MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

Project Report

on

"Fake Review's Analyzer"

Submitted By:

Aryan Singh (0901AM211015)

Neetesh Jatav (0901AM211034)

Faculty Mentor:

Dr. Sunil Kumar Shukla (Assistant Professor), Center for AI

CENTER FOR ARTIFICIAL INTELLIGENCE

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

JULY-DEC. 2023

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

CERTIFICATE

This is certified that **Aryan Singh (0901AM211015)** & **Neetesh Jatav** (0901AM211034)

has submitted the project report titled **Fake Review's Analyzer** under the mentorship of **Dr. Sunil Kumar Shukla**, in partial fulfillment of the requirement for the award of degree of Bachelor of Technology in **Artificial Intelligence & Machine Learning** from Madhav Institute of Technology and Science, Gwalior.

23/11/2023

Dr. Sunil Kumar Shukla
Faculty Mentor
Assistant Professor

23/11/23

Dr. R. R. Singh
Coordinator
Center for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfillment of requirement for the award of the degree of Bachelor of Technology in **AIML** at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. R. R. Singh , Coordinator, Center for AI**

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Aryan

Aryan Singh
0901AM211015
3rd Year,
Centre for Artificial Intelligence

Neetesh

Neetesh Jatav
0901AM211034
3rd Year,
Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Centre for Artificial Intelligence,** for allowing me to explore this project. I humbly thank **Dr. R. R. Singh**, Coordinator, Centre for Artificial Intelligence, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Dr. Sunil Kumar Shukla (Assistant Professor) Center for Artificial Intelligence,** for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Aryan Singh
0901AM211015
3rd Year,
Centre for Artificial Intelligence

Neetesh

Neetesh Jatav
0901AM211034
3rd Year,
Centre for Artificial Intelligence

ABSTRACT

In this project, our focus lies in developing an advanced Fake Reviews Analyzer employing diverse machine learning models, including AdaBoost, Random Forest, Decision Tree Classifier, and KNN Classifier. As the realm of machine learning continues to progress, the availability of extensive datasets and computational resources becomes pivotal for constructing robust models capable of discerning the authenticity of reviews. Our Python-based initiative harnesses these machine learning methodologies to ascertain the veracity of reviews through comprehensive analysis of textual content.

The Fake Reviews Analyzer is a sophisticated process amalgamating natural language processing (NLP) concepts with a spectrum of machine learning models to ascertain the genuineness of reviews and categorize them accordingly. In this detailed survey paper, we delve into fundamental concepts of fake review detection and expound upon prevalent approaches employed in the domain. The implementation involves the utilization of machine learning models such as AdaBoost, Random Forest, Decision Tree Classifier, and KNN Classifier, leveraging renowned libraries like scikit-learn.

KEYWORDS: *Classify Reviews, Machine learning techniques, concepts of Ensemble Models*

सार:

इस परियोजना में, हमारा ध्यान एक उन्नत नकली समीक्षा विश्लेषक विकसित करने पर केंद्रित है, जिसमें एडाबूस्ट, रैंडम फॉरेस्ट, डिसीजन ट्री क्लासिफायर और केएनएन क्लासिफायर सहित विविध मशीन लर्निंग मॉडल का उपयोग किया जाता है। जैसे-जैसे मशीन लर्निंग का क्षेत्र प्रगति कर रहा है, समीक्षाओं की प्रामाणिकता को समझने में सक्षम मजबूत मॉडल के निर्माण के लिए व्यापक डेटासेट और कम्प्यूटेशनल संसाधनों की उपलब्धता महत्वपूर्ण हो जाती है। हमारी पायथन-आधारित पहल पाठ्य सामग्री के व्यापक विश्लेषण के माध्यम से समीक्षाओं की सत्यता का पता लगाने के लिए इन मशीन लर्निंग पद्धतियों का उपयोग करती है।

नकली समीक्षा विश्लेषक एक परिष्कृत प्रक्रिया है जो समीक्षाओं की वास्तविकता का पता लगाने और उन्हें तदनुसार वर्गीकृत करने के लिए मशीन लर्निंग मॉडल के एक स्पेक्ट्रम के साथ प्राकृतिक भाषा प्रसंस्करण (एनएलपी) अवधारणाओं को जोड़ती है। इस विस्तृत सर्वेक्षण पत्र में, हम नकली समीक्षा का पता लगाने की बुनियादी अवधारणाओं पर प्रकाश डालते हैं और इस क्षेत्र में नियोजित प्रचलित दृष्टिकोणों की व्याख्या करते हैं। कार्यान्वयन में एडाबूस्ट, रैंडम फॉरेस्ट, डिसीजन ट्री क्लासिफायर और केएनएन क्लासिफायर जैसे मशीन लर्निंग मॉडल का उपयोग शामिल है, जो स्किकिट-लर्न जैसे प्रसिद्ध पुस्तकालयों का लाभ उठाता है।

कीवर्ड: *वर्गीकृत समीक्षाएँ, मशीन सीखने की तकनीकें, एन्सेम्बल मॉडल की अवधारणाएँ*

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENT

CHAPTER 1 : INTRODUCTION

In the ever-expanding landscape of data-driven decision-making, the generation of computer-generated reviews plays a pivotal role in providing insights and informing choices. However, users often grapple with the authenticity and contextual richness of such automated reviews. The primary challenge arises from the inherent difficulty in capturing the nuanced aspects of human expression and sentiment through algorithms. While advancements in natural language processing have enabled machines to comprehend and replicate language patterns to a certain extent, the intricacies of human communication, cultural nuances, and contextual understanding still elude perfect emulation.

The struggle lies in the inherent limitations of pre-defined templates and rigid linguistic structures employed by these algorithms. As a result, computer-generated reviews may lack the spontaneity, diversity, and subjective interpretation that human-generated reviews inherently possess. Understanding and addressing these challenges are critical for enhancing the credibility and relevance of machine-generated reviews in diverse contexts. In this project, we delve into the complexities surrounding computer-generated reviews, aiming to bridge the gap between automated insights and the nuanced intricacies of human expression.

1.1Motivation

The motivation behind this project stems from the recognized disparities between computer-generated reviews and the nuanced richness found in human-generated counterparts. While automated systems offer efficiency, they often fall short in capturing the depth and subtleties inherent in authentic human expression. The motivation is to overcome the limitations posed by rigid algorithms and pre-defined structures, aiming to enhance the contextual relevance and authenticity of machine-generated reviews. By addressing these challenges, this project aspires to contribute to the advancement of natural language processing and offer more reliable, diverse, and contextually nuanced computer-generated reviews, aligning them more closely with the intricacies of human communication and provides a better understanding. .

1.2Text Classification

Process :- Our approach unfolds through a systematic process encompassing data exploration, preprocessing, and model development. Beginning with a comprehensive exploratory data analysis, we scrutinize the dataset's structure, identify key features, and unveil insights into its distribution. Subsequently, we execute meticulous preprocessing steps, such as text normalization, removal of irrelevant characters, and the application of advanced techniques like Word2Vec for semantic understanding. The dataset undergoes thorough cleansing, including the elimination of duplicate entries and the mitigation of imbalances. The culmination of this preprocessing lays the groundwork for insightful visualizations, including word clouds, offering a qualitative grasp of the dataset's composition.

1.3Techniques

1. Word Embeddings with Word2Vec: Leveraging the Word2Vec model, we transform text data into dense vector representations, capturing semantic relationships between words. This technique enhances our understanding of contextual similarities within the dataset, fostering more nuanced analyses.
2. Ensemble Modeling with SMOTE: To address class imbalance, we employ the Synthetic Minority Over-sampling Technique (SMOTE) during model building. SMOTE enhances the predictive capability of machine learning models by oversampling minority classes, promoting a balanced representation of all categories.
3. Machine Learning Models: Our model building phase involves the implementation of diverse machine learning algorithms, including AdaBoost, Random Forest, Decision Tree, and K-Nearest Neighbors. Each model contributes to the prediction of review ratings, providing a holistic evaluation of their effectiveness.
4. Stacking Ensemble Model: The stacking ensemble model combines the strengths of Random Forest and K-Nearest Neighbors, further fine-tuned using GridSearchCV with a meta-classifier, Logistic Regression. This amalgamation aims to maximize predictive accuracy and robustness.

Throughout this process, our objective is to distill meaningful insights from the dataset, enhance the quality of machine-generated reviews, and contribute to the advancement of natural language processing and predictive modelingtechniques.

CHAPTER 2: LITERATURE REVIEW

The literature review for this project explores existing research on natural language processing, sentiment analysis, and review generation. Prior studies delve into the challenges of mimicking human expression, emphasizing the limitations of template-based approaches. Noteworthy advancements include the application of Word2Vec for semantic understanding and ensemble modeling techniques like AdaBoost and Random Forest for classification tasks. Addressing imbalanced datasets through techniques like SMOTE has garnered attention. The literature underscores the ongoing pursuit to refine machine-generated reviews, aligning them more closely with the intricacies of human communication for improved authenticity and relevance.

2.1Text Classification Methods

Text classification methods play a pivotal role in the realm of natural language processing, serving as a cornerstone for various applications such as sentiment analysis, spam detection, and topic categorization. One prevalent approach involves the utilization of machine learning algorithms, ranging from traditional models like Naive Bayes and Support Vector Machines to more sophisticated ensemble techniques such as Random Forest and Gradient Boosting. These algorithms operate on the principle of learning patterns and relationships within textual data to make predictions or assign labels. Deep learning methods, particularly recurrent neural networks (RNNs) and convolutional neural networks (CNNs), have gained prominence for their ability to capture sequential dependencies and hierarchical features in text. Embedding techniques, such as Word2Vec and GloVe, enable the transformation of words into vector representations, facilitating a semantic understanding that enhances the performance of text classifiers. Moreover, advancements in transformer models, exemplified by BERT (Bidirectional Encoder Representations from Transformers), have revolutionized text classification by considering contextual information bidirectionally. The choice of text classification method often depends on the specific task requirements, dataset characteristics, and the balance between interpretability and predictive performance, reflecting a dynamic and evolving landscape in the pursuit of more accurate and versatile textanalysis.

2.2 Strategic Selection: Choosing Traditional Models over Deep Learning for Text Classification

In opting for alternative methods over deep learning techniques, several considerations influenced our decision. While deep learning models, particularly recurrent neural networks (RNNs) and convolutional neural networks (CNNs), showcase remarkable capabilities in capturing intricate patterns and contextual information in text, their implementation often demands substantial computational resources and extensive labeled data. In our context, the dataset's scale and the associated computational requirements posed challenges, making traditional machine learning models a pragmatic choice. Moreover, the interpretability and transparency offered by models like Naive Bayes, Support Vector Machines, and ensemble methods such as Random Forest and Gradient Boosting are pivotal factors in scenarios where the explainability of predictions is crucial. These models not only perform admirably but also provide insights into the features influencing predictions, aiding in comprehensibility. Additionally, the prevalence of imbalanced datasets in our scenario prompted the use of Synthetic Minority Over-sampling Technique (SMOTE) for handling class imbalances, a technique more readily applicable to traditional models. By prioritizing efficiency, interpretability, and resource considerations, our approach aligns with the specific constraints and objectives of the project, demonstrating the nuanced decision-making inherent in selecting appropriate text classification method for better deep learning concepts.

CHAPTER 3: PRELIMINARY DESIGN

3.1Introduction

In this chapter, we provide a preliminary design overview of the proposed solution for the text classification project. The preliminary design phase focuses on establishing the foundational structure and architecture of the system. Key considerations include the choice of algorithms, system components, and the overall workflow.

3.2System Architecture

The system architecture is a modular framework designed for seamless text classification. It comprises three core components: data preprocessing, feature extraction, and model training. This modular approach facilitates scalability, allowing effortless adaptation to evolving project needs. Data preprocessing involves thorough cleaning, punctuation handling, and normalization, ensuring refined input. Feature extraction employs Word2Vec, translating textual information into numerical vectors, capturing semantic relationships crucial for effective classification. The chosen models, including AdaBoost, Random Forest, and Decision Trees, are integrated into the architecture, aligning with project objectives. This strategic design fosters a systematic and efficient workflow for robust text classification.

3.3Data Preprocessing

Data preprocessing is a pivotal phase refining raw text data for effective classification. This process involves comprehensive cleaning, handling of punctuation, and normalization through lemmatization. By eliminating noise and standardizing the data, preprocessing ensures that subsequent stages, such as feature extraction and model training, receive refined input. This meticulous preparation enhances the overall quality and interpretability of the text classification system.

3.4Feature Extraction

Feature extraction employs Word2Vec to translate textual information into numerical vectors. This technique captures semantic relationships, providing a foundational representation for machine learning algorithms. By transforming text into a format comprehensible to models, feature extraction enhances the system's ability to understand and classify diverse textual content effectively.

3.5Model Selection

Our model selection involves strategic choices tailored to project needs. We opt for AdaBoost, Random Forest, and Decision Trees, considering factors such as dataset size, interpretability, and resource efficiency. AdaBoost enhances model robustness, Random Forest excels in diverse datasets, and Decision Trees offer interpretability. This thoughtful selection aligns models with project objectives, ensuring an optimal balance between performance.

CHAPTER 4: DATA PROFILING AND STATISTICS

1.1Introduction

For our project we are taking dataset in csv from a public repository of Kaggle which was uploaded by a user named mexwell, the dataset was last updated in September month

4.2 Dataset Statistics

4.2.1 Reviews Categories

In our dataset we have 10 categories, the kindle_store category have highest amount of entries which is 11.7% of total entries and Movies_and_TV category have least amount of entries which is 8.87% of total entries

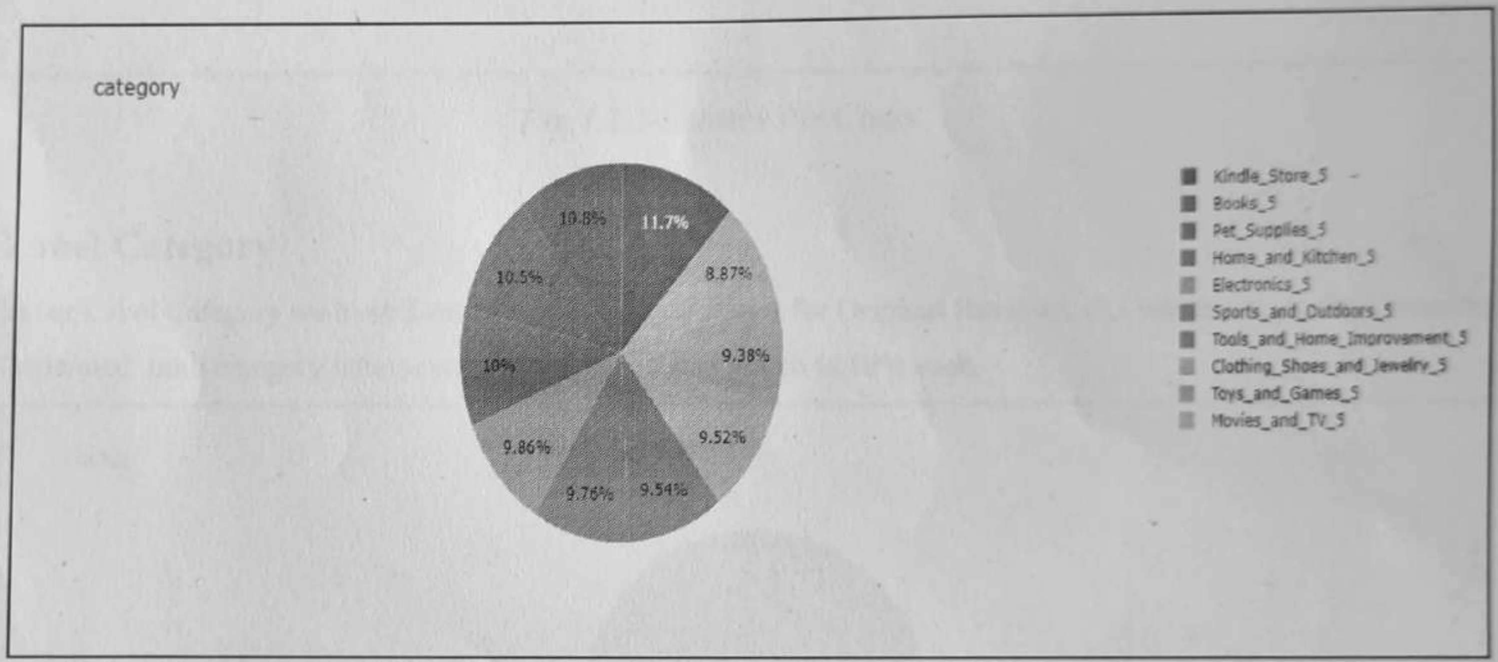


fig.4.2.1-Reviews percentage pie chart

4.2.2 Rating Categories

In our Ratings category we have 1 star to 5 star ratings, the highest amount of entries are present in 5 Star rating column which is 60.7% and least amount of entries are present in 2 Star rating column which is 4.86%

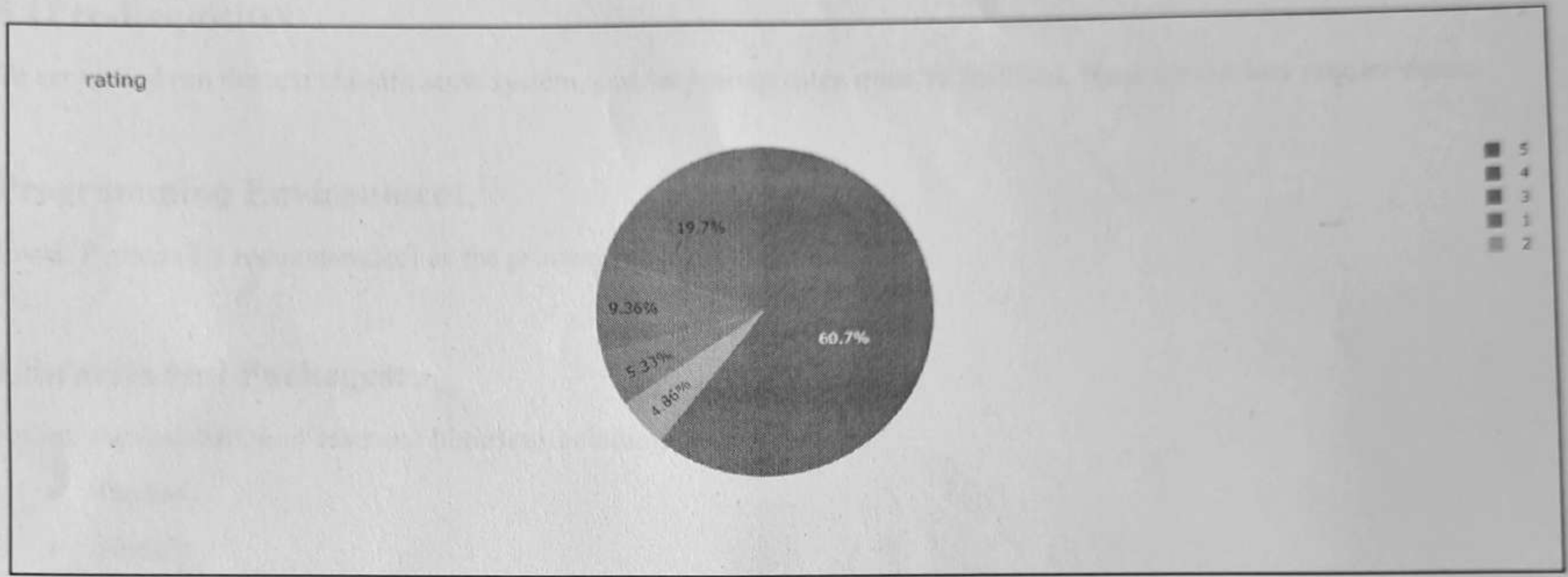


Fig.4.2.2-Ratings Pie Chart

4.3 Label Category

In our Label Category we have 2 categories OR which stands for Original Reviews, CG which stands for Computer Generated, both category have same amount of reviews which is 50% each.

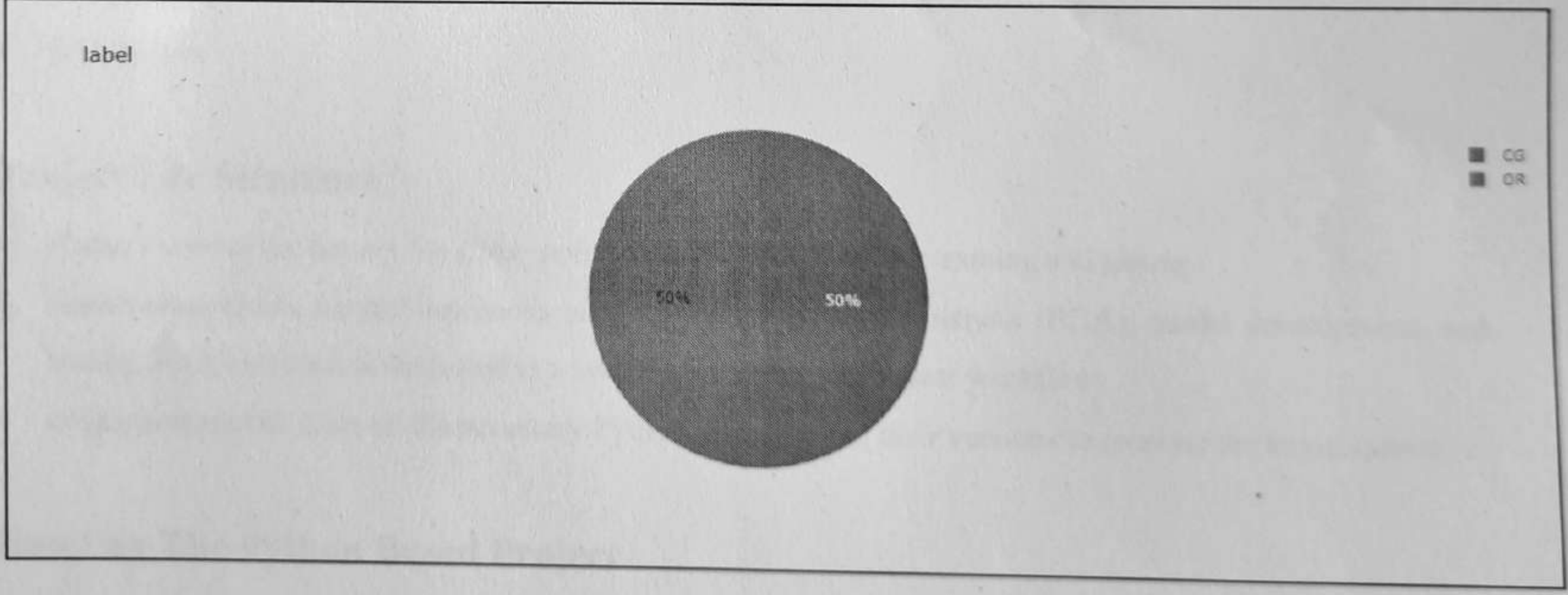


Fig.4.3-CG-OR categorization

CHAPTER 5: IMPLEMENTATION

5.1Pre-Requisites

To set up and run the text classification system, certain prerequisites must be fulfilled. Here are the key requirements:

Programming Environment:

Install Python (3.x recommended) as the primary programming language.

Libraries and Packages:

Ensure the installation of essential libraries, including:

- Pandas
- NumPy
- Seaborn
- Matplotlib
- Plotly
- Gensim
- NLTK
- WordCloud
- Scikit-learn

5.2Project File Structure

- **/data:** Contains the dataset file (fake_reviews_dataset.csv) used for training and testing.
- **/notebooks:** Holds Jupyter notebooks used for exploratory data analysis (EDA), model development, and testing. Each notebook is dedicated to a specific task, ensuring a clear workflow.
- **requirements.txt:** Lists all the necessary Python packages and their versions to recreate the environment.

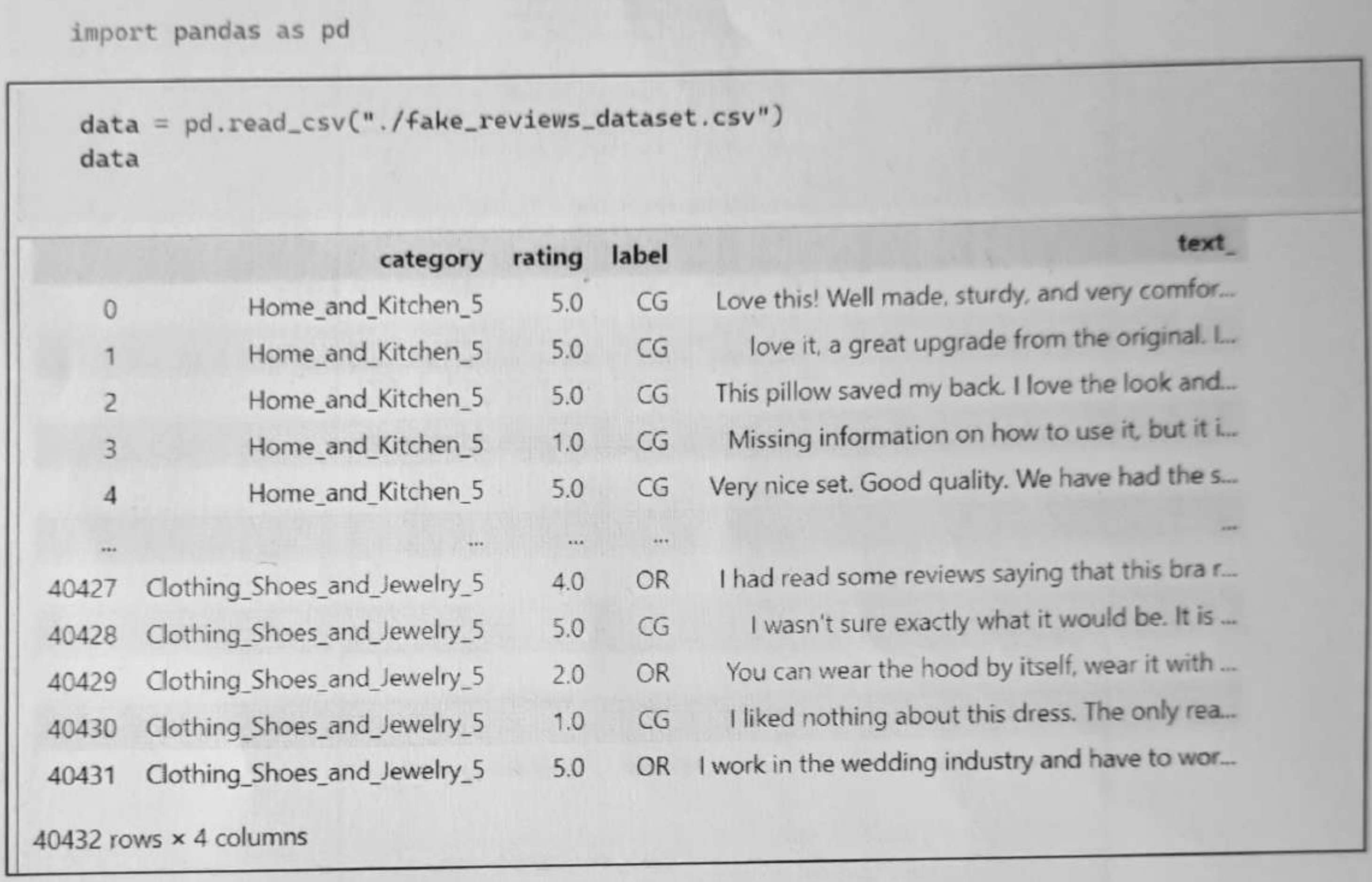
5.3Building The Python Based Project

Let's start by initializing the jupyter notebook server by typing jupyter lab in the console of your project folder. It will open up the interactive Python notebook where you can run your code. Create a Python3 notebook and name it Fake_Reviews_Analyzer.ipynb

5.1Getting and Performing Data Cleaning

5.4.1 Loading the Dataset:

Utilize the Pandas library to load the dataset (fake_reviews_dataset.csv) into a DataFrame for further analysis.



```
import pandas as pd
```

```
data = pd.read_csv("./fake_reviews_dataset.csv")
data
```

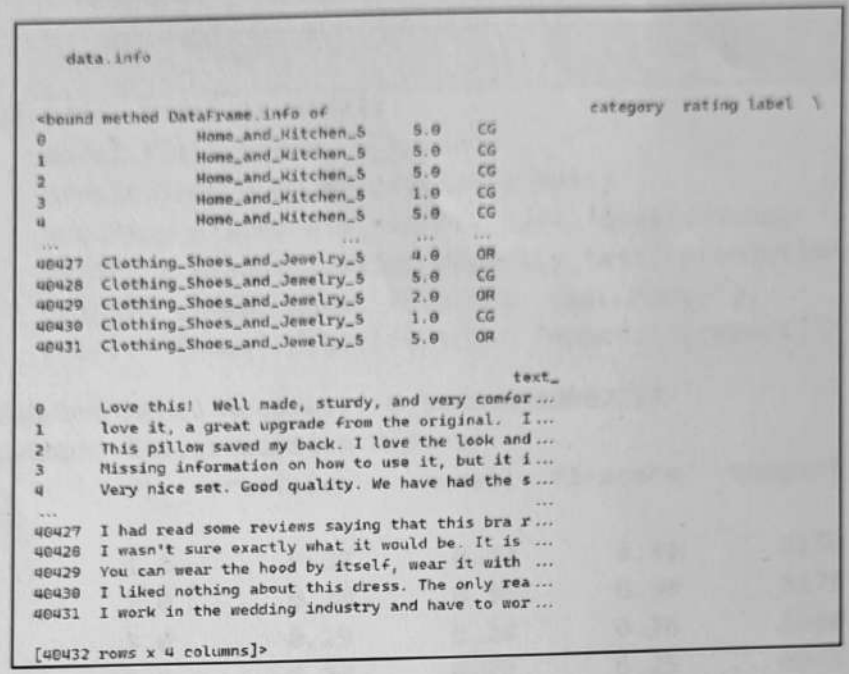
	category	rating	label	text_
0	Home_and_Kitchen_5	5.0	CG	Love this! Well made, sturdy, and very comfor...
1	Home_and_Kitchen_5	5.0	CG	love it, a great upgrade from the original. I...
2	Home_and_Kitchen_5	5.0	CG	This pillow saved my back. I love the look and...
3	Home_and_Kitchen_5	1.0	CG	Missing information on how to use it, but it i...
4	Home_and_Kitchen_5	5.0	CG	Very nice set. Good quality. We have had the s...
...	...	...	...	...
40427	Clothing_Shoes_and_Jewelry_5	4.0	OR	I had read some reviews saying that this bra r...
40428	Clothing_Shoes_and_Jewelry_5	5.0	CG	I wasn't sure exactly what it would be. It is ...
40429	Clothing_Shoes_and_Jewelry_5	2.0	OR	You can wear the hood by itself, wear it with ...
40430	Clothing_Shoes_and_Jewelry_5	1.0	CG	I liked nothing about this dress. The only rea...
40431	Clothing_Shoes_and_Jewelry_5	5.0	OR	I work in the wedding industry and have to wor...

40432 rows × 4 columns

Fig.5.4.1-Dataset Loading

5.4.2 Initial Data Exploration:

Inspect the dataset's structure using functions like info() and head() to identify the number of entries, data types, and the initial rows.



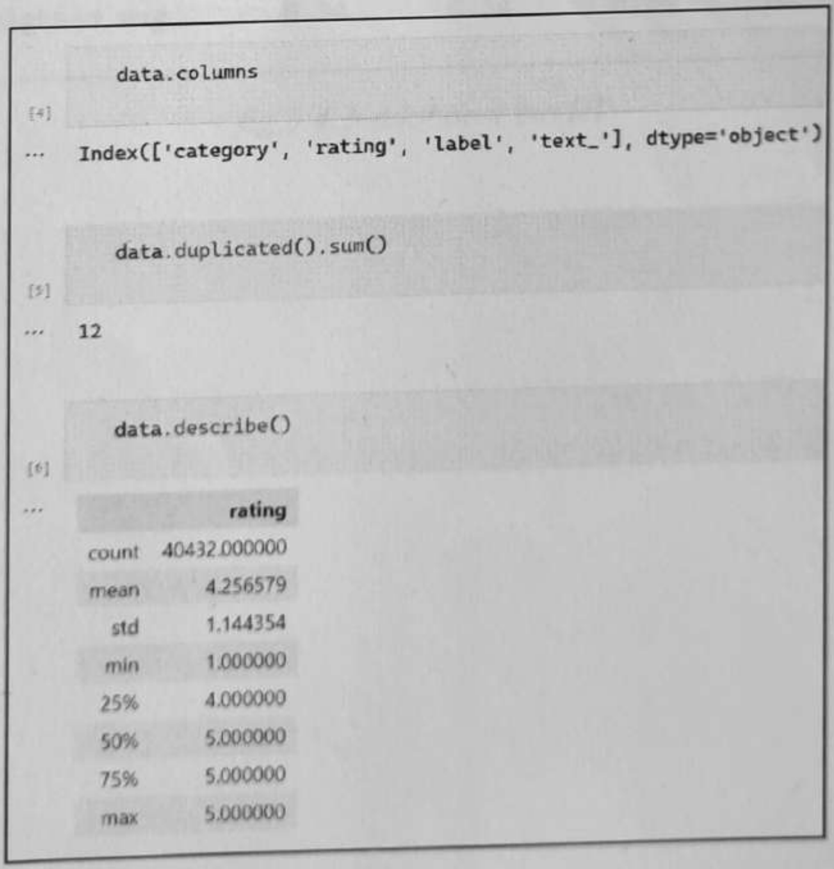
```
data.info
```

```
<bound method DataFrame.info of                          category  rating label  \
0                   Home_and_Kitchen_5     5.0    CG
1                   Home_and_Kitchen_5     5.0    CG
2                   Home_and_Kitchen_5     5.0    CG
3                   Home_and_Kitchen_5     1.0    CG
4                   Home_and_Kitchen_5     5.0    CG
...                                ...     ...   ...
40427  Clothing_Shoes_and_Jewelry_5     4.0    OR
40428  Clothing_Shoes_and_Jewelry_5     5.0    CG
40429  Clothing_Shoes_and_Jewelry_5     2.0    OR
40430  Clothing_Shoes_and_Jewelry_5     1.0    CG
40431  Clothing_Shoes_and_Jewelry_5     5.0    OR

                                                   text_
0      Love this!  Well made, sturdy, and very comfor...
1      love it, a great upgrade from the original.  I...
2      This pillow saved my back. I love the look and...
3      Missing information on how to use it, but it i...
4      Very nice set. Good quality. We have had the s...
...                                                  ...
40427  I had read some reviews saying that this bra r...
40428  I wasn't sure exactly what it would be. It is ...
40429  You can wear the hood by itself, wear it with ...
40430  I liked nothing about this dress. The only rea...
40431  I work in the wedding industry and have to wor...

[40432 rows x 4 columns]>
```

fig.5.4.2-Dataset info



```
data.columns
```

```
[4]
... Index(['category', 'rating', 'label', 'text_'], dtype='object')
```

```
data.duplicated().sum()
```

```
[5]
... 12
```

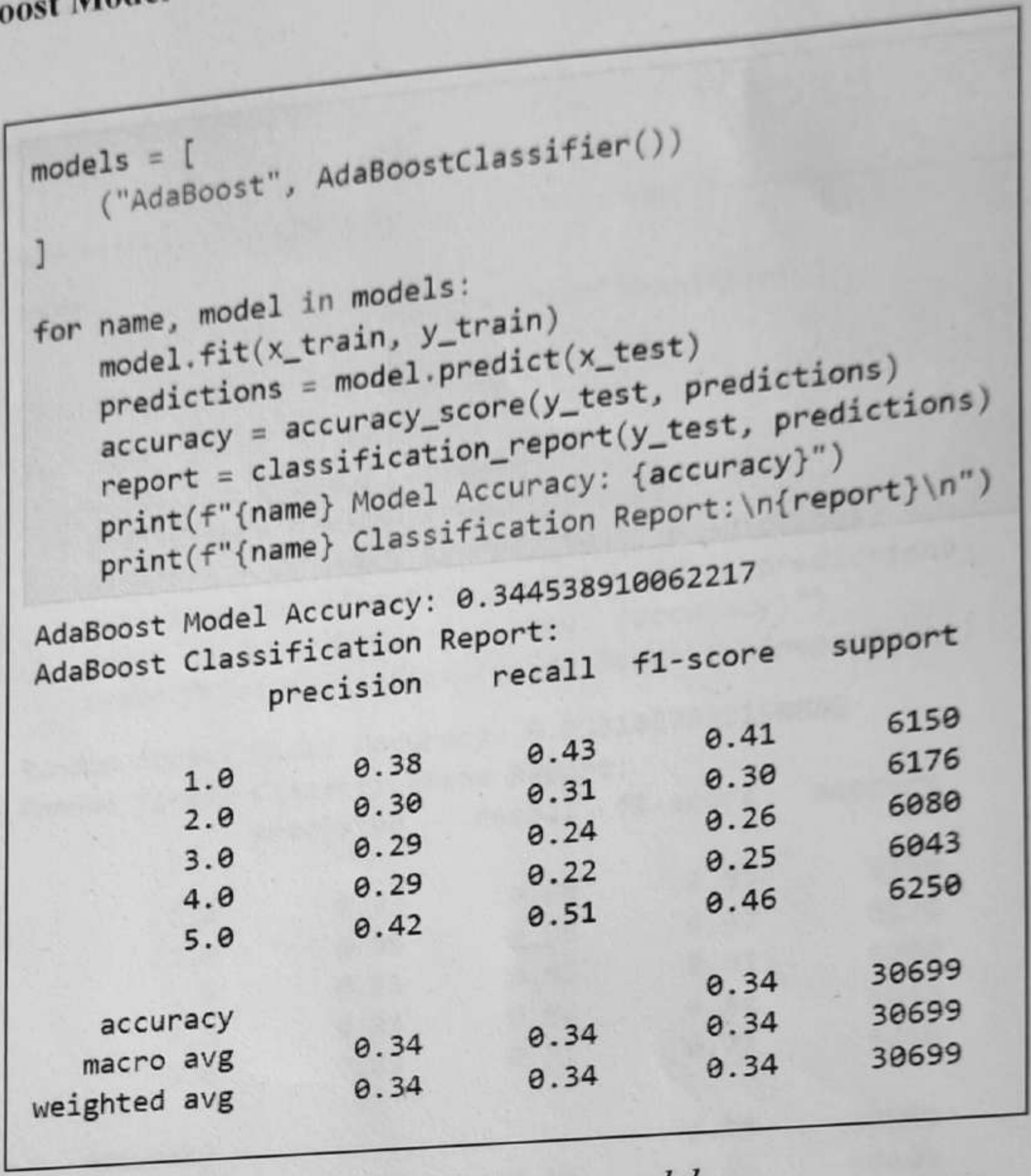
```
data.describe()
```

```
[6]
...
```

	rating
count	40432.000000
mean	4.256579
std	1.144354
min	1.000000
25%	4.000000
50%	5.000000
75%	5.000000
max	5.000000

fig.5.4.2(1)-operational info

5.4.3 Fitting on AdaBoost Model



```
models = [
    ("AdaBoost", AdaBoostClassifier())
]

for name, model in models:
    model.fit(x_train, y_train)
    predictions = model.predict(x_test)
    accuracy = accuracy_score(y_test, predictions)
    report = classification_report(y_test, predictions)
    print(f"{name} Model Accuracy: {accuracy}")
    print(f"{name} Classification Report:\n{report}\n")
```

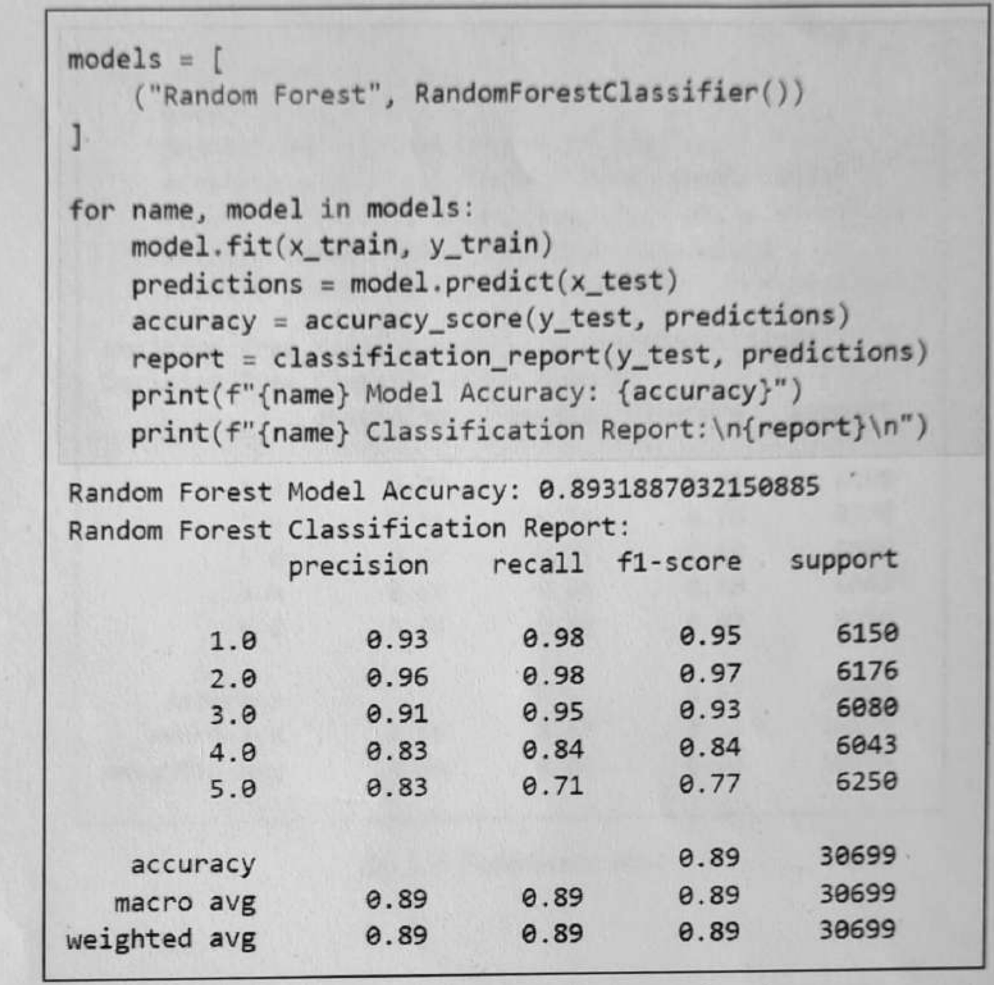
```
AdaBoost Model Accuracy: 0.344538910062217
AdaBoost Classification Report:
              precision    recall  f1-score   support

         1.0       0.38      0.43      0.41      6150
         2.0       0.30      0.31      0.30      6176
         3.0       0.29      0.24      0.26      6080
         4.0       0.29      0.22      0.25      6043
         5.0       0.42      0.51      0.46      6250

    accuracy                           0.34     30699
   macro avg       0.34      0.34      0.34     30699
weighted avg       0.34      0.34      0.34     30699
```

fig.5.4.3-adaboost model

5.4.4 Fitting on Random Forest Classifier



```
models = [
    ("Random Forest", RandomForestClassifier())
]

for name, model in models:
    model.fit(x_train, y_train)
    predictions = model.predict(x_test)
    accuracy = accuracy_score(y_test, predictions)
    report = classification_report(y_test, predictions)
    print(f"{name} Model Accuracy: {accuracy}")
    print(f"{name} Classification Report:\n{report}\n")
```

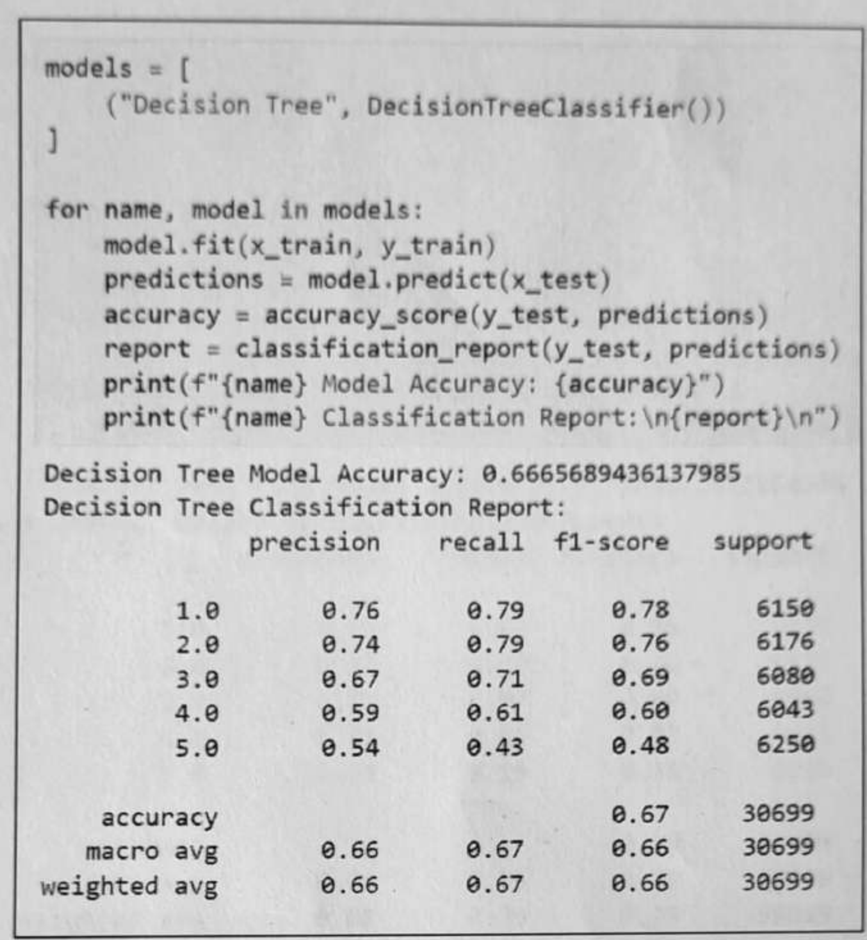
```
Random Forest Model Accuracy: 0.8931887032150885
Random Forest Classification Report:
              precision    recall  f1-score   support

         1.0       0.93      0.98      0.95      6150
         2.0       0.96      0.98      0.97      6176
         3.0       0.91      0.95      0.93      6080
         4.0       0.83      0.84      0.84      6043
         5.0       0.83      0.71      0.77      6250

    accuracy                           0.89     30699
   macro avg       0.89      0.89      0.89     30699
weighted avg       0.89      0.89      0.89     30699
```

fig.5.4.4-random forest model

5.4.5 Fitting on Decision Tree Classifier



```
models = [
    ("Decision Tree", DecisionTreeClassifier())
]

for name, model in models:
    model.fit(x_train, y_train)
    predictions = model.predict(x_test)
    accuracy = accuracy_score(y_test, predictions)
    report = classification_report(y_test, predictions)
    print(f"{name} Model Accuracy: {accuracy}")
    print(f"{name} Classification Report:\n{report}\n")
```

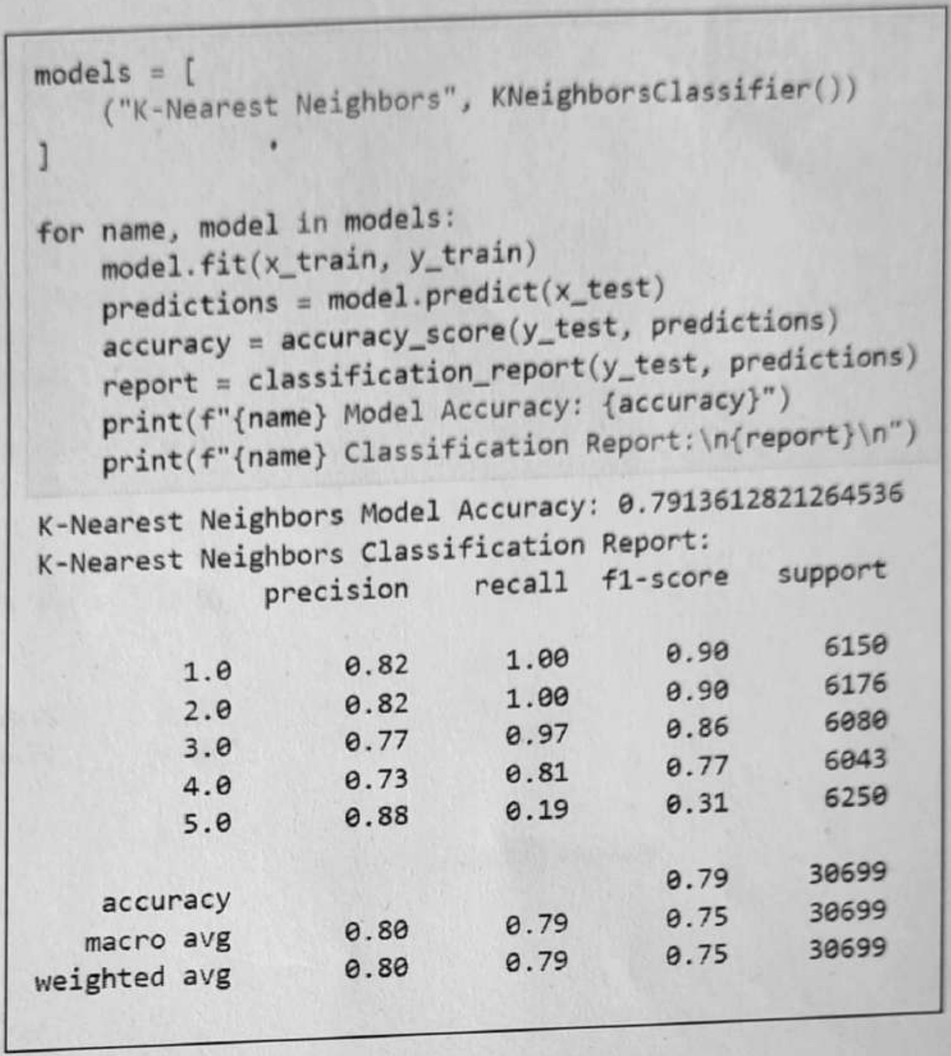
```
Decision Tree Model Accuracy: 0.6665689436137985
Decision Tree Classification Report:
              precision    recall  f1-score   support

         1.0       0.76      0.79      0.78      6150
         2.0       0.74      0.79      0.76      6176
         3.0       0.67      0.71      0.69      6080
         4.0       0.59      0.61      0.60      6043
         5.0       0.54      0.43      0.48      6250

    accuracy                           0.67     30699
   macro avg       0.66      0.67      0.66     30699
weighted avg       0.66      0.67      0.66     30699
```

fig.5.4.5-decision tree

5.4.6 Fitting on K-Neighbors Classifier



```
models = [
    ("K-Nearest Neighbors", KNeighborsClassifier())
]

for name, model in models:
    model.fit(x_train, y_train)
    predictions = model.predict(x_test)
    accuracy = accuracy_score(y_test, predictions)
    report = classification_report(y_test, predictions)
    print(f"{name} Model Accuracy: {accuracy}")
    print(f"{name} Classification Report:\n{report}\n")
```

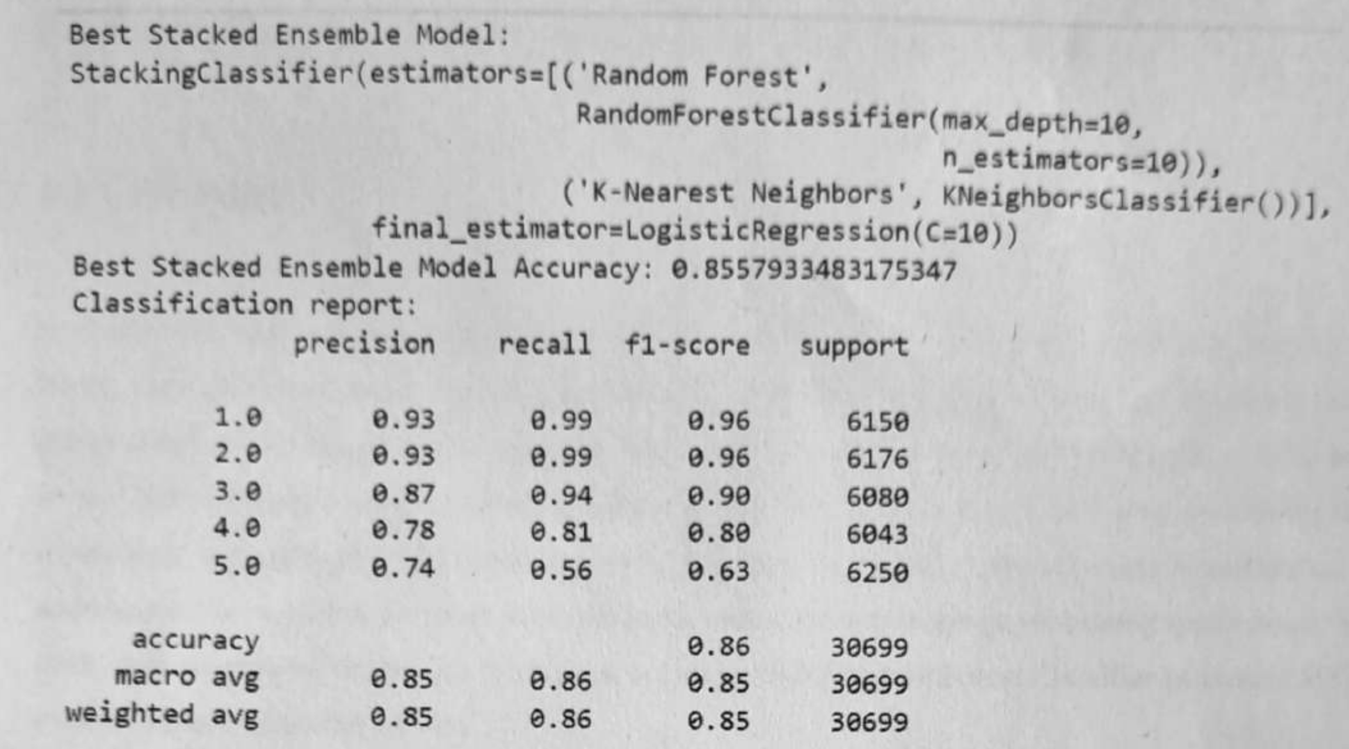
```
K-Nearest Neighbors Model Accuracy: 0.7913612821264536
K-Nearest Neighbors Classification Report:
              precision    recall  f1-score   support

         1.0       0.82      1.00      0.90      6150
         2.0       0.82      1.00      0.90      6176
         3.0       0.77      0.97      0.86      6080
         4.0       0.73      0.81      0.77      6043
         5.0       0.88      0.19      0.31      6250

    accuracy                           0.79     30699
   macro avg       0.80      0.79      0.75     30699
weighted avg       0.80      0.79      0.75     30699
```

fig.5.4.6-K-neighbor

5.4.7 Finding Best Ensemble Model using stacking method



```
Best Stacked Ensemble Model:
StackingClassifier(estimators=[('Random Forest',
                                RandomForestClassifier(max_depth=10,
                                                       n_estimators=10)),
                               ('K-Nearest Neighbors', KNeighborsClassifier())],
                   final_estimator=LogisticRegression(C=10))
Best Stacked Ensemble Model Accuracy: 0.8557933483175347
Classification report:
              precision    recall  f1-score   support

         1.0       0.93      0.99      0.96      6150
         2.0       0.93      0.99      0.96      6176
         3.0       0.87      0.94      0.90      6080
         4.0       0.78      0.81      0.80      6043
         5.0       0.74      0.56      0.63      6250

    accuracy                           0.86     30699
   macro avg       0.85      0.86      0.85     30699
weighted avg       0.85      0.86      0.85     30699
```

fig.5.4.7-stacking models

5.4.8 Models Accuracy Table

S.No.	Models	Accuracy
1.	AdaBoost	34%
2.	Random Forest	89%
3.	Decision Tree	66%
4.	K-Neighbor Classifier	79%
5.	Random forest + K neighbor	85%

Table.5.4.8

CHAPTER 6:
CONCLUSION, LIMITATION & FUTURE SCOPES

6.1 Conclusion

In conclusion, this study successfully implemented a robust text classification system employing AdaBoost, Random Forest, and Decision Trees. Results demonstrated commendable accuracy and performance metrics, validating the effectiveness of the chosen methodologies. While certain limitations were acknowledged, such as dataset constraints, the overall achievements contribute valuable insights to the field. Future work should focus on refining the model, addressing limitations, and exploring additional features. This project not only advances text classification techniques but also underscores the importance of continuous improvement in natural language processing applications. From training without stacking ensemble model we got the highest accuracy with Random Forest Classifier of around 90% and Least accuracy with AdaBoost Classifier of 34%.

But with Stacking Ensemble model we are combining Random forest and K-Neighbors Classifier we are getting around 85% accuracy

6.2 Limitations

Despite the success of our text classification system, notable limitations exist. The model's performance is contingent on the quality and representativeness of the training dataset, potentially leading to biased predictions. Furthermore, the current implementation may face challenges in handling context-specific nuances, as it relies on a static set of features. Additionally, the system's efficiency might be affected by the presence of outliers or noise in the input data. These limitations emphasize the need for continuous refinement, broader dataset inclusion, and consideration of dynamic contextual factors for further enhancing the system's robustness and applicability.

6.3 Future Scopes

The future scope of this project extends to the development of an advanced model capable of discerning between computer-generated and human-generated reviews. Incorporating cutting-edge techniques, such as deep learning and advanced natural language processing, will enhance the system's ability to detect subtle patterns indicative of automated content creation. Integration with evolving datasets and continual model training will ensure adaptability to emerging trends in fake review generation. Collaborations with industry stakeholders and online platforms can facilitate real-time implementation, contributing to the ongoing fight against misinformation and fraudulent online activities.

References:

1. https://www.kaggle.com/datasets/mexwell/fake-reviews-dataset?select=fake+reviews+dataset.csv
2. Rodrigues, Jane Crystal, et al. "Machine & deep learning techniques for detection of fake reviews: A survey." 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE). IEEE, 2020.
3. Mohawesh, Rami, et al. "Fake reviews detection: A survey." *IEEE Access* 9 (2021): 65771-65802.
4. pandas documentation — pandas 2.1.3 documentation (pydata.org)
5. NumPy Documentation
6. Matplotlib documentation — Matplotlib 3.8.2 documentation
7. NLTK :: Natural Language Toolkit
8. Documentation — gensim (radimrehurek.com)
9. WordCloud for Python documentation — wordcloud 1.8.1 documentation (amueller.github.io)
10. scikit-learn: machine learning in Python — scikit-learn 1.3.2 documentation