

A Comprehensive Beginner's Guide for Researchers to Access, Preprocess, Query and Evaluate Dataset using Python: A Case Study

Kavitha M¹ and Dr. Purohit Shrinivasacharya²

¹Department of Computer Science and Engineering, Siddaganga Institute of Technology, Tumkur, Karnataka, India,
Affiliated to VTU, Belagavi, Karnataka, India

Email: kavitham@sit.ac.in

²Department of Information Science and Engineering, Siddaganga Institute of Technology, Tumkur, Karnataka, India
Email: purohitsn@gmail.com

Abstract— The era of Data Science, Machine Learning and Deep Learning has showcased the significance of dataset usage in the field of research. Since, the selection of high-quality dataset is very much important to train, validate and test the models, which directly affects the performance of the model, researchers must be able to choose proper dataset for their respective applications. However, the major hurdle for novice researchers is the difficulty in knowing how to access, preprocess and evaluate the datasets effectively. This paper provides a comprehensive handout for the beginners which includes step-by-step methodology for dataset access, preprocessing, querying and evaluation. As a case study, ISOT fake news dataset with Python code snippet is discussed. This comprehensive guide functions as a useful resource for novice researchers seeking to build a strong foundation in data handling.

Keywords— Data Science, Machine Learning, Deep Learning, Dataset, Python.

I. INTRODUCTION

In this data intensive research world, the proficiency to handle datasets has become a basic skill for researchers across various domains. Irrespective of the domain, datasets serve as the backbone of experimental research, machine learning models, deep learning models, and fact-driven insights. However, novice researchers face several hurdles at the starting stages of their research which includes identifying suitable datasets, accessing them via valid sources, preprocessing them for evaluating their relevance for particular tasks etc. These hurdles are intensified by the lack of reliable sources that guide the early-stage researchers through the complete process of dataset handling, especially using user-friendly and robust framework such as Python.

Python has emerged as a prominent language for data science due to its simplicity, extensive standard library and community support. Python libraries such as Pandas, NumPy, Scikit-learn, TensorFlow, NLTK etc. provide robust features for data manipulation, efficient numerical computations, data mining, large-scale machine learning and deep learning model development and evaluation, natural language processing etc. In spite of these various tools, majority of the researchers find it difficult to fuse them into a smooth operational flow for dataset preparation and interpretation. To address this gap, our paper presents a comprehensive, step-by-step guide specifically designed for novice researchers who may be struggling to handle dataset.

To introduce the core concepts for this tutorial in the real-world context, we use the ISOT Fake News Dataset which is a well-known dataset for binary text classification tasks. This dataset contains labelled news articles classified as "True" or "fake," making it an ideal case study for demonstrating text-based data processing and evaluation.

II. OBJECTIVES

To assist the researchers in managing each phase of data preparation and analysis using Python, this paper aims to highlight clear and practical objective for dataset handling. The objective is to upgrade the researcher with practical knowledge involving code snippets to perform the following tasks:

- Gain insights into the datasets.
- Find and access datasets from public repositories
- Preprocess the dataset
- Perform exploratory data analysis
- Query the dataset
- Evaluate the dataset

III. OVERVIEW OF DATASETS

A. Dataset and its Types

A dataset is a collection of data, typically represented in tabular form, containing rows representing the records and columns representing the features. Datasets form the backbone of machine learning and deep learning models. Datasets are very much crucial for training, validating, and testing the models which has direct impact on the reliability of the results. Dataset interpretation usually results in discovery of gaps which can aid in new hypotheses, research directions, or the need for improved data collection techniques.

As a researcher, one must be aware of different types of datasets, since it is very much required for choosing appropriate tools, techniques and workflows in any data-oriented research. This section provides an overview of the dataset categories commonly encountered in research. This classification will help researchers understand how data is organized, processed, and used for various analytical tasks. Figure 1 depicts the simple classification of datasets.

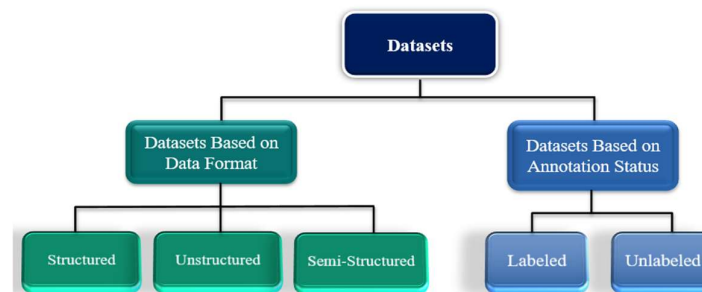


Figure 1: Dataset Types

- **Datasets Based on Data Format:** Elementary dataset classification based on the organization and representation of the data can be Structured, Unstructured and Semi-Structured datasets.
 - **Structured Datasets:** In these datasets, data is arranged tabular form with rows and columns. CSV, Excel, and other common formats are used where each row denotes a distinct record and each column usually indicates a feature of a specific data type, such as strings, integers, etc. Relational databases, like MySQL, facilitate the efficient searching, filtering, and updating of these datasets by making their handling and storage straightforward. Predictive modeling, statistical analysis, and data collection are the tasks that excel with organized datasets. Banking, healthcare, and business analytics are among the data-intensive industries that employ them extensively.
 - **Unstructured datasets:** Considering that unstructured datasets do not follow a predetermined data model or tabular layout, they are more difficult to process and analyse. They come in various formats, such as images, audio, video, and free-form text. Due to the variety and nonuniformity of unstructured datasets, more sophisticated preprocessing methods are necessary, such as computer vision for images and

natural language processing for text. It can be difficult to scale, integrate, and store these datasets because they typically come in a range of sizes and formats. For sentiment analysis, content classification, and image identification, they provide a multitude of context-specific data. They are extensively employed in innovative research domains such as media analytics.

- **Semi-Structured datasets:** A Semi-Structured dataset is one that combines aspects of both structured and unstructured data. Traditional relational data models organize data using strict tabular schemas, while semi-structured data uses flexible representations such as tags, nested items, and key-value pairs. JSON and XML are often used forms in applications such as system logs, API responses, and information storage. Because of its irregular nature, semi-structured data may occasionally require transformation and parsing before being examined using traditional methods. Because these datasets are self-descriptive by nature, they are able to capture hierarchical and nested relationships. Among the domains where they are commonly observed include web development, Internet of Things systems, and NoSQL databases.
- **Datasets Based on Annotation Status:** Labeled and unlabeled datasets which are based on annotation status can be distinguished by whether or not target values have been assigned to the data instances.
 - **Labeled datasets:** These datasets are the basic component that is mostly used in supervised learning, in which each instance of input data is associated with corresponding target outputs or annotations, which are also considered as labels. With the help of these ground truth labels, the models are able to identify trends and forecast outcomes that are already known. Accuracy and label quality directly affect the trained model's reliability and performance. It can take a lot of time and resources to generate labeled datasets because they often necessitate automated labeling methods, domain knowledge, or manual annotation. Many machine learning tasks, including classification, regression, object detection, and others, need labeled datasets. These datasets are essential for the deployment and performance analysis of predictive solutions since they serve as the foundation for model validation, benchmarking, and performance comparison.
 - **Unlabeled Datasets:** These datasets are frequently utilized for unsupervised learning tasks like clustering, anomaly detection, dimensionality reduction, etc., and lack target output variables or annotations. Techniques that use a small amount of labeled data, including representation learning, clustering algorithms, or semi-supervised learning, are required to handle unlabeled data. In real-world situations where manual labeling is challenging, time-consuming, or expensive, these datasets are massive. Finding hidden patterns, extracting pertinent features, and completing tasks where labeled data is scarce or non-existent, all depend on the efficient processing of unlabeled datasets.

B. Key Dataset Attributes

This section presents the essential dataset characteristics which helps the researchers to comprehend its structure, contents and relevance to a specific task such as ML model training, data analysis etc. Understanding the key dataset attributes also assist the researchers in evaluating the dataset quality and usability, choosing suitable preprocessing methods, building reliable ML and DL models. Figure 2 highlights the key dataset attributes.

- **Features:** They are also referred to as columns in tabular data. They are the independent input variables. Each feature represents the quantifiable property of the data instance. For example, the variables - title, text, date etc., are features in a news article dataset.

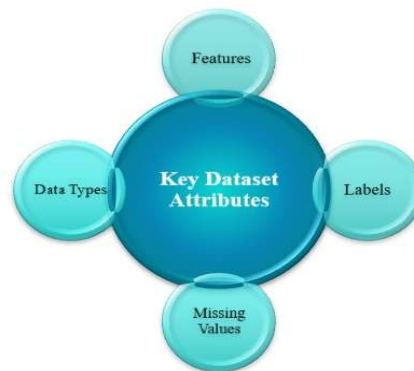


Figure 2: Key Dataset Attributes.

- **Labels:** They are also referred to as target variables. They are the dependent output variables or the values that the model attempts to predict. For example, the binary labels “fake” and “real” are present in a fake news dataset.
- **Missing Values:** This refers to the absence of data entries in one or more columns or rows. It is very much essential to handle missing values, since too many missing values in a dataset can bias or degrade the performance of the model. Common strategies to handle missing values include row deletion, estimating missing entries, or flagging absent values.
- **Data Types:** This specifies the form of data represented in each column of the dataset. Knowledge of data types of each feature is very much essential for selecting appropriate preprocessing and analysis techniques. Following are the commonly used data types:
 - String/Object is used for text data
 - Integer/Float is used for Numbers
 - Boolean is used for True/False values
 - Date Time for Date or Timestamps

IV. METHODOLOGY

This section outlines a practical approach for comprehensive data handling process in which, data acquisition, preprocessing, exploratory data analysis, querying, and evaluation stages are discussed using appropriate Python code snippets which are executed via Visual Studio code. Figure 3 depicts the sequential workflow for the data handling process.

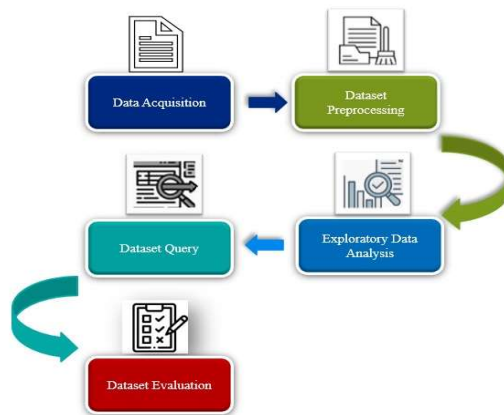


Figure 3: Sequential Workflow for Dataset Preparation and Evaluation in Data-Driven Research.

A. Data Acquisition

The first step in any data-oriented research is to access an appropriate dataset for a particular task. Novice researchers need to be aware of reliable sources to acquire open-access datasets. This section presents various public repositories from where datasets can be obtained and also demonstrates how to download the ISOT Fake news Dataset, which serves as the basis for the subsequent analysis in this paper.

- **Public Sources for Datasets:** This section gives a brief overview of the most commonly used public platforms that provide both structured and unstructured datasets across various domains. Most of the listed resources offer free accessibility with many also facilitating community collaboration, especially Kaggle, GitHub, and Papers with Code, making them indispensable for scholarly, exploratory, and practical investigations. Others such as UCI, AWS are reliable sources but with less focus on direct collaboration.
 - **Kaggle:** For data science competitions and community-driven datasets, Kaggle is one of the most well-liked platforms. It contains a large range of datasets, including unstructured text, images, and structured tables. In addition to offering integrated tools for collaboration, model training, and data exploration, Kaggle is most suitable for both novice as well as expertized researchers. The datasets can be found at the official website, <https://www.kaggle.com/datasets>.
 - **AWS Open Data Registry:** This is a registry of datasets that are made available to the public and are maintained within the cloud by Amazon Web Services (AWS). It contains sizable datasets utilized in

domains like satellite imaging, machine learning, genomics, and geospatial analysis. These datasets are directly accessible and processable by researchers on the AWS platform. The AWS Open Data Registry is available at <https://registry.opendata.aws/>.

- GitHub: On this well-known code-sharing website, researchers and developers regularly share datasets along with their open-source projects. Although GitHub is not primarily a dataset repository, it does provide domain-specific datasets as well as preprocessing and analysis code. One can access GitHub via the URL, <https://github.com/>.
 - UCI Machine Learning Repository: Carefully chosen datasets for machine learning applications are available from the reputable University of California, Irvine Machine Learning Repository. Its extensive documentation and metadata make it perfect for scholarly research and benchmarking models. UCI Machine Learning Repository website URL is <https://archive.ics.uci.edu/ml/index.php>.
 - Google Dataset Search: One specialized search engine that helps researchers find datasets available online is Google Dataset Search. It is a powerful tool for locating domain-specific datasets since it gathers metadata from a wide variety of public sources, including data publishers, governmental agencies, and academic institutions. One need to go to <https://datasetsearch.research.google.com/> to access Google Dataset Search.
 - Papers with Code: Papers with Code is an open-source platform that links machine learning research articles to the corresponding datasets and code implementations. It provides a single place to locate publicly available datasets used in benchmarking results, published research, and state-of-the-art models. The platform facilitates reproducibility in AI research and allows users to explore datasets by task, benchmark, and domain. The datasets are available at <https://paperswithcode.com/datasets>.
- Downloading the ISOT Fake News Dataset: To obtain the ISOT Fake News Dataset, visit <https://www.kaggle.com/datasets/csmalarkodi/isot-fake-news-dataset> and download the dataset. True.csv and Fake.csv are the two files that make up the dataset.

B. Dataset Preprocessing

Preprocessing is a crucial phase that prepares the data for querying and modeling. Raw datasets often contain inconsistencies, missing values, and noise, which must be addressed before any analysis can be performed. In this dataset preprocessing section, we will label the data with Fake = 0 and True = 1, combine both datasets, perform text cleaning, convert dates into proper datetime format, and remove duplicates and nulls.

- Import required libraries for ISOT Fake News Dataset Preprocessing: The tools required to read, clean, manipulate, and get the ISOT dataset ready for machine learning tasks are set up by the code presented in figure 4. Pandas library is necessary to handle data frames, filtering, grouping, and reading CSV files. Numpy library is helpful to efficiently handle missing or special values, arrays, and numerical operations. For text cleaning, such as eliminating punctuation, numbers, or undesired patterns in strings, the Regular Expression operations library is utilized. Shuffle function is imported from the scikit-learn library which is used to rearrange the dataset rows to remove order bias.

```
fake = pd.read_csv("Fake.csv")
true = pd.read_csv("True.csv")
```

Figure 4: Python Code to Import Required Libraries.

```
import pandas as pd
import numpy as np
import re
from sklearn.utils import shuffle
```

Figure 5: Python Code to Load ISOT Fake News Dataset.

- Load Datasets: Line1 code presented in figure 5 loads the Fake.csv file which contains the fake news articles into Pandas data frame named fake. Line2 code loads the True.csv file containing real news articles into a data frame named true.
- Add labels, combine and shuffle: The code presented in figure 6 helps to prepare the dataset for further steps of preprocessing, training and evaluation. Line1 and line2 codes are used to assign label 0 and 1 to all fake and real news articles respectively. With this, a supervised learning environment is created for the model to get trained in distinguishing fake and real news. Using line3 code, the two datasets are merged into one data frame for unified processing. Line4 code reorders the dataset rows to remove the order bias.

```
fake["label"] = 0
true["label"] = 1
data = pd.concat([fake, true])
data = shuffle(data).reset_index(drop=True)
```

Figure 6: Python Code to Add Labels, Combine and Shuffle.

- **Clean and Preprocess the Text:** The code snippet presented in figure 7 cleans and normalizes the text for further analysis and modeling. Line1 code converts date column to datetime format and replaces invalid date with NaT (Not a Time). Empty strings are filled for missing or null values in text column via Line2 code. Next 5 lines of code defines a text cleaning function which replaces newlines with spaces, eliminates special characters and converts entire text to lowercase. Line8 code applies clean_text function to each article in the dataset and creates a new column named clean_text.

```
data["date"] = pd.to_datetime(data["date"], errors='coerce')
data["text"] = data["text"].fillna("")
def clean_text(text):
    text = re.sub(r'\n', ' ', text)
    text = re.sub(r'^a-zA-Z ', '', text)
    text = text.lower()
    return text
data["clean_text"] = data["text"].apply(clean_text)
```

Figure 7: Python Code to Clean and Preprocess the Text.

C. Exploratory Data Analysis (EDA)

The purpose of Exploratory Data Analysis (EDA) is to statistically and visually summarize the dataset's key features. It helps with recognizing patterns, spotting irregularities, and comprehending how variables are distributed. This section reveals class distributions and textual patterns in real versus fake news articles through EDA using visualization libraries such as WordCloud, Seaborn, and Matplotlib.

- **Class Distribution Visualization:** To display the dataset's distribution of true and fake news articles, the code specified in figure 8 generates a bar plot.

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.countplot(x='label', data=data)
plt.title("Class Distribution")
plt.show()
```

Figure 8: Python Code for Class Distribution Visualization.

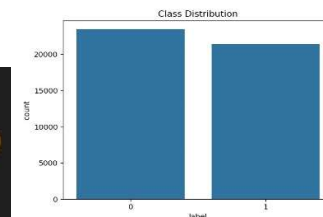


Figure 9: Bar Chart for Class Distribution.

Figure 9 shows the output bar chart generated after execution of the python code specified in figure 8 with two bars, label 0 and label 1 for fake and true news articles respectively. This information helps to check whether the dataset is balanced or unbalanced.

- **Word Count Distribution:** By classifying news articles by labels (fake or real), the code presented in figure 10 examines and displays the distribution of word counts. Line1 code determines the word count of each article's cleaned text and records it in a new word_count column. Line2 code generates a histogram that displays the distribution of word counts for both real (label=1) and fake (label=0) news. Line3 and line4 codes are used to add title and display the plot respectively.

```
data["word_count"] = data["clean_text"].apply(lambda x: len(x.split()))
sns.histplot(data=data, x="word_count", hue="label", bins=100, element="step", stat="density", common_norm=False)
plt.title("Word Count Distribution by Label")
plt.show()
```

Figure 10: Python Code for Word Count Distribution.

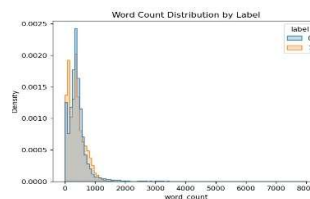


Figure 11: Histogram Displaying Word Count Distribution for Fake and Real News

Figure 11 shows the histogram generated for the code specified in figure 10. This information helps to determine whether the lengths of fake and real news are different, which might be a helpful classification feature.

- **Word Cloud Visualization:** The code presented in figure 12 is used for creating word clouds to show the most frequent words in both fake and real news articles. Line1 code imports wordcloud class to create wordcloud images. Line2 and line3 codes joins all words from the clean_text column associated with label 0 and label 1 into two long strings and stores them in fake_words and real_words respectively. Last two lines of code are used to create and show two separate word clouds, one for real news and the other for fake news.

```
from wordcloud import WordCloud
real_words = ' '.join(data[data['label']==1]['clean_text'])
fake_words = ' '.join(data[data['label']==0]['clean_text'])
wordcloud(width=800, height=400).generate(real_words).to_image().show()
wordcloud(width=800, height=400).generate(fake_words).to_image().show()
```

Figure 12: Python Code for Word Cloud Visualization.

Figure 13 and figure 14 show the word clouds for real and fake news respectively. Each word's size in the cloud indicates how frequently it appears in the text. This can provide information about linguistic patterns and possible classification features.



Figure 13: Word Cloud for Real News.

Figure 14: Word Cloud for Fake News.

D. Querying the Dataset

After preprocessing, researchers must query the dataset which helps to verify the integrity of data and provides early patterns useful for hypothesis formation. This section introduces basic querying techniques using Pandas which includes filtering to uncover structural patterns in the ISOT dataset.

- Basic Queries Using Pandas: Line1 query in figure 15 displays first five rows of the dataset to preview the structure and content. Line2 query counts the number of articles labeled as fake and real to understand the class distribution. Query specified in line3 filters the dataset based on real articles (label == 1) with subject containing the word “world” and line4 query displays the first five matching rows, showing only the title and subject columns to improve the readability.

```
print(data.head())
print(data['label'].value_counts())
real_politics = data[(data['label'] == 1) & (data['subject'].str.contains("world", case=False))]
print(real_politics[['title', 'subject']].head())
```

Figure 15: Python Code for Basic Queries.

The output of the queries specified in figure 15 code is presented in figure 16.

```

0      title                                     text ... label      clean_text
0  U.S. SEC chair grilled by Senate panel over cy...  WASHINGTON (Reuters) - The chairman of the U.S. sec...  1  washington reuters  the chairman of the us sec...
1  Anti-Trump Republican senator says efforts to ...  CLEVELAND (Reuters) - Efforts by some delegate...    1  cleveland reuters  efforts by some delegates t...
2  BREAKING VIDEO of Hillary supporter And hllack...  Hillary has been stirring up the hateful movem...    0  hillary has been stirring up the hateful movem...
3  THE LIST OF OBAMA'S HISTORIC FIRSTS AKA HOW CH...  Wow! What a list of accomplishments! The probl...    0  wow what a list of accomplishments the problem...
4  MIKE PENCE BOOED by Bitter Liberals AT Broadwa...  Here s Pence getting booed as he gets to his s...    0  here s pence getting booed as he gets to his s...

[5 rows x 6 columns]
label
0      23481
1      21417
Name: count, dtype: int64

0      title      subject
7  Bus bomb kills eight in Syria's Homs city; sta...  worldnews
8  Voter confidence in UK handling of Brexit ebbs...  worldnews
14  Turkey has not seen data on fugitive Raqqa fig...  worldnews
15  South Korea conducts anti-terror drills ahead ...  worldnews
29  Thousands march against coal ahead of climate ...  worldnews

```

Figure 16: Output of the Python Code Specified in Figure 15.

E. Dataset Evaluation

The dataset must be transformed and evaluated for analytical applicability prior to the application of machine learning models. This section presents vectorization technique to transform the text into numerical features and then performs train-test split. Finally, a baseline classification model is trained and assessed to show how well the feature extraction and preprocessing processes worked.

- **Text Vectorization:** The code presented in figure 17 uses TF-IDF vectorization technique to transform the cleaned text data into numerical feature.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(stop_words='english', max_df=0.7)
x = tfidf.fit_transform(data['clean_text'])
y = data['label']
```

Figure 17: Python Code for Vectorization.

- Train-Test Split: The code snippet in figure 18 splits the dataset into training and testing subsets for model training and evaluation. 80% of the data is used for training and 20% of the data is used for testing.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 18: Python Code for Train-Test Split.

- Model Training & Evaluation: The code presented in figure 19 is used to assess how well the logistic regression model can differentiate fake and real news articles based on the TF-IDF vectorized text. Line1 and line2 codes in figure 19 imports the Logistic Regression classifier and the accuracy_score metric from scikit-learn. Line3 code initializes the logistic regression model and line4 code trains the model using the x_train and y_train training features and labels respectively. Line5 code predicts the labels for the test set, x_test using the trained model. Line 6 code calculates and prints the accuracy i.e., the percentage of correctly predicted labels.

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Accuracy:", round(accuracy_score(y_test, y_pred),6))
```

Accuracy: 0.987862

Figure 19: Python Code for Model Training and Evaluation.

Figure 20: Output of the Python Code Specified in Figure 19

IV. CONCLUSION

Datasets serve as backbone of machine learning and statistical models. The choice of high-quality and effectively labeled data to train, validate, and test models often determines the scope, focus, and reliability of the research outcomes. Risk of false interpretations may arise from inadequate or biased datasets. However, novice researchers may face challenges in exploring about dataset and its' complete process of acquiring, preprocessing, querying, and evaluating it. By adhering to our handout, researchers will obtain a comprehensive overview of each of these processes. They will not only acquire theoretical insights but also gain practical skills required for data-oriented research. As each process in the paper is discussed in a very simple and precise way, it also benefits the researchers who lack programming skills. This paper serves as an excellent resource to foster deeper understanding of best practices in data handling, finally aiding the researchers in building more reliable and robust models for their respective tasks.

REFERENCES

- [1] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- [2] Python Software Foundation. (n.d.). Python 3.12.0rc3. <https://www.python.org>
- [3] The pandas development team. (2020, February). pandas-dev/pandas: Pandas 2.2.3 [Software]. Zenodo. <https://doi.org/10.5281/zenodo.3509134>
- [4] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- [5] J. D. Hunter, "Matplotlib: A 2D Graphics Environment", *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90-95, 2007. <https://doi.org/10.1109/MCSE.2007.55>
- [6] Waskom, M. L. (2021). Seaborn: Statistical data visualization (Version 0.13.2) [Computer software]. Zenodo. <https://doi.org/10.5281/zenodo.592845>
- [7] ISOT Fake News Dataset - <https://www.kaggle.com/datasets/csmalarkodi/isot-fake-news-dataset>