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

Adviser: Kholed Langsari

ABSTARCT

In recent years many news companies are trying to reduce paper 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.

The experiment will pull data from the New York Time API and divide the dataset into two parts: the first is the training set and the second is the testing set to be used to train model machine learning. The model used in this project is the random Forest Classifier model. Experimental results from the Random Forest Classifier model gave acceptable results with 65.5% accuracy.

หัวข้อ: การจำแนกหมวดหมู่ข่าวด้วยปัญญาประดิษฐ์

ผู้เขียน: มัรวาน ยีเด็ง 611431018, นิเฮง แม 601431012

สาขา: เทคโนโลยีสารสนเทศ

ปีการศึกษา: 2565

ที่ปรึกษาโครงงานวิจัย: อาจารย์คอลิด ลังสารี

บทคัดย่อ

ในช่วงไม่กี่ปีที่ผ่านมา บริษัทข่าวหลายแห่งพยายามลดการใช้กระดาษโดยใช้สื่อออนไลน์และแทนที่การพิมพ์หนังสือพิมพ์และบทความแบบเดิมๆ ด้วยเหตุนี้จึงมีบทความข่าวต่างๆ อย่างไรก็ตาม การจัดหมวดหมู่บทความข่าวตามหมวดหมู่ที่เกี่ยวข้องด้วยตนเองนั้นยากและใช้เวลานาน การจัดหมวดหมู่บทความข่าวโดยอัตโนมัติสามารถเป็นประโยชน์ต่อบริษัทข่าวและสังคมในหลายๆ ด้าน

การทดลองจะดึงข้อมูลจาก New York Time API และแบ่งชุดข้อมูลออกเป็นสองส่วน: ส่วนแรกคือชุดการฝึก และชุดที่สองคือชุดทดสอบที่จะใช้ในการฝึกโมเดล โดยโมเดลที่ใช้ในงานวิจัยนี้จะเป็นแบบจำลอง Random Forest Classifier ผลการทดลองจากแบบจำลอง Random Forest Classifier ให้ผลลัพธ์ที่ยอมรับได้ด้วยความแม่นยำ 65.5%

TABLE OF CONTENTS

Pages

ABSTARCT.....III

ABSTARCT (Translation).....IV

TABLE OF CONTENTS.....V

LIST OF ILLUSTRATION.....VII

LIST OF TABLES.....VIII

CHAPTER ONE.....1

INTRODUCTION.....1

1.0. PROJECT OVERVIEW.....1

1.1. PROBLEM STATEMENTS.....1

1.2. OBJECTIVE.....2

1.3. SIGNIFICANCE OF STUDY.....2

1.4. SCOPE OF STUDY.....3

1.5. SIGNIFICANCE OF STUDY.....3

1.6. CONCLUSION.....4

CHAPTER TWO.....6

LITERATURE REVIEW.....6

2.0. INTRODUCTION.....6

2.1. DEFINITION.....6

2.2. TOOLS USED.....7

2.3. RELATED WORK.....13

2.4. CONCLUSION.....16

CHAPTER THREE 17
METHODOLOGY 17
3.0. INTRODUCTION 17
3.1. BUSINESS OBJECTIVE 18
3.2. DATA COLLECTION 19
3.3. DATA CLEANSING 20
3.4. NATURAL LANGUAGE PROCESSING 23
3.5. EXPLORATORY DATA ANALYSIS 29
3.6. DATA PREPOCESSING 30
3.7. MODELING 31
3.8. DEPLOYMENT 36
3.9. CONCLUSION 36
CHAPTER FOUR 38
FINDING AND IMPLEMENTATION 38
4.0. INTRODUCTION 38
4.1. DATA COLLECTION 38
4.2. DATA CLEANSING 42
4.3. NATURAL LANGUAGE PROCESSING 50
4.4. EXPLORATORY DATA ANALYSIS 56
4.5. DATA PREPROCESSING 58
4.6. MODELING 61
4.7. DEPLOYMENT 65
4.8. CONCLUSION 68
CHAPTER FIVE 69
CONCLUSION 69
5.0. INTRODUCTION 69

5.1. RESULT.....69
5.2. RECOMMENDATION 70
REFERENCES 71

LIST OF ILLUSTRATION

FIGURE 1: CROSS INDUSTRY STANDARD PROCESS FOR DATA MINING (CRISP-DM).....17
FIGURE 2: PROJECT METHODOLOGY FLOWCHART. 18
FIGURE 3: DATA COLLECTION FLOWCHART. 19
FIGURE 4: THE SUMMARY OF EACH COLUMN IN THE DATA FRAME. 20
FIGURE 5: DIAGRAM OF DEALING WITH MISSING DATA PROCESS. 21
FIGURE 6: DIAGRAM OF SPECIAL CHARACTER REMOVING PROCESS. 22
FIGURE 7: DIAGRAM OF NLP PROCESS. 24
FIGURE 8: DIAGRAM OF NLP PROCESS. 25
FIGURE 9: LIST OF STOP WORDS 20 WORDS. 27
FIGURE 10: DIAGRAM OF STOPWORDS PROCESS. 28
FIGURE 11: DIAGRAM OF PREDICTION PROCESS. 33
FIGURE 12: DIAGRAM OF PREDICTION PROCESS. 34
FIGURE 13: WEB APPLICATION PROCESS. 36
FIGURE 14: DATA FRAME OF DATA COLLECTION. 41
FIGURE 15: DATA FRAME OF COUNT VALUES. 43
FIGURE 16: DATA FRAME OF FACTORIZE. 44

FIGURE 17: THE SUMMARY OF THE DATA FRAME IN THE CATEGORY SELECTION SECTION. 44
FIGURE 18: DATA FRAME AFTER CLEANSING. 44
FIGURE 19: WORD CLOUD THAT VISUALIZE ALL THE TEXT OF NEWS ARTICLES IN FORM OF WORD CLOUD. 58
FIGURE 20: COUNT VECTORIZER. 60
FIGURE 21: THE OUTPUT AMOUNT OF THE TRAIN AND TEST DATASET AFTER SPLITTING. 61
FIGURE 22: THE OUTPUT OF BEST HYPERPARAMETERS FROM FINE-TUNE. 62
FIGURE 23: MODEL ACCURACY USING CONFUSION MATRIX TABLE. 64
FIGURE 24: IMAGE OF NEWS CATEGORIZER WEB APPLICATION. 66

LIST OF TABLES

TABLE 1: TABLE RANDOMIZED-SEARCH HYPERPARAMETERS TUNING. 32
TABLE 2: COMPARING BEFORE AND AFTER CLEAR MISSING VALUES.... 47
TABLE 3: COMPARING BEFORE AND AFTER LOWERCASE. 48
TABLE 4: COMPARING BEFORE AND AFTER REMOVE SPECIAL CHARACTERS. 50
TABLE 5: COMPARING BEFORE AND WORD TOKENIZATION. 51
TABLE 6: COMPARING BEFORE AND AFTER STOPWORDS REMOVING. ... 52
TABLE 7: COMPARING BEFORE AND AFTER WORDS LEMMATIZATION. . 54

TABLE 8: COMPARING BEFORE AND AFTER POST-TAGGING.....55

TABLE 9: COMPARING BEFORE AND AFTER JOIN WORDS.....56

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. (*What Is Natural Language Processing? | IBM*, n.d.), (*Sklearn.Ensemble.RandomForestClassifier — Scikit-Learn 1.0.2 Documentation*, n.d.)

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 are 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. SIGNIFICANCE OF STUDY

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 many various news articles in the news website database.

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

2.1. DEFINITION

2.3.1. 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 & News, 2014)

2.3.1. 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)

2.3.1. 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)

2.3.1. 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)

2.3.1. 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 chapter there are three main parts to be found in this second chapter: Definition, Tools used, and Related works. The definition describes the processed 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 work describes the articles referenced 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 & Hotz, 2020)

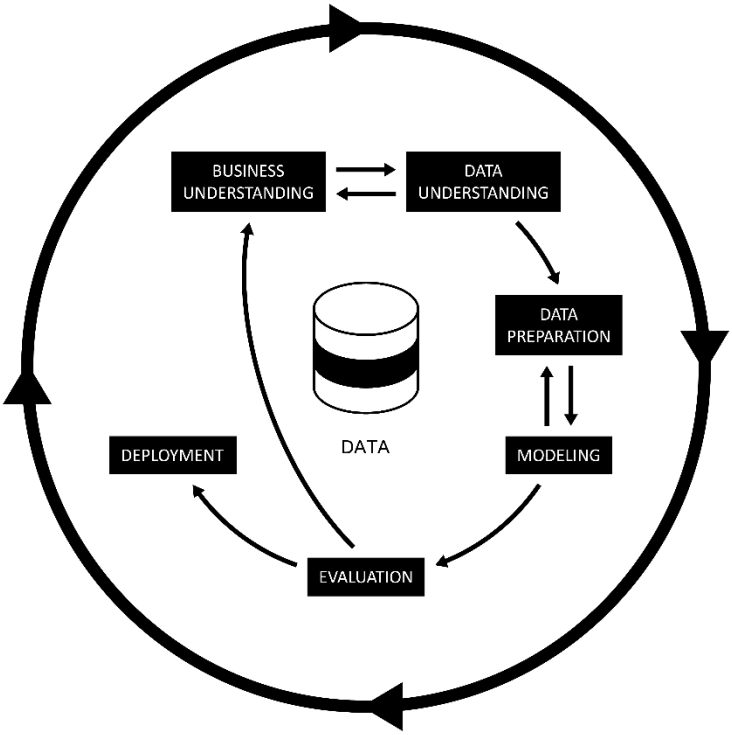


Figure 1: CRoss Industry Standard Process for Data Mining (CRISP-DM)

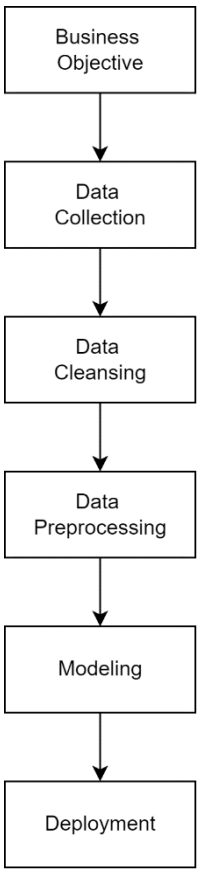


Figure 2: Project Methodology Flowchart.

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

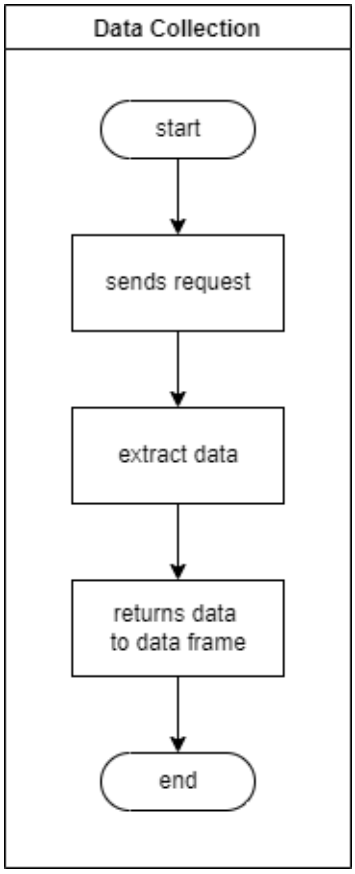


Figure 3: Data Collection Flowchart.

The 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. (*What Is an API? - Application Programming Interfaces Explained - AWS*, n.d.)

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. (*Knowledge__DataCategorySelection | Salesforce Knowledge Developer Guide | Salesforce Developers*, n.d.)

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.

```
 #    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
```

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.

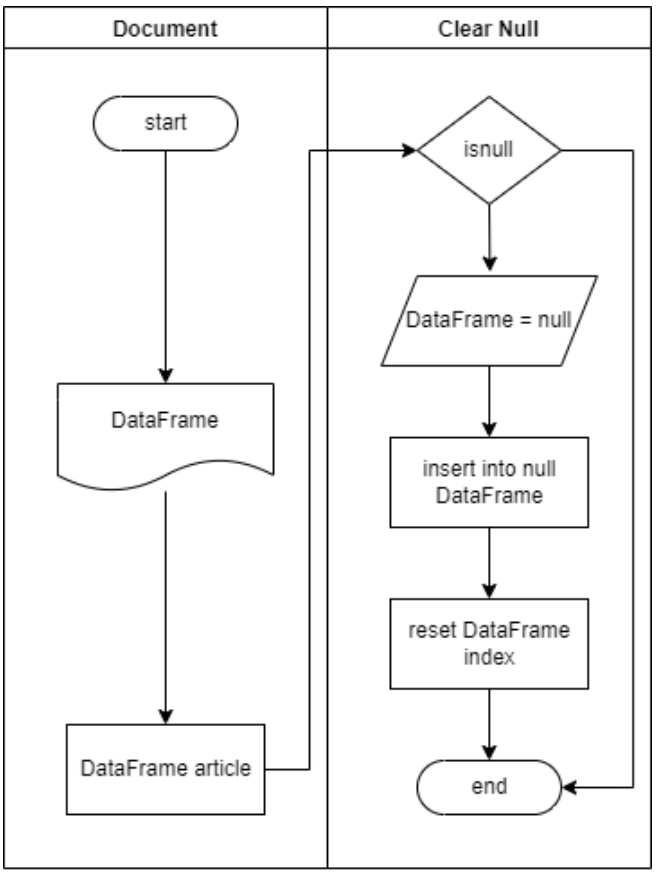


Figure 5: Diagram of Dealing With Missing Data Process.

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. (*Top Techniques to Handle Missing Values Every Data Scientist Should Know | DataCamp*, n.d.)

3.3.3. 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. (*Lowercase in Python Tutorial | DataCamp*, n.d.)

3.3.4. WRAPPING UNNECESSARY CHARACTERS

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

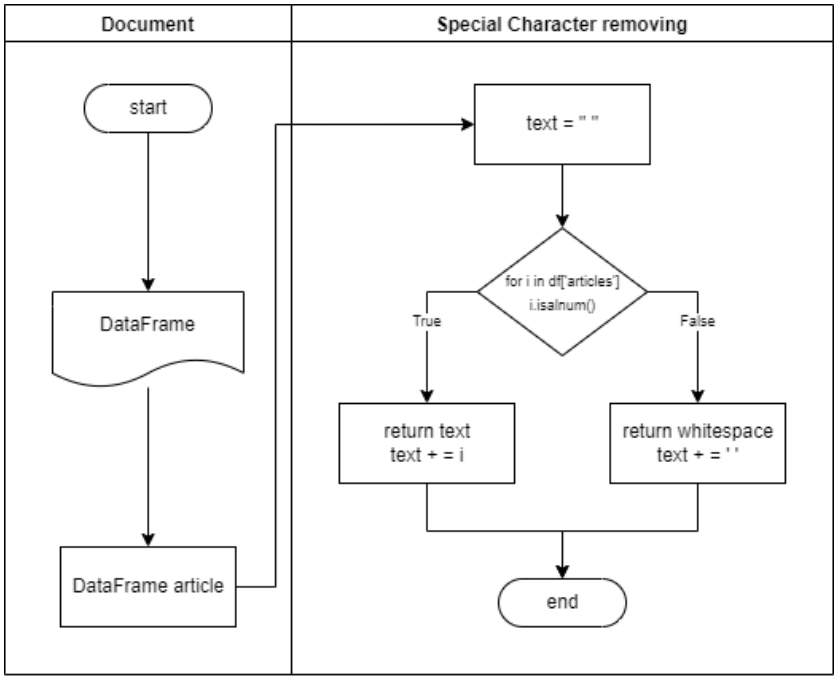


Figure 6: Diagram of Special Character Removing Process.

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. (*Python String Isalnum() | DigitalOcean*, n.d.)

3.4. NATURAL LANGUAGE PROCESSING

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

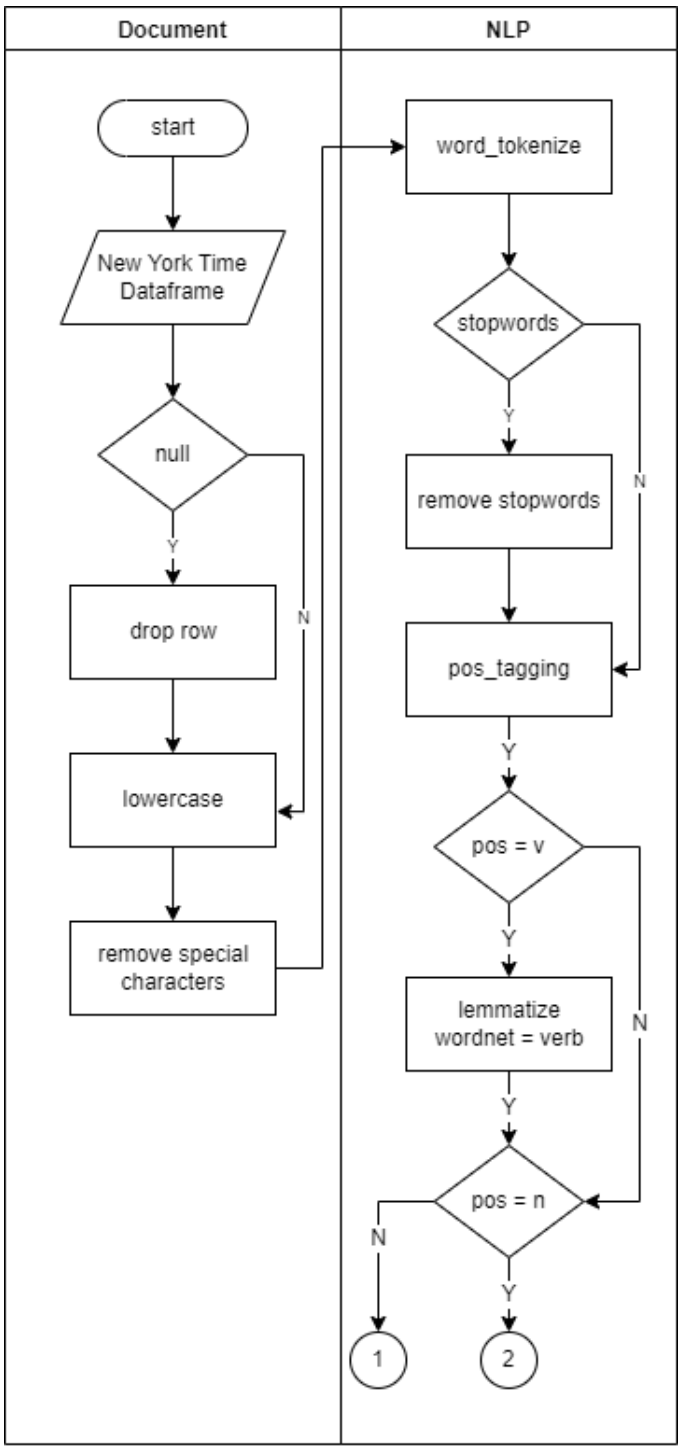


Figure 7: Diagram of NLP Process.

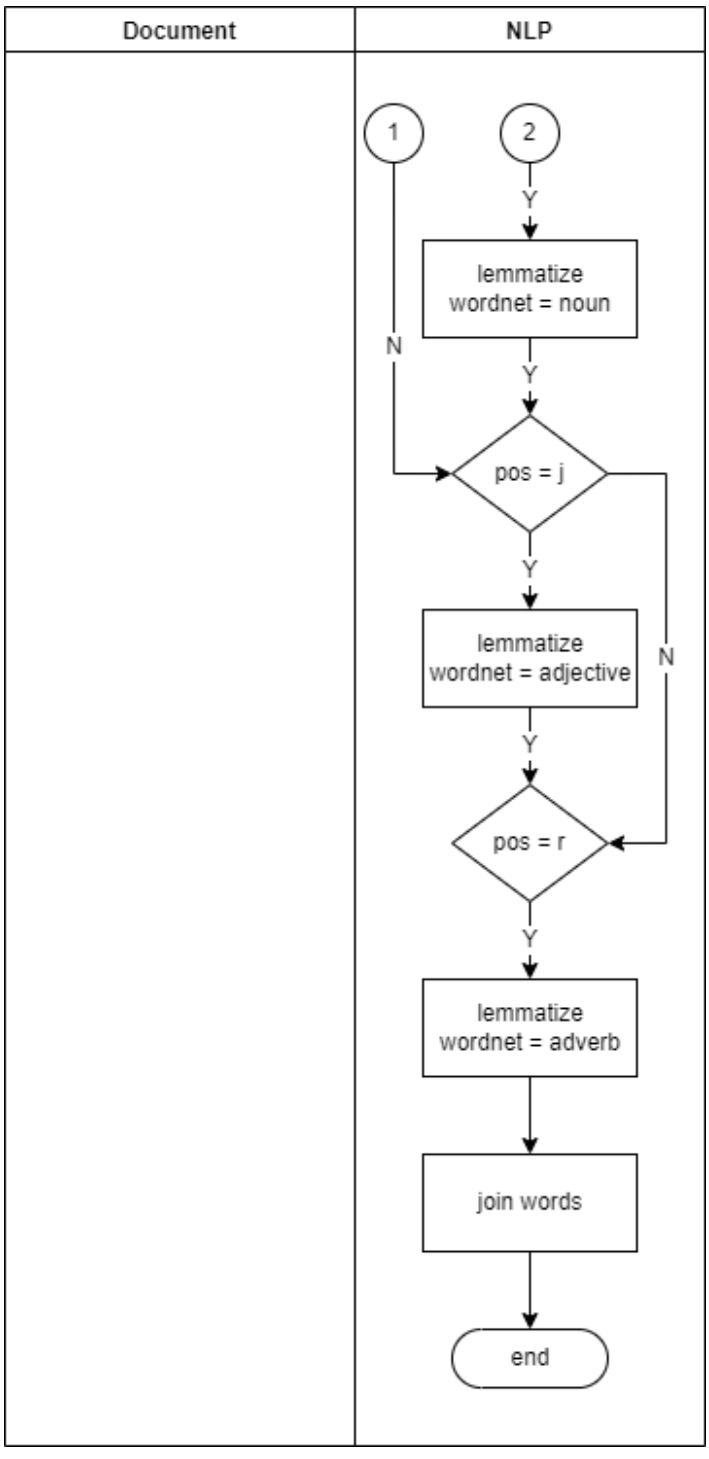


Figure 8: Diagram of NLP Process.

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. (*What Is Natural Language Processing? | IBM*, n.d.)

3.4.1. WORD TOKENIZATION

Is to cut out the words from the sentence one by one using nltk.tokenize package. (*NLTK Tokenize: How to Tokenize Words and Sentences with NLTK? - Holistic SEO*, n.d.)

3.4.2. STOPWORDS

3.4.2.1. 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. (*NLTK Tokenize: How to Tokenize Words and Sentences with NLTK? - Holistic SEO*, n.d.)

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

Figure 9: List of Stop Words 20 Words.

3.4.2.2. Stopwords removing

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

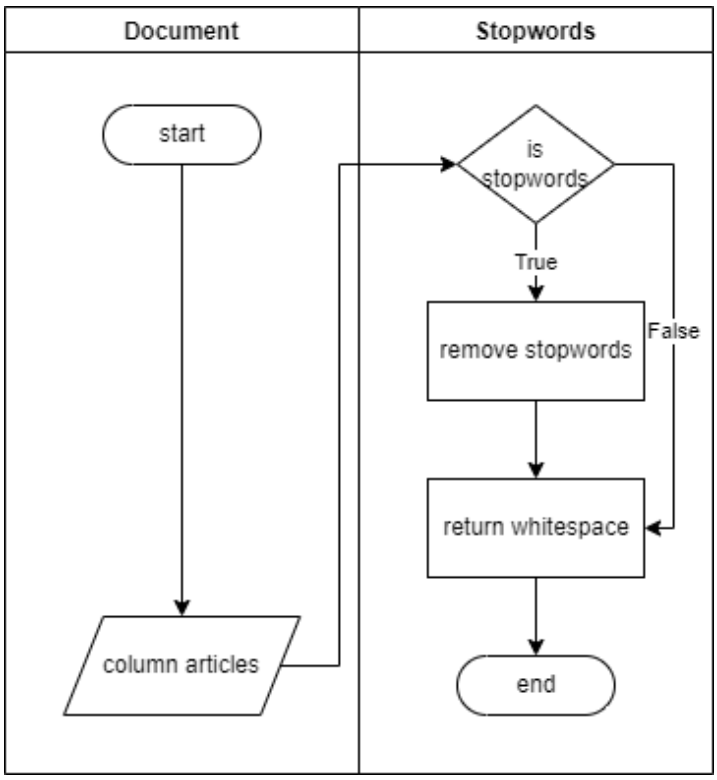


Figure 10: Diagram of Stopwords Process.

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. (*NLTK Tokenize: How to Tokenize Words and Sentences with NLTK? - Holistic SEO*, n.d.)

3.4.3. WORDS LEMMATIZATION

3.4.3.1. 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.

(*Understanding Part-of-Speech Tagging in NLP: Techniques and Applications - Shiksha Online*, n.d.)

3.4.3.2. 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. (*Stemming and Lemmatization in Python | DataCamp*, n.d.)

3.5. 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. (*What Is Exploratory Data Analysis? | IBM*, n.d.)

3.6.1. 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. (*Python Word Clouds Tutorial: How to Create a Word Cloud | DataCamp*, n.d.)

3.6. DATA PREPOCESSING

3.6.1. 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. (*Pandas Convert Column to Numpy Array - Spark By {Examples}*, n.d.)

3.6.2. 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. (*6.2. Feature Extraction — Scikit-Learn 1.2.2 Documentation*, n.d.)

3.6.3. 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. (*Sklearn.Model_selection.Train_test_split — Scikit-Learn 1.2.2 Documentation*, n.d.)

3.7. MODELING

3.7.1. FINE-TUNE

3.7.3.1. Fine-Tuning

Is the process in which parameters of a model must be adjusted very precisely in order to fit with certain observations. (*What Is Fine-Tuning and How Does It Work in Neural Networks?*, n.d.)

3.7.3.2. 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. (*Using Random Search to Optimize Hyperparameters | Engineering Education (EngEd) Program | Section*, n.d.)

3.7.3.3. 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.

3.7.2. PREDICTION

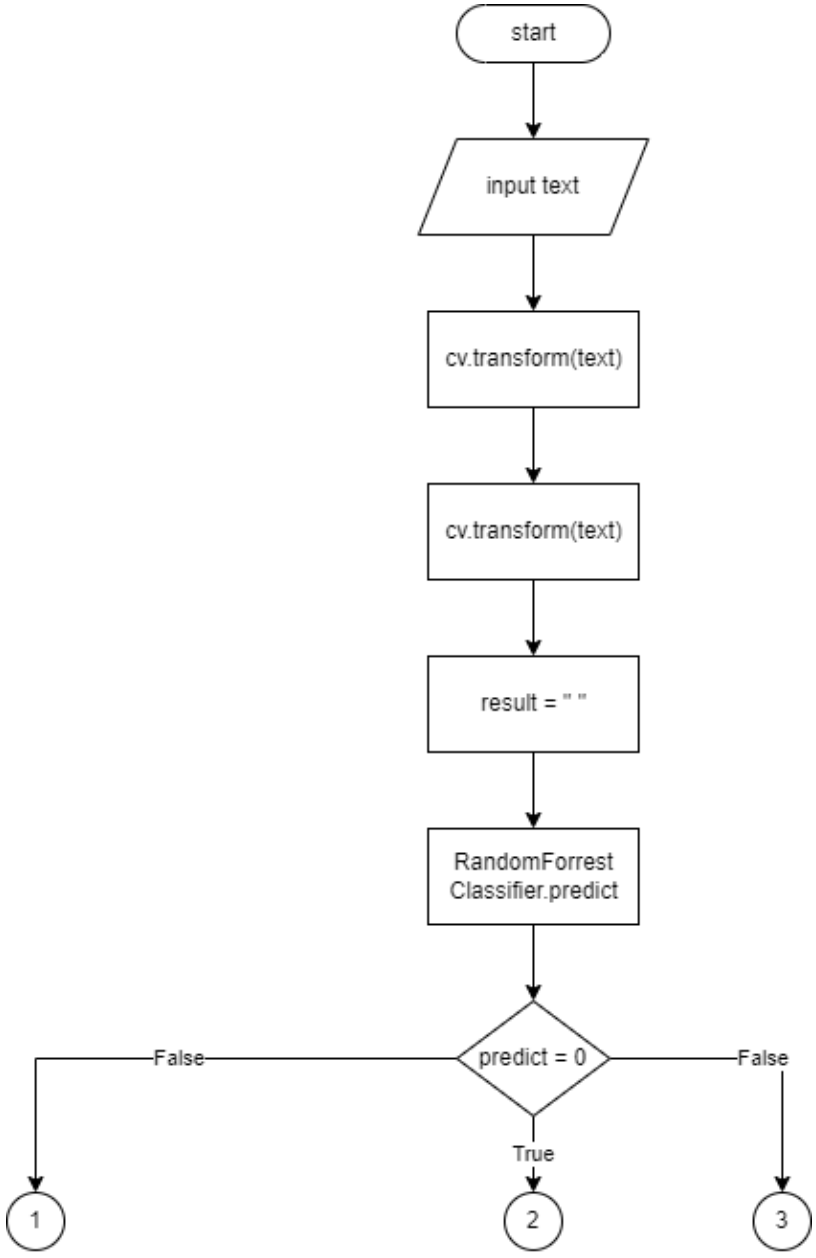


Figure 11: Diagram of Prediction Process.

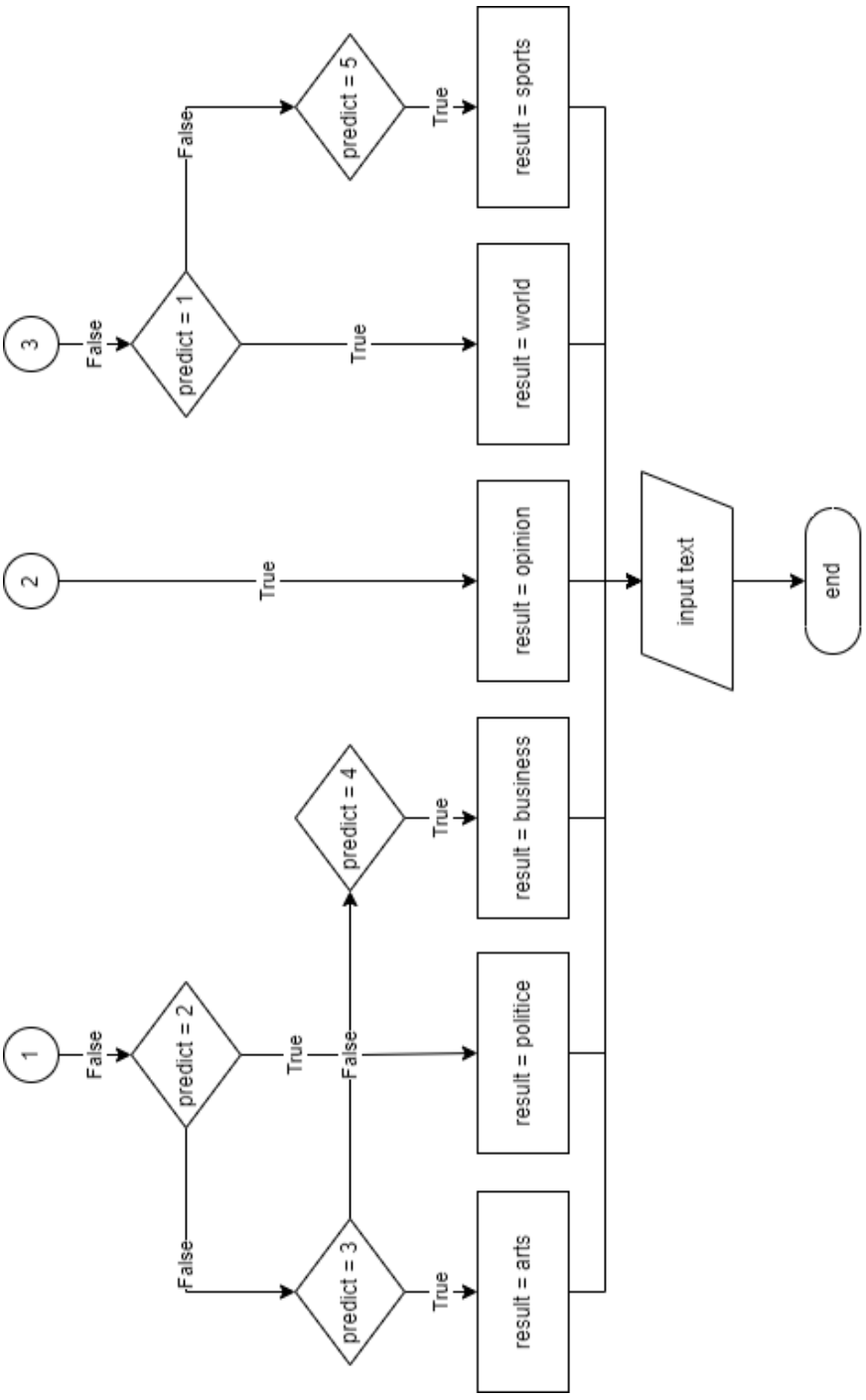


Figure 12: Diagram of Prediction Process.

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”. (*What Is Predictive Analytics? | IBM*, n.d.)

3.7.3. ACCURACY SCORE

3.7.3.1. 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. (*Sklearn.Metrics.ConfusionMatrixDisplay — Scikit-Learn 1.2.2 Documentation*, n.d.)

3.8. DEPLOYMENT

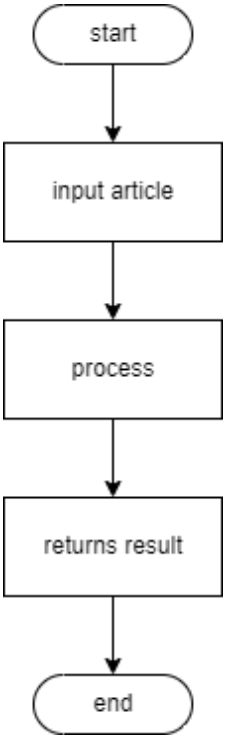


Figure 13: Web Application Process.

The deployment phase focused on deploying the model to the web application by using Flask Micro Framework. The web application work starts from input some article news after that the web application will process that article for a moment the web application will show the type of news from the article. (*How to Build a Web Application Using Flask and Deploy It to the Cloud*, n.d.)

3.9. CONCLUSION

Our project uses crisp-dm to lay out the project's build process, which includes Business understanding. It will be the process of defining the problem of the project wherein our project will have to create a model to classify the news and later on will

be Data understanding. It will be the data collection process. Our project will collect information about news from online news sources of the New York Times by fetching the data via API. Next is the Data Preparation process which is data cleaning. The transformation of the data cleanup involves selecting the desired data, deleting the lost data, converting the label data to numbers and performing the features extraction using the count vectorizer. Next is the Modeling process, creating a model. By the project we will use random forest classification is a model for classification and evaluation process. We use a confusion matrix to measure the model's performance and finally deployment. We use a pickle, a python built-in library, to export the model and create a user interface page using the flask framework. to create web applications for use.

CHAPTER FOUR

FINDING AND IMPLEMENTATION

4.0. INTRODUCTION

This chapter discusses the finding and implementation of this project. It can be divided into six main parts following the CRISP-DM. (Saltz & Hotz, 2020)

4.1. DATA COLLECTION

```
import json
import time
import requests
import datetime
import dateutil
import pandas as pd
from dateutil.relativedelta import relativedelta
```

The code shown imported modules used for this part.

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

The image shown is a variable for storing list of year and date for API request for use in send_request() function.

```
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
```

The code shown is function for sending API request to New York time api sever and response back as json. The function takes date variable as an argument to recreate new url for sending to api server after that response as json by using json() module. We delay each response for every 6 seconds by using time.sleep(6).

```
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)
```

The code shown is function for parsing responded api in form of json into Pandas DataFrame.

```
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]
```

The code is text extraction for extracting specific labels from url. the result from the function is the labels of each news category that is contained in news url. We have

used python built-in function str.replace() to get specific string in url and remove the rest.

```
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)
```

The code shown is DataFrame exporting to csv using to_csv() from pandas module.

	date	url	headline	articles	doc_type	material_type	section	keywords	label
0	2020-01-01	https://www.nytimes.com/2019/12/31/us/texas-ch...	'Battling a Demon': Drifter Sought Help Before...	The gunman who shot two parishioners at the We...	article	News	None	[Churches (Buildings), Murders, Attempted Murd...	us
1	2020-01-01	https://www.nytimes.com/2019/12/31/opinion/for...	Protect Veterans From Fraud	Congress could do much more to protect America...	article	Editorial	None	[Veterans, For-Profit Schools, Financial Aid (...	opinion
2	2020-01-01	https://www.nytimes.com/2019/12/31/health/e-ci...	F.D.A. Plans to Ban Most E-Cigarette Flavors b...	The tobacco and vaping industries and conserva...	article	News	None	[E-Cigarettes, Recalls and Bans of Products, M...	health
3	2020-01-01	https://www.nytimes.com/2019/12/31/crosswords/...	'It's Green and Slimy'	Christina Iverson and Jeff Chen ring in the Ne...	article	News	None	[Crossword Puzzles]	crosswords
4	2020-01-01	https://www.nytimes.com/2019/12/31/pageoneplus...	Corrections: Jan. 1, 2020	Corrections that appeared in print on Wednesda...	article	Correction	None	[]	us
...	...	...	...	...	...	...	...	...	...
4475	2020-01-31	https://www.nytimes.com/2020/01/31/sports/bask...	Lakers Fall to Blazers on Emotional Night Hono...	It was the Lakers' first game since Bryant and...	article	News	None	[Basketball]	sports
4476	2020-01-31	https://www.nytimes.com/2020/01/31/sports/olym...	Alberto Salazar Is Suspended by SafeSport Afte...	The famed running coach was already barred fro...	article	News	None	[Running, Coaches and Managers]	sports
4477	2020-01-31	https://www.nytimes.com/2020/01/31/health/cpr-...	CPR, by Default	When very old patients suffer cardiac arrest, ...	article	News	None	[Hospitals, Defibrillators, Living Wills and H...	health
4478	2020-01-31	https://www.nytimes.com/video/us/politics/1000...	Impeachment Trial Highlights: A Showdown Over ...	Senators rejected a call for additional witnes...	multimedia	Video	None	[Impeachment, Trump-Ukraine Whistle-blower Com...	politics
4479	2020-01-31	https://www.nytimes.com/2020/01/31/nyregion/pr...	Battle Lines Quickly Form Over Radical Propert...	A New York City commission's recommendations f...	article	News	None	[Property Taxes, Real Estate and Housing (Resi...	nyregion

4480 rows × 9 columns

Figure 14: Data Frame of Data Collection.

(*Using New York Times API and Jq to Collect News Data | by Dana Lindquist | Medium*, n.d.)

4.2. DATA CLEANSING

4.2.1. CATEGORY SELECTION

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

The code shown imported modules used for this part.

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

The code shown is code for selecting specific columns from DataFrame and store as new DataFrame.

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

The code shown is for selecting values that contains only "sport", "opinion", "world", "politics", "business", "arts" in column "label".

```
def count_values(dataframe ,a):
    """Counting the total numbers of the specific lebel"""

    label = []
    u_names = []
    count = []
    for i in dataframe[a]:
        label.append(i)
    for names in label:
        if names not in u_names:
            names = str(names)
            u_names.append(names)
    for num in u_names:
        count.append(label.count(num))
    return pd.DataFrame({'Label': u_names, 'Numbers': count})
```

The code shown is a function for counting specific values in DataFrame. The code works by looping through all rows in specific DataFrame column then append each value into list “label” after that the code will loop through list “label” and check if the value is not in list “u_names” it will append the value into list “u_names”.

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

Figure 15: Data Frame of Count Values.

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

The code shown is for factorize each value in column "label" into number and store it as new column named "category_id".

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

Figure 16: Data Frame of Factorize.

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

The code shown is selecting specific columns in DataFrame.

```
df1.info()
```

The code show is print the full summary.

```
<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 17: The summary of the data frame in the Category Selection section.

	date	category_id	category	headline	articles	url
0	2020-01-01	0	opinion	Protect Veterans From Fraud	Congress could do much more to protect America...	https://www.nytimes.com/2019/12/31/opinion/for...
1	2020-01-01	1	world	Hospitals and Schools Are Being Bombed in Syri...	Attacks on civilian sites in Syria have grown ...	https://www.nytimes.com/interactive/2019/12/31...
2	2020-01-01	1	world	Hong Kong Protesters Return to Streets as New ...	Weeks of relative calm ended on Wednesday, as ...	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	The president monitored a Middle East crisis f...	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: The Sound of My Voice" airs o...	https://www.nytimes.com/2020/01/01/arts/televi...
...	...	...	...	...	...	...
2343	2020-01-31	4	business	Richard Plepler and Josh Tyrangiel May Revive ...	The former boss of the premium cable network i...	https://www.nytimes.com/2020/01/31/business/me...
2344	2020-01-31	2	politics	Trump Hotel Patrons Relish Impeachment Finale	In the lobby of the president's Washington hot...	https://www.nytimes.com/2020/01/31/us/politics...
2345	2020-01-31	5	sports	Lakers Fall to Blazers on Emotional Night Hono...	It was the Lakers' first game since Bryant and...	https://www.nytimes.com/2020/01/31/sports/bask...
2346	2020-01-31	5	sports	Alberto Salazar Is Suspended by SafeSport Afte...	The famed running coach was already barred fro...	https://www.nytimes.com/2020/01/31/sports/olym...
2347	2020-01-31	2	politics	Impeachment Trial Highlights: A Showdown Over ...	Senators rejected a call for additional witnes...	https://www.nytimes.com/video/us/politics/1000...

2348 rows × 6 columns

Figure 18: Data Frame After Cleansing.

(*Knowledge__DataCategorySelection | Salesforce Knowledge Developer Guide | Salesforce Developers*, n.d.)

4.2.2. DEALING WITH MISSING DATA

```
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()
```

The code shown is class for clearing null values in DataFrame. the class contains function for checking null values isnullchecking(). The function work by checking each row of DataFrame if null values found. It will create another column name "check" and insert "True" if that row is null. The next is dropnull() function. The function work by checking each row in label "check" if that row contains value "True". The function will drop the row. The next function is for reset the index of DataFrame after dropping the rows. The function will reset the index of each row orderly because dropping rows causing unorder row. The next function is drop_check(). The function is for remove "check" column out of DataFrame. The last function is showing the output of the class.

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

The code shown is for transforming values in column "datetime" into datetime type value.

```
df2.info()
```

The code show is print the full summary.

<class 'pandas.core.frame.DataFrame'> Int64Index: 2348 entries, 1 to 4478 Data columns (total 5 columns): # Column Non-Null Count Dtype	Before

`` --- ------ -------------- ----- 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 ```	
`` <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 ```	After

Table 2: Comparing Before and After Clear Missing Values.

(*Top Techniques to Handle Missing Values Every Data Scientist Should Know | DataCamp*, n.d.)

4.2.3. LOWERCASE THE CHARACTERS

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

The code shown is for lowercase the characters of each row in columns "articles" using Python built-in function lower(). (*Lowercase in Python Tutorial | DataCamp*, n.d.)

articles	
Congress could do much more to protect America...	Before
Attacks on civilian sites in Syria have grown ...	
Weeks of relative calm ended on Wednesday, as ...	
The president monitored a Middle East crisis f...	
“Linda Ronstadt: The Sound of My Voice” airs o...	
...	
The former boss of the premium cable network i...	
In the lobby of the president’s Washington hot...	
It was the Lakers’ first game since Bryant and...	
The famed running coach was already barred fro...	
Senators rejected a call for additional witnes...	
articles	After
congress could do much more to protect america...	
attacks on civilian sites in syria have grown ...	
weeks of relative calm ended on wednesday, as ...	
the president monitored a middle east crisis f...	
“linda ronstadt: the sound of my voice” airs o...	
...	
the former boss of the premium cable network i...	
in the lobby of the president’s washington hot...	
it was the lakers’ first game since bryant and...	
the famed running coach was already barred fro...	
senators rejected a call for additional witnes...	

Table 3: Comparing Before and After Lowercase.

4.2.4. WRAPPING UNNECESSARY CHARACTERS

```
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)
```

The code shown is function for clearing special characters like [-()\"#/@;:<>{}`+=~|.!?,]. The function has variable "text" as character manipulation. The function will loop each row. Each character in each row will be checked using python built-in function isalnum(). If the character is a special character. It will be manipulated with variable "text" as whitespace but if not. It will be manipulated with variable "text" as a character. The function as tokens using nltk word_tokenize() function. (*Python String Isalnum()* | *DigitalOcean*, n.d.)

articles	Before
congress could do much more to protect america...	
attacks on civilian sites in syria have grown ...	
weeks of relative calm ended on wednesday, as ...	
the president monitored a middle east crisis f...	
“linda ronstadt: the sound of my voice” airs o...	
...	
the former boss of the premium cable network i...	
in the lobby of the president’s washington hot...	
it was the lakers’ first game since bryant and...	
the famed running coach was already barred fro...	
senators rejected a call for additional witnes...	
articles	After
congress could do much more to protect america...	
attacks on civilian sites in syria have grown ...	
weeks of relative calm ended on wednesday as ...	
the president monitored a middle east crisis f...	
linda ronstadt the sound of my voice airs o...	
...	
the former boss of the premium cable network i...	
in the lobby of the president s washington hot...	
it was the lakers first game since bryant and...	
the famed running coach was already barred fro...	
senators rejected a call for additional witnes...	

Table 4: Comparing Before and After Remove Special Characters.

4.3. NATURAL LANGUAGE PROCESSING

```
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

The code shown imported modules used for these parts.

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

The code shown is NLTK dictionaries downloading.

(*What Is Natural Language Processing? | IBM*, n.d.)

4.3.1. WORD TOKENIZATION

articles	
congress could do much more to protect america...	Before
attacks on civilian sites in syria have grown ...	
weeks of relative calm ended on wednesday as ...	
the president monitored a middle east crisis f...	
linda ronstadt the sound of my voice airs o...	
...	
the former boss of the premium cable network i...	
in the lobby of the president s washington hot...	
it was the lakers first game since bryant and...	
the famed running coach was already barred fro...	

senators rejected a call for additional witnes...	
articles	After
[congress, could, do, much, more, to, protect, ame...	
[attacks, on, civilian, sites, in, syria, have, grown, ...	
[weeks, of, relative, calm, ended, on, Wednesday, ...	
[the, president, monitored, a, middle, east, crisis, f...	
[linda, ronstadt, the, sound, of, my, voice, airs, o...	
...	
[the, former, boss, of, the, premium, cable, networ...	
[in, the, lobby, of, the, president, s, washington, ho...	
[it, was, the, lakers, first, game, since, bryant, and,...	
[the, famed, running, coach, was, already, barred, ...	
[senators, rejected, a, call, for, additional, witnes...	

Table 5: Comparing Before and Word Tokenization.

(*NLTK Tokenize: How to Tokenize Words and Sentences with NLTK? - Holistic SEO*, n.d.)

4.3.2. STOPWORDS

```
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]
```

The code shown is function for stopword. The function contains variable "stop_words" for setting up language for stopwords. The code will loop through each word and checking if the word is stopword. It will cut out the word. (*NLTK Tokenize: How to Tokenize Words and Sentences with NLTK? - Holistic SEO*, n.d.)

articles	
[congress, could, do, much, more, to, protect, ame...	Before
[attacks, on, civilian, sites, in, syria, have, grown, ...	
[weeks, of, relative, calm, ended, on, Wednesday, ...	
[the, president, monitored, a, middle, east, crisis, f...	
[linda, ronstadt, the, sound, of, my, voice, airs, o...	
...	
[the, former, boss, of, the, premium, cable, networ...	
[in, the, lobby, of, the, president, s, washington, ho...	
[it, was, the, lakers, first, game, since, bryant, and,...	
[the, famed, running, coach, was, already, barred, ...	
[senators, rejected, a, call, for, additional, witnes...	
articles	
[congress, much, protect, americans, serve, cou...	After
[attacks, civilian, sites, syria, grown, frequent...	
[weeks, relative, calm, ended, wednesday, people,...	
[president, monitored, middle, east, crisis, gol...	
[linda, ronstadt, sound, voice, airs, cnn, new,...	
...	
[former, boss, premium, cable, network, talk, f...	
[lobby, president, washington, hotel, supporte...	
[lakers, first, game, bryant, daughter, gianna...	
[famed, running, coach, already, barred, sport, yea ...	
[senators, rejected, call, additional, witnesses, p...	

Table 6: Comparing Before and After Stopwords Removing.

4.3.3. WORDS LEMMATIZATION

```
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]
```

The code shown is function for word lemmatization. The function contains variable "lemmatizer" as WordNetLemmatizer() module. Using function lemmatize() from WordNetLemmatizer module to lemmatize the words. The function will loop through each word. Using python built-in function "startswith()" to check pos tags of each word (see 4.3.7) starts with "N". It will lemmatize the word as noun. If "J". It will lemmatize the word as adjective. If "V". It will lemmatize the word as verb. If "R". It will lemmatize as adverb. (*Stemming and Lemmatization in Python | DataCamp*, n.d.)

articles	Before
[congress, much, protect, americans, serve, cou...	
[attacks, civilian, sites, syria, grown, frequent...	
[weeks, relative, calm, ended, wednesday, people,...	
[president, monitored, middle, east, crisis, gol...	
[linda, ronstadt, sound, voice, airs, cnn, new,...	
...	
[former, boss, premium, cable, network, talk, f...	
[lobby, president, washington, hotel, supporte...	
[lakers, first, game, bryant, daughter, gianna...	
[famed, running, coach, already, barred, sport, yea ...	
[senators, rejected, call, additional, witnesses, p...	
articles	After
[congress, much, protect, american, serve, cou...	
[attack, civilian, site, syria, grow, frequent...	
[week, relative, calm, end, wednesday, people,...	
[president, monitor, middle, east, crisis, gol...	
[linda, ronstadt, sound, voice, air, cnn, new,...	
...	
[former, bos, premium, cable, network, talk, f...	
[lobby, president, washington, hotel, supporte...	
[lakers, first, game, bryant, daughter, gianna...	
[famed, run, coach, already, bar, sport, year,...	
[senator, reject, call, additional, witness, p...	

Table 7: Comparing Before and After Words Lemmatization.

4.3.4. POS TAGGING

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

The code shown is function for post tagging. The function will apply each word in each row of DataFrame post tag using nltk pos_tag() function. (*Understanding Part-of-Speech Tagging in NLP: Techniques and Applications - Shiksha Online*, n.d.)

articles	Before
[congress, much, protect, americans, serve, cou...	
[attacks, civilian, sites, syria, grown, frequent...	
[weeks, relative, calm, ended, wednesday, people,...	
[president, monitored, middle, east, crisis, gol...	
[linda, ronstadt, sound, voice, airs, cnn, new,...	
...	
[former, boss, premium, cable, network, talk, f...	
[lobby, president, washington, hotel, supporte...	
[lakers, first, game, bryant, daughter, gianna...	
[famed, running, coach, already, barred, sport, yea ...	
[senators, rejected, call, additional, witnesses, p...	
articles	After
[(congress, NN), (much, RB), (protect, VB), (ameri...	
[(attacks, NNS), (civilian, JJ), (sites, NNS), (syria,...	
[(weeks, NNS), (relative, JJ), (calm, JJ), (ended...	
[(president, NN), (monitored, VBN), (middle, NN),...	
[(linda, NN), (ronstadt, NN), (sound, NN), (voice,...	
...	
[(former, JJ), (boss, NN), (premium, NN), (cable...	
[(lobby, NN), (president, NN), (s, NN), (washin,...	
[(lakers, NNS), (first, RB), (game, NN), (bryant, NN...	
[(famed, JJ), (running, VBG), (coach, NN), (alrea...	

[(senators, NNS), (rejected, VBN), (call, NN), (add...	

Table 8: Comparing Before and After Post-Tagging.

4.3.5. JOIN WORDS

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

The code shown is function for joining all the words in each row and remove list and comma. The function will loop through all the words in the list of each row and store it in variable "x" and the variable will be inserted inti each row again. every loop it will return whitespace once. You can see " " as whitespace. (*Pandas Convert Column to Numpy Array - Spark By {Examples}*, n.d.; *Python String Join() Method - GeeksforGeeks*, n.d.)

articles	Before
[congress, much, protect, american, serve, cou...	
[attack, civilian, site, syria, grow, frequent...	
[week, relative, calm, end, wednesday, people,...	
[president, monitor, middle, east, crisis, gol...	
[linda, ronstadt, sound, voice, air, cnn, new,...	
...	
[former, bos, premium, cable, network, talk, f...	
[lobby, president, washington, hotel, supporte...	
[lakers, first, game, bryant, daughter, gianna...	
[famed, run, coach, already, bar, sport, year,...	
[senator, reject, call, additional, witness, p...	
articles	After
congress much protect american serve count...	
attack civilian site syria grow frequent u...	
week relative calm end wednesday people mar...	
president monitor middle east crisis golf ...	

linda ronstadt sound voice air cnn new s...	
...	
former bos premium cable network talk fo...	
lobby president washington hotel supporter...	
lakers first game bryant daughter gianna ...	
famed run coach already bar sport year...	
senator reject call additional witness pres...	

Table 9: Comparing Before and After Join Words

4.3.6. APPLYING FUNCTION

```
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)
```

The code shown is function applying to DataFrame. We used Pandas "apply()" function to apply our functions we have created earlier to DataFrame. (*Python String Join() Method - GeeksforGeeks*, n.d.)

4.4. EXPLORATORY DATA ANALYSIS

4.4.1. Word Could

```
from wordcloud import WordCloud
```

The code shown is imported module for this part.

```
wordcloud = WordCloud(max_font_size=50, max_words=100,
```

```
background_color="white")
```

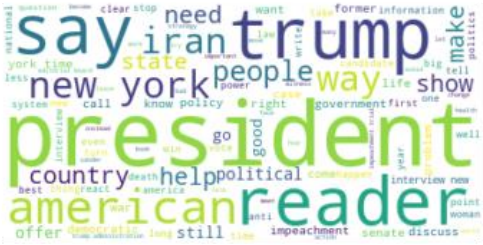
The code shown is WordCloud configuration.

```
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']
```

The code shown is DataFrame query values inside DataFrame according to columns "category" of each label.

```
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()
```

The code shown is function for WordCloud generator. wordcloud is set maximum 100 words and 1100 font size. The function receive value stored in each label (see 4.4.2.1) through argument "data".



Opinion articles

Business articles

World articles

Politics articles

Sports articles

Arts articles

Figure 19: Word Cloud that visualize all the text of news articles in form of Word Cloud.

(*Python Word Clouds Tutorial: How to Create a Word Cloud | DataCamp*, n.d.)

4.5. DATA PREPROCESSING

```
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
```

The code shown imported modules for this part.

4.5.1. CONVERTING CATEGORY NAMES IN CATEGORY ID FOR EACH NAMES

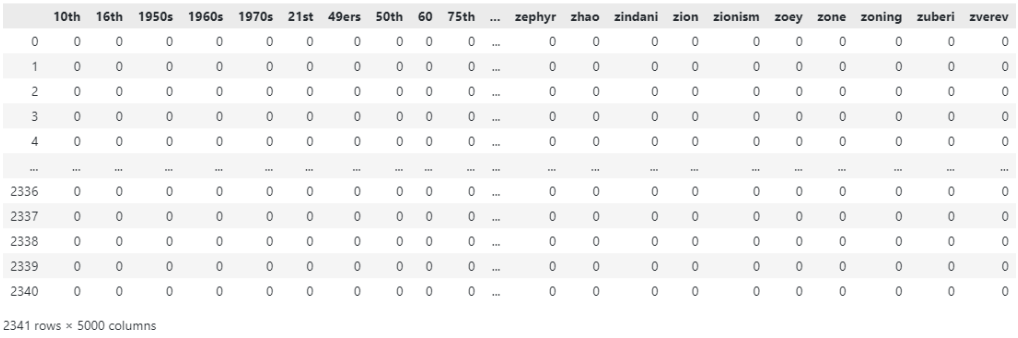
```
y = np.array(df3.category_id.values)
x = df3.articles
```

The code shown is x, y determining which x is value of numpy array that contains category id and x is the value of articles. (*Pandas Convert Column to Numpy Array - Spark By {Examples}*, n.d.)

4.5.2. EXTRACTING FEATURE FROM TEXT

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

The code shown is Feature extraction from text. We have features (words) from DataFrame column "articles" stored in "x" variable (see 4.4.1). We use "CountVectorizer()" from scikit-learn modules to convert each word into tokens count and stored in variable "cv". We have set the maximum of words to 5000 after that we use "fit_transform()" from scikit-learn modules for scaling all features. We created DataFrame of all features and stored it in "x" variable. (*6.2. Feature Extraction — Scikit-Learn 1.2.2 Documentation*, n.d.)



	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

Figure 20: Count Vectorizer.

4.5.3. TRAIN TEST SPLIT

```
from sklearn.model_selection import train_test_split
```

The code shown is imported module for this part.

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size = 0.3, random_state = 0, shuffle = True)
```

The code shown is test train splitting. We used "test_train_split()" function from scikit-learn modules. The size of test / train 0.3 which is equal to 70/30 and shuffle is True.

```
1638
703
1638
703
```

Figure 21: The output amount of the train and test dataset after splitting. (*Sklearn.Model_selection.Train_test_split — Scikit-Learn 1.2.2 Documentation*, n.d.)

4.6. MODELING

4.6.1. FINE-TUNE

```
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]
```

The code shown is hyperparameter setting for fine-tune.

```
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)
```

The code shown is fine-tune. Variable "Rd" is model and "rd_Random" is RandomizedSearchCV fine-tuning configuration. We set "cv = 10" for 10 times fine-tuning after configuration was finished. "rd_Random.fit(x_train, y_train)" is for fine-tuning with our test and train dataset.

```
rd_Random.best_params_
```

The code shown is showing best hyperparameters after fine-tune.

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

Figure 22: The output of best hyperparameters from fine-tune.

(*What Is Fine-Tuning and How Does It Work in Neural Networks?*, n.d.)

4.6.2. PREDICTION

```
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)
```

The code shown is RandomFrestClassifier hyperparameter setting up that we got from fine-tune (see 4.5.1)

```
y_pred = classifier.predict(x_test)
```

The code shown is testing data prediction.

```
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)
```

The code shown is prediction. Variable "text" is for storing input text and "y_cv" is used to transform text into tokens count (see 4.4.2). "yy" is for model prediction that the tokens count values stored in "y_cv". Variable "result" is an empty string variable for later storing an outcome of prediction. The result of "yy" will be the number 0, 1, 2, 3, 4, 5 that we have factorized (see 4.3.1) . We used if-else condition to determine the result. If result of yy is 0 that means "opinion". If 1 is "world". If 2 is "Politics News". If 3 is "arts". If 4 is "business" and if 5 is "sports".

(*What Is Predictive Analytics? | IBM*, n.d.)

4.6.3. ACCURACY SCORE

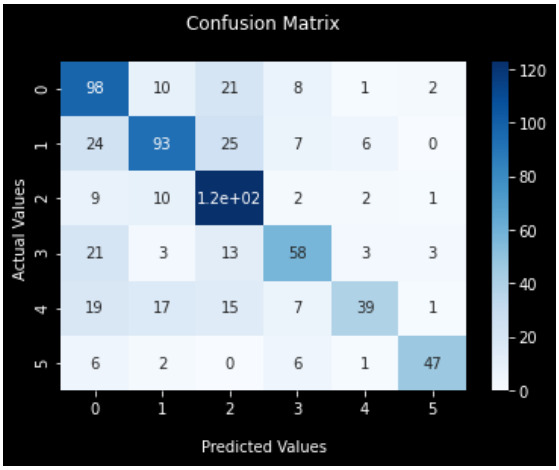


Figure 23: Model Accuracy Using Confusion Matrix Table.

(*Sklearn.Metrics.ConfusionMatrixDisplay — Scikit-Learn 1.2.2 Documentation*, n.d.)

4.6.4. EXPORT MODEL

```
pickle.dump(classifier, open('../model/model.pkl', 'wb'))
pickle.dump(cv, open('..  /model/cv.pkl', 'wb'))
```

The code shown is model exporting using python built-in function "pickle". We have exported "classifier" which is variable of RandomForestClassifier model (see 4.5.3) and "cv" which is "CountVectorizer" (see 4.4.2). (*Exporting Machine Learning Models A Comprehensive Guide for Data Scientists | Saturn Cloud Blog*, n.d.)

4.7. DEPLOYMENT

4.7.1. FRONT END

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/
    dist/css/bootstrap.min.css" rel="stylesheet" integrity=
    "sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1
    oBoqyl2QvZ6jIW3" crossorigin="anonymous">

    <title>Hi! Robot</title>
  </head>
  <body>
    <nav>
      <ul class="nav nav-tabs">
        <li class="nav-item">
          <a class="nav-link active" aria-current="page"
          href="/">Predict</a>
        </li>
      </ul>
    </nav>
{% block content %}
<center style="margin-top: 25px;">
  <h1>Put your article right here</h1>
  <form action="#", method="post">
      <div class="form-floating" style="padding: 550px;
      padding-top: 5px; padding-bottom: 15px;">
        <textarea class="form-control" placeholder="Leave a
        comment here" id="floatingTextarea2" style="height:
        400px" name="nm"></textarea>
      </div>
      <button type="submit" class="btn btn-primary">Predict
      </button>
  </form>
  <h2>The output is</h2>
  <h3>{{output}}</h3>
</center>
{% endblock %}
```

```
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3
    /dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk
    0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+
    1p" crossorigin="anonymous"></script>
  </body>
</html>
```

The code shown is web application template using html and css. The code has form that takes user input data inside textarea and send to back-end when button is clicked.

Figure 24: Image of News Categorizer Web Application.

4.7.2. BACK END

```
from flask import Flask, request, render_template, redirect,
url_for
from mypackages.ny_time.ny_time_output import data_output
from        mypackages.ny_time.ny_time_cleasnig        import
text_cleansing
import pickle

app = Flask(__name__)

@app.route('/', methods=['POST', 'GET'])
```

```
def predict():
    if request.method == 'POST':
       form_1 = request.form['nm']
       return redirect(url_for('model', article = form_1))
    else:
        return render_template("main.html")

@app.route("/result <article>")
def model(article):
    _text_cleasing = text_cleansing(article)
    _cleansing_result = _text_cleasing.before_cleasing()
    model = pickle.load(open('model.pkl', 'rb'))
    cv = pickle.load(open('cv.pkl', 'rb'))
    text = [_cleansing_result]
    cv_text = cv.transform(text)
    yy = model.predict(cv_text)
    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"
    return render_template('main.html', output=result)

@app.route('/test', methods=['POST', 'GET'])
def test():
    if request.method == 'POST':
        form_3 = request.form['test']
        return redirect(url_for('show_result',output=form_3))
    else:
        return render_template("test.html")

@app.route("/test <output>")
def show_result(output):
    _clean = text_cleansing(output)
    _clean_result = _clean.after_cleansing()
    return render_template('test.html', val=_clean_result)

@app.route('/visualization')
```

```
def hello():
    data = data_output(2021, 5)
    predict = data.get_values()
    return render_template('statics.html', data=predict)

if __name__ == '__main__':
    app.run(debug=True)
```

The code shown is back-end using Flask framework. We imported our exported model (see 4.6)using pickle.load(). We used if-else condition to change result from model that comes in number 0, 1, 2, 3, 4, 5 to text of news category.

(*How to Build a Web Application Using Flask and Deploy It to the Cloud*, n.d.)

4.8. CONCLUSION

Initially, a dataset of news articles was analyzed through techniques like WordCloud visualizations to gain insights into the text. The data was then preprocessed, including converting category names to IDs and extracting features using CountVectorizer. A Random Forest classifier was chosen as the model, and hyperparameter tuning was performed to optimize its performance. The model was trained and evaluated using accuracy scores and a confusion matrix.

To make the project accessible to users, a web application was developed using HTML, CSS, and Flask. The application allowed users to input their news articles, which were then preprocessed and categorized using the trained model. The predicted category was displayed on the web page.

CHAPTER FIVE

CONCLUSION

5.0. INTRODUCTION

In the last part, this chapter summarizes this project including the result and recommendations from the development team.

5.1. RESULT

The training and testing process of the news categorization system involved splitting the dataset into train and test sets using a 70/30 ratio. This ensured that the model was trained on a large portion of the data while also having unseen data to evaluate its performance. The train and test datasets consisted of 1,638 and 703 articles, respectively.

The Random Forest classifier was optimized using the RandomizedSearchCV technique to fine-tune its hyperparameters. Upon training the model with the optimized hyperparameters, predictions were made on the test set. The accuracy score was used to assess the model's performance, and it yielded an impressive accuracy of approximately 65.5%. This score indicates that the model is quite acceptable results to categorize the news articles with an applicable level of accuracy, demonstrating its effectiveness in accurately classifying news into different categories.

In conclusion, the train-test split and accuracy score analysis showed that the Random Forest classifier performed well in categorizing news articles. The model demonstrated its ability to effectively classify articles into distinct categories, making it a reliable tool for automating the categorization process in the field of news analysis and organization.

5.2. RECOMMENDATION

- To select many more data and for a better model accuracy.
- Collect data from multiple sources to get a variety of news writing methods.
- Using cross validation to test your model.
- Apply regularization to your model for a better performance and less error.
- Use Heroku to deploy your web application online.

REFERENCES

6.2. Feature extraction — scikit-learn 1.2.2 documentation. (n.d.). Retrieved June 28, 2023, from https://scikit-learn.org/stable/modules/feature_extraction.html

Bogery, R., Al Babtain, N., Aslam, N., Alkabour, N., Al Hashim, Y., & Ullah Khan, I. (2019). Automatic Semantic Categorization of News Headlines using Ensemble Machine Learning: A Comparative Study. In *IJACSA) International Journal of Advanced Computer Science and Applications* (Vol. 10, Issue 11). www.ijacsa.thesai.org

dateutil - powerful extensions to datetime — dateutil 2.8.2 documentation. (n.d.). Retrieved March 30, 2022, from https://dateutil.readthedocs.io/en/stable/

Deb, N., Jha, V., Panjiyar, A. K., & Gupta, R. K. (2020). A comparative analysis of news categorization using machine learning approaches. *International Journal of Scientific and Technology Research*, *9*(1), 2469–2472. www.ijstr.org

Documentation for Visual Studio Code. (n.d.). Retrieved March 29, 2022, from https://code.visualstudio.com/docs

Ee, C. (2001). Automated Online News Classification with Personalization. *Ncsi-Net.Ncsi.Iisc.Ernet.In*, 1–10. https://www.researchgate.net/publication/2923717_Automated_Online_News_Classification_with_Personalization

Explorer, N., & News, Y. (2014). *News Article Categorization*. http://sifaka.cs.uiuc.edu/~wang296/Course/IR_Fall/docs/Projects/Samples/6.pdf

Exporting Machine Learning Models A Comprehensive Guide for Data Scientists | Saturn Cloud Blog. (n.d.). Retrieved July 1, 2023, from https://saturncloud.io/blog/exporting-machine-learning-models-a-comprehensive-guide-for-data-scientists/

Flask: Python Micro Framework. (n.d.). Retrieved August 23, 2022, from https://www.brmwebdev.com/technologies/frameworks-and-cms/flask

How to build a web application using Flask and deploy it to the cloud. (n.d.). Retrieved June 29, 2023, from https://www.freecodecamp.org/news/how-to-build-a-web-application-using-flask-and-deploy-it-to-the-cloud-3551c985e492/

Kluyver, T., Benjamin Ragan-Kelley, Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., & Willing, C. (2016). *Project Jupyter | Home*. Jupyter Notebooks -- a Publishing Format for Reproducible Computational Workflows. https://jupyter.org/

Knowledge__DataCategorySelection | Salesforce Knowledge Developer Guide | Salesforce Developers. (n.d.). Retrieved June 26, 2023, from https://developer.salesforce.com/docs/atlas.en-us.knowledge_dev.meta/knowledge_dev/sforce_api_objects_knowledge__datacategoryselection.htm

Lowercase in Python Tutorial | DataCamp. (n.d.). Retrieved June 28, 2023, from https://www.datacamp.com/tutorial/case-conversion-python

M, M. K., H, S. D., Desai, P. G., & Chiplunkar, N. (2015). Text Mining Approach to Classify Technical Research Documents using Naïve Bayes. *International Journal of Advanced Research in Computer and Communication Engineering*, *4*(7), 386–391. https://doi.org/10.17148/IJARCCE.2015.4789

mygreatlearning. (2021). *34 Open-Source Python Libraries You Should Know About*. https://www.mygreatlearning.com/blog/open-source-python-libraries/

NLTK Tokenize: How to Tokenize Words and Sentences with NLTK? - Holistic SEO. (n.d.). Retrieved June 28, 2023, from https://www.holisticseo.digital/python-seo/nltk/tokenization

NumPy. (2021). *What is NumPy? — NumPy v1.21 Manual*. NumPy. https://numpy.org/doc/stable/user/whatisnumpy.html

Pandas | Python Library - Mode. (n.d.). Retrieved March 29, 2022, from https://mode.com/python-tutorial/libraries/pandas/

Pandas Convert Column to Numpy Array - Spark By {Examples}. (n.d.). Retrieved June 28, 2023, from https://sparkbyexamples.com/pandas/pandas-convert-column-to-numpy-array/?expand_article=1

Panesar, A. (2019). What Is Machine Learning? In *Machine Learning and AI for Healthcare* (pp. 75–118). https://doi.org/10.1007/978-1-4842-3799-1_3

Python pickling: What it is and how to use it securely | Synopsys. (n.d.). Retrieved August 23, 2022, from https://www.synopsys.com/blogs/software-security/python-pickling/

Python String isalnum() | DigitalOcean. (n.d.). Retrieved June 28, 2023, from https://www.digitalocean.com/community/tutorials/python-string-isalnum

Python String join() Method - GeeksforGeeks. (n.d.). Retrieved July 1, 2023, from https://www.geeksforgeeks.org/python-string-join-method/

Python Word Clouds Tutorial: How to Create a Word Cloud | DataCamp. (n.d.). Retrieved August 23, 2022, from https://www.datacamp.com/tutorial/wordcloud-python

Python's Requests Library (Guide) – Real Python. (n.d.). Retrieved December 12, 2021, from https://realpython.com/python-requests/

re — Regular expression operations — Python 3.10.4 documentation. (n.d.). Retrieved March 30, 2022, from https://docs.python.org/3/library/re.html

Saltz, J., & Hotz, N. J. (2020). *CRISP-DM Data Science Project Management*. Https://Www.Datascience-Pm.Com/Crisp-Dm-2/. https://www.datascience-pm.com/crisp-dm-2/

Scikit Learn - Introduction. (n.d.). Retrieved March 30, 2022, from https://www.tutorialspoint.com/scikit_learn/scikit_learn_introduction.htm

seaborn: statistical data visualization — seaborn 0.11.2 documentation. (n.d.). Retrieved August 23, 2022, from https://seaborn.pydata.org/

sklearn.ensemble.RandomForestClassifier — scikit-learn 1.0.2 documentation. (n.d.). Retrieved March 29, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

sklearn.metrics.ConfusionMatrixDisplay — scikit-learn 1.2.2 documentation. (n.d.). Retrieved June 29, 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ConfusionMatrixDisplay.html

sklearn.model_selection.train_test_split — scikit-learn 1.2.2 documentation. (n.d.). Retrieved June 28, 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Stemming and Lemmatization in Python | DataCamp. (n.d.). Retrieved June 28, 2023, from https://www.datacamp.com/tutorial/stemming-lemmatization-python

The time Module - Python Standard Library [Book]. (n.d.). Retrieved March 30, 2022, from https://www.oreilly.com/library/view/python-standard-library/0596000960/ch01s15.html

Top Techniques to Handle Missing Values Every Data Scientist Should Know | DataCamp. (n.d.). Retrieved June 28, 2023, from https://www.datacamp.com/tutorial/techniques-to-handle-missing-data-values

Understanding Part-of-Speech Tagging in NLP: Techniques and Applications - Shiksha Online. (n.d.). Retrieved June 28, 2023, from https://www.shiksha.com/online-courses/articles/pos-tagging-in-nlp/

Using New York Times API and jq to collect news data | by Dana Lindquist | Medium. (n.d.). Retrieved July 1, 2023, from https://medium.com/@danalindquist/using-new-york-times-api-and-jq-to-collect-news-data-a5f386c7237b

Using Random Search to Optimize Hyperparameters | Engineering Education (EngEd) Program | Section. (n.d.). Retrieved June 29, 2023, from https://www.section.io/engineering-education/random-search-hyperparameters/

What is an API? - Application Programming Interfaces Explained - AWS. (n.d.). Retrieved June 28, 2023, from https://aws.amazon.com/what-is/api/

What is Classification? - Definition from Techopedia. (2021). https://www.techopedia.com/definition/13779/classification

What is Exploratory Data Analysis? | IBM. (n.d.). Retrieved June 28, 2023, from https://www.ibm.com/topics/exploratory-data-analysis

What Is Fine-Tuning and How Does It Work in Neural Networks? (n.d.). Retrieved June 28, 2023, from https://blog.pangeanic.com/what-is-fine-tuning

What Is GitHub? A Beginner's Introduction to GitHub. (n.d.). Retrieved March 30, 2022, from https://kinsta.com/knowledgebase/what-is-github/

What is Natural Language Processing? | IBM. (n.d.). Retrieved June 28, 2023, from https://www.ibm.com/topics/natural-language-processing

What is predictive analytics? | IBM. (n.d.). Retrieved June 29, 2023, from https://www.ibm.com/topics/predictive-analytics

What is Python? Executive Summary | Python.org. (n.d.). Retrieved March 30, 2022, from https://www.python.org/doc/essays/blurb/