model on the training data using the text features. Fine-tune the model's hyperparameters and architecture if necessary.

- Transfer Learning: Utilize pre-trained language models (e.g., BERT, GPT, RoBERTa) to leverage their knowledge and adapt them to the fake news detection task by fine-tuning on your dataset.

3. Model Evaluation and Result Comparison:

- Evaluation Metrics: Use appropriate evaluation metrics for binary classification, such as accuracy, precision, recall, F1-score, and the ROC AUC (Receiver Operating Characteristic Area Under the Curve). However, fake news detection often requires a balance between precision and recall due to the consequences of false positives and false negatives.

- Baseline Models: Compare the performance of your NLP model(s) against baseline models or simple rule-based systems to establish a reference point.
- Model Comparison: Train and evaluate multiple models, potentially of different types, to determine which one performs best on your specific dataset. Consider ensembling techniques, such as stacking or majority voting, to combine the strengths of multiple models.

- Cross-Validation: Employ k-fold cross-validation to ensure that the model's performance is consistent and generalizes well across different data splits.

- Real vs. Fake News: Consider incorporating a dataset of real news articles to ensure that your model isn't merely detecting characteristics specific to a source or domain, as well as to benchmark its performance.

- Result comparison is essential for choosing the best-performing model, ensuring generalization to real-world data, and fine-tuning your fake news detection system.
- It's important to remember that fake news detection is a challenging task, and the choice of features, models, and evaluation metrics may vary depending on the specific characteristics of the dataset and the problem you are tackling.

Conclusion:

- **The above results show that if we do feature engineering, we can achieve greater accuracy using classical Machine learning algorithms.**

- **Using a transformer-based model is a time-consuming and resource-expensive algorithms.**

- **If we do feature engineering in the right way that is after analyzing our dataset we can get comparable results.**

- **We can also do some other feature engineering like, counting the number of emojis used, type of emojis used, what frequencies of unique words, etc.**

- **We can define our features by analyzing the dataset. I hope you have learned something from this blog, do share it with others.**

- **Check out my personal Machine learning blog(https://code-ml.com/) for new and exciting content on different domains of ML and AI**