Data Driven Exploration: Unleashing Topic Modelling Using Python

***I.V. Dwaraka Srihith*[1], *A. David Donald*[2], *T. Aditya Sai Srinivas*[3], *G. Thippanna*[4], *P. Vijaya Lakshmi*[5]**
[1]Student, Alliance University
[2]Assistant Professor, [3]Associate Professor, [4]Professor, [5]Student,
Ashoka Women's Engineering College, Kurnool

Corresponding Author
E-mail Id: - *dwarakanani525@gmail.com*

ABSTRACT
Topic Modeling is a crucial technique in natural language processing that involves assigning topic labels to a set of text documents. The primary objective of topic modeling is to unveil the latent themes or subjects present within the textual data. This article serves as a comprehensive guide for individuals seeking to acquire knowledge on performing topic modeling using machine learning algorithms with the aid of Python. Through this article, readers will gain insights into the fundamental concepts of topic modeling, various machine learning techniques used in the process, and a step-by-step implementation using Python programming language. By the end of this article, readers will have a solid foundation in topic modeling and the necessary skills to explore and extract meaningful topics from their own text data.

Keywords:- *Topic Modeling, Python.*

INTRODUCTION

Topic Modeling is a powerful Natural Language Processing (NLP) technique that enables us to uncover hidden themes or topics within a collection of text documents. By employing algorithms to analyze the frequency and distribution of words, topic modeling aims to identify relationships between the content of a document and the underlying topics it covers. This process has gained significant attention in various domains, including information retrieval, content analysis, social media analysis, and customer feedback analysis, among others.

To successfully conduct topic modeling, we require a suitable dataset that contains text documents. Fortunately, I have come across an ideal dataset specifically curated for this purpose. You can conveniently download the dataset from the provided link, which will serve as the foundation for our topic modeling exploration.

In the subsequent sections of this article, we will delve into the intricacies of performing topic modeling with Machine Learning techniques using the Python programming language. This comprehensive guide will equip you with the necessary knowledge and practical skills to extract meaningful topics from your own text data. We will cover essential topics such as preprocessing text data, selecting appropriate algorithms, evaluating topic models, and visualizing the results.

By the end of this article, you will have a solid understanding of topic modeling principles, along with the ability to implement topic modeling workflows in

Python. Whether you are a data scientist, a researcher, or simply interested in exploring the underlying themes in your textual data, this article will serve as an invaluable resource for unleashing the power of topic modeling through a data-driven approach.

So, let's embark on this exciting journey and discover how topic modeling can reveal insightful patterns and latent topics hidden within your text documents.

TOPIC MODELLING USING PYTHON

To start this task, we initiate by importing the requisite Python libraries and the dataset.

```python
import numpy as np
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.stem.wordnet import WordNetLemmatizer
import string
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('omw-1.4')

data = pd.read_csv("articles.csv", encoding = 'latin1')
print(data.head())
```

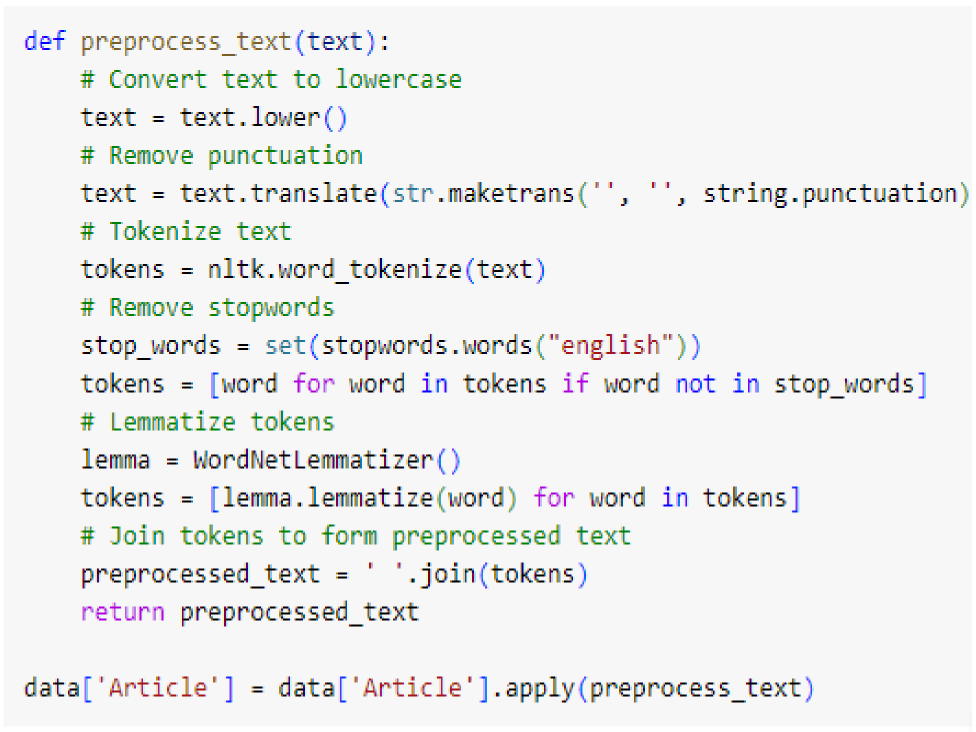
```
                                             Article  \
0  Data analysis is the process of inspecting and...
1  The performance of a machine learning algorith...
2  You must have seen the news divided into categ...
3  When there are only two classes in a classific...
4  The Multinomial Naive Bayes is one of the vari...


                                               Title
0                 Best Books to Learn Data Analysis
1         Assumptions of Machine Learning Algorithms
2          News Classification with Machine Learning
3  Multiclass Classification Algorithms in Machin...
4        Multinomial Naive Bayes in Machine Learning
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
```

In order to address the Natural Language Processing problem at hand, it is imperative to preprocess the textual content by eliminating punctuation and stopwords. The following steps outline the process of cleaning the textual data:
Removal of Punctuation: Punctuation marks such as periods, commas, and quotation marks serve as noise in the textual data. Hence, we commence the cleaning process by eliminating these punctuation marks.

Elimination of Stopwords: Stopwords are commonly occurring words in a language (e.g., "the," "is," "and") that do not carry significant meaning or contribute to the overall context of the text. These stopwords often hinder accurate topic modeling results. Consequently, we proceed by removing these stopwords from the textual data.

By performing these essential data cleansing steps, we can ensure that the textual data is optimized for subsequent topic modeling procedures, enabling us to derive more accurate and meaningful insights from the text corpus.

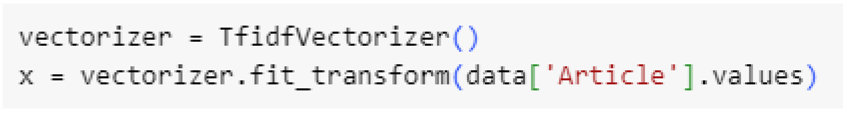


```python
def preprocess_text(text):
    # Convert text to lowercase
    text = text.lower()
    # Remove punctuation
    text = text.translate(str.maketrans('', '', string.punctuation)
    # Tokenize text
    tokens = nltk.word_tokenize(text)
    # Remove stopwords
    stop_words = set(stopwords.words("english"))
    tokens = [word for word in tokens if word not in stop_words]
    # Lemmatize tokens
    lemma = WordNetLemmatizer()
    tokens = [lemma.lemmatize(word) for word in tokens]
    # Join tokens to form preprocessed text
    preprocessed_text = ' '.join(tokens)
    return preprocessed_text

data['Article'] = data['Article'].apply(preprocess_text)
```

Next, a crucial step in our workflow is to transform the textual data into a numerical representation, facilitating the application of machine learning algorithms. To achieve this, we employ the technique of text vectorization:
Text vectorization involves the conversion of text documents into numerical vectors, where each element represents a specific feature or attribute of the text.
This numerical representation allows us to quantify and analyze the textual data using various mathematical and statistical operations.
By employing text vectorization, we can effectively capture the essential characteristics of the text, such as word frequencies or term presence, which are instrumental in uncovering the latent topics within the corpus. Furthermore, this transformation enables us to apply machine learning algorithms that require numerical inputs, facilitating the subsequent topic modeling tasks.

Through the process of text vectorization, we bridge the gap between the inherent nature of textual data and the computational capabilities of machine learning algorithms, empowering us to conduct comprehensive analyses and extract meaningful insights from the text corpus.

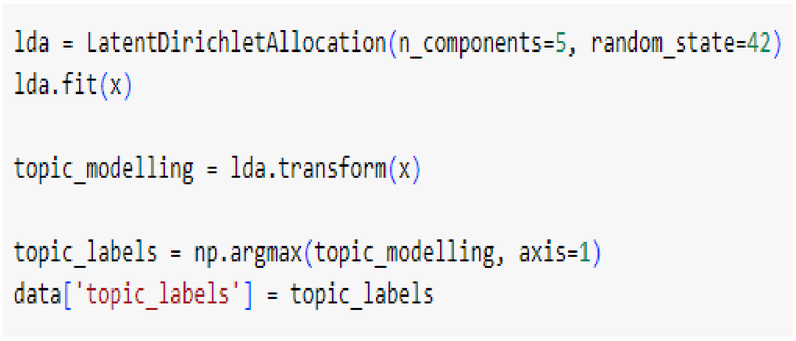


```
vectorizer = TfidfVectorizer()
x = vectorizer.fit_transform(data['Article'].values)
```

In the subsequent phase, we will leverage a sophisticated algorithm to establish meaningful relationships among the textual data and assign topic labels accordingly. For this purpose, we will employ the Latent Dirichlet Allocation (LDA) algorithm.
Latent Dirichlet Allocation (LDA) is a prominent generative probabilistic algorithm specifically designed to unveil the latent topics inherent within a textual corpus. By probabilistically modeling the co-occurrence patterns of words across documents, LDA enables us to identify underlying topic distributions and assign topic labels to the documents.

Utilizing the LDA algorithm in our topic modeling workflow empowers us to extract valuable insights and gain a comprehensive understanding of the thematic structure within the corpus. The algorithm facilitates the automatic discovery of topics by iteratively allocating words to topics and topics to documents based on their statistical distributions.

By employing the LDA algorithm, we can effectively assign topic labels to the textual data, allowing us to explore and analyze the composition of the corpus in a more structured and interpretable manner. This approach enhances our ability to derive valuable knowledge and make informed decisions based on the discovered topics.

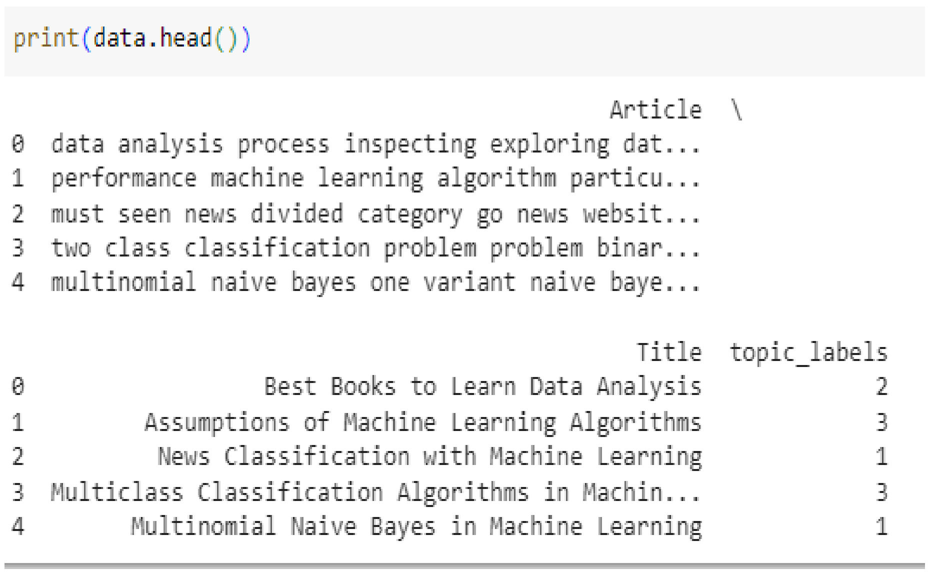


```
lda = LatentDirichletAllocation(n_components=5, random_state=42)
lda.fit(x)

topic_modelling = lda.transform(x)

topic_labels = np.argmax(topic_modelling, axis=1)
data['topic_labels'] = topic_labels
```

Presented below is the finalized dataset, enriched with topic labels:



```
print(data.head())
```

```
                                             Article  \
0  data analysis process inspecting exploring dat...
1  performance machine learning algorithm particu...
2  must seen news divided category go news websit...
3  two class classification problem problem binar...
4  multinomial naive bayes one variant naive baye...

                                               Title  topic_labels
0                    Best Books to Learn Data Analysis             2
1          Assumptions of Machine Learning Algorithms             3
2               News Classification with Machine Learning             1
3  Multiclass Classification Algorithms in Machin...             3
4          Multinomial Naive Bayes in Machine Learning             1
```

Thus, by leveraging Machine Learning techniques in conjunction with the Python programming language, you have successfully acquired the ability to assign topic labels to textual data.

CONCLUSION

Topic Modeling emerges as a vital Natural Language Processing technique for uncovering latent topics within text documents. Its ability to identify and establish relationships between textual content and topics holds significant value in understanding the underlying themes present in a corpus.

REFERENCES

1. https://thecleverprogrammer.com/2020/10/24/topic-modeling-with-python/
2. https://ourcodingclub.github.io/tutorials/topic-modelling-python/
3. https://monkeylearn.com/blog/introduction-to-topic-modeling/
4. https://www.analyticsvidhya.com/blog/2016/08/beginners-guide-to-topic-modeling-in-python/
5. https://www.loginworks.com/blogs/how-to-implement-topic-modeling-in-machine-learning-python/