AUTOMATED NEWS CATEGORIZER

MARWAN YEEDENG

NIHENG MAE

FATONI UNIVERSITY

1443/2022

AUTOMATED NEWS CATEGORIZER

MARWAN YEEDENG

611431018

NIHENG MAE

601431012

FATONI UNIVERSITY

1443/2022

FATONI UNIVERSITY

FACULITY OF SCIENCE AND TECHNOLOGY

DEAPARMENT OF INFORMATION TECHNOLOGY

TITLE

AUTOMATED NEWS CATEGORIZER

PRESENT BY

MARWAN YEEDENG

611431018

NIHENG MAE

601431012

………..………..……… ADVISOR

(MR. KHOLED LANGSARI)

DATE: …. /…. /1443

…. /…. /2022

DEPARMENT OF INFORMATION TECHNOLOGY APPROVES THIS PROJECT REPORT AS PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHERLOR OF INFORMATION TECHNOLOGY

ACADEMIC YEAR 1436/2015

………..………..……… HEAD DEPARTMENT

(MR. FAUZAN MAPA)

DATE: …. /…. /1443

…. /…. /2022

……….…………..………….………..……….

(MR. SOBREE HAYEEMAD)

DATE: …. /…. /1443

…. /…. /2022

Title: Automated News categorizer

Authors: Marwan Yeedeng 611431018, Niheng Mae 601431012

Department: Information Technology

Academic year: 2022

ABSTARCT

In recent years Many news companies are trying to reduce paper use by using online media and replacing traditional newspapers and articles printing. For this reason, there are various news articles. However, manually categorizing news articles into relevant categories is difficult and time-consuming. Automatic categorization of news articles can benefit news companies and society in many ways.

หัวข้อ:

ผู้เขียน:

สาขา:

ปีการศึกษา: 2565

บทคัดย่อ

ในช่วงไม่กี่ปีที่ผ่านมา บริษัทข่าวหลายแห่งพยายามลดการใช้กระดาษโดยใช้สื่อออนไลน์และแทนที่การพิมพ์หนังสือพิมพ์และบทความแบบเดิมๆ ด้วยเหตุนี้จึงมีบทความข่าวต่างๆ อย่างไรก็ตาม การจัดหมวดหมู่บทความข่าวตามหมวดหมู่ที่เกี่ยวข้องด้วยตนเองนั้นยากและใช้เวลานาน การจัดหมวดหมู่บทความข่าวโดยอัตโนมัติสามารถเป็นประโยชน์ต่อบริษัทข่าวและสังคมในหลายๆ ด้าน

TABLE OF CONTENTS

CHAPTER ONE

INTRODUCTION

1.0. PROJECT OVERVIEW

Due to the widespread availability of the Internet, there are many news sources that publish massive amounts of daily news. In addition, people's appetite for news has increased at an unprecedented rate. Therefore, it is important for news to be automatically categorized in order for people to have instant and efficient access to the news they want. One of the major problems with online news kits is the categorization of many news articles and articles.

In order to solve this problem, the machine learning model along with the NLP (Natural Language Processing) processes the human language that has been written in news articles along with Random Forest Classifier to categorize news into categories and summarize by using the technique of data visualization.

1.1. PROBLEM STATEMENTS

In recent years Many news companies are trying to reduce paper use by using online media and replacing traditional newspapers and articles printing. For this reason, there are various news articles. However, manually categorizing news articles into relevant categories is difficult and time-consuming. Automatic categorization of news articles can benefit news companies and society in many ways.

However, automatically categorizing news headlines is a challenging task as the length of news articles varies. An automated way is needed to retrieve and access news based on user interests. Therefore, this article aims to propose an automatic categorization of headlines using machine learning techniques.

1.2. OBJECTIVE

- To study and find the most suitable model to classifier news categories.
- To learn algorithms for developing classification machine learning to achieve accuracy and stability.
- To study the process adoption of text classification by using Random Forest Classifier.
- To bring the knowledge learned to develop academic use.

1.3. SIGNIFICANCE OF STUDY

- Gaining a way to adjust the accuracy of the model to be more stable.
- Gaining a model that can predict the category of news from the article with high efficiency, high accuracy, and stability.
- This method of customizing the model can be applied to other projects.

1.4. SCOPE OF STUDY

The scope of this project is to have a model that can automatically categorize news as the most efficient and accurate of all the models to be tested and improved accuracy. By using some news data from the New York Time Let's do some training and some testing. To make the model predict news categories from news articles.

1.5. SOFTWARE AND HARDWARE REQUIREMENT

1.5.1. SOFTWARE REQUIREMENT

1.5.1.1. Resource

- New York Time API

1.5.1.2. Code Editor

- Jupyter Notebook
- Visual Studio Code

1.5.1.3. Programing Languages

- Python

1.5.1.4. Python Libraries

- Dateutil

- NumPy
- Pandas
- Pickle
- Re (Regular expression)
- Requests
- Scikit Learn
- Seaborn
- Time
- WordCloud

1.5.1.5. Micro Framework

- Flask

1.5.1.6. Version Control

- GitHub

1.5.2. HARDWARE REQUIREME

1.5.2.1. Personal Computer

- Asus VivoBook 15 x512da
- HP Pavilion Power 15-cb035TX

1.6. CONCLUSION

With many changes, there has been an increase in the number of online news writing. Therefore, there are various news articles. Many in the news website database. And manually categorizing news articles into relevant categories is difficult and time consuming. Automatic categorization of news articles using machine learning techniques can benefit news companies and society in many ways. The start of this project will bring many changes to the news industry.

CHAPTER TWO

LITERATURE REVIEW

2.0. INTRODUCTION

This section discusses the literature reviews of the most important studies related to our topic. These topics are the definition, tools used, related work.

2.1. DEFINITION

2.1.1. NEWS CATEGORIZATION

News categorization or classification, is a way of assigning documents to one or more predefined categories. This helps the users to look for information faster by searching only in the categories they want to, rather than searching the entire information space. (Explorer and News 2014)

2.1.2. AUTOMATED NEWS CATEGORIZER

Automated News Categorization is the process of assigning text documents to one or more predefined categories. This allows users to find desired information faster by searching only the relevant categories and not the entire information space. (Ee 2001)

2.1.3. MACHINE LEARNING

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. (Panesar 2019)

2.1.4. CLASSIFICATION

Classification is the process of identifying and grouping objects or ideas in-to predetermined categories. In data management, classification enables the separation and sorting of data according to set requirements for various business or personal objectives. (What is Classification? - Definition from Techopedia 2021)

2.1.5. RANDOM FOREST CLASSIFIER

A meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. (sklearn.ensemble.RandomForestClassifier — scikit-learn 1.0.2 documentation n.d.)

2.2. TOOLS USED

2.2.1. CODE EDITOR

2.2.1.1. Jupyter Notebook

The Jupyter Notebook is a web application for creating and sharing documents that contain code, visualizations, and text. It can be used for data science, statistical modeling, machine learning, and much more. (Kluyver et al. 2016)

2.2.1.2. Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity). (Documentation for Visual Studio Code n.d.)

2.2.2. PROGRAMING LANGUAGE

2.2.2.1. Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. (What is Python? Executive Summary | Python.org n.d.)

2.2.3. PYTHON LIBRARIES

Python Libraries are a set of useful functions that eliminate the need for writing codes from scratch. (mygreatlearning 2021)

2.2.3.1. Dateutil

The dateutil module provides powerful extensions to the standard datetime module, available in Python. (dateutil - powerful extensions to datetime — dateutil 2.8.2 documentation n.d.)

2.2.3.2. NumPy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. (NumPy 2021)

2.2.3.3. Pandas

Pandas is a Python library for data analysis. Started by Wes McKinney in 2008 out of a need for a powerful and flexible quantitative analysis tool, pandas have grown into one of the most popular Python libraries. It has an extremely active community of contributors. (Pandas | Python Library - Mode n.d.)

2.2.3.4. Pickle

Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions, or transport data over the network. (Python pickling: What it is and how to use it securely | Synopsys n.d.)

2.2.3.5. Re (Regular expression)

A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a particular string, which comes down to the same thing). (re — Regular expression operations — Python 3.10.4 documentation n.d.)

-

2.2.3.6. Requests

The requests library is the de facto standard for making HTTP requests in Python. It abstracts the complexities of making requests behind a beautiful, simple API so that you can focus on interacting with services and consuming data in your application. (Python’s Requests Library (Guide) – Real Python n.d.)

2.2.3.7. Scikit Learn

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in

Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib. (Scikit Learn - Introduction n.d.)

2.2.3.8. Seaborn

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. (seaborn: statistical data visualization — seaborn 0.11.2 documentation n.d.)

2.2.3.9. Time

The time module provides a number of functions that deal with dates and the time within a day. It’s a thin layer on top of the C runtime library. A given date and time can either be represented as a floating-point value (the number of seconds since a reference date, usually January 1, 1970), or as a time tuple. (The time Module - Python Standard Library [Book] n.d.)

2.2.3.10. WordCloud

Many times, you might have seen a cloud filled with lots of words in different sizes, which represent the frequency or the importance of each word. This is called Tag Cloud or WordCloud. (Python Word Clouds Tutorial: How to Create a Word Cloud | DataCamp n.d.)

2.2.4. MICRO FRAMEWORK

2.2.4.1. Flask

Flask is a lightweight framework written in the Python programming language. It is easy to learn and simple to use, enabling you to create and build your own web applications in a short amount of time. (Flask: Python Micro Framework n.d.)

2.2.5. VERSION CONTROL

2.2.5.1. GitHub

GitHub is a for-profit company that offers a cloud-based Git repository hosting service. Essentially, it makes it a lot easier for individuals and teams to use Git for version control and collaboration. (What Is GitHub? A Beginner's Introduction to GitHub n.d.)

2.3. RELATED WORK

2.3.1. AUTOMATIC SEMANTIC CATEGORIZATION OF NEWS HEADLINES USING ENSEMBLE MACHINE LEARNING: A COMPARATIVE STUDY

Due to widespread availability of Internet, there are a huge of sources that produce massive amounts of daily news. Moreover, the need for information by users has been increasing unprecedently, so it is critical that the news is automatically classified to permit users to access the required news instantly and effectively. One of the major problems with online news sets is the categorization of the vast number news and articles. In order to solve this problem, the machine learning model along with the Natural Language Processing (NLP) is widely used for automatic news classification to categorize topics of untracked news and individual opinion based on the user's prior interests. However, the existing studies mostly rely on NLP but uses huge documents to train the prediction model, thus it is hard to classify a short text without using semantics. Few studies focus on exploring classifying the news headlines using the semantics. Therefore, this paper attempts to use semantics and ensemble learning to improve the short text classification. The proposed methodology starts with preprocessing stage then applying feature engineering using word2vec with TF-IDF vectorizer. Afterwards, the classification model was developed with different classifier KNN, SVM, Naïve Bayes and Gradient boosting. The experimental results verify that Multinomial Naïve Bayes shows the best performance with an accuracy of 90.12% and recall 90%. (Bogery et al. 2019)

2.3.2. A COMPARATIVE ANALYSIS OF NEWS CATEGORIZATION USING MACHINE LEARNING APPROACHES

The rapid growth of print and digital media increased the reach of one and all in terms of information, resulting in more amount of Text data to be mined. This data is nothing but a heap of unclassified information which when kept together means nothing. This means that there is a need to tag all these data i.e., News Classification. News classification is the task of automatically classify the news documents into their predefined classes based on their content with the confidence learned from the training news dataset. This research evaluates some most widely used machine learning techniques, mainly Naive Bayes, Random Forest, Decision Tree, SVM and Neural Networks, for automatic news classification problem. To experiment the system, a dataset from BBC that have two columns, one has the news headlines and the other contains the type it belongs to. There are 2225 rows in the data set is used. The average results show that the Naive Bayes is performing better than the other four algorithms with the classification accuracy of 96.8 %. Then follows the Random Forest with accuracy 94.1 %, Support Vector Machine (SVM) with accuracy 96.4 %, Neural Networks with accuracy 96.4 % and the Decision Tree with accuracy 83.2 %. (Deb et al. 2020)

2.3.3. TEXT MINING APPROACH TO CLASSIFY TECHNICAL RESEARCH DOCUMENT USING NAÏVE BAYES

World Wide Web is the store house of abundant information available in various electronic forms. Since past few years, the increase in the performance of computers in handling large quantity of text data has led researchers to focus on reliable and optimal retrieval of visible and implied information that exist in the huge resources. In text mining, one of the challenging and growing importance’s is given to the task of document classification or text characterization. In this process, reliable text extraction, robust methodologies and efficient algorithms such as Naive Bayes and other made the task of document classification to perform consistently well. Classifying text documents using Bayesian classifiers are among the most successful known algorithms for machine learning. This paper describes implementations of Naïve Bayesian (NB) approach for the automatic classification of Documents restricted to Technical Research documents based on their text contents and its results analysis. We also discuss a comparative analysis of Weighted Bayesian classifier approach with the Naive Bayes classifier. (M et al. 2015)

2.4. CONCLUSION

In this second chapter there are three main parts to be found in this second chapter: Definition, Tools used, and Related words. The definition describes the definitions that will be found in this project, The tools used will be Part of the tools used in this project and the importance of each tool is explained. Finally, the related word describes the articles used in the project. This will be an important part of the implementation and development of this project.

CHAPTER THREE

METHODOLOGY

3.0. INTRODUCTION

The CRoss Industry Standard Process for Data Mining (CRISP-DM) is a process model with six phases that naturally describes the data science life cycle. It's like a set of guardrails to help you plan, organize, and implement your data science (or machine learning) project. (Saltz and Hotz 2020)

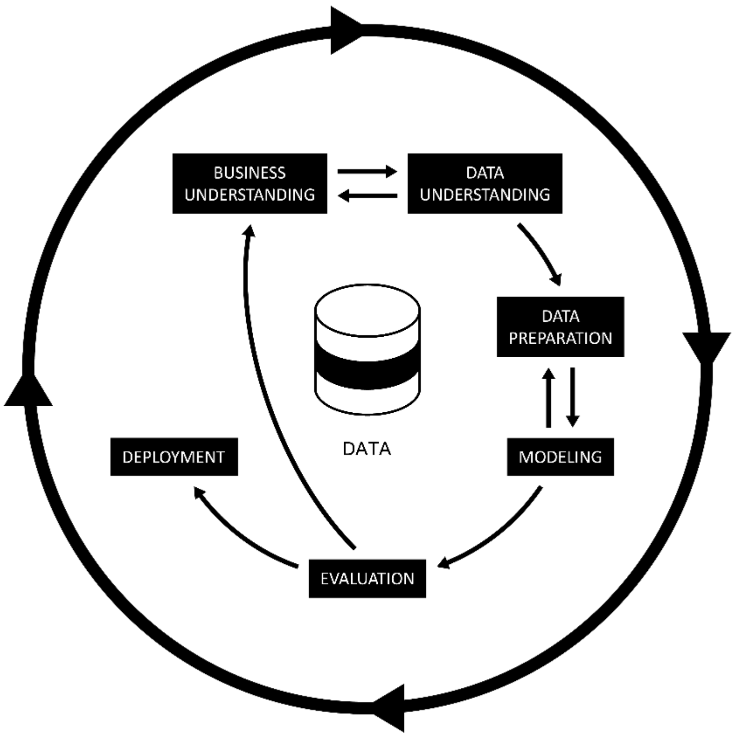


Figure 1: CRoss Industry Standard Process for Data Mining (CRISP-DM)

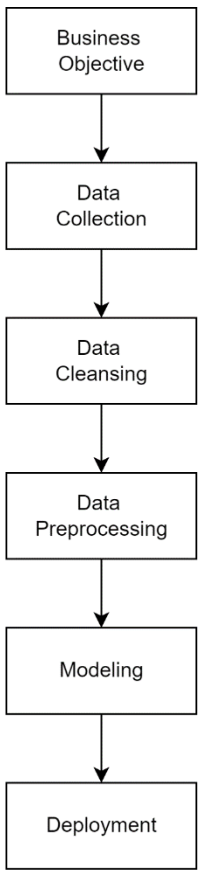


Figure 2: Project Methodology Flowchart.

3.1. BUSINESS OBJECTIVE

Business objective of this project is to education and development How to use machine learning techniques to deal with the enormous amount of news category beyond humans to handle. It can also benefit many news and social media companies.

3.2. DATA COLLECTION

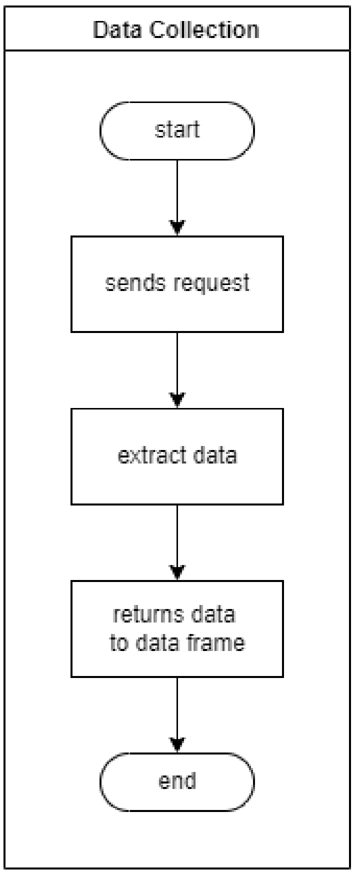


Figure 3: Data Collection Flowchart.

Data Collection phase, focused on collecting necessary data for using to training and testing machine learning. Start from sends a request to the NYT Archive API for access data second extract necessary data need to use and return that data to the Pandas data frame.

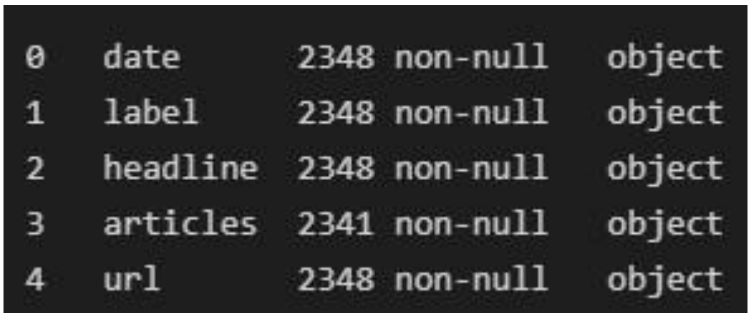
3.3. DATA CLEANSING

3.3.1. CATEGORY SELECTION

Out of 47 categories, we have chosen five because they are proportionately proportioned, with the category we selected being 1. opinion. A category about the opinions of individuals with a total of 503 sets of information, world. Category about news around the world with 482 sets of information, politics category about sports news with 226 sets of information, arts category about news, arts and entertainment, with all the information, Business News about various businesses

3.3.2. DEALING WITH MISSING DATA

The data that we have obtained has missing data in a column named articles which is column about news content. After checking the data. In summary, column named articles has 7 sets of missing data.



```
0   date      2348 non-null   object
1   label     2348 non-null   object
2   headline  2348 non-null   object
3   articles  2341 non-null   object
4   url       2348 non-null   object
```

Figure 4: The Summary of Each Column in The Data Frame.

We have cleared the missing data by eliminating the missing rows as new data cannot be replaced by the missing data as part of the news articles.

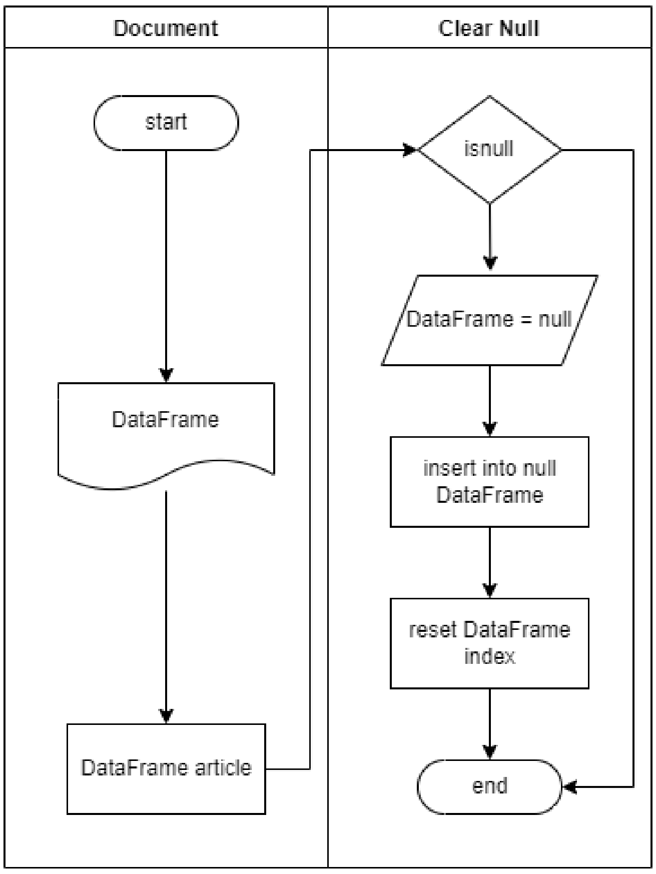


Figure 5: Diagram of Dealing With Missing Data Process.

Dealing with missing data process is the process that is to deal with missing values. We have found that the data we got contain missing values in articles column so we created. The process will start by checking the data from articles are whether null or not and we created the column name check, if the data are null the row will be inserted the value true if not the value will be false and then each row that contain the value true in check column will be removed.

3.3.3. WRAPPING UNNECESSARY CHARACTERS

To cut characters like [- () \"#/@; :<> {} `+=~|.!?,] discarded as it is unnecessary for processing in modeling.

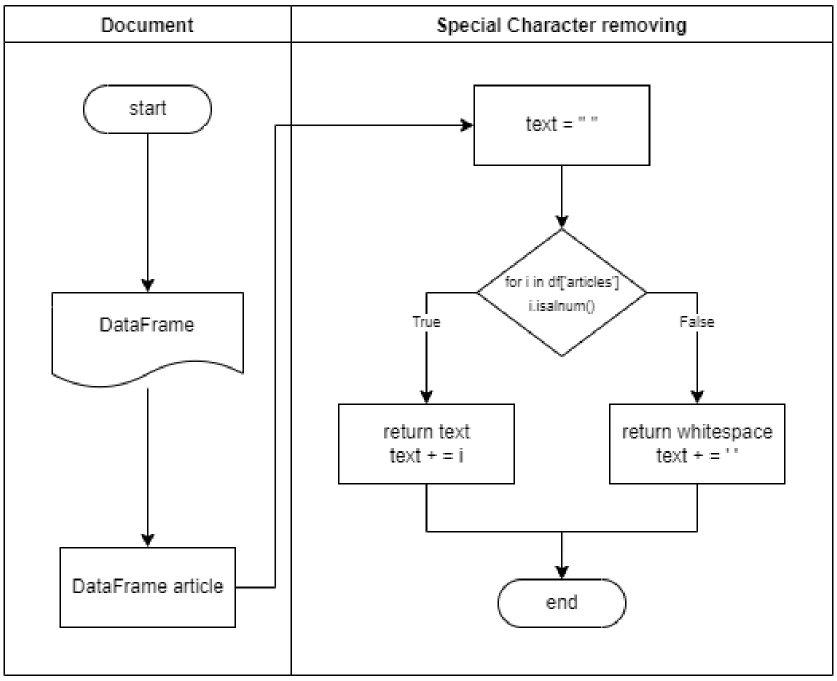


Figure 6: Diagram of Special Character Removing Process.

This process is to remove the special characters like [- () \"#/@; :<> {} `+=~|.!?,]. The process begin by checking if text are alphanumeric or not by using python built-in function isalnum() if the text are alphanumeric. It will return nothing but if not, it will return white space and remove the text.

3.3.4. LOWERCASE THE CHARACTERS

To lowercase all the text of news articles before the process of natural language processing because the words with uppercase will be different from the words with lowercase even the some meaning.

3.3.5. NATURAL LANGUAGE PROCESSING

In this step we have goals: 1. Extracting words 2. Removing unnecessary words 3. Turning words into root words.

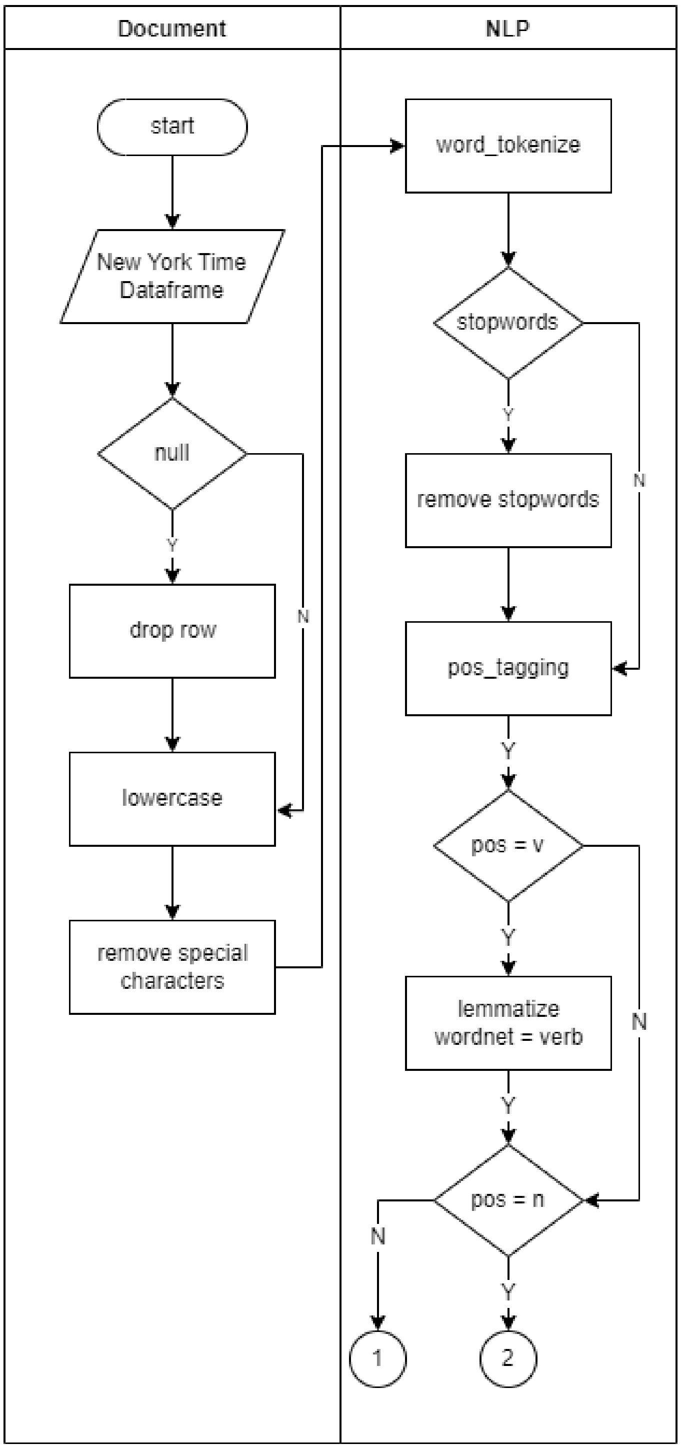


Figure 7: Diagram of NLP Process.

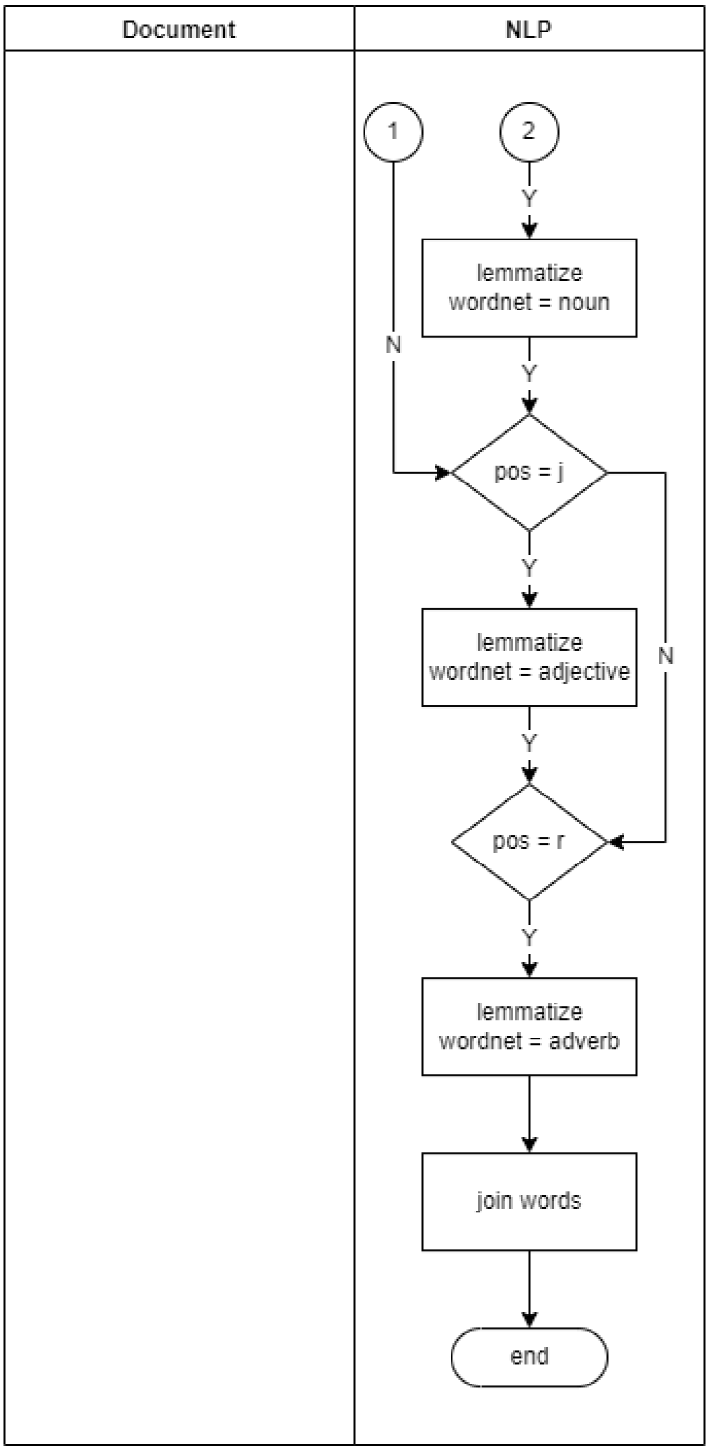


Figure 8: Diagram of NLP Process.

The process starts with drop the missing values from Dealing with Missing Data Process and then lowercase all the characters in articles column after that remove all special characters from Special Character Removing Process. After that we tokenize all the words into tokens by using nltk library function word_tokenize(). The tokens will be collected as list after that we will check if the characters are stopwords or not by using stopwords.words function from nltk library If the characters are stopwords will be removed. The next is pos taggin, we tag each word with its part of speech such as noun, verb so that we can use nltk wordnetlemmatizer dictionary to lemmatize each word. The next is word lemmatization. In this step we used nltk wordnetlemmatizer to lemmatize each word. The result will be all the words will be changed into its root words and the last is to join all the words in list into string as a paragraph.

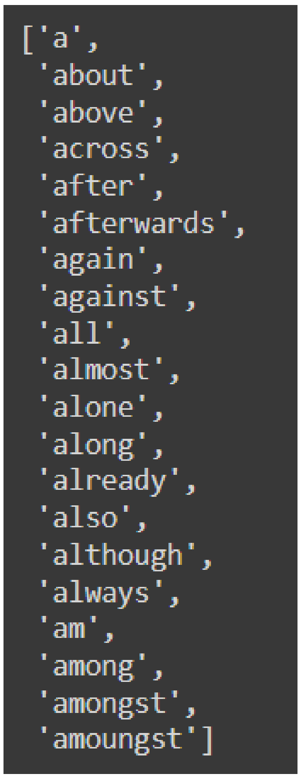
3.3.6. WORD TOKENIZATION

Is to cut out the words from the sentence one by one using nltk.tokenize package.

3.3.7. STOPWORDS

3.3.7.1. Stopwords

Stopwords are common words that we often find in documents that don't really help in conveying the meaning, such as a, an, so, the, also, just, etc. The words are from nltk.stem.wordnet



```
['a',
 'about',
 'above',
 'across',
 'after',
 'afterwards',
 'again',
 'against',
 'all',
 'almost',
 'alone',
 'along',
 'already',
 'also',
 'although',
 'always',
 'am',
 'among',
 'amongst',
 'amoungst']
```

Figure 9: List of Stop Words 20 Words.

3.3.7.2. Stopwords removing

Is to remove unnecessary words from news articles by using stopwords package from nltk.corpus.

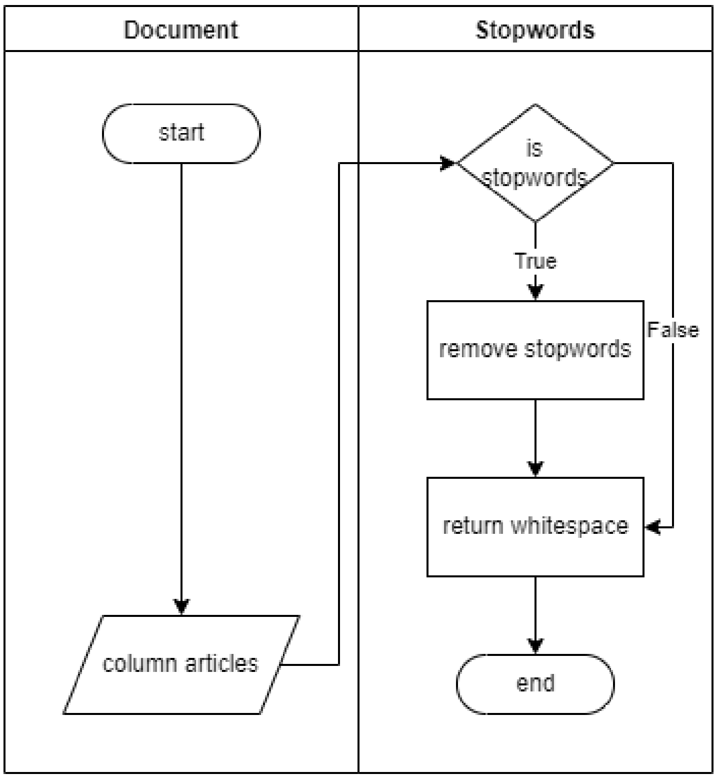


Figure 10: Diagram of Stopwords Process.

This process start by check if the characters are stopwords or not by using stopwords.words function from nltk library If the characters are stopwords will return as whitespace and the words will be removed.

3.3.8. WORDS LEMMATIZATION

3.3.8.1. Part of Speech tagging

Part of Speech tagging is to label which word is part of a sentence such as noun, verb, object by using nltk.pos_tag The pos_tag is so that we can use it to easily transform a word into a root word in word lemmatization stage.

3.3.8.2. Words Lemmatization

Is The process of converting words into their basic form, for example is, am, are when lemmatization. It becomes the word be so that the matrix formation process results in the same word if the word has the same meaning but has a different form.

3.3.9. EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

3.3.9.1. Word Could

Word Cloud is the process of data visualization for visualize texts to gain the insight. Word Cloud is the technique that we used in the project to visualize the most words in the articles for gaining the insight of each category.

3.4. DATA PREPOCESSING

3.4.1. CONVERTING CATEGORY NAMES IN CATEGORY ID FOR EACH NAMES

The procedure for converting category names into a number and creates a column named category_id by using pandas factorize function.

3.4.2. EXTRACTING FEATURE FROM TEXT

This process we have used CountVectorizer function from Scikit-learn to extract feature from text in news articles. The output from this process is the matrix of numeric of text frequency from each row of dataframe.

3.4.3. TRAIN TEST SPLIT

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. We used train_test_split function from Scikit-learn to split our dataset.

3.5. MODELING

3.5.1. FINE-TUNE

3.5.1.1. Fine-Tuning

Is the process in which parameters of a model must be adjusted very precisely in order to fit with certain observations.

3.5.1.2. Randomized-Search

Is used to find the optimal hyperparameters of a model which results in the most accurate predictions. In the project, we have used randomized-search to fine-tune the model for improving accuracy of RandomForest Classifier.

3.5.1.3. Randomized-Search Hyperparameters Tuning

bootstrap	[True, False]
max_depth	[None, 2, 4]
max_features	['auto', 'sqrt']
min_samples_leaf	[1, 2, 5]
min_samples_split	[2, 4]

n_estimators	[27, 93, 46, 31, 8, 18, 25, 11, 55, 42, 17, 81, 76, 2, 54, 44, 51, 24, 6, 52, 19, 95, 79, 53, 43, 47, 98, 45, 85, 58, 88, 10, 39, 89, 13, 20, 36, 50, 67, 34, 72, 0, 16, 71, 26, 94, 73, 75, 12, 59, 97, 82, 61, 48, 65, 90, 41, 21, 70, 78, 29, 68, 63, 35, 32, 69, 80, 23, 56, 7, 4, 60, 91, 64, 30, 99, 66, 49, 14, 22, 37, 77, 87, 84, 83, 74, 96, 9, 92, 28, 40, 5, 33, 15, 1, 86, 62, 57, 38, 3]

Table 1 : Table Randomized-Search Hyperparameters Tuning.

3.5.2. ACCURACY SCORE

3.5.2.1. Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing. We used confusion matrix to describe the accuracy of the model.

3.5.3. PREDICTION

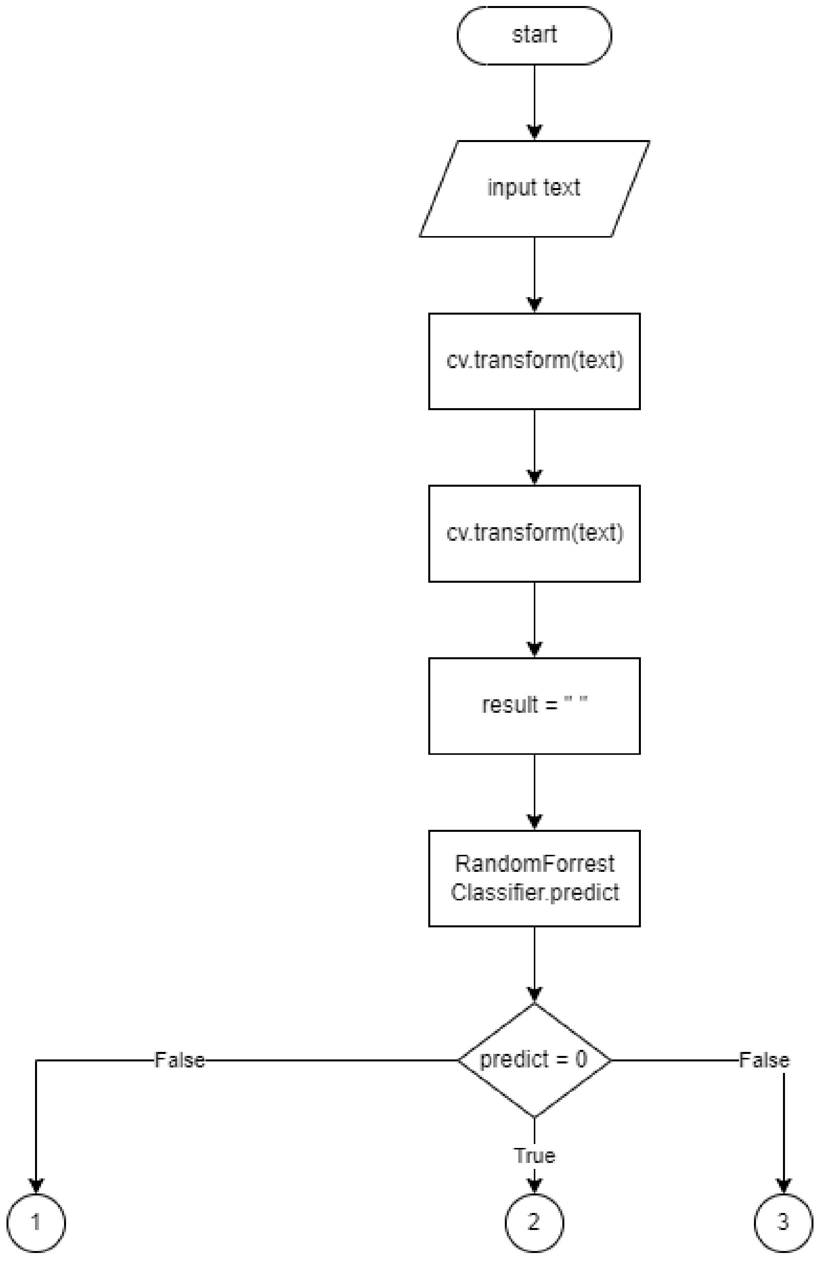


Figure 11: Diagram of Prediction Process.

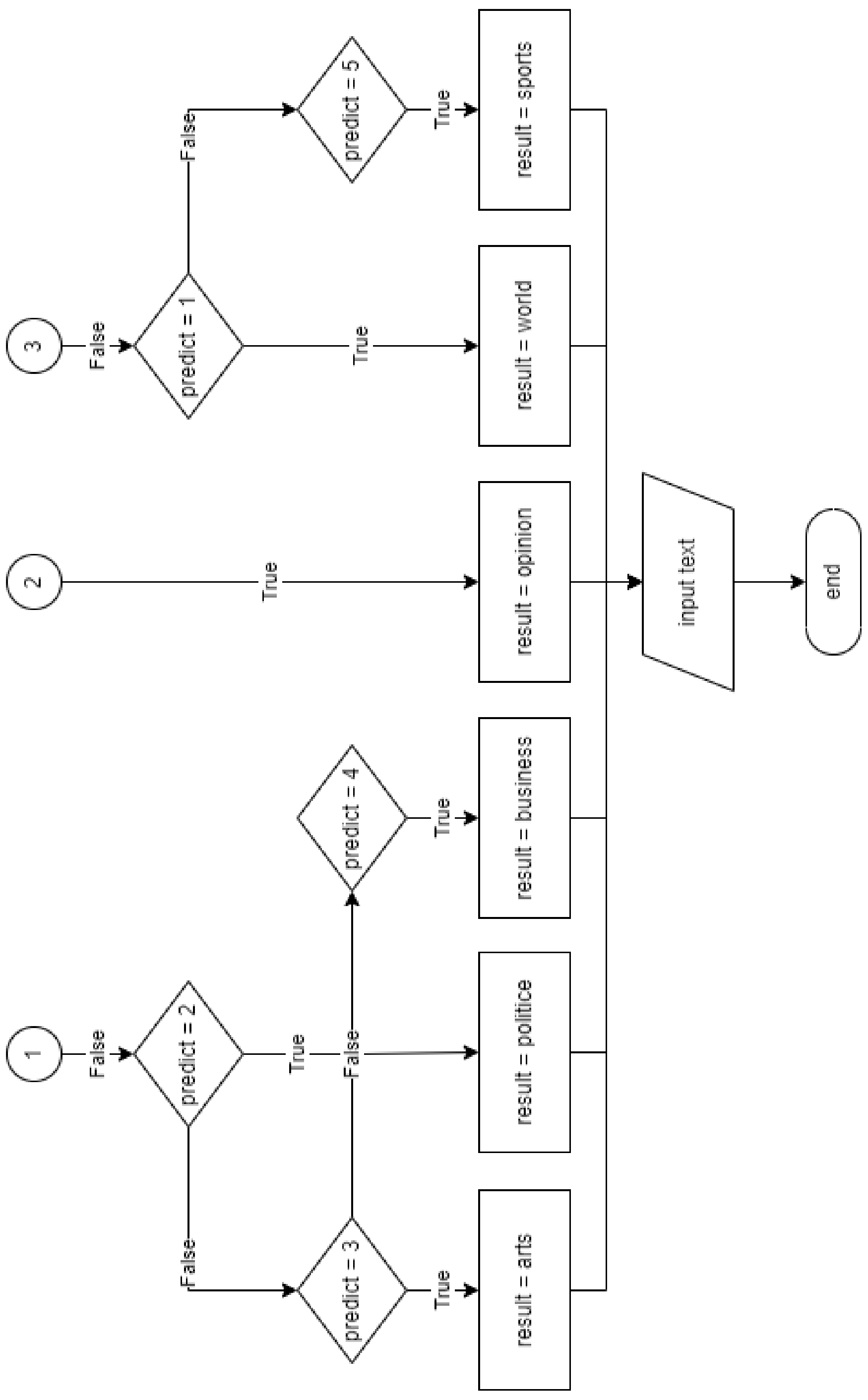


Figure 12: Diagram of Prediction Process.

This process start by taking all the text to cv.transform. After all the text are transformed, we have declared result variable to store value of prediction. The transformed text will be predicted by our model and after the process of prediction is finished. We define condition using if-else statement. If the value from the model is 0. the result will be “world”. if the value from the model is 2. the result will be “politics”. if the value from the model is 3. the result will be “arts”. If the value from the model is 4. the result will be “business”. If the value from the model is 5. the result will be “sport”.

3.6. DEPLOYMENT

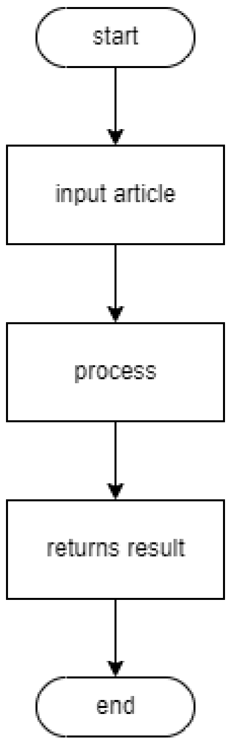


Figure 13: Web Application Process.

3.7. CONCLUSION

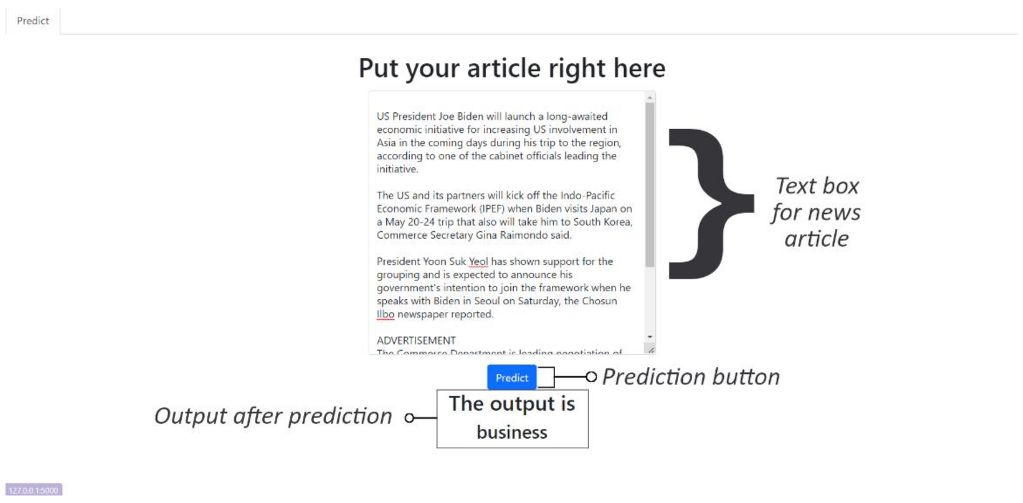
CHAPTER FOUR

FINDING AND IMPLEMENTATION

4.0. INTRODUCTION

4.1. FINDING

4.1.1. INTERFACE



US President Joe Biden will launch a long-awaited economic initiative for increasing US involvement in Asia in the coming days during his trip to the region, according to one of the cabinet officials leading the initiative.

The US and its partners will kick off the Indo-Pacific Economic Framework (IPEF) when Biden visits Japan on a May 20-24 trip that also will take him to South Korea, Commerce Secretary Gina Raimondo said.

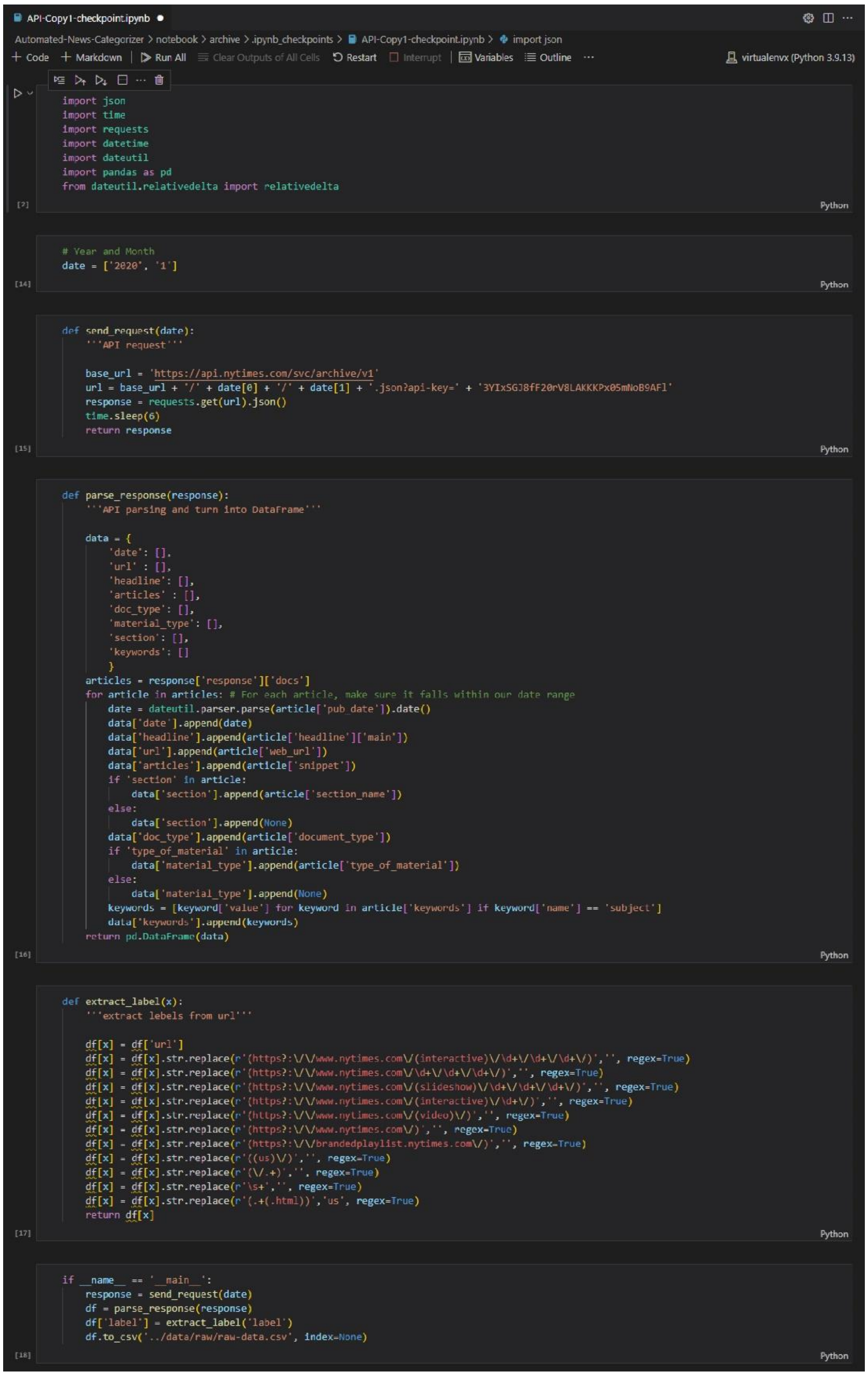
President Yoon Suk Yeol has shown support for the grouping and is expected to announce his government's intention to join the framework when he speaks with Biden in Seoul on Saturday, the Chosun Ilbo newspaper reported.

ADVERTISEMENT

Figure 14: Image of News Categorizer Web Application.

4.1.2. SOURCE CODE

4.1.2.1. Data Collection


API-Copy1-checkpoint.ipynb
Automated-News-Categorizer > notebook > archive > .ipynb_checkpoints > API-Copy1-checkpoint.ipynb > import json
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ...
virtualenvx (Python 3.9.13)

```python
import json
import time
import requests
import datetime
import dateutil
import pandas as pd
from dateutil.relativedelta import relativedelta
```
[2] Python

```python
# Year and Month
date = ['2020', '1']
```
[14] Python

```python
def send_request(date):
    '''API request'''

    base_url = 'https://api.nytimes.com/svc/archive/v1'
    url = base_url + '/' + date[0] + '/' + date[1] + '.json?api-key=' + '3YIxSGJ8fF20rV8LAKKKPx05mNoB9AFl'
    response = requests.get(url).json()
    time.sleep(6)
    return response
```
[15] Python

```python
def parse_response(response):
    '''API parsing and turn into DataFrame'''

    data = {
        'date': [],
        'url' : [],
        'headline': [],
        'articles' : [],
        'doc_type': [],
        'material_type': [],
        'section': [],
        'keywords': []
        }
    articles = response['response']['docs']
    for article in articles: # For each article, make sure it falls within our date range
        date = dateutil.parser.parse(article['pub_date']).date()
        data['date'].append(date)
        data['headline'].append(article['headline']['main'])
        data['url'].append(article['web_url'])
        data['articles'].append(article['snippet'])
        if 'section' in article:
            data['section'].append(article['section_name'])
        else:
            data['section'].append(None)
        data['doc_type'].append(article['document_type'])
        if 'type_of_material' in article:
            data['material_type'].append(article['type_of_material'])
        else:
            data['material_type'].append(None)
        keywords = [keyword['value'] for keyword in article['keywords'] if keyword['name'] == 'subject']
        data['keywords'].append(keywords)
    return pd.DataFrame(data)
```
[16] Python

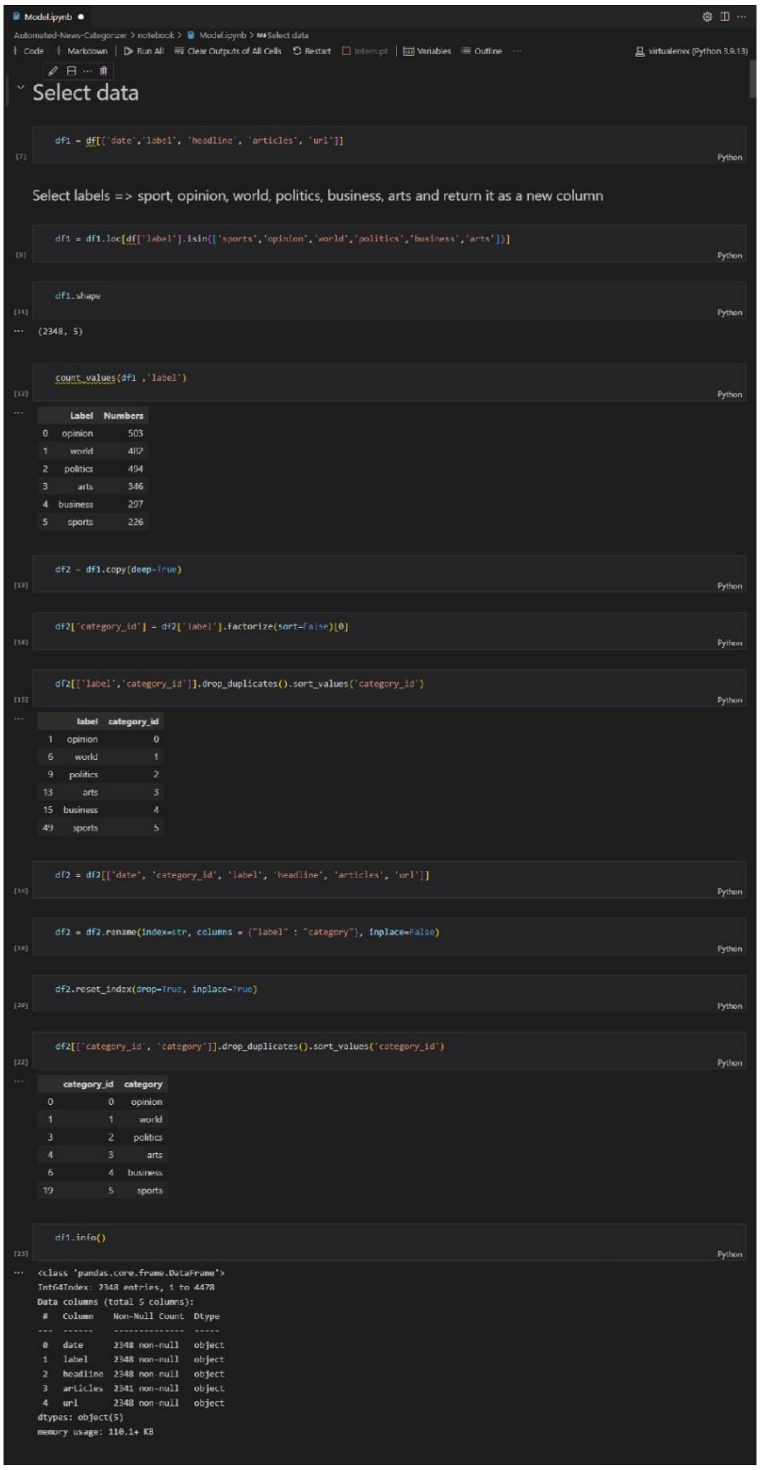
```python
def extract_label(x):
    '''extract lebels from url'''

    df[x] = df['url']
    df[x] = df[x].str.replace(r'(https?:\/\/www.nytimes.com\/(interactive)\/\d+\/\d+\/\d+\/)','', regex=True)
    df[x] = df[x].str.replace(r'(https?:\/\/www.nytimes.com\/\d+\/\d+\/\d+\/)','', regex=True)
    df[x] = df[x].str.replace(r'(https?:\/\/www.nytimes.com\/(slideshow)\/\d+\/\d+\/\d+\/)','', regex=True)
    df[x] = df[x].str.replace(r'(https?:\/\/www.nytimes.com\/(interactive)\/\d+\/)','', regex=True)
    df[x] = df[x].str.replace(r'(https?:\/\/www.nytimes.com\/(video)\/)','', regex=True)
    df[x] = df[x].str.replace(r'(https?:\/\/www.nytimes.com\/)','', regex=True)
    df[x] = df[x].str.replace(r'(https?:\/\/brandedplaylist.nytimes.com\/)','', regex=True)
    df[x] = df[x].str.replace(r'((us)\/)','', regex=True)
    df[x] = df[x].str.replace(r'(\/.+)','', regex=True)
    df[x] = df[x].str.replace(r'\s+','', regex=True)
    df[x] = df[x].str.replace(r'(.+(.html))','us', regex=True)
    return df[x]
```
[17] Python

```python
if __name__ == '__main__':
    response = send_request(date)
    df = parse_response(response)
    df['label'] = extract_label('label')
    df.to_csv('../data/raw/raw-data.csv', index=None)
```
[18] Python

Figure 15: Notebook of Data Collection.

4.1.2.2. Data Cleansing


Model.ipynb
Automated-News-Categorizer > notebook > Model.ipynb > Select data
Code
Markdown
Run All
Clear Outputs of All Cells
Restart
Interrupt
Variables
Outline
virtualenvx (Python 3.9.13)
Python

Select data

```python
df1 = df[['date','label', 'headline', 'articles', 'url']]
```

Select labels => sport, opinion, world, politics, business, arts and return it as a new column

```python
df1 = df1.loc[df['label'].isin(['sports','opinion','world','politics','business','arts'])]
```

```python
df1.shape
```

```
(2348, 5)
```

```python
count_values(df1 ,'label')
```

	Label	Numbers
0	opinion	503
1	world	482
2	politics	494
3	arts	346
4	business	297
5	sports	226

```python
df2 = df1.copy(deep=True)
```

```python
df2['category_id'] = df2['label'].factorize(sort=False)[0]
```

```python
df2[['label','category_id']].drop_duplicates().sort_values('category_id')
```

	label	category_id
1	opinion	0
6	world	1
9	politics	2
13	arts	3
15	business	4
49	sports	5

```python
df2 = df2[['date', 'category_id', 'label', 'headline', 'articles', 'url']]
```

```python
df2 = df2.rename(index=str, columns = {"label" : "category"}, inplace=False)
```

```python
df2.reset_index(drop=True, inplace=True)
```

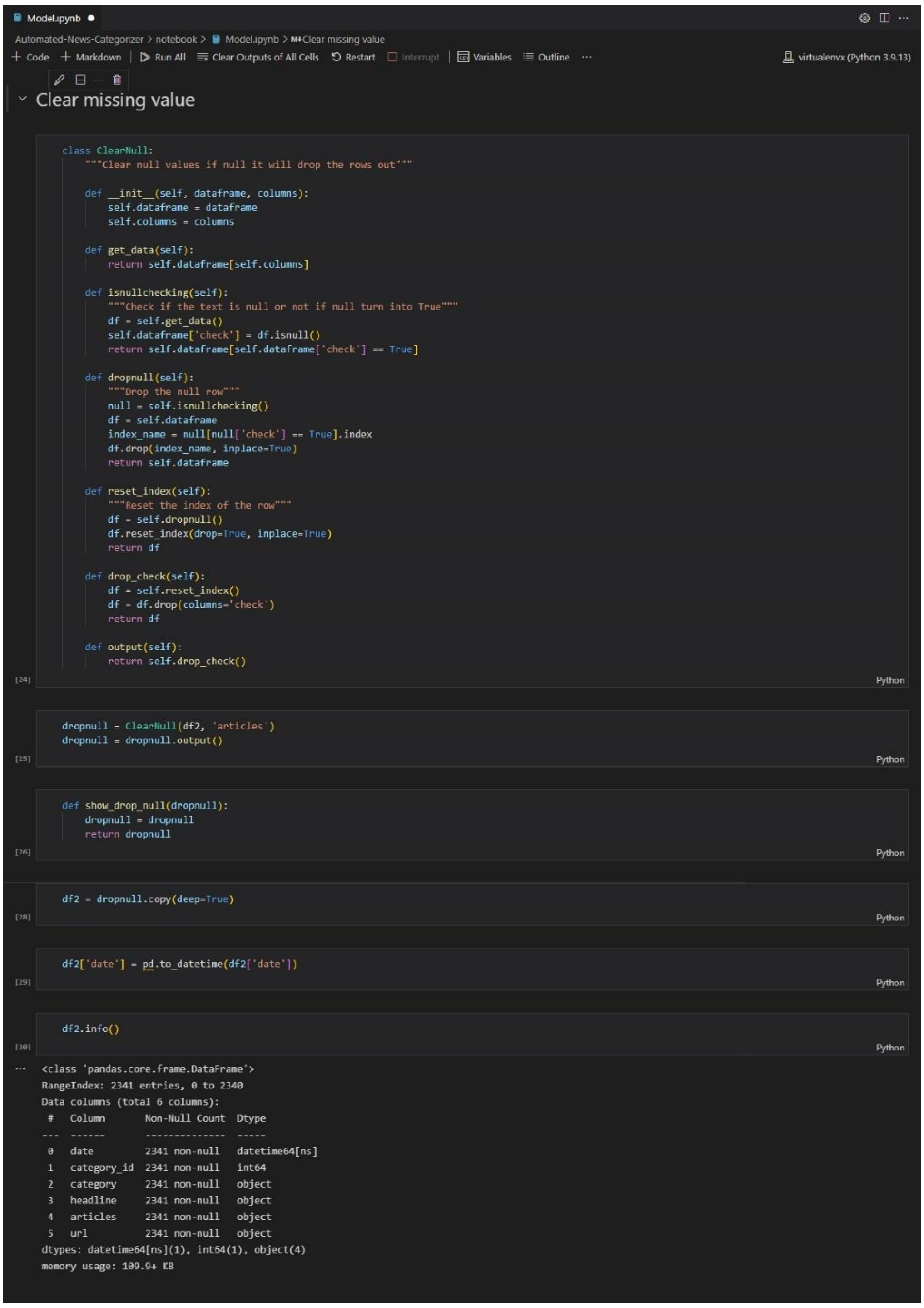
```python
df2[['category_id', 'category']].drop_duplicates().sort_values('category_id')
```

	category_id	category
0	0	opinion
1	1	world
3	2	politics
4	3	arts
6	4	business
19	5	sports

```python
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2348 entries, 1 to 4478
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   date      2348 non-null   object
 1   label     2348 non-null   object
 2   headline  2348 non-null   object
 3   articles  2341 non-null   object
 4   url       2348 non-null   object
dtypes: object(5)
memory usage: 110.1+ KB
```

Figure 16: Notebook of Select Data.


Model.ipynb
Automated-News-Categorizer > notebook > Model.ipynb > M↓Clear missing value
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ···
virtualenvx (Python 3.9.13)

Clear missing value

```
class ClearNull:
    """Clear null values if null it will drop the rows out"""

    def __init__(self, dataframe, columns):
        self.dataframe = dataframe
        self.columns = columns

    def get_data(self):
        return self.dataframe[self.columns]

    def isnullchecking(self):
        """Check if the text is null or not if null turn into True"""
        df = self.get_data()
        self.dataframe['check'] = df.isnull()
        return self.dataframe[self.dataframe['check'] == True]

    def dropnull(self):
        """Drop the null row"""
        null = self.isnullchecking()
        df = self.dataframe
        index_name = null[null['check'] == True].index
        df.drop(index_name, inplace=True)
        return self.dataframe

    def reset_index(self):
        """Reset the index of the row"""
        df = self.dropnull()
        df.reset_index(drop=True, inplace=True)
        return df

    def drop_check(self):
        df = self.reset_index()
        df = df.drop(columns='check')
        return df

    def output(self):
        return self.drop_check()
```

[24] Python

```
dropnull = ClearNull(df2, 'articles')
dropnull = dropnull.output()
```

[25] Python

```
def show_drop_null(dropnull):
    dropnull = dropnull
    return dropnull
```

[26] Python

```
df2 = dropnull.copy(deep=True)
```

[28] Python

```
df2['date'] = pd.to_datetime(df2['date'])
```

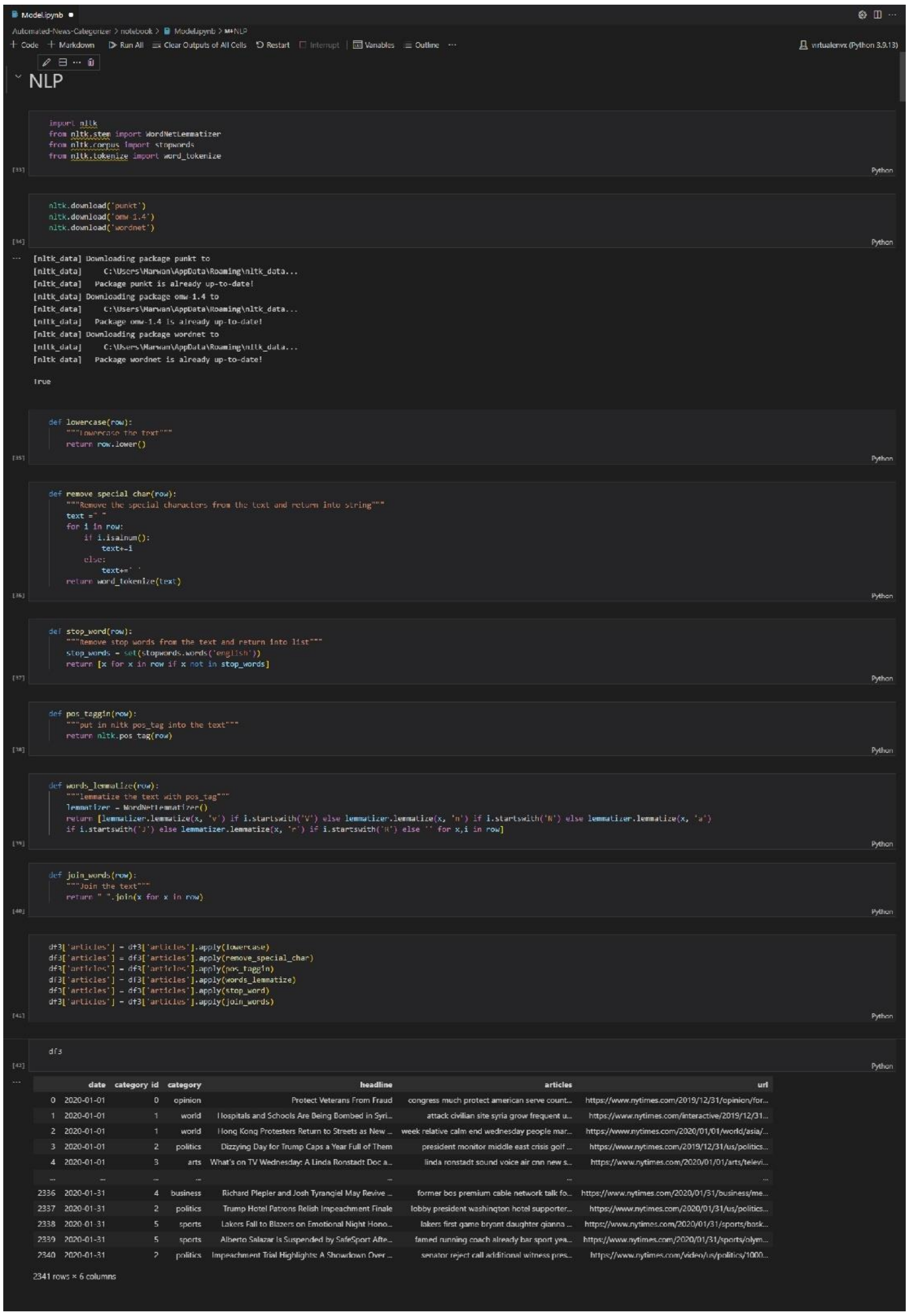
[29] Python

```
df2.info()
```

[30] Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2341 entries, 0 to 2340
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   date         2341 non-null   datetime64[ns]
 1   category_id  2341 non-null   int64
 2   category     2341 non-null   object
 3   headline     2341 non-null   object
 4   articles     2341 non-null   object
 5   url          2341 non-null   object
dtypes: datetime64[ns](1), int64(1), object(4)
memory usage: 109.9+ KB
```

Figure 17: Notebook of Clear Missing Value.


Model.ipynb
Automated-News-Categorizer > notebook > Model.ipynb > M↓NLP
+ Code
+ Markdown
Run All
Clear Outputs of All Cells
Restart
Interrupt
Variables
Outline
virtualenvx (Python 3.9.13)
Python

NLP

```
[33]
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

```
[34]
nltk.download('punkt')
nltk.download('omw-1.4')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Marwan\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\Marwan\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Marwan\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!

True
```

```
[35]
def lowercase(row):
    """Lowercase the text"""
    return row.lower()
```

```
[36]
def remove_special_char(row):
    """Remove the special characters from the text and return into string"""
    text =" "
    for i in row:
        if i.isalnum():
            text+=i
        else:
            text+=' '
    return word_tokenize(text)
```

```
[37]
def stop_word(row):
    """Remove stop words from the text and return into list"""
    stop_words = set(stopwords.words('english'))
    return [x for x in row if x not in stop_words]
```

```
[38]
def pos_taggin(row):
    """put in nltk pos_tag into the text"""
    return nltk.pos_tag(row)
```

```
[39]
def words_lemmatize(row):
    """lemmatize the text with pos_tag"""
    lemmatizer = WordNetLemmatizer()
    return [lemmatizer.lemmatize(x, 'v') if i.startswith('V') else lemmatizer.lemmatize(x, 'n') if i.startswith('N') else lemmatizer.lemmatize(x, 'a')
    if i.startswith('J') else lemmatizer.lemmatize(x, 'r') if i.startswith('R') else '' for x,i in row]
```

```
[40]
def join_words(row):
    """Join the text"""
    return " ".join(x for x in row)
```

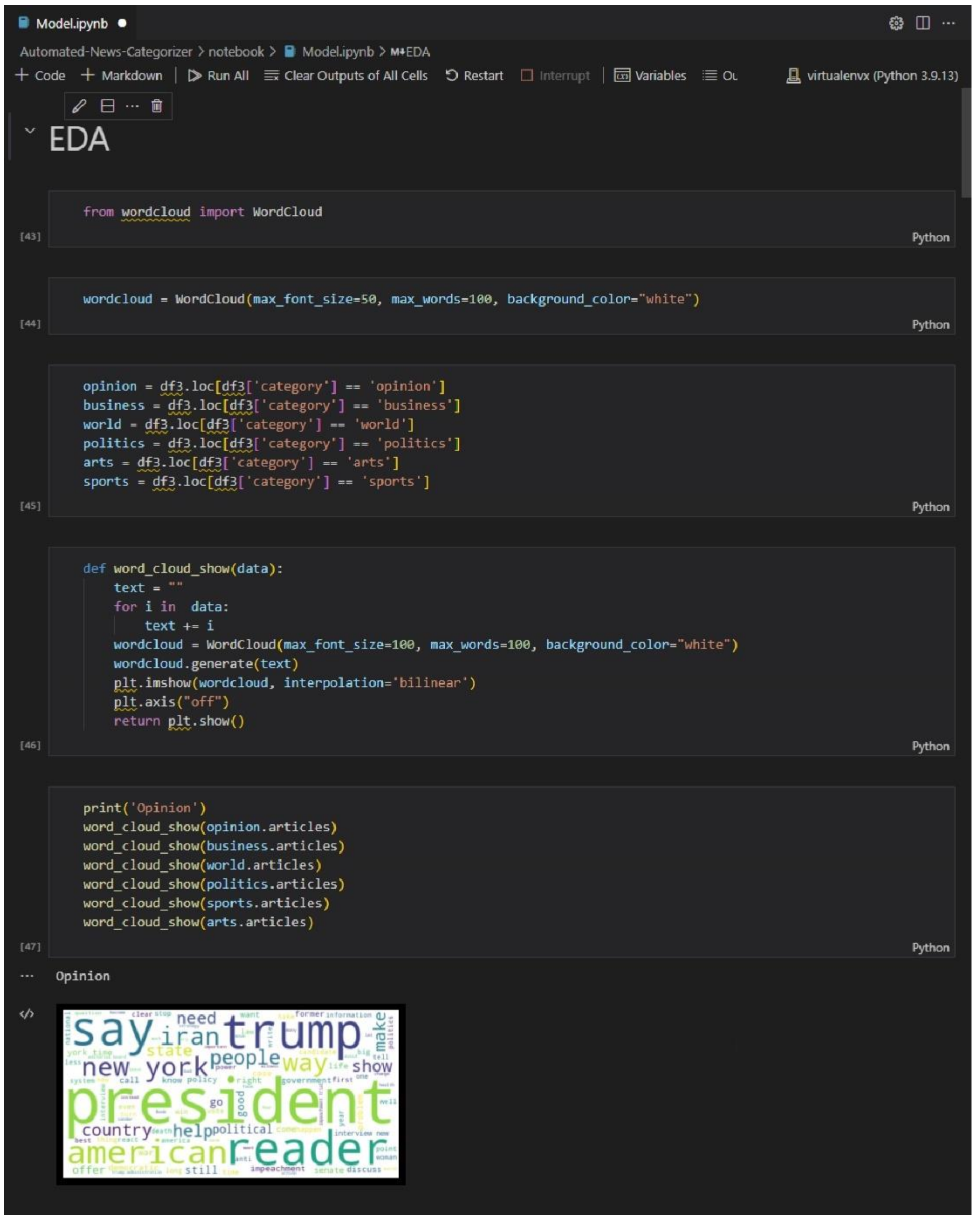
```
[41]
df3['articles'] = df3['articles'].apply(lowercase)
df3['articles'] = df3['articles'].apply(remove_special_char)
df3['articles'] = df3['articles'].apply(pos_taggin)
df3['articles'] = df3['articles'].apply(words_lemmatize)
df3['articles'] = df3['articles'].apply(stop_word)
df3['articles'] = df3['articles'].apply(join_words)
```

```
[42]
df3
```

	date	category id	category	headline	articles	url
0	2020-01-01	0	opinion	Protect Veterans From Fraud	congress much protect american serve count...	https://www.nytimes.com/2019/12/31/opinion/for...
1	2020-01-01	1	world	Hospitals and Schools Are Being Bombed in Syri...	attack civilian site syria grow frequent u...	https://www.nytimes.com/interactive/2019/12/31...
2	2020-01-01	1	world	Hong Kong Protesters Return to Streets as New ...	week relative calm end wednesday people mar...	https://www.nytimes.com/2020/01/01/world/asia/...
3	2020-01-01	2	politics	Dizzying Day for Trump Caps a Year Full of Them	president monitor middle east crisis golf ...	https://www.nytimes.com/2019/12/31/us/politics...
4	2020-01-01	3	arts	What's on TV Wednesday: A Linda Ronstadt Doc a...	linda ronstadt sound voice air cnn new s...	https://www.nytimes.com/2020/01/01/arts/televi...
...	...	...	...	...	...	...
2336	2020-01-31	4	business	Richard Plepler and Josh Tyrangiel May Revive ...	former hbo premium cable network talk fo...	https://www.nytimes.com/2020/01/31/business/me...
2337	2020-01-31	2	politics	Trump Hotel Patrons Relish Impeachment Finale	lobby president washington hotel supporter...	https://www.nytimes.com/2020/01/31/us/politics...
2338	2020-01-31	5	sports	Lakers Fall to Blazers on Emotional Night Hono...	lakers first game bryant daughter gianna ...	https://www.nytimes.com/2020/01/31/sports/bask...
2339	2020-01-31	5	sports	Alberto Salazar Is Suspended by SafeSport Afte...	famed running coach already bar sport yea...	https://www.nytimes.com/2020/01/31/sports/olym...
2340	2020-01-31	2	politics	Impeachment Trial Highlights: A Showdown Over ...	senator reject call additional witness pres...	https://www.nytimes.com/video/us/politics/1000...

2341 rows × 6 columns

Figure 18: Notebook of NLP.


Model.ipynb
Automated-News-Categorizer > notebook > Model.ipynb > M↓EDA
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Ou
virtualenvx (Python 3.9.13)
Python

EDA

```
[43]
from wordcloud import WordCloud
```

```
[44]
wordcloud = WordCloud(max_font_size=50, max_words=100, background_color="white")
```

```
[45]
opinion = df3.loc[df3['category'] == 'opinion']
business = df3.loc[df3['category'] == 'business']
world = df3.loc[df3['category'] == 'world']
politics = df3.loc[df3['category'] == 'politics']
arts = df3.loc[df3['category'] == 'arts']
sports = df3.loc[df3['category'] == 'sports']
```

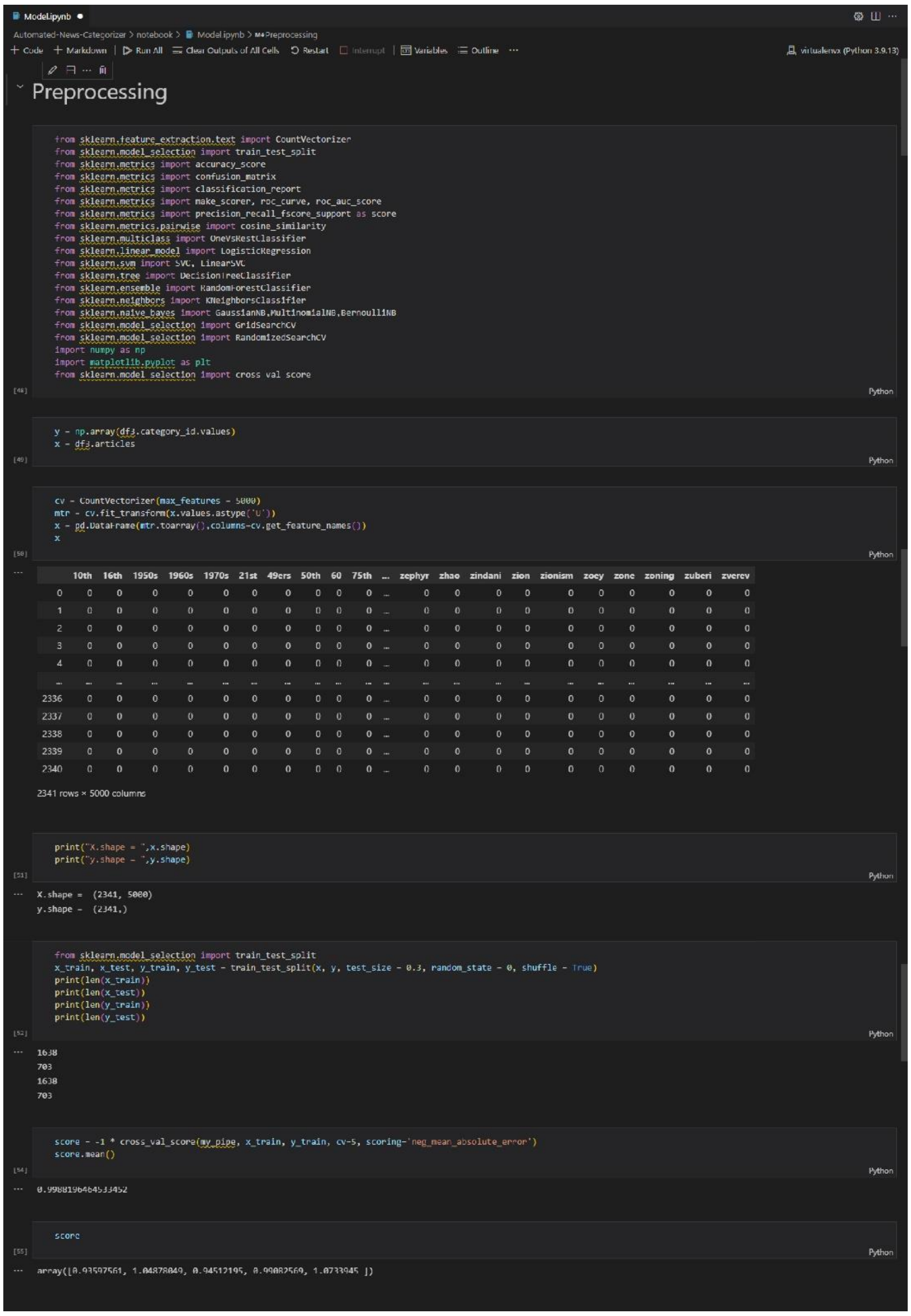
```
[46]
def word_cloud_show(data):
    text = ""
    for i in  data:
        text += i
    wordcloud = WordCloud(max_font_size=100, max_words=100, background_color="white")
    wordcloud.generate(text)
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    return plt.show()
```

```
[47]
print('Opinion')
word_cloud_show(opinion.articles)
word_cloud_show(business.articles)
word_cloud_show(world.articles)
word_cloud_show(politics.articles)
word_cloud_show(sports.articles)
word_cloud_show(arts.articles)
```

```
Opinion
```

Figure 19: Notebook of EDA.

4.1.2.3. Data Preprocessing


Model.ipynb
Automated-News-Categorizer > notebook > Model.ipynb > M↓Preprocessing
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ···
virtualenvx (Python 3.9.13)
Python

Preprocessing

```python
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import make_scorer, roc_curve, roc_auc_score
from sklearn.metrics import precision_recall_fscore_support as score
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.multiclass import OneVsRestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC, LinearSVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import cross_val_score
```
[48]

```python
y = np.array(df3.category_id.values)
x = df3.articles
```
[49]

```python
cv = CountVectorizer(max_features = 5000)
mtr = cv.fit_transform(x.values.astype('U'))
x = pd.DataFrame(mtr.toarray(),columns=cv.get_feature_names())
x
```
[50]

```
       10th  16th  1950s  1960s  1970s  21st  49ers  50th  60  75th  ...  zephyr  zhao  zindani  zion  zionism  zoey  zone  zoning  zuberi  zverev
   0      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
   1      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
   2      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
   3      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
   4      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
 ...    ...   ...    ...    ...    ...   ...    ...   ...  ..   ...  ...     ...   ...      ...   ...      ...   ...   ...     ...     ...     ...
2336      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
2337      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
2338      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
2339      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0
2340      0     0      0      0      0     0      0     0   0     0  ...       0     0        0     0        0     0     0       0       0       0

2341 rows × 5000 columns
```

```python
print("X.shape = ",x.shape)
print("y.shape = ",y.shape)
```
[51]

```
X.shape =  (2341, 5000)
y.shape =  (2341,)
```

```python
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 0, shuffle = True)
print(len(x_train))
print(len(x_test))
print(len(y_train))
print(len(y_test))
```
[52]

```
1638
703
1638
703
```

```python
score = -1 * cross_val_score(my_pipe, x_train, y_train, cv=5, scoring='neg_mean_absolute_error')
score.mean()
```
[54]

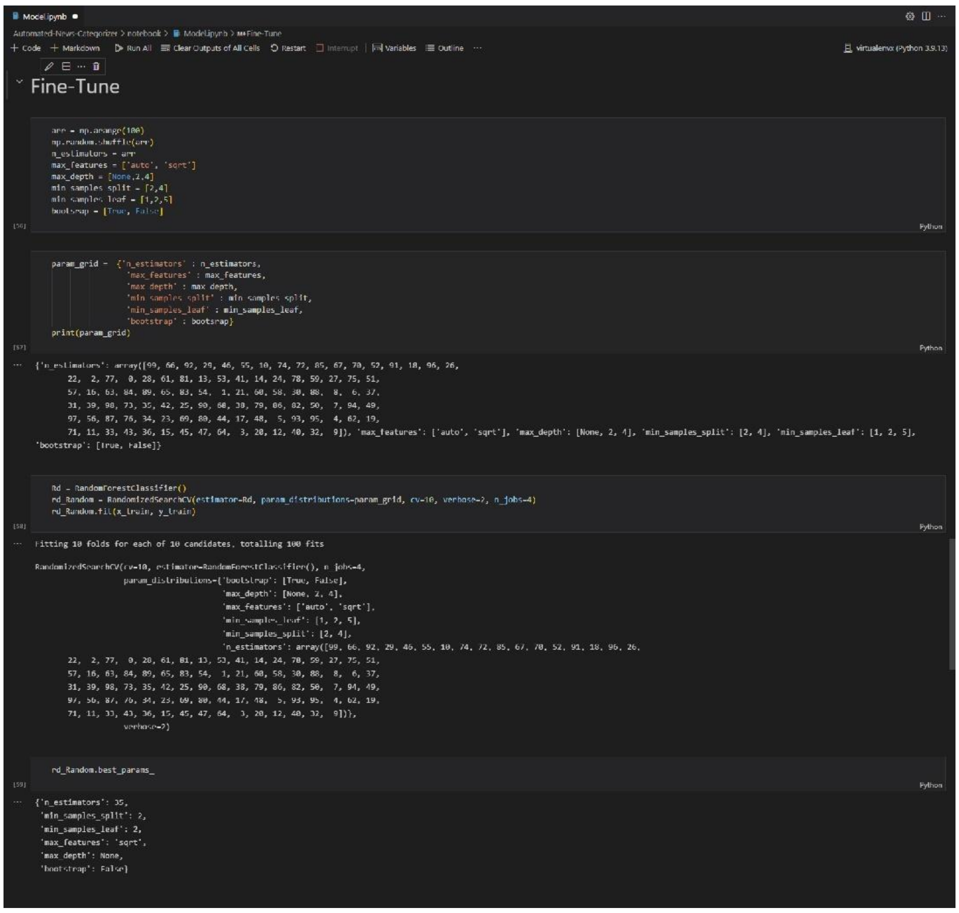
```
0.9988196464533452
```

```python
score
```
[55]

```
array([0.93597561, 1.04878049, 0.94512195, 0.99082569, 1.0733945 ])
```

Figure 20: Notebook of Preprocessing.

4.1.2.4. Modeling


Model.ipynb
Automated-News-Categorizer > notebook > Model.ipynb > M+Fine-Tune
+ Code
+ Markdown
Run All
Clear Outputs of All Cells
Restart
Interrupt
Variables
Outline
virtualenvx (Python 3.9.13)
Fine-Tune
Python

```
[56]
arr = np.arange(100)
np.random.shuffle(arr)
n_estimators = arr
max_features = ['auto', 'sqrt']
max_depth = [None,2,4]
min samples split = [2,4]
min samples leaf = [1,2,5]
bootsrap = [True, False]
```

```
[57]
param_grid = {'n_estimators' : n_estimators,
               'max_features' : max_features,
               'max depth' : max depth,
               'min samples split' : min samples split,
               'min_samples_leaf' : min_samples_leaf,
               'bootstrap' : bootsrap}
print(param_grid)
```

```
{'n_estimators': array([99, 66, 92, 29, 46, 55, 10, 74, 72, 85, 67, 70, 52, 91, 18, 96, 26,
       22,  2, 77,  0, 28, 61, 81, 13, 53, 41, 14, 24, 78, 59, 27, 75, 51,
       57, 16, 63, 84, 89, 65, 83, 54,  1, 21, 60, 58, 30, 88,  8,  6, 37,
       31, 39, 98, 73, 35, 42, 25, 90, 68, 38, 79, 86, 82, 50,  7, 94, 49,
       97, 56, 87, 76, 34, 23, 69, 80, 44, 17, 48,  5, 93, 95,  4, 62, 19,
       71, 11, 33, 43, 36, 15, 45, 47, 64,  3, 20, 12, 40, 32,  9]), 'max_features': ['auto', 'sqrt'], 'max_depth': [None, 2, 4], 'min_samples_split': [2, 4], 'min_samples_leaf': [1, 2, 5],
'bootstrap': [True, False]}
```

```
[58]
Rd = RandomForestClassifier()
rd_Random = RandomizedSearchCV(estimator=Rd, param_distributions=param_grid, cv=10, verbose=2, n_jobs=4)
rd_Random.fit(x_train, y_train)
```

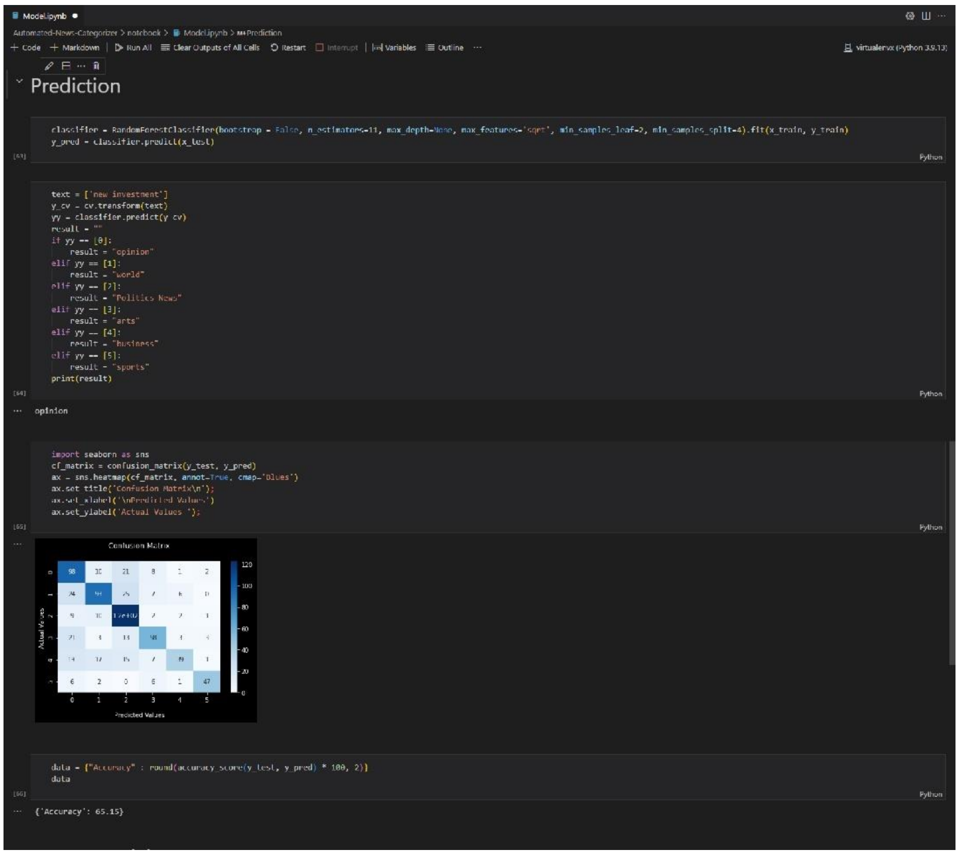
```
Fitting 10 folds for each of 10 candidates, totalling 100 fits

RandomizedSearchCV(cv=10, estimator=RandomForestClassifier(), n_jobs=4,
                   param_distributions={'bootstrap': [True, False],
                                        'max_depth': [None, 2, 4],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [1, 2, 5],
                                        'min_samples_split': [2, 4],
                                        'n_estimators': array([99, 66, 92, 29, 46, 55, 10, 74, 72, 85, 67, 70, 52, 91, 18, 96, 26,
       22,  2, 77,  0, 28, 61, 81, 13, 53, 41, 14, 24, 78, 59, 27, 75, 51,
       57, 16, 63, 84, 89, 65, 83, 54,  1, 21, 60, 58, 30, 88,  8,  6, 37,
       31, 39, 98, 73, 35, 42, 25, 90, 68, 38, 79, 86, 82, 50,  7, 94, 49,
       97, 56, 87, 76, 34, 23, 69, 80, 44, 17, 48,  5, 93, 95,  4, 62, 19,
       71, 11, 33, 43, 36, 15, 45, 47, 64,  3, 20, 12, 40, 32,  9])},
                   verbose=2)
```

```
[59]
rd_Random.best_params_
```

```
{'n_estimators': 35,
 'min_samples_split': 2,
 'min_samples_leaf': 2,
 'max_features': 'sqrt',
 'max depth': None,
 'bootstrap': False}
```

Figure 21: Notebook of Fine-Tune.



Model.ipynb

Automated-News-Categorizer > notebook > Model.ipynb > Prediction

Prediction

```python
classifier = RandomForestClassifier(bootstrap = False, n_estimators=11, max_depth=None, max_features='sqrt', min_samples_leaf=2, min_samples_split=4).fit(x_train, y_train)
y_pred = classifier.predict(x_test)
```

```python
text = ['new investment']
y_cv = cv.transform(text)
yy = classifier.predict(y_cv)
result = ""
if yy == [0]:
    result = "opinion"
elif yy == [1]:
    result = "world"
elif yy == [2]:
    result = "Politics News"
elif yy == [3]:
    result = "arts"
elif yy == [4]:
    result = "business"
elif yy == [5]:
    result = "sports"
print(result)
```

```
opinion
```

```python
import seaborn as sns
cf_matrix = confusion_matrix(y_test, y_pred)
ax = sns.heatmap(cf_matrix, annot=True, cmap='Blues')
ax.set_title('Confusion Matrix\n');
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values ');
```

```python
data = {"Accuracy" : round(accuracy_score(y_test, y_pred) * 100, 2)}
data
```

```
{'Accuracy': 65.15}
```

Figure 22: Notebook of Prediction.

4.1.2.5. Deployment

4.2. TESTING

4.3. CONCLUSION

CHAPTER FIVE

CONCLUSION

5.0. INTRODUCTION

5.1. RESULT

5.2. RECOMMENDATION

5.3. CONCLUTION